To appear in the proceedings of *The 31st International Symposium on Computer Architecture (ISCA-04)*, June 2004.

The Case for Lifetime Reliability-Aware Microprocessors *

Jayanth Srinivasan, Sarita V. Adve
University of Illinois at Urbana-Champaign
Department of Computer Science
{srinivsn,sadve}@cs.uiuc.edu,

Pradip Bose, Jude A. Rivers
IBM T.J. Watson Research Center
Yorktown Heights,NY
{pbose,jarivers}@us.ibm.com

Abstract

Ensuring long processor lifetimes by limiting failures due to wear-out related hard errors is a critical requirement for all microprocessor manufacturers. We observe that continuous device scaling and increasing temperatures are making lifetime reliability targets even harder to meet. However, current methodologies for qualifying lifetime reliability are overly conservative since they assume worst-case operating conditions. This paper makes the case that the continued use of such methodologies will significantly and unnecessarily constrain performance. Instead, lifetime reliability awareness at the microarchitectural design stage can mitigate this problem, by designing processors that dynamically adapt in response to the observed usage to meet a reliability target.

We make two specific contributions. First, we describe an architecture-level model and its implementation, called RAMP, that can dynamically track lifetime reliability, responding to changes in application behavior. RAMP is based on state-of-the-art device models for different wear-out mechanisms. Second, we propose dynamic reliability management (DRM) *– a technique where the processor can respond to changing application behavior to maintain its lifetime reliability target. In contrast to current worst-case behavior based reliability qualification methodologies, DRM allows processors to be qualified for reliability at lower (but more likely) operating points than the worst case. Using RAMP, we show that this can save cost and/or improve performance, that dynamic voltage scaling is an effective response technique for DRM, and that dynamic thermal management neither subsumes nor is subsumed by DRM.*

*This work is supported in part by an equipment donation from AMD Corp. and the National Science Foundation under Grant No. CCR-0209198, CCR-0205638, EIA-0224453, and CCR-0313286. A large part of the work was performed while Jayanth Srinivasan was a summer intern at IBM T. J. Watson Research Center.

1 Introduction

Ensuring long-term, or "lifetime" reliability, as dictated by the hard error rate due to wear-out based failures, is a critical requirement for all microprocessor manufacturers. However, relentless technology scaling coupled with increasing processor power densities are threatening the nearly unlimited lifetime reliability standards that customers have come to expect. This has led the International technology roadmap for semiconductors (ITRS) to predict the onset of significant lifetime reliability problems, and at a pace that has not been seen in the past. It is expected that in the future, product cost and performance requirements will be substantially affected, and in many cases, superseded by constraints brought on by processor wear-out and dwindling lifetime reliability.

Traditionally, microarchitects have treated the issue of processor lifetime reliability as a manufacturing problem, best left to be handled by device and process engineers. We observe that the microarchitecture's ability to track application behavior can potentially be leveraged to the benefit of reliability qualification, enabling reduced reliability design costs, increased processor yield, and/or increased performance.

This paper represents a first attempt at evaluating the potential benefit of designing lifetime aware microprocessors. Such design requires tools and models to evaluate lifetime reliability at early processor design stages. Using industrial strength models for lifetime failure modes, we develop a methodology, called RAMP, to estimate lifetime reliability from an architectural and application perspective. We then propose a new technique, *Dynamic Reliability Management (DRM)*, which adapts the processor in response to changing workloads to ensure a target lifetime reliability at better performance and/or cost.

1.1 Classification of processor errors

Processor errors can be broadly classified into two categories: soft and hard errors.

Soft errors, also called transient faults or single-event upsets (SEUs), are errors in processor execution due to elec-

trical noise or external radiation, rather than design or manufacturing related defects. Extensive research is being performed by the architecture community to make processors resilient to soft errors [11, 23]. Although soft errors can cause errors in computation and data corruption, they do not fundamentally damage the microprocessor and are not viewed as a lifetime reliability concern. As a result, we do not discuss soft errors in this paper.

Hard errors are caused by defects in the silicon or metalization of the processor package, and are usually permanent once they manifest. Hard errors, or hard failures can be further divided into extrinsic failures and intrinsic failures [13].

Extrinsic failures are caused by process and manufacturing defects and occur with a decreasing rate over time. For example, contaminants on the crystalline silicon surface, and surface roughness can cause gate oxide breakdown [24]. Other examples include short circuits and open circuits in the interconnects due to incorrect metalization during fabrication. Extrinsic failures are mainly a function of the manufacturing process – the underlying microarchitecture has very little impact on the extrinsic failure rate. After manufacturing, using a technique called burn-in [16], the processors are tested at elevated operating temperatures and voltages in order to accelerate the manifestation of extrinsic failures. Since most of the extrinsic failures are weeded out during burn-in, shipped chips have a very low extrinsic failure rate. Semiconductor manufacturers and chip companies continue to extensively research methods for improving burn-in efficiency, and reduce extrinsic failure rates [16].

Intrinsic failures are those related to processor wear-out and are caused over time due to operation within the specified conditions. These failures are intrinsic to, and depend on, the materials used to make the processor and are related to process parameters, wafer packaging, and processor design. If the manufacturing process was perfect and no errors were made during design and fabrication, all hard processor failures would be due to intrinsic failures. Intrinsic failures occur with an increasing rate over time. It is essential that these failures do not occur during the intended lifetime of the device when it is used under specified operating conditions [1]. Examples of intrinsic failures include time dependent dielectric breakdown (TDDB) in the gate oxides, electromigration and stress migration in the interconnects, and thermal cycling and cracking.

As discussed earlier, burn-in attempts to filter out all processors which manifest early-life or extrinsic failures. As a result, processor lifetime reliability tends to be almost completely dependent on wear-out failures or intrinsic hard failures. Very little microarchitectural research has been done on modeling and analyzing intrinsic failures, and these are the focus of our work.

1.2 Main lifetime reliability challenges

Although providing significant benefits in microprocessor performance, advances in technology are accelerating the onset of intrinsic failures, causing a reduction in processor lifetimes. Specifically, the three main reasons are:

Processor scaling and increasing power densities. Device miniaturization due to scaling is increasing processor power densities and eating away at process and design margins. Scaling decreases lifetime reliability by shrinking the thickness of gate and inter layer dielectrics, increasing current density in interconnects, and by raising processor temperature which exponentially accelerates wear-out failures. Scaled down transistors in deep sub-micron CMOS technologies also have significantly higher leakage power which has an exponential dependence on temperature leading to even higher processor temperatures. Finally, supply voltage and threshold voltage are not scaling appropriately with technology because of performance and leakage power concerns creating further reliability problems. We quantify the impact of technology scaling on processor lifetime reliability in [20].

Increasing transistor count. Increasing functionality, facilitated by increasing transistor densities, causes the transistor count of processors to grow rapidly. More transistors result in more failures which results in lower processor lifetimes. Hence, not only is the reliability of individual transistors decreasing, the number of transistors that can fail is also increasing.

The advent of on-chip power management techniques like gating and adaptive processing. In order to cope with escalating power, most modern processor designs employ some form of gating, usually of the clock. Other forms of dynamic, workload-driven adaptation of processor resources and bandwidth are also becoming part of on-chip power management [15, 18]. These techniques are promising from the point of view of reducing average power and temperature; however, they introduce new effects on chip like thermal cycling which may have a negative impact on reliability.

1.3 Architectural awareness of lifetime reliability

We make the case that lifetime reliability must be treated as a first class design constraint at the microarchitectural design stage. This is true for all market segments ranging from server class processors where lifetime reliability is an implicit requirement, to commodity processors where reliability impacts the number of processors shipped (yield) and resultant profit.

Extensive research has gone into techniques that can improve energy and thermal efficiency by exploiting microarchitectural features and adaptation capabilities. A similar approach can be used for lifetime reliability – the microarchitecture's unique knowledge of application run-time

behavior can be leveraged to increase processor reliability. Such an approach to reliability is fundamentally different from existing methodologies where processor reliability is qualified during device design, circuit layout, manufacture, and chip test. Current reliability qualification mechanisms are oblivious to application behavior and are based on worst-case temperature and utilization estimates.

Due to variations in IPC, resource utilization, and temperature, different applications have different effects on the processor's lifetime reliability. The microarchitecture's awareness of application behavior can be used in two scenarios:

Over-designed processors. Current reliability qualification is based on worst case temperature and utilization; however, most applications will run at lower temperature and utilization resulting in higher reliability and longer processor lifetimes than required. If the processor cooling solution can handle it, this excess reliability can be utilized by the processor to increase application performance. For example, a judicious increase of voltage/frequency and/or microarchitectural resources could increase application performance while still maintaining system target reliability. Such an approach would be particularly beneficial in high-end server class processors. These processors tend to have expensive cooling and packaging and are over-designed from a reliability perspective, providing reliability margins that can potentially be used to increase performance.

Under-designed processors. An adaptive approach can also be used to safely under-design processors from a reliability approach, thereby saving cost. In an approach similar to dynamic thermal management (DTM) [18], the processor reliability qualification can be based on expected processor utilization and temperature, rather than worst case values. This would result in significant design cost reductions and would provide higher processor yield. In situations where applications exceed the reliability design limit, the processor can adapt by throttling performance to maintain the system reliability target. Such an approach would be beneficial to commodity processors where increasing yield and reducing cooling costs would have significant impact on profits, even if they incur some performance loss.

1.4 Contributions

This paper makes two sets of contributions. First, we introduce the first application-aware architecture-level methodology for evaluating processor lifetime reliability, using state-of-the-art analytic models for important intrinsic failure mechanisms. This methodology, called RAMP (**R**eliability **A**ware **M**icro**p**rocessor) can be implemented in a simulator for design stage analysis, and in hardware to incorporate reliability awareness.

Second, we propose *dynamic reliability management (DRM)* – a technique where the processor can respond to changing application behavior to maintain its lifetime reliability target. DRM allows the exploitation of both the over-designed and under-designed processor scenarios discussed in Section 1.3, thereby improving performance and/or cost over current worst-case based methodologies. Our results using RAMP with SpecInt, SpecFP, and multimedia applications show that (1) DRM can be used to improve performance or lower cost, providing the designer with a spectrum of effective cost-performance tradeoffs, (2) dynamic voltage scaling is an effective response for DRM, and (3) in spite of the similarities between dynamic thermal management (DTM) and DRM, neither technique subsumes the other and future systems must provide mechanisms to support both together.

2 Related Work

As mentioned previously, the bulk of recent work on architectural awareness of reliability has concentrated on soft errors [11, 23]. Although some soft error correction schemes can be used to increase tolerance to hard errors, the bulk of this research will not impact the hard failure rate.

Current techniques for enhancing hard failure reliability focus on fault-tolerant computing methods like redundancy [19] and efficient failure recovery methods [12]. However, these techniques are typically used in server class processors and the redundancy is targeted at minimizing down-time. Fault-tolerant microarchitectures like DIVA [5, 14] can recover from hard failures in the main processing core, but at a huge performance cost. Recent work by Shivakumar et al. examines techniques to increase processor manufacturing yield by exploiting microarchitectural redundancy [17]. They also suggest that this redundancy can be exploited to increase useful processor lifetime. All the above techniques are targeted at error detection, recovery, minimizing down time, and increasing yield. They do not attempt to impact the rate of wear-out or lifetime reliability of processors.

There is an extensive body of knowledge in the device design and manufacturing community on understanding and modeling hard failure mechanisms [2, 21, 24]. However, most of this work looks at different failure mechanisms separately – it does not attempt to unify the mechanisms to form a system wide failure model and is also application oblivious.

3 The RAMP Model

Based on extensive discussions with front-end, back-end, and reliability engineers at IBM, we determined that the critical intrinsic failure mechanisms for processors are: Electromigration, stress migration, gate-oxide breakdown or time dependent dielectric breakdown (TDDB), and thermal cycling [2, 3]. Based on these discussions, we also identified the state-of-the-art device-level analytic models

for these failure mechanisms [2, 24]. These models assume steady state operation at specific (generally worst-case) temperature and utilization, and express reliability in terms of MTTF (mean time to failure or the expected lifetime of the processor). RAMP uses these models, discussed in Sections 3.1 to 3.4, to calculate an "instantaneous" MTTF based on current temperature and utilization. Much like previous power and temperature models [7, 18], RAMP divides the processor into a few structures – ALUs, FPUs, register files, branch predictor, caches, load-store queue, instruction window – and applies the analytic models to each structure as an aggregate. Section 3.5 describes how RAMP combines the structure-level MTTF to give the MTTF for the full processor. Section 3.6 describes how RAMP incorporates application-driven temporal variations of temperature and utilization, to calculate the processor MTTF for an application. Finally, Section 3.7 discusses the process of reliability qualification and its incorporation into RAMP.

As a simulation tool, RAMP should be used in conjunction with a timing simulator to determine workload behavior, and a power and thermal simulator for power and temperature profiles. This is discussed further in Section 6. In real hardware, RAMP would require sensors and counters that provide information on processor operating conditions.

3.1 Electromigration

Electromigration occurs in aluminum and copper interconnects due to the mass transport of conductor metal atoms in the interconnects. Conducting electrons transfer some of their momentum to the metal atoms of the interconnect – this "electron wind" driving force creates a net flow of metal atoms in the direction of electron flow. As the atoms migrate, there is depletion of metal atoms in one region and pile up in other regions. The depletion sites can see increased interconnect resistance or open circuits, and extrusions can occur at sites of metal atom pile up. Electromigration has an exponential dependence on temperature. Extensive research has been performed by the material science and semiconductor community on modeling the effects of electromigration and it is a well understood failure mechanism [2].

The currently accepted model for MTTF due to electromigration ($MTTF_{EM}$) is based on Black's equation and is as follows [2]:

$$MTTF_{EM} \propto (J - J_{crit})^{-n} e^{\frac{E_{a_{EM}}}{kT}} \quad (1)$$

where J is the current density in the interconnect, J_{crit} is the critical current density required for electromigration, $E_{a_{EM}}$ is the activation energy for electromigration, k is Boltzmann's constant, and T is absolute temperature in Kelvin. n and $E_{a_{EM}}$ are constants that depend on the interconnect metal used (1.1 and 0.9 copper interconnects as modeled in RAMP [2]). J tends to be much higher than J_{crit} in interconnects (nearly 2 orders of magnitude [2]). Hence, $(J - J_{crit}) \approx J$. J for an interconnect can be related to the switching probability of the line, p, as [8]

$$J = \frac{CV_{dd}}{WH} \times f \times p \quad (2)$$

where C, W, and H are the capacitance, width, and thickness, respectively of the line and f is the clock frequency.

Currently, RAMP assumes all interconnects in a structure to be similar, and does not differentiate interconnects on the basis of their C, W, and H (we currently fold these terms into the proportionality constant). The activity factor (or switching probability or utilization) of a structure, p, is obtained from the timing simulator.

3.2 Stress migration

Much like electromigration, stress migration is a phenomenon where the metal atoms in the interconnects migrate. It is caused by mechanical stress due to differing thermal expansion rates of different materials in the device. This stress, σ, is proportional to the change in temperature, $T_0 - T$, where T is the operating temperature and T_0 is the stress free temperature (metal deposition temperature) of the metal. That is, when the metal was originally deposited on the device, there is no thermo-mechanical stress. At other temperatures, there are stresses due to differing expansion rates. The exact mechanisms behind stress migration are still not completely understood and research is ongoing on the subject [2].

The mean time to failure due to stress migration, $MTTF_{SM}$, as modeled in RAMP is given by [2]:

$$MTTF_{SM} \propto |T_0 - T|^{-n} e^{\frac{E_{a_{SM}}}{kT}} \quad (3)$$

where the temperatures are in Kelvin, and n and $E_{a_{SM}}$ are material dependent constants (2.5 and 0.9 for the copper interconnects modeled [2]). RAMP assumes that sputtering (versus vapor deposition) was used to deposit the interconnect metal and uses a value of 500K for T_0 [2].

The relationship between stress migration and temperature is governed by two opposing properties. The exponential temperature relationship accelerates wear-out with increases in temperature. However, since metal deposition temperatures tend to be higher than typical operating temperatures, higher operating temperatures decrease the value of $T_0 - T$, thus reducing the value of σ and increasing the MTTF. However, this increase in MTTF is typically much smaller than the exponential decrease due to temperature.

3.3 Time-dependent dielectric breakdown

Time-dependent dielectric breakdown (TDDB), or gate oxide breakdown, is another well studied failure mechanism in semiconductor devices. The gate dielectric wears down with time, and fails when a conductive path forms in the

dielectric [21, 24]. The advent of thin and ultra-thin gate oxides, coupled with non-ideal scaling of supply voltage is accelerating TDDB failure rates [3].

The TDDB model we use is based on recent experimental work done by Wu et al. [24] at IBM. Wu et al. collected experimental data over a wide range of oxide thicknesses, voltages, and temperatures to create a unified TDDB model for current and future ultra-thin gate oxides. The model shows that the lifetime due to TDDB for ultra-thin gate oxides is highly dependent on voltage and has a larger than exponential degradation due to temperature. Based on [24], the MTTF due to TDDB, $MTTF_{TDDB}$, at a temperature, T, and a voltage, V, is given by:

$$MTTF_{TDDB} \propto (\frac{1}{V})^{(a-bT)} e^{\frac{(X+\frac{Y}{T}+ZT)}{kT}} \quad (4)$$

where a, b, X, Y, and Z are fitting parameters.

Based on data in [24], the values currently used in RAMP are $a = 78$, $b = -0.081$, $X = 0.759ev$, $Y = -66.8evK$, and $Z = -8.37E-4ev/K$.

3.4 Thermal cycling

Temperature cycles can cause fatigue failures. Damage accumulates every time there is a cycle in temperature, eventually leading to failure. Although all parts of the device experience fatigue, the effect is most pronounced in the package and die interface (for example, solder joints) [2].

The package goes through two types of thermal cycles – The first type are large thermal cycles that occur at a low frequency (a few times a day). These include powering up and down, or going into low power or stand-by mode for mobile processors. The second type are small cycles which occur at a much higher frequency (a few times a second). These are due to changes in workload behavior and context switching. The effect of small thermal cycles at high frequencies has not been well studied by the packaging community, and validated models are not available. As a result, we do not discuss models for the reliability impact of small thermal cycles.

Large thermal cycles are modeled using the Coffin-Manson equation [2]:

$$N_f = C_0(\Delta T)^{-q} \quad (5)$$

where N_f is the number of thermal cycles to failure, C_0 is an empirically determined material-dependent constant, ΔT is the temperature range experienced in the thermal cycle, and q is the Coffin-Manson exponent, an empirically determined constant.

Using Equation 5, we can see that the MTTF due to thermal cycling depends on the frequency of cycling, and on the magnitude of the cycles. Hence, the equation used to determine mean time to failure due to thermal cycles ($MTTF_{TC}$) is:

$$MTTF_{TC} \propto (\frac{1}{T - T_{ambient}})^q \quad (6)$$

where the proportionality constant also factors in the frequency of thermal cycling, which we assume stays constant. T is the average temperature of the structure and $T_{ambient}$ is the ambient temperature. Hence, $T - T_{ambient}$ represents the thermal cycle modeled. As mentioned, RAMP only models cycling fatigue in the package, since that is where the impact of cycling is most pronounced. For the package, the value of the Coffin-Manson exponent, q, is 2.35 [2].

3.5 Sum-of-failure-rates (SOFR) model

To obtain the overall reliability of a processor, we need to combine the effects of different failure mechanisms, across different structures. This requires knowledge of lifetime distributions of the failure mechanisms, and is generally difficult.

A standard model used by the industry is the Sum-of-failure-rates (SOFR) model, which makes two assumptions to address this problem: (1) the processor is a series failure system – in other words, the first instance of any structure failing due to any failure mechanism causes the entire processor to fail; and (2) each individual failure mechanism has a constant failure rate (equivalently, every failure mechanism has an exponential lifetime distribution). The failure rate (also known as the hazard function), $h(t)$ at a time t, is defined as the conditional probabilty that a component will fail in the interval $(t + \delta t)$, given that it has survived till time t. A constant failure rate implies that the value of $h(t)$ remains fixed, and does not vary with the component's age; i.e., $h(t) = \lambda$. This assumption is clearly inaccurate – a typical wear-out failure mechanism will have a low failure rate at the beginning of the component's lifetime and the value will grow as the component ages. Nevertheless, it is used for lack of better models.

The above two assumptions imply [22]: (1) the MTTF of the processor, $MTTF_p$, is the inverse of the total failure rate of the processor, λ_p; and (2) the failure rate of the processor is the sum of the failure rates of the individual structures due to individual failure mechanisms. Hence,

$$MTTF_p = \frac{1}{\lambda_p} = \frac{1}{\sum_{i=1}^{j}\sum_{l=1}^{k}\lambda_{il}} \quad (7)$$

where λ_{il} is the failure rate of the i^{th} structure due to the l^{th} failure mechanism.

The standard method of reporting constant failure rates for semiconductor components is Failures in Time (FITs) [22], which is the number of failures seen per 10^9 device hours – $MTTF_p = \frac{1}{\lambda_p} = \frac{10^9}{FIT}$. From this point on, we refer to processor reliability in terms of its FIT value.

Finally, it is important to understand that the processor FIT value alone does not portray a complete picture of pro-

cessor lifetime reliability. The time distribution of the lifetimes is also important. Incorporating time dependent failure models is an important area of our future work.

3.6 Computing FIT values for applications

The MTTF models used in RAMP (Equations 1, 3, 4, and 6) provide FIT estimates for fixed operating parameters (in particular, fixed temperature (T), voltage (V), and frequency (f), and activity factor (p)). However, when an application runs, these parameters can all vary with time (V and f vary in processors with DVS). We assume that we can account for the impact of this variation by: (1) calculating a FIT value based on instantaneous T, V, f, and p (measured over a reasonably small time granularity); and (2) using an average over time of these values to determine the actual FIT value of every structure for every failure mechanism when running the application (this averaging over time is similar to the assumption used in the SOFR model which averages over space). For thermal cycling, we calculate the average temperature over the entire run, which is used in Equation 6 to determine the thermal cycling FIT value. Sometimes we refer to this average FIT value as the FIT value of the application. To determine the FIT value for a workload, we can use a weighted average of the FIT values of the constituent applications.

3.7 The reliability qualification process

The FIT value targeted by reliability qualification, FIT_{target}, is a standard. Currently, processors are expected to have an MTTF of around 30 years[1] – this implies that FIT_{target} is around 4000.

Inherent in attaining this target FIT value is a cost-performance tradeoff. The cost-performance tradeoff that is made determines the proportionality constants in the individual failure model equations. These constants are technology and also depend on factors like design materials used and yield. High values for the proportionality constants imply more reliable processors, but higher cost.

For a specific set of proportionality constants, RAMP can provide an absolute FIT value for a given application. However, since we do not have the function that relates these constants to cost, we do the following: according to current worst-case based reliability qualification methodologies, the architectural parameters in the model equations (temperature (T), voltage (V), frequency (f), and activity factor (p)) are worst-case operating parameters. For a given technology, the target FIT value and the cost we are willing to pay for reliability qualification (i.e., the proportionality constants determined by materials, yield, etc.) determine these worst-case parameters. We therefore use these architecture level parameters as a proxy for the cost of reliability qualification. We call these proxy parameters as T_{qual}, V_{qual}, f_{qual}, and p_{qual} respectively. The higher the value of these parameters, the more expensive the reliability qualification. We then feed these parameters to RAMP to determine the proportionality constants that will give us the target FIT value for each failure mechanism for each structure. We use those constants to determine an absolute FIT value according to the actual T, V, f, and p seen by the workload.

In our experiments, to bound the space explored, we only vary T_{qual} to represent different costs. We fixed f_{qual} and V_{qual} to be the frequency and voltage of the base non-adaptive processor. We fixed p_{qual} to be the highest activity factor obtained across our application suite from our timing simulator. As described, the above methodology also requires setting a target FIT value for each structure for each failure mechanism. We assumed the target total failure rate of 4000 is distributed evenly across all four failure mechanisms and the failure rate for a given mechanism is distributed across different structures proportional to the area of the structure.

3.8 Validation of RAMP

Many aspects of RAMP have not yet been validated. However, most of our assumptions are grounded in "current practice," and were developed after extensive consultations with research and product groups that concentrate on processor reliability qualification at IBM. Further, the individual failure mechanism models, which are the key underlying components of RAMP, represent the state-of-the-art.

The approximations in RAMP involve architectural abstractions, and do not alter the physical failure models. Specifically, we make two key approximations. First, we assume that FIT values can be directly added across time and space. As explained in Section 3.5, adding failure rates across space is standard industry practice through the SOFR model; our extrapolation to adding FIT values over time is grounded in a similar underlying assumption. Second, we assume that the processor can be divided into discrete structures, and that the failures are evenly distributed across these structures. This assumption is analogous to that made in architecture-level power and thermal simulators due to the tradeoffs between simulation speed and accuracy. Power and thermal simulators assume uniform power and thermal profiles throughout structures, and do not differentiate finer features. Similarly, we do not differentiate individual features in a structure and instead use structure-wide abstractions like activity factor (for electromigration). We are continuing to refine our assumptions and approximations, and are currently validating different aspects of RAMP, also in collaboration with various groups at IBM.

[1] Processor MTTFs tend to be much higher than the expected lifetime of consumer use of the product. These values allow the product's consumer service life (typically 11 years) to fall far out in the tails of the lifetime distribution curve [1].

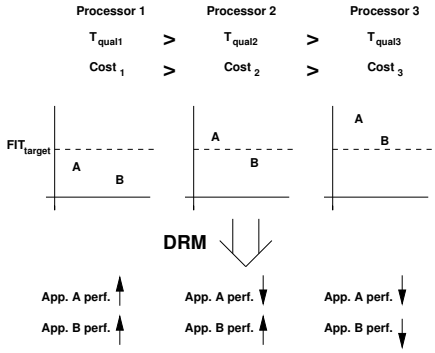

Figure 1. Dynamic Reliability Management (DRM). For different values of T_{qual}, FIT value of the processor running applications A and B is shown on the y-axis. DRM adapts the performance of the applications to meet the target FIT value.

4 Dynamic Reliability Management (DRM)

Figure 1 motivates Dynamic Reliability Management (DRM). Three processors 1, 2, and 3, are depicted. They have reliability design points, T_{qual_1}, T_{qual_2}, and T_{qual_3}, such that $T_{qual_1} > T_{qual_2} > T_{qual_3}$. This implies that processor 1 is more expensive to qualify for reliability than processor 2, and processor 3 is the cheapest to qualify. Consider two applications, A and B (depicted on the y-axis). These two applications will have different FIT values in the three processors, because the T_{qual} used to calculate the application's FIT value in each processor is different.

In processor 1, all applications meet the target FIT value, and in fact exceed it (i.e., their failure rates are lower than they are required to be). In processor 2, application A does not meet the target FIT value, but application B does. In processor 3, both applications do not meet the target FIT value. Hence, the expensive processor, 1, has been over-designed from a reliability perspective, while the cheaper processors, 2 and 3, have been under-designed.

Considering 2 and 3 first, although they are cheaper to design than 1, they can fail prematurely if no architectural intervention occurs, and so, do not represent acceptable design points by current reliability qualification methodologies. However, with DRM, we can design processors 2 and 3 to meet reliability targets by using processor adaptation to reduce processor temperature, utilization, voltage, and/or frequency, at the cost of throttling performance, but with higher reliability. Now, considering application B in processor 2 and both applications in processor 1, current systems will not exploit reliability over-design space. However, if the cooling solution can support it, DRM can be used to exploit the reliability margin and extract excess performance (e.g., by overclocking or increasing microarchitectural resources). Thus, DRM can be used both to decrease reliability qualification cost and to increase processor performance, while assuring reliability targets are met.

Like dynamic energy management (DEM) and dynamic thermal management (DTM), DRM requires adaptation response mechanisms and control algorithms to invoke these responses. We can leverage the extensive body of work on DEM and DTM [4, 15, 18] for DRM as well. However, it is important to note that designing for energy, temperature, and reliability are distinct problems. Solving one does not automatically solve the other. Like energy, but unlike temperature, reliability is a long-term phenomenon and can be budgeted over time. Similarly, like temperature, but unlike energy, reliability is directly affected by power density (a spatially local parameter). We highlight the differences between DRM and DTM in Section 7.3, and show that DRM violates thermal constraints in some situations, and DTM violates reliability constraints in some situations.

5 DRM evaluation

A true evaluation of DRM would require proposing a control algorithm for processor adaptations, and evaluating its performance for different values of T_{qual}. However, in this initial work, we only seek to show the potential of DRM, and do not study any actual control algorithms.

We study the potential of DRM by considering a wide range of microarchitectural configurations, and voltage and frequency settings, and selecting configurations that would give maximum performance, for different values of T_{qual}. This effectively simulates a DRM algorithm which 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 intra-application variability. The adaptations we explore for DRM are:

Microarchitectural adaptation (Arch). For every application, for a range of T_{qual} values, we explore a range of microarchitectural configurations, and select the one that gives the best performance while still within the target FIT value. The voltage and frequency is the same for all Arch configurations.

Dynamic voltage and frequency scaling (DVS). For every application, for a range of T_{qual} values, we explore a range of voltages and frequencies, and select the one which gives the best performance while still within the target FIT value. We use the most aggressive microarchitectural configuration supported.

Microarchitectural adaptation and DVS (ArchDVS). In this case, we explore combinations of microarchitectural configurations and DVS settings, for each application, for different T_{qual} values.

Technology Parameters	
Process technology	65 nm
V_{dd}	1.0 V
Processor frequency	4.0 GHz
Processor core size (not including L2 cache)	20.2mm^2 (4.5mm x 4.5 mm)
Leakage power density at 383K	0.5 W/mm^2
Base Processor Parameters	
Fetch/retire rate	8 per cycle
Functional units	6 Int, 4 FP, 2 Add. gen.
Integer FU latencies	1/7/12 add/multiply/divide
FP FU latencies	4 default, 12 div. (div. not pipelined)
Instruction window (reorder buffer)	128 entries
Register file size	192 integer and 192 FP
Memory queue size	32 entries
Branch prediction	2KB bimodal agree, 32 entry RAS
Base Memory Hierarchy Parameters	
L1 (Data)	64KB, 2-way associative, 64B line, 2 ports, 12 MSHRs
L1 (Instr)	32KB, 2-way associative
L2 (Unified)	1MB, 4-way associative, 64B line, 1 port, 12 MSHRs
Main Memory	16B/cycle, 4-way interleaved
Base Contentionless Memory Latencies	
L1 (Data) hit time (on-chip)	2 cycles
L2 hit time (off-chip)	20 cycles
Main memory (off-chip)	102 cycles

Table 1. Base non-adaptive processor.

The exact configurations used in each case are discussed in Section 6.1.

6 Experimental Methodology

6.1 Architectures

The base non-adaptive processor studied is summarized in Table 1. Given that reliability concerns will be amplified in future technologies, we model a 65nm processor, with a supply voltage, V_{dd}, of 1.0 V and a base frequency of 4 GHz. The core size (and size of different structures) was estimated from current processor sizes, scaled appropriately [3], and does not include the L2 cache. Also, although we model the performance impact of the L2 cache, we do not model its reliability. This is because the temperature of the L2 cache is much lower than the processor core [18], resulting in very low L2 intrinsic failure rates. Hence, we concentrate on intrinsic failures in the core. The base processor is similar to the MIPS R10000. We assume a centralized instruction window that integrates the issue queue and reorder buffer (ROB), but has a separate physical register file.

For the DRM voltage and frequency adaptations, we vary the processor frequency from 2.5 GHz to 5.0 GHz.We always set the voltage such that it supports the frequency being simulated. The relationship between voltage and frequency used was extrapolated from the information available for DVS on Intel's Pentium-M (Centrino) processor.

For the microarchitectural adaptations used in DRM, we model 18 configurations (consisting of combinations of the instruction window size, number of ALUs, and number of FPUs), ranging from a 128 entry instruction window, 6 ALU, 4 FPU processor, to a 16 entry instruction window, 2 ALU, 1 FPU processor. The issue width of the processor is equal to the sum of all active functional units and hence changes when we change the number of active functional units. Since we adapt the issue width of the processor with functional unit adaptation, we power down the selection logic corresponding to the functional units that are powered down. Also, when a functional unit is powered down, the corresponding part of the result bus, the wake-up ports to the instruction window, and write ports to the register file are also powered down. When a structure is powered down, since it has no current flow or supply voltage, it cannot have any failures due to electromigration or TDDB. Hence, the FIT value due to electromigration and TDDB of any adaptive structure on chip is proportional to the powered on area of the structure.

Finally, it should be noted that our base non-adaptive processor uses the most aggressive microarchitectural configuration available. Therefore, the microarchitectural adaptations we model can only reduce the performance of the processor, relative to base, and not increase it, since the DRM algorithm Arch cannot change processor frequency (Arch always runs at base voltage and frequency). Thus, the maximum possible performance of any application with Arch will be 1.0, relative to the base processor.

Application	Type	IPC	Base power (W)
MPGdec (Mpeg video decoder)	Multi-media	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

Table 2. Workload description. The IPC and power (dynamic+leakage) on the base non-adaptive processor is given.

6.2 Workload description

Table 2 summarizes the nine applications used in this paper. In order to study the reliability implications of various application classes, we choose three multimedia applications, three SPEC2000 integer applications, and three SPEC2000 floating point applications. As shown in Table 2, these applications exhibit a wide range of IPCs and power consumptions. For this study, it was more important to study applications which show a wide range of behavior, rather than perform a comprehensive study of the SPEC benchmark suite.

For the SPEC benchmarks, we fast forward 1.5 billion instructions to pass initialization code, and then we simulate 500 million instructions. The multimedia applications do not have an explicit initialization phase. So, we simulate 500 million instructions (at least 400 application frames) without fast forwarding.

6.3 Simulation methodology

Simulators. We use the RSIM simulator [10] for performance (timing) evaluation, the Wattch tool [7] integrated with RSIM for power measurement, and the HotSpot tool [18] for temperature evaluation. The chip floorplan fed to HotSpot resembles the MIPS R10000 floorplan (without L2 cache), scaled down to $20.2mm^2$ (4.5 mm x 4.5 mm). Wattch assumes extensive clock gating for all the components of the processor with 10% of its maximum power charged to a component when it is not accessed in a given cycle. Temperature and reliability measurements are performed at the granularity of 1μ second.

Leakage power. Leakage power is calculated based on structure areas. For the 65nm process modeled, a leakage power density of 0.5 W/mm^2 at 383K is used. This value was obtained from industry [6], and is based on aggressive leakage power control techniques being employed. We also model the impact of temperature on leakage power using the technique in [9]. At a temperature T, the leakage power, $P_{leak}(T) = P_{leak}(383K) * e^{\beta(T-383)}$ where β is a curve fitting constant. The value of β we use for 65nm (0.017) is taken from [9].

Heat sink temperature. As explained in [18], the RC time constant of the processor heat sink is significantly larger than the RC time constant of individual silicon structures. Hence, we cannot run our simulations long enough for the heat sink to reach its steady state temperature. Therefore, it is critical that HotSpot be initialized with the right heat sink temperature. For this purpose, we run all simulations twice. The first run is used to obtain average power consumption values for each structure on chip. These average values are then fed into the steady state temperature model to calculate a steady state heat sink temperature. This steady state heat sink temperature is then used to initialize the second simulation run.

Reliability calculation. Based on activity factor estimates obtained from RSIM, and temperature estimates obtained from HotSpot, RAMP calculates FIT values at 1 μsec intervals using the model in Section 3.

7 Results

7.1 Designing Processors for Different T_{qual}

Figure 2 shows the performance for all the applications, when using the combination of microarchitectural adaptation and DVS (ArchDVS) to control reliability by DRM for a range of T_{qual} values. Performance is represented as an increase or slowdown over the base non-adaptive processor, with a value of 1.0 representing no gain or loss. As mentioned in Section 3.7, we use T_{qual} as a proxy for reliability design cost. Results are shown for four values of T_{qual} – 400K, 370K, 345K, and 325K, which represent four qualification levels, ranging from most expensive to cheapest.

$\boldsymbol{T_{qual} = 400K}$. The hottest temperature reached on chip by any application for our benchmark suite was near 400K. Hence, this value of T_{qual} represents a lower bound on the qualification temperature that would be chosen using current methodology for reliability qualification, based on worst-case conditions. As can be seen, all the applications experience significant performance gains (ranging from a gain of 10% for MP3dec to 19% for twolf) while still maintaining required processor reliability levels. This is because the operating conditions on chip while an application runs 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 ocassionally exceed 400K but the total FIT value will not exceed the target because higher instantaneous FIT values are compensated by lower values at other times.

The performance gains experienced by the SPEC benchmarks tend to be higher on average than those of the multimedia benchmarks. This is because the multimedia benchmarks have higher IPCs, and consequently higher operating temperatures and activity factors, which gives them higher FIT values on the base processor than the SPEC benchmarks.

Based on the above results, we can see that qualifying for worst case operating conditions is overly conservative – instead, we could either design to a lower T_{qual}, which would result in cost savings, or the base non-adaptive processor can be marketed at a higher frequency (while still meeting the reliability target).

$\boldsymbol{T_{qual} = 370K}$. At a T_{qual} value of 370K, the applications with the highest FIT values on the base non-adaptive processor (MP3dec and MPGdec) have almost no performance gain. All the other applications have a performance gain ranging from 4% for bzip2 to 13% for twolf. This represents a processor which is qualified for reliability based on application behavior. Rather than selecting T_{qual} based on the worst case application operating temperature of 400K, T_{qual} was chosen such that the worst applications (MP3dec and MPGdec) just meet the reliability target. Such an *application oriented* approach to reliability qualification represents significant savings in qualification cost without any loss of performance (DRM never curtails performance in this scenario for these applications). Again, lower IPC applications see the largest performance gains (twolf and art).

$\boldsymbol{T_{qual} = 345K}$. A T_{qual} value of 345K represents a processor qualified for the average application, rather than worst case application. As can be seen, the performance seen by all the applications with DRM was within 10% of the base value, and in four cases, was within 5%. This potentially represents an excellent cost-performance tradeoff design point, where DRM can be used to underdesign a processor, without incurring significant performance penalties. As is expected, high IPC applications experience the largest

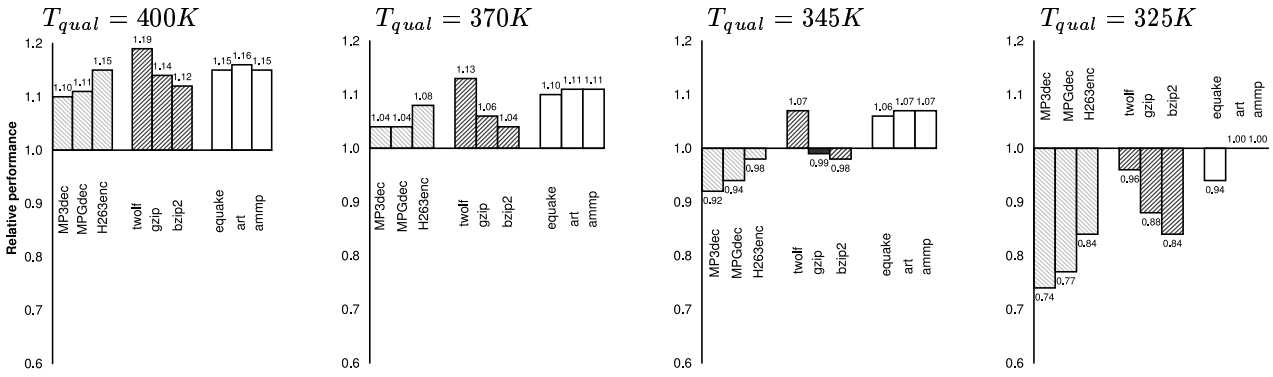

Figure 2. The performance of ArchDVS for DRM is shown on the y-axis, relative to the base non-adaptive architecture at 4 GHz. This is shown for all the applications for different T_{qual} values.

performance losses, while low IPC applications enjoy the largest gains.

$\boldsymbol{T_{qual} = 325K}$. A T_{qual} value of 325 K represents a processor which has been drastically underdesigned from a reliability perspective. All applications, with the exception of art and ammp, experience a slowdown. The high IPC multimedia applications experience the largest slowdown, with MP3dec suffering a loss of 26% in performance. This scenario potentially represents a case where the cost benefit of designing for a cheaper T_{qual} is overshadowed by the loss in performance seen.

Implications of designing for reliability. From the above results, we see that there is potential for significant cost benefit, without any performance loss, using DRM. Changing the reliability design point from T_{qual} of 400K to 370K, saves design cost, without requiring any of the applications to slow down. This shows that worst case reliability qualification is overly conservative.

Using DRM, by allowing some performance degradation, we can further lower the value of T_{qual}. In our results, even at a T_{qual} of 345K, the performance loss seen was limited. Hence, a wide spectrum of T_{qual} values (in our case, 345K to 400K) are available to designers, for a reasonable performance tradeoff.

Finally, we see that the performance-cost tradeoff depends on the processor's intended application domain. For example, a processor designed for SPEC applications could be designed to a lower T_{qual}, than a processor intended for multimedia applications. In the situation that an application causes the processor to exceed the reliability target, DRM can be employed to maintain reliability.

7.2 Comparing Different DRM Adaptations

Figure 3 compares the performance (relative to the base non-adaptive processor) for the three DRM adaptations, Arch, DVS, and ArchDVS, for one of the applications, bzip2, for a range of T_{qual} values. Due to a lack of space, we do not show these results for all the other applications. However, the trends are very similar.

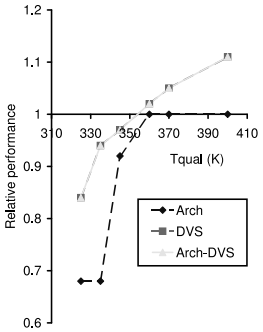

Figure 3. Comparison of different DRM adaptations for bzip2. The x-axis represents different T_{qual} values and the y-axis is performance speedup or slowdown. DVS and ArchDVS show identical behavior for bzip2 and are not distinguishable.

As can be seen, DVS and ArchDVS significantly outperform Arch, performing 25% better at a T_{qual} value of 335K. Also, ArchDVS chose to perform DVS on the base processor most of the time – hence, there is very little difference between DVS and ArchDVS.

DVS and ArchDVS, which can both adapt frequency and voltage, perform much better than Arch due to three main reasons: (1) Small drops in voltage and frequency result in large drops in temperature – this is because of the near cubic relationship between frequency and power, which translates into a large temperature drop. In comparison, Arch does not cause such a large drop in processor power, necessitating a larger performance hit. (2) As can be seen in Equation 4, there is a very large voltage dependence on TDDB FIT values. Hence, small drops in voltage and frequency reduce the TDDB FIT value drastically. (3) As mentioned, in Sec-

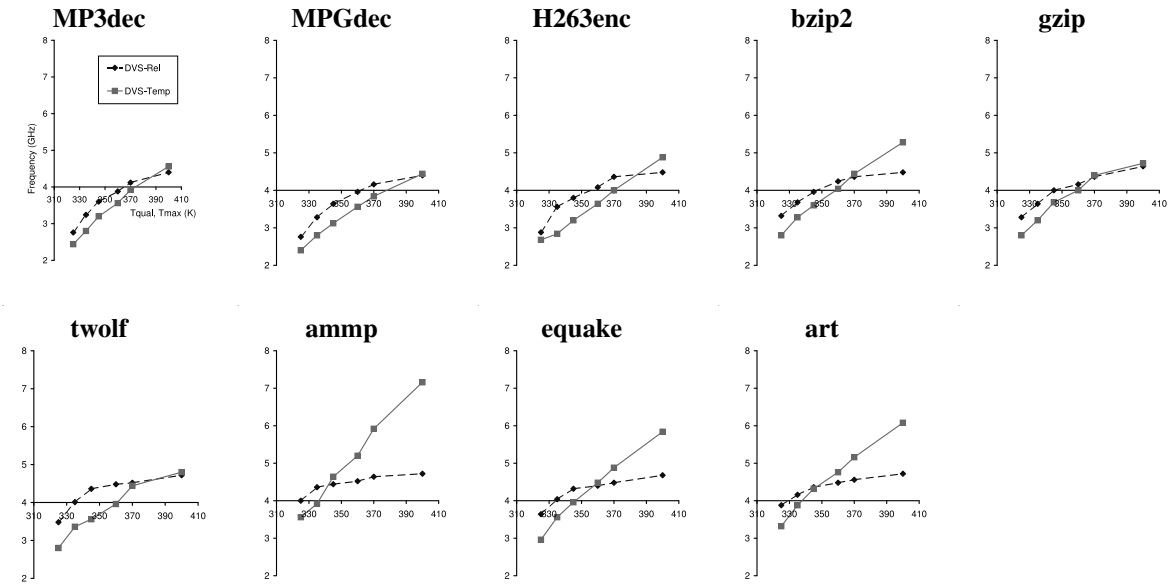

Figure 4. Comparing design for reliability and temperature. Temperatures on the x-axis represent T_{qual} for DRM and T_{max} for DTM. The frequency, in GHz, chosen by DVS for DRM (dotted line) and DTM (solid line) is shown on the y-axis. We evaluated values at 325K, 335K, 345K, 360K, 370K, and 400K.

tion 6.1, the performance due to Arch can never be greater than 1.0, since it cannot adapt the processor's frequency. Hence, in any scenario where processor performance can be increased because of an overdesigned T_{qual}, DVS and ArchDVS will perform better than Arch (this is seen for T_{qual} values between 360K and 400K in Figure 3).

Hence, it is clear that DVS is more beneficial than the microarchitectural adaptations for the space we explored for DRM.

7.3 Comparing DRM and DTM

This section makes the case that DTM algorithms do not subsume reliability concerns and vice versa; i.e., both thermal and reliability constraints need to be considered as first-class design entities.

Figure 4 compares DRM and DTM using voltage and frequency scaling (DVS) for all the applications. Each point on the horizontal axis is a temperature value, which represents the qualifying temperature for DRM (T_{qual}), and the thermal design point (T_{max}), for DTM. For each of these temperatures, the optimal frequency chosen by DVS on the base non-adaptive processor for DRM (Curve DVS-Rel in the figure) and DTM (curve DVS-Temp) is shown on the vertical axis. That is, the DVS-Rel curve ensures highest performance at the target FIT value and the DVS-Temp curve ensures highest performance without exceeding T_{max}.

Unlike T_{max} which represents the maximum temperature the processor is allowed to reach, T_{qual} does not impose a temperature restriction on the processor. The temperature can exceed T_{qual} as long as the target FIT value is not exceeded (because FIT value is also a function of voltage, frequency and utilization).

As can be seen, different frequencies are suggested by DRM and DTM. More significantly, at higher values of T_{qual} and T_{max}, using the DTM suggested frequency would violate the system reliability requirement; while at lower values of T_{qual} and T_{max}, using the DRM suggested frequency would violate the system thermal requirement. This occurs because the slope of the DVS-Temp curve in the figure is generally steeper than the slope of the DVS-Rel curve. The reliability curve is less steep because of the exponential dependence of reliability on temperature. A small change in frequency creates a temperature change which is amplified exponentially in the reliability equation. This effect is further compounded by the large dependence of the TDDB FIT value on DVS voltage, as dictated by Equation 4. Finally, we can also see that the crossover point of the two curves is not fixed, and instead changes depending on the application. Hence, it is clear that the relationship between design for reliability and design for temperature is not obvious. Neither subsumes the other, and algorithms that jointly consider reliability and temperature (and energy) are important areas of future work.

8 Conclusions

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. The effects of technology scaling in the deep sub-micron range are adding to this reliability concern. In this paper, we

present a new model, called RAMP, for enabling early-stage power/performance/reliability tradeoffs. Driven by technology, packaging, and chip floorplan parameters, this model can be used in conjunction with an existing cycle-accurate microarchitectural simulator to estimate variations in MTTF with the characteristics of the input workload. By using this model, we show how one can boost delivered performance in low-IPC phases, without deviating from the original MTTF specification. Similarly, we also show how processors could be designed with less-than-worst-case temperature qualifications to conserve cost, without giving up performance. In this latter case, the implications of adaptive control mechanisms are similar to earlier published dynamic energy management and thermal management methods. However, our experimental results clearly show that the tradeoffs involved in our new DRM methodology are significantly (if not fundamentally) different from those reported in prior adaptive work.

This paper deals primarily with the impact of temperature on *wearout* driven (un)reliability. It also considers the effect of degradations in reliability caused by technology scaling alone, such as dielectric breakdown. There are, of course, other lifetime reliability degradations that we do not currently consider in RAMP. One example is the effect of inductive noise on the voltage rails (Ldi/dt) caused by current surges in various units.

In future work, we will propose specific adaptive control algorithms (including intra-application and inter-application control) that offer the promise of close to optimal choice of adaptive configurations to increase reliability while meeting a performance target or to increase performance while meeting a reliability specification. We will extend the RAMP model to include other technology-dependent reliability degradation factors, besides the ones described in this paper. We also plan to incorporate time dependence in our reliability models and relax the series failure assumption.

9 Acknowledgments

We would like to thank Chao-Kun Hu, Barry Linder, and Ernest Wu of IBM for their help with the electromigration and TDDB models.

References

[1] Reliability in CMOS IC Design: Physical Failure Mechanisms and their Modeling. In *MOSIS Technical Notes, http://www.mosis.org/support/technical-notes.html*.

[2] Failure Mechanisms and Models for Semiconductor Devices. In *JEDEC Publication JEP122-A*, 2002.

[3] Critical Reliability Challenges for The International Technology Roadmap for Semiconductors. In *Intl. Sematech Tech. Transfer 03024377A-TR*, 2003.

[4] D. H. Albonesi et al. Dynamically Tuning Processor Resources with Adaptive Processing. In *IEEE Computer*, 2003.

[5] T. M. Austin. DIVA: A Reliable Substrate for Deep Submicron Microarchitecture Design. In *Proc. of the 32nd Annual Intl. Symp. on Microarchitecture*, 1998.

[6] P. Bose. Power-Efficient Microarchitectural Choices at the Early Design Stage. In *Keynote Address, Workshop on Power-Aware Computer Systems*, 2003.

[7] D. Brooks et al. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *Proc. of the 27th Annual Intl. Symp. on Comp. Arch.*, 2000.

[8] A. Dasgupta et al. Electromigration Reliability Enhancement Via Bus Activity Distribution. In *Design Automation Conference*, 1996.

[9] S. Heo et al. Reducing Power Density Through Activity Migration. In *Intl. Symp. on Low Power Elec. Design*, 2003.

[10] C. J. Hughes et al. RSIM: Simulating Shared-Memory Multiprocessors with ILP Processors. *IEEE Computer*, Feb. 2002.

[11] S. S. Mukherjee et al. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor. In *Proc. of the 36th Intl. Symp. on Microarch.*, 2003.

[12] D. Patterson et al. Recovery-Oriented Computing (ROC): Motivation, Definition, Techniques, and Case Studies. In *UC Berkeley CS Tech. Report UCB//SD-02-1175*, 2002.

[13] M. G. Pecht et al. Guidebook for Managing Silicon Chip Reliabilty. CRC Press, 1999.

[14] E. Rotenberg. AR/SMT: A Microarchitectural Approach to Fault Tolerance in Microprocessors. In *International Symposium on Fault Tolerant Computing*, 1998.

[15] R. Sasanka et al. Joint Local and Global Hardware Adaptations for Energy. In *Proc. of the 10th Intl. Conf. on Arch. Support for Prog. Langs. and Operating Sys.*, 2002.

[16] K. Seshan et al. The Quality and Reliability of Intel's Quarter Micron Process. In *Intel Technology Journal*, Q3,1998.

[17] P. Shivakumar et al. Exploiting Microarchitectural Redundancy for Defect Tolerance. In *21st Intl. Conf. on Comp. Design*, 2003.

[18] K. Skadron et al. Temperature-Aware Microarchitecture. In *Proc. of the 30th Annual Intl. Symp. on Comp. Arch.*, 2003.

[19] L. Spainhower et al. IBM s/390 Parallel Enterprise Server G5 Fault Tolerance: A Historical Perspective. In *IBM Journal of R&D*, September/November 1999.

[20] J. Srinivasan et al. The Impact of Scaling on Processor Lifetime Reliability. In *Proc. of the Intl. Conf. on Dependable Systems and Networks*, 2004.

[21] J. H. Stathis. Reliability Limits for the Gate Insulator in CMOS Technology. In *IBM Journal of R&D, Vol. 46*, 2002.

[22] K. Trivedi. Probability and Statistics with Reliability, Queueing, and Computer Science Applications. Prentice Hall, 1982.

[23] N. J. Wang et al. Characterizing the Effects of Transient Faults on a High-Performance Processor Pipeline. In *Proc. of the Intl. Conf. on Dependable Systems and Networks*, 2004.

[24] E. Y. Wu et al. Interplay of Voltage and Temperature Acceleration of Oxide Breakdown for Ultra-Thin Gate Dioxides. In *Solid-state Electronics Journal*, 2002.