Evaluation of Voltage Interpolation to Address Process Variations

Kevin Brownell, Gu-Yeon Wei, David Brooks
School of Engineering and Applied Sciences Harvard University
Cambridge, MA 01238
Email: {brownell, guyeon, dbrooks}@eecs.harvard.edu

Abstract— **Post-fabrication tuning provides a promising design approach to mitigate the performance and power overheads of process variation in advanced fabrication technologies. This paper explores design considerations and VLSI-CAD support for a recently proposed post-fabrication tuning knob called *voltage interpolation*. The paper discusses design tradeoffs between circuit tuning range and static power overheads that can be performed within the synthesis flow of the design process. The paper explores the scheme for a 64-core chip-multiprocessor machine using industrial-grade design blocks and shows that the scheme can be used to mitigate overhead arising from random and correlated within-die process variations. The analysis shows that the scheme can match the nominal delay target with a 10% power cost, or for the same power budget, incur only a 9% delay overhead after variations.**

I. Introduction

Advances in CMOS device technology have been a major driving force in the computing industry by providing smaller and faster transistors leading to tremendous gains in system integration and performance. However, in the most advanced fabrication technologies (65nm and beyond), difficulties in the manufacturing process manifest as potentially large variations in the device features of fabricated transistors. Worsening variations in semiconductor process technology will greatly impact the power and performance of future microprocessors and complex digital systems. Design level solutions to address process variation will be increasingly important, and to be cost-effective, such solutions must not drastically alter existing design flows.

Process variations occur at multiple spatial scales. Variations at the chip-level lead to performance differences between individual microprocessor dies, but increasingly, process variations are becoming more fine-grained. Uneven mask exposure due to lithography limitations leads to correlated (systematic) variations in transistor gate length and width at the granularity of units within cores. Random dopant fluctuations can change the threshold voltage of individual devices. Unlike die-to-die (D2D) variations, within-die (WID) correlated and random variations cannot easily be solved by traditional speed-binning techniques, because a handful of slow transistors can potentially lead to slow-speed paths that affect overall processor clock frequency.

Post-fabrication tuning through body bias is a potential design solution that seeks to tune the performance of individual blocks within a chip to smooth out the impact of variation [1], [2]. However, due to the lack of body control in SOI and future triple-gate devices, new post-fabrication tuning knobs have recently been proposed. This paper explores *voltage interpolation*, one such tuning knob [3] Voltage interpolation offers localized and fine-grained voltage tuning with two global supply voltages (VDDH and VDDL). The post-fabrication setting of the supply voltages enables local logic blocks to operate off of different effective voltages, interpolated between VDDH and VDDL. In turn, the delay of each block can be tuned with effective voltages to overcome the effects of delay variability.

Although voltage interpolation (VI) has been validated with a prototype chip implementing a simple floating-point adder [3], the technique has not been fully explored in the context of a standard VLSI-CAD design flow. This paper explores many design issues related to VI:

- The paper considers tradeoffs in the synthesis flow for VI, by investigating design choices related to the number and distribution of voltage domains. These choices impact the delay tuning range of the technique and static power overheads at domain boundaries.
- The paper investigates the potential for VI to compensate for random and systematic (correlated) variations, by demonstrating better performance and power for a given design yield.
- The paper investigates the concept for several logic blocks within the Sun UltraSPARC T1 core, using a standard flip-flop based design style [4]. The variation analysis is performed within the context of a 64-core chip-multiprocessor (CMP).

The paper provides a comprehensive and detailed analysis of voltage interpolation, demonstrating that the technique can compensate for variations in a power efficient manner. In the following section, we discuss background information and related work. Section III describes the basic concept of voltage interpolation. Section IV describes our architecture and circuit simulation platform. Section V presents simulation results and analysis. Lastly, this work is summarized in Section VI.

II. Related Work

Several techniques have recently been proposed to tune power/performance requirements post-fabrication. These techniques often make decisions about optimizing the design based on statistical analysis of the design features, in contrast to design-time-only optimization techniques that make the best decision to improve the expected yield [5]–[9]. Post-fabrication tuning techniques can loosely be grouped into three categories: those targeting supply voltage, transistor body bias, and clock frequency.

Global voltage and frequency scaling can address variability by adaptively tuning the supply voltage for a given frequency target or by fine-grained frequency adjustment. A major drawback of the technique is that the approach must be applied at the level of chips or cores (for CMPs). Fine-grained voltage scaling with a large number of voltage domains incurs the high cost of supplying multiple voltages and the voltage regulators that generate the voltages.

In contrast, adaptive body biasing (ABB) provides a fine-grained method to control threshold voltages and, therefore, leakage and performance [2]. Recently, individual well-adaptive body biasing schemes of locally generated body biases have been proposed providing fine-grained control [1]. The decision of how to group the devices is made at design time, but the tuning decision occurs after fabrication during test. For example, Kulkarni, et al. propose a variability-aware method that clusters gates at design time into a handful of groups [10]. Body biasing is then decided per group.

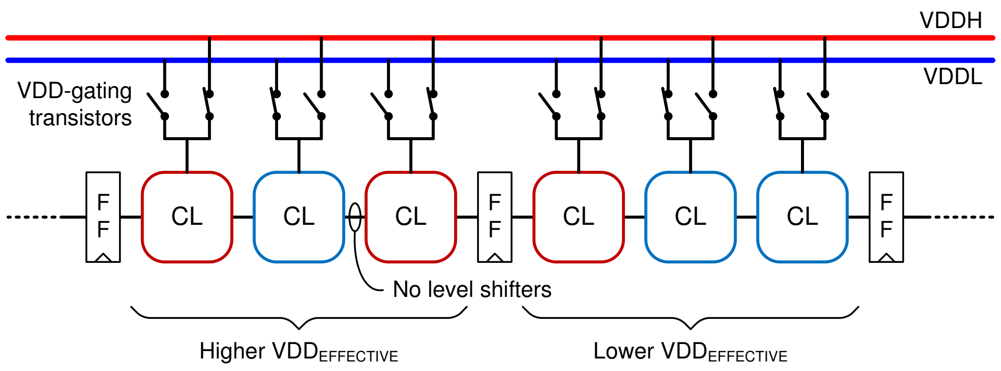


Fig. 1. Overview block diagram of voltage interpolation.

The results show significant improvement in reducing leakage power when compared to a fixed design-time based dual threshold voltage assignment method. Mani, et al. recently proposed an optimization framework that combines design-time decisions (gate sizing) with tuning decisions (ABB) [11]. The problem is formulated as an adjustable optimization problem where the decision-maker has a chance to update the optimization strategy upon learning additional information. Other researchers have considered body-bias placement based on microarchitectural blocks [12]. Unfortunately, ABB cannot be applied to technologies that lack the ability to control the body bias such as SOI and triple-gate CMOS. Thus, designers will need to explore additional tuning knobs.

Clock frequency provides another adjustable tuning knob. Tsai, et al. propose a flow that uses statistical timing analysis to synthesize a post-silicon-tunable clock tree that can combat timing uncertainty and yield degradation [13]. Other system-level design solutions have also been considered, mainly GALS (globally asynchronous locally synchronous) [14]. This technique however can only address some of the variations that occur at regional-to-global and slow-to-static spatial and temporal scales.

The basic concept of utilizing two supply voltages has been considered in the past to reduce power consumption. Usami and Horowitz describe a design-time methodology called clustered voltage scale that connects logic gates to one of two supply voltages in order to reduce power while minimizing performance degradation [15]. Agarwal and Nowka suggest a voltage-selection technique that enables power reduction in a simple adder [16]. All of these works require level shifters when crossing different voltage boundaries, which introduce delay and power overheads. The voltage interpolation technique discussed in this paper obviates level shifters and avoids the related overheads. In addition to employing multiple voltages, there is a large body of work that propose design-time selection of devices with high and low threshold voltages to selectively speed up or slow down different circuit paths. Again, such design-time tuning cannot efficiently deal with increasing localized random fluctuations of device parameters set by the fabrication process and time-varying aging effects.

III. Voltage Interpolation

Voltage interpolation (VI) offers a fine grained, voltage tuning technique to combat the effect of process variations. A typical design must accommodate the slowest paths in the design, which may be much slower than expected due to process variations. In accommodating these slow paths, the supply voltage may need to be raised, or the frequency may need to be lowered. While this voltage or frequency shift is necessary for the proper operation of the design, a large portion of the design is now using a much higher voltage, or much lower frequency, than would otherwise be necessary. Voltage interpolation seeks to address this problem by providing two supply voltages to a design; VDDH, a higher voltage, and VDDL, a lower voltage.

Figure 1 illustrates the basic concept of voltage interpolation. Each stage of logic is split into several substages, with each substage being able to choose between VDDH or VDDL through power gating pMOS devices. By choosing a supply voltage appropriate to each particular block of logic, the entire pipe stage can be viewed as operating off of a single "effective" voltage. In the single voltage supply case, all logic blocks must run off of the voltage required by the worst case path, whereas with voltage interpolation, each logic block uses only the "effective" voltage necessary to meet timing, depending on the process variations experienced by that block during fabrication. Logic blocks severely slowed down by process variations may need to choose VDDH as their supply voltage, whereas faster blocks may achieve power savings, yet still meet timing, by using a mix of VDDH and VDDL. Very fast blocks can save the maximum amount of power by running only off of VDDL.

An important design parameter for voltage interpolation is how large a voltage difference between VDDH and VDDL should be used. While a larger voltage difference allows for a larger tuning range, there is a static power penalty at the boundary between a VDDL and a VDDH stage. While level shifters could be used to alleviate this problem, they introduce unacceptable overheads of their own in terms of both delay and power. This static power penalty will be elaborated upon and quantified in Section V.

Another important design parameter for voltage interpolation is how many cuts each pipe stage should be split into. For N cuts, there are 2^N possible effective voltages achievable. An increase in the number of cuts provides an increase in the tuning resolution of a pipe stage, allowing for finer tuning of a particular block of logic. However, an increase in the number of cuts also results in an increase in the number of possible VDDL/VDDH substage boundaries, thus exasperating the static power problem. Additionally, a larger number of cuts suffers from an increase in the area overhead due to the power MUXes and routing of control signals.

A third consideration is the balance of the cuts. If stages are split in order to balance delay between substages, they may be severely imbalanced in terms of power. Likewise, if stages are split to balance power, delay will have to be imbalanced. Moreover, perfectly-balanced delays between substages may not yield the best result.

The goal of this paper is to explore tradeoffs in these design considerations for a set of highly-optimized design blocks. These tradeoffs will depend on the system architecture and amount and

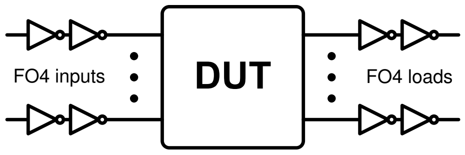


Fig. 2. HSPICE simulation setup to evaluate impact of random variations on standard cells.

type of variation experienced by the system. The paper addresses the VLSI-CAD support that is necessary to exploit voltage interpolation within a standard synthesis flow.

IV. Simulation Framework

Our simulation framework allows us to explore voltage interpolation in the context of a standard synthesis flow. In order to quantify the impact of random process variations, exhaustive HSPICE Monte Carlo simulations were preformed on each cell in the Faraday standard cell library for the UMC 130nm process [17]. Effective gate length, oxide thickness and threshold voltage (Vt) were varied. Sigma over mean numbers were chosen to be representative of modern 65nm technologies, and were chosen to be 3%, 3%, and 8% respectively.

The decision to perform our analysis using a commercial 130nm technology is motivated by our eventual plan to fabricate a test chip using VI and validate this simulation-based study. However, since more aggressive technologies will exhibit greater amounts of process variations, we should evaluate the effectiveness of VI for modern chips and, therefore, choose variations appropriate for 65nm technologies. Additional simulations show that for a given tuning range, the static power seen using 65nm technology models is actually less than that for the 130nm technology node. This is because the slight decrease in the threshold voltage from 130nm to 65nm allows for a lower difference between VDDH and VDDL to achieve the same tuning range. This lower voltage difference in turn reduces the static power penalty. Hence, post-fabrication tuning with VI ought to readily scale to more advanced technology nodes.

We characterized our standard cell library using the simulation setup shown in Figure 2. All inputs are shaped to resemble transitions of fanout-out-of-4 (FO4) inverters and all outputs see FO4 loading. Two hundred and fifty Monte Carlo simulations were performed for each set of inputs and input transitions that caused a transition on the output of each cell, across a range of supply voltages (0.9V, 1.05V, and 1.2V). Assuming each input combination is equally likely, the propagation delays resulting from each input combination were averaged. Then, the two hundred and fifty different average propagation delays resulting from different values for gate length, oxide thickness, and Vt were processed to obtain a standard deviation (σ) and mean (μ) values for the delay of each cell, representing the spread of delays for each gate caused by random process variations. Average energy numbers were also obtained for each cell.

Figure 3 illustrates the multi-level quad tree approach used to model the effects of correlated within-die variations [18]. To model the differing within-die correlated variations for each individual chip, each square in the multiple levels exhibits a shift in the mean delay of all circuits. Lower-level squares represent localized spatial correlations while higher-level squares represent larger-scale spatial correlations. One drawback of this simple approach is the sharp boundaries between squares. The standard deviation of the total delay variation seen across all four levels of the quad tree was set roughly to be 8.33% and equally distributed across the levels.

The voltage interpolation concept relies on cutting combinational logic blocks into several pieces. Our approach is based on the Synopsys Design Compiler (DC) pipelining package. This surrogate cutting algorithm enables us to rapidly evaluate different cut strategies and better understand different constraints and limitations for an eventual VI cutting algorithm. While the tool is able to balance delays reasonably well, it cannot balance the power consumed in each of the substages. The DC pipeline package is used to pipeline a block of combinational logic, with the goal of minimizing clock speed for a given number of pipe stages and, thus, balancing the delay of each stage. Since actual re-pipelining of the designs was not our goal, we set the tool to ignore any delays due to flip flop clock-to-Q and setup time when pipelining. This results in Design Compiler attempting to split the combinational logic into stages, with the goal to balance the delay of each stage, only considering combinational logic. After running through the pipelining tool to obtain cut points, flip flops added by the tool are removed. By instructing the tool to avoid adding or removing any cells, the resulting design, with delay-balanced substages, exactly matches the original design. This basic flow enables us to rapidly get realistic cut points to throughly investigate tradeoffs with respect to implementing voltage interpolation.

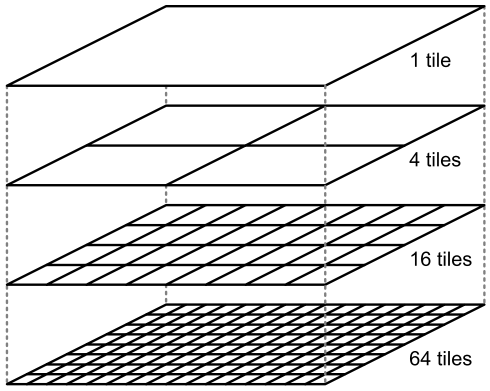


Fig. 3. Illustration of multi-level quad tree model for correlated within-die variations.

V. Analysis

Based on the simulation framework described above, this section investigates the potential drawbacks and benefits of applying VI to multiple blocks in the UltraSPARC T1. We first consider the static power issue as VI does not implement level shifters between groups of combinational logic operating off of different voltages. With static power costs in place, this section then explores various strategies to divide up or cut a block of combinational logic for voltage interpolation. After verifying that three cuts offer the best tradeoff in terms of overall energy efficiency despite higher static power costs, we study the impact of random and correlated process variations in the context of individual process blocks and with respect to a 64-core chip multiprocessor. Simulation results show that VI not only improves energy efficiency, but improves performance yield compared to not using VI.

A. *Static power*

One of the primary concerns associated with VI is the potential for static power at the interface between a VDDL stage and a VDDH stage. Figure 4(inset) illustrates how a weak "1" at the output of an inverter powered off of VDDL causes the pMOS of the following inverter, which is powered off of VDDH, to not fully shut off. This small residual Vgs, corresponding to VDDH-VDDL (or ΔV) leads to

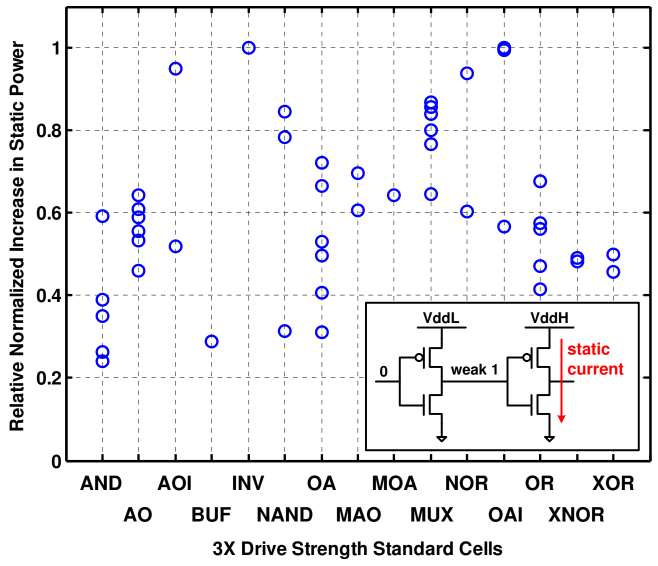


Fig. 4. Relative increase in static power, normalized to an inverter, for 3X drive strength foundry-provided standard cells in UMC 130nm CMOS process.

an increase in static power when the output of the inverter is driven to "0." In order to take this static power increase into account, exhaustive HSPICE simulations of every possible input combination, for each gate in the standard cell library, were preformed. Assuming each input combination is equally likely for any given cell, an average static power for each cell was obtained. For these simulations, we assume a worst-case ΔV of 300mV. Lower ΔV's lead to lower static power.

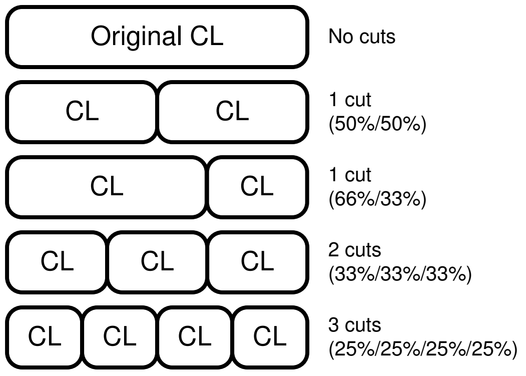
Figure 4 presents the relative increase in static power at a VDDL/VDDH boundary for a representative collection of 3X drive strength standard cells, normalized to the relative increase observed for an inverter. Each type of standard cell has one or more points, depending on the number of gate types with different numbers of inputs for that particular type of cell with 3X drive strength. For example, the library has 2-, 3-, and 4-input NAND gates, corresponding to three points in the plot. The inverter clearly suffers the highest relative static power increase, with most other cells suffering static power increase at much lower levels. These results suggest inverters ought to be avoided for the first stage at the boundaries between blocks operating off of different voltages. The impact of the static power penalty between VDDL and VDDH stages is carefully taken into account in our later analysis of VI.

B. Cutting Strategies for Voltage Interpolation

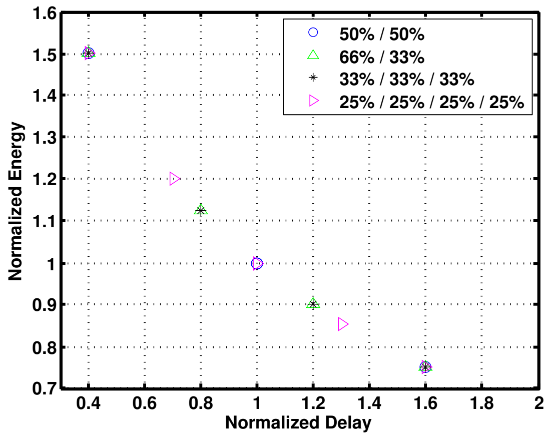
Voltage interpolation relies on a block of combinational logic to be cut into multiple subdivisions in order to offer fine-grained voltage tuning. There are multiple ways one can implement the cuts—depending on the number of cuts and division of delay between cuts. Figure 5(a) illustrates different cutting possibilities assuming an ideal block of combinational logic where all delay paths are perfectly balanced and consume the same amount of energy. Four scenarios are considered: one cut, delays split 50%/50%; one cut, split 66%/33%; two cuts, split 33%/33%/33%; and three cuts, split 25%/25%/25%/25%. Three out of the four scenarios implement the cuts with delays even distributed across each subdivision. One scenario (1 cut, 66%/33%) considers the case where the delay through the first group of logic is twice as long as the second group. Figure 5(b) plots the normalized energy versus normalized delay scatter plot as a result of implementing VI. The two extreme ends correspond to cases where all of the logic groups, regardless of the cut strategy, operate off of VDDH or VDDL. The one cut scenario with delays split 50%/50% yields three distinct delay vs. energy points while the 66%/33% delay split scenario yields four distinct points. The delay imbalance offers a richer set of possible delays since VDDH/VDDL and VDDL/VDDH configurations lead to different delays. For larger numbers of cuts, Figure 5(b) shows that the number of energy/delay points do not increase substantially. While, for example, two cuts has the potential to provide up to $2^3 = 8$ different effective voltages, many of the configurations overlap if the delays and energy consumption of each stage are perfectly balanced. With equal-delay cutting, two cuts provides only 5 effective voltages—either all four groups operate off of VDDH, three groups operate off of VDDH, two groups operate off of VDDH, one group operates off of VDDH, or no groups operate off of VDDH (all off of VDDL). This leads one to conclude that for an ideal block of logic, a more intelligent cutting strategy is needed to provide the full tuning resolution offered by voltage interpolation, and that moving to a larger number of cuts will always yield more effective voltage settings.

In contrast to the ideal block of logic considered above, real implementations exhibit a wide range of delay path imbalances. Figure 6 presents normalized energy vs. normalized delay scatter plots for three blocks of logic found in the UltraSPARC T1 RTL code. These three blocks were synthesized aggressively for the same frequency target using *Design Compiler* from Synopsys and the different cut scenarios were obtained using the built-in *pipeline_design* and *balance_registers* commands as described in Section IV. Despite implementing cut strategies that strive to meet balanced delay targets, a wider variety of effective voltages are available due to inherent imbalances. Since there is a discrete number of cells along each path, each with a different delay, it is impossible to split the paths into perfectly equal stages. This can be advantageous for voltage interpolation, as it can increase the number of effective voltages available. However, not every configuration is usable since some configurations exhibit higher energy and higher delay compared to others. The ALU, as depicted in Figure 6(a), exhibits two sets of energy vs. delay trends. Although the ALU is balanced relatively well in terms of delay, is is not balanced very well for power. The first subdivision of logic in the ALU consumes a disproportionate fraction of power, nearly 60% of the total, thus causing the observed shift in overall energy consumption whenever it switches between VDDL and VDDH. Hence, only a subset of the configurations that offer the best energy-delay tradeoff ought to be used. This power imbalance introduces a large energy cost to meet normalized delay targets below 0.95 in the ALU. The other two blocks, FADD2 and FADD3, do not suffer as much from power imbalances and exhibit smoother trends.

Figure 7 presents the same set of relationships as Figure 6, but includes the static power penalty when a VDDL stage precedes a VDDH stage, as discussed in Section V-A. While some points have shifted up slightly, the overall change in the achievable energy/delay points is relatively small. While Figures 6 and 7 show that more cuts yield a richer set of possible voltage settings, it is not clear which cut strategy would be best. While more cuts offer more voltage settings, static power penalties are higher. On the other hand, Figure 8(a) recreates a subset of the results in Figure 5(b) to illustrate the impact of having fewer voltage settings. For the 50%/50% scenario, there are three distinct configurations. If a normalized delay of 1.5 is the desired target, then the configuration corresponding to 1 and

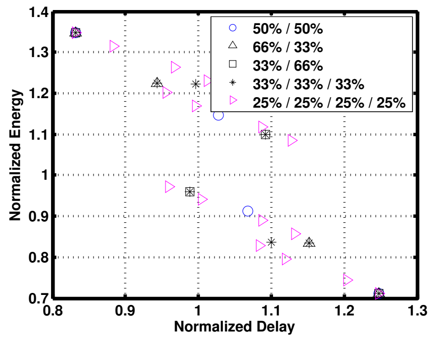


(a) Four combinational logic cutting scenarios

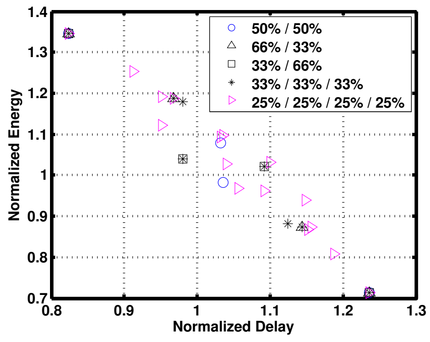


(b) Normalized energy vs. normalized delay

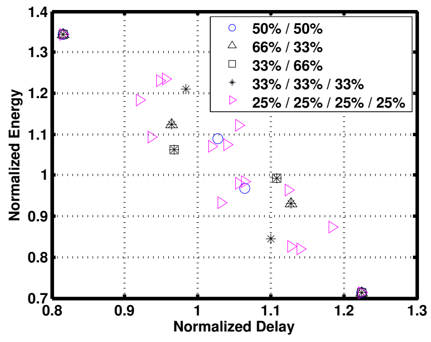
Fig. 5. Investigation of different cutting strategies for an ideal block of logic.



(a) ALU

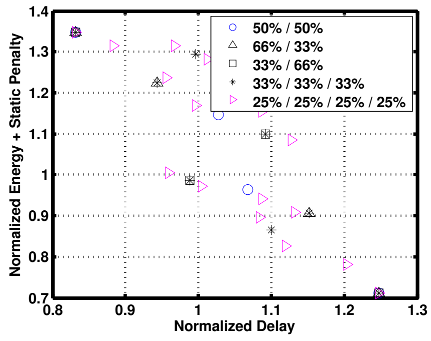


(b) FADD2

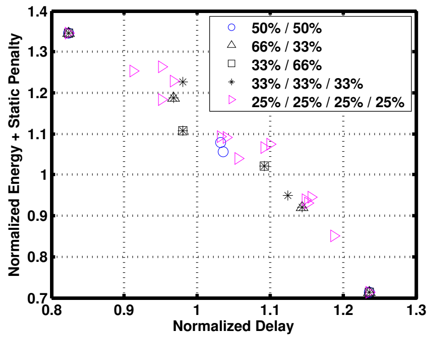


(c) FADD3

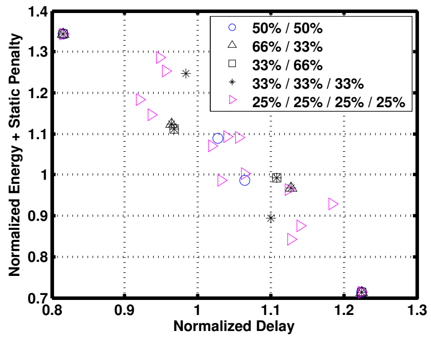
Fig. 6. Impact of different cutting strategies on the normalized energy vs. normalized delay for three blocks of combinational logic from UltraSPARC T1.



(a) ALU



(b) FADD2

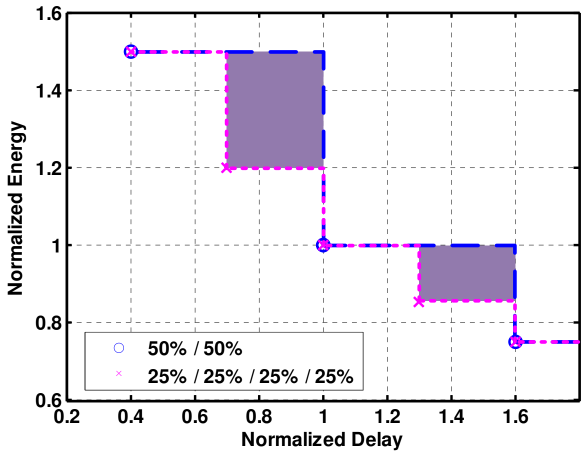


(c) FADD3

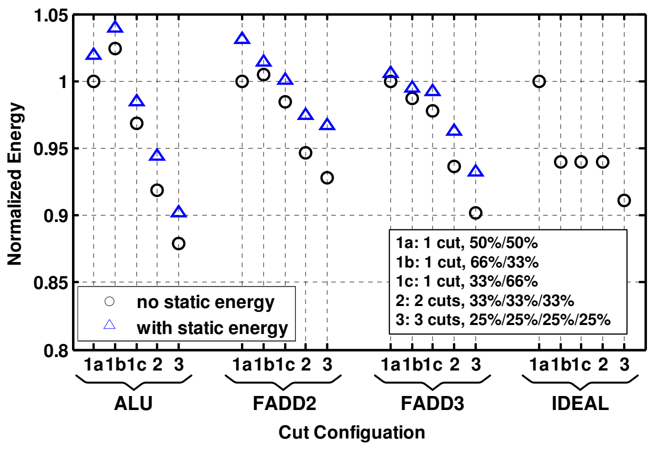
Fig. 7. Impact of static power penalties added to the energy vs. delay points in Figure 6.

its associated energy must be expended. On the other hand, for the 25%/25%/25%/25% cut scenario, the configuration corresponding to a delay of 1.3 can be used. The shaded region represents the relative energy cost associated with the 50%/50% scenario. A larger number of possible configurations and voltage settings offers more opportunities for the design to meet target delays while expending less energy.

Figure 8(b) quantifies the relationship between the different cutting strategies, before and after the static power penalty is considered. The y-axis represents the average energy of a circuit whose target delay is equally distributed between the all VDDH and all VDDL settings, normalized to the energy of the 1 cut, equally split case for each circuit. This average energy is calculated by taking the integral of the usable configurations, as depicted in Figure 8(a). Each of the three logic blocks are shown, in addition to the case for an ideal circuit that can be perfectly cut for delay and energy. Figure 8(b) shows that as one moves to increasing numbers of balanced cuts, the average energy falls, even when static power is taken into account. In the case of the ALU, three cuts achieves $\sim$12% less average energy than one cut both when considering and ignoring static energy, despite



(a) Quantized energy between delay targets.



(b) Normalized energy versus cut configurations with and without static energy penalties.

Fig. 8. Evaluating the impact of quantized energy settings and static energy penalties to find the best cutting strategy.

the per-stage power imbalance noted earlier. The FADD2, FADD3, and ideal blocks use 7%, 10%, and 9% less energy respectively by moving from one cut to three cuts. The unbalanced one cut strategies are sometimes better, and sometimes worse than the balanced one cut strategies, depending on the circuit. Since the balanced three cut strategy provides a lower average energy for each block than all of the other options, we use this cut strategy for the rest of our analyzes.

C. Impact of variations

With the cutting strategy in place, we now shift gears to investigate how VI can be used to combat the impact of random and correlated variations. In order to analyze the effects of variations with respect to a whole processor core, we assume one core contains one of each of the three circuit blocks (ALU, FADD2, and FADD3). The critical path of any one block limits the frequency achievable by the entire core. This is a fair consideration, as all three blocks were synthesized aggressively for the same frequency target, and all represent typical blocks which could set the frequency of a microprocessor. We carefully model every path and every instance delay in each block, taking into account factors such as loading, fast/slow inputs, and rising edge vs. falling edge when calculating the delay of each instance in each path. Since each block is synthesized assuming cells operating off of 1.2V, we can scale the delay of each cell with respect to HSPICE simulations of the cell at different voltages. By using the σ/μ delay numbers of each standard cell, gathered from Monte Carlo simulations, random variations can be applied to every cell for 1000 different cores (each containing an ALU, a FADD2, and a FADD3). For each core, the worst resulting critical path sets the frequency of that core.

To demonstrate VI's delay-tuning capabilities, Figure 9 compares histograms of the 1000 cores with random variations, with and without voltage interpolation. The x axis is normalized to the target delay (nominal delay) of the design without any variations. Without voltage interpolation (ΔV=0mV) and when only random variations are considered, the worst-case delay distribution of the cores is relatively tight. However, there is a shift in the mean and none of the cores can meet the nominal delay target. The three subsequent subplots show the resulting delay distributions for three different settings of ΔV. Each subplot contains histograms for of all 1000 cores and all 16 voltage configurations, corresponding to the 3-cut strategy. While a larger ΔV allows for wider tuning range, a core

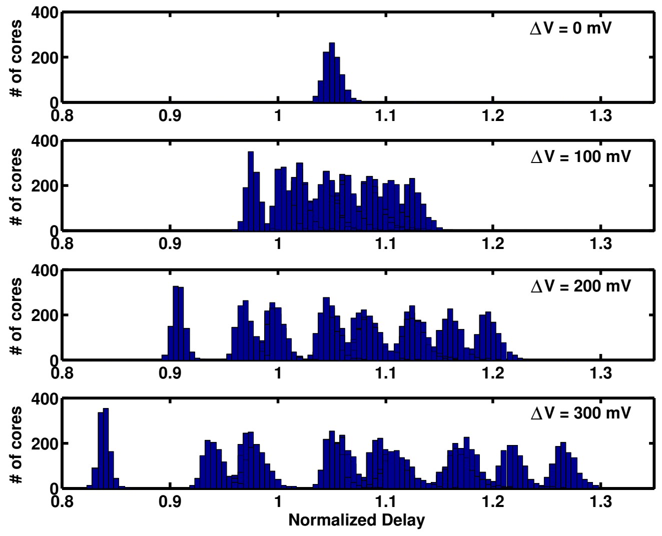


Fig. 9. Impact of random variations on normalized delay for 1000 cores without VI (ΔV=0mV) and with voltage interpolation.

only suffering random variations does not require a high tuning range, and can subsist with a ΔV of 100 mV. On the other hand, a large percentage of the cores require all logic groups to operate off of VDDH, thus being no different from simply raising the global supply voltage. In contrast, with ΔV = 300mV, more voltage configurations that use a mix of VDDH and VDDL stages can meet the nominal delay target, thus providing an advantage over simple voltage scaling. Moreover, larger tuning range is needed to also combat the effects of correlated variations in large chips.

While a small ΔV is sufficient for random variations, large chips also suffer from die-to-die and within-die variations. We use a multi-level quad-tree method to model correlated within-die variations as described earlier in Section IV. We assume the total correlated variation has roughly equal magnitude to cell-level random variations, applied evenly across four levels of the quad tree. Figure 10 presents the effect of adding in correlated variations to the 1000 cores considered in Figure 9, assuming the cores are a part of a much larger multi- or many-core CMP. Correlated variations significantly increase the spread of critical path delays in cores operating off of a

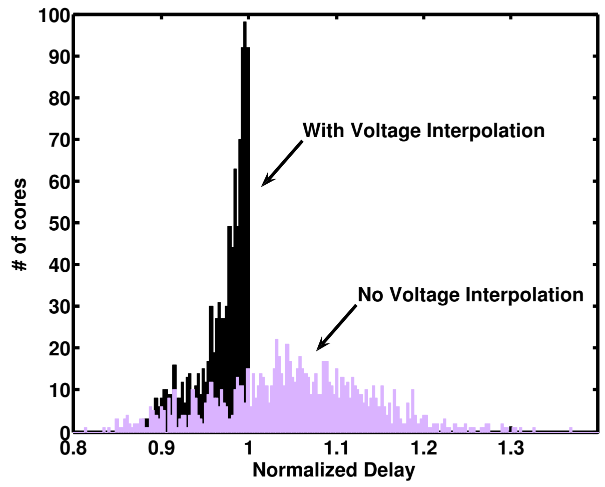


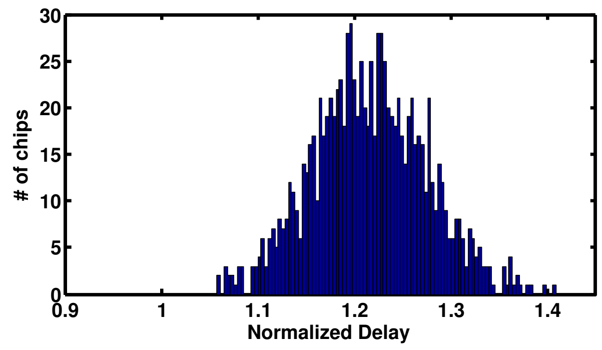
Fig. 10. Effectiveness of VI for 1000 cores with random and correlated within-die variations.

single nominal voltage (no VI, nominal VDD = 1.05V) and a large ΔV is required to provide adequate tuning range to compensate for both random and correlated variations. When VI is applied to this scenario with ΔV = 300mV (VDDH = 1.2V and VDDL = 0.9V), over 99% of the cores can be configured to meet the nominal delay target and often choose configurations with lower energy than merely using a single higher global voltage.

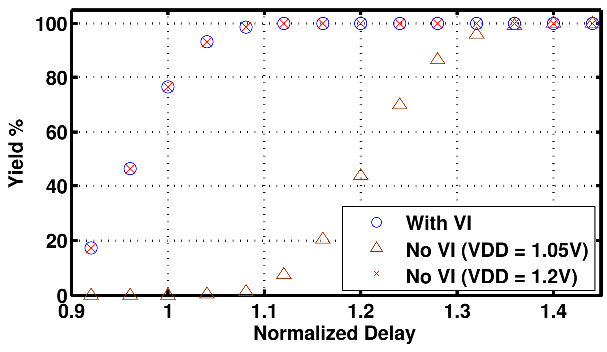
To examine the effects of within-die correlated variations on a full system, we consider a CMP chip with 64 of our previously analyzed cores, arranged in an 8X8 grid. We investigate 1000 chips, by randomly choosing cores with random variations from our previous analysis and applying the multi-level quad tree to model correlated variations. The worst-case core sets the frequency of the entire chip. Figure 11(a) shows the delay distribution of the single-voltage chips operating off of a nominal VDD = 1.05V, with the x-axis again normalized to a target delay without variations. With the addition of correlated variations, not a single chip using the nominal voltage can meet the original timing target. Limited by the worst-case core out of 64 per chip, the distribution tightens up and the mean shifts to longer delays when compared to the results in Figure 10, which considers the impact of random and correlated variations on a core-by-core basis.

We first investigate the benefits of VI (VDDH = 1.2V, VDDL = 0.9V) by comparing yields. Figure 11(b) plots the yield vs. delay for both the voltage interpolated and the single-voltage chips (at two voltage settings) for a range of target delays, normalized to the no variations target delay. At the original target delay, with both random and correlated within-die variations, 76.5% of the VI chips meet timing, whereas none of the single-voltage chips meet timing when operating off of the nominal voltage (VDD = 1.05V). Raising the single global VDD to 1.2V leads to yields equivalent to implementing VI with VDDH = 1.2V. In either case, this maximum voltage limit caps the yield at the nominal delay target. By relaxing the timing target by 8%, 98.7% of the VI chips can meet timing, whereas only 1.1% of the chips operating off of VDD = 1.05V meet timing. Further relaxing the original timing target by 12% improves yield to 100% for the VI chips, whereas only 7.2% of the 1.05V chips meet timing.

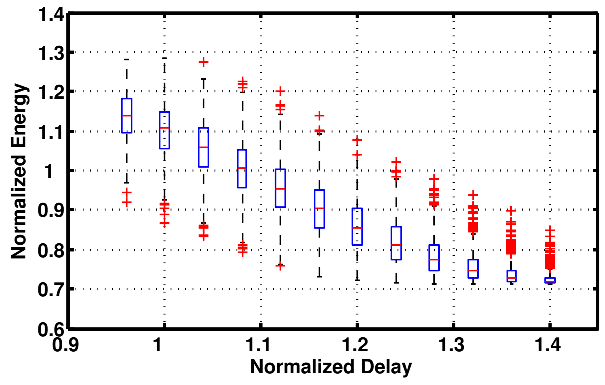
While results thus far have show that VI can improve timing, it is important to also consider energy. Otherwise, simply meeting timing with VI is no better than scaling the global voltage with respect to the worst-case critical path in each chip to meet timing. Figure 11(c)



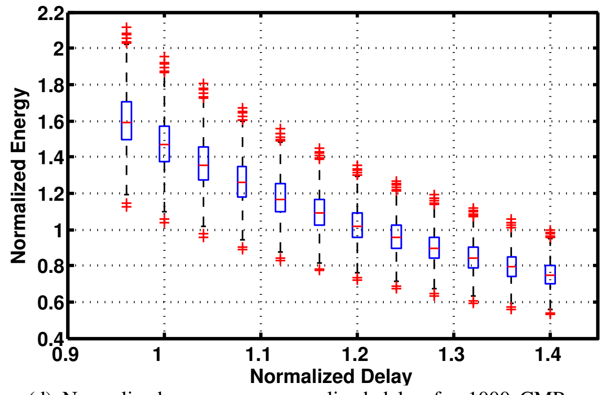
(a) Delay distribution of CMP chips operating off of a single fixed nominal VDD (without voltage interpolation)



(b) Performance yield with and without VI



(c) Normalized energy vs. normalized delay for 1000 CMPs with VI (VDDH = 1.2V, VDDL = 0.9V)



(d) Normalized energy vs. normalized delay for 1000 CMPs with global voltage scaling (Max. VDD = 1.2V)

Fig. 11. Evaluation of VI applied to 1000 CMP chips suffering random and correlated variations.

presents boxplots[1] for the 1000 CMP chips with VI (VDDH = 1.2V, VDDL = 0.9V) across a number of different chip frequency targets, but again presented in terms of delay. The x-axis is normalized to the nominal, no variations delay, and the y-axis is normalized to the nominal voltage chip energy (nominal VDD = 1.05V). At the nominal delay target, 76.5% of the VI chips can meet timing and require higher than nominal energy for most of the functional chips. The median chip requires 11% more energy compared to the nominal. If we relax the timing target to the mean of the delay distribution (22% slow down) in Figure 11(a), all but a couple of the VI chips meet timing, and the median chip consumes 15% less energy than the nominal chip energy. If the delay target is slowed down further to allow 99% of the nominal voltage chips to meet timing (33% slow down), all VI chips operate at lower than nominal energy, with the median chip consuming 25% less than nominal energy. The flattening out of the minimum energy for longer timing targets arises because VDDH and VDDL are fixed throughout the plot.

In contrast to using VI, Figure 11(d) presents boxplots for the same 1000 chips but using global voltage scaling. Across the range of delays shown for 1000 chips, voltage scales from a minimum of 0.77V to a maximum limited to 1.2V. This voltage limit again leads to yields equivalent to that of using VI above. At the nominal delay target, there is a >45% energy penalty for the median chip using chip-wide global voltage scaling, whereas the median chips with VI suffered an 11% energy penalty. Since the lower limit of voltage scaling is not constrained in this study, global voltage scaling offers energy savings as delay targets are relaxed. And yet, for the slowest delay target shown (40% slow down), the energy of the median chip with VI is still lower than the median chip with global voltage scaling. While global voltage scaling must choose a voltage that accommodates the worst-case delay path in the worst-case core, VI offers the effect of providing fine-grained voltages to each logic block in each core to maximize energy efficiency.

VI. Conclusion

This paper has explored a number of design issues related to voltage interpolation, which can combat the deleterious effects of process variations. Tradeoffs related to the number of stage cuts and the magnitude of ΔV were considered. The study centered on a number of blocks from an UltraSPARC T1 core, considering both random and correlated within-die process variations for a 130nm CMOS process with foundry-provided standard cells. We show that for these blocks, a higher number of cuts provides benefits that outweigh the static power cost associated with more cuts. Additionally, in the context of a 64-core CMP, a ΔV = 300 mV is required to cover the delay spread seen by random and correlated variations. We show that, by using voltage interpolation, the median chip can hit the original timing target with only a 10% increase in energy consumption, whereas the median single-voltage chip requires a 46% increase in energy. Additionally, the median VI chip can hit the original energy budget with only a 9% delay overhead, whereas the median single-voltage domain chip suffers a 22% delay overhead.

This evaluation of voltage interpolation motivates follow-on investigations to integrate it into a fully-automated design flow. While existing synthesis tools have built-in algorithms that can divide combinational logic into groups with nominally balanced delays, power balancing between groups is also important.

[1]Boxplots are graphical displays of data that measure location (median) and dispersion (interquartile range), identify possible outliers, and indicate the symmetry or skewness of the distribution.

While our analysis was preformed using a commercial 130nm technology and standard cell library, the effectiveness of VI should scale well with technology. As the threshold voltage decreases, albeit gradually, the voltage difference between VDDH and VDDL may be decreased to achieve a comparable tuning range. This decrease offsets the static power penalty increase one would expect to see in aggressively-scaled technologies.

In addition to efficient logic cutting algorithms for VI, future work entails development of efficient tools that can intelligently place standard cells of common voltage groupings, insert cells with power-gating transistors, and route configuration signals throughout a design. We envision such design automation will enable designers to fully leverage VI to address the delay and energy overheads introduced by process variations.

References

[1] J. Gregg and T. Chen, "Post silicon power/performance optimization in the presence of process variations using individual well-adaptive body biasing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, March 2007.

[2] J. W. Tschanz, J. T. Kao, and S. G. Narendra, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage," *IEEE Journal of Solid-State Circuits*, November 2002.

[3] X. Liang, D. Brooks, and G.-Y. Wei, "A process-variation-tolerant floating-point unit with voltage interpolation and variable latency," in *International Solid-State Circuits Conference*, Feb. 2008.

[4] Sun Microsystems Inc., "OpenSPARC T1 Chip Design," http://www.opensparc.net/.

[5] A. Davoodi and A. Srivastava, "Variability driven gate sizing for binning yield optimization," in *43rd Design Automation Conference*, June 2006.

[6] Y. Lu and V. D. Agrawal, "Statistical leakage and timing optimization for submicron process variation," in *20th International Conference on VLSI Design*, January 2007.

[7] Y. Xu, X. Li, K. Hsiung, S. Boyd, and I. Nausieda, "Opera: optimization with ellipsoidal uncertainty for robust analog ic design," in *42nd Design Automation Conference*, June 2005.

[8] A. Srivastava, T. Kachru, and D. Sylvester, "Low-power-design space exploration considering process variation using robust optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 1, pp. 67–79, 2007.

[9] D. Sinha, N. V. Shenoy, and H. Zhou, "Statistical timing yield optimization by gate sizing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 10, pp. 1140–1146, October 2006.

[10] S. Kulkarni, D. Sylvester, and D. Blaauw, "A statistical framework for post-silicon tuning through body bias clustering," in *IEEE/ACM International Conference on Computer Aided Design*, 2006.

[11] M. Mani, A. Singh, and M. Orshansky, "Joint design-time and post-silicon minimization of parametric yield loss using adjustable robust optimization," in *IEEE/ACM International Conference on Computer Aided Design*, November 2006.

[12] R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "Mitigating parameter variation with dynamic fine-grain body biasing," in *40th International Symposium on Microarchitecture (MICRO-40)*, Dec. 2007.

[13] J. Tsai, L. Zhang, and C. Chen, "Statistical timing analysis driven post-silicon-tunable clock-tree synthesis," in *International Conference on Computer Aided Design*, November 2005.

[14] D. Marculescu and E. Talpes, "Variability and energy awareness: a microarchitecture-level perspective," in *42nd Annual Conference on Design Automation*, June 2005.

[15] K. Usami and M. Horowitz, "Clustered voltage scaling technique for low-power design," in *International Workshop on Low Power Design*, April 1995.

[16] K. Agarwal and K. Nowka, "Dyanmic power management by combination of dual static supply voltages," in *International Symposium on Quality Electronic Design*, March 2007.

[17] Faraday Technology Corporation, "UMC 0.13um Logic core cell library," http://www.faraday-tech.com/.

[18] A. Agarwal, V. Zolotov, and D. Blaauw, "Statistical timing analysis using bounds and selective enumeration," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 9, September 2003.