Dependable Multicore Architectures at Nanoscale: the view from Europe

M. Ottavi, S. Pontarelli, D. Gizopoulos, C. Bolchini, M. K. Michael, L. Anghel, M. Tahoori, A. Paschalis, P. Reviriego, O. Bringmann, V. Izosimov, H. Manhaeve, C. Strydis, S. Hamdioui

I. Introduction

For almost a decade, the shift from increasing core clock frequencies to exploiting parallelism and multicore chip architectures has been the main design drive across all application domains in the electronics and computing industry. The introduction of multicore chips allowed the constant increase in delivered performance otherwise impossible to achieve. Multiple microprocessor cores from different instruction set architectures stay at the epicenter of such chips and are surrounded by memory cores of different technologies, sizes and functionalities, as well as by peripheral controllers, special function cores, analog and mixed-signal cores, reconfigurable cores, etc.

This work is part of a collaboration in the framework of COST ICT Action 1103 "Manufacturable and Dependable Multicore Architectures at Nanoscale".

M Ottavi and S Pontarelli are with University of Rome "Tor Vergata", Italy; email: {pontarelli, ottavi}@ing.uniroma2.it). D Gizopoulos and A Paschalis are with University of Athens, Greece, email: {dgizop, paschali}@di.uoa.gr). C Bolchini is with Politecnico di Milano, Italy, email: cristiana.bolchini@polimi.it. MK Michael is with University of Cyprus, Cyprus, email: mmichael@ucy.ac.cy. L Anghel is with TIMA laboratory, Grenoble, France, email: lorena.anghel@imag.fr. M Tahoori is with Karlsruhe Institute of Technology, Germany, email: tahoori@ira.uka.de. P. Reviriego is with Universidad Antonio de Nebrija, Madrid, Spain, email: previrie@nebrija.es. O Bringmann is with University of Tuebingen, Germany, email: bringman@fzi.de. V Izosimov is with Semcon AB, Sweden, email: Viacheslav.Izosimov@semcon.com. H Manhaeve is with Ridgetop Europe, Belgium, email: hans.manhaeve@ridgetop.eu. C Strydis is with Neurasmus B.V., The Netherlands email: c.strydis@erasmusmc.nl. S Hamdioui is with Delft University of Technology, The Netherlands, email: S.Hamdioui@tudelft.nl

The functionality as well as the complexity of multicore chips is unprecedented. This is the aggregate result of several technologies that emerged and matured together the last few decades: (a) manufacturing process now approaching the 10nm regime and soon expected to go beyond, (b) sophisticated electronic design automation tools assisting and refining every step of the design process, (c) new processor architectures across the entire spectrum of performance and power consumption.

Electronics-based systems (either for computing or other purposes) deliver scalable performance under a domain's power constraints for many decades now. Unfortunately, performance scaling at a given power envelope faces today, more than ever before, several closely related challenges with respect to two major chip development aspects: *manufacturability* and *dependability*. From a temporal point of view manufacturability deals with the cost-effectiveness of chips during production (time < 0 before release to market) while dependability deals with the correctness of their operation in the field (time > 0 after release to market). Manufacturability and dependability share common challenges and threats, have common objectives and utilize common solutions regardless of the employment of chips in systems at the low end of performance and power (low-cost embedded systems or consumer electronics) or the high end of performance (data centers, cloud computing facilities, or extremely powerful supercomputers).

This survey article stems from the view of MEDIAN [1], a large network of researchers from academia and industry funded by the European Science Foundation that collaborate in the areas of manufacturability and dependability of multicore architectures and their deployment in different computing application domains. In particular the focus of this survey is dependability of multicore architectures. Given the short nature of this overview, it is worth to note that we discuss only a partial and limited subset of the works done in this field, focusing on the representative and seminal publications that are closer to the vision of the research network.

The article first describes the main challenges that threaten dependability, then presents and classifies the state-of-the-art in the solutions and methodologies space.

II. Dependability Threats

Dependable operation in the field is a function of the circuit's ability to overcome faults occurring during the circuit mission time. *Dependability* is a very generic term and refers to the confidence a user has on a system's ability to operate correctly. Dependability is quantified by *reliability*, *availability*, etc. which are measures of the ability of the

system to operate under its specifications at a given point of time or during a period of operation and plays a key role in the international roadmaps of advanced computing systems segments [2], [3] [4]. A detailed taxonomy of dependable systems can be found in [4].

A major threat of modern chips comes from intermittent faults induced by environmental conditions, in particular particle strikes on the circuit nodes, causing *soft errors*, [7] Soft errors have traditionally been a major concern for storage elements (registers, memories) but they are already a serious problem for logic nodes as well. Intermittent faults are usually due to environment reasons but can be also the result of voltage and temperature variability when the circuit operates in more than one mode (e.g. when voltage or frequency scaling techniques are employed).

Unlike the past when manufacturing technologies were robust enough to guarantee correct operation virtually "forever", today's circuits suffer from degradation phenomena that significantly reduce their life time even when they operate in nominal conditions. Such mechanisms are typically referred to as *aging* or *wear-out* and get more frequent and more severe as manufacturing nodes get smaller, and circuit frequencies get higher. Aging effects combined to technology node scaling can cause both the occurrence of permanent faults but also the increase of the rate of elusive faults, which manifest themselves as intermittent faults in certain operating conditions such as effects of temperature variation or system load exceeding certain levels [4].

Figure 1 reports a timeline showing the points of time during a chip's lifetime where each threat appears. Manufacturing defects (MD) occur during production (i.e. when t<0), transient errors due to radiation (SE) or to voltage and temperature variability (V) [6] are expected during all the life cycle and aging phenomena (A) at the end of the chip's life.

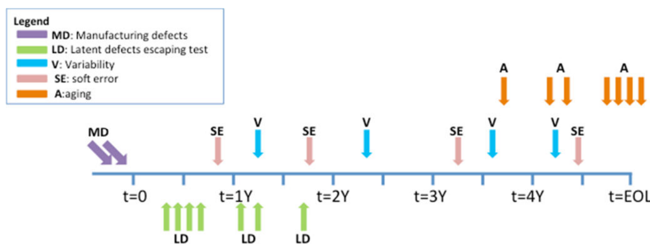

Fig. 1 Occurrence of different kinds of faults during the life of a device

Dependable operation of a circuit throughout its lifetime can be only guaranteed when it employs hardware and software mechanisms to tolerate all possible threats. Since such fault tolerance mechanisms are planned during system design and realized during system implementation and manufacturing, they can improve both quality aspects of the system: *cost-effective production* (through *yield improvement*) and *highly reliable operation* in the field (through *mitigation of threats*) [2], [7].

Conceptually, a chip at any stage may require to employ mechanisms (external or internal as well as hardware or software based) to *detect* and/or *diagnose*, *recover*, and even *repair* itself from *permanent*, *intermittent*, or *transient faults*.

The subsequent sections of the article concentrate on prevalent technological threats (i.e. soft errors and aging effects causing both permanent and intermittent faults) and discuss the main approaches used in practice to improve the in-field dependability of a chip as well as the classic techniques and measures that are employed to evaluate a system's dependability.

III. THE TECHNOLOGY VIEWPOINT ON CURRENT CHALLENGES TO DIGITAL ARCHITECTURES

In this section, we elaborate on some of the issues outlined in the previous section from a technology point of view. In particular, we focus on soft error and aging susceptibility. The proposed analysis and mitigation models are applicable not only to multicore but to digital architectures in general.

A. *Overview of Soft Errors effects*

Soft errors are one of the most important challenges in nanoscale CMOS technologies impacting field-level product reliability. In an electronic component, the failure rate induced by soft errors can be relatively high compared to other reliability issues [8]. Whether or not soft errors impose a reliability risk for electronic systems strongly depends on the application. Soft-error rate is generally not an issue for consumer applications such as mobile phones, but can be a problem for applications that have very severe reliability and safety requirements: automotive, data centers, space industry, medical electronics, etc.

Soft errors are events usually provoked by radiations (i.e. neutrons from cosmic rays, alpha particles emitted by radioactive impurities present in manufacturing used materials, etc.). Aggressive technology scaling, higher clock frequencies and lower voltage operation strategies result in a reduction of capacitance per transistor, and as a consequence particles with lower energy, can generate sufficient charge to cause soft errors. Soft Error Rates (SER) relate to cases when data is corrupted, but the device itself is not permanently damaged. Recent experiments show that the SER of combinational logic in sub-50nm technologies is comparable with that of sequential elements (i.e. latches and flip-flops) and will be a dominant factor in the future technologies [9]. With technology scaling to nanoscale regimes, a single radiation strike can affect multiple cells through secondary particles (neutrons) and shared diffusion which result in multiple bit upsets (MBUs) [10].

Soft errors can have different effects on applications. They may result in data corruption at the system level, or provoke a timing delay or malfunctioning of a circuit or even a system crash. Understanding how a particle impact may lead to failures, has been made possible due to numerous software/hardware methodologies and tools for evaluating SER during the design phase. Such methodologies and tools aim to reduce the extra design and validation cycles during chip fabrication due to late modifications that could be necessary to reach the desired reliability level. Their success on doing that depends on their accuracy in SER evaluation. The full

characterization framework includes Technology CAD models for the radioactive environment, their interaction with the integrated circuit technology as well as design topologies facilitating good estimation of Failures-In-Time (FIT) rates for memory cells and library elements. Moreover, cell level SPICE simulation frameworks and gate level statistical and probabilistic approaches push forward the FIT estimation at sequential cells [9], RTL level and IP blocks [11]. They take into consideration electrical masking, logic masking and timing masking phenomena. Higher-level estimation done at the RTL and the full SOC level allows the integration of architectural vulnerability aspects. All the masking factors have been widely studied in recent years and very sophisticated models have been proposed [12]. During the last years, the accuracy of sensitivity evaluation of soft error effects has been improved as well as the evaluation speed is increased by properly taking into account the full environment of the system, the operating modes, the systematic and random variations as well as the application.

Mitigation approaches for soft errors at the circuit level are aimed at avoiding that the effect of the injected charge could modify logic values and are generally known as hardening techniques. Hardening involves using redundant transistors, capacitors, or resistors to make sure that the effect of the upsets does not propagate. Some typical examples of hardening techniques for memory elements can be found in [13][14][15].

B. *Overview of Aging effects*

Variation effects are among the main reliability issues [16] and can be categorized into two main categories. *Process variation* is the variation of transistor physical parameters due to uncertainties during fabrication process. This type of variation leads to significant performance variation at time $t = 0$. *Aging (temporal variation)*, leads to variation in behavior of a circuit over its lifetime, $t > 0$. Bias Temperature Instability (BTI), Hot Carrier Injection (HCI) and Time-Dependent Dielectric Breakdown (TDDB) are three main degradation phenomena [16], [17]. When the transistor is negatively (positively) biased, traps are generated at the silicon to dielectric interface and also inside the dielectric of PMOS (NMOS) transistors leading to an increase in the threshold voltage of transistor. This phenomenon is called Negative (Positive) BTI and leads to increase of the propagation delays in a circuit and eventually the circuit fails to meet its timing specifications. HCI is an irreversible aging effect, which is caused by the accelerated carriers (electron/holes) under lateral electric fields. HCI also manifests as an increase in the threshold voltage of mostly NMOS transistors. While BTI contains both stress and relaxation (recovery) phases, depending on the transistor gate-source bias, HCI has no recovery mechanism. The NBTI and HCI complex physical models are currently enriched with more contributions: the recoverable part of BTI, the HCI dependence on the workload and the bulk bias, the interaction between them.

TDDB degrades the gate oxide and leads to an increase in its conductance. As a result, the current through the gate insulator increases and eventually it can lead to an abrupt increase of gate leakage current causing a catastrophic failure for the device (hard breakdown) or timing degradation. TDDB becomes more severe as the gate oxide thickness becomes thinner due to technology scaling. TDDB is primarily a major issue for low-k device materials. However, after the introduction of high-k dielectrics in highly scaled logic devices, TDDB is an even more severe issue because of the breakdown of the interfacial layer as well as the high-k layer. Another aging issue which affects interconnect is electromigration [17]. It is caused by physical migration of atoms in a metal wire, when the current flows through the wire for a long time. In order to mitigate the reliability threats, there are two main categories of techniques: 1) "model, predict, and margin", and 2) "sense and adapt". Both styles can be applied at different levels of abstraction.

For aging mitigation at circuit level, guard-banding, body biasing, and gate sizing are the most well-known methods [18]. For effective guard-banding methods, the guard-band has to be accurately predicted. Therefore, it is crucial to effectively predict the circuit failure expectations due to aging; this approach has been extensively explored in the literature [19]. Circuit failure prediction can be combined with optimized self-tuning in order to improve the lifetime by minimizing circuit aging guard-bands [19]. Dynamic voltage/frequency scaling is another approach that can be used at either coarse or fine granularity. There are also methods to mitigate BTI by rebalancing the signal probabilities applied to memory blocks, logic cores and the entire processor [20]. Another approach is input vector control, which can be used at the circuit or the instruction level (using NOP instructions) [16].

In "sense and adapt" strategies against aging effects, the circuit behavior has to be constantly monitored at runtime. For this purpose, signatures can be collected during normal system operation by using special sensors such as the ones proposed in [21] or in [22] for monitoring NBTI and oxide degradation.

Another opportunity to obtain the signatures is by using periodic on-line self-test/diagnostics [19]. Since aging effects are highly dependent of circuit temperature, voltage and power profiles, collecting such signatures are useful in order to predict the aging rates [19]. Adaptive mechanisms can be utilized based on the feedback from performance metric sensors [18]. Various monitoring schemes and canary circuitries such as replica monitors, in-situ monitors, online self-test and software-based inference have been proposed [18].

IV. Dependability Techniques For Multicore Systems

In this section we describe various dependability-enhancing techniques that can be applied to the different components of a multicore system. Table I summarizes the main classes of approaches and their main characteristics. For each technique it is reported: 1) its fault detection (FD) and/or fault tolerant (FT) capabilities with respect to the fault type, 2) the phase during which they are devised (design time vs. run time), and 3) the solution type (software, hardware or architectural).

A. *Faults in processor cores*

To cope with faults in the processors constituting the multi-core system, various approaches have been adopted. Some of

the approaches work at the i) *architectural* level, by enhancing the exploitation of existing cores and/or additional ones, ii) *micro-architectural* level by hardening the core structure, or at iii) *application/software* level, by working at the system and application software levels. Circuit-level techniques are usually not explicitly considered in this scenario since they provide viable solutions (e.g., for the design of a hardened memory element) independently of the circuit being designed and thus we will not include them here.

1) Architectural solutions

One of the most straightforward but costly techniques is based on the use of a second processing core, to monitor the execution of the applications and, if a deviation from the expected logical/timing behavior is detected, restart the core. This can be achieved by means of a dedicated core working in close synchronization with the master one (lock-step), or by means of a simplified hypervisor or watch-dog. The presence of several cores in multicore architectures enables for a direct implementation of this kind of solutions (e.g. [23][24]). A different way to exploit core redundancy is by means of active replication and primary backup. In the former case, the whole process is executed on another processing core in parallel and, in case of fault the result from a healthy replica is used. In the latter approach, only one replica is executed at a time and, when it is detected as corrupted, the "backup" or "shadow" replica is activated and executed. The main difference between the two approaches is that the active replication does not require timing penalty in case of error recovery but require higher redundancy with respect to the primary backup (e.g., [25]).

2) Micro-Architectural solutions: Processor hardening

Several solutions have been envisioned to harden the processor architecture itself, so that any fault is dealt with inside the processor offering a transparent fault tolerant execution of the applications. Addressing of faults at the hardware level leads to increased silicon area, increased critical part of a chip and increased development cost and power consumption (e.g., [26]). A well-known example of this approach is the SPARC-based LEON processor used by the European Space Agency [27].

3) Software techniques

This class of approaches includes solutions that replicate the tasks or threads of the application a number of times (two for detection, three for tolerance), executing on different processing cores and compares the produced outputs. Among the various solutions proposed in literature, the one presented in [28] proposes an execution model based on chip-level redundant threading to deal with soft-errors in processors, achieving fault detection properties without incurring in high overheads and without a significant impact on the architecture. Following the same strategy redundant execution of the tasks or threads, re-execution consists of re-running the portion of the application that has been corrupted, upon detection (e.g., [29]). Overheads are related to maintaining the necessary information to perform a delayed execution of the portion of the application. However, this approach also requires non-trivial effort and may incur high overheads.

B. Faults in memory

In multicore architectures, the amount of silicon area occupied by memories is continuously increasing and also its functionalities (*e.g.* memory coherence) and architectures become more and more complex. As a consequence yield and reliability enhancement of cache memories is becoming more and more important. This is accentuated by the shift from SRAM-based caches to novel memories based on resistive devices that show promising results with respect to the current available charge based memories but present several reliability issues (see [30] for Phase Change Memories and [31] for nanoionics based memories). Here we shortly describe some of the solutions to memory related faults that show how novel dependability techniques can be designed to exploit the specific characteristics of multicore architectures or how already widely used techniques require a deep rethinking to be applied in this new scenario.

1) Cache-coherence solutions

Specific methods related to the multicore paradigm exploit the cache coherence protocols also to add error detection and correction features to the system. The simplest method consists in detecting errors by means of parity bits, correcting the error using a copy of the data stored in another cache. As an example, coherence protocol aimed at dealing with transient failures that affect the interconnection network of a multicore architecture is presented in [32].

2) Transactional-memory approach

Another family of techniques related to the multicore paradigm exploits the transactional memory approach. Transactional memory operations attempt to simplify concurrent programming, by allowing a group of load and store instructions to execute in an atomic way. It can be used to add fault tolerant capacities to the multicore memories. As one of the various examples, [33] uses a hardware transactional memory to detect transient and permanent faults.

3) Information redundancy

Well-known techniques based on redundant row/columns and/or on the use of error correction codes, received renewed attention, since the different requirements in terms of power consumption and speed, and the different type and likelihood of errors in memories, require different types of solutions (e.g., [34]).

C. Communication

Also here we selected only a small subset of possible solutions, focusing on those that exemplify the relationship between the specific solution and the multicore architecture. Different topologies have been proposed to connect together multicore architectures [35]. Some of these topologies are based on shared bus communication infrastructures, often connecting together private memories (*e.g.* L1 private caches are connected to a shared L2 cache). When bus based topologies are used, error codes are the most viable solution. In particular, it is straightforward to use the same ECC used in memory, also for data movement. When complex interconnection structures like Networks-on-Chip (NoC) are employed [36] [37], more error tolerant techniques can be

used, since error can be handled at link level or at network level. Automatic Repeat Request (ARQ) uses error detection codes to detect errors, retrying the failed transmission, while Forward Error Control (FEC) is based on ECC codes. Hybrid ARQ/FEC (HARQ) schemes combine both techniques to obtain a better reliability/performance trade-off.

TABLE I. DEPENDABILITY TECHNIQUES FOR MULTI-CORE ARCHITECTURES

Technique name	Technique family	Target	Fault Type			Phase		Solution Type		
			Transient	Permanent	Intermittent	Design-time	Run-time	SW	HW	Architectural
Dual Lock-Step [24]	Architectural solutions	Processor	FT	FD	FT		X			X
Hypervisor [23]	Architectural solutions	Processor	FT	FD	FT		X			X
Watchdog [24]	Architectural solutions	Processor	FT	FD	FD		X			X
Active replication & backup [25]	Architectural solutions	Processor	FT	FD	FD	X				X
Processor hardening [26][27]	Micro-Architectural solutions	Processor	FT	FD/FT	FD/FT	X			X	
Redundant execution [28]	Software techniques	Processor	FD/FT	FD/FT	FD/FT	X	X	X		
RE-execution and recovery [29]	Software techniques	Processor	FD/FT	FD/FT	FD/FT	X	X	X		
Cache-coherent solutions [32]		Memory	FD/FT	FD/FT	FD/FT		X	X		
Transactional memory approaches [33]		Memory	FD/FT	FD/FT	FD/FT		X	X		
Information redundancy [34]		Memory	FD/FT	FD/FT	FD/FT	X		X	X	
ARQ [35][36]		Communications	FD	FD	FD	X		X		
FEC [35][36]		Communications	FT	FT	FT	X			X	
HARQ [35][36]		Communications	FD/FT	FD/FT	FD/FT	X		X	X	

V. DEPENDABILITY METRICS AND EVALUATION

The design of dependable multicore systems cannot be done without having defined clear dependability metrics and the way to measure them. This section discusses metrics used for evaluating the dependability of multicore systems and their evaluation methodologies.

A. Metrics

A system's dependability can be quantified by various metrics and techniques, typically at different abstraction levels of the system. Such metrics include Failures-in-Time (FIT) rate[1], Mean Time To Failure (MTTF)[2], Mean Time Between Failures (MTBF)[3], bit failure Probability (P_{fail})[4], Architectural and Program Vulnerability Factors (AVF[5], PVF[6]), Silent Data Corruptions (SDC)[7], Detectable Unrecoverable Errors (DUE)[8], Soft Error Rate (SER)[9], etc. ([50][51]) . Estimation techniques can be performed either pre-silicon, comprised of statistical fault injection in simulators or model-based analysis, or post-silicon on the actual hardware prototypes, which can be more accurate but come relatively late in the design flow.

Most reliability estimation techniques consider the circuit- and gate-level. At the circuit-level, reliability device simulators 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 [38], [39]. These simulators have become an integral part of the design process, modeling the variety of physical failure mechanisms (TDDB, BTI, EM, HCI) discussed in Section III. Recently it has become apparent that the impact of process variations must be integrated in the circuit simulation process, along with the physical failure mechanisms [40]. At the gate-level, the entire netlist is considered to estimate the error susceptibility of a node. This requires computing the probability of sensitizing the node with an input vector able to propagate the erroneous value to the circuit's outputs [41], a task that requires the simulation of several random vectors whose number significantly increases with the size of the circuit. To tackle this complexity, chip level reliability prediction methods are mostly statistical following sampling approaches as the one described in [49].

B. Evaluation

Traditionally, dependability metrics have been evaluated analytically using technology and empirically derived foundry and in-field data. For example, the FIT rate of a system is additive on the constituent FIT rate of its components; MTTF can be calculated either from individual component MTTFs or, in a less cumbersome manner, using the inverse of the system FIT rate; MTBF is the addition of MTTF and MTTR (Mean Time to Repair), etc [50][51]. SDC and DUE rates are important metrics used to categorize and quantify the impact

[1] FIT rate is the number of failures per 10^9 hours of operation.

[2] MTTF is the average elapsed time to system failure (gives expected lifetime of single point of failure systems).

[3] MTBF is the average elapsed time between failures of a system.

[4] P_{fail} is the raw technology failure probability of a cell or gate.

[5] AVF is the probability that a fault in a hardware component will lead to an architecturally visible error.

[6] PVF calculates the percentage of architectural-level masking in a program.

[7] SDC is the number or rate of faults in a hardware component that lead to output errors and are not detected by any mechanism.

[8] DUE is the number of rate of faults in a hardware component that are detected by a detection mechanism or lead to an exception but can't be corrected.

[9] SER is the rate at which a device encounters or is predicted to encounter soft errors.

of faults in a system and the effectiveness of its underlying detection/correction mechanism(s). Both rates can be applied separately to each of the various dependability metrics. For the case of soft-errors, it is typical to express the SER as the summation of SDC FIT and DUE FIT [50].

A significant body of recent work at different levels in the area of resiliency involves the study of the impact and susceptibility of transient (soft) errors through fault injections [48]. Transient faults can be injected into a microprocessor in various ways leading to different control capabilities over the time and location of the fault injection, the level of perturbation to the processor, and the simulation time and cost requirements. Commonly used hardware methods are processor pin-level injection, heavy-ion radiation, power or electromagnetic disturbances and non-destructive laser fault injection. All these methods closely imitate real fault situations, but are usually expensive and applicable only after the physical chip is available. Software fault injection is a low-cost alternative that can be applied to designs, programs and O/S and allows observing the final impact on the system. Software methods can be classified into two classes: (i) software-implemented methods, where the processor state or programs are modified during compile- or run-time and the injection takes place on real hardware and (ii) simulation-based method, where the processor, workload, and fault injections are all modeled in a software simulator of the architecture. In general, the latter is more flexible as it provides better controllability of fault injection and observability of the system behavior. However, it requires a very accurate processor microarchitecture and system model developed in software and it runs several orders of magnitude slower than hardware or software-implemented methods.

In the recent years, there has been a considerable effort in estimating the vulnerability of microprocessors considering correlation models or even reliability estimation models that work concurrently at different abstraction levels where the micro-architecture is changed. The most popular measure is the AVF, which is the probability of a bit-flip in a microprocessor structure leading to a user visible system error [42]. AVF gives more realistic SER estimates than circuit- and device-level SERs (for SDCs and DUEs) as it tracks observable errors. As a result, circuit- and device-level SER can be considerably derated. Accurate estimation of AVF is a complex process involving a large number of fault injections and simulations requiring many resources to track values and instructions as they travel through a processor [43]. The process becomes even more demanding when multiple bit-flips, which are expected in future technologies, must be considered.

When analyzing the lifetime reliability of processor-based systems, it is essential to investigate the impact at system level. Srinivasan et al. [44] described a model for lifetime analysis for microprocessors and conducted dynamic reconfigurations based on the model. Other works predict lifetime reliability based on simulations but, as with [44], the failure mechanisms do not consider aging effects leading to inaccuracies in the simulation results. In [45], one of the few works on system-level lifetime reliability analysis for many-core processors, the impact of workloads and associated temperature variations are considered. Recently, researchers have begun to explore the system-level impact of variations on power, performance, and reliability by developing models of process variation.

Scarce work has focused on systematically including the software into the reliability evaluation process. Some work analyzes various compiler optimization effects on the AVF of embedded processors. However, the experiments lack new guidelines regarding software reliability improvement at compiler level. In [46] the authors proposed a first attempt of performing static analysis of a computer system including its software. While representing a new idea to include the software in the error susceptibility estimation, the approach is limited to errors in the instruction op-codes of the program prior to their execution and does not consider the data and control part of the microprocessor. Recent interesting solutions include the software layer by computing the PVF [47] for a set of applications exploited to improve AVF computation for several microprocessors. However, neither the final software workload, nor the full stack is explicitly considered.

TABLE II. POPULAR DEPENDABILITY METRICS

Metric	Level	Phase	Calculation Method
Failures-in-Time (FIT)	circuit/component/microarchitecture	Post-Silicon/Design	Experiment/Simulation
Mean Time To Failure (MTTF)	circuit/component/microarchitecture	Post-Silicon/Design	Experiment/Simulation
Mean Time Between Failures (MTBF)	circuit/component/microarchitecture	Post-Silicon/Design	Experiment/Simulation
Bit failure probability (P_{fail})	circuit/component	Post-Silicon	Experiment
Silent Data Corruptions (SDC)	microarchitecture/architecture	Design	Simulation
Detectable Unrecoverable Errors (DUE)	microarchitecture/architecture	Design	Simulation
Soft Error Rate (SER)	circuit/component/microarchitecture/architecture	Post-Silicon/Design	Experiment/Simulation
Architectural Vulnerability Factor (AVF)	microarchitecture/architecture	Design	Simulation/Analytical
Program Vulnerability Factors (PVF)	architecture	Application (S/W)	Simulation

VI. CONCLUSIONS AND FUTURE CHALLENGES

This article presented a survey of dependability issues faced by multi-core architectures at nanoscale technology nodes. Existing solutions against these challenges were also discussed, describing their scope of application, from technology level methodologies, to design approaches to the metrics required to evaluate the overall dependability of a system.

In the future, the constant reduction of the feature size of the devices will exacerbate the issues related to aging and soft errors. This will create further challenges and at design level, an integrated design approach that will cope with the occurrence of faults at any time of their occurrence i.e. at manufacturing (thus increasing yield) and in the field (thus increasing reliability) will become more and more important to obtain economically viable and dependable systems. Dependability assessment will also need an integrated approach for cross-layer, pre- and post-silicon techniques for “just right” dependability assessment in order to avoid “over-design” for dependability using classic guard-banding methodologies.

ACKNOWLEDGMENT

The MEDIAN COST Action (http://www.median-project.eu) is funded by the European Science Foundation and the European Union.

REFERENCES

[1] M. Ottavi, “Introducing MEDIAN: A new COST Action on manufacturable and dependable multicore architectures at nanoscale,” in Proc. IEEE European Test Symposium, pp. 28-31, 2012.

[2] A. DeHon, N.Carter, H.Quinn, “Final Report for CCC Cross-Layer Reliability Visioning Study”, Computing Community Consortium, http://www.cra.org/ccc/xlayer.php.

[3] Gupta, P.; Agarwal, Y.; Dolecek, L.; Dutt, N.; Gupta, R.K.; Kumar, R.; Mitra, S; Nicolau, A.; Rosing, T.S.; Srivastava, M.B.; Swanson, S.; Sylvester, D, "Underdesigned and Opportunistic Computing in Presence of Hardware Variability," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.32, no.1, pp.8,23, Jan. 2013

[4] A. Avizienis et al. "Basic concepts and taxonomy of dependable and secure computing." IEEE Transactions on Dependable and Secure Computing, vol. 1 no. 1, pp. 11-33, 2004.

[5] M. Duranton, D. Black-Schaffer, K. De Bosschere, J. Maebe, “The HiPEAC Vision for Advanced Computing in Horizon 2020”, http://www.hipeac.net/system/files/hipeac_roadmap1_0.pdf

[6] M. Alam, K. Roy, C. Augustine, “Reliability-and Process-variation aware design of integrated circuits—A broader perspective”, In Proc. of *the IEEE International Reliability Physics Symposium (IRPS),* 2011.

[7] ITRS, http://public.itrs.net, June 2011.

[8] T.Heijmen, “Soft Errors from Space to Ground Historical Overview,” Soft Errors in Modern Electronic Systems, Springer, 2011.

[9] B. Gill, N. Seifert, and V. Zia, “Comparison of alpha-particle and neutron-induced combinational and sequential logic error rates at the 32nm technology node,” In Proc. of the IEEE International Reliability Physics Symposium, 2009, pp. 199 –205.

[10] R. A. Reed, M. A. Carts, P.W. Marshall, C.J. Marshall, O. Musseau, P.J. McNulty, ... & T. Corbiere. “Heavy ion and proton-induced single event multiple upset”. *IEEE Transactions on Nuclear Science, 44*(6), 1997, 2224-2229.

[11] A. Evans, D. Alexandrescu, E. Costenaro, L. Chen, “Hierarchical RTL_Based Combinatorial SER Estimation,” In Proc. of the IEEE International On Line Testing Symposium, IOLTS, 2013, pp 139-144.

[12] R. R. Rao, K. Chopra, D. T. Blaauw, D. M. Sylvester, “Computing the soft error rate of a combinational logic circuit using parameterized descriptors,” Trans. Comp.-Aided Des.Integ. Cir. Sys., vol. 26, no. 3, pp. 468–479, 2007.

[13] M. Zhang, S. Mitra, , T. M. Mak, N. Seifert, , N. J. Wang, , Q. Shi, S. J. Patel, (2006). Sequential element design with built-in soft error resilience. *Transactions on Very Large Scale Integration (VLSI) Systems, IEEE*, *14*(12), 1368-1378.

[14] T. Calin, M. Nicolaidis, and R. Velazco. "Upset hardened memory design for submicron CMOS technology." *IEEE Transactions on Nuclear Science*43.CONF-960773-- (1996).

[15] P. Hazucha, T. Karnik; S. Walstra, B. Bloechel, J. Tschanz, J. Maiz, K. Soumyanath, G.Dermer, S. Narendra, V. De, S. Borkar, "Measurements and analysis of SER tolerant latch in a 90 nm dual-Vt CMOS process," *Proceedings of the Custom Integrated Circuits Conference, 2003. IEEE 2003* , vol., no., pp.617,620, 21-24 Sept. 2003

[16] J. Fang, S. Gupta, S. V. Kumar, S. K Marella, V. Mishra, P. Zhou, S. S. Sapatnekar, “Circuit reliability: from physics to architectures,” In Proc. ACM Int. Conf. Computer-Aided Design, 2012, pp. 243–246.

[17] B. Mesgarzadeh, I. Soderquist, A. Alvandpour, “Reliability challenges in avionics due to silicon aging,” In Proc. of the 15th IEEE International Symposium on,Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2012, pp. 342–347.

[18] J. Henkel, L. Bauer, N. Dutt, P. Gupta, S. Nassif, M. Shafique, M. Tahoori, and N. When, “Reliable on-chip systems in the nano-era: lessons learnt and future trends,” In Proc. ACM Design Automation Conf., 2013, pp.1-10.

[19] S. Mitra, K. Brelsford, Y. M. Kim, H.-H. K. Lee, Y. Li, “Robust system design to overcome CMOS reliability challenges,” IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 1, no. 1, pp. 30–41, 2011.

[20] V. Huard et al., “A Predictive bottom-up hierarchical approach to digital system reliability”, in Proc. IEEE Int. Reliability Physics Symp., pp. 4B.1.1.-4B.1.10, 2012.

[21] A. C. Cabe, Z. Y. Qi, S. N. Wooters, T. N. Blalock, and M. R. Stan, "Small Embeddable NBTI Sensors (SENS) for Tracking On-Chip Performance Decay," in Proc. Int. Symp. Quality Electronic Design, pp. 1-6, 2009.

[22] E. Karl, P. Singh, D. Blaauw, and D. Sylvester, "Compact in situ sensors for monitoring negative-bias-temperature-instability effect and oxide degradation," Solid State Circircuts Conferance, pp. 410– 411, 2008.

[23] S. Campagna, M. Hussain, M.Violante, “Hypervisor-Based Virtual Hardware for Fault Tolerance in COTS Processors Targeting Space Applications”, in Proc. Int. Symp. Defect and Fault Tolerance in VLSI Systems, pp. 44-51, 2010.

[24] D. Gizopoulos, M. Psarakis, S. V. Adve, P. Ramachandran, S. Kumar Sastry Hari, D. Sorin, A. Meixner, A. Biswas, X. Vera, "Architectures for Online Error Detection and Recovery in Multicore Processors", in Proc. of Design, Automation & Test in Europe (DATE), 2011, pp. 533-538.

[25] N. Budhiraja, K. Marzullo, F. B. Schneider, S. Toueg, “The primary-backup approach”, Distributed systems (2nd Ed.), 1993.

[26] A. Pan, O. Khan, S. Kundu, “Improving yield and reliability of chip multiprocessors,” in Proc. Conference on Design, Automation and Test in Europe, pp. 490-495, 2009.

[27] J. Gaisler, “A portable and fault-tolerant microprocessor based on the SPARC v8 architecture”, In Proc. of Int. Conf. Dependable Systems and Networks, pp. 409-415, 2002.

[28] J. C. Smolens, B. T. Gold, B. Falsafi, and J. C. Hoe, "Reunion: Complexity-effective multicore redundancy", IEEE MICRO, pp. 223-234, 2006.

[29] N. Kandasamy, J. P. Hayes, B. T. Murray, “Transparent Recovery from Intermittent Faults in Time-Triggered Distributed Systems”, IEEE Tranactions on Computers, 52(2), 113–125, 2003.

[30] A. Pirovano et al. "Reliability study of phase-change nonvolatile memories." *IEEE Transactions on Device and Materials Reliability, vol.* 4 no. 3 pp. 422-427, 2004.

[31] R. Waser and A. Masakazu "Nanoionics-based resistive switching memories." *Nature materials* vol. 6 no.11 pp. 833-840, 2007.

[32] R. Fernández-Pascual, J. M. Garcia, M. E. Acaci and J. Duato, "A low overhead fault tolerant coherence protocol for CMP architectures," In Proc. of the IEEE Int. Symp. on High Performance Computer Architecture, 2007, pp. 157-168.

[33] G. Yalcin, O. S. Unsal, A. Cristal, M. Valero, "FaulTM-multi: Fault Tolerance for Multithreaded Applications Running on Transactional Memory Hardware," in Proc. Workshop on Wild and Sane Ideas in Speculation and Transactions, 2011.

[34] Su, Chin-Lung, Yi-Ting Yeh, and Cheng-Wen Wu. “An integrated ECC and redundancy repair scheme for memory reliability enhancement,” 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 2005. DFT 2005.

[35] D. Bertozzi, L. Benini, and G. De Micheli, “Error control schemes for on-chip communication links: the energy-reliability tradeoff,” IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 24 no. 6, pp.818-831, 2005.

[36] S. Murali,, N. Vijaykrishnan, M.J. Irwin, L. Benini, and G. De Micheli, “Analysis of error recovery schemes for networks on chips,” IEEE Design & Test of Computers, vol. 22, no. 5, pp.434-442, 2005.

[37] Ejlali, Alireza, et al. "Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks." In Proc. Design, Automation & Test in Europe Conference & Exhibition, 2007.

[38] G. Papasso, D. Rossi, and C. Metra M. Omana, “A model for transient fault propagation in combinatorial logic,” in Proc. IEEE On-Line Testing Symposium, pp. 111-115, 2003.

[39] A. Maheshwari, I. Koren, N. Burleson, “Techniques for transient fault sensitivity analysis and reduction in VLSI circuits,” in Proc. of the IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems, 2003, p. 597-604.

[40] D. Blaauw, and V. Zolotov A. Agarwal, “Statistical timing analysis for intra-die process variations with spatial correlations,” in Proc. Int. Conf. Computer-Aided Design, 2003, pp. 900-907.

[41] N. Touba K. Mohanram, “Partial error masking to reduce soft error failure rate in logic circuits,” in Proc. IEEE Int. Symp. Defect and Fault-Tolerance in VLSI Systems, 2003, p. 433.

[42] S. Mukherjee, C. Weaver, J. Emer, S.K. Reinhardt, and T. Austin, “A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor,” in Proc. Int. Symp. Microarchitecture (MICRO), 2003, pp. 29-40.

[43] N. Wang, A. Mahesri, and S.J. Patel, “Examining ACE Analysis Reliability Estimates Using Fault-Injection,” in Proc. Int.Symp. Computer Architecture (ISCA), 2007, 460-469.

[44] J. Srinivasan, S.V. Adve, P. Bose, and J.A. Rivers, "The Case for Lifetime Reliability-Aware Microprocessors," in Proceedings of International Symposium on Computer Architecture (ISCA), 2004, pp. 276-287.

[45] L. Huang and Q. Xu, “Characterizing the Lifetime Reliability of Manycore Processors with Core-Level Redundancy”, In Proc. of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2010, pp. 680-685.

[46] S. Di Carlo, G. Di Natale, and P. Prinetto A. Benso, “Static analysis of SEU effects on software applications,” in Proc. of the International Test Conference, 2002, pp. 500-508.

[47] V. Sridharan, D. R. Kaeli, “Eliminating microarchitectural dependency from architectural vulnerability,” in Proc. of IEEE 15th International Symposium on. High Performance Computer Architecture, 2009, HPCA 2009.

[48] J.A.. Clark, D.K. Pradhan, “Fault injection: a method for validating computer-system dependability,” IEEE Computers, vol. 28, no. 6, pp. 47-56, June 1995.

[49] R.Leveugle, A.Calvez, P. Maistri, P. Vanhauwaert, “Statistical Fault Injection: Quantified Error and Confidence,” ACM/IEEE Design, Automation, and Test in Europe Conference (DATE), 2009.

[50] S. Mukherjee, “Architecture Desing for Soft Errors,” Morgan Kaufmann, 2008.

[51] D.K. Pradhan, “Fault-Tolerant Computer System Design”, Prentice-Hall, 2003.