ExtraTime: Modeling and Analysis of Wearout due to Transistor Aging at Microarchitecture-Level

Fabian Oboril and Mehdi B. Tahoori
Chair of Dependable Nano Computing (CDNC), Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
Email: {fabian.oboril, mehdi.tahoori}@kit.edu

***Abstract*—With shrinking feature sizes, transistor aging due to NBTI and HCI becomes a major reliability challenge for microprocessors. These processes lead to increased gate delays, more failures during runtime and eventually reduced operational lifetime. Currently, to ensure correct functionality for a certain operational lifetime, additional timing margins are added to the design. However, this approach implies a significant performance loss and may fail to meet reliability requirements. Therefore, aging-aware microarchitecture design is inevitable. In this paper we present *ExtraTime*, a novel microarchitectural aging analysis framework, which can be used in early design phases when detailed transistor-level information is not yet available to model, analyze, and predict performance, power and aging. Furthermore, we show a comprehensive investigation using *ExtraTime* of various clock and power gating strategies as well as aging-aware instruction scheduling policies as a case study to show the impact of the architecture on aging.**

***Index Terms*— NBTI; HCI; Wearout Modeling; Microarchitecture; Performance Simulator;**

I. Introduction

Aggressive transistor scaling of CMOS technology over the past decades allowed increasing transistor counts and by this means the realization of more and more features in modern microprocessors. However, the success of future generations is threatened by various reliability challenges associated with decreasing feature sizes to nanoscale dimensions. Thereby faster transistor aging, is one major reliability issue [6]. Among several physical effects that cause transistor aging *Negative Bias Temperature Instability* (NBTI) [37] and *Hot Carrier Injection* (HCI) [31] are the dominant effects [4]. Both phenomena lead to a shift of the threshold voltage (V_{th}) of the impaired transistors, which manifests in increasing switching and path delays. This will eventually lead to timing violations and finally to faster wearout of the system. Thereby, NBTI- and HCI-induced wearout strongly depends on several key factors such as usage (gate bias, number of transitions, etc.), temperature and supply voltage of the affected transistors.

To combat these runtime degradation issues, manufacturers are currently adding safety margins (*guardbands*) to their designs, to ensure that the chips will be functional for a certain lifetime. However, for a 32 nm technology the necessary overdesign in terms of performance can easily exceed 10% for a 3 year lifetime [20]. To make it even worse, transistor aging will accelerate with downscaling, leading to larger margins and thus to an even higher performance loss [6], [20]. This also implies increasing costs for development and manufacturing.

Hence, new approaches are necessary to take further advantage of scaled technology nodes. Therefore, an aging-aware design of the entire processor, including the microarchitecture, is inevitable. To achieve this goal, it is necessary to assess transistor aging already in early design phases and to balance aging with other key design aspects (performance, power and area). However, a major challenge is that detailed transistor-level information is not yet available in these phases making an accurate wearout estimation very difficult.

Therefore, we propose in this work *ExtraTime*: a novel aging-aware microarchitectural framework. ExtraTime enables design space exploration in early design phases (e.g. development of the microarchitecture) not only for performance and power but also for aging, without having detailed knowledge about the final hardware implementation or layout. This framework is based on a cycle-accurate performance simulator with extensions to model power consumption and temperature. Moreover, ExtraTime includes novel and accurate aging models for NBTI and HCI at microarchitecture-level, derived from and validated with transistor-level models. By this means it is possible to investigate various aging mitigation techniques and their influence on many critical design parameters like temperature, power, performance and particularly wearout, while various applications are running on the processor.

Compared to the previous work, our framework comes with several benefits. First, no detailed transistor-level information is necessary, so that ExtraTime can be used in early design phases to optimize the architecture for performance, power and aging mitigation.Even though some methods existing [3], [12], [16], [29], [32] focus also at microarchitecture-level, they require detailed transistor-level information, making them inappropriate for this purpose. Second, the previous models used to estimate aging are often too simplistic (e.g. usage-dependency is not correctly considered, inaccurate temperature modeling) leading to overestimations [14], [23], [30], [32]. As shown later in this paper, neglecting the influence of usage or temperature has a tremendous impact on wearout estimation. In addition the impact of such techniques on power or performance is often not investigated in previous work. Hence, an accurate microarchitectural framework that does not need transistor-level information is still missing. This gap can be closed with ExtraTime.

To illustrate the applicability of this framework, we show a comprehensive analysis of various clock and power gating strategies as well as aging-aware instruction scheduling policies for a superscalar architecture. The results obtained with ExtraTime show that it is possible to extend lifetime by more than 4 times, while performance is only reduced by 4% in average. This performance loss can be eliminated using some of the gained timing headroom (i.e. clock frequency can be increased).

In summary, the key contributions of this work are as follows. First, ExtraTime, a microarchitectural framework to model and analyze wearout due to transistor aging is presented. Second, accurate microarchitectural aging models for NBTI and HCI are introduced and validated with detailed transistor-level models. And finally a thorough analysis of various aging mitigation techniques using ExtraTime is shown.

The rest of this paper is organized as follows. In Section II the considered aging phenomena are introduced and previous work in the scope of transistor aging is discussed. The novel ExtraTime framework including the microarchitectural aging models, an accuracy analysis and comparison with state-of-the-art solutions is presented in Section III. The applied aging mitigation techniques are provided in Section IV. The corresponding results can be found in Section V. Afterwards, we conclude in Section VI.

II. Preliminaries on Transistor Aging

Transistors age mainly due to the physical phenomena *Negative Bias Temperature Instability* (NBTI) and *Hot Carrier Injection* (HCI) [4]. Both phenomena and their transistor-level models are described in more details in the following. Afterwards, some previous work is discussed.

1) NBTI: The NBTI effect consists of two different phases. When a logic '0' is applied at the gate of a PMOS transistor, this transistor is under *stress*. During this phase, traps are generated in the interface between gate oxid and channel, which increases $|V_{th}|$. In contrast, when a logic '1' is applied at the gate of the same transistor, some traps are filled, which leads to a decreasing $|V_{th}|$ (*recovery* phase). However, the initial shift cannot be entirely compensated leading to an overall V_{th} drift over time. Thereby, the shift depends on several different aspects, e.g. temperature T, supply voltage V_{dd} and the ratio between the time a transistor is under stress and total time (duty cycle δ).

In [37] an analytical model for the NBTI process is derived. With this analytical model it is possible to make a long term prediction of the V_{th} shift for a couple of years. Thereby, ΔV_{th} at time $t > 0$ is given by:

$$\Delta V_{th}(\delta, T, V_{dd}, t) \leq A_N \cdot u(V_{dd}) \cdot \frac{(v(T) \cdot \delta(t) \cdot t_m)^n}{w(\delta, T, t)^{2n}} \quad (1)$$

with

$$\begin{aligned} u &= (V_{dd} - V_{th}) \cdot \exp((V_{dd} - V_{th})/E_0) \\ v &= \xi_4 \cdot \exp(-E_a/kT) \\ w &= 1 - \left(1 - \frac{\xi_1 + \sqrt{\xi_3 \cdot v(T) \cdot (1-\delta(t)) \cdot t_m}}{\xi_2 + \sqrt{v(T) \cdot t}}\right)^{\frac{1}{2n}} \end{aligned}$$

Factor	NBTI	HCI
Supply Voltage	exponential	exponential
Temperature	exponential	exponential
"Usage"	nonlinear (see Figure 1(a))	sublinear (root)

TABLE I
Influence of several parameters on NBTI/HCI-induced aging

where A_N, n, E_0 and ξ_i are technology dependent constants, E_a is the activation energy (positive), k is the Boltzmann constant and t_m is the period between two measurements.

2) HCI: HCI is mainly affecting NMOS transistors, where accelerated electrons inside the channel can collide with the gate oxide interface and thereby create electron-hole pairs. Thus, free electrons get trapped in the gate oxide layer, which leads to an increasing V_{th}.

Since the "hot" energetic electrons are only generated when the NMOS transistor is making a transition [11], the voltage shift is very sensitive to the number of transitions. The authors in [31] have shown, that the relationship between the number of transitions, i.e. runtime, and voltage shift is sublinear. Hence, the voltage shift has a sublinear dependency on the clock frequency f, runtime t and the activity factor α, which is the ratio of the cycles the transistor is doing transitions and the total amount of cycles. Furthermore, the HCI effect has an exponential dependency on temperature [8]. Putting all the dependencies together leads to the following model, which describes the HCI effect:

$$\Delta V_{th}(\alpha, T, V_{dd}, t) = A_H \cdot u(V_{dd}) \cdot v(T) \cdot \sqrt{\alpha \cdot f \cdot t} \quad (2)$$

with

$$u(V_{dd}) = \exp((V_{dd} - V_{th})/E_1) \;, \;\; v(T) = \exp(-E_a/kT)$$

A_H and E_1 are technology dependent constants and the activation energy E_a is again considered to be positive. Please note that the temperature relation for technology nodes larger than 100 nm is reversed [8].

3) Summary: NBTI and HCI lead to a V_{th} shift, which results in an increased switching delay of the affected transistors. Thereby, the shift depends on various parameters as shown in Table I, namely temperature T, supply voltage V_{dd}, frequency f and "usage", i.e. the time a transistor is active (HCI) or under stress (NBTI). Hence, neglecting just one of these influences or the correlation between these (i.e. voltage $\rightarrow$ temperature, usage $\rightarrow$ temperature, etc.) can lead to a very inaccurate aging estimation. In Figure 1 the first order influence of temperature and "usage" is illustrated. As one can easily see, even small deviations from the real "usage"/temperature value can have a huge impact on the estimated wearout. For this reason it is necessary to consider all aspects at once.

4) State of the Art: A lot of research is done at various design levels to mitigate aging effects. To name just a few, special NBTI-resilient circuits [1], input vector control [15], [38], power gating [9], [10], adaptive body biasing [32], dynamic voltage and frequency scaling (DVFS) [3] and enhanced instruction and application scheduling techniques [29], [32] are some of the existing aging mitigation methods. However, the focus of all these approaches is just on aging mitigation.

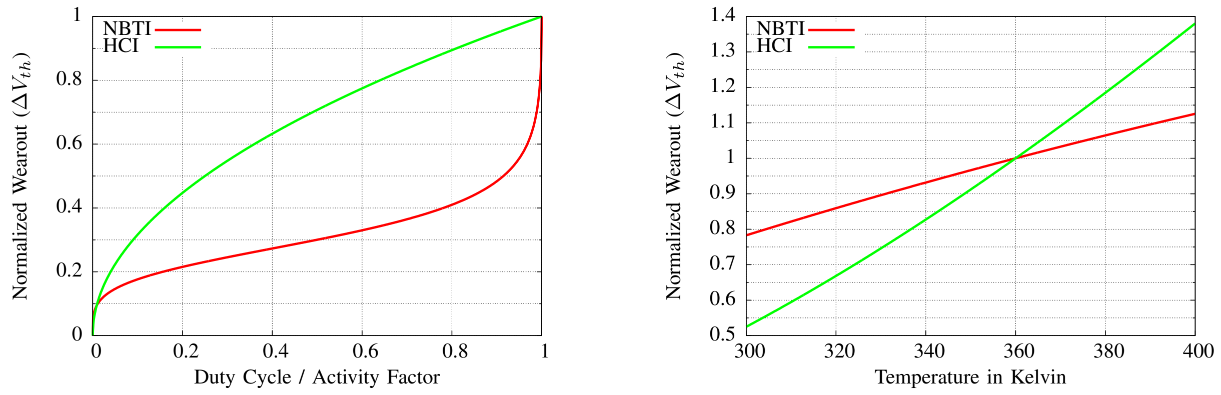


Fig. 1. Influence of Temperature (normalized to 360 K) and Usage (Duty Cycle for NBTI, Activity Factor for HCI) on Aging due to NBTI and HCI

A thorough study taking both performance and power into account is still missing. Furthermore various "aging sensors" have been proposed [2], [13], [21]. However, these cannot slow down aging, at most they can indicate that a device will fail soon.

Some work [12], [14], [16], [23], [30], [32] also proposes frameworks to model and analyze aging at different abstraction level. An overview and comparison with our approach will be presented in Section III-D.

III. EXTRATIME FRAMEWORK

The goal of this work is to model and analyze wearout due to transistor aging at microarchitecture-level. For this reason we have developed a novel aging-aware microarchitectural framework called *ExtraTime*. In this section we will present the overall idea and the main components of ExtraTime. Furthermore, the aging models at microarchitecture-level are introduced and at the end a comparison with state-of-the-art solutions is presented.

A. Basic Components

The basis of our ExtraTime framework is the *gem5* performance simulator [5] that includes a cycle-accurate model for a pipelined, out-of-order, superscalar architecture, which is based on the Alpha 21264 core [22]. While running several different workloads gem5 delivers detailed information about the overall performance of the modeled processor and about the usage of different microarchitectural blocks such as pipeline stages or execution units. However, this information is not enough to make an accurate aging estimation as shown in Table I. Therefore, sophisticated temperature models are necessary. Since temperature strongly correlates with power consumption, also detailed models for power are needed. For the power model (both dynamic and static power) we use a customized version of *McPAT* [26] and the temperature model is based on *HotSpot* [17].

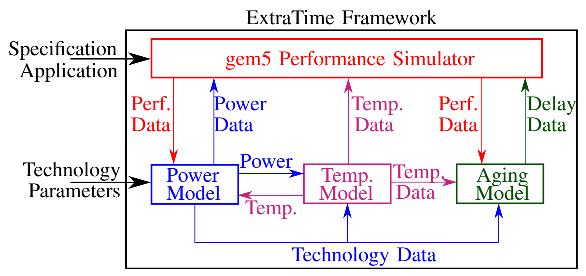


Fig. 2. Data/Information flow in our ExtraTime Framework

The temperature information in conjunction with information about the usage/activity of different microarchitectural blocks is used by our microarchitectural aging models for NBTI and HCI. These will be explained in detail in the following Subsection III-B. With the help of these models the actual and future aging status of the transistors inside each block can be estimated by using just microarchitectural information. An illustration of the model interaction can be also found in Figure 2. Please note that we have integrated all models in one common framework. This enables a runtime analysis of power, temperature and wearout (online), which reduces simulation time and makes the investigation of dynamic runtime adaptation techniques possible.

A simplified version of ExtraTime is presented in [27].

B. Aging Models at Microarchitecture-Level

For our purpose, to model aging at microarchitecture-level without having detailed transistor-level implementations, it is necessary to use special models. In the following we will introduce these models and how these are derived from transistor-level models.

1) Definition of an Aging Metric: As the first step to build the microarchitectural aging models, the definition of a useful metric to estimate aging is necessary. For this purpose we use the *relative delay change* ($\Delta^{rel} d_B$) of a microarchitectural block B at time $t > t_0$ induced by aged transistors, which is defined as the ratio of the delay change (Δd_B at time t) over the original delay d_B (at time t_0):

$$\Delta^{rel} d_B(t) = \frac{\Delta d_B(t)}{d_B(t_0)} = \frac{d_B(t)}{d_B(t_0)} - 1.$$

The delay of a block always depends on its transistor layout (i.e. the number, size and type of transistors in the critical paths, the delay of the transistors, etc.). This information is usually not available at microarchitecture-level, e.g. in early phases of the microprocessor development the RTL- or transistor-level circuit description is not available. Hence, in order to build microarchitectural aging models some assumptions of the underlying hardware layout are necessary. Therefore, we assume that all transistors in one microarchitectural block behave similar, i.e. age at the same rate. This means that all transistors inside one microarchitectural block can be represented by one single transistor $\mathcal{T}_B$ (*representative transistor*). By this means $\Delta^{rel} d_B$ of a block can be estimated by the relative delay change of the representative transistor $\Delta^{rel} d_{\mathcal{T}_B}$, inside this block, i.e. :

$$\Delta^{rel} d_B = \Delta d_B / d_B \approx \Delta^{rel} d_{\mathcal{T}_B}. \tag{3}$$

The aging effects we consider in this work, i.e. NBTI and HCI, lead to a shift of the threshold voltage V_{th} of the impaired transistor. This shift manifests in an increased transistor delay d. The relation between V_{th} and d follows a power law [7]:

$$d = C \cdot (V_{dd} - V_{th})^{-r}, \tag{4}$$

where C and r are a technology dependent constants. With this relationship, the relative delay change $\Delta^{rel} d_{\mathcal{T}_B}$ of the representative transistor in block B and by this means the relative delay change of all transistors in block B is:

$$\Delta^{rel} d_{\mathcal{T}_B}(t) = \left(1 - \frac{\Delta V_{th}(t)}{V_{dd} - V_{th}(t_0)}\right)^r - 1 = f(\Delta V_{th}(t)).$$

Since everything in this equation is constant in time (and known) except ΔV_{th}, just ΔV_{th} needs to be calculated to determine $\Delta^{rel} d_{\mathcal{T}_B}$. In combination with equation (3) aging of the microarchitectural block B can be estimated. Moreover, as it will be explained in the following subsections, ΔV_{th} can be estimated using *only* microarchitectural information and some known constants. Thus, $\Delta^{rel} d$ of each block can be calculated using *only* data which is known at microarchitecture-level.

$$\begin{aligned} \Delta^{rel} d_B &= \Delta d_B / d_B \approx \Delta^{rel} d_{\mathcal{T}_B} = f(\Delta V_{th}) \\ &= f(g(\text{para. known at microarchitecture-level})) \end{aligned}$$

The last equation is also the reason, why the relative delay change is a better choice for the aging metric than the total delay change. While the former one can be estimated using only microarchitecutral information, for the latter the original delay $d_B(t_0)$ needs to be known, which is usually not the case at microarchitecture-level. Furthermore, the *Mean Time To Failure* (MTTF) is a function of the maximum $\Delta^{rel} d_B$, which does not lead to timing failures. Hence, also MTTF can be determined using the relative delay change as metric.

2) *Negative Bias Temperature Instability:* The goal in the following is to estimate the V_{th} shift of the representative transistor $\mathcal{T}_B$ of a microarchitectural block B due to NBTI by using only microarchitectural information.

While the technology dependent constants ξ_i, n, the Boltzmann constant k and the activation energy E_a in equation (1) and V_{dd} are fixed and hence known, the transistor dependent temperature T and duty cycle δ are variables, that have to be attained. Since we assume that all transistors in one microarchitectural block behave similar, they have the same temperature (as the whole block) T_B, which can be obtained from the temperature model of ExtraTime. By this means, also $\mathcal{T}_B$ has this temperature. In order to estimate the duty cycle δ of $\mathcal{T}_B$ (ratio between the time the transistor is under stress and total time) at microarchitecture-level, the duty cycle δ_B of the block B is used, which represents the usage behavior of the block. Therefore, the stress time of block B is defined as the time in which at least one transistor inside this block can be under stress:

$$\delta_B = \frac{t_{stress,B}}{t_{total}} = \frac{cyc_{stress,B}}{cyc_{total}} \tag{5}$$

Thus, the newly defined microarchitectural duty cycle δ_B of an entire block can be derived from parameters delivered by a performance simulator (total cycle count $= \#cyc_{total}$, number of stress cycles $= \#cyc_{stess,B}$). In case power gating is used, the number of stress cycles can be easily calculated using the following relation (number of power gated cycles $= \#cyc_{pg,B}$):

$$\#cyc_{stess,B} = \#cyc_{total} - \#cyc_{pg,B}.$$

Putting all this together and using δ_B as an estimation of duty cycle of the representative transistor $\mathcal{T}_B$ leads to an upper bound estimation at microarchitecture-level for the V_{th} shift of $\mathcal{T}_B$ induced by NBTI:

$$\Delta^{worst} V_{th}(t) \le A_N \cdot u(V_{dd}) \cdot \frac{(v(T_B) \cdot \delta_B \cdot t_m)^n}{w(\delta_B, T_B, t)^{2n}}, \tag{6}$$

whereby u, v and w are defined as in the transistor-level model (see Equation (1)), but using the microarchitectural values T_B and δ_B instead of the transistor dependent ones (T and δ).

However, the definition of the block duty cycle in Equation (5) is an upper bound estimation for the transistor duty cycle, since the duty cycle of every transistor inside a block will be far less than the duty cycle of the entire block in typical workloads. Hence, the real V_{th} shift will be smaller than the one calculated in Equation (6), which means that this equation is an overestimation. To improve the estimation, we use the fact that the duty cycle of a transistor inside a block B is the product of the duty cycle of the block δ_B and the effective duty cycle δ_e. Together with the assumption that the effective duty cycle of different transistors inside one block is uniformly distributed between 0 and 1, one can calculate the "average" V_{th} shift (over all transistors in block B) as follows:

$$\Delta^{avg} V_{th}(t) \le \int_0^1 A_N u(V_{dd}) \frac{(v(T_B) \cdot \delta_B \cdot \delta_e \cdot t_m)^n}{w(\delta_B \cdot \delta_e, T_B, t)^{2n}} \mathrm{d}\delta_e. \tag{7}$$

One should note that this estimation might be too optimistic for certain transistors inside the block, but since we are interested in the V_{th} shift of the representative transistor, i.e. the delay change of the entire microarchitectural block rather than in the change of a single transistor, the "average" V_{th} shift is a good estimation. Hence, we will use this equation as our microarchitectural NBTI model to estimate the V_{th} shift of the representative transistor $\mathcal{T}_B$. This is then used to determine $\Delta^{rel} d$ with the help of Equation (4).

If detailed knowledge about the underlying hardware implementation is available, one can also use a partially weighted mean (integral) to achieve higher accuracy, for example to represent duty cycles which follow a normal distribution.

3) Hot Carrier Injection: As for NBTI, the goal of the following part is to estimate the V_{th} shift of the representative transistor $\mathcal{T}_B$ due to HCI by using only microarchitectural information.

The approach to transfer the transistor-level model for HCI in Equation (2) to microarchitecture-level is quite similar to the one used for NBTI. Again the problem is that the temperature T corresponds to individual transistors and the activity factor α is also transistor-dependent. Hence, again the temperature T_B of an entire microarchitectural block B is used for all transistors inside this block. Since the activity factor of a transistor is the product of the activity factor of this transistor while the complete microarchitectural block B is active (effective activity factor α_{eff}) and the activity factor α_B of the block, Equation (2) can be written as follows:

$$\Delta V_{th}(t) = A_H \cdot \sqrt{\alpha_{eff}} \cdot u(V_{dd}) \cdot v(T) \cdot \sqrt{\alpha_B \cdot f \cdot t}$$

Thereby a block is active, when at least one transistor inside this block is active (i.e it can make a transition):

$$\alpha_B \quad = \quad \frac{t_{active,B}}{t_{total}} = \frac{cyc_{active,B}}{\#cyc_{total}}$$

As the equations above illustrate, the activity factor α_B of a block, can be calculated using only parameters delivered by the performance simulator (total cycle count $= \#cyc_{total}$, number of active cycles $= \#cyc_{active,B}$). In case clock or power gating is available, the number of active cycles of a block can be calculated using the following relation (number of power gated cycles $= \#cyc_{pg,B}$, number of clock gated cycles $= \#cyc_{cg,B}$):

$$\#cyc_{active,B} = \#cyc_{total} - \#cyc_{pg,B} - \#cyc_{cg,B}.$$

With regard to the representative transistor, the effective activity factor α_{eff} has a specific meaning. It represents the *average switching activity* $\alpha_{avg,B}$ of all gates in the microarchitectural block B. In the worst case the average switching activity for the (near-)critical paths can be 1, according to our transistor-level investigations using SPEC2000 workloads. Hence, we use the value of 1 for $\alpha_{avg,B}$ to obtain the maximum V_{th} shift that can be induced by typical applications. If detailed transistor-level information is available this value can also be adjusted.

Thus, the HCI model at microarchitecture-level and in this way the estimation of the V_{th} shift of the representative transistor $\mathcal{T}_B$ has the form:

$$\Delta V_{th}(t) = A_H \cdot \sqrt{\alpha_{avg,B}} \cdot u(V_{dd}) \cdot v(T_B) \cdot \sqrt{\alpha_B \cdot f \cdot t} \quad (8)$$

Similar to NBTI induced aging, Equation (4) and (8) enable us to also calculate the $\Delta^{rel}d$ due to HCI during runtime to determine the actual aging status.

C. Accuracy Analysis

As mentioned earlier, the abstraction of the aging models from transistor-level to microarchitecture-level has an impact on the accuracy. The purpose in the following is to quantify the amount of inaccuracy due to this abstraction.

1) Temperature: One major assumption of our proposed microarchitectural aging models is that all transistors within a block have the same temperature and hence we neglect temperature fluctuations within a block. Using HotSpot [17] and the experimental setup described in Section V-A we found that the maximum temperature difference in one microarchitectural block is at most 6 °C. As it is shown for an ALU in Table II, this leads to a difference in terms of the total delay of less than 1 %, which is negligible. Also the difference in terms of $\Delta^{rel}d$ is considerably small (< 5 %), i.e. the deviation of our microarchitectural aging models compared to an accurate transistor-level model is less than 5 %.

2) Usage: Beside the assumption that all transistors have the same temperature, we also made an assumption regarding the "usage" of the transistors in a microarchitectural block, i.e. the time a transistor is active (HCI) or under stress (NBTI).

For NBTI we derived the "average" V_{th} shift given in Equation (7), to take into account that different transistors in one block will have different stress time. We have validated this approach using a detailed transistor-level model of an ALU by considering the influence of different input vectors (i.e. different instruction opcodes and different operands) [38] and the stacking effect [35]. Therefore, the ALU of the Illinois Verilog Model (IVM) [36], which is a Verilog model for the Alpha 21264 core, was synthesized using the Synopsys Design Compiler and mapped to the SAED 90 nm standard cell library. Afterwards, for each instruction 10,000 random operands together with the instruction opcode were applied at the inputs of the ALU to find out the duty cycle of each transistor in the 10% most critical paths. In the next step, the delay change for each considered transistor was estimated using the transistor-level model given in Equation (1). Using these results the overall delay change for the entire ALU was calculated. For a runtime of three years under a constant temperature of 70 °C, this gate-level approach estimates the maximum $\Delta^{rel}d_{ALU}$ (maximum over all input vectors) to be 9.7%. Using our microarchitectural NBTI model the $\Delta^{rel}d_{ALU}$ is estimated to be 9.4% under the same conditions (i.e. temperature, technology, etc.). As shown in Table III our model is just 3% off. This shows that our microarchitectural aging model for NBTI comes with a reasonable accuracy given the level of abstraction, but in addition microarchitectural investigations are possible. Please note that investigating the maximum $\Delta^{rel}d_{ALU}$ is sufficient here, since the maximum

	Temperature	$\Delta^{rel}d_{ALU}$ after 3 years due to NBTI	$\Delta^{rel}d_{ALU}$ after 3 years due to HCI
Minimum	71 °C	9.4%	8.5%
Maximum	77 °C	9.6%	8.9%
Relative Difference		2.1%	4.5%

TABLE II
TEMPERATURE EFFECT ON AGING INDUCED DELAY CHANGE FOR AN ALU

	$\Delta^{rel} d_{ALU}$ after 3 years	
	NBTI	HCI
Detailed Transistor-Level Model (Max)	9.7%	8.4%
Proposed Microarchitectural Model	9.4%	8.4%
Relative Difference	3.1%	0.0%

TABLE III
DIFFERENCE BETWEEN AN DETAILED TRANSISTOR-LEVEL MODEL AND OUR PROPOSED MICROARCHITECTURAL MODELS REGARDING THE ESTIMATE DELAY FOR AN ALU AFTER 3 YEARS DUE TO NBTI

delay degradation that can occur matters most, i.e. it does not matter how many input combinations are working, as long as one can lead to a timing failure, the entire system can fail. Furthermore, it is worth to note that the accuracy of our microarchitectural model for NBTI can be improved, if detailed knowledge about the underlying hardware implementation is available. In that case the "average" V_{th} shift given in Equation (7) can be modified using a partially weighted mean (integral) to represent duty cycles which follow a normal distribution.

For HCI the accuracy investigation is very similar. The only difference is, that pairs of input combinations have to be investigated, in order to obtain the switching behavior of the circuit. The results are even better, showing that also the model for HCI comes with a satisfying accuracy for the estimation of the maximum degradation with respect to the level of abstraction.

D. Advantages of ExtraTime

Beside the accurate aging models, ExtraTime possesses several other advantages compared to previous work at microarchitecture-level. In the following part the advantages are explained and underlined with experimental data.

To increase the accuracy of the aging estimation we have extended the power model (based on McPAT) in various areas. Our customized version uses the actual temperature of each microarchitectural block to estimate leakage power. Compared to the original version, in which all blocks were assumed to have the same temperature, this is a huge benefit. This is also a major lack in current solutions such as [12], [30], [32], where usually first the performance simulation is accomplished, afterwards a power trace is created followed by a modeling of the temperature behavior. Hence, an accurate coupling between temperature and power is missing in most existing solutions. Thereby, our results obtained with the adjusted version of McPAT show that a temperature difference of 10 degrees can impact leakage power by up to 50%, if temperature increases by 30 degrees leakage power can even increase by a factor of 3. Due to the coupling between temperature and power, the increase in leakage power leads to an increase in temperature, which can be up to 23 degrees (analysis based on HotSpot) in the latter case. If power and temperature are considered independently, the increase in leakage power and its impact on temperature is neglected. For this reason it is very important to consider the tight coupling between temperature and power, in order to obtain accurate power, temperature results, which are necessary to accurately estimate aging.

Furthermore, most previous techniques use aging models that are too simplistic. Often the applied models do not consider usage at all [14], [23], [30], meaning a microarchitectural block has always the same aging rates, no matter if the block is used, clock gated or power gated. Hence, these frameworks are not capable of accurately modeling aging of modern microprocessors under varying applications. In [32] usage is taken into account, but for HCI the usage dependency is considered to be linear. In fact, this relationship is sublinear [31], which leads to overestimations. Another problem of some techniques is that the temperature is estimated on a too coarse level, e.g. in [30] the highest temperature of the entire microprocessor is used to estimate aging. Our experimental analysis shows that this would lead to around 30% wearout-overestimation for the execution units of the processor under investigation (see Section V-A).

Another advantage of our approach is the integration of all models in one common framework. In order to allow dynamic runtime adaption (see [28]) based on the current state (temperature, power, aging, performance) of the microprocessor it is necessary to calculate the power consumption, temperature or wearout every X cycles while the simulation is running (online) rather than only once after the simulation is done (offline). Transferring the necessary data every X cycles among standalone (offline) solutions can result in considerable performance overhead (up to 25%) and huge data storage overhead (up to 25x). In summary, the only viable solution to accurately investigate wearout, temperature, and power including the runtime behavior of the executed applications is the integration of the necessary models in one holistic framework. However, this is missing in all previously published techniques, which have their focus on microarchitectural solutions. ExtraTime closes this gap.

IV. Aging Mitigation Techniques

With all these features the ExtraTime framework gives one the possibility to explore many different techniques and policies at microarchitecture-level to investigate and balance performance, power and wearout. Major factors which can be addressed to slow down aging are the temperature T, supply voltage V_{dd} and usage, i.e. the time a transistor is active (HCI) or under stress (NBTI). Hence, a good workload distribution among the available execution units in a superscalar architecture is crucial for efficient aging mitigation. Thereby, in this work the focus is on clock and power gating and "aging-aware" instruction scheduling.

A. Clock Gating

Clock gating is a well known and very efficient technique to reduce the active power consumption of a microprocessor. By switching off the clock signal for a logic block, the latches inside this block are no longer toggling. Hence the overall switching activity can be reduced leading to a lower (dynamic) power consumption [33]. Furthermore, clock gating can be used to mitigate the effects of NBTI and HCI. By reducing the power consumption, the temperature decreases, and hence the V_{th} shift due to NBTI and HCI is damped. Regarding HCI, clock gating additionally reduces the switching activity which also lowers the V_{th} shift. Since the clock can be (de)activated every cycle without any delay, clock gating does not come along with performance losses, which makes it very attractive.

B. Power Gating

Power gating is nowadays not as widespread as clock gating. Currently it is mainly used in multi-core processors to switch off unused cores to reduce power consumption [25]. Like clock gating, power gating can be used also to mitigate aging effects. Since it reduces the power consumption, temperature decreases which in turn mitigates the NBTI and HCI effect. Furthermore the stress time of PMOS transistors is reduced and thereby δ_B, since a power gated PMOS transistor is in recovery mode ($V_{gs} = 0$). In addition power gating also lowers the number of (unnecessary) transitions and thereby α_B. By this means the NBTI and HCI effects are mitigated. However, it takes some time to power down or up a block of a processor, whereby the time periods depend on the size and amount of the power gate transistors as well as the size of the power-gated block. Recent work has shown, that an ALU can have a wake up time of 3 ns to 10 ns [24], [34]. Hence the power gated block is not available for a certain amount of time, which can lead to performance losses. In this work, a wakeup time of 7 ns and a power down period of 3 ns are used. The time period, t_{idle}, after which a unit can be power gated and the minimum duration, t_{dur}, of power gating can be chosen freely. Please note that the time a unit is power gated, t_{sleep}, can be much longer than t_{dur}. An illustration of the relationship between the different periods is given in Figure 3.

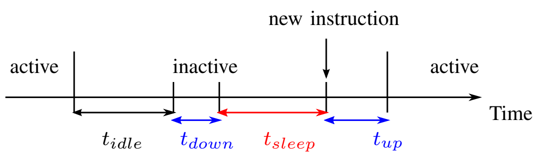


Fig. 3. Time flow of power gating periods of an execution unit

C. Aging-Aware Scheduling Techniques

State-of-the-art superscalar microprocessors contain several execution units of the same type, that can be used in parallel depending on the degree of instruction level parallelism (ILP) of the executed code. For example, Intel's Core architecture shelters 3 ALUs and 2 SSE units [18] and IBM's Power7 includes 2 fixed-point units, 2 load-store units and 4 floating-point pipelines per core [19]. Thereby, it is up to the instruction scheduler and its policy which units to use. However, the degree of ILP is often less than the number of available execution units (EUs). Hence, different scheduling policies can influence the workload distribution and thereby the amount of time a unit is under stress or active. Given this fact, the scheduling techniques described in Table IV are investigated if they can slow down aging. The differences between the five scheduling policies are thereby also illustrated in Table IV, for an example with 2 execution units and an execution time of 1 cycle per instruction.

V. Experimental Results

In this section the experimental results of clock and power gating as well as instruction scheduling under the scope of aging mitigation are discussed.

A. Evaluation Setup

The evaluated processor runs at 3 GHz and has one core based on the architecture of the Alpha 21264 [22], which is similar to modern superscalar architectures. Since this work focuses on finding out how aging affects the execution units, the evaluation of a single-core is sufficient. The modeled core accommodates 2 ALUs for logic operations and integer add/sub instructions, 2 CALUs for fix-point mul/div instructions and 2 FPUs for floating-point operations. Further details of the processor configuration can be found in Table V. The fabrication technology is a 32 nm node with a supply voltage of 1.0 V and the initial temperature is 57 °C. Furthermore, a delay degradation of 10% in 3 years ($\delta_B = 1$, $\alpha_B = 1$, $T_B = 90$ °C) due to HCI and NBTI is assumed [4], [20]. Since the critical delay degradation is set to be 10%, the MTTF under these conditions is 3 years. Please note that under typical applications the temperature never comes close to 90 °C. Hence, under typical workloads the MTTF is longer even if no aging mitigation techniques are applied.

The applied workloads are part of the SPEC2000 benchmark suite. Overall 6 integer and 7 floating point benchmarks, with a runtime of 0.5 seconds each, are used. This runtime does not include the initialization phase of each benchmark, which is executed but not included in the measurements. Based on the parameter behavior during the runtime the wearout after a certain amount of time is predicted.

If not mentioned otherwise, in the following Subsection V-B to Subsection V-F the performance is given as the average MIPS (million instructions per second) count over all executed

Policy	Description	Cycle: Instruction(s):	Cycle 1 A	Cycle 2 B,C	Cycle 3 D,E	Cycle 4 -	Cycle 5 -
Stat1	Always use the same EU		EU1 (A)	EU1 (B)	EU1 (C)	EU1 (D)	EU1 (E)
Stat2	Always use the same EU If this one is not available[1] take another one		EU1 (A)	EU1 (B) EU2 (C)	EU1 (D) EU2 (E)		
CQ1	Change EU with every new instruction If unit is not available try next one		EU1 (A)	EU2 (B) EU1 (C)	EU2 (D) EU1 (E)		
CQ2	Change EU only, if it is not available		EU1 (A)	EU1 (B) EU2 (C)	EU2 (D) EU1 (E)		
CQ3-X	Always use the same EU Change EU every X cycles (here: $X = 3$)		EU1 (A)	EU1 (B)	EU1 (C)	EU2 (D)	EU2 (E)

TABLE IV
Scheduling policies and how they influence which execution unit (EU) executes which instruction (A,. . . ,E)([1]: busy, power gated)

Processor	Single-core @ 3 GHz, out-of-order, 4-issue
L1-Cache	64 KByte, 2-way, 64 Byte line, 3 cyc latency
L2-Cache	2 MByte, 16-way, 64 Byte line, 15 cyc latency
Execution Units	2x ALU, 2x CALU, 2x FPU
Expected wearout	$\Delta^{rel}d$ = 10% in 3 years, i.e MTTF = 3 years ($\delta_B = 1$, $\alpha_B = 1$, $T_B = 90°$C)
Conditions	T_{start} =57 °C, $V_{dd} = 1.0$ V $V_{th} = 0.21$ V, $t_m = 100$ μs
Power gating	power down 7 ns, power up 3 ns
SPEC2000 benchmarks	applu, bzip2, equake, gcc, gzip, lucas, mcf, mesa, mgrid, parser, swim, twolf, wupwise

TABLE V
CONFIGURATION DETAILS FOR THE EXPERIMENTS

benchmarks and the power consumption (dynamic + static) is the average over all workloads and includes all execution units. Please note that the power consumption for the power gating controller is negligible (i.e. less than 0.3 mW) and hence does not affect the shown results [34]. The determined relative delay change $\Delta^{rel}d$ is always the worst possible value for a runtime $t = 3$ years. For HCI induced aging we use the microarchitectural model from Equation (8) and for NBTI we use the model from Equation (7). Please note that $\Delta^{rel}d$ caused by NBTI is equivalent to the $\Delta^{rel}d$ of PMOS transistors and aging due to HCI corresponds to the wearout of NMOS transistors.

B. *Effect of Clock/Power Gating Techniques*

Usually clock and power gating techniques are used to reduce the power consumption of a microprocessor. However, as explained in Section IV-A, both techniques can be used also to reduce aging induced by NBTI and HCI. To show the potential of these techniques, the results obtained with the native scheduling technique (CQ1) are presented in the following. Here a power gating strategy, with $t_{idle} = 50$ cycles and $t_{dur} = 0$ cycles is used.

Clock gating is extremely helpful to mitigate HCI. The results illustrated in Figure 4 show that the worst $\Delta^{rel}d$ (additional relative delay due to aging) over all execution units and all workloads is just 55% of the original value without using any optimization techniques. However, the benefit for the entire unit is much smaller. Since clock gating without input vector control has only a second order mitigation effect on NBTI (only due to reduced temperature), $\Delta^{rel}d$ due to NBTI is just reduced by 2%. Hence, PMOS transistors age much faster ($\Delta^{rel}d_{HCI}$ = 3.8% after 3 years, $\Delta^{rel}d_{NBTI}$ = 9.3% after 3 years).

To reduce NBTI-induced wearout, power gating is better suited. However, the efficiency of power gating strongly depends on the duty cycle of the transistors. If the duty cycle is very high (i.e. $\delta > 0.9$), power gating yields a high mitigation of NBTI induced aging (here: more than 30%). If the duty cycle is medium - as we assume (see Section III-B2) - the power gating strategy used here reduces $\Delta^{rel}d$ just by 3%. This is due to the fact that the relationship between NBTI induced aging and the duty cycle is nonlinear, as depicted in Figure 1(a) and that power gating primarily reduces the duty cycle. Moreover, the benefit for HCI-induced wearout is much smaller compared to the benefit obtained by clock gating, since clock gating can be used much more often than power gating. As a result HCI-induced aging is reduced by 4%.

For an efficient mitigation of HCI and NBTI, the combination of clock and power gating is the best choice. However, in this scenario PMOS transistors age again much faster than NMOS transistors ($\Delta^{rel}d_{HCI}$ = 4.4% after 3 years, $\Delta^{rel}d_{NBTI}$ = 9.2% after 3 years) due to the already mentioned fact, that power gating cannot be applied that often compared to clock gating and only the first one has a first order influence on NBTI (see Section IV-A).

Looking at the results from the perspective of power consumption, it shows that the average consumption is already heavily reduced by clock gating (just 35% of the original consumption). Hence, power gating does not provide further noticeable power reduction. On the other hand, the chosen power gating strategy comes along with a performance loss of 13%, which is a major reason why power gating is still not very wide spread.

C. *Effect of Aging-Aware Scheduling*

While clock gating already yields very good mitigation results for HCI induced wearout using the native (CQ1) scheduling technique, power gating is less efficient. Therefore, the instruction scheduler can be optimized, to enhance the aging mitigation potential of power gating. To investigate which instruction scheduling policy is giving the best compromise between high performance and aging reduction of the execution units, the policies Stat1, Stat2, CQ1, CQ2 and CQ3-X with $X = 10^3, \ldots, 10^9$ cycles introduced in Section IV-C are examined. In these experiments clock and power gating ($t_{idle} = 50$ cycles, $t_{dur} = 0$ cycles) are activated.

As the results in Figure 5 confirm, the best policy for performance optimization is not the best policy for aging. In fact, the CQ3-X policy gives the best compromise of performance and aging. Even power consumption is the best for this technique. Since NBTI is worse than HCI even when CQ3-X is applied, the recommendation is the usage of CQ3-X with $X = 10^6$ or 10^9 cycles. With this setting $\Delta^{rel}d$ due to HCI is reduced by almost 7% compared to the native CQ1 technique. Aging due to NBTI is reduced by more than 35%, while at the same time the performance loss is only 5 to 7%.

The reason why this scheduling technique is better suited than the others is that, it can better unleash the aging mitigation

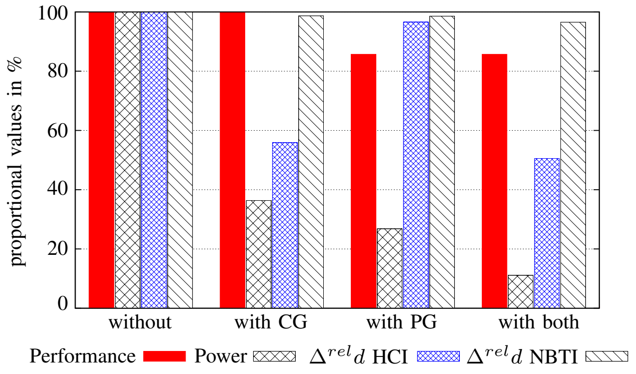


Fig. 4. Effects of clock gating (CG) and power gating (PG)

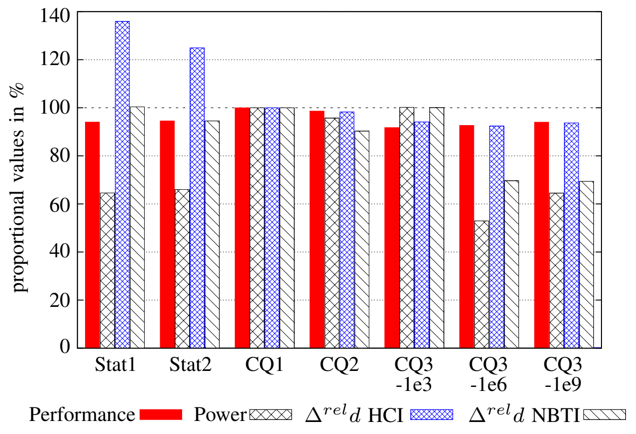


Fig. 5. Effects of diff. scheduling policies (with CG/PG)

potential of power gating. Since the same unit is used for X cycles - and only this unit - the other unit of the same type can be power gated for almost X cycles. Thereby a bigger X is of advantage, because less power down and up phases are necessary, which do not contribute to wearout reduction.

In contrast the static scheduling techniques are not suitable for aging mitigation, since one unit is stressed almost the entire runtime, which leads to a high wearout of this unit. The other unit of the same type instead has almost no wearout, since it is power gated more or less the entire time. This shows that a balanced load is necessary to efficiently slow down aging, which is only given by the CQ-policies.

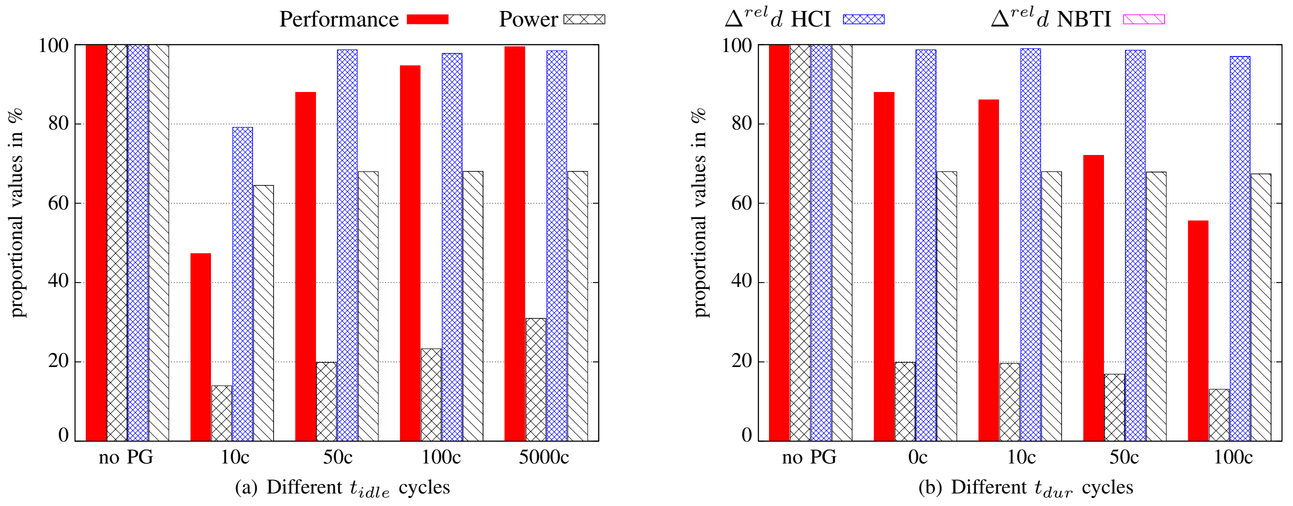
D. *Effect of different Power Gating Strategies*

Using the CQ3-10^9 scheduling policy the performance loss caused by the activation of the power gating strategy, where $t_{idle} = 50$ cycles and $t_{dur} = 0$ cycles, is about 12%. To optimize power gating for achieving good aging mitigation results with minimal performance impact, two basic parameters can be adjusted. First of all, the (idle) time t_{idle} of a unit, before the unit can be power gated, is adjustable. The other parameter is the minimum power gating duration t_{dur}, i.e. the minimum time a unit is power gated, whenever power gating is activated for that unit. In this section, the effects of different settings for both parameters are investigated. In the following experiments the scheduling technique CQ3-10^9 is used and clock gating is always applied, hence in all configurations NBTI induced wearout is worse than that due to HCI.

In Figure 6(a) and Figure 6(b) the results for different settings are illustrated with $t_{dur} = 0$ cyc. for the different t_{idle} settings and $t_{idle} = 50$ cyc. for the different t_{dur} settings. Since an increasing t_{idle} or a decreasing t_{dur} means, that power gating is activated less often, the performance is improved. On the other hand this leads also to faster wearout. However, the performance benefits are bigger than the aging deficits. Hence, the best choice by looking at aging and performance is a high t_{idle} and a small t_{dur}. The benefit for HCI is smaller than that for NBTI. However, since NBTI is worse in all configurations, the benefits for NBTI reduction are more important.

In the *"preferred" configuration* ($t_{idle} = 5000$ cyc., $t_{dur} = 0$ cyc.), which is a combination of high performance and slow aging, power gating yields a reduction of 32% for $\Delta^{rel}d$ caused by NBTI. Thereby the performance loss is less than 0.5% compared to a configuration without power gating. However, this configuration does not provide large benefits to mitigate HCI. In absolute terms this means, that after 3 years $\Delta^{rel}d$ is 6.3% due to NBTI and is 4.1% due to HCI. Therefore, the lifetime (MTTF) of the execution units can be prolonged by more than 4 times compared to a solution without power gating, i.e. the critical $\Delta^{rel}d$ induced by NBTI exceeds 10% after 16 years, while without power gating this takes less than 4 years.

Furthermore, one should note that the average power consumption for all the shown strategies is very low ($<$ 0.4 Watt for all execution units together). However, the "preferred" configuration has the highest power consumption, which shows that optimizing for power and optimizing for aging with minimal performance impact do not always go hand in hand.



(a) Different t_{idle} cycles

(b) Different t_{dur} cycles

Fig. 6. Effect of various power gating strategies (with clock gating) on performance, power and aging

Configuration		Native	Native & CG	Native & CG & PG	Preferred
Average Performance [MIPS]		**2329**	**2329**	1996	2234 (-4%)
Average Power (all ex. units) [Watt]		3.62	1.32	0.40	**0.40** (-89%)
$\Delta^{rel}d$ due to HCI	3 years [%]	8.7	4.9	3.8	**4.1** (-63%)
	5 years [%]	11.5	6.4	5.8	**5.4** (-63%)
	7 years [%]	13.8	7.6	6.9	**6.4** (-64%)
$\Delta^{rel}d$ due to NBTI	3 years [%]	9.5	9.3	9.2	**6.3** (-34%)
	5 years [%]	10.9	10.8	10.5	**7.3** (-33%)
	7 years [%]	11.9	11.8	11.5	**7.9** (-34%)
MTTF (lifetime)		$<$ 4 years	$<$ 4 years	$<$ 4 years	$>$ **16 years** (+4x)

TABLE VI
NATIVE SOLUTION (CQ1, NO CG/PG) VS. "PREFERRED" CONFIGURATION (CQ3-10^9, CG, PG: $t_{dur} = 0, t_{idle} = 5000$)

E. Combination of Clock/Power Gating and Scheduling

Putting the advantages of clock gating, different power gating strategies and the aging-aware scheduling techniques together, we obtain the best compromise among performance, power consumption and aging. *Under the made assumptions, the CQ3-*10^9 *scheduling policy combined with clock gating and the power gating strategy with* $t_{idle} = 5000$ *cyc. and* $t_{dur} = 0$ *cyc., is the "preferred" solution, since this configuration yields the best compromise between performance, power and wearout.*

As illustrated in Table VI, with the "preferred" solution $\Delta^{rel}d$ of NMOS transistors is reduced by 63%. For PMOS transistors a reduction of 34% compared to the native solution is achieved. For this reason, after 3 years $\Delta^{rel}d$ due to HCI is just 4.1% and due to NBTI just 6.3%. In terms of lifetime, even after 16 years $\Delta^{rel}d$ neither caused by NBTI nor by HCI exceeds the 10% threshold, i.e. the MTTF is improved by a factor of 4. Moreover, power consumption is just around 0.4 Watt, which is an improvement of 89%. All these come at the cost of only 4% performance (MIPS) reduction.

Furthermore, due to the huge aging reduction, the guardbands can be reduced, which gives additional headroom to increase the frequency to compensate the implied performance losses. For example, for a 7 year lifetime the guardband has to be 13.8% for the original technique, but just 7.9% for the preferred solution. Hence, the frequency can be increased, which can be used to (partly) compensate the performance loss. In this example, it is necessary that the processor can achieve a clock frequency of 3.42 GHz (clock period = 292 ps) at $t = 0$ to ensure that it can run with 3 GHz (clock period = 333 ps) the entire 7 years. Instead, with the "preferred" solution the processor can be clocked with 3.17 GHz[1] (clock period = 315 ps, $\Delta^{rel}d_{HCI} \leq 6.4\%$, $\Delta^{rel}d_{NBTI} \leq 7.9\%$) for at least 9 years, which leads to an average MIPS value of 2345. Hence the performance loss of the "preferred" configuration can be (over)compensated by the means of higher clock frequencies, while the lifetime (MTTF) is still longer compared to the native configuration.

F. Application Dependencies

The improvements of the "preferred" configuration compared to the native solution (CQ1 scheduling, no clock/power gating) described in the previous section refer only to average values for performance and power or worst-case values for aging. However, the efficiency of the chosen mitigation techniques and aging itself are strongly application dependent. Therefore, we also investigated the improvements of the "preferred" configuration for each workload separately. The corresponding relative results (i.e. improvements = 1-preferred/native) are depicted in Figure 7. Please note that the performance changes are negative to indicate that the performance decreases compared to the native solution.

The improvements for HCI-induced wearout range from 53% to 76% and the benefits for aging due to NBTI range from 33% to 41%. At the same time power consumption can be in best case reduced by 97%, while the performance impacts only range between nothing and 12.5%. An application that fits very well to the chosen techniques is thereby for example the *mcf* workload in which power consumption is reduced by 97%, HCI is mitigated by 75% and NBTI by 39%, while the performance impact is negligible.

This rises the question, what can be done at software-level to achieve low aging rates on a processor setup that is similar to our "preferred" configuration. A crucial point for a high efficiency of our proposed techniques are the cycles during the execution of an application, in which an execution unit is doing nothing, i.e. it is idle. Particularly with regard to the efficiency of power gating it is better to have less idle periods, which are indeed very long, than to have many short idle periods

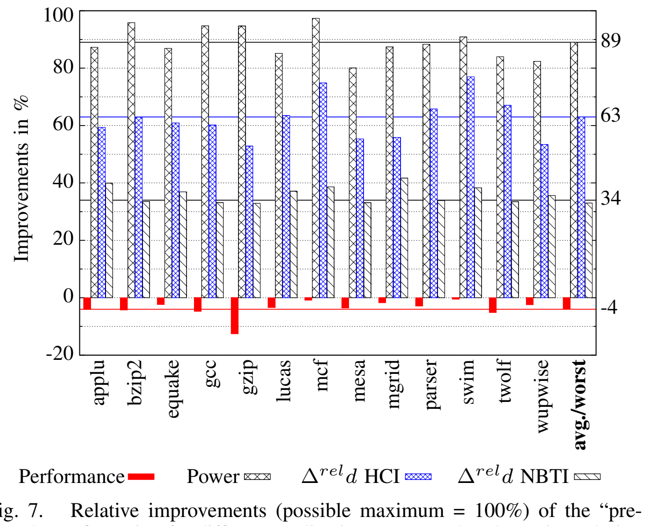
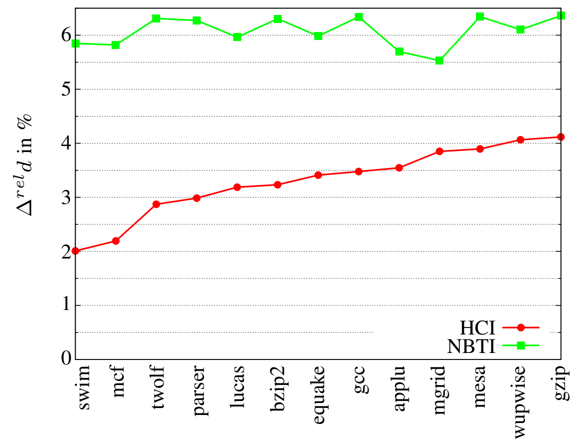
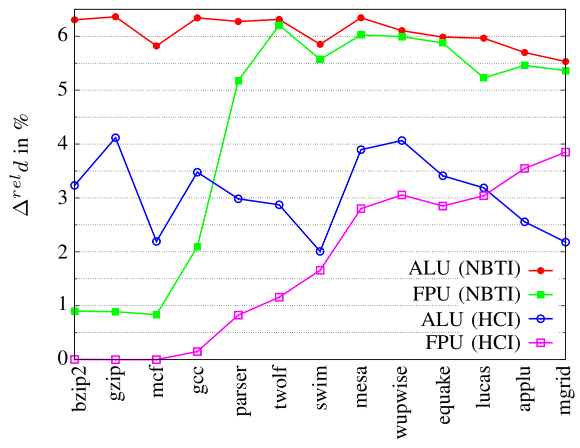


Fig. 7. Relative improvements (possible maximum = 100%) of the "preferred" configuration for different applications compared to the native solution (CQ1, no CG/PG)

[1]higher temperature, i.e. faster wearout, due to increased frequency is respected



(a) Worst $\Delta^{rel}d$ for NBTI and HCI for several workloads



(b) $\Delta^{rel}d$ for ALU and FPU (NBTI and HCI) for several workloads

Fig. 8. Influence of various applications on NBTI- and HCI-induced transistor wearout (using the "preferred" configuration)

that are very short. For example, in case of our "preferred" configuration power gating is activated after 200 idle cycles. Hence, it is better to have just one idle period with 300 cycles than three periods with just 100 cycles each. Good application examples for this observation are the *bzip2* and *lucas*. Both have nearly the same ratio of idle to total cycles for the ALU0, however the ratio of power gated cycles in *bzip2* is much smaller than in *lucas* leading to higher aging rates due to NBTI (see Figure 8(a)). This means for the software development that it is better in terms of aging to group operations of the same type (i.e. ALU, FPU, memory accesses) together as much as possible to create longer idle periods (more effective power gating) for other blocks.

Beside the relative improvements also the absolute aging rates for each application are very interesting. In Figure 8(a), the influence of several workloads on NBTI and HCI induced wearout is depicted, in which the "preferred" aging mitigation configuration of Section V-E is used. As one can see, HCI and NBTI do not always follow the same trend, although, aging caused by NBTI is always worse than wearout due to HCI. Indeed if the NBTI effect in application B is smaller than in application A, it is not necessarily the same for HCI. This is thereby mainly due to the chosen mitigation techniques, especially clock and power gating. If the number of power gated cycles in application B is higher than that in application A, the NBTI effect can be smaller in B than in A. If at the same time the total amount of idle cycles (power gated or clock gated) in B is lower than in A, the HCI effect in B can be higher than in A. Hence the amount of idle time and its distribution during the execution of the workload has a high influence not only on NBTI and HCI, but also on their ratio.

This effect can be seen also in Figure 8(b), in which the aging for the ALU and FPU induced by HCI and NBTI is illustrated. Furthermore, the obtained results show another interesting phenomenon. Although in the *applu* and *mgrid* benchmarks, the instruction ratio for the FPU is much higher (up to 3x) than for the ALU, the NBTI-induced aging of the ALU is faster than that of the FPU. This is due to the fact, that the ALU can be power gated less often. In contrast, since HCI can be mitigated using clock gating, and clock gating can be applied every time the unit is idle, HCI-induced aging of the ALU is less than that of the FPU. The not-depicted CALU ages much slower (NMOS and PMOS) than the ALU or FPU, since the load for this unit is much lower than for the other units. For all types of execution units both units, e.g. ALU0 and ALU1, age at the same rate, because of the chosen scheduling technique, similar load and similar temperatures.

The results shown in Figure 8 underline also the statement, that transistor usage needs to be modeled in order to have accurate wearout estimation results. If the usage is not considered, all applications would lead to a similar wearout.

VI. Conclusion

Microprocessors at nano-scale are exposed to various reliability issues, which include more rapid aging of all components. Physical effects like *Negative Bias Temperature Instability* (NBTI) and *Hot Carrier Injection* (HCI) cause a shift of the transistors threshold voltage, which manifests in increasing path delays. This eventually leads to slower devices and reduces MTTF.

To model and analyze wearout due to NBTI and HCI at microarchitecture-level, this paper presented a novel microarchitectural framework *ExtraTime* and its integrated tool set containing a performance simulator combined with microarchitectural models for power, temperature and particularly aging. With this setup ExtraTime does not only enable designers to investigate aging mitigation techniques at hardware-level, but also at software-level (application). Furthermore, ExtraTime can be used very early in the design process of a microprocessor, enabling design space exploration for performance, power consumption, temperature and wearout.

Using this framework, we investigated various clock and power gating strategies in combination with several instruction scheduling policies for aging mitigation with minimal performance and power impacts. The simulation results show that using these techniques together lifetime (MTTF) of the execution units of a 32 nm superscalar microprocessor can be extended by 4 times. At the same time performance is only decreased about 4%. Alternatively some of the gained headroom can be used to increase the frequency by relaxing the guardbands to overcompensate this performance loss.

REFERENCES

[1] J. Abella, X. Vera, and A. Gonzalez, "Penelope: The NBTI-Aware Processor," in *Proc. of the Int'l Symp. on Microarchitecture*. IEEE Computer Society, Dec. 2007, pp. 85–96.

[2] M. Agarwal, B. C. Paul, M. Zhang, and S. Mitra, "Circuit Failure Prediction and Its Application to Transistor Aging," in *Proc. of the VLSI Test Symp.* IEEE Computer Society, May 2007, pp. 277–286.

[3] M. Basoglu, M. Orshansky, and M. Erez, "NBTI-Aware DVFS: A New Approach to Saving Energy and Increasing Processor Lifetime," in *Proc. of the Int'l Symp. on Low Power Electronics and Design*. ACM, Aug. 2010, pp. 253–258.

[4] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer, "High-performance CMOS variability in the 65-nm regime and beyond," *IBM Journal of Research and Development - Advanced silicon technology*, vol. 50, pp. 433–449, July 2006.

[5] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, "The M5 Simulator: Modeling Networked Systems," *IEEE Micro*, vol. 26, no. 4, pp. 52–60, July 2006.

[6] S. Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation," *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Nov.-Dec. 2005.

[7] K. A. Bowman, B. L. Austin, J. C. Eble, X. Tang, and J. D. Meindl, "A physical alpha-power law MOSFET model," in *Proc. of the Int'l Symp. on Low Power Electronics and Design*. New York, NY, USA: ACM, 1999, pp. 218–222.

[8] A. Bravaix, C. Guerin, V. Huard, D. Roy, J. Roux, and E. Vincent, "Hot-Carrier Acceleration Factors for Low Power Management in DC-AC stressed 40nm NMOS node at High Temperature," in *Int'l Reliability Physics Symposium*, April 2009, pp. 531–548.

[9] A. Calimera, E. Macii, and M. Poncino, "NBTI-Aware Power Gating for Concurrent Leakage and Aging Optimization," in *Proc. of the Int'l Symp. on Low Power Electronics and Design*. ACM, Aug. 2009, pp. 127–132.

[10] T. Chan, J. Sartori, P. Gupta, and R. Kumar, "On the Efficacy of NBTI Mitigation Techniques," in *Proc. of the Conf. on Design, Automation and Test in Europe*, March 2011, pp. 1–6.

[11] K.-L. Chen, S. Saller, and R. Shah, "The case of AC stress in the hot-carrier effect," *IEEE Trans. on Electron Devices*, vol. 33, no. 3, pp. 424–426, Mar. 1986.

[12] M. DeBole, R. Krishnan, V. Balakrishnan, W. Wang, H. Luo, Y. Wang, Y. Xie, Y. Cao, and N. Vijaykrishnan, "New-Age: A Negative Bias Temperature Instability-Estimation Framework for Microarchitectural Components," *Int'l Journal of Parallel Programming*, vol. 37, pp. 417–431, Aug. 2009.

[13] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation," in *Proc. of the Int'l Symp. on Microarchitecture*. IEEE Computer Society, 2003, pp. 7–19.

[14] S. Feng, S. Gupta, and S. Mahlke, "Olay: Combat the signs of aging with introspective reliability management," *Ann Arbor*, vol. 1001, p. 48109, 2008.

[15] F. Firouzi, S. Kiamehr, and M. B. Tahoori, "A Linear Programming Approach for Minimum NBTI Vector Selection," in *Proc. of the Great Lakes Symp. on VLSI*. ACM, 2011, pp. 253–258.

[16] E. Gunadi, A. A. Sinkar, N. S. Kim, and M. H. Lipasti, "Combating Aging with the Colt Duty Cycle Equalizer," in *Proc. of the Int'l Symp. on Microarchitecture*. IEEE Computer Society, 2010, pp. 103–114.

[17] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A Compact Thermal Modeling Methodology for Early-Stage VLSI Design," *IEEE Trans. on VLSI Systems*, vol. 14, no. 5, pp. 501–513, May 2006.

[18] Intel, "Intel 64 and IA-32 Architectures Software Developers Manual: Basic Architecture," vol. 1, 2006.

[19] R. Kalla, B. Sinharoy, W. Starke, and M. Floyd, "Power7: IBM's Next-Generation Server Processor," *IEEE Micro*, vol. 30, no. 2, pp. 7–15, March 2010.

[20] K. Kang, S. P. Park, K. Roy, and M. A. Alam, "Estimation of statistical variation in temporal NBTI degradation and its impact on lifetime circuit performance," in *Proc. of the Int'l Conf. on Computer-Aided Design*. IEEE Press, Nov. 2007, pp. 730–734.

[21] J. Keane, X. Wang, D. Persaud, and C. Kim, "An All-In-One Silicon Odometer for Separately Monitoring HCI, BTI, and TDDB," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 4, pp. 817–829, Apr. 2010.

[22] R. E. Kessler, "The Alpha 21264 microprocessor," *IEEE Micro*, vol. 19, no. 2, pp. 24–36, Mar. 1999.

[23] O. Khan and S. Kundu, "A Self-Adaptive System Architecture to Address Transistor Aging," in *Proc. of the Conf. on Design, Automation and Test in Europe*. European Design and Automation Association, 2009, pp. 81–86.

[24] S. Kim, S. V. Kosonocky, and D. R. Knebel, "Understanding and Minimizing Ground Bounce During Mode Transition of Power Gating Structure," in *Proc. of the Int'l Symp. on Low Power Electronics and Design*. ACM, Aug. 2003, pp. 22–25.

[25] R. Kumar and G. Hinton, "A Family of 45nm IA Processors," in *IEEE Int'l Solid-State Circuits Conf. - Digest of Technical Papers*, Feb. 2009, pp. 58 –59.

[26] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *Proc. of the Int'l Symp. on Microarchitecture*. ACM, Dec. 2009, pp. 469–480.

[27] F. Oboril and M. B. Tahoori, "ExtraTime: A Framework for Exploration of Clock and Power Gating for BTI and HCI Aging Mitigation," in *Proc. of Zuverlssigkeit und Entwurf*, Sep. 2011.

[28] F. Oboril and M. B. Tahoori, "Reducing Wearout in Embedded Processors Using Proactive Fine-Grain Dynamic Runtime Adaptation," in *Proc. of the European Test Symp.*, May 2012.

[29] T. Siddiqua and S. Gurumurthi, "A Multi-Level Approach to Reduce the Impact of NBTI on Processor Functional Units," in *Proc. of the Great lakes Symp. on VLSI*. ACM, 2010, pp. 67–72.

[30] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "Lifetime Reliability: Toward an Architectural Solution," *IEEE Micro*, vol. 25, pp. 70–80, May 2005.

[31] E. Takeda, Y. Nakagome, H. Kume, and S. Asai, "New hot-carrier injection and device degradation in submicron MOSFETs," *IEEE Proc. I, Solid-State and Electron Devices*, vol. 130, no. 3, pp. 144–150, June 1983.

[32] A. Tiwari and J. Torrellas, "Facelift: Hiding and slowing down aging in multicores," in *Proc. of the Int'l Symp. on Microarchitecture*. IEEE Computer Society, Nov. 2008, pp. 129–140.

[33] V. Tiwari, D. Singh, S. Rajgopal, G. Mehta, R. Patel, and F. Baez, "Reducing Power in High-performance Microprocessors," in *Proc. of the Design Automation Conf.* ACM, 1998, pp. 732–737.

[34] K. Usami, T. Shirai, T. Hashida, H. Masuda, S. Takeda, M. Nakata, N. Seki, H. Amano, M. Namiki, M. Imai, M. Kondo, and H. Nakamura, "Design and Implementation of Fine-Grain Power Gating with Ground Bounce Suppression," in *Int'l Conf. on VLSI Design*, Jan. 2009, pp. 381–386.

[35] R. Vattikonda, W. Wang, and Y. Cao, "Modeling and Minimization of PMOS NBTI effect for Robust Nanometer Design," in *Proc. of the Design Automation Conf.* ACM, 2006, pp. 1047–1052.

[36] N. J. Wang, J. Quek, T. M. Rafacz, and S. J. patel, "Characterizing the Effects of Transient Faults on a High-Performance Processor Pipeline," in *Proc. of the Int'l Conf. on Dependable Systems and Networks*. IEEE Computer Society, June 2004, pp. 61–71.

[37] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The Impact of NBTI Effect on Combinational Circuit: Modeling, Simulation, and Analysis," *IEEE Trans. on VLSI Systems*, vol. 18, no. 2, pp. 173–183, Feb. 2010.

[38] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, and H. Yang, "On the efficiancy of Input Vector Control to mitigate NBTI effects and leakage power," in *Proc. of the Int'l Symp. on Quality of Electronic Design*. IEEE Computer Society, Mar. 2009, pp. 19–26.