40th IEEE/ACM International Symposium on Microarchitecture

Mitigating Parameter Variation with Dynamic Fine-Grain Body Biasing*

Radu Teodorescu, **Jun Nakano**, **Abhishek Tiwari** and **Josep Torrellas**
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

Abstract

Parameter variation is detrimental to a processor's frequency and leakage power. One proposed technique to mitigate it is Fine-Grain Body Biasing (FGBB), where different parts of the processor chip are given a voltage bias that changes the speed and leakage properties of their transistors. This technique has been proposed for static application, with the bias voltages being programmed at manufacturing time for worst-case conditions.

In this paper, we introduce Dynamic FGBB (D-FGBB), which allows the continuous re-evaluation of the bias voltages to adapt to dynamic conditions. Our results show that D-FGBB is very versatile and effective. Specifically, with the processor working in normal mode at fixed frequency, D-FGBB reduces the leakage power of the chip by an average of 28–42% compared to static FGBB. Alternatively, with the processor working in a high-performance mode, D-FGBB increases the processor frequency by an average of 7–9% compared to static FGBB — or 7–16% compared to no body biasing. Finally, we also show that D-FGBB can be synergistically combined with Dynamic Voltage and Frequency Scaling (DVFS), creating an effective means to manage power.

1. Introduction

As high-performance processors move beyond 45nm technologies, designers face the major roadblock of parameter variation — the deviation of Process, Voltage, and Temperature (PVT) [2] values from nominal specifications. Variation makes designing processors harder because they have to work under a range of parameter values.

Variation is induced by several fundamental effects. Process variation is caused by the inability to precisely control the fabrication process at small-feature technologies. It is a combination of systematic effects (e.g., lithographic lens aberrations) and random effects (e.g., dopant density fluctuations). Voltage variation can be caused by *IR* drops in the supply distribution network or by *L dI/dt* noise under changing load. Finally, temperature variation is largely due to different levels of activity across the processor. All these variations become harder to tolerate as technology scales to minute feature sizes.

A key process parameter subject to variation is the transistor threshold voltage (V_{th}). Variation in V_{th} directly impacts two major properties of the processor, namely the frequency it attains and the leakage power it dissipates. Moreover, V_{th} is also a function of temperature, which increases its variability [42].

*This work was supported in part by the National Science Foundation under grant CCR-0325603; DARPA under grant NBCH30390004; SRC GRC under grant 2007-HJ-1592; and gifts from IBM and Intel. Radu Teodorescu is supported by an Intel PhD Fellowship. Jun Nakano is now with IBM Japan. His e-mail address is nakanoj@jp.ibm.com.

A recently-proposed technique to mitigate V_{th} variation within a chip is Fine-Grain Body Biasing (FGBB) [45]. FGBB applies different body biases to different sections of the chip, which we call *Cells*. A body bias is a voltage applied between the source or drain of a transistor and its substrate, effectively changing the transistor's V_{th} [42]. Depending on the polarity of the voltage applied, V_{th} increases or decreases. If it increases, the transistor becomes less leaky and slower; if it decreases, the transistor becomes leakier and faster. By reducing the V_{th} in cells with slow transistors and increasing the V_{th} in cells of leaky transistors, we reduce the variation within the die and attain a better frequency-leakage operation for the chip.

Previous work has proposed determining the body bias voltages at manufacturing time and setting them permanently for the lifetime of the chip. This means that the optimal values for the bias voltages have to be selected considering worst-case temperature and, therefore, delay conditions. This results in an overly-conservative configuration. In practice, the processor does not normally run at worst-case temperature and delay conditions. To take advantage of this, we propose to continuously adjust the body biases dynamically, adapting to changes in operating conditions. We call the scheme *Dynamic FGBB (D-FGBB)*.

The main contribution of this paper is to introduce and evaluate D-FGBB. We show that D-FGBB is very versatile and significantly more effective than S-FGBB. Specifically, with the processor working in normal mode at fixed frequency, D-FGBB reduces the leakage power of the chip by an average of 28–42% compared to static FGBB — the higher savings corresponding to the cases with more body bias cells per chip. Alternatively, with the processor working in a high-performance mode, D-FGBB increases the processor frequency by an average of 7–9% compared to static FGBB — or 7–16% compared to no body biasing. We also show that D-FGBB can complement Dynamic Voltage Scaling (DVS) and that it scales well when combined with Dynamic Voltage and Frequency Scaling (DVFS).

A second contribution of this paper is the development of a new, parametrized model of V_{th} variation within the chip. We use it to model batches of processor chips with variation and to estimate the effectiveness of D-FGBB.

This paper is organized as follows. Section 2 gives a background; Section 3 presents our V_{th} variation model; Section 4 introduces D-FGBB; Section 5 selects the body bias cells; Sections 6 and 7 evaluate D-FGBB; and Section 8 discusses related work.

2. Background

Process variation can be die-to-die (D2D) or within die (WID). We focus on the latter, which offers more opportunities for microarchitectural solutions. If required, D2D variation can be modeled by adding a chip-wide offset to the WID variation. We also consider

1072-4451 2007

DOI 10.1109/MICRO.2007.43

temperature (*T*) variation. However, in this paper, we regard voltage (*V*) variation as beyond our scope. In the following, we review transistor leakage, gate delay, and body biasing.

2.1. Transistor Leakage and Gate Delay

Subthreshold leakage is the main source of leakage, given the adoption of high-k gate dielectric materials. The following model for a transistor's subthreshold leakage is based on HotLeakage [48], itself a simplification of the full BSIM3 SPICE model:

$$I_{\text{leak}} \propto (kT/q)^2 e^{\frac{q(V_{off}-V_{th})}{\eta kT}} \quad (1)$$

where k and q are constants, and η and V_{off} are empirically determined parameters. From the equation, transistors with low V_{th} have a high I_{leak}. Moreover, as *T* increases, the transistor quickly becomes leakier, both because of I_{leak}'s dependence on *T* and because V_{th} goes down as *T* goes up [42].

The delay of an inverter gate is given by the alpha-power model [36] as:

$$T_g \propto \frac{L_{\text{eff}} V}{\mu(V - V_{\text{th}})^{\alpha}} \quad (2)$$

where L_{eff} is the effective channel length of a transistor, α is typically 1.3, and μ is the mobility of carriers which, as a function of *T*, is $\mu(T) \propto T^{-1.5}$. As V_{th} decreases, $V - V_{\text{th}}$ increases and the gate becomes faster. As *T* increases, V_{th} decreases and, as a result, $V - V_{\text{th}}(T)$ increases. However, $\mu(T)$ decreases [42]. The second factor dominates and, with higher *T*, the gate becomes slower.

2.2. Body Biasing

Body Biasing (BB) a transistor involves applying a voltage between its source or drain and substrate to alter its V_{th} [24]. In Forward BB (FBB), the voltage polarity is such that V_{th} decreases, creating a faster and leakier transistor. In Reverse BB (RBB), V_{th} increases, creating a slower, less leaky transistor. BB can be applied in a way such that the chip receives a single bias voltage or that it receives different bias voltages in different regions of the chip [45] — we call the latter Fine-Grain Body Biasing (FGBB). We call each of the regions with a different bias voltage a *Cell*.

2.2.1. Uses of BB and FGBB

At least two commercial processors use BB, namely Intel's Xscale [7] and Transmeta's Efficeon [10]. Both apply a single, chip-wide BB. Xscale uses RBB in standby mode to reduce leakage. There are fewer details on Efficeon, but it appears that the chip uses BB either to reduce leakage or to boost frequency. In addition, an experimental 80-core network-on-chip from Intel [46] uses FGBB. Specifically, it uses FBB to increase frequency in active mode and RBB to save leakage power in idle mode.

Another proposed use of BB is to reduce D2D process variation [2, 45]. After fabrication, different dies from the same batch run at different frequencies and leak different amounts. Applying different levels of chip-wide BB to different chips — RBB to high-leaking chips and FBB to slow ones — pushes the chips into a more homogeneous region of operation with acceptable frequency and leakage.

Other work has focused on using FGBB to mitigate WID variation [1, 5, 45]. Specifically, Tschanz *et al*. [45] implement FGBB on a test chip with 21 cells, each containing one critical path and circuitry to determine the optimal BB for the cell. Cells with a slow critical path are made faster with FBB, while cells with a fast (and leaky) critical path are made less leaky with RBB. The result is that WID variation in speed and leakage decreases.

2.2.2. Overhead of BB and FGBB

Implementing BB in a chip requires adding power lines for the BB voltage and including circuitry to determine and generate the optimal BB voltage [24]. In addition, to apply BB to NMOS, the manufacturing process has to be enhanced with a triple-well process [45]. There are three overheads to consider, namely area, power, and time.

The area overhead of BB is examined by Kuroda and Sakurai [24], who discuss various circuits to apply BB. The circuitry that controls the BB is simple, and its area overhead is estimated to be 1% of the chip area. On top of that, there is the area overhead of routing the power lines for BB. However, Narendra *et al*. [28] implement a router chip with BB for PMOS that contains a central bias generator, 24 local bias generators distributed in the chip with their own control circuits, and the needed global routing, and report a full-chip area overhead of 2%. Similarly, in an experimental 150nm FGBB chip, with 21 cells that contain one critical path each, Tschanz *et al*. [45] report an overall chip area overhead due to FGBB of 2-3%. Furthermore, an optimized design of BB circuits using recently-proposed approaches such as Chen and Gregg's [5] or Azizi and Najm's [1] may further reduce the area overhead.

Applying and controlling BB consumes some static and dynamic power. The static power dissipated is proportional to the area and, therefore, is small. The dynamic power consumed charging and discharging the substrate capacitance when BB levels change is small because the currents are small. Overall, according to Kuroda and Sakurai [24], the power overhead of BB and the circuitry that controls it is 1% of the chip's power.

The timing overhead of BB is also negligible. Kuroda and Sakurai [24] present designs that allow large changes in BB voltage to occur in the order of 1μs or 10μs. In this paper, we only change the BB voltage in small increments when *T* changes, which is in the order of ms. Moreover, the processor does not stop while the BB voltage is being adjusted. Finally, Narendra *et al*. [28] report that the presence of the BB circuitry does not hurt the frequency of their router chip. Consequently, we assume there is no timing overhead.

Finally, determining the optimal amount of BB to apply can be done using a circuit that is representative of the critical paths in the cell. Using a phase detector similar to the one used in Razor [11] on that representative circuit, Tschanz *et al*. [45] determine the frequency that the transistors in that cell can support. Based on it, they set the BB to apply to the cell.

3. Process Variation and its Impact

WID process variation is impacted by both systematic and random effects. Limitations of the lithography and other manufacturing processes introduce systematic variations. Typically, such variations exhibit a spatial correlation and, therefore, nearby transistors share similar systematic parameter values [12, 30, 39]. On the other hand, a variety of materials effects such as changes in the dopant density of the channel and lithographic phenomena like line

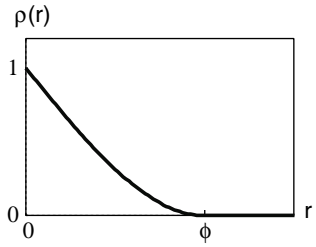

Figure 1. Spherical function.

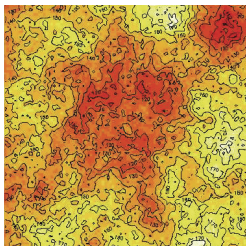

Figure 2. Systematic V_{th} variation map for a chip with $\phi = 0.5$.

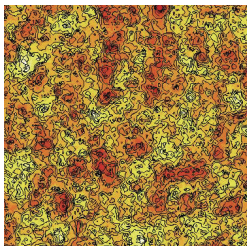

Figure 3. Systematic V_{th} variation map for a chip with $\phi = 0.1$.

edge roughness introduce random variations. Such random variations have a different profile for each transistor.

From a microarchitectural perspective, the variation in V_{th} — and in the related parameter L_{eff} — is of key importance because it directly affects a chip's leakage and frequency. In this section, we outline a model of such variation and its impact on the chip behavior.

3.1. A Model of Process Variation

We treat the systematic and random components of variation separately, since each arises from different physical phenomena. As described in [38], we model these components as normal distributions and assume that their effects are additive:

$$V_{\text{th}} = V_{\text{th}}^{0} + \Delta V_{\text{th}}^{\text{sys}} + \Delta V_{\text{th}}^{\text{rand}} \tag{3}$$

$$L_{\text{eff}} = L_{\text{eff}}^{0} + \Delta L_{\text{eff}}^{\text{sys}} + \Delta L_{\text{eff}}^{\text{rand}} \tag{4}$$

where V_{th}^{0} and L_{eff}^{0} are the nominal values of these parameters, and the deltas are the systematic and random components of variation.

We model systematic variation using a multivariate normal distribution [32] with a spherical spatial correlation structure [8]. For this, we divide the chip with a grid of $n \times m$ points. Each grid point has a value of the systematic component of V_{th} (and L_{eff}) that is distributed normally with zero mean and standard deviation σ_{sys} — where the latter is different for V_{th} and L_{eff}. This is a general approach that has been used elsewhere [38].

For simplicity, we assume that the spatial correlation is homogeneous (position independent) and isotropic (not depending on the direction). This means that, given two points $\vec{x}$ and $\vec{y}$ on the chip, the correlation of their systematic variation values depends only on the euclidean distance between $\vec{x}$ and $\vec{y}$. These assumptions have been used elsewhere [47].

Assuming position independence and isotropy, we can express the correlation function of a parameter P as $corr(P(\vec{x}), P(\vec{y})) = \rho(r)$, where $r = |\vec{x} - \vec{y}|$. By definition, $\rho(0) = 1$ (i.e., totally correlated). Intuitively, $\rho(\infty) = 0$ (i.e., totally uncorrelated) if we only consider WID variation.

To specify how $\rho(r)$ changes between the limits, we choose the spherical function [8]. This function is similar to the correlation function experimentally measured by Friedberg *et al.* [12] for the WID variation of gate length. In the spherical function, $\rho(r) = 1 - (3r/2\phi) + (r/\phi)^3/2$ if $r \leq \phi$; otherwise $\rho(r) = 0$.

Figure 1 plots the function $\rho(r)$. The systematic variation values of a transistor are highly correlated to those of transistors in its immediate vicinity. The correlation decreases approximately linearly with distance at small distances. Then, it decreases more slowly. At a finite distance ϕ that we call *Range*, the function converges to zero. This means that, at distance ϕ, there is no longer any correlation between the values of two transistors.

In this paper, we express ϕ as a fraction of the chip's width. A large ϕ implies that large sections of the chip are correlated with each other; the opposite is true for small ϕ. As an illustration, Figures 2 and 3 show example systematic V_{th} variation maps for chips with $\phi = 0.5$ and $\phi = 0.1$, respectively. These maps were generated by the geoR statistical package [35] of R [33]. In the $\phi = 0.5$ case, we discern large spatial features, whereas in the $\phi = 0.1$ one, the features are small. A distribution without any correlation ($\phi = 0$) appears as white noise.

A former ITRS report [17] projected that the year 2006 design target for the σ/μ of the total L_{eff} variation would be roughly half of that of V_{th}. Lacking better data, we assume that L_{eff}'s σ_{sys}/μ is half of V_{th}'s σ_{sys}/μ. Moreover, since the systematic variation in L_{eff} causes much of the systematic variation in V_{th}, we use the following equation to compute the systematic component of L_{eff} given the systematic component of V_{th} in the same chip grid point:

$$\Delta L_{\text{eff}}^{\text{sys}} = L_{\text{eff}}^{0} \times \frac{\Delta V_{\text{th}}^{\text{sys}}}{2V_{\text{th}}^{0}} \tag{5}$$

Random variation occurs at a much finer granularity than systematic variation — at the level of individual transistors. Hence, rather than augmenting the grid with random effects, we add random variation to the model analytically. We assume that the random components of V_{th} and L_{eff} are both normally distributed with zero mean. Each has a different σ_{rand}. Moreover, the random V_{th} and L_{eff} values for a given transistor are uncorrelated.

Finally, since the random and systematic components of V_{th} and L_{eff} are normally distributed and independent, the total variation is normal with zero mean and a standard deviation of:

$$\sigma = \sqrt{\sigma_{rand}^2 + \sigma_{sys}^2} \tag{6}$$

where V_{th} and L_{eff} have a different σ. More details on our model can be found in [43].

3.2. Impact on Leakage Power and Frequency

We can use the model to estimate the impact of process variation on the leakage power and frequency of a chip. To estimate the

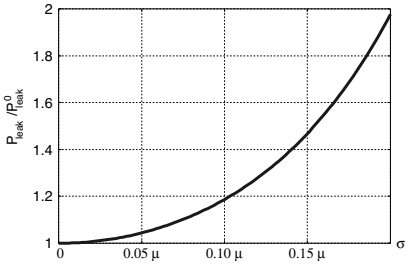

Figure 4. Relative leakage power in the chip as a function of V_{th}'s σ.

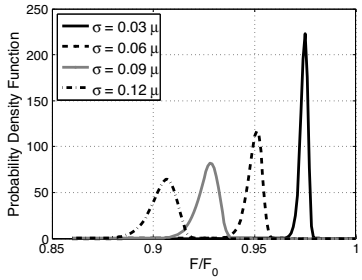

Figure 5. Probability distribution of the relative chip frequency as a function of V_{th}'s σ.

leakage power, we start by integrating Equation 1 over all the transistors in the chip, while using our V_{th} distribution. The result is the total leakage current in the chip. Let P_{leak} and I_{leak} be the chip's leakage power and current under variation, P^0_{leak} and I^0_{leak} the same parameters when there is no variation, and σ the standard deviation of V_{th}'s variation. The ratio of post-variation to pre-variation leakage is:

$$P_{\text{leak}}/P^0_{\text{leak}} = I_{\text{leak}}/I^0_{\text{leak}} = e^{(q\sigma/\eta kT)^2/2} \quad (7)$$

which shows that the increase in the chip's leakage power and current depends on σ. Figure 4 plots the relative leakage power as a function of σ for T=25 $^{\text{o}}$C, V_{th}'s mean μ=150mV, and ϕ=0.5. Typical values of V_{th}'s σ that we will use are 0.09–0.12$\times\mu$. We can see that the leakage increases rapidly as σ goes up.

Given that V_{th} is normally distributed and that the systematic components of L_{eff} and V_{th} are correlated as per Equation 5, it can be shown that the gate delay of Equation 2 is approximately normally distributed. Assuming that every critical path in a processor consists of n_{cp} gates, and that a modern processor chip has thousands of critical paths, Bowman *et al*. [3] compute the probability distribution of the longest critical path delay in the chip ($\max\{T_{\text{cp}}\}$). Such path determines the processor frequency ($1/\max\{T_{\text{cp}}\}$). Using this approach, we can estimate how the value of V_{th}'s σ affects the chip frequency.

Figure 5 shows the probability distribution of the chip frequency as a function of σ for the same conditions as Figure 4, n_{cp}=12 FO4s, and 10,000 critical paths. The frequency is given relative to a processor without variation (F/F_0). The figure shows that, as σ increases, (i) the mean chip frequency decreases and (ii) the chip frequency distribution gets more spread out. In other words, given a batch of chips, as V_{th}'s σ increases, the mean frequency of the batch decreases and, at the same time, individual chips' frequencies deviate more from the mean.

In conclusion, V_{th} variation is very detrimental to a chip's frequency and leakage power. As V_{th}'s σ increases, chip leakage increases rapidly, and the mean chip frequency decreases and varies more. Therefore, we would like to find a way to reduce V_{th} (and L_{eff}) variation.

4. Dynamic Fine-Grain Body Biasing

Judicious application of FGBB can redress the problem of WID V_{th} variation. As suggested by Tschanz *et al*. [45], RBB can be applied to cells with low V_{th} and FBB to cells with high V_{th}. The net effect is to lower V_{th}'s σ. As a result, the chip may increase its frequency, reduce its leakage, or a combination of both.

While Tschanz *et al*. proposed to use FGBB statically, we propose to use FGBB dynamically. Moreover, our approach and goal are different than Intel's 80-core network-on-chip [46]. In the latter, active cores receive FBB to increase their frequency and idle cores receive RBB to reduce their leakage.

Our approach is different in two ways. First, we apply D-FGBB in a *fine time scale*, adapting it as an application runs and the T changes. Secondly, we are redressing parameter variation within a core. Our goal is different in that we want to run a core at the highest frequency and/or at the lowest power that can be attained *at any given time*. In this section, we propose a mechanism to apply D-FGBB and use it in different scenarios.

4.1. A Mechanism to Apply D-FGBB

To implement FGBB in a chip, we divide it into cells that can be body-biased independently. In each cell, we add a Local Bias Generator, which is a simple circuit to generate the BB voltage. Then, we determine the optimal BB voltage that should be applied to each cell. BB essentially trades off leakage power for delay. The optimal BB voltage is therefore the one that results in the minimum leakage consumption while ensuring that all the critical paths in the cell meet timing. The optimal BB voltage is therefore a function of the cell's critical path delays, which in turn are dependent on the V_{th} and T distributions.

Analytical solutions to determine the optimal BB voltage are not practical because of the non-determinism caused by V_{th} and T variation. We instead rely on direct measurement of critical path delay to determine the BB voltage to apply to each cell.

In [45], a dedicated control circuit estimates the delay of the transistors in the cell and adjusts the BB for the cell accordingly. The circuit consists of a critical path replica and a phase detector that recognizes when the critical path replica is not meeting the target frequency. A feedback mechanism is used to adjust the BB of the cell until the target frequency is met.

We modify that design to work for D-FGBB. A diagram of our circuit for a single cell is shown in Figure 6. We use multiple critical-path replicas distributed across the cell. This allows for a more accurate assessment of the cell's delay, in the presence of variation. Each critical path replica is paired with its own phase detector, forming what we call a Sample Point (Figure 6).

In cases of severe variation, it may happen that none of the critical path replicas captures the worst-case delay of the cell. This will be detected during normal testing of the chip. To solve this problem, we add some inverters to one of the critical path replicas of each cell

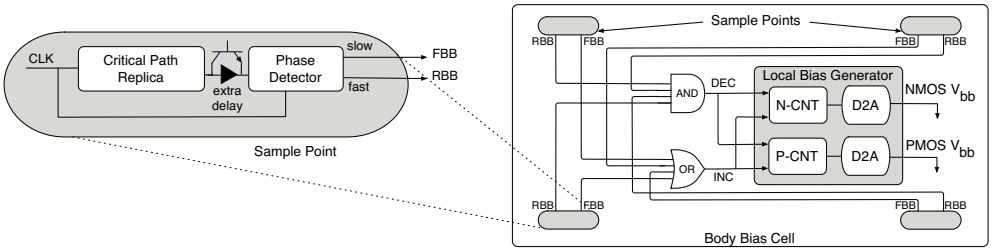

Figure 6. Proposed circuit to support D-FGBB in a cell.

(Figure 6). These inverters are normally bypassed by pass transistors. If a cell fails to meet the target timing during testing, some of the pass transistors in its corresponding critical path replica are enabled. This increases the delay of the critical path replica so that it becomes representative for that cell.

We use a bidirectional phase detector that identifies when the frequency supported by the critical path replica is noticeably higher than or not as high as current conditions. In the former case, it raises the RBB signal; in the latter, it raises the FBB signal (Figure 6). This allows the circuit to fine-tune the BB voltage applied dynamically, by either increasing or decreasing it depending on the signal raised. It saves both time and power compared to the unidirectional calibration performed statically in [45] — which starts at the maximum RBB and gradually reduces it, finally applying FBB until the target frequency is met.

An alternative to using critical path replicas is to directly measure the delay of the actual critical paths as proposed in [9]. The critical paths in each cell are identified by the CAD tools and their inputs and outputs sampled after fabrication by a circuit similar to our phase detector. There are two advantages to this solution. First, it incurs a smaller area overhead because it does not need to replicate critical paths. Secondly, it can be more accurate because it measures the actual critical paths rather than replicas. The downside is that it is more intrusive to the hardware design.

Each cell has a local bias generator that generates separate BB voltages for NMOS and PMOS transistors (Figure 6). This is because NMOS and PMOS transistors can be affected differently by variation and have different optimal BB voltages. The BB values for PMOS and NMOS are stored in two bidirectional counters called P-CNT and N-CNT, respectively. The counters are incremented and decremented dynamically. Their initial values are set at a post-manufacturing calibration phase that determines the optimal BB values for NMOS and PMOS at a calibration temperature (Section 4.2). Thereafter, their values change only together by the same amount.

The counter values are converted to voltages by two digital-to-analog (D2A) converters based on a resistor ladder and an OP-AMP. By setting the appropriate reference voltages to the resistor ladders, and incrementing/decrementing the counters, the BB voltages range from a maximum RBB of -500mV to a maximum FBB of 500mV in 32mV steps. This has the effect of changing V_{th} by a range of $\pm$ 70mV.

The conditions for changing the BB are as follows. The counters are incremented as long as *at least one* of the critical path replicas asserts its FBB signal. This ensures that the cell receives higher BB until the slowest critical path replica meets timing. The counters are decremented as long as *all* the critical path replicas assert their RBB signals. This means that the cell receives lower BB only as long as all the critical path replicas are faster than the desired delay. This saves leakage power while meeting the target frequency. When neither of the above conditions is met, the counters hold their current values.

4.2. Static Calibration of FGBB

We envision that the chip manufacturer calibrates the FGBB for each chip. The goal is to bring each chip to its best frequency-leakage operating point before shipment. The calibration is performed in a controlled environment, at worst-case temperature conditions T_{cal} (for Calibration Temperature). This ensures that the chip will function properly at any *T*. At the same time, the manufacturer runs a set of test vectors designed to exercise the chip at full load and generate the maximum dynamic power dissipation.

This process first sets the initial values of the P-CNT and N-CNT counters in all cells. The goal is to correct the potential imbalance between the V_{th} of the NMOS and PMOS transistors. Using a simple circuit like the one proposed in [13, 29], each cell measures the local imbalance. Then, the cell increments the counter of the slower transistor type so that the imbalance is eliminated. For cells with no V_{th} imbalance, P-CNT and N-CNT remain at 0.

Next, an initial *Target Frequency* F^0_{cal} is selected and applied to the chip. F^0_{cal} can be a fixed percentage of the frequency expected for a chip with no V_{th} variation. In each cell, the phase detectors automatically time the critical path replicas and the local bias generator sets the optimal BB voltages for PMOS and NMOS. After the cells have been body-biased, the chip's total power is measured — both dynamic and leakage power. If the power is below a tolerable maximum value, F^0_{cal} can be increased; if it is above, F^0_{cal} must be decreased. Let us call F^1_{cal} the new frequency. The calibration process is then repeated for F^1_{cal}. This process may be repeated a few times, each time decreasing ΔF_{cal} until the highest possible frequency, subject to the power constraint, is found. Let us call the final value F_{cal}.

4.3. Using D-FGBB to Save Leakage Power

We propose to use our D-FGBB control circuit of Figure 6 to save leakage power by continuously adapting BB voltages as *T* changes — without changing the frequency. Recall that, as *T* goes up, gate delay goes up (Section 2.1). As a result, a chip's critical path delays also increase. The static FGBB (S-FGBB) settings are necessarily conservative because they are calibrated using the conservatively high T_{cal}. In reality, there is a significant *T* variation across and within workloads.

To illustrate it, Figure 7 shows, for each unit in a processor, the T_{max} and T_{avg} of that unit when running a sequence of SPECint

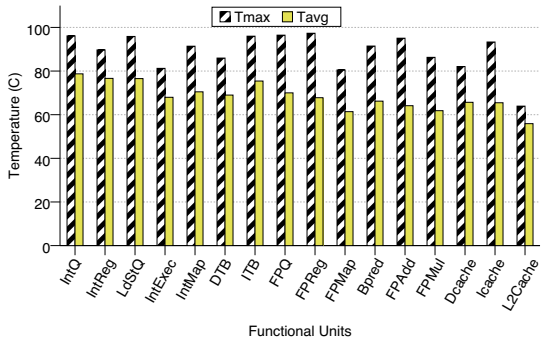

Figure 7. T_{max} and T_{avg} in different units of a processor running a sequence of SPECint and SPECfp codes.

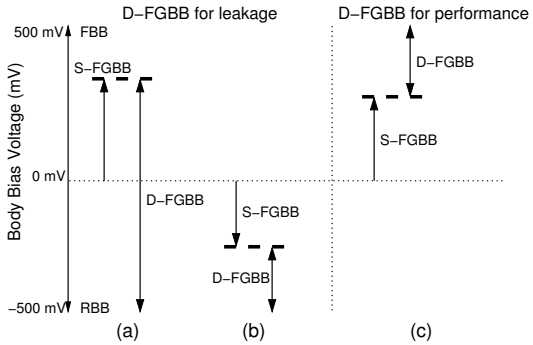

Figure 8. Changing the BB voltage with D-FGBB.

and SPECfp codes. The difference between T_{max} and T_{avg} is often 20-30 ^{o}C. The initial S-FGBB calibration is performed at a T_{cal} that is higher than the highest T_{max}, while units typically operate at close to T_{avg}. As a result, critical paths are generally faster than during calibration, and we can reduce the BB applied and save leakage.

When the chip is first powered-up, each cell's BB is set to its S-FGBB calibration value. As the chip workload changes, our D-FGBB control circuit adjusts the BB for each cell to the optimal value for the current *T*. This is done without changing the frequency. Figures 8(a) and (b) show the BB voltage values under S-FGBB and D-FGBB. In (a), the cell is initially slow, and S-FGBB applies FBB at calibration time. Then, D-FGBB can dynamically reduce the FBB and even apply RBB to save leakage power. Figure 8(b) shows the case when the cell is initially fast enough to meet the target frequency, and S-FGBB applies RBB at calibration time. D-FGBB can still save additional leakage by applying further RBB when conditions permit.

D-FGBB dynamically adjusts the BB voltages without stopping the running application. Consequently, it induces no time overhead.

4.4. Using D-FGBB to Improve Performance

A second use of D-FGBB is to increase performance by adapting frequency and BB voltages as power consumption changes. This approach is used when the user wants to run the processor in a *high-performance* mode, where the goal is to deliver the highest possible performance while staying within the chip's power budget (P_{max}) at all times.

The insight that enables this mode of operation is that the manufacturer determines the chip's frequency F_{cal} conservatively, assuming a worst-case power consumption P_{max} — including worst-case dynamic power consumption — in addition to worst-case T_{cal}. Since he assumes the maximum dynamic power, he imposes a conservative limit on the leakage power — such that when both are added together, P_{max} is not exceeded. At run time, such maximum dynamic power is not always reached. Consequently, as long as P_{max} is not exceeded, we can dynamically increase the frequency beyond F_{cal} — which will increase the dynamic power and, at the same time, require our D-FGBB circuit to increase the BB of cells. The approach is shown in Figure 8 (c).

To support this mode of operation, we extend the S-FGBB calibration process. Specifically, recall that we calibrated the chip's F_{cal} under full load, generating the maximum dynamic power dissipation (Section 4.2). After this, we place the chip in idle mode, to dissipate little dynamic power, and repeat the calibration. Because the dynamic power is low, more leakage power can be expended. This in turn allows higher BB in the cells, enabling a higher processor frequency. The resulting frequency (F_{cal}^{max}) is the absolute maximum frequency that the chip's circuits can meet while not exceeding P_{max}. F_{cal} and F_{cal}^{max} are recorded in a programmable table on-chip; they are used as lower and upper bounds, respectively, on the processor frequency in high-performance mode.

At run time, the processor starts at F_{cal}. As it runs under load, it adjusts its frequency at regular intervals, taking values between F_{cal} and F_{cal}^{max}, depending on the current power consumption of the chip. We assume that the chip includes circuits to measure average power, possibly like those in Itanium's Foxton [27]. As long as the average power is less than P_{max} and the processor is under load, the frequency is increased. To meet the new frequency, the D-FGBB control circuit quickly increases the BB levels. Safety mechanisms are in place to ensure that P_{max} or T_{cal} are not exceeded. If this is about to happen, the frequency is reduced.

4.5. D-FGBB and Dynamic Voltage Scaling

While D-FGBB trades-off circuit delay for leakage power, Dynamic Voltage Scaling (DVS) largely trades-off circuit delay for dynamic power. Consequently, we can combine both techniques so that, for a given frequency of operation — e.g., F_{cal} — the processor consumes less power than with either technique alone. Previous work has shown that BB can complement DVS to improve the power savings of DVS alone [26]. However, that work decided the optimal combination of techniques using an analytical expression, which is not suitable in the presence of variation.

A given circuit delay can be obtained with different combinations of supply voltage and BB values, each with a different power cost. In some cases, more power can be saved with a lower supply voltage (saving dynamic power) and a higher BB to compensate for the circuit slowdown (consuming more static power). In other cases, it is better to have a higher supply voltage (consuming more dynamic power) and use up the time slack with a lower BB (saving leakage power). The best approach depends on the fraction of power dissipated of each type and on the *T*.

We propose to augment the S-FGBB calibration process of Section 4.2 with one additional step to find a configuration that sub-

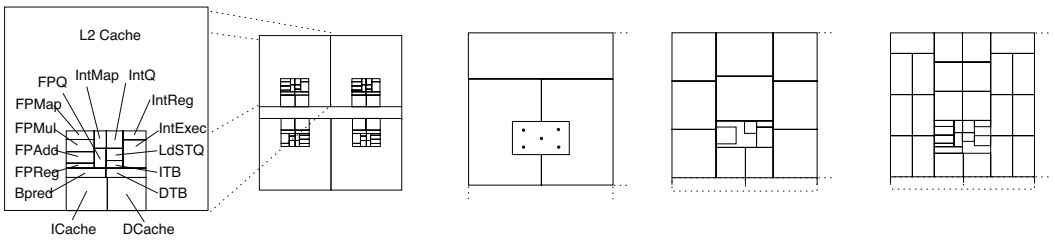

Figure 9. CMP floor-plan used (a) and the partitioning of one processor and its share of the bus into BB cells (b–d). Chart (b) shows the five critical path replicas in one cell.

stantially reduces the power consumed at F_{cal}. Specifically, after the manufacturer has set the BB voltages for each cell at F_{cal}, he proceeds as follows. The supply voltage is reduced in small steps. At each step, our D-FGBB circuit of Figure 6 recomputes the BB values, and the total power in the chip is also measured. When the voltage drops so much that F_{cal} can barely be met, the process stops. Then, we select the combination of supply voltage and BB values that consumes the least power. If the processor has multiple DVS domains (e.g., one for the core and one for the L2), this algorithm is first run reducing the voltage of one domain only. Once the best configuration is found, the configuration is used to run the algorithm reducing the voltage of another domain, and so on.

5. Selecting the BB Cells

Microarchitectural structure plays an important role in deciding how to partition the chip into BB cells. There are advantages to using BB cells with shapes that follow the contour of microarchitectural modules such as caches, registers, or execution units. We suggest two main reasons for this, namely variations in *T* and differences in the types of critical paths in different modules.

5.1. Temperature Effects

Equations (1) and (2) show that *T* significantly affects transistor leakage and gate delay. At high *T*, transistors become vastly leakier and gates slower. As a result, the BB voltage applied can be better targeted if *T* does not vary much within a cell. It is well known that the spatial *T* profile in a chip under load follows the layout of microarchitectural modules. For example, the execution unit is hot while the L2 cache is cold. Consequently, we propose organizing the chip into cells that follow the contours of groups of hot and groups of cold microarchitectural modules.

5.2. Critical Paths in Logic and Memories

Different microarchitectural modules have different types of critical paths. This is most obvious when comparing logic blocks such as functional units to memory structures such as the L1 cache or TLB. In the former, a critical path contains many, physically close gates and a modest amount of wire — e.g., 8-16 FO4-equivalent gates in high-end processors connected by short wires. In contrast, the critical path in memory structures has a few, physically separated transistors and much more wire — e.g., the path that stretches from a driver through a word line, a pass transistor, a bit line, and then to a sense amplifier.

From a V_{th} variation point of view, these two critical paths differ dramatically. The transistors in a logic path are many and physically close. Their large number enables a better averaging of random V_{th} variations, while physical proximity makes them subject to the same systematic V_{th} variation. On the other hand, the transistors in the memory path are few and distant from each other. Fewer transistors means less averaging of random V_{th} variations, while farther distances implies better averaging of systematic V_{th} variations. Since these two types of critical paths are affected differently by a given BB voltage, we separate logic and memory structures into different BB cells.

6. Evaluation Methodology

6.1. Processor Chip Architecture

We use detailed simulations using the SESC [34] cycle-accurate simulator to evaluate a chip multiprocessor (CMP) with four high-performance processors at 45nm. The processor is based on the Alpha 21364, and has a 64KB L1 I-cache, a 64KB L1 D-cache, and a 2MB L2 cache. We estimate a nominal frequency of 4GHz with a supply voltage of 1V. We generate the processor layout from the Alpha 21364 chip floor-plan, without the router and I/O pads, and with an L2 cache as in [37]. We use constant scaling to scale the dimensions to 45nm. Finally, we put four such units on a chip, and interconnect them with a wide snoopy bus. The resulting 8MB L2 cache is shared by all the cores. The resulting 132 mm^2 chip is shown in Figure 9(a).

6.2. Power and Temperature Model

To estimate power, we scale the results given by popular tools using technology projections from ITRS [18]. Specifically, we use SESC augmented with dynamic power models from Wattch [4] to estimate dynamic power at a reference technology and frequency. In addition, we use HotLeakage [48] to estimate leakage power at the same reference technology. Then, we obtain ITRS's scaling projections for the per-transistor dynamic power-delay product, and for the per-transistor static power. With these two factors, given that we keep the number of transistors constant as we scale, we estimate the dynamic and leakage power for the scaled technology and the frequency relative to the reference values.

We use HotSpot [37] to estimate the on-chip *T* profile. To do so, we use the iterative approach of Su *et al.* [40]: the *T* is estimated based on the current total power; the leakage power is estimated based on the current *T*; and the leakage power is added to the dynamic power. This is repeated until convergence. In our experiments, the maximum temperatures reached in the chip are in the 95-100 $^{\text{o}}$C range.

We run several applications from SPECint (*bzip2, crafty, gap, gzip, mcf, parser, twolf*) and from SPECfp (*applu, equake, mesa, mgrid, swim*). A workload consists of running four instances of the same application at a time, one on each core. We use the reference input set for 1B instructions after discarding the first 1B instructions.

6.3. Critical Path Model

We do not have access to detailed information on the structure and distribution of a processor's critical paths. For this reason, we build a simple model that we use in our experiments. Specifically, we design different critical paths for logic modules, small memories, and large memories. For logic modules, we model a critical path as 12 FO4 gates connected in series by short wires. The wires account for 35% of the path delay [15]. We use CACTI [41] to estimate wire delays and Equation 2 to compute gate delays. For memory modules, we separate large memories (the two L1 caches) from the remaining SRAM structures (e.g., the register file). The latter are assumed to cycle at twice the frequency of the former. We use CACTI to determine the optimal sub-array sizes and the physical layout. In both structures, a critical path stretches from a driver driving a word line, through the word line, a pass transistor, the bit line, and the sense amplifier. We model the path as three logic gates connected by word- and bit-line wires laid out as per CACTI.

We model each logic module and each memory module as having many, spatially-distributed critical paths. Specifically, we use Bowman *et al.*'s estimate that a high-performance processor chip at our technology has about 10,000 critical paths [3]. We distribute these paths uniformly on the area taken by the cores and L1 caches — we assume that the L2 and the bus do not have critical paths. Each module gets critical paths of its type. Finally, as we superimpose the V_{th}, L_{eff}, and T variation maps on the chip, parameter variation impacts the delay of these paths. The frequency supported by a module is determined by the slowest of its critical paths.

6.4. Variation Model Parameters

We only model WID variation. Table 1 shows some of the parameter values used. For V_{th}'s σ/μ, the 1999 ITRS [17] gave a design target of 0.06 for year 2006 (although no solution existed); however, the projection has since been discontinued. On the other hand, Kahng [19, 20] reckons that the ITRS variability projections (for at least the gate-length parameter that he examines) are too optimistic. Consequently, we use a default V_{th}'s σ/μ of 0.12, which we vary in some experiments. Moreover, according to [21], the random and systematic components are approximately equal. Hence, we assume that they have equal variances. This means that, using Equation 6, $\sigma_{sys} = \sigma_{rand} = \sigma/\sqrt{2}$.

Parameter Values
Tech: 45nm; Nominal frequency: 4GHz; V_{dd}: 1V; T_{cal}: 100 °C V_{th}: μ: 150mV at 100 °C; σ/μ: 0.12; $\sigma_{sys} = \sigma_{rand} = \sigma/\sqrt{2}$; ϕ: 0.5 L_{eff}: σ/μ: 0.5 $\times V_{\text{th}}$'s σ/μ; $\sigma_{sys} = \sigma_{rand} = \sigma/\sqrt{2}$; ϕ: 0.5
Body bias application: Maximum bias: $\pm$500mV; Resolution: 32mV Number of FGBB cells per chip: 16, 64, or 144 Number of critical path replicas per cell: 5 Number of chips per experiment: 200

Table 1. Parameter values used.

To set V_{th}'s ϕ, we note that Friedberg *et al.* [12] found that the gate length had a correlation range close to 0.5 of the chip's width. Since the systematic component of V_{th}'s variation directly depends on the gate length's variation, we use a default ϕ = 0.5 for V_{th}.

As indicated in Section 3.1, based on the 1999 ITRS [17], we set L_{eff}'s σ/μ to 0.5 of V_{th}'s σ/μ. Moreover, for L_{eff}, we also assume that $\sigma_{sys} = \sigma_{rand} = \sigma/\sqrt{2}$ and ϕ = 0.5.

Each individual experiment uses a batch of 200 chips that have a different V_{th} (and L_{eff}) map generated with the same μ, σ, and ϕ. To generate the per-chip V_{th} and L_{eff} maps, we use the geoR statistical package [35] of R [33]. We use a resolution of 1M grid points per chip. To relate BB to V_{th}, we use the nonlinear formula from [42] that takes into account short-channel effects.

6.5. BB Environments Evaluated

We evaluate chips with FGBB applied at different granularities, from the trivial case that has a single BB cell (*FGBB1*), to environments with 16, 64 and 144 BB cells per CMP chip (*FGBB16, FGBB64, and FGBB144*). When partitioning a chip into cells, we first separate groups of hot units from groups of cold ones. Then, in each group, we separate logic, large memories, and small memories (Figures 9(b) to (d)). A large module like the L2 cache is broken into multiple cells. Each cell has five uniformly-spaced critical path replicas (Figure 9(b) shows them for one cell). The slowest of such replicas determines the cell's BB voltage.

We consider three different scenarios: FGBB set statically (*S-FGBB*), FGBB set dynamically as the chip runs (*D-FGBB*), and no BB applied (*NoBB*). As a reference, we also consider chips with no process-induced V_{th} variation (*NoVar*). Note that *NoVar*'s V_{th} is not constant due to T variation. For our DVS experiments, we use one DVS domain per processor and one for the L2 cache.

In D-FGBB, the BB voltage changes infrequently because it tracks gate delay changes due to *T*. A BB update occurs when the delay changes by $\approx$2%, which corresponds to a *T* change of $\approx$5 °C. We assume that the circuit in Figure 6 can detect such delay changes. Otherwise, we can use more elaborate circuits, which have high accuracy [14, 22]. In our D-FGBB simulations, we recompute the BB every 2ms.

7. Evaluation

We first assess the effect of V_{th} variation on frequency and leakage. Then, we focus on how D-FGBB improves on S-FGBB in three scenarios: normal operation, high performance, and low power.

7.1. Characterizing V_{th} Variation

Figure 11 shows chip frequency (a) and chip leakage power (b) as V_{th} variation (measured in σ/μ) changes. For each value of σ/μ, the figure shows bars for three different ϕ. In all cases, frequency and leakage are relative to the *NoVar* chip. The bands in the bars show the variation across chips in the batch.

As V_{th} variation increases, the average frequency of the chips decreases and their average leakage power goes up. On average, at 0.12 variation and ϕ=0.5, the frequency is 10% lower and the leakage over 20% higher. Clearly, variation is undesirable. The long bands show high variation across chips in the batch. This is due to the *T* variation. At high *T*, a transistor becomes slower and leakier. Consequently, if transistors with very high V_{th} happen to

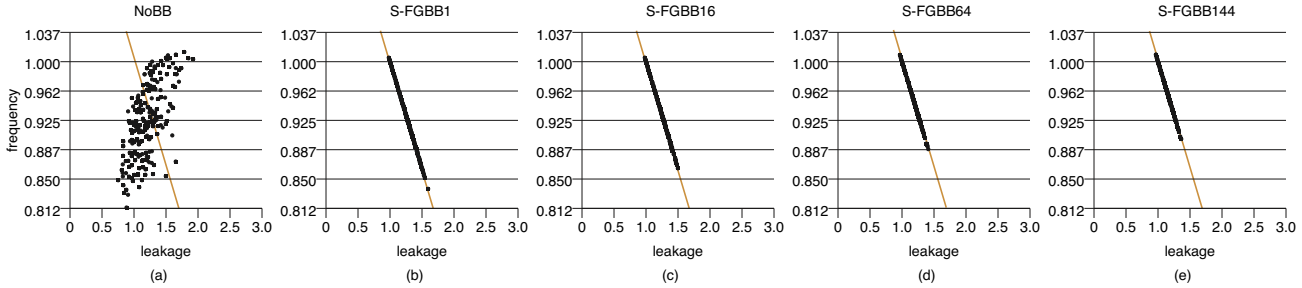

Figure 10. Frequency versus leakage power for a batch of 200 chips at T_{cal} and full load under various schemes.

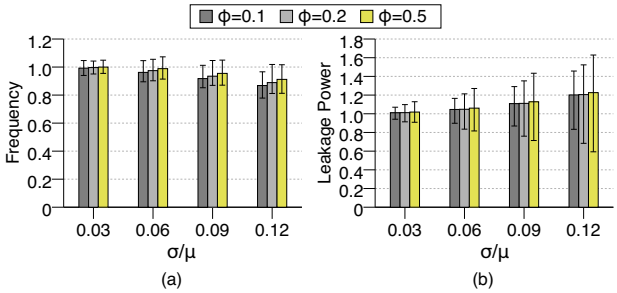

Figure 11. Impact of V_{th} variation on the chip's frequency (a) and leakage power (b).

"fall" on the hottest region of the chip, the chip is likely to have low frequency. On the other hand, if many transistors with very low V_{th} fall on the hottest area, the chip is likely to have high leakage.

We see two main trends. First, across chips in one experiment, leakage varies more than frequency — since leakage is exponential with T, an unfavorable V_{th} distribution can significantly increase leakage power. Second, as ϕ decreases, the average frequency decreases as well. The reason is that, given a set of high-V_{th} transistors, if they are uniformly spread out in the chip (low ϕ), there is a higher chance that some will fall on the hottest region of the chip, thus reducing the chip's frequency.

7.2. Normal Operation: D-FGBB Improves a Chip's Operating Point

S-FGBB can be used to tune the chips in a batch so that they fall into desirable frequency-leakage bins [45]. The goal is to place each chip at the highest possible frequency bin where it still meets the power consumption constraint. In this section, we summarize the impact of S-FGBB and then show how D-FGBB further improves a chip's operating point.

The Acceptable Region for a chip [45] is bounded by two conditions: (i) the frequency should be higher than a given minimum value, and (ii) the sum of dynamic and leakage power should be less than a given maximum value. In a frequency-leakage plot such as Figure 10(a), these constraints require that the chip be above a horizontal line and to the left of a slanted line, respectively. The slanted line has this shape because, as frequency increases, the dynamic power increases linearly and, therefore, the amount of tolerable leakage power decreases linearly. Inside the Acceptable Region, higher frequency is better.

Figure 10(a) shows a scatter plot of the frequency and leakage power for our 200 chips, with axes normalized to *NoVar* (no process-induced V_{th} variation). We build the slanted line so that it would include the *NoVar* chip, which is point (1,1). We then arbitrarily set the horizontal line to 0.85 of the frequency of the *NoVar* chip, and divide the range into four equally-spaced frequency bins. As a fraction of the *NoVar* frequency, the ranges of the bins are: 0.850–0.887, 0.887–0.925, 0.925–0.962, and over 0.962. These bins are in the ballpark of those used in commercial processors.

7.2.1. Impact of S-FGBB

In Figure 10(a), some chips fall outside the Acceptable Region. By applying S-FGBB to a chip, we can move it into the Acceptable Region or, if it is already there, move it to a higher frequency point. Using the axes and the slanted line of Figure 10, Figure 12 graphically shows the impact of our S-FGBB calibration algorithm of Section 4.2.

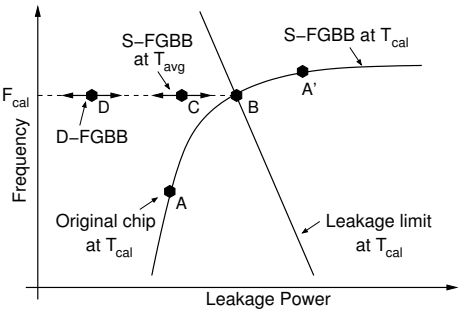

Figure 12. Impact of S-FGBB and D-FGBB on a chip's operating point.

Consider a chip that is originally operating at point *A*. Our algorithm can move the chip along the curve labeled *S-FGBB at* T_{cal}. The result of the algorithm is to bring the chip to point *B*, at frequency F_{cal}, where the chip dissipates the maximum allowed power — thus, point *B* is on the slanted line. Point *B* is more desirable than *A* in that it is inside the Acceptable Region and is potentially in a higher frequency bin than *A*. Increasing the frequency beyond F_{cal} would push the chip to the left of the slanted line, where power consumption is excessive. In cases where the original chip is operating at point *A'*, the S-FGBB algorithm reduces the frequency and brings it to point *B*.

The actual curve followed from *A* depends on the number of FGBB cells. The schemes with more cells such as FGBB144 target their BB voltages better and push the chip to a *B* position that is higher in the slanted line — thus delivering chips in better bins.

To show it, we take the batch of chips of Figure 10(a) and apply our S-FGBB algorithm using the FGBB1, FGBB16, FGBB64, and FGBB144 schemes. The resulting frequency-leakage scatter plots are shown in Figures 10(b)-(e). The charts show that all the schemes

move practically all the chips to the slanted line, in the Acceptable Region. However, the schemes differ in how high they push the chips. The more BB cells they use, the more effective they are.

The different impact of the schemes is best seen in Figure 13, which shows how many chips fall in each frequency bin for the different schemes as a fraction of the 200 chips. Chart (a) corresponds to our experiment, while (b) repeats it for V_{th}'s $\sigma/\mu = 0.09$.

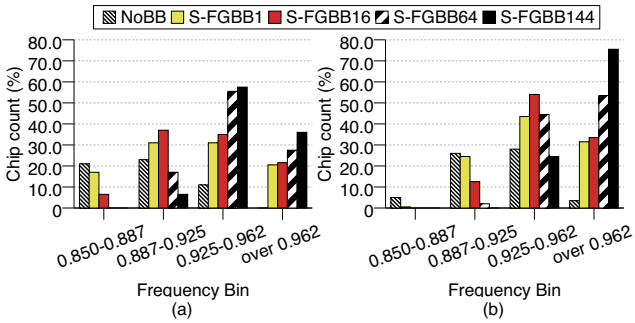

Figure 13. Frequency binning obtained by S-FGBB with different numbers of BB cells, for $\sigma/\mu = 0.12$ (a) and $\sigma/\mu = 0.09$ (b).

Figure 13(a) shows that FGBB64 and FGBB144 move many chips to the top bin. Specifically, FGBB144 has 36% of the chips in the top bin and 93% in the top two. On the other hand, NoBB has none in the top bin and only 11% in the top two. Chart (b) shows that the trends are the same for $\sigma/\mu = 0.09$. Specifically, as we go from NoBB to FGBB144, the number of chips in the top bin changes from 4% to 75%. Consequently, our results are valid for smaller variations as well.

In the rest of the paper, when we refer to the average frequency and leakage of the NoBB or other schemes, we count all the chips — rather than dropping from the average those that fall outside the Acceptable Region. While in a practical environment they would be dropped, we feel the results are more intuitive this way.

7.2.2. Leakage Reduction with D-FGBB

Applying the D-FGBB algorithm of Section 4.3 can substantially reduce the leakage power consumed by the chip. To see it graphically, consider Figure 12. The chip was calibrated with S-FGBB at T_{cal}, resulting in point *B*. However, given that the chip's *T* during execution is close to T_{avg}, the chip typically operates around point *C*, moving to the left and right as shown depending on the current *T* conditions. If we apply D-FGBB, we push the chip's working point to moving around point *D* in the figure. The result is leakage power savings.

Figure 14(a) compares the leakage power of the chips under NoBB, and with 1, 16, 64, or 144 cells under S-FGBB and D-FGBB. We report the average across all the applications and normalize the bars to NoBB. We see that D-FGBB reduces the leakage substantially over S-FGBB. Specifically, with D-FGBB, the leakage power is reduced by 28–42% compared to S-FGBB — where the highest reductions correspond to the chips with more cells. In all cases, S-FGBB dissipates about the same amount of leakage power as NoBB.

Figure 14(b) shows the total power in this experiment. The figure also includes an environment with DVS alone and one where D-FGBB is combined with DVS as detailed in Section 4.5. All bars

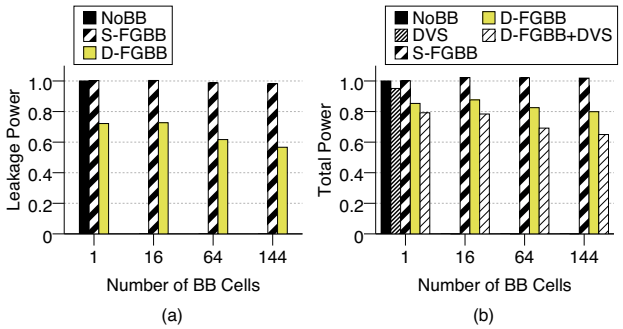

Figure 14. Leakage (a) and total power (b) of the chips for different FGBB schemes in normal operation.

are normalized to NoBB. Recall that, as we increase the number of cells, the frequency increases. However, for the same number of cells, the frequency is the same. From the figure, we see that D-FGBB reduces the total power consumption by 15–22% relative to S-FGBB for the same frequency, with the higher reductions corresponding to the schemes with more cells. If we combine D-FGBB and DVS, the total power saved is 21–36% of the S-FGBB power — again, with the schemes with more cells doing the best. This large impact is possible because DVS lowers the voltage of the domain that dissipates the most dynamic power (namely, the core), while D-FGBB applies higher BB to ensure that the target frequency is met. This results in dynamic power savings that add to the leakage savings of D-FGBB. Finally, DVS alone can only reduce less than 5% of the power in NoBB. This is because the voltage can be lowered little while still meeting the target frequency.

7.3. High Performance: D-FGBB Improves Frequency

A second application of D-FGBB is to improve performance by increasing the average frequency of a chip beyond the F_{cal} determined at calibration (Section 4.4). Figure 15 compares the average frequency of the chips with S-FGBB and this use of D-FGBB. The figure considers chips with different numbers of cells, and normalizes the bars to NoBB. We see that D-FGBB increases the frequency by 7–9% over S-FGBB for the same number of cells. Compared to NoBB, the frequency increase is 7–16%. With more cells, the frequency is higher because BB can be tuned better.

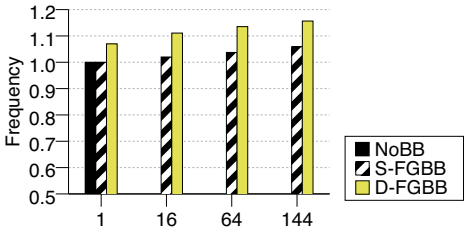

Figure 15. Average frequency of the chips for different FGBB schemes.

The frequency increase varies across applications, depending on their dynamic power consumption. Those with low dynamic power consumption see the biggest boosts in frequency. However, applications benefit differently from a frequency boost, depending on whether they are memory- or compute-intensive. Figure 17 compares the execution time of the applications with S-FGBB144 and

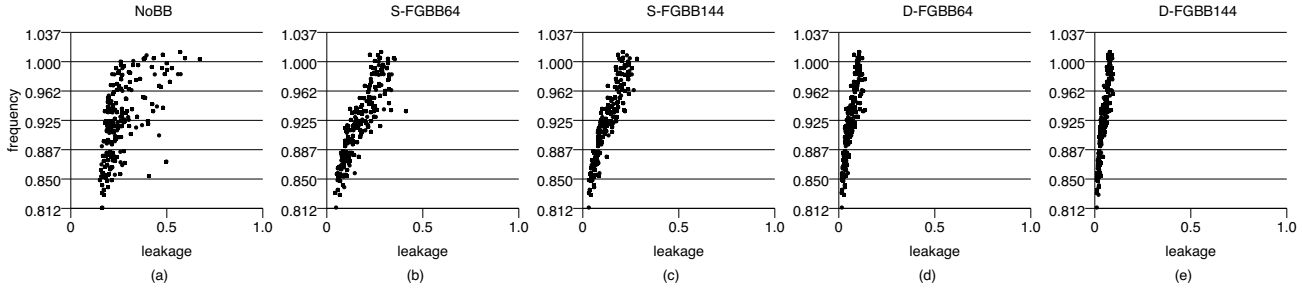

Figure 16. Frequency versus leakage power for a batch of 200 chips at usual T and load conditions.

D-FGBB144. In the figure, the bars are normalized to NoBB. On average, D-FGBB144 reduces the execution time by 6% over S-FGBB144. Moreover, compared to NoBB and S-FGBB1 (not shown in the figure), the reduction is 10%.

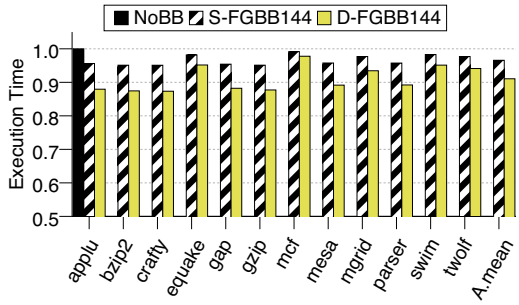

Figure 17. Execution time of the applications for different FGBB schemes.

The speedups delivered by D-FGBB come at a significant cost in total power consumption. Increasing the frequency induces higher dynamic power; applying the more aggressive BB voltage needed to increase frequency induces higher leakage power. The resulting total power for S-FGBB and D-FGBB is shown in Figure 18. Because of the high power cost, this mode of operation is only appealing when the highest possible performance is needed.

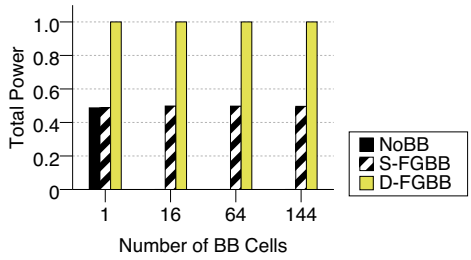

Figure 18. Total power of the chips for different FGBB schemes.

7.4. Low Power: D-FGBB Reduces Leakage

Finally, we consider an environment where we do not attempt to improve the original frequency of the chip with the S-FGBB calibration step of Section 4.2. Instead, we take each chip in the batch in turn, identify the frequency at which it runs, and then apply D-FGBB (or S-FGBB) to save leakage. Our goal is to save as much leakage as possible. We call this environment *low power* mode.

7.4.1. Constant Frequency

First, we look at the case when the frequency of the chip does not change. The result is shown in Figure 16. In Figure 16(a), we repeat the frequency-leakage scatter plot of Figure 10(a), this time at usual T and load conditions. As a result, the leakage power is significantly lower than in the worst case presented in Figure 10(a). Then, Figures 16(b)-(e) show the result of applying S-FGBB or D-FGBB with different numbers of cells, to reduce leakage at constant frequency.

Comparing Chart (a) to (b)-(c), we see that, if we apply S-FGBB, the chips move to the left, therefore saving leakage. Moreover, Charts (d)-(e) show that D-FGBB reduces the leakage of the chips even further. The higher the number of cells per chip is, the higher the leakage reduction is.

Figure 19 extends these experiments to all the BB environments. Figure 19(a) shows the average leakage power of the chips normalized to NoBB. The figure shows that both S-FGBB and D-FGBB save substantial leakage, especially as the number of cells per chip increases. However, D-FGBB is much more effective. D-FGBB reduces the leakage by 10–51% compared to S-FGBB, and by 12–69% compared to NoBB. Even with only 16 cells per chip, D-FGBB saves substantial leakage.

Figure 19(b) shows the total power consumption for the different FGBB schemes, DVS, and D-FGBB+DVS. The savings induced by D-FGBB are still large. Specifically, D-FGBB reduces the total power consumption by 6–19% relative to S-FGBB. When combined with DVS, D-FGBB+DVS reduces total power consumption by 15–36% compared to S-FGBB. DVS alone is not very effective.

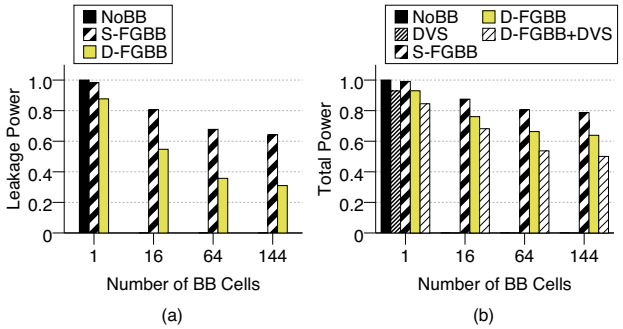

Figure 19. Leakage (a) and total power (b) of the chips for different FGBB schemes at constant frequency.

7.4.2. Dynamic Voltage and Frequency Scaling (DVFS)

Since many processors today use DVFS to save power, we would like to examine how the effectiveness of D-FGBB changes as

V_{dd} decreases with DVFS. For that, we take each chip in the batch and, for a set of supply voltages V_{dd}^i ranging from 1V to 0.6V, determine the corresponding frequency F_i before BB. Then, we apply D-FGBB at F_i. Finally, we measure the leakage and total powers for each V_{dd}^i before and after applying D-FGBB. The results are shown in Figure 20, where all bars are normalized to NoBB with V_{dd}=1V.

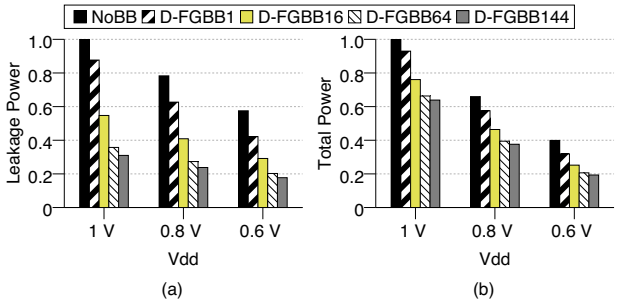

Figure 20. Leakage (a) and total power (b) at different voltage-frequency pairs, without and with D-FGBB.

Figure 20(a) shows that D-FGBB retains its relative effectiveness at reducing leakage as V_{dd} decreases from 1V to 0.6V — for all numbers of cells. Naturally, the absolute reduction decreases as V_{dd} decreases because there is less leakage to start with. Figure 20(b) shows that the total power savings are smaller but still very significant.

On the other hand, if we use S-FGBB, the BB levels are fixed at manufacturing time and cannot change with different voltages. When the same experiment is attempted with S-FGBB, we observe that the BB levels set at V_{dd}=1 are such that, as the voltage decreases, the processor cannot meet timing at the lower frequencies. Consequently, S-FGBB and DVFS cannot be easily combined.

7.5. Estimated Area Overhead of D-FGBB

To estimate the area overhead of D-FGBB, we use published data on BB support in real chips. Specifically, we use the area overhead reported in [28, 45] and scale it down to 45nm. We consider two implementations: one that uses critical path replicas and one that uses actual critical paths. Figure 21 shows the overhead as a fraction of the chip area. We see that the overhead with replicas varies between $<$2% and 4%, increasing with the number of BB cells. If actual critical paths are used rather than replicas, the overhead decreases to $\approx$3% for 144 cells.

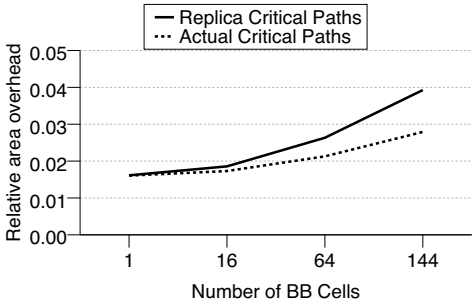

Figure 21. Area overhead of D-FGBB as a fraction of the chip area.

8. Other Related Work

While the problem of process variation has long been known to the VLSI community, Borkar *et al*. [2] are one of the first to provide a microarchitectural perspective. Other key contributors are Bowman *et al*. [3], who provided a model to estimate chip frequency in the presence of WID process variation.

Substantial effort has been devoted to modeling parameter variation [38]. While many of the models are analytical, some have been obtained through actual measurements of test chips (e.g., [12, 30, 39]). An important issue has been how to model the spatial correlation of systematic variation. While we use a multivariate normal distribution with a spherical spatial correlation structure, another approach is to use a quad-tree [25]. With that approach, however, it may be difficult to control aspects of the correlation structure.

There is abundant work on BB. Section 2.2 has outlined some of the main issues. In addition, Kumar *et al*. [23] pointed out the importance of BB adaptation to *T* changes. However, they rely on a static method, based on a mathematical model, to find the optimal BB voltages at manufacturing time, for all possible values of $V_{\rm th}$ and *T* that a circuit can have. In the presence of variation and given the scale of today's processors, this is a daunting task. Finally, Martin *et al*. [26] and Chen and Naffziger [6] examined the combination of BB and DVS.

Several researchers have proposed microarchitectural techniques to mitigate or tolerate parameter variation. They target register file and execute units [25], data caches [31], pipeline balancing [44], or intelligent floorplaning [16]. These techniques may be able to use D-FGBB to increase their effectiveness.

9. Conclusions

Parameter variation is a major challenge for processor designers. To address this challenge, we will likely need a combination of solutions at different layers, such as lithography, layout, circuits, and microarchitecture. The main contribution of this paper has been to introduce and evaluate a novel solution to this challenge that has a microarchitecture component, namely D-FGBB.

Our results showed that D-FGBB is very versatile and effective. We outlined three uses of D-FGBB: (i) reducing the leakage power at constant frequency in normal processor operation, (ii) increasing the processor frequency in a high-performance mode, and (iii) reducing the leakage power at constant frequency in a low power mode.

In its first use, D-FGBB reduces the leakage power of the chip by an average of 28–42% compared to S-FGBB. The higher savings correspond to the cases with more BB cells per chip. If, in addition, we combine D-FGBB with DVS, we save both leakage and dynamic power. In the high-performance mode, D-FGBB increases the processor frequency by an average of 7–9% compared to S-FGBB and by 7–16% compared to no BB. Finally, in the low-power mode, D-FGBB reduces the leakage power of the chip by an average of 10–51% compared to S-FGBB and by 12–69% compared to no BB.

We also show that D-FGBB can be synergistically combined with DVFS. While DVFS mostly controls dynamic power, D-FGBB controls leakage power. Overall, like DVFS, D-FGBB is a versatile control hook that can be managed in hardware or in software, and that can be used at different time and area granularities.

Acknowledgements

The authors would like to thank the anonymous reviewers for valuable comments and the members of the I-ACOMA group at the University of Illinois for their feedback. Special thanks go to Brian Greskamp, Smruti Sarangi, and Deming Chen for helpful discussions.

References

[1] N. Azizi and F. Najm. Compensation for within-die variations in dynamic logic by using body-bias. In *NEWCAS*, June 2005.

[2] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *Design Automation Conference*, June 2003.

[3] K. A. Bowman, S. G. Duvall, and J. D. Meindl. Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. *Journal of Solid-State Circuits*, February 2002.

[4] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *International Symposium on Computer Architecture*, June 2000.

[5] T. Chen and J. Gregg. A low cost individual-well adaptive body bias (IWABB) scheme for leakage power reduction and performance enhancement in the presence of intra-die variations. In *Design, Automation and Test in Europe*, February 2004.

[6] T. Chen and S. Naffziger. Comparison of adaptive body bias (ABB) and adaptive supply voltage (ASV) for improving delay and leakage under the presence of process variation. *Transactions on VLSI Systems*, October 2003.

[7] L. Clark, E. Hoffman, J. Miller, M. Biyani, Y. Liao, S. Strazdus, M. Morrow, K. Velarde, and M. Yarch. An embedded 32-b microprocessor core for low-power and high-performance applications. *Journal of Solid-State Circuits*, November 2001.

[8] N. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, 1993.

[9] R. Datta, A. Sebastine, A. Raghunathan, and J. A. Abraham. On-chip delay measurement for silicon debug. In *Great Lakes Symposium on VLSI*, April 2004.

[10] D. Ditzel. Power reduction using LongRun2 in Transmeta's Efficeon processor. In *Spring Processor Forum*, May 2006.

[11] D. Ernst, N. S. 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 *International Symposium on Microarchitecture*, December 2003.

[12] P. Friedberg, Y. Cao, J. Cain, R. Wang, J. Rabaey, and C. Spanos. Modeling within-die spatial correlation effects for process-design co-optimization. In *International Symposium on Quality Electronic Design*, March 2005.

[13] A. Ghosh, R. Rao, R. Brown, and C. Chuang. On-chip process variation detection and compensation for parametric yield enhancement in sub-100nm CMOS technology. IBM Austin Center for Advanced Studies, 2007.

[14] M. Hatzilambrou, A. Neureuther, and C. Spanos. Ring oscillator sensitivity to spatial process variation. In *First International Workshop on Statistical Metrology*, June 1996.

[15] Z. Huang and M. Ercegovac. Effect of wire delay on the design of prefix adders in deep-submicron technology. In *Asilomar Conference on Signals, Systems, and Computers*, October 2000.

[16] E. Humenay, D. Tarjan, and K. Skadron. The impact of systematic process variations on symmetrical performance in chip multi-processors. In *Design, Automation and Test in Europe*, April 2007.

[17] International Technology Roadmap for Semiconductors (1999).

[18] International Technology Roadmap for Semiconductors (2006 Update).

[19] A. Kahng. The road ahead: Variability. *Design & Test of Computers*, May-June 2002.

[20] A. Kahng. How much variability can designers tolerate? *Design & Test of Computers*, November-December 2003.

[21] T. Karnik, S. Borkar, and V. De. Probabilistic and variation-tolerant design: Key to continued Moore's Law. In *Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, February 2004.

[22] S. Krishnamurthy, S. Paul, and S. Bhunia. Adaptation to temperature-induced delay variations in logic circuits using low-overhead online delay calibration. In *International Symposium on Quality Electronic Design*, March 2007.

[23] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar. Mathematically assisted adaptive body bias (ABB) for temperature compensation in gigascale LSI systems. In *Asia South Pacific Design Automation Conference*, January 2006.

[24] T. Kuroda and T. Sakurai. Body biasing. In S. Narendra and A. Chandrakasan, editors, *Leakage in Nanometer CMOS Technologies*. Springer US, 2006.

[25] X. Liang and D. Brooks. Mitigating the impact of process variations on processor register files and execution units. In *International Symposium on Microarchitecture*, December 2006.

[26] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads. In *International Conference on Computer Aided Design*, November 2002.

[27] R. McGowen, C. A. Poirier, C. Bostak, J. Ignowski, M. Millican, W. H. Parks, and S. Naffziger. Power and temperature control on a 90-nm Itanium family processor. *Journal of Solid-State Circuits*, January 2006.

[28] S. Narendra, M. Haycock, V. Govindarajulu, V. Erraguntla, H. Wilson, S. Vangal, A. Pangal, E. Seligman, R. Nair, A. Keshavarzi, B. Bloechel, G. Dermer, R. Mooney, N. Borkar, S. Borkar, and V. De. 1.1V 1GHz communications router with on-chip body bias in 150 nm CMOS. In *International Solid-State Circuits Conference*, February 2002.

[29] G. Ono and M. Miyazaki. Threshold-voltage balance for minimum supply operation. In *International Solid-State Circuits Conference*, February 2003.

[30] M. Orshansky, L. Milor, and C. Hu. Characterization of spatial intrafield gate CD variability, its impact on circuit performance, and spatial mask-level correction. *Transactions on Semiconductor Manufacturing*, February 2004.

[31] S. Ozdemir, D. Sinha, G. Memik, J. Adams, and H. Zhou. Yield-aware cache architectures. In *International Symposium on Microarchitecture*, December 2006.

[32] A. Papoulis. *Probability, Random Variables and Stochastic Process*. McGrawHill, 2002.

[33] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2006. http://www.R-project.org.

[34] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L. Ceze, K. Strauss, S. Sarangi, P. Sack, and P. Montesinos. SESC Simulator, January 2005. http://sesc.sourceforge.net.

[35] P. J. Ribeiro and P. J. Diggle. geoR: a package for geostatistical analysis. *R-NEWS*, 2001.

[36] T. Sakurai and R. Newton. Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. *Journal of Solid-State Circuits*, April 1990.

[37] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *International Symposium on Computer Architecture*, June 2003.

[38] A. Srivastava, D. Sylvester, and D. Blaauw. *Statistical Analysis and Optimization for VLSI: Timing and Power*. Springer, 2005.

[39] B. Stine, D. Boning, and J. Chung. Analysis and decomposition of spatial variation in integrated circuit processes and devices. *Transactions on Semiconductor Manufacturing*, February 1997.

[40] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif. Full chip leakage estimation considering power supply and temperature variations. In *International Symposium on Low Power Electronics and Design*, August 2003.

[41] D. Tarjan, S. Thoziyoor, and N. P. Jouppi. CACTI 4.0. Technical Report HPL-2006-86, HP Labs, 2006.

[42] Y. Taur and T. H. Ning. *Fundamentals of Modern VLSI Devices*. Cambridge University Press, 1998.

[43] R. Teodorescu, B. Greskamp, J. Nakano, S. R. Sarangi, A. Tiwari, and J. Torrellas. VARIUS: A model of parameter variation and resulting timing errors for microarchitects. In *Workshop on Architectural Support for Gigascale Integration*, June 2007.

[44] A. Tiwari, S. R. Sarangi, and J. Torrellas. ReCycle: Pipeline adaptation to tolerate process variation. In *International Symposium on Computer Architecture*, June 2007.

[45] J. Tschanz, J. Kao, S. Narendra, R. Nair, D. Antoniadis, A. Chandrakasan, and V. De. Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. *Journal of Solid-State Circuits*, February 2002.

[46] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar. An 80-tile 1.28TFLOPS network-on-chip in 65nm CMOS. In *International Solid-State Circuits Conference*, 2007.

[47] J. Xiong, V. Zolotov, and L. He. Robust extraction of spatial correlation. In *Internation Symposium on Physical Design*, April 2006.

[48] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan. HotLeakage: A temperature-aware model of subthreshold and gate leakage for architects. Technical Report CS-2003-05, University of Virginia, March 2003.