LIFETIME RELIABILITY: TOWARD AN ARCHITECTURAL SOLUTION

AS SCALING THREATENS TO ERODE RELIABILITY STANDARDS, LIFETIME RELIABILITY MUST BECOME A FIRST-CLASS DESIGN CONSTRAINT. MICROARCHITECTURAL INTERVENTION OFFERS A NOVEL WAY TO MANAGE LIFETIME RELIABILITY WITHOUT SIGNIFICANTLY SACRIFICING COST AND PERFORMANCE.

Jayanth Srinivasan
University of Illinois, Urbana-Champaign

IBM T.J. Watson Research Center

Sarita V. Adve
University of Illinois, Urbana-Champaign

Pradip Bose
Jude A. Rivers
IBM T.J. Watson Research Center

Developing and maintaining industrywide standards for lifetime reliability is a critical task for all microprocessor manufacturers. Although technology scaling continues to provide significant performance benefits, increasingly smaller feature sizes and increasing power densities are accelerating the onset of wearout-based failures, thus shortening processor life.[1]

Microarchitects have traditionally treated processor lifetime reliability as a manufacturing problem, best left to device and process engineers. In current processors, manufacturers enforce lifetime reliability, or *qualify* it, during device design, circuit layout, manufacture, and chip test. This reliability qualification, which is application-oblivious, is based on estimates of worst case temperature and processor utilization. However, most applications will run at lower temperature and utilization, resulting in higher reliability and longer processor lifetimes than required. As a result, current reliability qualification methodologies are overly conservative, unnecessarily increasing cost or decreasing performance. Sustaining this approach will likely be infeasible in future scaled systems.

We believe that designing *lifetime-reliability-aware microarchitectures* can address this problem. Manufacturers can use the microarchitecture's unique knowledge of application runtime behavior to qualify reliability on the basis of target application behavior, offering an opportunity to decrease cost and increase performance. Viewing reliability as a design constraint of the microarchitecture is a departure from typical practice, and requires new tools and much novel research. To support this view, we at UIUC and IBM jointly evolved models and tools as a first step toward a lifetime-reliability-aware microarchitecture.[2] On the basis of extensive discussions with front-end, backend, and reliability engineers at IBM, we determined five critical wearout-based failure mechanisms and identified state-of-the-art device-level analytic models that could serve as input for an architecture-level model.[3-5] We then developed Ramp (short for Reliability-Aware Microprocessor), the first architecture-level model that allows lifetime-reliability-aware analysis of applications and architectures. Ramp uses the wearout models to calculate an instantaneous mean time to failure (MTTF)—the expected processor life—that is based on current processor temperature and utilization. Much like previous microarchitectural power and temperature models,[6,7] Ramp divides the processor into a few discrete structures—arithmetic logic units (ALUs), floating-point units

Published by the IEEE Computer Society

(FPUs), register files, branch predictor, caches, load-store queue, reorder buffer—and applies the analytic models to each structure as an aggregate. It then combines the instantaneous structure-level MTTFs to give the final MTTF for the entire processor.

Using RAMP, we have so far explored two microarchitectural techniques that can trade off processor lifetime reliability, cost, and performance. One set of techniques lets designers apply selective redundancy at the structure level and/or exploit existing microarchitectural redundancy.[8] Another technique, which we describe in detail in this article, is *dynamic reliability management (DRM).* In DRM, the processor uses runtime adaptation to respond to changing application behavior to maintain its lifetime reliability target.[2] In contrast to methods in which reliability qualification is based on worst case behavior, DRM lets manufacturers qualify reliability at lower (but more likely) operating points than the worst case. This lowers the cost of reliable design. As in dynamic thermal management,[9] if applications exceed the reliability design limit, the processor can adapt by throttling performance to maintain the system reliability target. Conversely, for applications that do not stress the reliability limit, DRM uses adaptation to increase performance while maintaining the target reliability.

Reliability challenges

Emerging technology trends may pose significant challenges to lifetime reliability unless counteracted by architectural and design innovations. The most important issues are scaling and increased power densities. Additionally, increasing transistor count, and on-chip power management techniques potentially exacerbate the problem.

Scaling and increased power density

Supply voltages and threshold voltages are not scaling appropriately with technology because of performance and leakage power concerns. This nonideal scaling is increasing processor power densities and temperatures, exponentially accelerating wearout failures. In addition, scaling decreases lifetime reliability by shrinking the thickness of gate and interlayer dielectrics and increasing interconnect current density. Finally, scaled-down transistors in deep submicron CMOS technologies also have significantly higher power leakage, which has an exponential dependence on temperature and thus leads to even higher processor temperatures.

To understand how scaling is likely to affect future processors, we quantified its impact on processor temperature and reliability using Ramp.[10] Figures 1a and 1b show our results (the same as previous results[10] but with an additional failure mechanism). Figure 1a shows peak processor temperature for eight SpecInt applications running on a contemporary superscalar processor across technology generations from 180 nm to 65 nm. Figure 1b shows the decrease in reliability for the same applications and processor with scaling. The Max curve is the MTTF calculated on the basis of worst case current density and temperature seen over all the applications. Our results assume the same microarchitecture over technology generations with no additional transistors and no changes during manufacturing or device design for reliability and temperature.

As Figure 1a shows, peak processor temperature rises significantly with scaling. On average from 180 nm to 65 nm, the hottest structure's temperature increased 14 degrees Kelvin. This rise is due to increasing power densities on chip, which in turn is due to nonideal scaling.

Figure 1b shows that simple scaling of the type mentioned above would result in a dramatic reduction of reliability. If we choose the model constants (see Figure 2 and associated descriptive text) such that the average MTTF across the eight SpecInt applications is about 30 years at 180nm, then at 65nm we would see that average value drop by 76 percent. (The 30 year value is an arbitrary assumption). Figure 1b also shows that estimates based on worst case behavior are much more severe than the actual applications (the Max curve versus any application curve). These results clearly demonstrate the impact of technology scaling on lifetime reliability. Although much work is ongoing to mitigate these scaling concerns with circuit, device, and manufacturing techniques, continued reliance on an application-oblivious methodology based on worst case-behavior will become difficult to sustain.

Increasing transistor count

As functionality increases, so does transistor count. The more transistors, the more failures are likely, which in turn means a shorter processor life. Hence, the problem is twofold: indi-

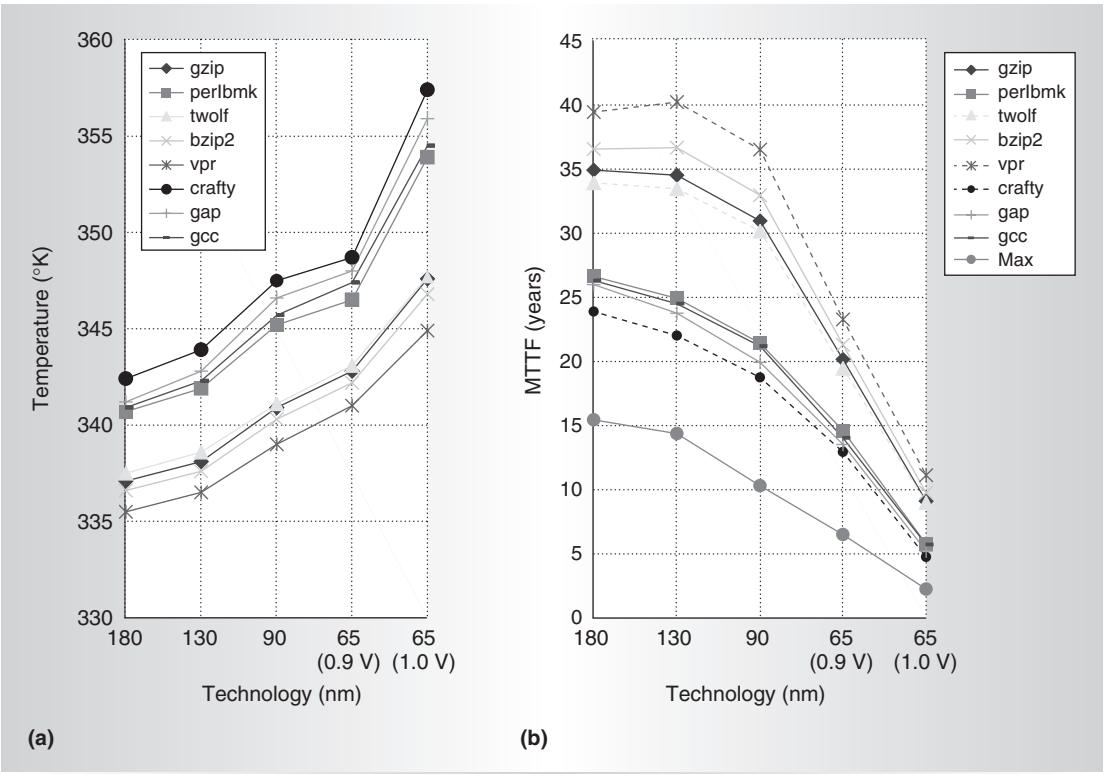

Figure 1. Peak temperature (a) and MTTF using Ramp (b) for eight SpecInt applications. At 65 nm, we show the points for the unrealistic 0.9 V and the more industry-expected 1 V. In (b), the Max curve shows that estimates based on worst case behavior are far worse than any application MTTF. These are worst case conditions only for the applications studied; the worst possible conditions for the processor are likely to result in an even lower MTTF. (The model constants are chosen so that the average MTTF at 180nm is 30 years. This baseline MTTF is chosen arbitrarily).

vidual transistor reliability is decreasing, and the number of transistors that can fail is increasing.

On-chip power management

To cope with escalating power, most modern processor designs use some form of gating, usually clock gating. Other dynamic, workload-driven adaptations of processor resources and bandwidth are also becoming part of on-chip power management.[7,9] These techniques offer the promise of reduced average power and temperature, but they are also introducing new on-chip effects, such as thermal cycling, that can degrade reliability.

Wearout failure models

Researchers and engineers led us to identify five critical wearout-based failure mechanisms for processors.[2,8] We then identified the state-of-the art device-level analytic models for these failure mechanisms,[3-5] and these models became the basis for Ramp. These models express reliability in terms of MTTF and assume steady-state operation at specific (generally worst case) temperature and use.

Electromigration

Electromigration occurs when momentum transfer transports conductor metal atoms within the processor interconnects. Atoms migrate from one end of the interconnect to the other, eventually leading to increased resistance and shorts.

Ramp uses the electromigration model[3]

$$\mathrm{MTTF}_{\mathrm{EM}} \propto \left(J\right)^{-n} \mathrm{e}^{\frac{E_{a\mathrm{EM}}}{\mathrm{k}T}}$$

where J is the current density in the interconnect, $\mathrm{E}_{a\mathrm{EM}}$ is the activation energy for electromigration, k is Boltzmann's constant, and T is absolute temperature in degrees Kelvin. n and $\mathrm{E}_{a\mathrm{EM}}$ are constants that depend on the interconnect metal used. In Ramp, these constants are 1.1 and 0.9, respectively, for copper

interconnects.[3]

Stress migration

In this phenomenon, mechanical stress causes metal atoms in the interconnect to migrate, much as they do in electromigration. Materials differ in their thermal expansion rate, and this difference causes thermomechanical stress.[3]

Ramp uses the stress migration model[3]

$$\mathrm{MTTF}_{\mathrm{SM}} \propto \left|T_0 - T\right|^{-n} \mathrm{e}^{\frac{E_{a_{\mathrm{SM}}}}{\mathrm{k}T}}$$

where T is the absolute temperature in degrees Kelvin, T_0 is the metal's stress-free temperature (the metal-deposition temperature), and m and $\mathrm{E}_{a_{\mathrm{EM}}}$ are material-dependent constants. In Ramp, these constants are 2.5 and 0.9, respectively, for copper interconnects.[3] Ramp assumes that manufacturers use sputtering to deposit the interconnect metal and thus uses a value of 500 degrees Kelvin for T_0.[3]

Time-dependent dielectric breakdown

Also known as gate-oxide breakdown, time-dependent dielectric breakdown (TDDB) is the result of the gate dielectric's gradual wearout, which leads to transistor failure.[4]

The model Ramp uses for the MTTF from TDDB is based on Wu et al.'s recent experimental work at IBM:[4]

$$\mathrm{MTTF}_{\mathrm{TDDB}} \propto \left(\frac{1}{V}\right)^{(a-bT)} \mathrm{e}^{\frac{[X+(Y/T)+ZT]}{\mathrm{k}T}}$$

where T is the absolute temperature in degrees Kelvin and a, b, X, Y, and Z are fitting parameters. The values Ramp uses are based on existing data:[4] $a = 78$, $b = -0.081$, $X = 0.759$ eV, $Y = -66.8$ eV/K and $Z = -8.37 \times 10^{-4}$ eV/K.

Thermal cycling

Permanent damage accumulates with each temperature cycle experienced by the processor, eventually leading to failure.[3] There are two types of thermal cycles. Large cycles occur at a low frequency because of changes like powering up and down. Small cycles occur at a high frequency because of changes in workload behavior and fine-grained power management. The packaging community has not extensively studied the effect of small thermal cycles at high frequency, so validated models are not available.

Ramp models large thermal cycles with

$$\mathrm{MTTF}_{\mathrm{TC}} \propto \left(\frac{1}{T - T_{\mathrm{ambient}}}\right)^{q}$$

where T_{ambient} is the ambient temperature in degrees Kelvin, $T_{\mathrm{average}} - T_{\mathrm{ambient}}$ is the average thermal cycle that an on-chip structure experiences, and q is the Coffin-Manson exponent, an empirically determined material-dependent constant (2.35 for the package we modeled in Ramp[3]).

Negative bias temperature instability

This electrochemical reaction takes place in PFETs. Negative bias temperature instability (NBTI) leads to upward shifts in the transistors' threshold voltage, which in turn lead to processor failure because of timing constraint violations.[5]

The NBTI model we used in Ramp is based on recent experimental work by Zafar et al. at IBM:[5]

$$\mathrm{MTTF}_{\mathrm{NBTI}} \propto \left\{\left[\begin{array}{l} \ln\left(\dfrac{A}{1+2\mathrm{e}^{B/\mathrm{k}T}}\right) \\ -\ln\left(\dfrac{A}{1+2\mathrm{e}^{B/\mathrm{k}T}} - C\right) \end{array}\right] \times \frac{T}{\mathrm{e}^{-D/\mathrm{k}T}}\right\}^{1/\beta}$$

where A, B, C, D, and β are fitting parameters, and k is Boltzmann's constant. The values we use in Ramp are based on previously derived data:[5] A = 1.6328, B = 0.07377, C = 0.01, D = −0.06852, and $\beta = 0.3$.

Sum-of-failure-rates model

To calculate the MTTF for the entire processor, Ramp must combine the effects of all the failure mechanisms across all chip structures and over time. In general, this combination requires knowledge of the different mechanisms' lifetime distributions, which is difficult to attain. Ramp addresses this problem by using the sum-of-failure-rates (SOFR) model.[11] The SOFR model, widely used in industry, makes two assumptions:

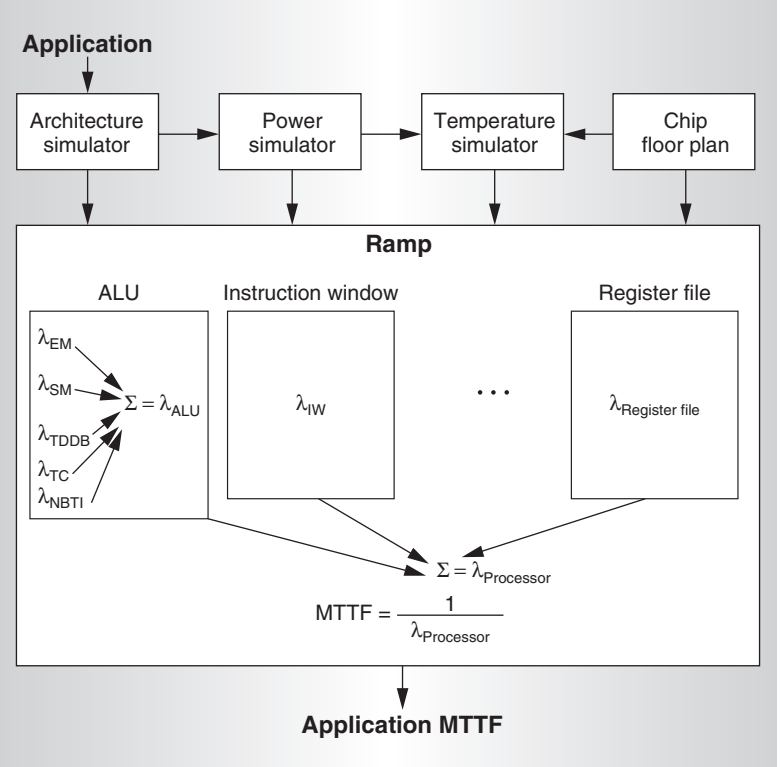

Figure 2. Ramp summary. Ramp requires cycle-by-cycle application behavior, power, and temperature information, and the chip floor plan. The architectural simulator simulates the application and feeds the power simulator. The temperature simulator uses power information and the chip floor plan as input.

- *The processor is a series failure system.* That is, the first instance of any structure failing because of any failure mechanism causes the entire processor to fail.
- *Each failure mechanism has a constant failure rate.* Equivalently, every failure mechanism has an exponential lifetime distribution. This assumption is clearly inaccurate—the failure rate of a typical wearout failure mechanism will be low at the beginning of the component's lifetime and grow as the component ages. However, manufacturers often use this assumption for lack of any better, validated models.

These assumptions have two significant implications.[11] The first is that the processor's MTTF, $MTTF_p$, is the reciprocal of the processor's total failure rate, λ_p. The second is that the processor's failure rate is the sum of the individual structures' failure rates that are due to individual failure mechanisms. Hence,

$$MTTF_p = \frac{1}{\lambda_p} = \frac{1}{\sum_{i=1}^{j}\sum_{l=1}^{k}\lambda_{il}} \quad (6)$$

where λ_{il} is the failure rate of the ith structure because of the lth failure mechanism (which is the reciprocal of the corresponding MTTF).

Further, the MTTF models assume that temperature (T), interconnect current density (J), and voltage (V) are fixed. However, when an application runs, these operating conditions vary with time. Ramp accounts for this variation through two actions. The first is by calculating instantaneous values of λ_{il} on the basis of instantaneous T, V, and J (measured over a reasonably small time granularity). The second is by using an average over time of these values to determine the actual failure rate when running the application. This averaging over time is similar to the assumption in the SOFR model, which averages over space. We sometimes refer to the processor MTTF that Ramp calculates when running an application as the application's MTTF.

Finally, to calculate absolute MTTFs, Ramp must have the proportionality constants in the individual failure mechanism models (Equations 1 through 5). These constants are a function of the reliability-qualification process and depend on many factors, such as the materials used for design, and yield. For a target MTTF, these constants dictate the worst case operating conditions (and hence performance) the processor can safely attain. High values for the proportionality constants imply more aggressive operating conditions (higher temperature, for example) and higher performance, but at a higher cost. Conversely, cheaper systems will have low values for the constants and must operate at conservative operating points with lower performance.

Ramp as a simulation tool must work in conjunction with a timing simulator to determine workload behavior and with a power and thermal simulator for power and temperature profiles. In real hardware, Ramp would require sensors and counters that provide information on processor operating conditions.

Figure 2 summarizes the process Ramp uses to obtain application MTTFs. A detailed discussion of Ramp's implementation is available elsewhere.[2]

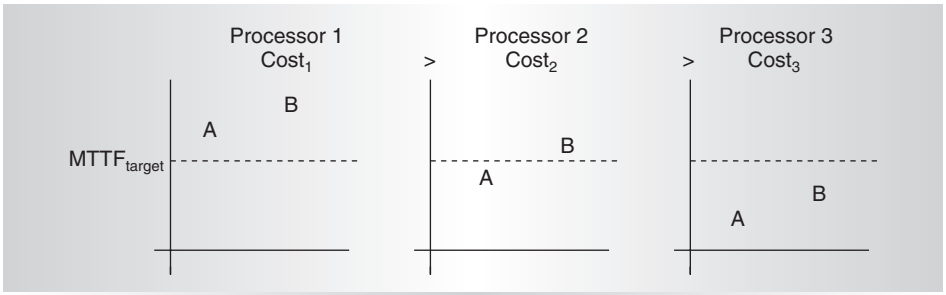

Figure 3. Motivation for dynamic reliability management (DRM). The dashed line represents the reliability target. Processor 1 is overdesigned for reliability (applications A and B are far over the target) and thus the cost is not justified. Processor 3 is cheap to qualify but neither application A nor B meets its reliability target, so the cost benefit is lost. DRM can avoid these drawbacks by adapting application performance to meet the target MTTF value.

Dynamic reliability management

Figure 3 uses three systems that trade off cost and reliability to motivate dynamic reliability management (DRM). Processor 1 is the most expensive to qualify for reliability, and processor 3 is the cheapest. Consider two applications, A and B. The applications will have different MTTFs on the three processors because the reliability qualification of the three processors is different. Processor 1 is overdesigned for reliability, so all applications it runs will meet the target MTTF value, but their MTTFs are higher than required. In processor 2, application A does not meet the target MTTF value, but application B does. In processor 3, both applications do not meet the target MTTF value. Hence, processors 2 and 3 are underdesigned for reliability for these applications.

Consider the cases of processor 2 running application A and processor 3 running either A or B. Although processors 2 and 3 are cheaper than 1 to qualify, they can fail prematurely without any architectural intervention. Hence, they do not represent acceptable design points by current reliability-qualification methodologies. With DRM, the designers of processors 2 and 3 can meet reliability targets by using processor adaptation to reduce processor temperature, current density, voltage, and/or frequency as needed at runtime. These actions incur a performance loss, but they maintain target reliability.

Now consider the cases of processor 2 running application B and processor 1 running either A or B. Current systems will not exploit the excess reliability margin. However, if the cooling solution can support it, designers can use DRM to exploit the excess reliability margin and extract additional performance, for example, by overclocking or increasing microarchitectural resources. Thus, DRM can decrease reliability-qualification cost and/or increase processor performance, while assuring that the processor meets reliability targets.

DRM evaluation

A true DRM evaluation would require proposing a control algorithm for processor adaptations and evaluating its performance on processors with different reliability-qualification cost. Our goal, however, was only to show DRM's potential, so we studied an oracular control algorithm. We considered several configurations with a range of microarchitectural resources and voltage and frequency settings. For an application running on a processor with a specific reliability-qualification cost, we selected the configuration that gave maximum performance and met the required MTTF target. This effectively simulates a DRM algorithm that adapts once per application run and chooses the adaptation configuration with oracular knowledge of the application behavior. Although the algorithm is oracular, it does not represent the best possible DRM control algorithm because it does not exploit the variability within the application.

We evaluated DRM using the RSim simulator[12] for performance (timing) evaluation, the Wattch tool[6] for power measurement, and the HotSpot tool[7] for temperature evaluation. The base microarchitecture (with no adaptations) we simulated is similar to the MIPS

Table 1. Applications used in evaluating DRM.

Application	Type	IPC	Base power* (W)
MPGdec (Mpeg video decoder)	Multimedia	3.2	36.5
MP3dec (Mp3 audio decoder)		2.8	34.7
H263enc (H263 video encoder)		1.9	30.8
bzip2	SpecInt	1.7	23.9
gzip		1.5	23.4
twolf		0.8	15.6
art	SpecFP	0.7	17.0
equake		1.4	20.9
ammp		1.1	19.7

*Dynamic power plus leakage

R10000. We used a supply voltage of 1.0 V and a base frequency of 4.0 GHz.

The specific DRM adaptations in our evaluation are a combination of microarchitectural adaptation (instruction-window resizing and changing the number of ALUs and FPUs) and dynamic voltage scaling (DVS). We modeled 18 microarchitectural configurations—from a processor with a 128-entry instruction window, six ALUs, and four FPUs to a processor with a 16-entry instruction window, two ALUs, and one FPU. For DVS, we varied the processor frequency from 2.5 GHz to 5.0 GHz. We always set the voltage to support the frequency being simulated.

Table 1 summarizes the nine applications we used for our evaluation. To study the reliability implications of various application classes, we chose three multimedia applications, three Spec2000 integer applications, and three Spec2000 floating-point applications. As the table shows, these applications exhibit a wide range of instructions per cycle (IPC) and power consumption. We describe our experimental methodology in greater detail elsewhere.[2]

Results

Figure 4 shows the performance for all the applications for processors with a range of reliability qualification costs. The performance is the result of combining microarchitectural and DVS adaptations to control reliability, represented as an increase or slowdown over the base processor (which is the most aggressive microarchitectural configuration, running at 4.0 GHz). A value of 1.0 represents no gain or loss. In the Ramp model, the proportionality constants in Equations 1 to 5 reflect reliability qualification costs. However, since we do not have the function that relates these constants to cost, we use an architectural parameter as a proxy for cost. We use the maximum constant temperature the processor can run at while meeting the target MTTF, denoted T_{qual}, as the proxy.[2] For a higher cost, we have a higher T_{qual}, which lets the processor reach higher temperatures, and so higher performance, for a target reliability. Figure 4 shows results for processors with four values of T_{qual}—400°K, 370°K, 345°K, and 325°K.

T_{qual} = 400° K. The hottest on-chip temperature any application in our evaluated workload reached was near 400°K. Hence, this value of T_{qual} represents a lower bound on the qualification temperature that designers could choose using current worst-case-based reliability qualification. As Figure 4 shows, all the applications experience performance gains—from a gain of 6 percent for MP3dec to 16 percent for twolf—while maintaining the required processor reliability levels. On-chip operating conditions generally tend to be much lower than the worst case values, so all the applications can run at higher than base frequency. At the higher frequency, the temperature will occasionally exceed 400°K but the processor will maintain the target MTTF value because lower values at other times compensate for high instantaneous temperature values.

The performance gains for the Spec benchmarks tend to be higher on average than those for the multimedia benchmarks, primarily because the multimedia benchmarks have higher IPC and consequently higher operating temperatures and current densities, which gives them lower MTTFs.

Although 400°K is the highest temperature we saw in our workload suite, the worst possible temperature the processor can reach is likely to be even higher. If the designer qualified the processor for that temperature, DRM would provide even higher performance gains.

These results imply that qualifying for worst case operating conditions is overly conservative. A more economic alternative is to qualify at lower operating points, which would be less expensive, or to market the base processor at a higher frequency while still meeting the MTTF target.

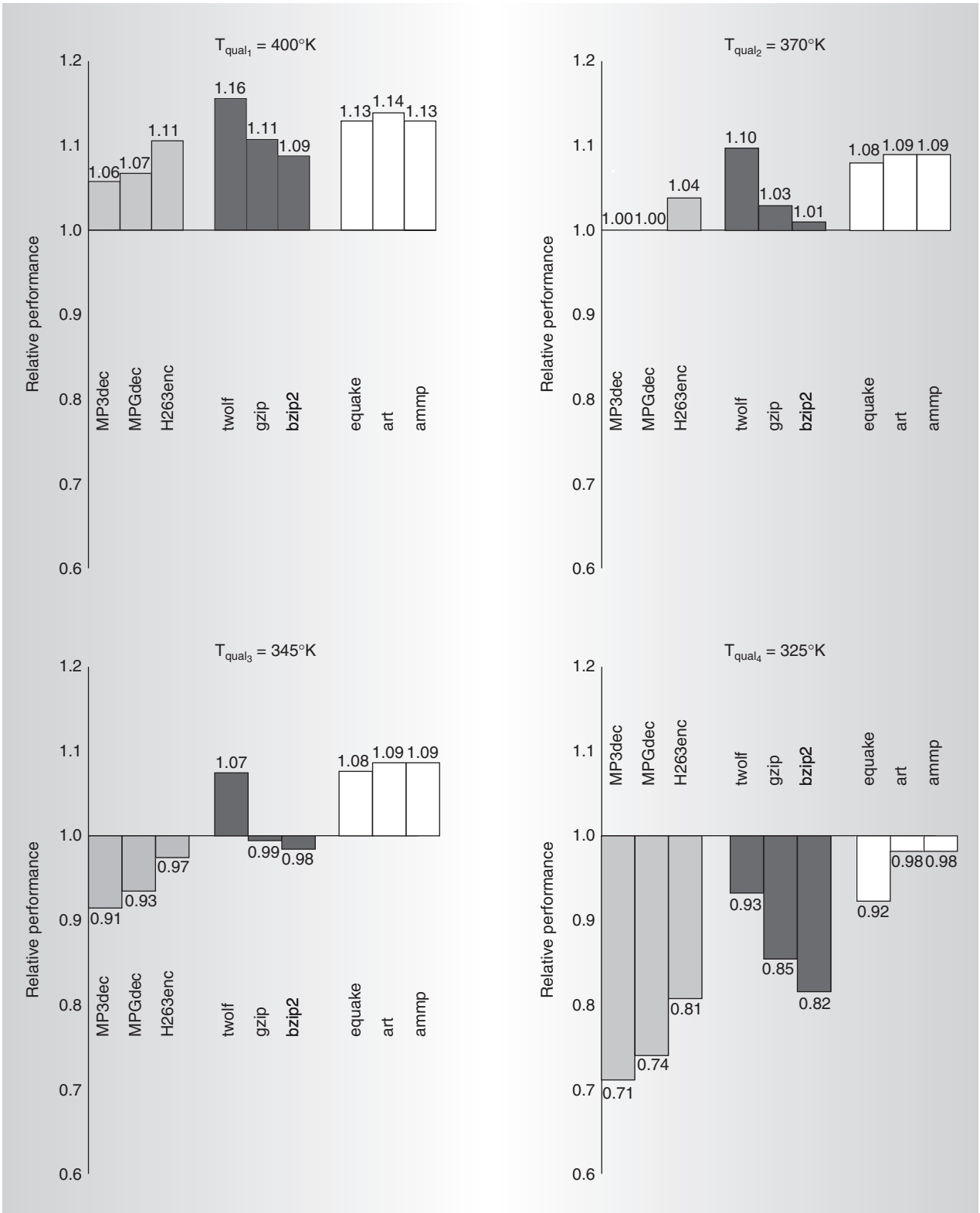

Figure 4. Performance on nine applications with DRM adaptations for four processors with different reliability qualification costs. Performance is relative to that for a base processor (without DRM adaptations) running at 4 GHz. The processor with T_{qual} = 400°K is the most expensive and the processor with T_{qual}=325°K is the cheapest.

T_{qual} = 370°K. For this processor, MP3dec and MPGdec, the applications with the lowest MTTF values on the base processor, have no performance gain. All the other applications have a performance gain of from 1 percent for bzip2 to 10 percent for twolf. This represents a processor qualified for reliability on the basis of application behavior.

Rather than selecting T_{qual} based on the worst case application operating conditions of 400°K, reliability qualification is based on what it would take for the worst applications (MP3dec and MPGdec) to just meet the reliability target. Such an application-oriented approach to reliability qualification represents savings in qualification cost without any performance loss. DRM never curtails performance in this scenario for these applications. Again, lower IPC applications see the largest performance gains (twolf, art and ammp).

T_{qual} = 345°K. This represents a processor qualified for the average, rather than worst case, application. As Figure 4 shows, the performance of all the applications with DRM adaptations was within 10 percent of the base value; in three cases, it was within 5 percent. This potentially represents an excellent cost-performance tradeoff design point, where designers can use DRM to reduce the cost of processor qualification without incurring significant performance penalties. Predictably, high IPC applications experience the largest performance losses, while low IPC applications enjoy the largest gains.

T_{qual} = 325°K. This represents a processor with drastically reduced reliability qualification cost. As Figure 4 shows, all applications experience a slowdown. In this scenario, the performance loss overshadows the cost benefit of designing for a cheaper reliability-qualification point. The multimedia applications with high IPC experience the largest slowdown, with MP3dec suffering a 29 percent loss in performance.

Implications for reliability-aware design

These results imply that DRM offers the potential for cost benefits without performance loss. Changing the reliability design point from a T_{qual} of 400°K to 370°K saves design cost without slowing any of the applications—evidence that worst case reliability qualification is unnecessarily conservative. Using DRM, and allowing some performance degradation, designers can further lower the reliability-qualification cost. In our results, even at a T_{qual} of 345°K, performance loss was limited.

Finally, the results show that the performance-cost trade-off depends on the processor's intended application domain. A processor designed for Spec applications, for example, could use cheaper reliability qualification than a processor intended for multimedia applications.

Open research problems

With emerging technology-scaling trends and increasing power densities, processor manufacturers can no longer afford to consider lifetime reliability as solely a problem for device and circuit designers. The stakes are high enough that lifetime reliability should be a first-class design constraint even at the microarchitectural level. Ramp and DRM are steps toward an architectural solution to cost-effective lifetime reliability assurance, but many open research problems remain.

Ramp validation and extension

As with any simulator or model, Ramp makes several assumptions. Although some of these remain unvalidated, most are grounded in "current practice," and are the result of extensive consultations with research and product groups about processor reliability at IBM. Further, the individual failure mechanism models, which are Ramp's key underlying components, represent the state of the art. Nevertheless, either a relaxation of the assumptions or a validation with real-world failure data would increase Ramp's utility as an industry and research tool. Recently, we have relaxed some of the assumptions we describe here in an upgraded Ramp version.[8]

Microarchitectural techniques for lifetime reliability

We have shown the potential for DRM, but its effective use requires nonoracular control algorithms. The selective redundancy techniques we have explored[8] require more work on designing such redundancy and determining when failure occurs. Doubtless, other microarchitectural techniques for designing reliability-aware microarchitectures await exploration. Effectively, our work allows the

system to view reliability as a resource that it can judiciously distribute over time and space to meet the required MTTF target with the best performance. Further, microarchitectural solutions will need to proceed hand in hand with reliability advances in other stages of system design (circuit- and device-level solutions).

Early-life failures

Early-life processor failures resulting from manufacturing defects are of critical concern to microprocessor manufacturers because they directly affect processor yield and cost. Burn-in, a process that subjects manufactured processors to elevated temperature and voltage conditions to accelerate the onset of early-life failures, can eliminate many early-life failures.[13] Longer burn-in periods capture more early-life failures, increasing shipped processor reliability. However, excessive burn-in is undesirable because it reduces the lifetime of nondefective processors. Also, because operating burn-in ovens costs both time and resources, manufacturers should carefully determine burn-in times. A joint approach to burn-in efficiency and lifetime reliability could potentially provide more cost effective burn-in options. On the basis of the processor's target application suite, manufacturers could determine burn-in times that provide the target reliability for the lowest cost.

Soft errors

A complete picture of processor reliability requires considering the effect of wearout failures and soft (or transient) errors in combination. Researchers have done much work recently on soft error modeling and mitigation,[14,15] but a fundamental tension exists between processor lifetime and soft error control. Increasing lifetime through the common DVS power-management technique will likely increase susceptibility to soft errors. On the other hand, soft error protection through redundancy could increase power and temperature, which could lead to lower processor lifetimes. Integrated solutions that balance the two reliability issues are an important avenue of future research.

Optimizing energy, temperature, reliability together

Currently, processor designers independently address energy, temperature, and reliability during design. Handling these issues in a unified fashion could provide large cost and performance benefits. Despite the similarities in their root causes, the relationship between processor energy, temperature, and reliability is not obvious. In earlier work,[2] we compared processor design for temperature (dynamic thermal management) and design for reliability (DRM) and showed that neither technique subsumes the other. Thus more work is required to understand these effects individually and in concert. Such an effort would provide a path to a unified framework that would likely have a significant impact on processor design, allowing optimal cost-performance trade-offs that are based on the target system's requirements.

In this era of power-constrained design, the effect of escalating on-chip temperatures on chip reliability is of increasing concern to processor and system developers. Ramp represents a first attempt at addressing this emerging challenge from a microarchitectural perspective. Much work remains to improve Ramp's modeling methodology and validation and to design microarchitectural techniques for lifetime reliability control and enhancement. We have recently addressed the limitations of the SOFR model using Monte-Carlo methods with log-normal failure models instead of exponential models and series-parallel failure models instead of series-only models.[8] We are also evaluating burn-in algorithms that will provide maximum shipped reliability at the least cost. In future work, we will study nonoracular algorithms for DRM in addition to exploring other microarchitectural reliability enhancement techniques. MICRO

References

1. "Critical Reliability Challenges for the International Technology Roadmap for Semiconductors," *Int'l Sematech Tech. Transfer 03024377A-TR*, 2003.
2. J. Srinivasan et al., "The Case for Lifetime Reliability-Aware Microprocessors," *Proc. Int'l Symp. Computer Architecture,* IEEE CS Press, 2004, pp. 276-287.
3. "Failure Mechanisms and Models for Semiconductor Devices," *Joint Electron Device Eng. Council Pub. JEP122-A*, 2002.
4. E.Y. Wu et al., "Interplay of Voltage and Tem-

perature Acceleration of Oxide Breakdown for Ultra-Thin Gate Dioxides," *Solid-state Electronics J.*, Nov. 2002, pp. 1787-1798.
5. S. Zafar et al., "A Model for Negative Bias Temperature Instability (NBTI) in Oxide and High K PFETs," *Symposia VLSI Technology and Circuits*, 2004, pp. 45-50.
6. D. Brooks et al., "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," *Proc. Int'l Symp. Computer Architecture,* IEEE CS Press, 2000, pp. 83-94.
7. K. Skadron et al., "Temperature-Aware Microarchitecture," *Proc. Int'l Symp. Computer Architecture*, IEEE CS Press, 2003, pp. 2-13.
8. J. Srinivasan et al., "Exploiting Structural Duplication for Lifetime Reliability," *Proc. Int'l Symp. Comp. Architecture*, IEEE CS Press, 2005 (to be published).
9. R. Sasanka et al., "Joint Local and Global Hardware Adaptations for Energy," *Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems,* ACM Press, 2002, pp. 144-155.
10. J. Srinivasan et al., "The Impact of Technology Scaling on Lifetime Reliability," *Proc. Int'l Conf. Dependable Systems and Networks*, IEEE Press, 2004, pp. 177-186.
11. K. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice Hall, 1982.
12. C. J. Hughes et al., "RSIM: Simulating Shared-Memory Multiprocessors with ILP Processors," *Computer*, Feb. 2002, pp. 40-49.
13. K. Seshan et al., "The Quality and Reliability of Intel's Quarter Micron Process," *Intel Technology J.*, no. 3,1998.
14. S.S. Mukherjee et al., "A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor," *Proc. Int'l Symp. Microarchitecture,* IEEE CS Press, 2003, pp. 29-40.
15. P. Shivakumar et al., "Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic," *Intl. Conf. Dependable Systems and Networks*, 2002, pp. 389-398.

Jayanth Srinivasan is a PhD candidate in computer science at the University of Illinois at Urbana-Champaign. His research focuses on reliability, power and temperature constraints for future architectures, and adaptive processors and systems. Srinivasan has an MS in computer science from the University of Illinois and a BTech from the Indian Institute of Technology, Madras. He is a recipient of the IBM PhD Fellowship and a student member of the IEEE and ACM. The bulk of the work in this article was performed while Srinivasan was an intern at IBM T.J. Watson Research Center during the summers of 2003 and 2004.

Sarita V. Adve is an associate professor in the Department of Computer Science at the University of Illinois at Urbana-Champaign. Her research interests are computer architecture, with a current focus on power-efficient and reliable systems. Adve received a PhD in computer science from the University of Wisconsin-Madison.

Pradip Bose is a research staff member and manager at the IBM T.J. Watson Research Center, as well as editor in chief of *IEEE Micro*. His current research interests include high-performance computer architectures, power- and complexity-aware design, and reliable systems. Bose has a PhD in electrical and computer engineering from the University of Illinois, Urbana-Champaign. He is a senior member of the IEEE and the IEEE CS and a member of the ACM.

Jude A. Rivers is a research staff member at the IBM T.J. Watson Research Center. His research interests focus on high-performance computer architectures, and include memory systems architecture and power-, temperature-, and reliability-aware design. Rivers has a PhD in computer science and engineering from the University of Michigan, Ann Arbor. He is a member of the IEEE and ACM.

Direct questions and comments about this article to Jayanth Srinivasan, CS Dept., University of Illinois at Urbana-Champaign, 401 N. Goodwin Ave., Urbana, IL 61801; srinivsn@cs.uiuc.edu.