

Mitigating the Effects of Process Variation in Ultra-low Voltage Chip Multiprocessors using Dual Supply Voltages and Half-Speed Stages*

Timothy N. Miller, Renji Thomas, Radu Teodorescu
Department of Computer Science and Engineering
The Ohio State University
{millerti, thomasr, teodores}@cse.ohio-state.edu

Abstract

Energy efficiency is a primary concern for microprocessor designers. One very effective approach to improving the energy efficiency is to lower chip supply voltage very near to the transistor threshold voltage. This reduces power consumption dramatically, improving energy efficiency by an order of magnitude. Low voltage operation, however, increases the effects of parameter variation resulting in significant frequency heterogeneity between (and within) otherwise identical cores. This heterogeneity severely limits the maximum frequency of the entire CMP. We present a combination of techniques aimed at reducing the effects of variation on the performance and energy efficiency of near-threshold, many-core CMPs. Dual Voltage Rail (DVR), mitigates core-to-core variation with a dual-rail power delivery system that allows post-manufacturing assignment of different supply voltages to individual cores. This speeds up slow cores by assigning them to a higher voltage and saves power on fast cores by assigning them to a lower voltage. Half-Speed Unit (HSU) mitigates within-core variation by halving the frequency of select functional blocks with the goal of boosting the frequency of individual cores, thus raising the frequency ceiling for the entire CMP. Together, these variation-reduction techniques result in almost 50% improvement in CMP performance for the same power consumption over a mix of workloads.

Keywords: Energy efficiency, chip multiprocessors, process variation, low voltage.

1 Introduction

Power consumption is one of the most significant roadblocks to future technology scaling according to a recent report by the International Technology Roadmap for Semiconductors (ITRS) [1]. Power delivery and heat removal capabilities [2] are already limiting performance in microprocessors today and will continue to severely restrict performance in the future [3]. If current integration trends continue, chips could see a 10-fold increase in power density by the time 11nm technology is in production. The only way to ensure continued scaling and performance growth is to develop solutions that dramatically increase computational energy efficiency.

A very effective approach to improving the energy effi-

ciency of a microprocessor is to lower its supply voltage (V_{dd}) to very close to the transistor's threshold voltage (V_{th}), into the so-called near-threshold (NT) region [4, 5, 6, 7]. This is significantly lower than what is used in standard dynamic voltage and frequency scaling (DVFS), resulting in aggressive reductions in power consumption (up to 100 \times) with about a 10 \times loss in maximum frequency. Even with the lower frequency, chips running in near-threshold often achieve significant improvements in energy efficiency. In a power-constrained CMP, near-threshold operation will allow more cores to be powered on (albeit at much lower frequency) than in a CMP at nominal V_{dd} . Despite lower individual core throughput, aggregate throughput can be much higher, especially for highly parallel workloads.

Unfortunately, near-threshold CMPs are very sensitive to *process variation*. Variation is caused by difficulties in the manufacturing process at very small feature technologies. One parameter most severely affected by variation is the transistor threshold voltage (V_{th}). Variation in V_{th} causes heterogeneity in transistor delay and power consumption within processor dies leading to sub-optimal performance. Near-threshold operation greatly exacerbates these effects because supply voltage is much closer to the threshold voltage, making the impact of V_{th} variation much more pronounced. For 32nm technology, variation at near-threshold voltages can easily increase by an order of magnitude or more compared to nominal voltage. Since processor frequency is determined by the slowest critical path, this level of variation severely limits the frequency of near-threshold chips.

This paper presents two simple, low-overhead, but highly effective techniques for mitigating frequency variation in near-threshold CMPs. These techniques improve the energy efficiency of CMPs allowing them to run at higher frequencies for the same power consumption. The first technique, Dual Voltage Rails (DVR), consists of a power supply system that provides the CMP with two power supply rails. Each power rail supplies a different, externally controlled voltage. Each core in the CMP can be assigned to either of the two power supplies using a simple power gating circuit [8]. We show that by calibrating the voltage difference between the two power rails and by carefully choosing the assignment of cores to each rail, post-manufacturing, frequency variation can be reduced from 30.6% standard deviation from the mean (σ/μ) down to 23.1%, improving CMP frequency by 30%.

*This work was supported in part by an allocation of computing time from the Ohio Supercomputer Center.

The second technique, Half-Speed Unit (HSU), mitigates within-core variation. Within-core variation increases the delay of some of the core’s critical paths, lowering the maximum frequency individual cores can achieve. Previous work has proposed techniques for reducing within-core variation in processors operating at nominal voltages including body biasing [9, 10, 11], variable pipeline latency [12, 13] and the GALS architecture [14]. Most previous solutions fine-tune the delay of pipeline stages to reduce delay variation and improve frequency. These designs incur significant overheads: multiple independent bias voltages (and wells) for body biasing, complex calibration and control for variable pipeline latency designs. The GALS (globally asynchronous, locally synchronous) architecture runs the main functional units on independent clocks (each at the fastest frequency it can achieve) improving the overall performance of the core in the presence of variation. The GALS design is complex to implement because it uses synchronization queues for inter-stage communication and requires independent clock signals that must be calibrated for each pipeline stage.

HSU uses a simpler design to mitigate within-core variation. With HSU, functional units have two possible speeds: full speed (running at the core’s frequency) and half speed (running at half the core’s frequency). Slower units run at half speed, allowing the core frequency to be increased substantially. Because slow units run at precisely half the speed of the fast ones, they can be easily synchronized with the rest of the core, albeit with increased latencies. For instance, access to a slow register file might take two cycles instead of one. Variation is unpredictable, which means we cannot know before manufacturing how many stages will need to be slowed down to reach the desired frequency. Depending on which (and how many) units are slowed down, the impact on core performance will range from minimal to significant.

Our evaluation shows DVR alone improves the performance of a variation-unaware CMP design at near-threshold by 30% and HSU alone by 33%. When combined, DVR and HSU together achieve a 48% average performance improvement.

Overall, this paper makes the following contributions:

- Analyzes the impact of process variation on large CMPs running at near-threshold voltages.
- Presents DVR, a simple and powerful solution for reducing core-to-core frequency variation in NT CMPs.
- Presents HSU, a low-overhead, low-complexity solution for mitigating within-core variation in NT CMPs.

2 Architecture Design

2.1 Dual Voltage Rails (DVR)

Within-die variation causes power consumption and maximum operating frequency to vary widely from core to core. This heterogeneity is an important because the CMP system clock is limited by the slowest core, which can severely limit CMP frequency. At the same time, any core that can run faster than the system clock is wasting energy. This is because these cores could run at a lower voltage for the same speed and therefore save power.

DVR addresses these inefficiencies providing two power supply rails in the CMP. Each power rail supplies a different voltage, both near-threshold, with one slightly higher than the other. Cores can be assigned, post-manufacturing, to either of the two supply voltages as follows: “fast” cores are assigned to run on the lower V_{dd} , reducing their power consumption, while “slow” cores run on the higher V_{dd} , improving their frequency. This reduces within-die frequency variation and therefore reduces wasted energy. At near-threshold even small changes in V_{dd} have a significant effect on frequency. Thus, even a small difference (100mV) in between the two rails dramatically reduces frequency variation.

DVR is low overhead and relatively easy to implement. Some existing designs [15, 16] already use multiple power rails to supply different voltages to different sections of the chip such as cores, caches or memory controller. These designs, however, have a single power rail for each section of the chip and assign all cores to the same power supply. With DVR, each core has two power gates [8], allowing it to be assigned to either power rail. In addition, two external voltage regulators are required to independently regulate supply for the two rails. Figure 1 shows an overview of a near-threshold CMP with the proposed DVR power delivery system. The only additional overhead DVR introduces in the power distribution network is a second power supply line to each core. Within each core, only a single power distribution network is needed, resulting in a much lower overhead compared to solutions that employ dual voltages at much finer granularity [13, 17, 18, 19].

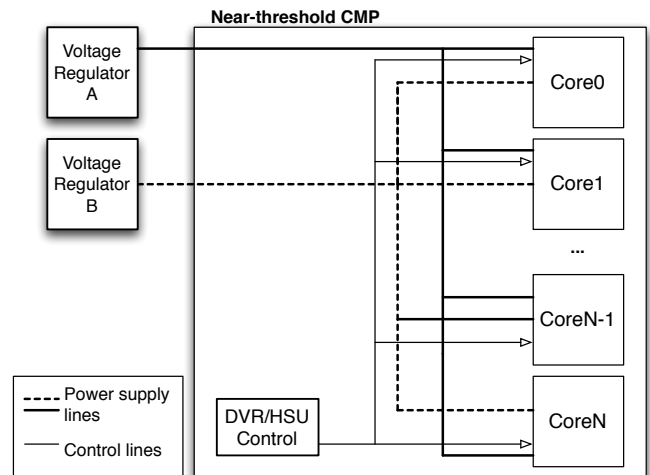


Figure 1: Overview of the proposed near-threshold CMP with DVR.

2.1.1 Post-manufacturing Calibration

Process variation is hard to predict. For DVR to be effective at reducing within-die variation, a post-manufacturing calibration process is needed. Calibration can be performed during burn-in while the chip is also tested for defects. Calibration involves two stages. In the first stage, a set of built-in self-tests (BIST) will be used to characterize the variation profile of the die. The variation profile provides a mechanism for estimating the maximum frequency each core can achieve as a

function of V_{dd} and its internal V_{th} distribution. This process is detailed in Section 2.3.

The second calibration step uses the variation profile of each chip to perform an off-line (and off-chip) optimization to choose the V_{dd} levels for the two DVR rails and which cores should be assigned to each rail. Various optimization criteria may be used for this step; for instance, to maximize CMP frequency under iso-power constraints. One straightforward optimization is to maximize CMP frequency under iso-power constraints.

Since calibration is performed off-line and off-chip, it does not increase testing time of the processor significantly. Once calibration is complete the DVR configuration is programmed in each chip’s firmware. Neither the variation profile nor the power estimations have to be very precise. Any imprecision will result in slight deviations in the actual power profile achieved. The chip will still undergo the regular frequency binning process to determine its maximum safe frequency.

2.2 Half-Speed Unit (HSU)

Within-core variation is another important hinderance to the efficiency of NT CMPs. At very low V_{dd} , delay variation between functional units can be substantial, resulting in lower core frequencies. This is because the frequency of a core is dictated by the critical path delay of the slowest functional unit. To improve individual core frequency in the presence of a few slow units, Half-Speed Unit (HSU) allows slow units in a core to operate at half the main clock frequency. This moves the slow units out of the critical path, allowing core frequency to be raised substantially.

Figure 2 shows the effect of HSU on the SPEC benchmark mean performance of a core randomly chosen from our variation model. At baseline frequency, all functional units are running at full speed. As frequency increases, the first unit that becomes critical is, in this case, the integer ALU cluster (“int”). It is set to half speed, and performance initially drops by 20%. Frequency however can be raised by about 15%, making up for some of the performance loss, before the next slower unit must have HSU applied. After applying HSU to the “fp” cluster the frequency can continue to rise, bringing performance above the initial baseline. If there is more than $2\times$ frequency variation within a core, then once frequency reaches maximum ($2\times$ baseline), not all units will be at half speed, for an overall increase in performance over baseline.

While individual cores can benefit from improved performance with HSU, a more substantial benefit is the improved frequency of the entire CMP. Applying HSU to the slowest cores allows the CMP clock frequency to be raised, significantly improving the aggregate CMP performance. Even if the performance of some cores is reduced by HSU, the loss is more than offset by an increase in performance of the other cores of the CMP that can now run at higher frequency.

2.2.1 HSU Implementation

HSU has several implementation advantages. Since the HSU clock is $1/2$ the system clock, skew between the two domains is fixed and can be kept to a minimum. Moreover, because slow units run at precisely half the speed of the fast ones,

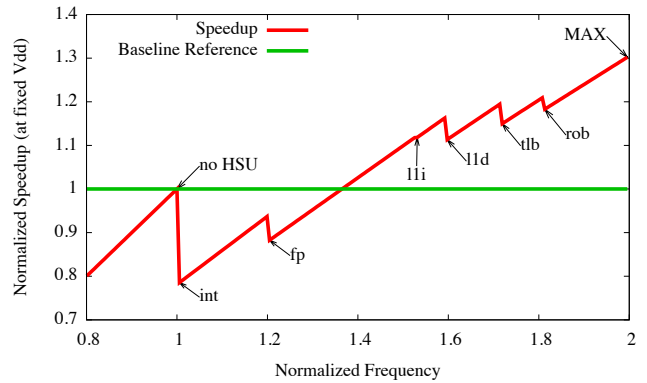


Figure 2: Frequency vs. average speedup for a core with HSU running SPEC2000 benchmarks. Performance drops when a unit’s frequency is dropped to half-speed.

these units can be easily synchronized with the rest of the core. The previously proposed GALS [14] architecture runs the main functional units on completely independent clocks to mitigate variation. GALS requires asynchronous queues to control dataflow between clock domains, and these can add significant latency. The HSU design is much simpler because it does not require inter-stage communication queues beyond those present in an out-of-order processor. Slow functional units will simply have double the latency of the same unit running at full speed.

HSU employs clock dividers for each functional block that can be switched on when the block has to be run at half speed. This avoids the clock net redundancy that would be required with a centralized divider. The clock divider circuit is essentially a multiplexer between the system clock and the output of a toggle flip-flop that is driven by the system clock; since delay through the toggle flip-flop will skew the half-rate clock relative to the system clock, additional delay is also added to the system clock, after the multiplexer, to keep the clock edges aligned.

Our HSU implementation divides a processor into functional blocks (groups of functional units) so as to minimize the architectural challenges associated with having one component communicating with another that is operating at half speed. Figure 3 shows the HSU granularity in our design. The following functional blocks can be independently switched to half-speed if needed: *inor*, the entire in-order section (fetch, decode, etc.); *l1i* and *l1d*, the L1 caches; *tlb*, the translation lookaside buffer; *ls*, loads, stores, the load-store queue, and address calculations; *int*, all integer ALU units; *fp*, all floating-point ALU units; and *rob*, the unified reorder buffer.

For basic architectural reasons, there is no benefit to subdividing the in-order section. Besides certain limited functions like branch prediction, the in-order section is a straight pipeline, where limiting the rate of any one component would effectively limit the rate of all others in the same way. Communication between the in-order section and the rest of the CPU typically involves instruction queues; bridging the clock boundary requires a synchronous queue that allows the head to run at half or double the speed of the tail.

In many CPU architectures, there are separate schedulers

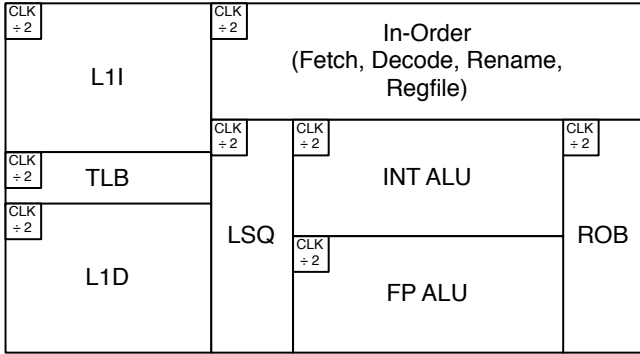


Figure 3: Overview of Half-Speed Unit, with clock dividers for each functional unit block. Units can run on the system clock or enable the divider to run at half-speed.

for different classes of instructions. For instance, integer and floating point ALUs may operate independently. *ls* must be designed to accommodate either or both of *l1d* and *tlb* at half speed. Result forwarding within the *int* and *fp* clusters requires no special considerations, since all units within these blocks operate at the same frequency. Data communication between clusters occurs through buffers like the *rob*. If the *rob* operates at half-speed, it restricts instruction commit to every other clock cycle relative to an ALU at full speed. This requires special consideration within each ALU’s instruction scheduler, to schedule instructions so that no completing instruction is passed to the *rob* on the falling edge of the half-speed clock. Thus, the most intrusive architectural change is to the instruction schedulers. Since both *inor* and *rob* access the physical register file (PRF), one or both may be limited to half-speed if the PRF itself is slow.

2.3 Chip Variation Mapping

In order to compensate for a die’s variation, we must build a profile of that variation that can be used in the post-manufacturing calibration step. Previous work has shown that post-manufacturing device characterization can be achieved with low overhead [20].

One approach is to use existing BIST hardware during burn-in. Burn-in times range from minutes to hours, depending on the chip and its application, and efficiency is maintained by performing burn-in in parallel on large batches of chips [21]. The BIST circuit must have sufficient coverage to identify which functional unit has failed the test. Our objective is to identify the frequency/voltage relationship for each functional block so that we can predict the maximum frequency for every block at any V_{dd} . Testing begins at a frequency low enough that every valid circuit will pass BIST. Frequency is increased in small steps, and at each step, all BIST circuits are activated in parallel. If any circuit fails BIST, we can estimate the functional block’s worst V_{th} as a function $V_{th} \approx f(V_{dd} + V_{guardband}, F_{fail} - F_{step})$. Testing continues at higher and higher frequency until the fastest functional block of the fastest core finally fails. The completed procedure results in a V_{th} map for every chip in the testing batch, at functional block granularity.

CMP architecture	
Cores	64, out-of-order
Fetch/issue/commit width	2/2/2
Register file size	40 entry
L1 data cache	2-way 16K, 1-cycle access
L1 instruction cache	1-way 16K, 1-cycle access
Shared L2	8-way 16 MB, 10 cycle access
Technology	32nm
Nominal V_{dd}	900mV
Near threshold V_{dd}	300mV – 500mV
Nominal Frequency	2GHz @ 900mV
Near threshold Frequency	400Mhz @ 400mV
Variation parameters	
V_{th} mean (μ),	210mV
V_{th} std. dev./mean (σ/μ)	3% – 12%
ϕ (correlation distance)	0.1 – 1.0 of die width

Table 1: Summary of the experimental parameters.

3 Evaluation Methodology

3.1 Architectural Simulation Setup

We model a 32nm 64-core CMP. Each core is dual-issue out-of-order, similar to the ARM Cortex-A9 (see Table 1). We modified SESC [22] to simulate the CMP and ran the SPEC CPU2000 benchmarks, SPECint (*crafty*, *mcf*, *parser*, *gzip*, *bzip2*, *vortex*, and *twolf*) and SPECfp (*wupwise*, *swim*, *mgrid*, *applu*, *apsi*, *equake*, and *art*).

To simulate the impact of HSU on performance, we ran all benchmarks for each possible HSU profile. Since there are eight different blocks that can be run at half speed, this required 256 (e.g. mcf_0 to mcf_{255}) simulations for each benchmark.

3.2 Technology Models

We model variation in threshold voltage (V_{th}) using VARIUS [23]. Each chip is modeled as a grid of points and each point is given one value of V_{th} assumed to have a normal distribution with mean μ and standard deviation σ . Variation is also characterized by a spatial correlation, so that adjacent areas on a chip have roughly the same V_{th} . Spatial correlation is characterized by a correlation distance ϕ , at which there is no significant correlation between two grid points. ϕ is expressed as a fraction of the chip width.

Table 1 shows some of the process parameters used. Each individual experiment uses a batch of 100 chips that have a different variation map generated with the same mean μ , standard deviation σ , and correlation distance ϕ . To generate each map, we use the geoR statistical package [24] of R [25].

For power and delay at NT, we use the Marković [6] model.

4 Evaluation

We evaluate the performance improvement and energy savings achieved by a CMP with DVR and HSU applied both independently and in conjunction. We begin by evaluating the impact of process variation on the frequency of NT CMPs.

4.1 Frequency Variation at Near-Threshold

Process variation has a much greater effect on core frequency at near-threshold than at nominal V_{dd} . Figure 4 illustrates

$V_{th} \sigma/\mu$	Freq. σ/μ at 900mV	Freq. σ/μ at 400mV
3%	1.0%	7.5%
6%	2.1%	15.1%
9%	3.2%	22.8%
12%	4.4%	30.6%

Table 2: Frequency variation as a function of V_{th} variation and V_{dd} .

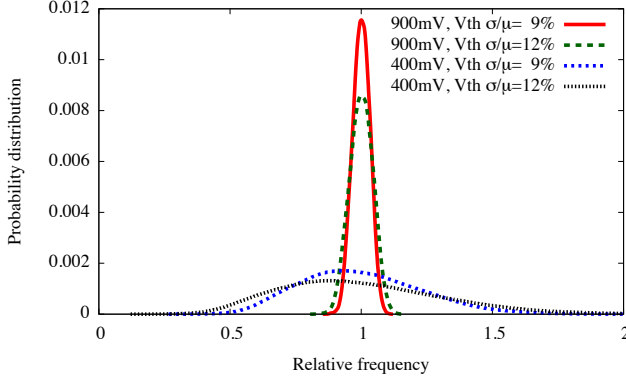


Figure 4: Core-to-core frequency variation at nominal and near-threshold V_{dd} , relative to die mean.

core-to-core variation in frequency as a probability distribution function (PDF) of core frequency divided by die mean (average over all cores in the same die). Distributions are shown for 9% and 12% within-die V_{th} variation (σ/μ). At nominal V_{dd} the distribution is tight, with only 4.4% frequency σ/μ . At NT, cores vary from less than half to more than $1.5\times$ mean, for a very large 30.6% σ/μ variation. Table 2 shows the impact of different V_{th} variation levels on the σ/μ of frequency variation at nominal and NT voltages.

The high within-core variation has a dramatic impact on CMP frequency. Without variation, a 32nm CMP would be expected to run at about 400MHz at $V_{dd} = 400mV$. With a 12% V_{th} variation our model indicates an average frequency across all dies of 149MHz, with a minimum of 75MHz and a maximum of 230MHz, for the same V_{dd} . Clearly, variation has a very detrimental effect on the frequency of NT CMPs.

Figure 5 shows the within-core effect of variation at nominal V_{dd} versus near-threshold. The graph shows the PDF of the maximum frequency of a functional unit divided by core mean (average over all units in the same core). Distributions are shown for 9% and 12% V_{th} variation (σ/μ). Within-core variation is smaller than core-to-core but still substantial.

4.2 Variation Reduction with DVR and HSU

4.2.1 Performance Improvements from DVR

DVR reduces core-to-core variation by assigning cores to one of two different voltages according to their variation profile. The goal of the optimization is to improve frequency while keeping power consumption constant. Figure 6 shows the effect of DVR on the core frequency distribution, compared to a single voltage rail (SVR), for the same power. DVR significantly tightens the frequency variation, reducing the right tail of the bell curve and reducing the left tail even more. As result, core frequency σ/μ is reduced from 30.6% to 23.1%.

Mean frequency actually goes down with DVR, but per-

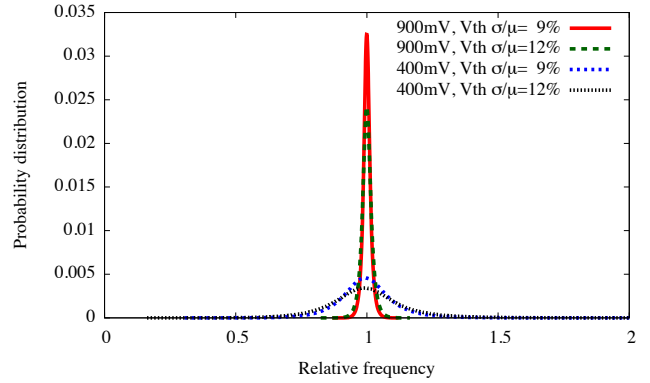


Figure 5: Within-core frequency variation at nominal and near-threshold V_{dd} .

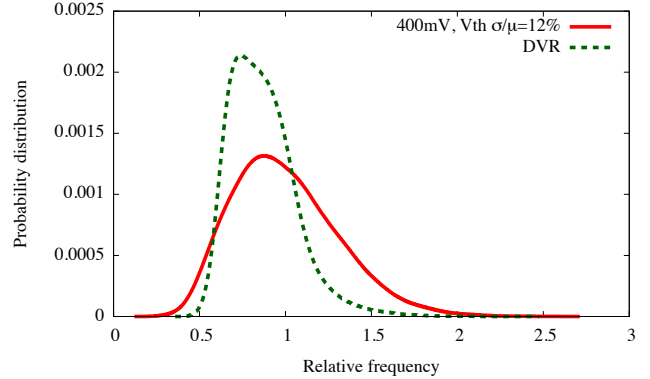


Figure 6: Core-to-core frequency variation for DVR versus SVR. Data points are normalized to SVR die mean.

die worst-case frequency (which limits system clock speed) increases by about 30% on average, as shown in Figure 7. We also compare the DVR improvement with the ideal case of having each core at its own optimal V_{dd} (64V_{dd} in Figure 7). DVR, with only two voltage rails, improves efficiency (+30%) by more than half as much as having independent voltage rails for each core (+57%). Note that the ideal case is not practical to implement because of the large number of power lines and voltage regulators required.

DVR yields significant performance improvements even though the voltage difference between two power rails is not very large. The average difference between $V_{DVR-low}$ and $V_{DVR-high}$, across all chips we simulate is 66mV. The maximum difference is 120mV, and the minimum is 30mV. The average $V_{DVR-low} = 364mV$ and $V_{DVR-high} = 429mV$.

4.2.2 Performance Improvements from HSU

HSU helps improve chip performance by mitigating within-core variation. We show two options for applying HSU. The first (HSU_{isoP}) is iso-power. Both the supply voltage and the HSU profile are optimized to improve CMP performance while keeping power consumption the same as baseline. This may reduce the performance of some cores to below baseline.

The second (HSU_{isoV}) keeps V_{dd} unchanged at 400mV and raises frequency as much as possible to achieve the greatest performance, without limiting power. This has the advantage of ensuring that no core's performance is lower than baseline.

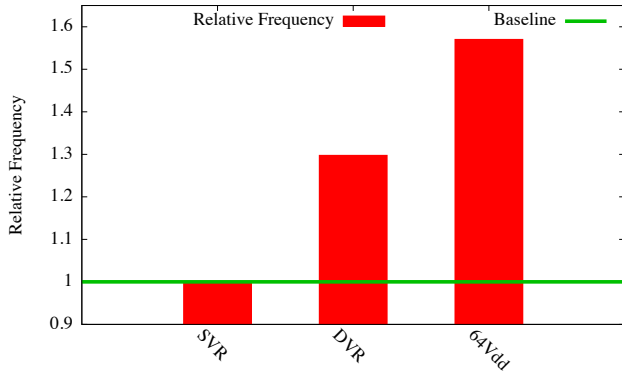


Figure 7: Average frequency increase from DVR relative to the SVR baseline. For reference, we show the theoretical best case where every core has its own ideal voltage supply ($64V_{dd}$).

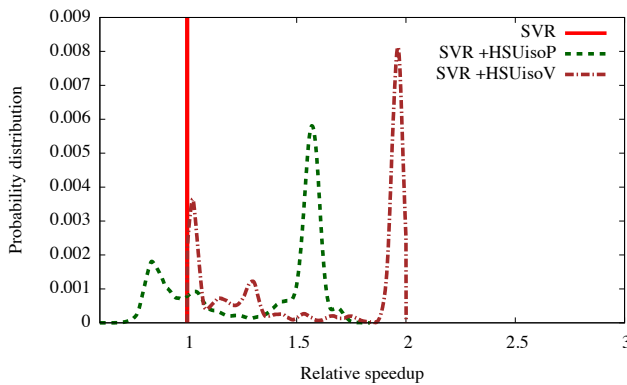


Figure 8: Core speedup (IPS increase) relative to unoptimized baseline (SVR, no HSU)

Figure 8 shows the effects of HSU_{isoP} and HSU_{isoV} on core performance. For HSU_{isoP} most cores see a performance improvement with the greatest number of cores clustering around 50% speedup. Some cores do see a performance degradation. HSU_{isoV} has a similar distribution, but shifted to the right; no cores are slower than the baseline and the majority have an almost $2\times$ increase in performance.

Figure 9 shows the performance improvement from HSU, averaged across all chips in our experiments, broken down by benchmark. HSU_{isoP} achieves an average speedup of 32% over the baseline, for the same power consumption. HSU_{isoV} does even better, with a speedup of 58% over the baseline, at the same V_{dd} , but with a higher power consumption.

4.2.3 Performance Improvements from DVR and HSU

DVR and HSU can be combined to further improve performance in the presence of variation. DVR and HSU address different variation issues and therefore synergize well.

Figure 10 shows the per-benchmark effects of DVR, HSU, and their combination. On average DVR alone improves performance by 29%. When combined with HSU_{isoP} and HSU_{isoV} the performance improvement jumps to 48% and 49% respectively. This shows that DVR and HSU combine very well to achieve an almost 50% performance improvement over the baseline NT CMP.

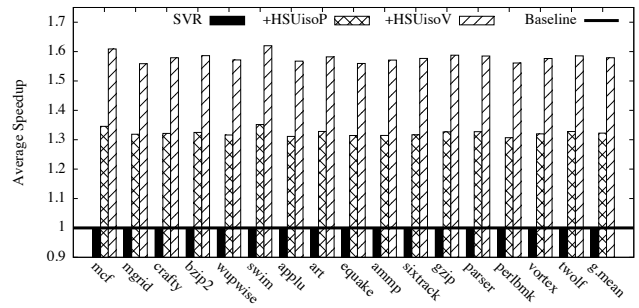


Figure 9: Per-benchmark speedup (IPS increase) relative to unoptimized (SVR, no HSU)

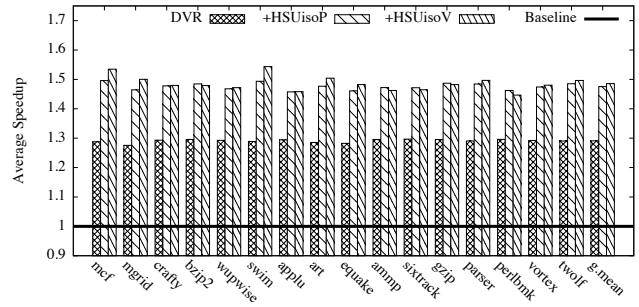


Figure 10: Per-benchmark speedup (IPS increase) relative to unoptimized (SVR, no HSU)

4.3 Energy Savings

Since DVR and HSU reduce runtime for the same power, energy is reduced. Figure 11 shows CMP energy reduction, averaged across chips. DVR reduces CMP energy by about 23% of baseline, HSU by around 25%, and together around 32%.

5 Related Work

Zhai et al [26] examine a chip multiprocessor architecture designed to run in near-threshold. Since optimal frequency and voltage differ for cores and caches, they organize the CMP in clusters of cores that share a single fast L1 cache. This improves energy efficiency over a traditional architecture. Dreslinski et al [27] developed a reconfigurable, hybrid cache architecture designed to operate reliably at near-threshold.

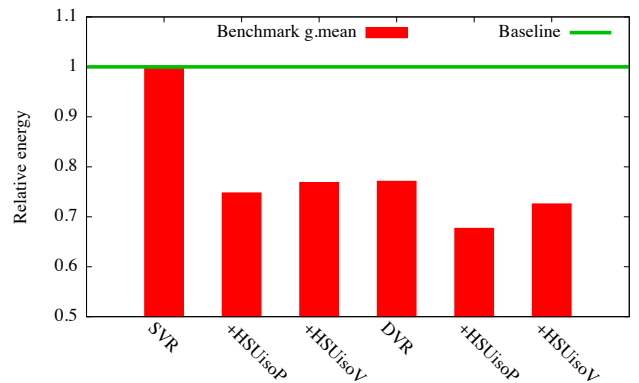


Figure 11: Energy (execution time \times average power) for DVR and HSU relative to baseline (SVR, no HSU). Post-manufacturing optimization goal is performance improvement.

Previous work has examined dual and multi- V_{dd} designs with the goal of improving energy efficiency. Most previous work has focused on tuning the delay vs. power-consumption of paths at fine granularity within the processor. For instance, in [17], circuit blocks along critical paths are assigned to the higher power supply, while blocks along non-critical paths are assigned to a lower power supply. This converts the timing slack from non-critical paths to energy savings. In [28] power optimization is achieved with simultaneous V_{dd} and V_{th} assignment. [29] presents a solution that uses a second higher V_{dd} rail for speeding up critical paths in near-threshold circuits at very fine (standard cell row) granularity. Revival [13] proposes voltage interpolation for reducing delay variation. Their solution involves very fine-grained voltage selection, at the pipeline-stage level. These solutions assign multiple voltages at much finer granularity than in our design, therefore incurring a higher design complexity.

6 Conclusion

Process variation significantly degrades performance in NT chips. This paper presents a set of simple, low-overhead, and highly effective techniques for mitigating core-to-core and within-core frequency variation in NT CMPs. By reducing variation, our solutions improve CMP performance by 48% compared to a variation-unaware CMP at near-threshold.

References

- [1] "International Technology Roadmap for Semiconductors (2009)."
- [2] R. McGowen, C. Poirier, C. Bostak, J. Ignowski, M. Millican, W. Parks, and S. Naffziger, "Power and temperature control on a 90-nm Itanium family processor," vol. 41, no. 1, pp. 229–237, January 2006.
- [3] J. Torrellas, "Architectures for extreme-scale computing," *IEEE Computer*, vol. 42, pp. 28–35, November 2009.
- [4] A. Chandrakasan, D. Daly, D. Finchelstein, J. Kwong, Y. Ramadass, M. Sinangil, V. Sze, and N. Verma, "Technologies for ultradynamic voltage scaling," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 191–214, February 2010.
- [5] R. Dreslinski, M. Wiecekowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: Reclaiming Moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, feb. 2010.
- [6] D. Markovic, C. Wang, L. Alarcon, T.-T. Liu, and J. Rabaey, "Ultralow-power design in near-threshold region," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 237–252, feb. 2010.
- [7] T. Miller, J. Dinan, R. Thomas, B. Adcock, and R. Teodorescu, "Parichute: Generalized turbocode-based error correction for near-threshold caches," in *International Symposium on Microarchitecture (MICRO)*, 2010.
- [8] H. Jiang and M. Marek-Sadowska, "Power gating scheduling for power/ground noise reduction," in *Design Automation Conference*. New York, NY, USA: ACM, 2008, pp. 980–985.
- [9] 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*, vol. 37, no. 11, pp. 1396–1402, February 2002.
- [10] S. 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*, 2002, pp. 721–725.
- [11] R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "Mitigating parameter variation with dynamic fine-grain body biasing," in *International Symposium on Microarchitecture*, December 2007, pp. 27–39.
- [12] A. Tiwari, S. R. Sarangi, and J. Torrellas, "ReCycle: Pipeline adaptation to tolerate process variation," in *International Symposium on Computer Architecture*, June 2007.
- [13] X. Liang, G.-Y. Wei, and D. Brooks, "Revival: A variation-tolerant architecture using voltage interpolation and variable latency," *IEEE Micro*, vol. 29, no. 1, pp. 127–138, 2009.
- [14] D. Marculescu and E. Talpes, "Variability and energy awareness: A microarchitecture-level perspective," in *Design Automation Conference*, June 2005.
- [15] J. Dorsey, S. Searles, M. Ciraula, S. Johnson, N. Bujanos, D. Wu, M. Braganza, S. Meyers, E. Fang, and R. Kumar, "An integrated quad-core Opteron processor," in *International Solid-State Circuits Conference*, February 2007, pp. 102–103.
- [16] 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.
- [17] S. Kulkarni, A. Srivastava, and D. Sylvester, "A new algorithm for improved VDD assignment in low power dual VDD systems," in *International Symposium on Low Power Electronics and Design*, May 2004, pp. 200–205.
- [18] K. Kim and V. D. Agrawal, "True minimum energy design using dual below-threshold supply voltages," *VLSI Design, International Conference on*, vol. 0, pp. 292–297, 2011.
- [19] K. Kim and V. Agrawal, "Minimum Energy CMOS Design with Dual Subthreshold Supply and Multiple Logic-Level Gates," in *Proc. 12th International Symposium on Quality Electronic Design*, 2011.
- [20] F. Koushanfar, P. Boufounos, and D. Shamsi, "Post-silicon timing characterization by compressed sensing," in *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2008, pp. 185–189.
- [21] C.-Y. Lee, R. Uzsoy, and L. A. Martin-Vega, "Efficient algorithms for scheduling semiconductor burn-in operations," *Operations Research*, vol. 40, no. 4, pp. 764–775, 1992.
- [22] J. Renau, B. Fraguola, 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>.
- [23] S. R. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "VARIUS: A model of parameter variation and resulting timing errors for microarchitects," *IEEE Transactions on Semiconductor Manufacturing*, February 2008.
- [24] P. Ribeiro Jr. and P. Diggle, "geoR: A package for geostatistical analysis," *R-NEWS*, vol. 1, no. 2, 2001. [Online]. Available: <http://cran.R-project.org/doc/Rnews>
- [25] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, 2006, <http://www.R-project.org>.
- [26] B. Zhai, R. G. Dreslinski, D. Blaauw, T. Mudge, and D. Sylvester, "Energy efficient near-threshold chip multiprocessing," in *International Symposium on Low Power Electronics and Design*. ACM, 2007, pp. 32–37.
- [27] R. G. Dreslinski, G. K. Chen, T. Mudge, D. Blaauw, D. Sylvester, and K. Flautner, "Reconfigurable energy efficient near threshold cache architectures," in *International Symposium on Microarchitecture*. IEEE Computer Society, 2008, pp. 459–470.
- [28] K. Roy, L. Wei, and Z. Chen, "Multiple-Vdd multiple-Vth CMOS (MVC MOS) for low power applications," in *IEEE International Symposium on Circuits and Systems*, vol. 1, 1999, pp. 366–370.
- [29] M. R. Kakoei, A. Sathanur, A. Pullini, J. Huisken, and L. Benini, "Automatic synthesis of near-threshold circuits with fine-grained performance tunability," in *Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design*, ser. ISLPED '10. New York, NY, USA: ACM, 2010, pp. 401–406.