

IBM Research Report

The Case for Microarchitectural Awareness of Lifetime Reliability

Jayanth Srinivasan, Sarita V. Adve
Department of Computer Science
University of Illinois at Urbana-Champaign

Pradip Bose, Jude Rivers
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598



Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

The Case for Microarchitectural Awareness of Lifetime Reliability

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

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

Submitted for publication. Please do not distribute.

Abstract

Ensuring long processor lifetimes by limiting failures due to hard errors is a critical requirement for all micro-processor manufacturers. Current methodologies for qualifying long-term lifetime reliability are overly conservative since they seek to maintain reliability for peak usage of the processor. 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. We use state-of-the-art models for different wear-out mechanisms and apply them to calculate failure rates of individual architectural structures. These failure rates are a function of temperature, switching activity, and voltage. RAMP is coupled with a conventional performance and power simulator to track these parameters over an application run.

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, dynamic voltage scaling is an effective response technique for DRM, and dynamic thermal management neither subsumes nor is subsumed by DRM.

1 Introduction

Ensuring long-term reliability by reducing early lifetime failures due to hard errors is a critical requirement for all microprocessor manufacturers. However, in order to achieve

competitive processor performance targets, semiconductor design and manufacturing is undergoing changes that will threaten the nearly unlimited lifetime and high long-term reliability standards that customers have come to expect. This has led the ITRS to predict the onset of significant reliability problems in the future [5], 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 the device and process engineers. However, the microarchitecture’s ability to track application behavior can potentially be leveraged to the benefit of reliability qualification, by decreasing reliability design costs, and by increasing processor yield. This would be particularly beneficial in situations where conscious tradeoffs between performance, cost, and reliability are made. This motivates the need for microarchitectural awareness of lifetime reliability, and also precipitates a requirement for tools and models to evaluate lifetime reliability at early processor design stages.

This paper represents a first attempt at evaluating the potential benefit of microarchitectural intervention in lifetime reliability. Using industrial strength models for lifetime failure modes, we develop a simulation methodology, called RAMP, to estimate lifetime reliability from an architectural perspective. We also propose a new technique, Dynamic Reliability Management (DRM), which can use microarchitectural adaptations to enhance lifetime reliability.

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 electrical noise or external radiation, rather than design or manufacturing related defects. Extensive research is being performed by the architecture community to make proces-

sors resilient to soft errors [25]. Although soft errors can cause errors in computation and corruption to data, 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. Given that hard errors result in permanent processor failure, processor lifetime is inversely proportional to the hard error rate. Since hard errors lead to processor failure, we will henceforth refer to them as hard failures.

Hard failures can be further divided into extrinsic failures and intrinsic failures [22].

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 [6]. Other examples of extrinsic failures 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 [20, 24], 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 [20, 24].

Intrinsic failures are those related to processor wear-out, and are caused during operation within the specified processor use 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, fabrication, and use, all hard processor failures would be due to intrinsic failures. Intrinsic failures occur with an increasing rate over time and are usually caused by inherent defects in the processor material. It is essential that these fails do not occur during the intended useful lifetime of the device when it is used under specified operating conditions [1, 6]. 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.¹ 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.

¹Some early life failures can be intrinsic. Burn-in should detect these failures also.

1.2 Main Challenges to Maintaining Lifetime Reliability

Although providing significant benefits in microprocessor performance, advances in technology are accelerating the onset of reliability problems due to intrinsic failures and are causing a resultant reduction in processor lifetimes. Specifically, the four main challenges to maintaining lifetime reliability are:

- **Increase in processor power densities and resultant processor temperatures:** Increasing power consumption and increasing transistor densities are causing higher power densities and temperatures on chip - this causes problems as processor wear-out mechanisms tend to accelerate exponentially with temperature.
- **Scaling of transistors for performance:** Scaling to smaller transistors increases failure rates by shrinking the thickness of dielectrics (both gate dielectrics and inter layer dielectrics (ILD)). Scaling also leads to higher current densities in the interconnects causing accelerated interconnect wear-out. Scaled down transistors in deep submicron CMOS technologies also have significantly higher leakage power which has an exponential dependence on temperature - this can create reliability problems like thermal runaway [14]. Supply voltage and threshold voltage are not scaling appropriately with technology because of performance and leakage power concerns creating further reliability problems. Finally, device miniaturization is also eating away at process and design margins.
- **Increase in number of transistors in a processor:** New features and increasing functionality, facilitated by increasing transistor densities, cause 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 costs, 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. 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.

These lifetime reliability challenges have an integral impact on processor yield, design cost and design time.

1.3 Microarchitectural awareness of intrinsic failures

Reliability has to be treated as a first class design constraint, necessitating reliability analysis 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 will impact the number of processors shipped (yield) and resultant profit.

Extensive research has gone into techniques that can minimize energy and maximize thermal efficiency by exploiting architectural features and adaptation capabilities. A similar approach can be used for lifetime reliability - the microarchitecture's 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, manufacture and chip test. Current reliability qualification mechanisms are oblivious to application behavior and are based on worst-case or peak temperatures and utilization estimates.

Due to variations in IPC, power, and temperature, different applications have different effects on the processor's lifetime reliability. The micro-architecture's awareness of application behavior can be used in two ways:

- Current reliability qualification is based on worst case temperatures and utilization - however, most applications will run at lower temperatures and utilizations resulting in higher reliabilities and longer processor lifetimes than required. This excess reliability can be utilized by the microarchitecture to increase application performance - in other words, the microarchitecture can use adaptations to increase application performance while still maintaining system reliability goals and target lifetimes. 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.
- In contrast, microarchitectural adaptation can also be used to handle processors which have been under-designed from a reliability perspective. In an approach similar to Dynamic Thermal Management (DTM), the processor reliability qualification can be based on expected processor utilizations rather than worst case values. This would result in significant design cost reductions and would provide higher processor yield, increasing profit. In situations where applications exceed the reliability design limit, micro-architectural adaptation can be used to maintain system reliability targets. However, just like DTM, these adaptations would result in a performance hit. Such an approach would be beneficial to commodity processors where increasing yield and reducing cooling solution costs would have significant impact on profits, even if they come at the cost of some performance loss.

The performance and cost benefits of the above techniques will be amplified with transistor scaling.

1.4 Contributions

We make two sets of contributions. First, we introduce the first architecture-level methodology for evaluating processor lifetime reliability, using state-of-the-art analytic models for important intrinsic failure mechanisms. We present an implementation of this methodology, called RAMP (**R**eliability **A**ware **M**icro**p**rocessor), which can be added to microarchitectural simulators to obtain high-level lifetime reliability estimates based on intrinsic failure rates.

Second, we propose *dynamic reliability management* - 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. This saves cost, but possibly at a loss of some performance in the unexpected case. Conversely, DRM allows processors that are over-designed for reliability for some applications to respond by improving performance for these applications. 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 microarchitectural awareness of reliability has concentrated on soft errors. 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. A good bibliography of research targeted at soft errors in memory and an introduction to soft errors in combinational logic can be found in [25].

Current techniques for enhancing hard failure reliability focus on fault-tolerant computing methods like redundancy [28] and efficient failure recovery methods [21]. However, these techniques are typically used in server class processors and the redundancy is targeted at minimizing down-time. There has also been work on detection and recovery from errors that occur during program execution [7, 23]. Recent work by Shivakumar et al. examines techniques to increase processor manufacturing yield by exploiting micro-architectural redundancy [26]. 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 long-term reliability of processors. Fault-tolerant microarchitectures like DIVA [7, 23] can recover from hard failures in the main processing core, but at a huge performance cost. In other words, such a scheme does increase the effective mean time to *total* failure of the system, but the intended use of the scheme is to recover from a hard failure by replacement of a failed part, or to recover from a soft error. Fully duplicated, dual-core solutions [28] have a similar function and goal; but here the area cost is very large, while the performance is unaffected after recovery from a hard failure sustained by one of the redundant cores.

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

RAMP uses state-of-the-art models taken from the device design and manufacturing community and attempts to create a unified model for analyzing intrinsic failures from a system-wide perspective, and applies it at the architecture level.

3 Intrinsic Failure Mechanism Models and Implementation of RAMP

Sections 3.1 to 3.4 discuss the four main wear-out intrinsic failure mechanisms experienced by processors, the state-of-the-art analytical models for the mechanisms, and their implementation in RAMP. Currently, the main intrinsic failure mechanisms faced by processors are: Electromigration (EM), Stress Migration (SM), Gate-oxide breakdown or Time Dependent Dielectric Breakdown (TDDB), and thermal cycling (TC) [5, 4]. RAMP implements the failure models at a microarchitectural structure level.

Section 3.5 discusses the integration of the different failure mechanism models over the different structures into one system model. Section 3.6 describes the process of reliability qualification and its use in RAMP, and Section 3.7 discusses the use of this model for real applications.

The standard reliability metric used in the analytical models is MTTF, which is the average expected lifetime of the processor.

3.1 Electromigration

Electromigration is one of the best studied and well understood failure mechanisms in semiconductor devices and occurs in interconnects. Extensive research has been performed by the material science and semiconductor community on modeling and understanding the effects of electromigration [18, 4, 10, 9].

Electromigration in aluminum and copper interconnects is due to the mass transport of conductor metal atoms in the interconnects due to momentum transferred by the electron current. Conducting electrons transfer some of their momentum to the metal atoms of the interconnect – this "elec-

tron 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. This can lead to the formation and growth of voids at sites of depletion leading to open circuits, increased interconnect resistance, and other problems. At the site of metal atom pile up, extrusions can form causing shorts between adjacent metal lines causing circuit failure.

3.1.1 Model

The currently accepted model for MTTF due to electromigration ($MTTF_{EM}$) is based on Black's original electromigration equation [18], and is as follows [4, 10]:

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

where A_{EM} is an empirically determined constant, J is the current density in the interconnect, J_{crit} is the critical current density required for electromigration, E_{aEM} is the activation energy for electromigration, k is Boltzmann's constant, and T is absolute temperature in Kelvin. n and E_{aEM} are constants that depend on the interconnect metal used. J tends to be much higher than J_{crit} in interconnects (nearly 2 orders of magnitude [2]). Hence, $(J - J_{crit}) \approx J$.

The current density, J , of a line can be related to the switching probability of the line, p , as [12]

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

where C , W , and H are the capacitance, width, and thickness, respectively of the line and f is the clock frequency. C , W , and H can be changed to examine the impact of process scaling on $MTTF_{EM}$.

3.1.2 Implementation in RAMP

RAMP currently assumes all interconnects in a structure to be similar, and does not differentiate interconnects on the basis of their C , W , and H . The activity factor of a structure, p , is obtained from the timing simulator.

RAMP assumes copper interconnects and uses a value of 1.1 for n and 0.9 for E_a [3, 4] for electromigration.

3.2 Stress Migration

Stress migration is very similar to electromigration. Stress migration is a phenomenon where the metal atoms in the interconnects migrate due to mechanical stress. Stress migration is caused by thermo-mechanical stresses which are caused by differing thermal expansion rates of different materials in the device [3, 4]. The exact mechanisms behind stress migration are still not completely understood and research is ongoing on the subject.

3.2.1 Model

As mentioned, stress migration is caused by materials with different expansion rates, whose stress, σ , is proportional

to the change in temperature. The change in temperature is measured with respect to the stress free temperature of the metal. The stress free temperature is the metal deposition temperature – in other words, when the metal was originally deposited on the device, there were no thermal stresses. However, at any temperature different from the metal deposition temperature, there are thermo-mechanical stresses. The mean time to failure due to stress migration, $MTTF_{SM}$, is given by [4]:

$$MTTF_{SM} = A_{SM} \sigma^{-n} e^{\frac{E_{a,SM}}{kT}} \quad (3)$$

where σ is the mechanical stress caused due to differing expansion rates, A_{SM} is an empirically determined constant, T is the absolute temperature in Kelvin, and n and E_a are material dependent constants.

As mentioned previously, the mechanical stress, σ , is proportional to the change in temperature from the stress free temperature of the metal, i.e., $\sigma \propto |T_0 - T|$ where T_0 is the stress free temperature of the metal, and T is the operating temperature. Abstracting out only the architectural parameters from Equation 3, the MTTF due to stress migration as modeled is given by:

$$MTTF_{SM} \propto |T_0 - T|^{-n} e^{\frac{E_a}{kT}} \quad (4)$$

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.

3.2.2 Implementation in RAMP

RAMP assumes copper interconnects and uses a value of 2.5 for n and 0.9 for E_a for stress migration. RAMP assumes that sputtering (versus vapor deposition) was used to deposit the interconnect metal and uses a value of 500K for T_0 [13]. Also, as mentioned previously, RAMP tracks stress migration failures at the granularity of a microarchitectural structure.

3.3 Time-dependent dielectric breakdown (TDDB)

Time-dependent dielectric breakdown, 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. When a conducting path forms between the gate and the substrate, it is no longer possible to control current flow between the drain and the source with a gate electric field, effectively rendering the transistor device useless [6, 11, 17, 31].

In the past 10 years, due to the advent of thin and ultra-thin gate oxides, intrinsic gate oxide failure is becoming increasingly important. The failure rate is also increasing due to the fact that the supply voltage is not scaling down appropriately with technology [11].

3.3.1 Model

The TDDB model we use is based on recent experimental work done by Wu et al. [31] 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 [31], the MTTF due to TDDB, $MTTF_{TDDB}$, at a temperature, T , and a voltage, V , is given by:

$$MTTF_{TDDB} = A_{TDDB} V^{(a-bT)} e^{\frac{(X+Y+ZT)}{kT}} \quad (5)$$

where A_{TDDB} is an empirically determined constant, and a , b , X , Y , and Z are fitting parameters.

3.3.2 Implementation in RAMP

Based on the experimental data collected by Wu et al. [31], the values used in RAMP for the TDDB model are $a = 78$, $b = -0.081$, $X = 0.759ev$, $Y = -66.8evK$, and $Z = -8.37e - 4ev/K$.

3.4 Temperature Cycling

Fatigue failures can occur due to temperature cycling. Permanent damage accumulates every time there is a cycle in temperature eventually leading to failure. Normal powering up and powering down will also cause damage. Although all parts of the device experience fatigue due to thermal cycling, the effect is most pronounced in the package and die interface (for example, solder joints) [4, 16].

3.4.1 Model

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). For example, 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).

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, which has been found to model thermal cycling effects on semiconductors well [16]. The Coffin-Manson equation for thermal cycling is [4]:

$$N_f = C_0(\Delta T)^{-q} \quad (6)$$

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 6, 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:

$$MTTF_{TC} \propto A_{TC} \left(\frac{1}{T_{average} - T_{ambient}} \right)^{-q} \quad (7)$$

where A_{TC} is an empirically determined constant (which also factors in the frequency of thermal cycling, which we assume stays constant), and $T_{average} - T_{ambient}$ is the actual average thermal cycle a structure on chip experiences.

3.4.2 Implementation in RAMP

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 [4, 16].

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 would cause 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 probability 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 $h(t)$ will remain fixed, and will 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 (the probability that a component will fail will increase, the older the component gets).

The above two assumptions imply [30]: (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,

$$MTTF_p = \frac{1}{\lambda_p} = \frac{1}{\sum_{i=1}^j \sum_{l=1}^k \lambda_{il}} \quad (8)$$

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 in Failures in Time (FITs) [19], which is the number of failures seen per 10^9 device hours – $MTTF_p = \frac{1}{\lambda_p} = \frac{10^9}{FIT_{value}}$. From this point on, we always 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 processor reliability. The time distribution of processor reliability is also important. Incorporating time dependent failure models is an important area of future work.

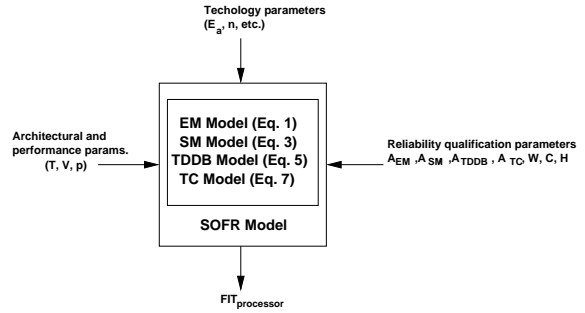


Figure 1. Summary of the Reliability Methodology

3.6 The Reliability Qualification Process: Determining a FIT value

The FIT value targeted by reliability qualification, FIT_{target} , is a standard. Currently, processors are expected to have an MTTF of around 30 years – this implies that FIT_{target} is around 4000. Inherent in attaining this target FIT value is a cost performance tradeoff.

Figure 1 summarizes the reliability model, and shows the parameters that are used as inputs to obtain a FIT value. There are three kinds of input parameters: (1) technology parameters, that depend on the process technology and materials used in the design of the processor; (2) microarchitectural and performance related parameters; and (3) reliability qualification parameters, that depend on the cost of reliability qualification. For a given process technology, and for a given FIT_{target} , the cost we are willing to pay for reliability qualification will fix the values of the architectural and performance parameters – temperature (T), voltage (V) and activity factor (p). Thus, associated with a reliability qualification point is a set of operating parameters. The more aggressive (i.e. high performance) these parameters, the more the cost of reliability qualification. We call these parameters, T_{qual} , V_{qual} , and p_{qual} . The current reliability

qualification methodology requires that these be worst-case (peak) values, even if an application will never see the worst case. Since we do not have the function that relates the qualification operating points to cost, we will use T_{qual} , V_{qual} , and p_{qual} as proxies for reliability cost.

3.6.1 Reliability Qualification in RAMP

RAMP takes as input, the values of T_{qual} , V_{qual} , and p_{qual} . For a given process technology, and FIT_{target} , it calculates the values of the reliability qualification parameters. Then, for any architectural parameters input, RAMP generates the actual FIT value.

3.7 Measuring FIT Value as a Function of an Application

As shown in Figure 1, Equations 1, 3, 5, and 7, provide FIT estimates for fixed operating conditions - however, when an application is simulated, the temperature, activity factor, and voltage data (in processors with dynamic voltage scaling) all vary continuously with time.

We assume that the impact of this variation can accounted for by: (1) calculating an instantaneous FIT value based on instantaneous T , V , and p (measured over a reasonably small time granularity); and (2) using an average of these FIT values to determine the actual FIT value of the processor when running the application. Sometime we refer to this as the FIT rate of the application. Our assumption of averaging over time is similar to the assumption used in the SOFR model.

To determine the FIT value of a processor for a workload, we use a weighted average of the FIT values obtained for each of the workload’s applications.

Through its close coupling with a timing and power simulator, RAMP obtains instantaneous values for T , V , and p , and it uses the reliability model summarized in Figure 1 to generate FIT values for the processor running an application or a workload.

4 Dynamic Reliability Management (DRM)

Figure 2 illustrates the case for microarchitectural involvement in reliability qualification. Three processors 1, 2, and 3 are depicted. They have reliability design points, T_{qual_1} , T_{qual_2} , and T_{qual_3} ², 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. The vertical axis in the graphs in Figure 2 represents FIT values - 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 rate, and in fact exceed it (i.e., their failure rates are lower than they are required to be). In processor 2, application A does

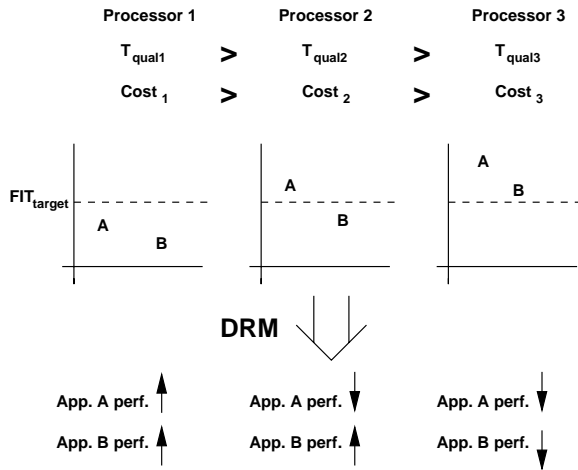


Figure 2. Dynamic Reliability Management(DRM). For different values of T_{qual} , FIT value of the processor running applications A and B is shown on y-axis. DRM adapts perf. of apps. to meet FIT_{target} .

not meet the target FIT rate, but application B does. In processor 3, both applications do not meet the target FIT rate.

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 Dynamic Reliability Management(DRM), we can design processors 2 and 3 to meet reliability targets.

Now, considering processor 1, current systems will not exploit over-design space. However, using DRM, processors can adapt to exploit the reliability margin and extract excess performance. Thus, DRM can be used, both to decrease reliability qualification cost, and to increase processor performance, while assuring reliability targets are met.

4.1 Comparison of DRM and DTM

Although the goals motivating DRM are very similar to DTM, it is important to note that designing for temperature and designing for reliability are distinct problems - solving one does not automatically solve the other. We highlight the differences between the two 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

Many different predictive and reactive control algorithms have been proposed to control processor adaptations for DTM [27, 29]. These control algorithms seek to maximize DTM performance for a range of thermal design points.

²We vary p_{qual} proportional to T_{qual} , and we keep V_{qual} constant.

Similarly, 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 paper in this area, 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 architectural 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 do any sort of reactive adaption to exploit local variability in the application run.

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 that gives the best performance while still within FIT_{target} . The 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 which give the best performance while still within the FIT_{target} . This is run on the most aggressive architecture supported.
- **Microarchitectural adaptation and DVS (ArchDVS):** In this case, we explore combinations of architectural configurations and DVS settings, for each application, for different T_{qual} values.

The exact configurations used in each case are discussed in the methodology section, 6.1, and results are discussed in Section 7.1.

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, and does not include the L2 cache. We do not model the reliability of the L2 cache because we did not have a reasonable leakage power model for the L2 cache. We do however simulate the L2 cache behavior on the performance simulator.

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.

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 (pipelined)
FP FU latencies	4 default, 12 div. (all but div. pipelined)
Instruction window (reorder buffer) size	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 Contentless 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.

For the DRM voltage and frequency adaptations, we vary the processor frequency from 2.8GHz to 4.4GHz. 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 architectural adaptations used in DRM, we model 18 architectural 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 can not have any failures due to electromigration or TDDDB. Hence, the FIT value due to electromigration and TDDDB of any adaptive structure on chip is proportional to the powered on area of the structure. The leakage power consumption of adaptive structures on chip is also proportional to the powered on area of the structure (Section 6.3.2 describes leakage power modeling methodology).

Finally, it should be noted that our base nonadaptive processor uses the most aggressive architectural configuration available. The architectural adaptations we model can only reduce the complexity of the processor, relative to base, and not increase it. Also, Arch can not change processor frequency. As a result, the maximum possible performance of any application with DRM algorithm Arch will be 1.0, where it will be running at the base configuration at the base frequency. On the other hand, DRM algorithms DVS and

Application	Type	IPC	Base power (W)
MPGdec (Mpeg video decoder)	Multi-media	3.4	39.7
MP3dec (Mp3 audio decoder)		3.0	37.1
H263enc (H263 video encoder)		2.2	35.5
bzip2	SPEC2k Integer	1.8	24.7
gzip		1.7	25.6
twolf		1.2	22.4
art	SPEC2k Float	0.7	17.3
equake		1.6	25.1
ammp		1.2	21.5

Table 2. Workload description.

ArchDVS can increase the processor frequency greater than the base value , and can have a performance greater than 1.0.

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, 3 SPEC2k integer applications, and 3 SPEC2k floating point applications. For each of the applications, the IPC and power consumption of the base non-adaptive processor is given. The base power consumption shown in Table 2 also includes leakage power.

As can be seen, a wide range of IPCs and power consumptions are observed. 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 are frame based applications which do not have an explicit initialization phase. Hence, we simulate the multimedia applications for 500 million instructions (atleast 400 frames) without fast forwarding.

6.3 Simulation Methodology

6.3.1 Simulator

We use the RSIM simulator [15] for performance evaluation. We use the Wattch tool [8] integrated with RSIM for power measurement. We derive temperature from power using the HotSpot tool [27]. The chip floorplan fed to HotSpot resembles the MIPS R10000 floorplan (without L2 cache), scaled down to $20.2mm^2$ ($4.5mm \times 4.5mm$).

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.

6.3.2 Leakage Power

Leakage power is calculated based on structure areas. For the 65nm process modeled, a leakage power density of $0.5W/mm^2$ at 383K is used. This value was obtained from

industry³, 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 [14]. At a temperature T, the leakage power, $P_{leakage}(T)$, is given by:

$$P_{leakage}(T) = P_{leakage}(383K) \times e^{\beta(T-383)} \quad (9)$$

where β is a curve fitting constant. The value of β we use for 65nm is taken from [14].

6.3.3 Initial Temperatures

As explained in [27], HotSpot has to be initialized correctly to produce accurate temperature values. We ran all simulations twice - the first run is used to obtain average power consumption values for every structure on chip. These average power values are then used to calculate the steady state temperature of every structure on chip, and more importantly, the steady state temperature of the heat sink. These steady state values serve as the initialization temperatures for the second run, in which temperature is modeled accurately.

6.3.4 Reliability Calculation

Based on temperature estimates obtained from HotSpot, and power estimates obtained from Wattch, RAMP calculates FIT values, for every structure on chip at $1\mu sec$ intervals. As discussed earlier, in Section 3.6, at qualification, the total FIT value due to each failure mechanism is set to 1000. These 1000 FITs are distributed on chip based on each structure's area.

7 Results

7.1 Designing Processors for Different T_{qual}

Figure 3 shows the performance for all the applications, when using the combination of architectural 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 nonadaptive processor, with a value of 1.0 representing no gain or loss. As mentioned in Section 3.6, we use T_{qual} as a proxy for reliability design cost. Results are shown for four values of T_{qual} , 405K, 375K, 355K, and 335K, which represent four qualification levels, ranging from most expensive to cheapest.

$T_{qual} = 405K$: The hottest temperature reached on chip by any application for our benchmark suite was near 405K. 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 11% for MP3dec to 19% for art) while still maintaining required processor reliability levels. This is because the operating conditions on chip while an application

³Reference omitted for anonymity

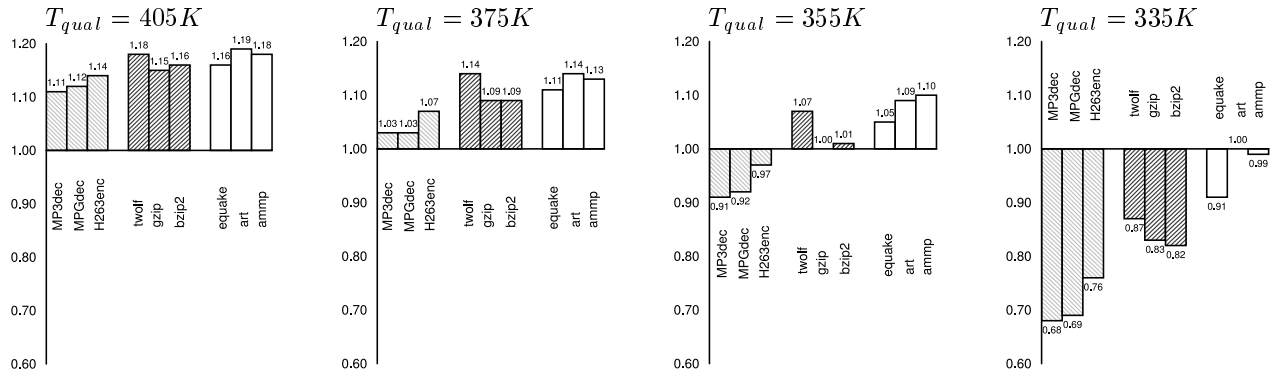


Figure 3. The performance of ArchDVS for DRM is shown on the y-axis, relative to the base non-adaptive architecture at 4 GHz. This is shown for all the applications for different T_{qual} values.

runs tend to be much lower than the worst case values. The performance gains experienced by the SPEC benchmarks tend to be higher on average than that 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 rates on the base architecture than the SPEC benchmarks. Based on these 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 nonadaptive processor can be marketed at a higher frequency (while still meeting the reliability target).

$T_{qual} = 375K$: At a T_{qual} value of 375K, 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 7% for H263enc to 14% for twolf and art. 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 405K, 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).

$T_{qual} = 355K$: A T_{qual} value of 355K represents a processor which was 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 4 cases, was within 5%. This 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 performance losses, while low IPC applications enjoy the

largest gains.

$T_{qual} = 335K$: A T_{qual} value of 335 K represents a processor which has been drastically underdesigned from a reliability perspective. All applications, with the exception of art, experience a slowdown. The high IPC multimedia applications experience the largest slowdown, with MP3dec suffering a loss of 32% 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.

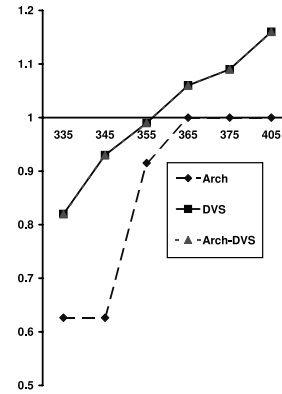


Figure 4. Comparison of different DRM adaptations for bzip2. The x-axis represents different T_{qual} values and the y-axis is performance speedup or slowdown. DVS and ArchDVS show identical behavior for bzip2 and are not distinguishable.

7.1.1 Implications of Designing for Reliability

- From the 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 405K to 375K, 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 355K, the performance loss seen was limited. Hence, a wide spectrum of T_{qual} values (in our case, 355K to 405K) 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 4 compares the performance (shown on the vertical axis, as an increase or slowdown over the base non-adaptive processor) of the three DRM adaptations, Arch, DVS, and ArchDVS, for one of the applications, bzip2, for a range of T_{qual} values (shown on the horizontal axis). Due to a lack of space, we do not show these results for all the other applications. However, the trends are very similar in all the other applications.

As can be seen, DVS and ArchDVS significantly outperform Arch, performing up to 30% better at a T_{qual} value of 345K. 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, tend to perform much better than Arch due to four main reasons: (1) Small drops in voltage and frequency result in large drops in temperature - this is because of the near cubic change in power consumption with performance of the processor with DVS, which translates to a large change in processor temperature. In comparison, architectural adaptation does not cause such a large change in processor power, and resultant temperature. Hence, Arch would be required to take a larger performance hit than DVS or ArchDVS to see the same temperature drop. (2) As can be seen in Equation 5, 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 earlier, in Section 6.1, the performance due to Arch can never be greater than 1, since it can not adapt the processor’s frequency. Hence, in any scenario where processor performance can be increased because of an over-designed T_{qual} , DVS and ArchDVS will perform much better than Arch (this is seen for T_{qual} values between 365K and 405K in Figure 4). (4) We explore a limited architectural adaptation space. Increasing the number of architectural configurations available could increase the performance of Arch.

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

since architectural changes affect reliability much less than DVS, it makes sense to have as aggressive a base processor as possible, and to use DVS for DRM.

7.3 Comparing Design for Reliability (DRM) and Temperature (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 5 compares DRM and DTM using voltage and frequency scaling (DVS) for all the applications. Every 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.

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; and 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 in Equation 5.

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 both 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 (CMOS) scaling in the deep submicron 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 mean-time-to-failure (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

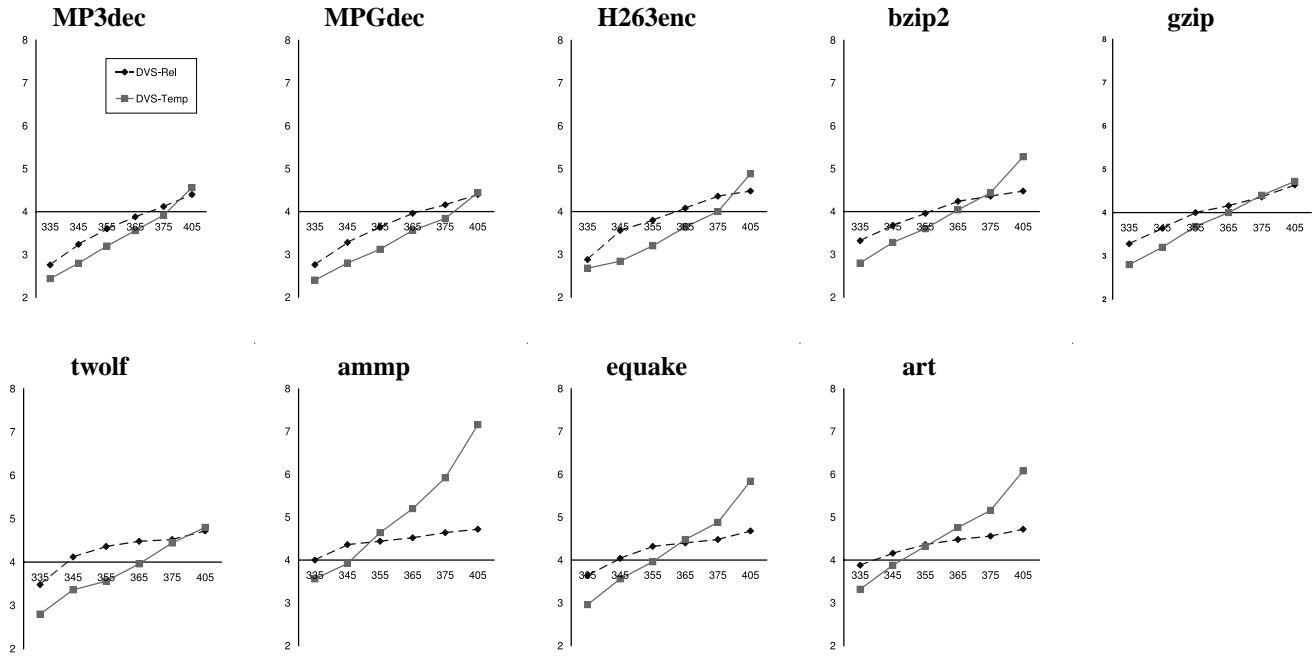


Figure 5. Comparing design for reliability and temperature. Temperatures on the x-axis represent T_{qual} for DRM and T_{max} for DTM. The frequency, in GHz, chosen by DVS for DRM (dotted line) and DTM (solid line) is shown on the y-axis.

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 DTM 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 DTM work.

This paper deals primarily with the impact of temperature (and related effects like thermal cycling) 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. Such current variations are increasingly common in processors that use on-chip power management techniques like clock- or power-gating. Since the supply voltage keeps scaling down, the concern about inductive noise is on the rise, since distinguishing between a '1' and a '0' becomes less reliable in processing and storage of data. The concern is not just about transient errors or machine "check-stops", but also about hard failures that may be caused by package-level resonance effects triggered by current swing events inside the processor. However, careful design and integration of on-chip and on-package decoupling capacitors can alleviate such resonance problems quite adequately. As such, we did not feel the need to model the workload-driven failure probabilities arising from such inductive noise effects.

In future work, we will propose specific adaptive control

algorithms that offer the promise of semi-optimal choice of microarchitectural adaptive techniques 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. Finally, we also plan to incorporate time dependence in our reliability 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] Electromigration for Designers, Cadence Design Systems White Paper. In <http://www.cadence.com/whitepapers/electromigration.html>, 1999.
- [3] Sony Semiconductor Quality and Reliability Handbook. In *Sony Global Corp.*, 2001.
- [4] Failure mechanisms and models for semiconductor devices. In *JEDEC Publication JEP122-A, Jedec Solid State Technology Association*, 2002.
- [5] Critical Reliability Challenges for The International Technology Roadmap for Semiconductors. In *International Sematech Technology Transfer Document 03024377A-TR*, 2003.
- [6] W. Abadeer et al. Key Measurements of Ultrathin Gate Dielectric Reliability and In-Line Monitoring. In *IBM Journal of Research and Development*, 1999.
- [7] T. M. Austin. Diva: A reliable substrate for deep submicron microarchitecture design. In *Proc. of the 32nd Annual Intl. Symp. on Microarchitecture*, 1998.

- [8] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *Proc. of the 27th Annual Intl. Symp. on Comp. Architecture*, 2000.
- [9] C.-K.Hu et al. Scaling effect on electromigration in on-chip cu wiring. In *International Electron Devices Meeting*, 1999.
- [10] A. Chandrakasan, W. J. Bowhill, and F. Fox. Design of High-Performance Microprocessor Circuits. 2001.
- [11] K. P. Cheung. Thin Gate-oxide Reliability - the current status. In *Keynote paper, Symposium on Nano Device Technology 2001*, 2001.
- [12] A. Dasgupta and R. Karri. Electromigration Reliability Enhancement Via Bus Activity Distribution. In *33rd Design Automation Conference*, 1996.
- [13] E.Eisenbraun et al. Integration of cvd w- and ta-based lines for copper metallization. In *MKS white paper*, <http://www.mksinst.com/techpap.html>, 2000.
- [14] S. Heo, K. Barr, and K. Asanovic. Reducing power density through activity migration. In *Proc. of Intl. Symp. on Low Power Electronics Design*, 2003.
- [15] C. J. Hughes, V. S. Pai, P. Ranganathan, and S. V. Adve. RSIM: Simulating Shared-Memory Multiprocessors with ILP Processors. *IEEE Computer*, February 2002.
- [16] H.V.Nguyen et al. Fast temperature cycling stress-induced and electromigration-induced interlayer dielectric cracking failure in multilevel interconnection. 2001.
- [17] J.H.Stathis. Reliability limits for the gate insulator in cmos technology. In *IBM Journal of Research and Development*, 2002.
- [18] J.R.Black. A brief survey of electromigration and some recent results. In *IEEE Transactions on Electron Devices*, 1969.
- [19] P. K. Lala. Self-checking and fault-tolerant digital design.
- [20] N. P. Mencinger. A mechanism-based methodology for processor package reliability assessments. In *Intel Technology Journal*, Q3,2000.
- [21] D. Patterson et al. Recovery-oriented computing (roc): Motivation, definition, techniques, and case studies. In *UC Berkeley Computer Science Technical Report UCB//SD-02-1175*, 2002.
- [22] M. G. Pecht et al. Guidebook for Managing Silicon Chip Reliability. 1998.
- [23] E. Rotenberg. Ar/smt: A microarchitectural approach to fault tolerance in microprocessors. In *International Symposium on Fault Tolerant Computing*, 1998.
- [24] K. Seshan et al. The Quality and Reliability of Intel's Quarter Micron Process. In *Intel Technology Journal*, Q3,1998.
- [25] P. Shivakumar et al. Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic. In *International Conference on Dependable Systems and Networks*, 2002.
- [26] P. Shivakumar et al. Exploiting microarchitectural redundancy for defect tolerance. In *21st International Conference on Computer Design*, 2003.
- [27] K. Skadron et al. Temperature-Aware Microarchitecture. In *Proc. of the 30th Annual Intl. Symp. on Comp. Architecture*, 2003.
- [28] L. Spainhower and T. A. Gregg. Ibm s/390 parallel enterprise server g5 fault tolerance: A historical perspective. In *IBM Journal of Research and Development*, September/November 1999.
- [29] J. Srinivasan and S. V. Adve. Predictive dynamic thermal management for multimedia applications. In *Proc. of the 2003 Intl Conf. on Supercomputing*, 2003.
- [30] K. Trivedi. Probability and statistics with reliability, queuing, and computer science applications.
- [31] 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.