Architectural Reliability: Lifetime Reliability Characterization and Management of Many-Core Processors

William Song, Saibal Mukhopadhyay, *Member, IEEE* and Sudhakar Yalamanchili, *Fellow, IEEE*
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332
wjhsong@gatech.edu, {saibal.mukhopadhyay, sudha.yalamanchili}@ece.gatech.edu

Abstract—This paper presents a lifetime reliability characterization of many-core processors based on a full-system simulation of integrated microarchitecture, power, thermal, and reliability models. Under normal operating conditions, our model and analysis reveal that the mean-time-to-failure of cores on the die show normal distribution. From the processor-level perspective, the key insight is that reducing the variance of the distribution can improve lifetime reliability by avoiding early failures. Based on this understanding, we present two variance reduction techniques for proactive reliability management; i) proportional dynamic voltage-frequency scaling (DVFS) and ii) coordinated thread swapping. A major advantage of using variance reduction techniques is that the improvement of system lifetime reliability can be achieved without adding design margins or spare components.

1 Introduction

With continued technology scaling, diminishing lifetime reliability is a problem for future processors. It is anticipated that significant reduction of failure rates will be required to sustain the current level of lifetime reliability [4], [8]. At the circuit-level, additional design margins can be added to enhance reliability but are costly solutions. The margins are determined based on the worst-case operations. In practice, processors rarely operate under the worst-case conditions for significant periods of time. At the system-level, component redundancy can be exploited such that spare components add extra lifetime or reduce per-component load (e.g., rotational scheduling). However, exploiting component redundancy underutilizes available resources and is also an expensive solution. Therefore, architectural approaches such as dynamic reliability management (DRM) [2], [4] have gained favor as a cost-effective approach to enhancing the lifetime reliability.

In this paper, we present a reliability characterization of a many-core processor. In particular, we target server-class multicore processors, where the operational model seeks to maximize processor utilization. In contrast to core-redundancy models [2], [3] where only a fraction of cores are utilized with many idle cores, we seek methods to control the core executions to improve lifetime reliability while fully utilizing resources. Cores experience different levels of stresses depending on the characteristics of parallel workloads, power states, thermal coupling behaviors, etc. Consequently, the processor produces non-uniform degradation across cores, and processor-level reliability and throughput are limited by early failing cores. The key to the reliability characterization is to analyze 1) how the degradation variance across the cores affects processor-level lifetime reliability, and 2) how much non-uniformity in degradation across the cores is created by the execution of workloads. This characterization leads to developing techniques to improve processor lifetime reliability. To address the problem, we use a full-system simulation framework with integrated microarchitecture, power, thermal, and reliability models [6], [11]. Through the microarchitecture-physics co-simulations, the relative impacts of different workloads and executions are evaluated, and the results are projected to estimate the mean-time-to-failure (MTTF) of the processor. The key insight from the experiments is that lifetime reliability can be improved by reducing the variance in core-level MTTF across the processor. This brings out new ways of applying well-known mechanisms for reliability management, and we present two exemplars of such applications for variance reduction; 1) proportional DVFS and 2) coordinated thread swapping.

The remainder of the paper is organized as follows. First, we describe the method for architecture-level reliability modeling via transient simulations of integrated microarchitecture and physical models. Then, we describe how to characterize the lifetime reliability of many-core processors in terms of variance, MTTF, and performability. Experiments are performed with Parsec and Splash-2 benchmarks [1]. Finally, we present two variance reduction techniques, and the results are discussed.

2 Wear Modeling and Lifetime Reliability

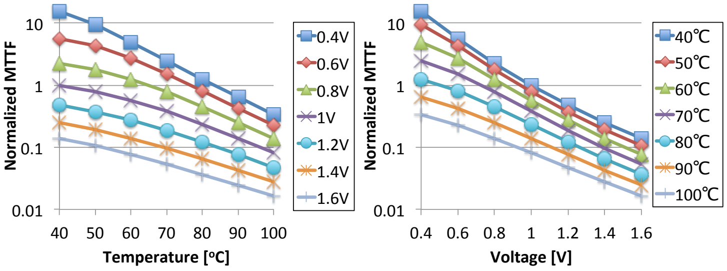
There are several well-known failure mechanisms, and various models have been developed to express degradation and failure processes [8], [12]. Comprised of billions of transistors, processor-level reliability analysis becomes a statistical modeling problem. We use the common exponential models and variables to express the failure rate λ for different failure mechanisms as summarized in Table 1. The processor-level reliability is expressed using these models similar to the works in [3], [8]. The Weibull distribution is known to fit long-term degradation behavior better than the exponential distribution by modeling non-constant failure rates (for the same stress conditions). However, microarchitectural simulations take place over small intervals of time relative to lifetime, where the change of failure rates over time is infinitesimally small. Therefore, constant failure rate approximation can be used, and the modeling can be simplified by using the exponential distribution. Similar comments apply to the lognormal distribution. In addition, lognormal parameters μ and σ do not have clearly known correspondence with physical parameters (e.g., voltage and temperature) for different failure mechanisms, which requires Monte Carlo simulation or statistical fitting [9] that are independent of microarchitecture and workload states. Consequently, in this paper we retain the use of common exponential models for microarchitectural exploration of lifetime reliability with respect to workload and physical states.

The total failure rate with R different failure models can be expressed as Eq. (1) that is a weighted average equation for all $\lambda_{r\in R}$. Since the relative criticality of different failure mechanisms is not known, we assume failure types are equally likely [3], [8] at the baseline condition (defined at T=65^{o}C, V=0.8V), which leads to the sum-of-failure-rates model.

$$\lambda_{\text{TOTAL}} = \sum_{r\in R} p_r \lambda_r, \text{ and} \sum_{r\in R} p_r = 1 \tag{1}$$

TABLE 1. Failure Models and Parameters

Failure Types	Models & Description
Hot Carrier Injection (HCI) [12]	Electrons with sufficient kinetic energy overcome the barrier to gate oxide and cause degradation. $\lambda_{\text{HCI}} = \alpha \times {V_{ds}}^{n} \times e^{-E_a/kT}$ α= tech-dependent, $n = 3, E_a = -0.1$ k= Boltzmann's const., T= temperature
Electromigration (EM) [8]	Directional transport of electrons in interconnect wires causes degradation and failure. $\lambda_{\text{EM}} = \alpha \times J^{n} \times e^{-E_a/kT}$ J= current density, $n = 2, E_a = 0.9$
Negative Bias Temperature Instability (NBTI) [10], [12]	Gradual degradation causes threshold voltage shift and timing errors. $\lambda_{\text{NBTI}} = \alpha \times {V_{gs}}^{n} \times e^{-E_a/kT}$ $n = 5, E_a = 0.4, V_{dd}$= supply voltage
Stress Migration (SM) [8]	Differences in the expansion rates of metals cause mechanical stress. $\lambda_{\text{SM}} = \alpha \times (T_0 - T)^{n} \times e^{-E_a/kT}$ $T_0 = 500, n = 2.5, E_a = 0.9$
Time Dependent Dielectric Breakdown (TDDB) [8]	Wearout of gate oxide leads to short between gate and substrate. $\lambda_{\text{TDDB}} = \alpha \times {V_{gs}}^{c(a+bT)} \times e^{\frac{x+y/T+zT}{kT}}$ $a = 78, b = -0.081, c = 0.1,$ $x = -0.759, y = 66.8, z = 8.37e^{-4}$



(a) (b)

Fig. 1. MTTF varies with respect to (a) temperature and (b) voltage levels (see Eq. (1) and Table 1). The failure models are assumed to meet 5 years of MTTF at the baseline operating conditions (defined at T=65°C, V=0.8V).

In transient modeling where the failure rate λ_r changes over time with temperature and voltage stresses (see Table 1), the total failure rate up to $t = t_n$ with t_i time steps ($i = 1, 2, ..., n$) is expressed as Eq. (2). $\lambda_{\text{TOTAL}(t=t_n)}$ in this equation means the failure rate based on the λ trends up to t_n, and the MTTF due to such trends is calculated as $\text{MTTF} = 1/\lambda_{\text{TOTAL}(t_n)}$.

$$\lambda_{\text{TOTAL}(t=t_n)} = \sum_{i=1}^{n} \left\{ \lambda_{\text{TOTAL}(t=t_i-t_{i-1})} \times \frac{(t_i - t_{i-1})}{t_n} \right\} \qquad (2)$$

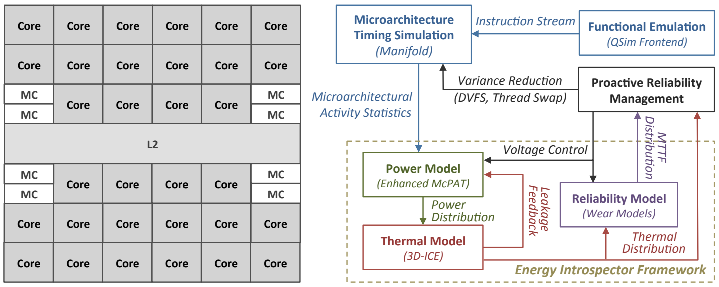
Fig. 1 shows the MTTF variation with respect to different temperature and voltage levels that are major acceleration factors to the degradation (represented as failure rates in Table 1). The MTTF in the figure is normalized to 5 years of baseline MTTF (determined by the baseline conditions at T=65°C, V=0.8V). The graphs show that both temperature and voltage have significant impact on the reliability. Hence, the reliability cannot be simply managed by regulating thermal or voltage history since various patterns of temperature and voltage pair can lead to different reliability results.

3 Experiment Setup

We use a microarchitecture-physics co-simulation framework [6] and configure a 32-core homogeneous processor as shown in Table 2 and Fig. 2-(a). QSim front-end [11] is used for x86 functional emulation of applications on a Linux kernel and to provide the timing simulator with instruction streams. A parallel timing simulator [11] is configured to simulate 32 out-of-order cores and coherent cache hierarchy, connected to a torus network. McPAT [5] and 3D-ICE [7] are used for power and temperature modeling. In particular, McPAT is substantially enhanced to support transient power simulation including dynamic voltage-frequency scaling and leakage-temperature feedback. 3D-ICE is configured to simulate a 2D package. The interactions between multiple physical properties (e.g., voltage, frequency, power, temperature, failure rate, etc.) are coordinated via the physical interaction interface as shown in Fig. 2-(b). This framework handles the data transfers and synchronizations of calculated results and shared data (e.g., temperature, voltage) between multiple models that simulate different physical phenomena [6].

TABLE 2. Microarchitecture-Physics Co-Simulation Setup

Models	Description
Frontend	Qsim (QEMU) functional emulation [11]
Benchmarks	Multi-threaded Parsec & Splash-2 [1]
Cores	32 out-of-order cores (timing model) 128-entry ROB, 6 issue, 80-entry LSQ
Caches	32KB coherent L1 / 32MB shared L2
On-Chip-Network	6x7 torus network
Memory System	8 MCs, 1 channel, 2 ranks, 8 banks, t_{RAS}=30, t_{CAS}=10, t_{RCD}=10, t_{RP}=10
Power Modeling	Enhanced McPAT [5] to support DVFS and leakage feedback, modeled at 16nm
Thermal Modeling	3D-ICE [7] configured to 2D package
Reliability Modeling	Wear models adjusted to meet 5 years of MTTF at the baseline conditions (defined at T=65°C, V=0.8V). Refer to Section 2.
Microarchitecture	Energy Introspector interface [6] that orchestrates the interactions between power, temperature, reliability, and other runtime conditions (e.g, V_{DD}, freq) with microarchitecture simulator [11]

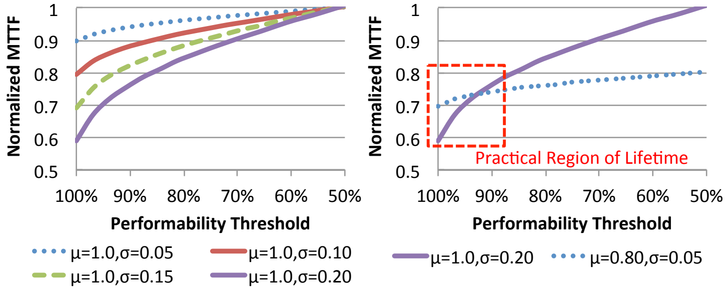

Core Core Core Core Core Core
Core Core Core Core Core Core
MC Core Core Core Core MC
MC MC
L2
MC Core Core Core Core MC
MC MC
Core Core Core Core Core Core
Core Core Core Core Core Core
Microarchitecture Timing Simulation (Manifold)
Instruction Stream
Functional Emulation (QSim Frontend)
Variance Reduction (DVFS, Thread Swap)
Proactive Reliability Management
Microarchitectural Activity Statistics
MTTF Distribution
Voltage Control
Power Model (Enhanced McPAT)
Leakage Feedback
Reliability Model (Wear Models)
Power Distribution
Thermal Model (3D-ICE)
Thermal Distribution
Energy Introspector Framework

(a) (b)

Fig. 2. (a) Simulated 32-core floor-plan and (b) architecture-physics co-simulation framework via the Energy Introspector framework that coordinates the interactions between power, temperature, and reliability models [6].

4 Reliability Characterization: Variance, Performability, and Mean-Time-to-Failure

We first explore how the variance of degradation (represented as failure rate) affects the processor-level reliability. Cores have non-uniform degradation and lifetime, and the processor-level MTTF (that is the inverse of failure rate) depends on how many failed cores it would tolerate. *Performability* is defined



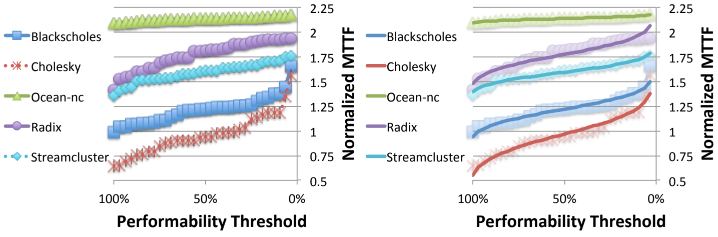
(a) (b)

Fig. 3. MTTF with respect to different performability when the cores have normal distribution with (a) $N(\mu = 1.0,\ \sigma = 0.05{\sim}0.20)$ and (b) $N(\mu = 1.0,\ \sigma = 0.20)$ vs $N(\mu = 0.8,\ \sigma = 0.05)$, normalized to the baseline MTTF.

as the ratio of survived cores over total number of cores. The processor has 100% performability when all cores are available, and the performability decreases as core failures occur. The processor is regarded as operable until it reaches the *performability threshold*. For instance, if the performability threshold is 90%, the processor continues to operate until 10% of cores fail. Therefore, the processor-level MTTF is determined by the performability threshold. We assume that the MTTF of cores are normally distributed on the die with μ and σ. Fig. 3-(a) shows the change of MTTF with μ = the baseline MTTF (fixed) and σ is varying from 5% to 20% of the baseline MTTF. With the same mean, the processor-level MTTF decreases with larger variance, which represents the case that some cores are more stressed and likely to fail earlier. Fig. 3-(b) shows the curves of two different cases, i) high MTTF and high variance and ii) low MTTF and low variance. The first case represents the situation that the processor has low average degradation (or high MTTF) with large non-uniformity, whereas the second case is that the processor has more degradation on average but with relatively even degradation across the cores. Higher performability is the practical region to define the processor lifetime, and it is unlikely that the processor is used with large number of failed components. Although the first case has higher average, the processor has shorter lifetime due to earlier failures caused by large variance. It implies that initial failures are critical to processor lifetime. The key to the reliability management is to regulate the variance, and we call this *variance reduction*. On the other hand, controlling the mean is challenging that it may require throttling the overall processor execution to increase the average, which has an adverse impact on the performance.

5 Experiment and Analysis

We examine how much degradation variance is created by executing different applications at the normal operating conditions (defined at 2.0GHz, 0.8V), using Parsec and Splash-2 benchmarks [1]. Each benchmark is executed in multi-threaded mode utilizing all 32 cores, but it may use only a fraction of cores depending on execution phases. Initial serialized portion of the benchmark is fast-forwarded, and then transient microarchitecture-physics co-simulation is performed (refer to Section 3). Workloads have variable phases of power and heat dissipation, and thus the MTTF (i.e., the projected lifetime based on the cumulative operation history) also fluctuates along the phases. We find that the steady-state analysis such as using average power numbers [8] has noticeable difference to the results of transient analysis. Steady-state analysis produces larger MTTF than transient analysis since it underestimates high power and temperature phases that suffer from accelerated degradation and thus have greater impacts on the reliability. Note that our analysis does not include the models that have much longer duty cycles than the length of microarchitecture simulation, such as thermal cycling [8].



(a) (b)

Fig. 4. Reliability characterization of Parsec and Splash-2 benchmarks: (a) resulting MTTF of cores from the transient simulations and (b) MTTF curves based on normal distribution with sample MTTF mean ($\bar{\mu}$) and standard deviation (σ_N) from (a) - (a) is copied and layered underneath (b).

TABLE 3. Reliability Characterization: Normal Distribution Models of Parsec and Splash-2 Benchmarks

Parsec	$N(\mu,\sigma)$	Splash-2	$N(\mu,\sigma)$
Blackscholes	$N(1.214, 0.135)$	Cholesky	$N(0.959, 0.199)$
Canneal	$N(2.187, 0.016)$	FFT	$N(1.976, 0.041)$
Fluidanimate	$N(1.240, 0.103)$	LU-nc	$N(2.192, 0.015)$
Streamcluster	$N(1.590, 0.096)$	Ocean-nc	$N(2.135, 0.021)$
Swaptions	$N(1.897, 0.050)$	Radiosity	$N(1.107, 0.151)$
Vips	$N(2.012, 0.045)$	Radix	$N(1.768, 0.141)$

Fig. 4-(a) shows the MTTF distribution of 32 cores (sorted in ascending order along the performability threshold) at the end of execution for 5 different benchmarks; load-sharing problem (i.e., increasing stresses per core as the failures occur) is not considered in the experiment. Parallel execution of each benchmark stresses the cores at different levels, resulting in different mean and variance. In the graph, higher average means better MTTF, and steeper slope indicates larger variance. Sample mean ($\bar{\mu}_N$) and standard deviation (σ_N) are calculated from the core MTTF distribution of each benchmark simulation, and the normal distribution curve based on $N(\bar{\mu}_N,\ \sigma_N)$ is plotted in Fig. 4-(b); (a) graph is layered underneath for comparison. This reveals that the reliability characteristics of parallel executions in the many-core processor show normal distribution under the normal operating conditions (i.e., uncontrolled execution). Table 3 lists the reliability characteristics of other benchmarks based on the normal distribution model. The results show that our reliability analysis in Section 4 holds using the normal distribution model with the terms of variance, performability, and MTTF.

6 Proactive Reliability Management

Based on the reliability characterization with the normal distribution model in the many-core processor, we present two variance reduction techniques for reliability management; 1) proportional DVFS and 2) coordinated thread swapping. The contribution of this work is proactive reliability management via the notion of variance reduction, and thread swapping and DVFS themselves are widely used techniques (but vary in details) for processor execution controls including power, thermal, or reliability management.

TABLE 4. DVFS Levels for Variance Reduction

(Volt., Freq.)	Δ MTTF at 65^oC	(Volt., Freq.)	ΔMTTF at 65^oC
(0.740V, 1.700GHz)	+25%	(0.828V, 2.140GHz)	-10%
(0.750V, 1.750GHz)	+20%	(0.843V, 2.215GHz)	-15%
(0.762V, 1.810GHz)	+15%	(0.860V, 2.300GHz)	-20%
(0.774V, 1.870GHz)	+10%	(0.876V, 2.380GHz)	-25%

TABLE 5. Variance Reduction by DVFS, Compared with Normal Executions (see Table 3)

Benchmarks	$(\Delta\mu, \Delta\sigma)$	Processor MTTF	Exec. Time	Energy
Blackscholes	(+0.003, -0.090)	+18.00%	-1.40%	-1.63%
Cholesky	(+0.018, -0.154)	+25.72%	even	-2.07%
Fluidanimate	(-0.032, -0.058)	+8.53%	-0.85%	+0.54%
Radiosity	(+0.022, -0.098)	+21.34%	+3.31%	+2.33%
Radix	(+0.040, -0.113)	+35.67%	-0.60%	-0.56%
Streamcluster	(-0.045, -0.051)	+13.22%	+2.11%	+3.93%

6.1 DVFS for Variance Reduction

The DVFS technique exploits the impact of voltage on reliability to adjust the degradation levels of cores such that the variance of core MTTF distribution is reduced. The difficulty is in determining how much of a voltage shift is required to adjust the core MTTF. To simplify the problem, we pre-calculated the necessary voltage-frequency levels at 65°C to adjust the MTTF toward the mean considering voltage stress. The pre-calculated DVFS levels in Table 4 are applied to each core depending on the difference between its expected MTTF and the sample mean. Table 5 shows the results of using DVFS in this fashion as a variance reduction technique for reliability management. $(\Delta\mu, \Delta\sigma)$ means the relative change of mean and standard deviation of the core MTTF distribution, compared with the normal executions in Table 3. The DVFS effectively reduces the variance and improves processor MTTF at 100% performability (i.e., time to the first failure). Note that DVFS is not applied to shift the mean, and there is less than 4.5% change of μ, while σ is reduced up to 15.4%. The Streamcluster case highlights the reliability management by variance reduction. Applying DVFS, μ of the core MTTF distribution decreases by 4.5%, but instead σ is reduced by 5.1%, leading to 13.2% improvement of processor MTTF at 100% performability. In this case, variance reduction shows greater impact on improving processor MTTF than the shift of the mean. In general, we note that when DVFS is used for regulating power or temperature, it has an immediate impact on performance. However, when DVFS is applied to regulating degradation, transitions in execution phases do not immediately incur regulations due to cumulative characteristic of degradation, leading to small performance changes.

6.2 Coordinated Thread Swapping for Variance Reduction

Recall that the MTTF of cores have a normal distribution that is symmetric around the mean. It would seem that swapping threads between the cores at the opposite ends of the MTTF curve would reduce the variance. We refer to a core that has relatively low MTTF as a *weak core*, and the opposite one as a *strong core*. However, due to cumulative characteristic of degradation, simple thread swapping between weak and strong cores is not effective since immediate power/thermal stresses are not taken into consideration. For instance, if a thread on the strongest core transitions to high-power phase, moving this thread to the weakest core further weakens the core and increases the MTTF variance. Therefore, we study *coordinated thread swapping*; 1) swap threads between the weakest core and the core with a thread that has the lowest dynamic power (*coolest thread*) and 2) swap threads between the strongest core and the core of the highest dynamic power (*hottest thread*). The coordinated thread swapping is applied to the benchmarks that have native variance larger than the threshold ($\sigma \gg 0.05$) in Table 3. Table 6 shows the results of the coordinated thread swapping as a variance reduction technique for reliability management. It effectively reduces the variance and improves processor MTTF at 100% performability, where the improvement also derives from the shift of the mean ($\Delta\mu$). For example, thread swapping for Radix increases μ by 7.3% and reduces σ by 8.9%, resulting in 31.9% increase in processor MTTF, while there are small changes to total execution time and energy consumption. However, coordinated thread swapping for Blackscholes decreases the reliability. Because the application is embarrassingly parallel and all threads have similar dynamic power dissipation, there is little benefit of swapping threads in this case. Instead, it increases performance and hence power and heat dissipation, leading to decreased reliability.

TABLE 6. Variance Reduction by Coordinated Thread Swapping, Compared with Normal Executions (see Table 3)

Benchmarks	$(\Delta\mu, \Delta\sigma)$	Processor MTTF	Exec. Time	Energy
Blackscholes	(-0.233, +0.014)	-23.30%	-14.48%	-2.36%
Cholesky	(+0.213, -0.129)	+40.45%	-0.86%	-10.59%
Fluidanimate	(+0.013, -0.005)	+2.47%	-4.43%	-2.28%
Radiosity	(+0.034, -0.026)	+10.43%	+0.93%	-0.99%
Radix	(+0.073, -0.089)	+31.88%	+0.37%	+0.08%
Streamcluster	(-0.023, -0.012)	+2.43%	+4.22%	+4.88%

7 Conclusion

In processor-level reliability management, we observe that MTTF variance reduction across cores is the key to improve the lifetime reliability at higher performability threshold (i.e., time to a few number of failures). An advantage of this approach is that the principle of variance reduction can in general be applied to other forms of degradation distributions that in turn may require different methods to reduce variance.

References

[1] C. Bienia et al., "Parsec vs Splash-2: Quantitative Comparison of Two Multithreaded Benchmark Suites on Processors," *ISWC*, 2008.

[2] S. Feng et al., "Maestro: Orchestrating Lifetime Reliability in Chip Multiprocessors," *HiPEAC*, Jan. 2010.

[3] L. Huang et al., "Characterizing The Lifetime Reliability of Manycore Processors with Core-level Redundancy," *ICCAD*, 2010.

[4] H. Kim et al., "Use It Or Lose It: Wear-out and Lifetime in Future Chip Multiprocessors," *MICRO*, Dec. 2013.

[5] S. Li et al., "McPAT: Integrated Power, Area, Timing Modeling Framework for Multicore Architectures," *MICRO*, Dec. 2009.

[6] W. Song et al., "Energy Introspector: A Parallel, Composable Framework for Integrated Power-Reliability-Thermal Modeling for Multicore Architectures," *ISPASS*, Mar. 2014.

[7] A. Sridhar et al., "3D-ICE: Fast Compact Transient Thermal Modeling for 3D ICs with Inter-Tier Liquid Cooling," *ICCAD*, Nov. 2010.

[8] J. Srinivasan et al., "Lifetime Reliability: Toward An Architectural Solution," *Micro*, May 2005.

[9] J. Srinivasan et al., "Exploiting Structural Duplication for Lifetime Reliability Enhancement," *ISCA*, June 2005.

[10] W. Wang et al., "The Impact of NBTI Effect on Combinational Circuit: Modeling, Simulation, and Analysis," *TVLSI*, May 2009.

[11] J. Wang et al., "Manifold: A Parallel Simulation Framework for Multicore Systems," *ISPASS*, Mar. 2014.

[12] M. White et al., "Microelectronics Reliability: Physics-of-Failure Based Modeling and Lifetime Evaluation," *NASA JPL*, 2008.