Employing Circadian Rhythms to Enhance Power and Reliability

Saket Gupta, Broadcom Corporation
Sachin S. Sapatnekar, University of Minnesota, Twin Cities

This paper presents a novel scheme for saving architectural power by mitigating delay degradations in digital circuits due to bias temperature instability (BTI), inspired by the notion of human circadian rhythms. The method works in two alternating phases. In the first, the compute phase, the circuit is "awake" and active, operating briskly at a greater-than-nominal supply voltage, which causes tasks to complete more quickly. In the second, the idle phase, the circuit is power-gated and "put to sleep," enabling BTI recovery. Since the wakeful stage works at an elevated supply voltage, it results in greater aging than operation at the nominal supply voltage, but the sleep state involves a recovery that more than compensates for this differential. We demonstrate, both at the circuit and the architectural levels, that at about the same performance, this approach can result in appreciable BTI mitigation, thus reducing the guardbands necessary to protect against aging, which results in power savings over the conventional design.

Categories and Subject Descriptors: B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids; C.1.0 [**Processor Architectures**]: General

General Terms: Digital Circuits, Architectures, Reliability, Low Power

Additional Key Words and Phrases: BTI, power dissipation, low-power design, aging guardbands, supply voltage, power gating

ACM Reference Format:
Gupta, S. and Sapatnekar, S. S. 2012. Employing Circadian Rhythms to Enhance Power and Reliability. ACM Trans. Des. Autom. Electron. Syst. V, N, Article A (January YYYY), 23 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

1.1. Background and Motivation

With the continued scaling of CMOS technology, the demand for low power consumption in circuits and computer systems has risen sharply. Increased on-chip power due to switching and leakage can have numerous undesirable effects [Zhan et al. 2008], and techniques for achieving reliable, low-power operation have therefore become a critical issue in the design flow.

Amongst the various factors that add to the power dissipation of a chip, one of the major contributors is the design overhead for ensuring functional correctness over its lifetime. The presence of variations and various aging effects causes changes of the chip frequency, and requires delay guardbanding to ensure that the T_{clk} specification is met throughout the lifetime of the chip. In case of aging, this design overhead can increase the power of various circuits on the chip by about 30% in the 45nm regime [Kumar et al. 2011]; this becomes increasingly significant at more deeply scaled technology nodes. Reducing or removing such design overhead is an important component of low power design.

The most major component of aging in digital circuits is attributable to negative/positive bias temperature instability (NBTI/PBTI) in PMOS/NMOS devices; collectively, these effects are referred to as BTI. In a CMOS gate, when a PMOS (NMOS) device is stressed under BTI, typically by applying a logic 0 (logic 1) at its gate input, its threshold voltage degrades, resulting in a possible increase in the gate delay. When the stress is removed, there is partial (but not complete) recovery in the threshold voltage, and hence the delay degradation is partially ameliorated.

Various approaches have been proposed to overcome this degradation. As discussed above, some methods introduce delay guardbands using sizing or resynthesis [Kumar et al. 2007] to add a delay margin to the nominal ($t = 0$) design. Since this can incur a significant overhead, other methods have also been pursued to mitigate/compensate for these effects and reduce the design overhead. At the circuit level, adaptive body bias and adaptive supply voltage schemes [Chen et al. 2009; Kumar et al. 2011] compensate for BTI degradation by dynamically increasing the values of V_{dd} and V_{bb} voltages to speed up the circuit. Since the optimum for each circuit block may be different, this could involve the generation of a large set of V_{dd} and V_{bb} values, which poses a significant challenge. Such a solution requires a voltage control system to supply different values of V_{dd} and V_{bb} to different circuit blocks. Implementing these multiple values at the architectural level, with a small number of chip-level supply voltage and body-bias voltage regulators, is a significant challenge.

Chip-level dynamic voltage scaling (DVS) schemes [Srinivasan et al. 2005; Abella et al. 2007; Tiwari and Torrellas 2008; Karpuzcu et al. 2009; Zhang and Dick 2009] to recapture lost performance overcome this problem by dynamically varying the supply voltage at the processor level. These methods also mitigate BTI by managing the workload amongst multiple cores. Some of these schemes consider optimization of both reliability and power. The DVS scheme in [Karpuzcu et al. 2009] minimizes architectural power with a BTI-aware multicore design, working with a set of throughput and expendable cores: the throughput cores can gain lost performance due to degradation, while the expendable cores are utilized for reducing power consumption. Extending the concept of DVS to dynamic voltage and frequency scaling (DVFS), [Chen et al. 2009; Basoglu et al. 2010] propose to vary both the supply voltage and frequency for minimizing active and leakage energy, while meeting the target timing specification throughout lifetime. The approach in [Mintarno et al. 2010] uses a dynamic fine-grained self-tuning mechanism that changes the amount of cooling, voltage, and clock frequency to optimize on both the lifetime of the part and the total energy consumed over lifetime. Such mechanism is designed to work adaptively depending on the type of workload on a processor at a certain time. Various schemes for power-penalty minimization and aging-related error-resilience are described in [Mitra et al. 2011] and consider multiple layers of operation for a synthetic analysis: from devices and circuits to architectures and applications.

Many of the chip level schemes and those that optimize power and reliability, permit the use of either an analog change or a 5-10mV step change in the supply voltage. To achieve this at the architectural level is itself a challenge. Firstly the circuitry required to achieve such a fine grained control over V_{dd} is quite expensive in terms of area and power (for example, requiring highly accurate charge pumps with large number of capacitors and transistors, and the associated DC-DC converters). Secondly, even a slight IR-drop of 5-10mV is sufficient to offset the functioning of this circuitry. The approach proposed in this paper adopts a standard voltage scaling scheme with 100mV resolution.

Another set of schemes for addressing BTI delay degradation are state-based schemes, that detect the idle states of the circuit during computation [Shin et al. 2008; Li et al. 2010], and apply a suitable recovery mechanism to lower the degradation. Other methods in this class distribute tasks over partitioned functional units to balance aging [Siddiqua and Gurumurthi 2010] and perform node vector control [Bild et al. 2009] or power-gating [Calimera et al. 2010] during idle times.

Such idle state approaches have some common limitations. First, the idle states are dependent on the workload/circuit configuration: the precise idle times tend to be unpredictable, or difficult to predict dynamically. Hence, the schemes require a complex hardware/software control mechanism that can (a) dynamically detect the idle times during execution, (b) apply the appropriate recovery mechanism, and (c) keep track of which parts of the circuit have partially recovered after the idle time, and by how much. Second, it may not be easy to exploit such idle times fully, since modern out-of-order execution and multi-threading endeavor to hide idle periods. A better approach would be to have predictable idle times of fixed durations, requiring a potentially much simpler control mechanism.

1.2. Circadian Rhythms for Circuits

In this work, we employ an entirely new approach to reduce circuit aging. We propose Greater-than-NOMinal Operation (GNOMO)[1], a novel and superficially counter-intuitive scheme for mitigating BTI that goes against the conventional wisdom that operation at a higher V_{dd} will result in a higher delay degradation and higher power dissipation. On the contrary, we show that by elevating V_{dd} to an optimal, greater-than-nominal value, we can achieve both lower delay degradation and power dissipation than that incurred at the nominal V_{dd}, at roughly constant performance.

Our ideas are inspired by the notion of human circadian rhythms of wakefulness and sleep. A human is likely to age more quickly without adequate rest, and we show that a similar argument can be made for an inanimate circuit. The normal way to exercise a circuit is to subject it to a nominal supply voltage, $V_{dd,n}$, throughout its lifetime. Under our scheme, we apply a *greater-than-nominal* voltage, $V_{dd,g} > V_{dd,n}$, interspersed with predictable periods of sleep (i.e., power-gating), to reduce aging effects. Intuitively, just as a human uses sleep to recover from fatigue, and can operate at greater intensity after adequate sleep, a circuit can also recover from BTI while it is "asleep." Thus, it can operate at the larger supply voltage value, $V_{dd,g}$, and yet age *less* than the scenario where it is constantly "awake" and subjected to the $V_{dd,n}$ voltage under nominal operation.

For a given nominal V_{dd}, this paper develops a procedure that allows the static determination of the optimal $V_{dd,g}$ for a circuit. We show that the GNOMO scheme can result in enhanced reliability and lower aging. This reduction, as well as other aspects of GNOMO, can then be parleyed into a reduction in the power overhead of guardbanding a circuit against aging degradation. We exercise the GNOMO approach from the circuit level to the architecture level, and show how power savings can be achieved through a practical adoption scheme that is applied up to the architectural level. Specifically, we show in Sections 6.1 and 6.2 that GNOMO enables a reduction of about 1.3× to 1.8× in delay degradation, for various values of $V_{dd,n}$ considered in this paper. For the same lifetime, this reduction in degradation implies that reduced guardbands are necessary as compared to the nominal voltage case. This yields a reduction of about 1.8× to 3.2× in area overhead and about 1.5× to 3.1× in the power overhead.

GNOMO does not require fine-grained voltage supplies/control. Nor does it require the detection of idle times (or the potentially complex associated circuitry) since the idle times are *generated*, and not *detected*, and are hence predictable-by-construction. Further, the idle times are orthogonal to those that dynamically occur during workload execution (due to cache misses, branch mispredictions, etc.). Moreover, they do not depend on a precise characterization of signal probabilities, as is the case for other approaches: characterizing BTI aging based on signal probabilities is inherently unreliable, in that probabilities represent an often unpredictable average rather than a worst case[2]. The use of predictable idleness, on the other hand, provides safe, correct-by-construction, guarantees on the amount of recovery.

The remainder of the paper is organized as follows. We first present the preliminaries for this work in Section 2. Sections 3 and 4 then present the framework and ar-

[1]Part of this work was originally published in [Gupta and Sapatnekar 2012].
[2]While such averages are useful in working with the "softer" constraints associated with power dissipation, they are much more unreliable for the "harder" constraints involved in timing.

chitectural implementation for GNOMO, and are followed by an analysis of the power dissipation under GNOMO in Section 5. We present our results and conclusion in Sections 6 and 7.

2. PRELIMINARIES

2.1. BTI Modeling

We work with a widely adopted model [Bhardwaj et al. 2006; Wang et al. 2007; DeBole et al. 2009] for predicting delay degradation due to BTI. We present an expression for PMOS NBTI under alternate stress/relax cycles, for a given V_{dd} and signal probability α at the input of the PMOS (for PBTI, similar equations may be used since the mechanism of NMOS delay degradation is similar to that of PMOS, albeit with a lower degradation magnitude [Kumar et al. 2011]):

$$\textit{Stress: } \Delta V_{th}(t) = \left(K_v\sqrt{t-t_0} + \sqrt[2n]{C(t-t_0)}\right)^{2n} \tag{1}$$

$$\textit{Recovery: } \Delta V_{th}(t) = \Delta V_{th}(t_1)\left(1 - \frac{2\xi_1 t_e + \sqrt{\xi_2 C(t-t_1)}}{2t_{ox} + \sqrt{Ct}}\right) \tag{2}$$

where Equations (1) and (2) model the all-stress and all-recovery modes. Here, $\Delta V_{th}(t)$ is the degradation in the threshold voltage with time t, with the beginning of the stress and relaxation phase being marked by time points t_0 and t_1, respectively. The precise definitions/values of the symbols are described in [Bhardwaj et al. 2006; Wang et al. 2007], and listed in Table I, where the standard symbols as q, t_{ox}, etc., have their usual meaning and T is the temperature of operation.

This is extended to build a long-term model that predicts the envelope of the BTI degradation pattern with alternating stress and recovery:

Long-term model:

$$\Delta V_{th}(t) = \frac{(K_v^2 \alpha T_{clk})^n}{(1-\beta_t^{1/2n})^{2n}};\ \beta_t = 1 - \frac{2\xi_1 t_e + \sqrt{\xi_2 C(1-\alpha)T_{clk}}}{2t_{ox} + \sqrt{Ct}} \tag{3}$$

where α is the duty cycle of operation with a clock period of T_{clk}.

Table I. Model paramater details

K_v	$\left(\frac{qt_{ox}}{\epsilon_{ox}}\right)^3 K^2 C_{ox}(V_{gs}-V_{th})\sqrt{C}exp\left(\frac{2E_{ox}}{E_0}\right)$		
t_e	t_{ox}		$t-t_1 \geq t'$
	$t_{ox}\sqrt{\frac{t-t_1}{t'}} - \frac{\sqrt{\xi_2 C(t-t_1)}}{2\xi_1}$		otherwise
E_{ox}	$\frac{V_{gs}-V_{th}}{t_{ox}}$	C	$T_o^{-1}\cdot exp(\frac{-E_a}{kT})$
E_a (eV)	0.49	E_0 (V/nm)	0.335
K (C$^{-0.5}$·nm$^{-2.5}$)	8e4	t' (ms)	1.5
ξ_1	0.9	ξ_2	0.5
T (°C)	105	t_{ox} (nm)	0.75
T_o (s/nm^2)	1e-8		

From this model, it is important to note that the exponent n = 1/6, and that K_v (and hence $\Delta V_{th}(t)$) is a superlinear function of V_{dd} (as V_{gs} can assume values of either 0V, $-V_{dd}$, or V_{dd}, during the stress and relax phases, depending on the bias).

Model Usage: In the compute phase, the processor undergoes repeated stresses and relaxations, and its degradation can be captured by the long term model in Equation (3). During the sleep phase, the processor is idle and the recovery of its components can be determined by the relaxation model of Equation (2).

2.2. Delay and Power Modeling

For the circuits considered in this paper, as in past research, we use compact sensitivity-based performance models for the delay (D) and the logarithm of the leakage power ($\log L$) in terms of V_{th} [Kumar et al. 2011]. For $\mathcal{X} \in \{D, \log L\}$, we characterize

$$\mathcal{X}(t) = \mathcal{X}_0 + \sum_{i=1}^{n} \frac{\partial \mathcal{X}}{\partial V_{th_i}} \Delta V_{th_i}(t) \tag{4}$$

where $\partial \mathcal{X} / \partial V_{th_i}$ denotes the sensitivity of the quantity $\mathcal{X}$ with respect to the V_{th} of the i^{th} transistor along the input-output path.

It is also useful, for the discussion to follow, to understand the trends of dynamic and leakage power of a device in the circuit, with changes in V_{dd} [Jejurikar et al. 2004]:

$$P_{AC} = C_{eff} V_{dd}^2 f \quad \text{and} \quad L = K_3 V_{dd} e^{K_4 V_{dd}} \tag{5}$$

where P_{AC} and L are the dynamic and leakage power, C_{eff} the effective loading on the device, and f is the frequency of operation. K_3, K_4 are technology dependent modeling constants and can be determined through best-fit curves of HSPICE simulation data. It should be noted that dynamic and leakage power are a quadratic and exponential functions of V_{dd}, respectively.

3. GNOMO: GREATER-THAN-NOMINAL V_{DD} OPERATION

We now present the GNOMO framework for BTI mitigation. As defined earlier, the term $V_{dd,n}$ refers to the nominal supply voltage and $V_{dd,g}$ to the GNOMO value.

3.1. Circuit Recovery through Power Gating

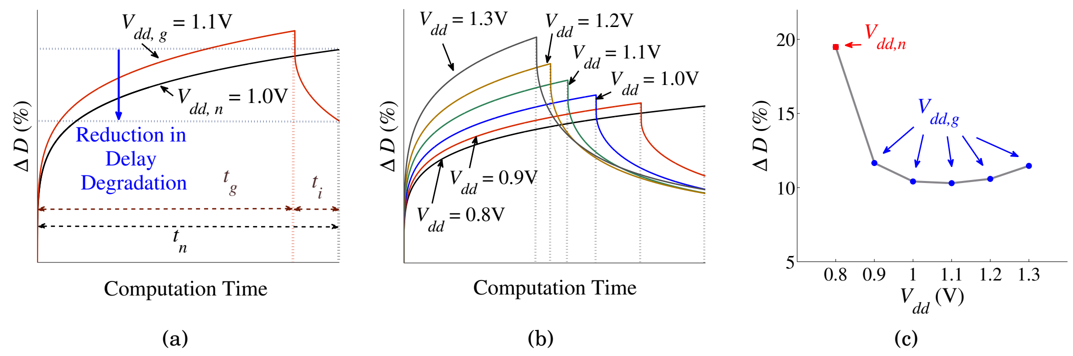


Fig. 1. The delay degradation patterns of MCNC benchmark alu4 at (a) nominal supply voltage $V_{dd,n}$ = 1.0V and greater-than-nominal supply voltage $V_{dd,g}$ = 1.1V, (b) V_{dd} $\in$ [0.8V, 1.3V] values, and (c) delay degradation for alu4 at time t_n in (b).

3.1.1. Motivating Intuition. We illustrate the idea of GNOMO through the example of an ALU circuit, the MCNC benchmark alu4. We define the delay degradation, ΔD, as the increase in circuit delay due to BTI effects, and express it as a percentage of the nominal delay, $D(0)$, at time 0. The curve marked "$V_{dd,n}$" in Fig. 1(a) shows the temporal change in ΔD (%) for alu4 when the circuit is operated at a nominal supply voltage, $V_{dd,n}$ = 1.0V. The time required to complete the ALU computation is denoted as t_n. Note that monotone degradation under stress shown here captures the effect of alternate stress/recovery cycles and plots the *envelope* of BTI degradation [Bhardwaj et al. 2006].

Under GNOMO, at a higher supply voltage $V_{dd,g}$ (chosen to be 1.1V in this figure), the ALU has a lower delay and completes the same computation in time $t_g < t_n$. To maintain the same throughput as the $V_{dd,n}$ case, in principle the data may be latched at time t_g, and the circuit could then be power-gated[3] during an idle time, $t_i = t_n - t_g$. During the idle time, the circuit recovers from BTI degradation, as shown in the figure. The net result is that at time t_n, the BTI degradation for GNOMO is lower than that under the nominal supply voltage.

We explore this tradeoff further in Fig. 1(b), for the case where a different baseline voltage, $V_{dd,n} = 0.8$V, is used, and several $V_{dd,g}$ values are considered. A higher value of $V_{dd,g}$ implies greater degradation during the compute period, and a larger idle time. Since the degradation increases superlinearly with the supply voltage, but the idle times increase sublinearly (as will be shown in Section 3.3), it is possible to identify a supply voltage point at which the overall degradation at time t_n is optimized. This is illustrated in Fig. 1(c): as $V_{dd,g}$ is increased, the percentage ΔD first decreases, reaches a minimum at 1.1V, and then increases again.

3.1.2. Mechanism. In reality, it is quite impractical to implement such a scheme in the form described above, where circuits must be put to sleep and woken up within a single clock cycle. The essence of the idea can nonetheless be extended to a realistic framework: at a higher V_{dd} value (corresponding to a higher clock frequency), instead of switching-off/waking-up the circuit within each cycle, we run the circuit for a large number of cycles at a faster-than-nominal clock. This completes a part of the overall computation more rapidly than at the nominal supply voltage/clock frequency, but we maintain the same throughput by introducing idle time, during which the circuit is power-gated and allowed to recover from BTI degradation.

This idea effectively provides the same sleep/wakeup "duty cycle" as in the earlier conceptual exposition, and therefore the same pattern of temporal degradation and recovery. From the notion of frequency independence of BTI [Bhardwaj et al. 2006; Kumar et al. 2006], the degradation/recovery depends on the duty cycle rather than the precise distribution of on/off periods, and therefore this alternative, more practical, formulation results in the same amount of recovery as the conceptual idea presented earlier.

Therefore, a practical implementation of GNOMO works as follows: the processor functions under a *circadian rhythm*, where it is awake and runs at high speed for several (typically, millions of) cycles at the GNOMO supply voltage, $V_{dd,g}$, during the *compute phase*, and then sleeps for some cycles during the *idle phase* where it is power-gated. The circadian cycle is then continued throughout its lifetime.

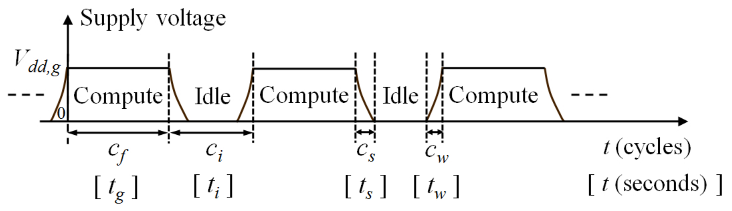


Fig. 2. The compute and idle phases in GNOMO in the practical implementation. This figure is not drawn to scale; in reality, $t_g, t_i >> t_s, t_w$.

[3]For convenience, we will temporarily assume that such a power-gating operation is instantaneous. We will remove this assumption later.

The alternation of the compute/idle phases is depicted in Fig. 2, which shows the supply voltage value along the y-axis and time (in cycles as well as seconds) along the x-axis. We use this figure to introduce some notation that will be used in the remainder of this paper. For a given workload, consider the operation of the processor during a compute phase, corresponding to a fixed number of clock cycles, c_f.

Let the number of instructions committed, while operating at $V_{dd,n}$ ($V_{dd,g}$), be I_n (I_g), and let the corresponding execution time be t_n (t_g) time units. Clearly,

$$t_n = c_f \cdot T_{clk,n} \text{ and } t_g = c_f \cdot T_{clk,g} \tag{6}$$

The duration of the idle phase is denoted by t_i (in seconds) and c_i (in terms of the number of cycles). During this period, the circuits are power-gated and do not perform any computation[4].

The additional costs associated with the idle phase are also illustrated in the figure. Power-gating a circuit incurs an overhead of t_s time units (c_s cycles) for the circuits to transition to the sleep state, and an overhead of t_w time units (c_w cycles) for wakeup. The sleep/wakeup transitions are deliberately designed to occur within the idle phase, ensuring that the execution of instructions is not affected by the GNOMO scheme.

3.1.3. Constraints on the Choice of c_i and c_f. At the chosen value of $V_{dd,g}$, for a specific lifetime goal, a prescribed ratio of t_g to t_i can be calculated. Therefore, if c_i (and hence t_i) is very large, then c_f (and hence t_g) will also be large; conversely if c_i is small, then c_f will also be small. In this section, we will discuss the constraints that place double-sided bounds on the choice of c_i (and hence, on c_f).

Existing power-gating frameworks offer sleep transition times (c_s) of about 10 to 50 cycles for various circuits in a processor [Calimera et al. 2010], while the wakeup time (c_w) is typically about 5-10 cycles. Since these transitions are designed to occur within the idle phase, the effective idle time may decrease significantly if c_s and c_w are comparable to c_i. To amortize the effects of sleep/wakeup transitions, it is necessary to choose c_i to be significantly larger than c_s or c_w. This constraint places a lower bound on the value of c_i that must be used, an issue that is discussed in greater detail in Section 6.4.

Thermal constraints also place an upper bound on the choice of c_i. If a large value of c_i is chosen, then the processor could operate under a high supply voltage, $V_{dd,g}$, for a prolonged period, possibly leading to thermal problems. The value of c_i must be chosen in such a way that t_g is well below the thermal time constant of silicon, so that if the power is relatively unchanged from the nominal case, the alternate compute/idle phases do not affect the peak temperature.

In practice, choosing c_f to be of the order of ten million cycles provides a reasonable balance that meets the double-sided constraints discussed above.

3.2. Idle Time Generation – Practical Considerations

If the frequencies of all the components in a CPU (both on-chip and off-chip components) were to scale at the same rate as V_{dd} is changed, the number of instructions committed in c_f cycles would be the same, i.e., $I_n = I_g$. The idle time $t_{i,1}$ could then be computed as:

$$t_{i,1} = t_n - t_g = c_f \cdot (T_{clk,n} - T_{clk,g}) \tag{7}$$

However, in practice, the voltages and frequencies are scaled up only for on-chip components (processor, cache, on-chip buses, etc.), and the access time for off-chip memory

[4]Note that this idle phase is deliberately inserted and therefore easily predictable, and is thus different from the idle periods that may occur within the compute phase due to cache misses, TLB misses, branch mispredictions, etc.

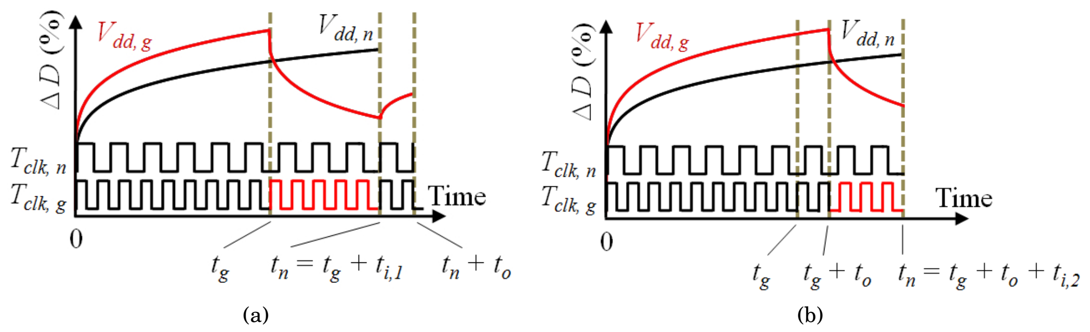


Fig. 3. The illustration of our scheme for generating (a) fixed idle time, $t_{i,1}$, and (b) variable idle time, $t_{i,2}$. The figure is not drawn to scale; in reality, $t_n, t_g, t_{i,1}, t_{i,2} >> t_o$.

(upon a cache miss) remains the same at both $V_{dd,n}$ and $V_{dd,g}$. This constant access time corresponds to a larger number of cycles under the faster clock at $V_{dd,g}$.

The overhead of off-chip operations such as cache misses therefore corresponds to a larger number of cycles of penalty under $V_{dd,g}$, implying that during a fixed number of clock cycles, c_f, while the processor is awake, the number of instructions committed under GNOMO will be smaller. In other words, $I_g = I_n - I_o$, where I_o is the number of instructions that could be committed at the nominal supply voltage, but not at $V_{dd,g}$. We denote the overhead of completing the execution of these I_o instructions by c_o (in clock cycles) and t_o (in seconds).

This overhead can be accommodated in two ways:

— By keeping the duration of idle phase fixed (= $t_{i,1}$) and deferring the execution of I_o instructions to *after* the idle phase, as shown in Fig. 3(a): this incurs a performance penalty of t_o time units; we show that this performance penalty is very small (about 1.5% on an average) in Section 6.4.

— By executing the I_o instructions *within* the idle phase, as shown in Fig. 3(b) (which shows the same operations as in Fig. 3(a), except for the placement of the execution overhead). This reduces idle time from $t_{i,1}$ to $t_{i,2}$:

$$t_{i,2} = t_n - (t_g + t_o) = t_{i,1} - t_o \tag{8}$$

This reduction in idle time also reduces the overall recovery possible, albeit without a performance penalty.

For a specific value of $V_{dd,n}$, the value of $t_{i,1}$ is *fixed* as it depends only on the fixed number of cycles c_f and the frequency corresponding to $V_{dd,n}$, which is also fixed according to Table II. On the other hand, the value of $t_{i,2}$ *varies* with the number of off-chip accesses during execution.

A second practical consideration is the need to preserve the state of the processor during the sleep periods, thus facilitating a rapid return to normal execution upon wakeup. To ensure this, the circadian cycle is applied to computational units such as ALUs, decoders, and sense amplifiers. Storage elements such as caches, register files, the reorder buffer (ROB), tables for virtual address translation (TLBs) and for branch prediction (BPTs), and load-store queue (LSQs), may contain state data that must be preserved, and are maintained with suitable optimizations, as reported in Section 4.2.3.

3.3. Idle Time Generation – Implementation

In our experiments, we use the first of the two schemes proposed above: after completing c_f cycles, we use a fixed idle time of $t_{i,1}$ time units (corresponding to $c_{i,1}$ cycles), as given by Equation (7). Under this scheme, the idle phase duration is fixed, and a fixed and predictable amount of recovery is guaranteed. A substantially similar approach may be used for the second scheme.

Since the idle time is fixed, the completion time of $I_g + I_o$ instructions is delayed by c_o cycles, i.e., there is a performance penalty involved, but the amount of recovery time is guaranteed. The total performance penalty for a particular workload is given by:

$$\text{Performance Penalty} = \frac{t_o}{t_n} = \frac{c_o \cdot T_{clk,g}}{c_f \cdot T_{clk,n}} \tag{9}$$

In our experiments, the choice of $V_{dd,n}$ and $V_{dd,g} > V_{dd,n}$ may take one of several values; each such value corresponds to a different frequency of operation for the processor. We use a set of realistic discrete supply voltage/frequency (V_{dd}/f) pairs, adopted from Intel's recent 48-core IA-32 Processor [Howard et al. 2011] as shown in Table II, where V_{dd} lies in the range [0.7V, 1.3V] (this choice is only for illustration purposes; any other V_{dd}/f framework can be used instead)[5]. This allows us to operate within the framework of existing technologies to illustrate the principles of GNOMO. Table II confirms our earlier observation in Section 3.1 that with a linear increase in the value of V_{dd}, the increase in the clock frequency (i.e., circuit speed) is sublinear.

Table II. Operational V_{dd}/f pairs adopted from Intel's IA-32 Processor [Howard et al. 2011]

V_{dd} (Volts)	0.7	0.8	0.9	1.0	1.1	1.2	1.3
Frequency (GHz)	0.25	0.47	0.68	0.86	1.03	1.17	1.30
T_{clk} (ns)	4.00	2.13	1.47	1.16	0.97	0.85	0.77

Using the data in Table II, the fraction of the idle time at the GNOMO supply voltage in the "ideal" case (where t_o = 0), to the execution time for c_f cycles at $V_{dd,n}$, can be computed for valid combinations of ($V_{dd,n}$, $V_{dd,g}$) as follows:

$$\frac{t_{i,1}}{t_n} = \frac{c_f \cdot (T_{clk,n} - T_{clk,g})}{c_f \cdot T_{clk,n}} = 1 - \frac{T_{clk,g}}{T_{clk,n}} \tag{10}$$

Note that this expression is an approximation of the realistic idle time fraction, $t_{i,2}$ / t_n, since our experiments show that the performance penalty is small.

Table III. Percentage idle time $t_{i,1}$ for various $(V_{dd,n}, V_{dd,g})$

$V_{dd,g} \rightarrow$ / $V_{dd,n} \downarrow$	0.8V	0.9V	1.0V	1.1V	1.2V	1.3V
0.7V	46.8%	63.2%	70.9%	75.7%	78.6%	80.8%
0.8V	–	30.8%	45.3%	54.3%	59.8%	63.9%
0.9V	–	–	20.9%	34.0%	41.9%	47.7%
1.0V	–	–	–	16.1%	26.5%	33.6%
1.1V	–	–	–	–	12.5%	20.8%
1.2V	–	–	–	–	–	10.0%

[5]It is important to emphasize here that although this table is taken from the allowable DVFS values for an Intel processor, GNOMO is *not* an adaptive supply voltage scheme (ASV) for BTI mitigation. Under the baseline GNOMO scheme, the processor operates at a constant voltage and frequency. However, it is possible to extend the baseline GNOMO framework to the case where ASV is required for power management.

Table III shows this percentage, wherein each entry is computed by Equation (10), using the respective values of ($T_{clk,n}$, $T_{clk,g}$) from Table II. We observe the following diminishing returns in idle times:

— For a particular value of the nominal supply voltage $V_{dd,n}$ (say 0.8V), a linear increase in the value of $V_{dd,g}$ (along the row from 0.9V to 1.3V) increases the idle time durations only in a sublinear fashion.
— At higher values of $V_{dd,n}$ (≥ 1.1V), the available idle time is low.

The idle times discussed above correspond to available time for BTI recovery, and therefore the delay degradation improvements also show diminishing returns, as was illustrated earlier in Fig. 1(c).

4. ARCHITECTURAL IMPLEMENTATION OF GNOMO

In this section, we discuss the details of GNOMO implementation on a processor with out-of-order execution, and the simulation framework that we utilize to validate the gains of this approach.

4.1. Processor Details

The structure for a general purpose processor is depicted by Fig. 4. Broadly, the components of the processor can be categorized into on-chip components and off-chip components.

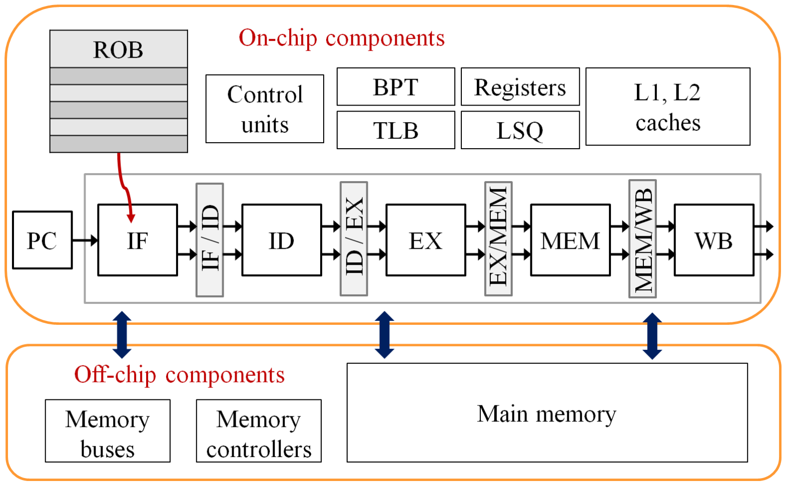


Fig. 4. Schematic of an out-of-order processor, with its on-chip and off-chip components.

The on-chip components include all the circuitry and resources that require fast communication during the execution of instructions, such as the ROB, registers, TLBs, BPTs, and LSQs. The processor architecture is assumed to be a MIPS-like five-stage pipeline shown in Fig. 4, with the fetch, decode, execution, memory and commit stages. On-chip caches store the copy of a part of the data from the main memory and are further divided into L1 and L2 caches. Communication between the various components is achieved through on-chip buses.

The off-chip components include the peripheral memory (main memory), the associated memory controllers and the memory buses. An access to the main memory, which occurs when there is a cache miss, is carried out by the buses connecting the on-chip and the off-chip components.

4.2. Simulation Framework

4.2.1. Processor Model. We implement our GNOMO framework in a MIPS architecture based out-of-order execution processor simulation environment built upon Simplescalar [SimplesScalar LLC 2003], with the added functionality of being able to model the power dissipation of the various components of the processor using Wattch [Brooks et al. 2000]. We adopt SPEC 2000 suite [SPEC CPU utility programs 2000] as the benchmarks for simulations, which are executed using Wattch, with the inputs for these benchmarks derived from the MinneSPEC set [KleinOsowski and Lilja 2002]. The technology parameters for our simulations are at the 32nm node: since the available implementation of Wattch is based on older technology nodes (100nm and above), the parameters used in the power model were updated for the 32nm node from Orion2.0 [Kahng et al. 2009].

During execution, a small, fixed number of instructions are fetched from the instruction cache into the ROB. The processor then repeatedly extracts an instruction from the tail of the ROB and starts execution in the pipeline beginning with fetch and decode. Functional and control units are used during the pipeline operation, and the underlying devices undergo stress-relax cycles during execution.

4.2.2. Idle Time Insertion. We now describe the modifications to incorporate GNOMO in the simulation environment. The first step is to introduce periods of idle times in the execution. To enable this, we maintain a clock counter that is initialized whenever a compute phase begins. After c_f cycles have been completed, the pipeline is flushed, and the sleep signal is activated for all on-chip components. Note that since the pipeline is flushed, the execution in the next compute phase restarts at the instruction that follows the last committed instruction.

4.2.3. Power Gating with State Preservation. When the processor enters into the idle state, power-gating is applied to some on-chip components. As discussed in Section 3.1, power-gating disconnects the devices in these components from the power supply, resulting in a possible loss of state. For computational elements such as ALUs and control units that do not contain useful state information, this is entirely acceptable. We augment Wattch to model the functionalities and power dissipation of the processor when its operation is halted upon sleep, and resumed upon wakeup, and describe this augmentation scheme below.

For caches, register files and other storage structures, we use the sleep signal to preserve state through hybrid drowsy cache techniques [Kim et al. 2004; Meng et al. 2005] that reduce standby leakage. Under this scheme, the sleep signal, instead of cutting off the supply voltage, triggers circuitry that scales the supply voltage of the devices to an appreciably low value (for instance, 0.3V) where the device has a very small leakage and the state can be preserved. The sleep/wakeup overhead associated with these schemes is 1-10 cycles and is thus negligible as compared to the duration of the idle period (which is of the order of a million cycles).

In this scheme, we first save the register files and the storage structures in the cache (their sizes are typically small: 32 to 64 bytes for different structures, compared to the 16KB size of the cache; hence, this takes a negligible amount of storage), and then allow the sleep signal to activate the hybrid drowsy cache mode. When the next compute phase begins, the data for register files and storage structures is restored. This scheme is depicted in Fig. 5, which shows how the same sleep signal acts in a different manner for various on-chip components.

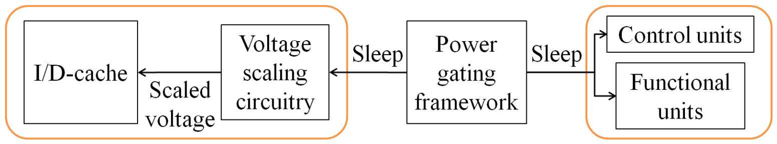


Fig. 5. The power-gating scheme applied differently for various on-chip components. Units with combinational circuits are completely switched off by the sleep signal. Cache on the other hand preserves state by the use of a special circuitry that scales the cache supply voltage to a relatively low value.

Under this scheme, a main memory access transaction (either for a load or for a store) may be in progress just before the processor enters into the idle phase. Since the pipeline is flushed, this instruction would have required a re-execution of the memory transaction. This is avoided by our scheme of saving the LSQ entries during the idle phase. Upon wakeup, the processor checks the old LSQ entries and finds the load/store operation already served during the idle phase (since the memory transaction continues to take place in the off-chip components even during the idle time and the off-chip components are not affected by on-chip GNOMO).

5. POWER ANALYSIS

In this section, we examine the implications of GNOMO on power at both the circuit and architectural levels. In sequence, we examine a set of factors that cause the power dissipation to be altered under GNOMO. First, in Section 5.1, we analyze the change in power consumption, averaged over its lifetime, of a circuit operating at an elevated V_{dd}, as compared to the nominal operation, for a unit without state that is completely power-gated during the idle phase. Next, in Section 5.2, we examine the impact of reduced aging on the delay guardbands, and hence the power dissipation, associated with such a unit. Finally, in Section 5.3, we consider the complete picture: change in the total power dissipation due to GNOMO, which includes the impact of units that are turned off, as well as those placed in a drowsy state, during the idle phase.

5.1. Changes in Power as a Function of $V_{dd,g}$

We begin by considering a circuit that is designed to meet the delay specification at the beginning of lifetime, and is not resilient to BTI[6], and set the delay of this nominal circuit to a normalized power value of 1 unit. When such a circuit is operated at GNOMO, the power dissipation changes as follows:

— As discussed in Section 2.2, with the supply voltage increased to $V_{dd,g}$, dynamic power increases quadratically and leakage increases exponentially [Jejurikar et al. 2004; Bhunia and Mukhopadhyay 2010].
— Even though the V_{th} increase due to BTI degradation is small, the exponential relationship in leakage power leads to a significant reduction of leakage over the lifetime of the chip [Kumar et al. 2011]. This effect is more pronounced at higher V_{dd} values (due to increased BTI degradation).
— A further reduction in the average dynamic and leakage power consumption occurs due to generation of idle time, since power dissipation now occurs only in the compute phase and not in the idle phase (except during sleep/wakeup cycles), which is a fraction of the total nominal computation time.

For our running example of the MCNC benchmark alu4, Fig. 6 illustrates the changes of power (dynamic+leakage), normalized to the nominal value defined above,

[6]Such a circuit is not practically useful, as it fails with time. We merely consider this as a baseline for all comparisons, for convenience.

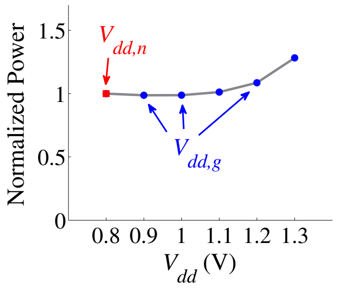


Fig. 6. Change in average power (dynamic + leakage) for alu4 as a function of $V_{dd,g}$; $V_{dd,n}$ = 0.8V.

for a typical case with $V_{dd,n}$ = 0.8V, with $V_{dd,g}$ ranging from 0.9V to 1.3V. Similar trends are seen for other values of $V_{dd,n}$. Considering the combined impact of the three effects listed above, Fig. 6 shows that the power consumption therefore remains about flat until $V_{dd,g}$ = 1.0V and then begins to increase beyond this point.

5.2. Power Savings in Delay Guardbanding

We now consider the scenario where the delay guardbanding is introduced in the circuit to make it BTI-resilient throughout its lifetime. In our discussion, we use the terms "guardbanding" and "compensation" interchangeably. We work with a guardbanding technique under which the transistors in the circuits are synthesized with a tighter timing constraint such that the end-of-lifetime delay meets the delay specification, D_{spec}. This delay margin algorithm is identical to that used in Section IV of [Kumar et al. 2011]. The cost of this compensation is in the form of an increased area overhead.

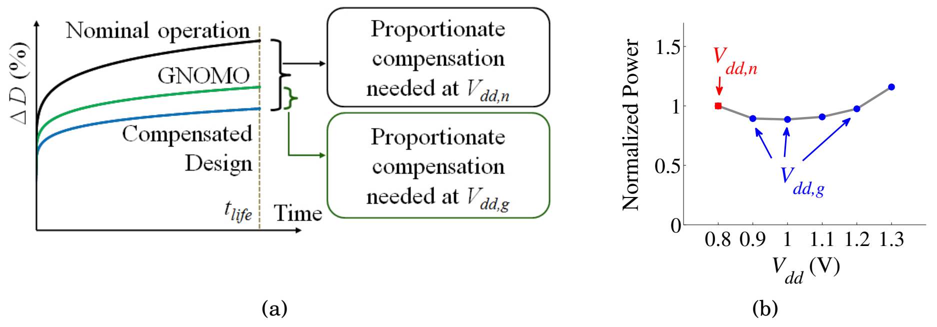


Fig. 7. (a) The temporal delay degradation of alu4. The area overhead required to compensate the circuit under GNOMO, is less than that required under nominal operation. (b) Trends in power for alu4, with power overhead due to compensation incorporated, as a function of $V_{dd,g}$; $V_{dd,n}$ = 0.8V.

The idea presented in this section can be depicted schematically through Fig. 7(a), which shows the temporal delay of a circuit along the y-axis and time along the x-axis. A circuit operated at $V_{dd,g}$ has a lower end-of-lifetime delay degradation than the same circuit when operated at $V_{dd,n}$. Hence, in making each case BTI-resilient, a lower amount of compensation (and associated area overhead) is needed with GNOMO than for the $V_{dd,n}$ case. Since both the dynamic and subthreshold leakage power are proportional to the widths of the transistors used in these circuits, a lower compensation area overhead corresponds to a lower power dissipation of the circuit. This results in a further lowering of the power dissipation at the GNOMO supply voltage points, as compared to the points in Fig. 6.

Fig. 7(b) shows the changes in the normalized average total power consumption for circuit alu4, with $V_{dd,n}$ = 0.8V and $V_{dd,g}$ ranging from 0.9V to 1.3V, under the same workload conditions as Fig. 6. This plot combines the effects on power from Section 5.1 and the effect of BTI compensation. The net result is a further decrease in the GNOMO power compared to the power under nominal operation: total power remains below the nominal dissipation as $V_{dd,g}$ is increased, up to 1.2V.

5.3. Overall Power Dissipation

The analysis so far only considers components that can be fully switched off in the idle state and do not need to retain their states. When we consider the power overhead of state preservation, the total power can be higher than shown in Fig. 7(b). The power dissipation in all the components of the processor is evaluated next.

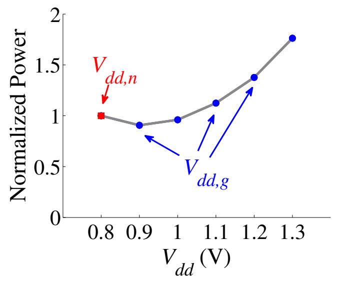


Fig. 8. Change in the total power with GNOMO, showing the power savings at lower values of $V_{dd,g}$.

The changes of the overall power dissipation (again, normalized to the nominal case) is shown in Fig. 8 for $V_{dd,n}$ = 0.8V and $V_{dd,g}$ ranging from 0.9V to 1.3V, for the execution of the applu workload (similar trends are seen for other workloads). We observe that as $V_{dd,g}$ is increased, the power dissipation with GNOMO reduces until about 1.0V, and then becomes greater. Note that the rate of power increase is higher in Fig. 8 as compared to Fig. 7(b).

The power dissipation of cache and storage structures forms a significant portion of the total power dissipation. At higher $V_{dd,g}$ values (beyond $V_{dd,g}$ = 1.0V), the power dissipation is also higher. Thus the total power begins to dominate the power savings that we achieve through GNOMO.

5.4. Choosing the Optimal GNOMO Supply Voltage

Based on the above analysis of overall power dissipation, we can select an optimal choice of $V_{dd,g}$ (= $V_{dd,g}^{opt}$) for a given $V_{dd,n}$, that maximizes the power savings. For instance, we observe in Fig. 8 that for $V_{dd,n}$ = 0.8V, this happens at $V_{dd,g}$ = 0.9V.

We therefore consider the power savings for the benchmark applu, for all $V_{dd,n}$ values, in Table IV. This table is quite similar to Table III, except that it lists the power savings instead of idle times, for valid combinations of $(V_{dd,n}, V_{dd,g})$. Positive entries imply a power savings, whereas negative entries imply a power overhead. From this table, we determine the optimal $V_{dd,g}$ value (for a particular $V_{dd,n}$), to be the value that maximizes power savings, and indicate it by the symbol "*." We observe that for a fixed $V_{dd,n}$, increase in $V_{dd,g}$ (moving along a row) first yields power savings, and then power overhead. Further, as $V_{dd,n}$ is increased, the optimal savings in power (at $V_{dd,g}^{opt}$) although substantial, decrease sublinearly.

Table IV. Percentage overall power savings for various $(V_{dd,n}, V_{dd,g})$ for benchmark applu

$V_{dd,g} \rightarrow$ / $V_{dd,n} \downarrow$	0.8V	0.9V	1.0V	1.1V	1.2V	1.3V
0.7V	14.9%*	13.6%	0.7%	-23.4%	-54.5%	-93.4%
0.8V	–	9.4%*	3.9%	-12.4%	-37.7%	-76.2%
0.9V	–	–	6.9%*	1.8%	-20.2%	-55.7%
1.0V	–	–	–	5.3%*	-12.0%	-35.9%
1.1V	–	–	–	–	4.2%*	-21.4%
1.2V	–	–	–	–	–	-15.5%

Although the simulations above show results only for applu workload, we have observed that these values of optimal $V_{dd,g}$, in fact, are the optimal values across a range of SPEC 2000 benchmarks considered in Section 6.3. The optimal ($V_{dd,n}$, $V_{dd,g}^{opt}$) pairs for each value of $V_{dd,n}$, are tabulated in Table V. We make the following observations about the data in this table:

— For $V_{dd,n}$ = 1.2V, $V_{dd,g}^{opt}$ = $V_{dd,n}$ and no gain is possible. This is attributed to the fact that for various workloads, the total power increases by 14.4% to 16.2% for $V_{dd,g}$ = 1.3V candidate value, resulting is no power savings. This is primarily attributed to a steep increase in the leakage power due to the higher voltage value and also due to the low idle time.
— The GNOMO scheme works best when the value of $V_{dd,n}$ is lower, but it provides significant improvements for all $V_{dd,n} \leq 1.1$V.

Table V. The optimal $V_{dd,g}$ values for various values of $V_{dd,n}$

$V_{dd,n}$ (V)	0.7	0.8	0.9	1.0	1.1	1.2
$V_{dd,g}^{opt}$ (V)	0.8	0.9	1.0	1.1	1.2	1.2

6. RESULTS

The goal of Section 5 was to determine a set of optimal $(V_{dd,n}, V_{dd,g})$ pairs, as illustrated in Table V. Having determined these values, we now quantify, in Section 6.1, the precise reduction in the delay degradation for a set of circuits that could be subcircuits of a processor running under the GNOMO scheme. We use a representative set of benchmark circuits for this purpose; we consider the GNOMO scheme from Table V and the corresponding idle times, as derived from Table II.

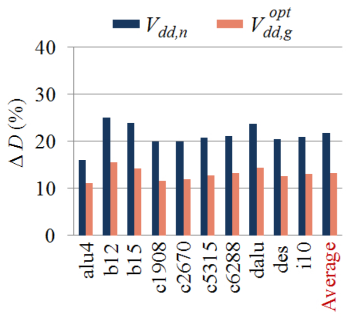
Given this reduction in the delay degradation, in Section 6.2, we determine the circuit-level power reductions that arise as a result of the reduced delay guardbands. The overall power savings, incorporating all factors – increased supply voltage, sleep times, reduced circuit guardbands, and the overhead of state preservation using drowsy caches, are then computed in Section 6.3 for a range of SPEC2000 benchmarks, and the corresponding performance overhead is presented in Section 6.4.

At the circuit level, we examine the application of GNOMO on various ISCAS85, MCNC and ITC99 benchmarks, synthesized using ABC [Berkeley Logic Synthesis and Verification Group 2007] on the 32nm PTM [ASU Nanoscale Integration and Modeling Group 2008] based library. Our library consists of INVs; BUFs; 2-4 input NANDs and NORs; 2 input XORs and XNORs; all with different sizes. We choose $t_{life} = 10$ years.

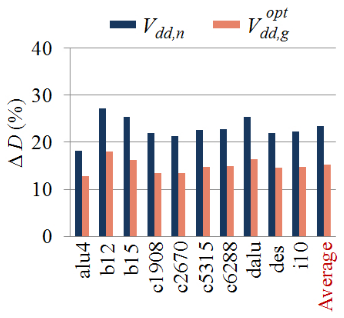
We optimize the circuits by introducing delay margins to compensate for BTI aging, using the algorithms in [Kumar et al. 2011]. At the architectural level, we use the framework detailed in Section 4.

6.1. Delay Degradation Reduction

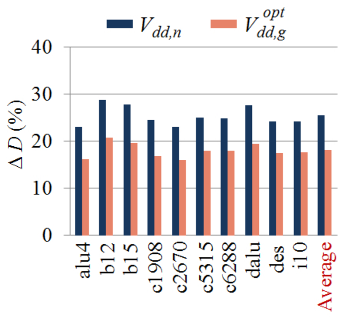
We evaluate the extent to which GNOMO can reduce the degradation in various benchmark circuits at the transistor level. In Fig. 9, for a variety of circuits, we present the end-of-lifetime percentage delay degradation, ΔD (%), for three different ($V_{dd,n}$, $V_{dd,g}^{opt}$) pairs from Table V: (0.8V, 0.9V), (0.9V, 1.0V) and (1.0V, 1.1V).



(a) $V_{dd,n}$ = 0.8V, $V_{dd,g}^{opt}$ = 0.9V



(b) $V_{dd,n}$ = 0.9V, $V_{dd,g}^{opt}$ = 1.0V



(c) $V_{dd,n}$ = 1.0V, $V_{dd,g}^{opt}$ = 1.1V

Fig. 9. The reduction in delay degradation with GNOMO, shown for various circuits for three different ($V_{dd,n}$, $V_{dd,g}^{opt}$) pairs.

In all cases, we see that under GNOMO, the end-of-lifetime delay degradation is significantly smaller than under the nominal scheme. In general, we observe over all ($V_{dd,n}$, $V_{dd,g}^{opt}$) pairs that value of average ΔD (%) over all benchmarks reduces by about 1.3× to 1.8× for GNOMO as compared to nominal operation. This impacts the reduction in area and power overhead significantly, which we discuss next. Further, as expected from our prior discussion in Section 3.3, on diminishing returns in idle times with increasing $V_{dd,n}$, our lifetime gains are higher for lower values of $V_{dd,n}$.

6.2. Area and Power Savings in BTI Compensation

In Section 5.2, we had analyzed that a reduction in delay degradation also results in lower power, due to a lower compensation area overhead. In this subsection, we show this result over a set of benchmark circuits.

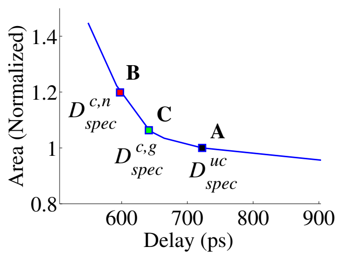


Fig. 10. The normalized-area vs. delay curve for alu4, with area normalized by the area of the uncompensated circuit.

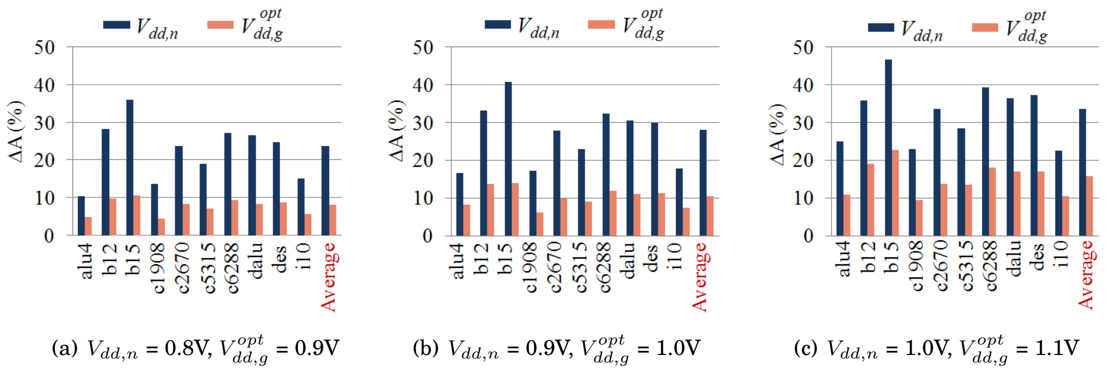


Fig. 11. The reduction in BTI compensation area overhead with GNOMO, shown for various circuits for three different ($V_{dd,n}$, $V^{opt}_{dd,g}$) pairs.

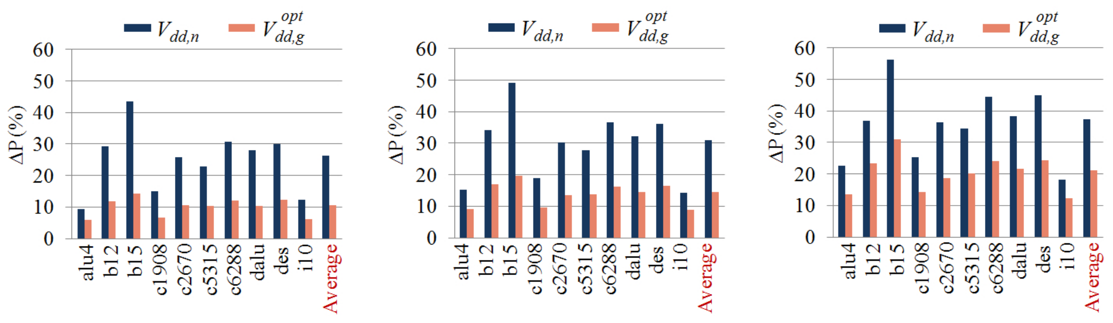
We begin with a specific example; consider the application of GNOMO to the circuit alu4 with ($V_{dd,n}$, $V_{dd,g}$) = (0.9V, 1.1V). The area vs. delay curve for this circuit, for various target delay specifications, is shown in Fig. 10. The area values are normalized to point A, which corresponds to the uncompensated circuit for which no delay margins are added. We compare optimizations using the nominal and the GNOMO supply voltages:

— At $V_{dd,n}$, the ALU incurs a 20.9% delay degradation over its lifetime, which is compensated by mapping the circuit with a tighter specification, $D^{c,n}_{spec}$, using the delay margin algorithm in [Kumar et al. 2011]. This corresponds to point B on the curve, which incurs an additional area overhead of 19.9% over point A. We call this circuit at point B as the "$V_{dd,n}$ circuit": a delay-margined circuit at a supply voltage of $V_{dd,n}$, and this circuit is guaranteed to be functional throughout the projected chip lifetime.
— At the GNOMO voltage, $V_{dd,g}$, the BTI degradation is reduced to 12.6%, and hence the delay margin is relaxed, corresponding to a delay specification of $D^{c,g}_{spec}$ at Point C. This reduces the area overhead to 6.3%. Thus, the area overhead for BTI compensation is reduced by 3$\times$ for GNOMO as compared to the $V_{dd,n}$ case. We call the circuit at point C as the "$V_{dd,g}$ circuit": this is a delay-margined circuit at a supply voltage of $V_{dd,g}$ under a GNOMO-based circadian rhythm, and is guaranteed-functional throughout the projected chip lifetime.

This analysis is applied to all benchmark circuits and the results that show the reduction in compensation area overhead, ΔA, are presented in Fig. 11, for three different ($V_{dd,n}$, $V^{opt}_{dd,g}$) pairs, as in Fig. 9. For each of these circuits, we show the area overhead for the $V_{dd,n}$ and $V_{dd,g}$ circuits.

Comparing the area overhead in both the $V_{dd,n}$ and $V_{dd,g}$ circuits, the overhead for the $V_{dd,g}$ circuits is consistently smaller for each benchmark. These reductions in area overhead impact the power overhead, ΔP, of the $V_{dd,n}$ and the $V_{dd,g}$ circuits, as shown in Fig. 12. Recall that the power overhead in the $V_{dd,g}$ circuit comes from two sources: operation at the GNOMO supply voltage, and from compensation[7], while the $V_{dd,n}$ circuit has power overhead only due to compensation. Again, the gains in area/power overhead are highest for lower values of $V_{dd,n}$. Over all ($V_{dd,n}$, $V^{opt}_{dd,g}$) pairs, we observe

[7]For the time being, since we consider units that will be power-gated completely during the idle phase, we do not consider total power; we will add this consideration in the next subsection.



(a) $V_{dd,n}$ = 0.8V, $V_{dd,g}^{opt}$ = 0.9V (b) $V_{dd,n}$ = 0.9V, $V_{dd,g}^{opt}$ = 1.0V (c) $V_{dd,n}$ = 1.0V, $V_{dd,g}^{opt}$ = 1.1V

Fig. 12. Reduction in BTI compensation area overhead also lowers the power overhead with GNOMO, shown for various circuits for three different ($V_{dd,n}$, $V_{dd,g}^{opt}$) pairs.

that the value of average ΔA over various benchmark circuits reduces by about 1.8× to 3.2× for GNOMO as compared to nominal operation. For average ΔP values, this reduction is about 1.5× to 3.1×.

6.3. Overall Power Savings

The total power evaluation framework and the potential for power savings was discussed in Section 5.3. In this section, we use various types of workloads to gauge the average power savings under various workloads. To determine the overall power savings of the GNOMO scheme, the SPEC 2000 benchmark suite was simulated using SimpleScalar and Wattch, with the processor configuration described in Table VI, under the MinneSPEC input set. The processor was first run at a nominal supply voltage, $V_{dd,n}$, and then under GNOMO at the optimal value of $V_{dd,g}$, with state-sensitive elements being placed in drowsy mode, as described earlier.

Table VI. Configuration of the processor

Fetch/Decode/ Issue/Commit width	4/4/4/4 (instructions/cycle)
RUU size	64 entries
LSQ size	32 entries
Private L1 Data cache	16KB, 4-way set associative, 32B block size
Private L1 Instruction cache	16KB, 4-way set associative, 32B block size
Private L2 Unified Data and Instruction cache	512KB, 8-way set associative, 64B block size
Memory access bus width	8 bytes
Data Translation Lookaside Buffer	512KB, 4-way set associative, 4KB block size
Instruction Translation Lookaside Buffer	256KB, 4-way set associative, 4KB block size
Number of integer ALUs	4
Number of integer multiplier/dividers	4
Number of floating point ALUs	2
Number of floating point multipliers/dividers	2
Number of memory system ports available to CPU	2 (1 read, 1 write)

We discussed in Section 5.4, that the operation at optimal GNOMO supply voltage, $V_{dd,g}^{opt}$ gives us the highest power savings for the benchmark applu, under the processor

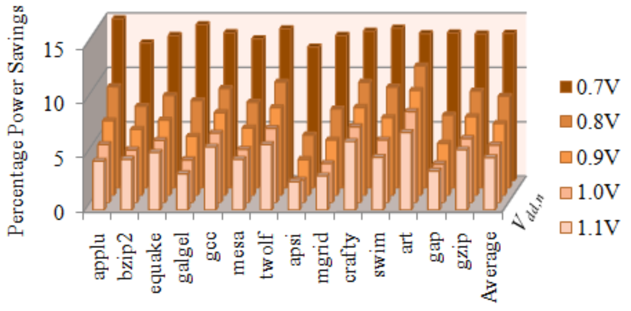


Fig. 13. The power savings corresponding to the ($V_{dd,n}$, $V_{dd,g}^{opt}$) point for various SPEC 2000 workloads.

configuration described in Table VI. A similar trend is observed with other workloads in the SPEC 2000 suite. This data is presented in Fig. 13, which shows the power savings at the ($V_{dd,n}$, $V_{dd,g}^{opt}$) point, for $V_{dd,n} \in$ [0.7V, 1.1V]. On average, over all benchmarks, GNOMO achieves power savings of up to 13.6%. This shows that reducing guardbanding overhead can appreciably improve the overall power consumption. Further, the power savings decrease sublinearly as $V_{dd,n}$ increases, as the idle time durations become smaller at higher $V_{dd,n}$ points. Thus, the average power savings are the highest at ($V_{dd,n}$, $V_{dd,g}^{opt}$) = (0.7V, 0.8V), and the lowest at ($V_{dd,n}$, $V_{dd,g}^{opt}$) = (1.1V, 1.2V).

6.4. Analyzing the Architectural Performance Penalty

As described in Section 3.3, the idle time scheme used here incurs a performance penalty due to the increased mismatch between on-chip and off-chip speeds under GNOMO. For the benchmarks and processor configuration considered in Section 6.3, we quantify this penalty. After c_f cycles of instructions, we record the values of t_o required by the corresponding set of I_o instructions. Fig. 14 shows the average performance penalty (over all c_f sets), based on Equation (9), for the execution of various benchmarks, with values of $V_{dd,n}$ = 0.7V to 1.1V and the corresponding $V_{dd,g}^{opt}$ value.

We find that choosing c_f = 10 million ensures that at $V_{dd,n}$ = 0.7V (which shows the largest penalty), the performance penalty for GNOMO is an average of 1.9% over all benchmarks. The worst-case penalty is under 3% for most of the workloads, and is 5.9% and 13.5% for the remaining two. Further, our simulations show that for the workloads with the largest overhead, such cases are rare: over 90% of the c_f sets for these workloads have $< 1.5\%$ overhead. The remaining c_f sets are characterized by a higher number of memory accesses, thus incurring a higher performance penalty.

This choice of a large value of c_f has other benefits. The repeated compute-standby operation in our scheme may seem to create regular interruptions in workload execution. Since c_f = 10 million, these occur much less frequently (and also more predictably) as compared to the unpredictable interruptions and pipeline flushes caused by cache read/write misses, branch mispredictions, etc. Further, as discussed in Section 3.1, the power-gating overhead of 10 to 50 cycles, becomes completely negligible for this choice of c_f. At the frequencies under consideration, this choice of c_f also keeps the temperature unchanged since the power dissipation is similar (or slightly lower), and the compute/idle phases change at a rate that is below the thermal time constant of the material.

Further, we note that as c_f is increased from 100,000 to 10 million cycles, the maximum performance penalty (over all c_f sets) over the execution of a benchmark de-

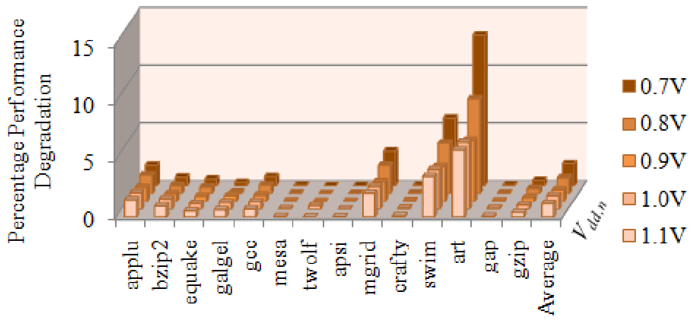


Fig. 14. The performance penalties for various SPEC CPU 2000 workloads.

creases. This can be explained by the fact that a larger value of c_f corresponds to a larger number of on-chip operations, offering a greater potential for hiding latencies for off-chip operations through out-of-order execution. The average penalty, however, remains approximately the same.

With an increase in $V_{dd,n}$, the penalty decreases sublinearly. This is because the increase in off-chip latency in cycles is directly related to the difference between the clock periods at $V_{dd,g}$ and $V_{dd,n}$. As shown in Table II, this difference decreases sublinearly as $V_{dd,n}$ goes up, implying a lower additional overhead from off-chip accesses.

As a last note, it is possible to recover performance to the original value by operating the GNOMO design at a slightly higher $V_{dd,g}$. It is observed, however, that the target average percentage performance penalty of $< 2\%$ (incurred by the GNOMO scheme) corresponds to an increase in $V_{dd,g}$ of only about 5 to 10mV. For this level of change, the aging overheads are essentially unchanged from those for the original GNOMO voltage. However, this approach is impractical for several reasons: first, changing the voltage in steps of 5 or 10mV is quite expensive, as discussed earlier in Section 1, and second, the precise performance penalty is benchmark-dependent and it is not easy to predict the required offset to the GNOMO voltage.

7. CONCLUSION

This paper introduces the idea of using the concept of circadian rhythms to operate a processor in alternating wakeful and sleepy states. The wakeful state uses an elevated supply voltage under the GNOMO scheme, and the resulting reliability degradation is better than the processor that "pulls an all-nighter" without going to sleep. We demonstrate at the architectural and circuit levels that this scheme is viable, and that it provides significant gains in power at about the same performance. The current implementation focuses on a constant nominal V_{dd}; however, in principle, the idea can be extended when the nominal case uses dynamic voltage and frequency scaling.

Acknowledgment

This work was supported in part by the NSF under award CCF-1017778 and the SRC under contract 2012-TJ-2234.

REFERENCES

Abella, J., Vera, X., and Gonzalez, A. 2007. Penelope: The NBTI-aware processor. In *Proceedings of the International Symposium on Microarchitecture*. IEEE Computer Society, Washington, DC, USA, 85–96.
ASU Nanoscale Integration and Modeling Group. 2008. http://www.ptm.asu.edu.

BASOGLU, M., ORSHANSKY, M., AND EREZ, M. 2010. NBTI-aware DVFS: a new approach to saving energy and increasing processor lifetime. In *Proceedings of the International Symposium on Low Power Electronics and Design*. ACM, New York, NY, USA, 253–258.

BERKELEY LOGIC SYNTHESIS AND VERIFICATION GROUP. 2007. ABC: a system for sequential synthesis and verification, release 70930.

BHARDWAJ, S., WENPING, W., VATTIKONDA, R., CAO, Y., AND VRUDHULA, S. 2006. Predictive modeling of the NBTI effect for reliable design. In *Proceedings of the Custom Integrated Circuits Conference*. IEEE Computer Society Press, Los Alamitos, CA, USA, 189 –192.

BHUNIA, S. AND MUKHOPADHYAY, S. 2010. *Low-power variation-tolerant design in nanometer silicon*. Springer, New York.

BILD, D. R., BOK, G. E., AND DICK, R. P. 2009. Minimization of NBTI performance degradation using internal node control. In *Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 3001 Leuven, Belgium, 148–153.

BROOKS, D., TIWARI, V., AND MARTONOSI, M. 2000. Wattch: a framework for architectural-level power analysis and optimizations. In *Proceedings of the International Symposium on Computer Architecture*. ACM, New York, NY, USA, 83–94.

CALIMERA, A., MACII, E., AND PONCINO, M. 2010. NBTI-aware clustered power gating. *ACM Transactions on Design and Automation of Electronic Systems 16,* 1, 1–25.

CHEN, X., WANG, Y., CAO, Y., MA, Y., AND YANG, H. 2009. Variation-aware supply voltage assignment for minimizing circuit degradation and leakage. In *Proceedings of the International Symposium on Low Power Electronics and Design*. ACM, New York, NY, USA, 39–44.

DEBOLE, M., KRISHNAN, R., BALAKRISHNAN, V., WANG, W., LUO, H., WANG, Y., XIE, Y., CAO, Y., AND VIJAYKRISHNAN, N. 2009. New-age: a negative bias temperature instability-estimation framework for microarchitectural components. *International Journal of Parallel Programming 37,* 4, 417–431.

GUPTA, S. AND SAPATNEKAR, S. S. 2012. GNOMO: Greater-than-NOMinal V_{dd} operation for BTI mitigation. In *Proceedings of the Asia and South Pacific Design Automation Conference*. IEEE Press, Piscataway, NJ, USA, 271–276.

HOWARD, J., DIGHE, S., VANGAL, S., RUHL, G., BORKAR, N., JAIN, S., ERRAGUNTLA, V., KONOW, M., RIEPEN, M., GRIES, M., DROEGE, G., LUND-LARSEN, T., STEIBL, S., BORKAR, S., DE, V., AND VAN DER WIJNGAART, R. 2011. A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling. *IEEE Journal of Solid-State Circuits 46,* 1, 173–183.

JEJURIKAR, R., PEREIRA, C., AND GUPTA, R. 2004. Leakage aware dynamic voltage scaling for real-time embedded systems. In *Proceedings of the Design Automation Conference*. ACM, New York, NY, USA, 275–280.

KAHNG, A. B., BIN, L., PEH, L.-S., AND SAMADI, K. 2009. Orion 2.0: A fast and accurate NoC power and area model for early-stage design space exploration. In *Proceedings of the Design, Automation and Test in Europe*. European Design and Automation Association, 3001 Leuven, Belgium, 423–428.

KARPUZCU, U. R., GRESKAMP, B., AND TORRELLAS, J. 2009. The BubbleWrap many-core: popping cores for sequential acceleration. In *Proceedings of the International Symposium on Microarchitecture*. IEEE Computer Society, Washington, DC, USA, 447–458.

KIM, N. S., FLAUTNER, K., BLAAUW, D., AND MUDGE, T. 2004. Circuit and microarchitectural techniques for reducing cache leakage power. *IEEE Transactions on Very Large Scale Integrated Systems 12,* 2, 167–184.

KLEINOSOWSKI, A. J. AND LILJA, D. J. 2002. MinneSPEC: A new SPEC benchmark workload for simulation-based computer architecture research. *Computer Architecture Letters 1,* 1, 7–10.

KUMAR, S. V., KIM, C. H., AND SAPATNEKAR, S. S. 2006. An analytical model for negative bias temperature instability. In *Proceedings of the International Conference on Computer-Aided Design*. ACM, New York, NY, USA, 493–496.

KUMAR, S. V., KIM, C. H., AND SAPATNEKAR, S. S. 2007. NBTI-aware synthesis of digital circuits. In *Proceedings of the Design Automation Conference*. ACM, New York, NY, USA, 370–375.

KUMAR, S. V., KIM, C. H., AND SAPATNEKAR, S. S. 2011. Adaptive techniques for overcoming performance degradation due to aging in CMOS circuits. *IEEE Transactions on Very Large Scale Integration Systems 19,* 4, 603–614.

LI, L., ZHANG, Y., YANG, J., AND ZHAO, J. 2010. Proactive NBTI mitigation for busy functional units in out-of-order microprocessors. In *Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 3001 Leuven, Belgium, 411–416.

MENG, Y., SHERWOOD, T., AND KASTNER, R. 2005. Exploring the limits of leakage power reduction in caches. *ACM Transactions on Architecture and Code Optimization 2,* 3, 221–246.

MINTARNO, E., SKAF, J., ZHENG, R., VELAMALA, J., CAO, Y., BOYD, S., DUTTON, R., AND MITRA, S. 2010. Optimized self-tuning for circuit aging. In *Proceedings of the Design, Automation Test in Europe*. European Design and Automation Association, 3001 Leuven, Belgium, Belgium, 586–591.

MITRA, S., BRELSFORD, K., KIM, Y. M., LEE, H.-H. K., AND LI, Y. 2011. Robust system design to overcome CMOS reliability challenges. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems 1,* 1, 30–41.

SHIN, J., ZYUBAN, V., BOSE, P., AND PINKSTON, T. M. 2008. A proactive wearout recovery approach for exploiting microarchitectural redundancy to extend cache SRAM lifetime. In *Proceedings of the International Symposium on Computer Architecture*. ACM, New York, NY, USA, 353–362.

SIDDIQUA, T. AND GURUMURTHI, S. 2010. A multi-level approach to reduce the impact of NBTI on processor functional units. In *Proceedings of the Great Lakes Symposium on VLSI*. ACM, New York, NY, USA, 67–72.

SIMPLESSCALAR LLC. 2003. http://www.simplescalar.com.

SPEC CPU UTILITY PROGRAMS. 2000. http://www.spec.org/cpu2000/Docs/utility.html.

SRINIVASAN, J., ADVE, S. V., PRADIP, B., AND RIVERS, J. A. 2005. Lifetime reliability: toward an architectural solution. *IEEE Micro 25,* 3, 70–80.

TIWARI, A. AND TORRELLAS, J. 2008. Facelift: hiding and slowing down aging in multicores. In *Proceedings of the International Symposium on Microarchitecture*. IEEE Computer Society, Washington, DC, USA, 129–140.

WANG, W., REDDY, V., KRISHNAN, A., VATTIKONDA, R., KRISHNAN, S., AND CAO, Y. 2007. Compact modeling and simulation of circuit reliability for 65-nm CMOS technology. *IEEE Transactions on Device and Materials Reliability 7,* 4, 509–517.

WANG, Y., LUO, H., HE, K., LUO, R., YANG, H., AND XIE, Y. 2011. Temperature-aware NBTI modeling and the impact of standby leakage reduction techniques on circuit performance degradation. *IEEE Transactions on Dependable and Secure Computing 8,* 5, 756 –769.

ZHAN, Y., KUMAR, S. V., AND SAPATNEKAR, S. S. 2008. Thermally aware design. *Foundations and Trends in Electronic Design Automation 2,* 3, 255–370.

ZHANG, L. AND DICK, R. P. 2009. Scheduled voltage scaling for increasing lifetime in the presence of NBTI. In *Proceedings of the Asia and South Pacific Design Automation Conference*. IEEE Press, Piscataway, NJ, USA, 492–497.

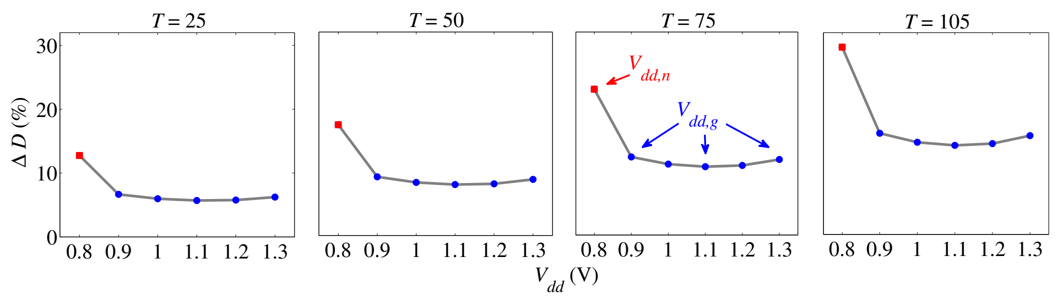
Appendix

Impact of Temperature Variations

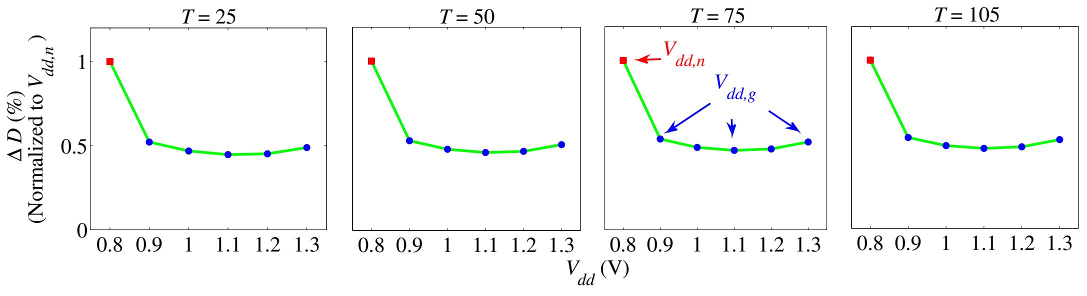
We presented the analysis and results of reduction in percentage delay degradation ($\Delta D(\%)$) with GNOMO in Sections 3 and 6, at a fixed temperature of T = 105°C. Typically, as a processor operates, its temperature may reach below or go above T = 105°C, depending upon the workload at any specific time. For example, the thermal profile of a processor in [Wang et al. 2011] (Fig. 2 of this paper), while executing various tasks, shows that the chip temperature can vary from about 50°C up to 120°C. Since it is practically very difficult to predict the nature of variations in tempertature before execution, experiments and characterizations are conducted at an elevated temperature of T = 105°C, to capture the averaging effect of delay degradation throughout lifetime with fluctuations in temperature, and a safety margin for designers to incorporate guardbands that allow functional correctness throughout lifetime.

In this section, we show that the gains in percentage delay degradation, and the consequent choice of the optimal GNOMO V_{dd} (Section 5.4), obtained by our GNOMO approach are practically invariant with the variations in temperature. Fig. 15(a) shows the percentage delay degradation of MCNC benchmark b12 for $V_{dd} \in$ [0.8V, 1.3V] after 10 years of operation, for four different temperature values, T = 25°C, 50°C, 75°C, and 105°C.

It is observed in general that the lifetime delay degradation is higher at higher temperatures, since BTI mechanism is exacerbated at higher temperatures. For instance, the ΔD values corresponding to V_{dd} = 1.0V, increases in magnitude as we move from T = 25°C to T = 105°C. However, for each of the four temperature points, it is observed



(a)



(b)

Fig. 15. Trends in (a) percentage delay degradation (ΔD (%)), and (b) percentage delay degradation normalized to the value at $V_{dd,n}$, of MCNC benchmark b12, with changes in operating temperature. Trends in (b) show that the gains due to GNOMO are practically invariant accross all temperatures.

that operating at the GNOMO V_{dd} ensures reduction of delay degradation as compared to that at the nominal V_{dd}.

In order to estimate the extent of this reduction at different temperatures, we plot all the ΔD (%) values of Fig. 15(a) as normalized by the ΔD (%) value of the nominal V_{dd}, at the respective temperature point, in Fig. 15(b). It can be seen that compared to the nominal V_{dd} (0.8V in the figure), the normalized reduction in delay degradation obtained by operating at any specific GNOMO V_{dd} (for instance 1.0V), is nearly the same for all temperature points.

This illustrates that the gains due to GNOMO remain nearly invariant with the changes in temperature. Similar observations are made for other values of nominal V_{dd} as well. Since the area overhead and power results also depend upon the extent of reduction in delay degradation obtained by GNOMO, the optimal GNOMO V_{dd} choice also remain the same with varying temperatures.