Microarchitecture Parameter Selection To Optimize System Performance Under Process Variation

Xiaoyao Liang and David Brooks
Division of Engineering and Applied Sciences
Harvard University
33 Oxford Street, Cambridge MA 02138 USA
{xliang,dbrooks}@eecs.harvard.edu

Abstract— **Design variability due to within-die and die-to-die process variations has the potential to significantly reduce the maximum operating frequency and the effective yield of high-performance microprocessors in future process technology generations. This variability manifests itself by increasing the number and criticality of long delay paths. To quantify this impact, we use an architectural process variation model that is appropriate for the analysis of system performance in the early-stages of the design process. We propose a method of selecting microarchitectural parameters to mitigate the frequency impact due to process variability for distinct structures, while minimizing IPC (instructions-per-cycle) loss. We propose an optimization procedure to be used for system-level design decisions, and we find that joint architecture and statistical timing analysis can be more advantageous than pure circuit level optimization. Overall, the technique can improve the 90% yield frequency by about 14% with 3% IPC loss for a baseline machine with a 20FO4 logic depth per pipestage. This approach is sensitive to the selection of processor pipeline depth, and we demonstrate that machines with aggressive pipelines will experience greater challenges in coping with process variability.**

I. Introduction

Future advanced process technologies will continue to provide transistor density and speed improvements through aggressive feature scaling and novel device topologies. Unfortunately, chip designers will soon be forced to design with the expectation of significant variation in transistor sizes and threshold voltages due to random dopant fluctuations and sub-wavelength lithography. Process variations (PV) will manifest in several ways – through random or correlated variations that may occur within a single die (WID: within-die variation) or across multiple dies (D2D: die-to-die variation) in a production run. Recent estimates suggest that process variability could impact performance by a full process generation[1].

While the last few years have seen increased interest in developing statistical timing models and circuit-level techniques to reduce the frequency impact, there has been comparably little work at the microarchitectural level. However, key decisions that chip architects make to increase performance (e.g. selection of pipeline depth and width, sizing of architectural parameters, etc.) have a substantial impact on the number and distribution of critical paths. Furthermore, many techniques that can be used to combat process variation in various microarchitectural structures for delay compensation have inherent costs and must be applied selectively during the definition of the chip architecture. Total system performance, $clockfrequency \times IPC$, should be used as the design metric, and designers must be careful to avoid naive optimization of only one of the two performance components, as such approaches may not boost the overall system performance and post-fabrication yield.

This paper takes several steps in the direction of PV-tolerant design at the system-level. Specifically, this paper makes three major contributions:

- We study the potential of several simple techniques to reduce the frequency impact of process variations that are appropriate for various types of microarchitectural structures. IPC and frequency tradeoffs for these techniques are analyzed in detail.
- We propose an approach to select performance optimal design parameters under PV based on a baseline design that assumes nominal delay. This approach provides significant performance benefits over the PV-unaware design and motivates the need for system designers to integrate statistical performance analysis into the early stages of the design process.
- We show that the effectiveness of the optimization methods will change with pipeline depth and can significantly impact architectural choices.

In the next section we discuss background and other related work. Section III discusses our PV-modeling approach. Section IV describes ways to select architecture parameters to mitigate the frequency impact of PV and evaluates these techniques for a baseline 20FO4 microprocessor. Section V demonstrates how these approaches scale with deeper and shallower pipelines. Finally, our work is summarized in Section VI.

II. Related Work

Most research exploring process variation has been performed at the logic, circuit, and device levels. Bowman et al.

perform circuit-level analysis and point out that a technology generation of performance can be lost due to device variability. This paper also presents the F_{MAX} model and validates the model for a large number of dies [1]. In recognition of process variability, researchers have begun to develop statistical timing analysis techniques to be applied to deep sub-micron chip designs [6], [13], [15], [16], [17], [18], [19], [20].

Borkar et al. conceptually demonstrate that the performance gain of deeper pipelines decreases due to the impact of within-die process variation [2]. Kim et al. quantify this effect and determine that process variation will shift the optimal logic depth from 6 to 8 FO4 for a 1-wide processor due to the averaging effect of the delay variations [4]. Tschanz et al. [7] propose adaptive body bias (ABB) and Narendra et al. [11] propose forward body bias (FBB) to mitigate the impact of variation for several critical path structures, but neither of these studies consider system-level effects. Datta et al. demonstrate that an unbalanced pipeline design can increase yield [5].

Comparably little work has considered the impact of process variations at the architectural and system level. Recently, Agarwal et al. proposed a variation tolerant cache architecture [3]. Marculescu and Talpes discuss the merits of globally-asynchronous, locally synchronous (GALS) techniques to design processors under process variation [8]. In this paper, we base our analysis on fully-synchronous processors and propose several simple techniques to reduce the variation impact within standard architectural structures.

III. ARCHITECTURE MODELING UNDER PROCESS VARIATION

In this section, we discuss our experimental methodology, including an introduction to the modeling method used to predict the delay distribution of the processor. Our system-level approach necessitates a somewhat simplified model compared to circuit-level statistical timing models. As models for process variation at the architectural-level mature, newer models can be easily integrated into our design flow.

A. Architecture Delay Distribution Modeling

For early-stage modeling of processor architectures, it is not possible to perform detailed circuit-level statistical analysis, because no hardware has been implemented and RTL code is often unavailable. However, system architects must make key early-stage design decisions such as selecting the logic depth for each pipeline stage (in terms of levels of FO4 logic) before carrying out detailed circuit design. The logic depth of the machine is defined as the nominal critical path delay of the machine which decides the final chip frequency.

Our delay distribution model builds upon the generic critical path (GCP) and maximum frequency distribution (FMAX) model presented in [1]. The method is summarized below. For clarity, we consider gate length as the only variation source in this paper, but any other independent variation source can be modeled with the same approach. The effective gate length of a device is modeled in Eq. (1), and the corresponding gate delay is in Eq. (2).

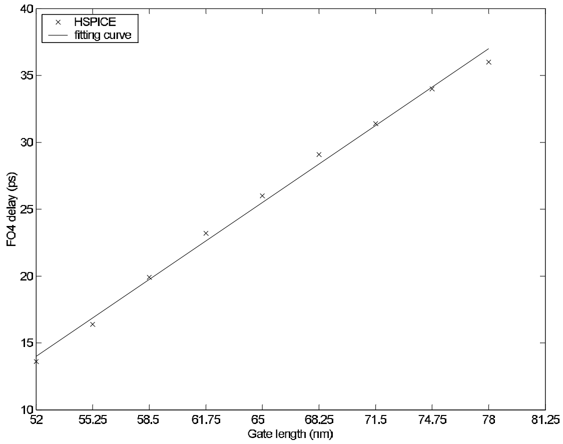


Fig. 1. Delay fitting curve with gate length variation.

Here L_0 is the nominal gate length, ΔL_{D2D} and ΔL_{WID} are die-to-die and within-die gate length variations which obey the normal distribution [1]. D_0 is the gate nominal delay. All the devices on a chip share a single ΔL_{D2D} but may have different ΔL_{WID}. In order to extract the delay variation due to gate length variation, we use a first order approximation which is widely used in statistical timing analysis [6], [13]. Fig. 1 shows the FO4 gate delay and gate length fitting curve under the 65nm technology node using Berkeley Predictive Technology Models (BPTM) [14]. A_{fit} is extracted from the linear fitting curve.

$$L = L_0 + \Delta L_{D2D} + \Delta L_{WID} \tag{1}$$

$$\begin{aligned} D =& D_0 + \Delta D_{D2D} + \Delta D_{WID} \\ =& D_0 + A_{fit} \times \Delta L_{D2D} + A_{fit} \times \Delta L_{WID} \end{aligned} \tag{2}$$

For a critical delay path with n_{cp} gates, the path delay is the summation of each individual gate on the path, shown in Eq. (3). We model delay variation due to die-to-die gate length using a normal distribution as shown in Eq. (4, 7). For within-die variation, the path's standard deviation is shown in Eq. (5) for completely systematic WID variations and Eq. (6) for pure random WID variations. The real variation is reported to be in between these two extreme cases [1]. We use pure random WID variation with same assumptions described in [4], [8] and model the within-die delay distribution for a single path as in Eq. (8). If the independent critical path number is N_{cp}, the final maximum critical path delay distribution is defined in Eq. (9, 10) according to the FMAX model, where F_{WID} is a cumulative distribution function (CDF) of f_{WID}(PDF) and $\delta(n_{cp}D_0)$ is an impulse function at path nominal delay value.

$$\begin{aligned} D_{path} =& n_{cp}D_0 + n_{cp}A_{fit} \times \Delta L_{D2D} \\ & + A_{fit} \times (\Delta L_{WID1} + \Delta L_{WID2} + ... + \Delta L_{WIDncp}) \end{aligned} \tag{3}$$

$$\sigma D_{D2D} = n_{cp} A_{fit} \times \sigma L_{D2D} \tag{4}$$

$$\sigma D_{WID} = n_{cp} A_{fit} \times \sigma L_{WID} \tag{5}$$

$$\sigma D_{WID} = \sqrt{n_{cp}} A_{fit} \times \sigma L_{WID} \tag{6}$$

Two parameters (n_{cp} and N_{cp}) are used in the formula. We use the same approach to calculate N_{cp} as in [8]. n_{cp} is the logic depth of each pipeline stage [4]. We also use CACTI [12] to report the most suitable SRAM nominal delay to fit the machine. We assume $\sigma L_{WID}/L_0 = \sigma L_{D2D}/L_0 = 5\%$, which is similar to the assumption used in [1], [8].

$$f_{D2D} = N(0, \sigma D_{D2D}) \tag{7}$$

$$f_{WID} = N(0, \sigma D_{WID}) \tag{8}$$

$$f_{WID-dmax} = N_{cp} f_{WID} \times (F_{WID})^{N_{cp-1}} \tag{9}$$

$$f_{dmax} = \delta(n_{cp} D_0) * f_{D2D} * f_{WID-dmax} \tag{10}$$

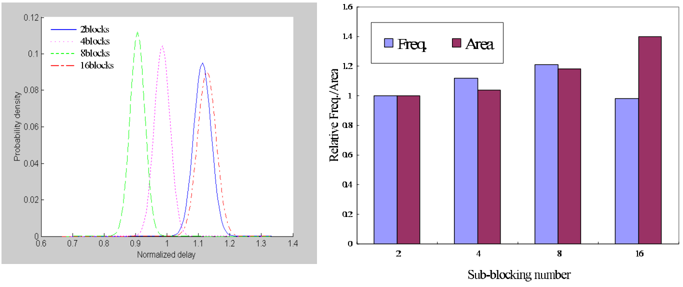
B. IPC Simulation Methodology

For our baseline machine, we assume a 20FO4 design which is comparable to the reported pipeline logic for out-of-order microprocessors such as the Alpha 21264 [9]. For our IPC simulations, we utilize the validated *sim-alpha* simulator which provides ample parameters for size and latency scaling [10]. We use 21 of the 26 SPEC2000 benchmarks (we had difficulty simulating the other five benchmarks within our environment). Single IPC numbers reported in this paper refer to the mean of all benchmarks.

IV. ARCHITECTURE PARAMETER SELECTION

Most PV-reduction techniques target the circuit or device level to reduce frequency loss. In this work, we consider a range of *system-level* approaches. Wise selection of architectural parameters considering the PV impact and chip yield will ease the following circuit design and physical implementation. Most architectural design parameters such as resource capacity and latency can boost the machine's IPC, but potentially also have a substantial impact on the chip's delay distribution. For example, a large number of entries in the physical register file can increase IPC by assisting in the extraction of parallelism via out-of-order execution. However, the physical register file also has a large number of critical paths and increasing the size of this structure can deteriorate the chip frequency distribution under strong PV. Thus, it is important that early-stage architectural performance studies combine joint IPC and statistical frequency analysis.

This section discusses simple PV-aware techniques for distinct microarchitectural structures. We then propose an algorithm to select the optimal architecture parameters starting with a non PV-aware machine designed with nominal delay values.



a. SRAM sub-blocking to improve delay.

b. Frequency-area trade-offs.

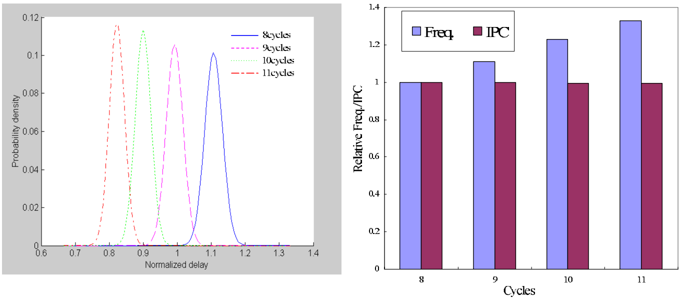
Fig. 2. SRAM sub-blocking.

A. Memory Sub-blocking

Memory sub-blocking is well-suited to large SRAM structures such as first-level instruction and data caches. Traditionally, cache designers will build a sub-blocked SRAM that can meet the nominal delay value (20FO4 in our example). But under process variation, delay exhibits a distribution and the nominal-delay SRAM will not meet timing requirements for some of the fabricated chips. By sub-blocking the SRAM into more blocks at design time and reducing the nominal delay of the SRAM, the post-fabrication delay distribution of the SRAM will shift to lower (i.e. faster) values, which will result in a higher yield for the same timing requirement. Fig. 2 (a) shows the delay distribution of a 16KB SRAM. We find that by sub-blocking the SRAM from 2 blocks to 8 blocks, the mean delay improves by more than 20%. If this SRAM array dominates the whole chip delay, chip timing yield will improve after sub-blocking this structure.

However, additional sub-blocking cannot always help. We can see from Fig. 2 (a) that when sub-blocking from 2 to 8, the delay always decreases, but when sub-blocking from 8 to 16, the delay increases and even exceeds the delay of 2 blocks. There are two reasons behind this. First, the nominal delay due to wires, multiplexors, and output logic increases quickly with more sub-blocks. Second, the amount of PV-canceling provided by sub-blocking decreases due to both smaller memory blocks (each block having a shorter critical path) and more memory blocks (a larger number of datapaths in the system). If the original design has not been sub-blocked intensively, there is room to gain some frequency back by performing additional sub-blocking. However, if the original design has already been excessively sub-blocked (e.g. in a very deeply pipelined machine), additional sub-blocking may actually harm delay.

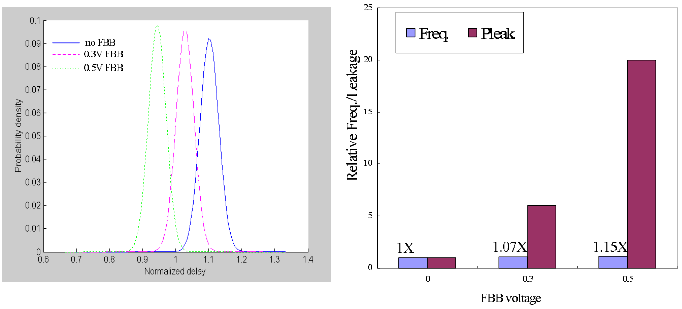
The cost of sub-blocking is not free. By sub-blocking, additional hardware is needed within the SRAM, such as more sense amplifiers or wordline decoders. Fig. 2 (b) shows the frequency-area trade-offs that we have measured.



a. Cycle extension to improve delay.

b. Frequency-IPC trade-offs.

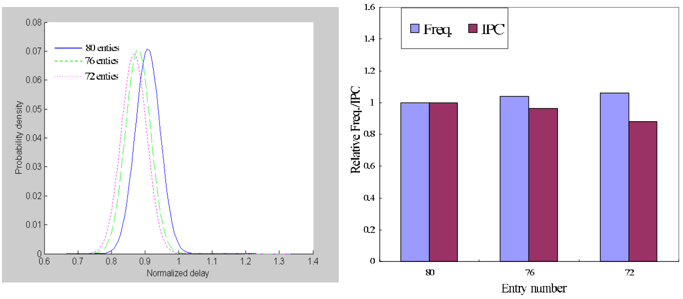
Fig. 3. Access cycles extension.



a. FBB to improve delay.

b. Frequency-power trade-offs.

Fig. 5. Forward body bias.



a. Entry reduction to improve delay.

b. Frequency-IPC trade-offs.

Fig. 4. Entry reduction.

B. Access Cycle Extension

Access cycle extension is well-suited to memory structures with long, multi-cycle access delays, such as L2 caches. Under strong PV, these structures may not easily meet nominal delay values. For example, if the L2 cache is originally designed with a 10 cycle access latency, PV may restrict many L2 caches from meeting this access time. If we take PV into account early in the design phase, we can intentionally relax the access delay (e.g. design the cache to utilize more access cycles) of the L2 cache compared with a machine designed for nominal timing. Fig. 3 (a) shows the delay distribution of cycle extension. We find that the mean delay can be improved about 15% if we extend the access time by a single cycle.

Access cycle extension will negatively impact the IPC of the processor. Fig. 3 details the IPC-frequency tradeoff that we have observed with our baseline processor model. Since the IPC loss of cycle extension is so small, it is clear that designing a machine with more access cycles could be beneficial if the nominal L2 cache would lose frequency due to PV. Jointly considering architectural IPC simulation and statistical delay analysis is necessary to determine the best L2 access latency to achieve the optimal system performance.

C. Entry Reduction

Increasing the access latency and sub-banking may not always be viable option for relatively small, performance-critical structures like register files, instruction queues, and reorder buffers. Furthermore, while adaptive body bias (ABB) [7] is a general approach that can be used to boost the speed of sub-components, it may not be appropriate for many structures. For example, for certain performance-critical components like the register file and issue queue, forward body bias can not be blindly applied because these structures are thermal hotspots. FBB may cause a large increase in leakage power and compound the thermal problem.

In order to reduce the delay-criticality of these structures, we propose to reduce the number of entries to these structures. This approach is based on the theory that the IPC of the machine may not be sensitive to reductions in the number of entries in certain buffers and queues by a small amount, because the large capacity in these resources is usually designed for worst case runtime situations. This will allow these structure to run faster and be less susceptible to PV. Reducing the number of entries in these structure will both decrease the overall nominal delay and reduce the number of critical paths which will improve the post-fabrication delay distribution. Fig. 4 (a) shows the distribution of a register file. The mean delay can be improved about 4% by cutting four entries from the register file.

However, entry reduction must be used with caution because there can be a strong IPC impact as seen in Fig. 4 (b). From this figure, we see that cutting eight entries from the physical register file decreases the IPC by 12%. We observe somewhat less IPC sensitivity with other queue structures, but designers must be careful not to adversely impact system performance with this technique. Detailed performance simulations are necessary to determine the best configuration to achieve the optimal performance point.

D. Forward Body Bias

We propose to use forward body bias for pure logic structures. Fig. 5(a) shows the delay distribution of an integer execution unit before and after FBB. The mean frequency improvement is about 15%.

FBB has the advantage of not requiring significant design changes and does not incur any IPC loss compared with the previously introduced techniques. Thus, FBB will always bring a performance boost to the system. However, the major limitation of FBB is the rapid increase in leakage power as shown in Fig. 5 (b). In this paper, we limit the usage of FBB to pure logic structures, because the leakage of logic is usually much smaller then large SRAM array structures.

Techniques	L1-cache	L2-cache	RF	Queue	ALU	TLB
Sub-blocking	X	X	X	X		X
Cycle Extension		X				
Entry Reduction	X	X	X	X		X
FBB					X	

TABLE I

AVAILABLE TECHNIQUES FOR DIFFERENT STRUCTURES.

Machines	20FO4, 4-issue
L1-cache	32KB, 2-way, sub-blocking 4, 3-cy
L2-cache	2MB, 4-way, 8-cy
Register file (RF: I-RF & F-RF)	80 entries, sub-blocking 1, 1-cy
IssueQ (IQ & FQ)	20 entries, 1-cy
Load/storeQ (LSQ: LDQ & STQ)	32 entries, 4-cy
Int EXE.	1-cy, 0FBB
Float EXE.	4-cy, 0FBB

TABLE II

SAMPLE MACHINE ARCHITECTURE PARAMETERS.

E. Choosing Techniques for Distinct Structures

The four techniques that we propose are best applied to distinct microarchitectural structures in order to maximum system performance and minimize design overhead. By trading delay with IPC, area, and leakage power during the design phase of the chip architecture, we can improve the post-fabrication timing yield.

Tab. I summarizes the proposed techniques and the microarchitectural components that are best suited for each approach. We find that sub-blocking works well with any large SRAM structure. We only apply access cycle extension to L2 caches, and we assume that applying this technique to the L2 cache will not change the effective pipeline depth of the machine. The entry reduction technique is best suited to buffer and queue structures that may be thermal hotspots. FBB can be used on any structure, but is best applied to pure logic structures that are not thermal hotspots and have a small contribution to total system power dissipation. In some cases, combinations of these techniques will also work.

F. Optimization Procedure

The delay of the processor is limited by all system components. Once a single component bottlenecks the system frequency, further optimization of the other system components is useless. By studying the delay distribution of each microarchitectural component, we can determine the best architectural parameters for each component to ensure that it will not be over or under-optimized. The advantage of performing statistical delay and IPC analysis in the architecture design stage is that the machine is no longer blindly designed with a nominal delay value. Instead, designers can target a required yield for a certain clock frequency. Our analysis shows that the best parameters chosen for the nominally designed machine can become sub-optimal under strong PV, and the statistical design methodology proposed in this work can locate and solve these problem early in the design process.

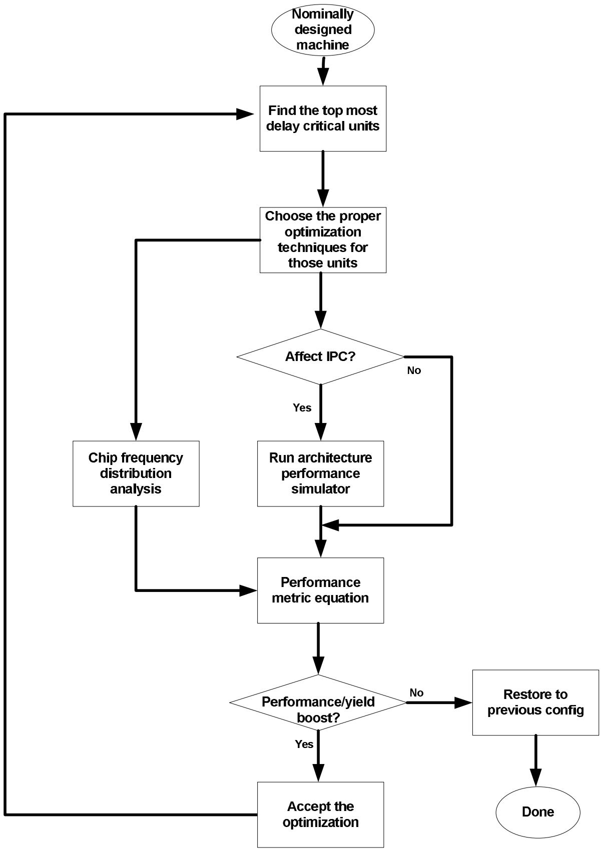

Nominally designed machine
Find the top most delay critical units
Choose the proper optimization techniques for those units
Affect IPC?
No
Yes
Chip frequency distribution analysis
Run architecture performance simulator
Performance metric equation
Performance/yield boost?
No
Restore to previous config
Yes
Accept the optimization
Done

Fig. 6. Optimization procedure.

Using the four techniques that we introduced, we propose an optimization procedure to achieve the maximum system performance under certain frequency yield requirements. The key point is to identify the microarchitectural units that are most likely to be the delay critical parts in the system under PV. Once these units are determined, we can choose from the above techniques to optimize those units. If the technique causes IPC degradation, a detailed IPC simulation must be carried out to ensure this step will yield a positive impact on system performance. We follow this procedure until all optimization techniques are exhausted for any of the units in the system that may boost performance to obtain the best design. This procedure is illustrated in Fig. 6. The starting point of the procedure is the nominally designed machine.

G. Case Study

In this section, we present a case study that uses our PV-optimization algorithm for our baseline 20FO4 machine. Table II details the baseline size and latency parameters for several of the key structures. In this study, we model 11 distinct microarchitectural units in the machine (chosen based on their relative performance and delay sensitivity), but other units can be modeled in the same way.

Fig. 8(b) shows the delay distribution of the 20FO4 machine under PV. The frequency of the 90% yield point degrades by about 20% compared with the nominally designed machine.

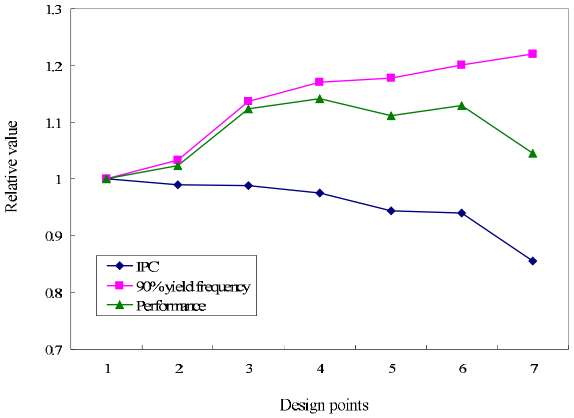


Fig. 7. Performance metric parameters change with design point.

Design point config	L1 sub-blocking	L2 cycles	RF entry	IssueQ entry	LSQ entry	EXE FBB
1	4	8	80 1block	20	32	0V
2	8	9	80 1block	16	32	0.5V
3	16	9	80 2blocks	16	28	0.5V
4	16	10	78 2blocks	16	24	0.5V
5	16	10	78 2blocks	12	24	0.5V
6	16	11	78 2blocks	12	20	0.5V
7	16	11	76 2blocks	8	16	0.5V

TABLE III
DESIGN POINTS CONSIDERED BY OPTIMIZATION ALGORITHM.

Machines	L1-cache	L2-cache	RF	LSQ	IssueQ	Fix Exe.	Float Exe.
23FO4	32KB, sub-blk 4, 3-cy	2MB, 7-cy	80, sub-blk 1, 1-cy	32, 4-cy	20, 1-cy	1-cy, 0FBB	3-cy, 0FBB
20FO4	32KB, sub-blk 4, 3-cy	2MB, 8-cy	80, sub-blk 1, 1-cy	32, 4-cy	20, 1-cy	1-cy, 0FBB	4-cy, 0FBB
17FO4	32KB, sub-blk 8, 3-cy	2MB, 10-cy	80, sub-blk 2, 1-cy	32, 5-cy	20, 2-cy	1-cy, 0FBB	5-cy, 0FBB

TABLE IV
ORIGINAL DESIGN PARAMETERS FOR 3 MACHINES.

We use the PV-optimization algorithm with the techniques presented in the previous section to improve the performance of this design. This algorithm requires many iterations in the optimization procedure as we test several design points and calculate the frequency and IPC to achieve the highest performance under PV. Tab. III documents seven sample design points that the algorithm tested (as well as a few extra to demonstrate that performance can degrade if the techniques are pushed too far). Each design point applies different techniques to reduce PV (e.g. sub-blocking, entry reduction), and thus each point has different architectural parameters. Fig.7 shows the 90% yield frequency, the IPC, and the overall system performance for the seven design points, all normalized to the machine designed with nominal delay. The optimal point in this study is shown to be design point 4 which has the best mixture of frequency boost and minimal IPC loss. We see that IPC starts to drop significantly causing system performance to degrade (point 5 through 7).

Fig. 8(b) shows the system delay distribution after optimization for design point 4. The delay distribution shifts to smaller delays and is very close to the designed nominal delay value, which ensures that 90% of the chips can run at a faster speed. For this example, we improve the 90% yield frequency about 14% at the cost of less than 3% average IPC compared with the original machine. Our analysis highlights several specific examples that demonstrate the benefit of performing statistical performance analysis during architecture definition. For example, the nominal machine utilizes a load/store queue (LSQ) with 32 entries to ensure a high IPC value. Under PV, the large delay distribution of this structure means that the frequency impact of PV surpasses the IPC benefit. The optimization procedure finds that a 24-entry LSQ is the best design to consider both IPC and delay distribution. In this example, the parameters chosen for the nominal delay machine (design point 1) become sub-optimal under strong PV.

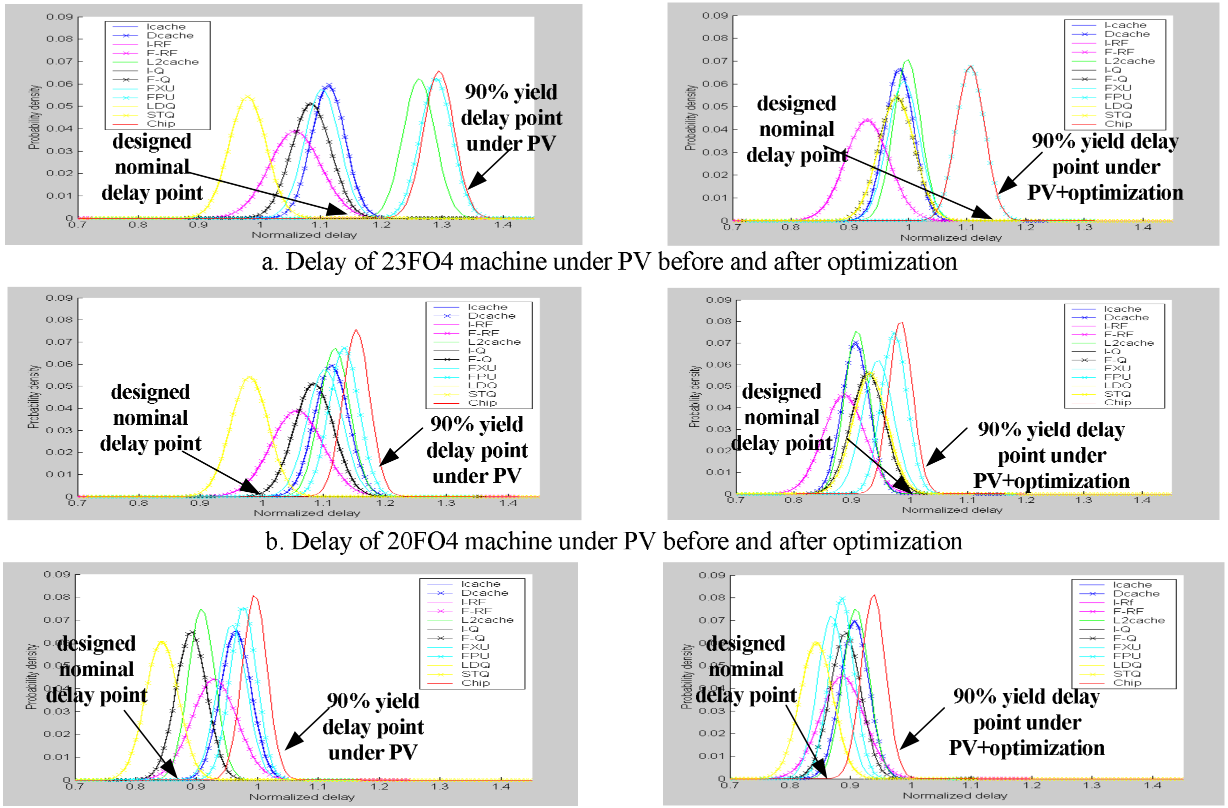
V. IMPACT OF BASELINE MICROARCHITECTURE

The techniques introduced in the previous section depend strongly on the baseline machine design. For example, if the SRAM in the original machine is aggressively sub-blocked, it is possible that further sub-blocking will not improve delay. Similarly, if certain queues and buffers in the machine are already reduced to a critical value, it may not be possible to reduce the entries further without significant IPC drop. The FBB technique also has limits, as it can provide a maximum of 10-20% delay improvement to logic, so over-optimizing other non-FBBed units more than this amount is useless, because delay will be clamped by the logic paths. Similarly, if SRAM structures cannot be further optimized and clamp the delay, providing excessive FBB to the logic will simply waste power without any frequency gain. Another key point is that the different optimization methods must be carefully tailored to the architectural structures. This again stresses the criticality of merging the analysis and optimization of process variations early in the design process.

A. Best Design Choice for Three Machines

In this section, we study three sample machines and apply the four techniques to maximize the performance of these machines under PV. We designed 23FO4, 20FO4, 17FO4 machines under nominal delay values. The design parameters of the original machines are listed in Tab. IV. We follow the optimization algorithm introduced in Sec.IV and use these techniques to achieve the best performance design point. The optimized machine parameters are listed in Tab. V.

Fig. 8 shows the delay distribution of each of the three machine before and after optimization. For clarity, the X-axis in each diagram uses the same scale. With the optimizations applied, we can improve delay distribution from each of the machines, but to a different degree. From the figure, we see that the 23FO4 machine has the largest frequency improvement. This is because most of the SRAM structures in the 23FO4 machine have not been substantially optimized for delay since the nominal delay requirement for this machine



a. Delay of 23FO4 machine under PV before and after optimization

b. Delay of 20FO4 machine under PV before and after optimization

c. Delay of 17FO4 machine under PV before and after optimization

Fig. 8. Delay distribution of 3 sample machines.

Machines	L1-cache	L2-cache	RF	LSQ	IssueQ	Fix Exe.	Float Exe.
23FO4	32KB, sub-blk 8, 3-cy	2MB, 9-cy	80, sub-blk 2, 1-cy	32, 4-cy	18, 1-cy	1-cy, 0.5V FBB	3-cy, 0.5V FBB
20FO4	32KB, sub-blk 16, 3-cy	2MB, 10-cy	78, sub-blk 2, 1-cy	24, 4-cy	16, 1-cy	1-cy, 0.5V FBB	4-cy, 0.5V FBB
17FO4	32KB, sub-blk 16, 3-cy	2MB, 10-cy	78, sub-blk 2, 1-cy	32, 5-cy	20, 2-cy	1-cy, 0.3V FBB	5-cy, 0.3V FBB

TABLE V

OPTIMIZED DESIGN PARAMETERS FOR 3 MACHINES.

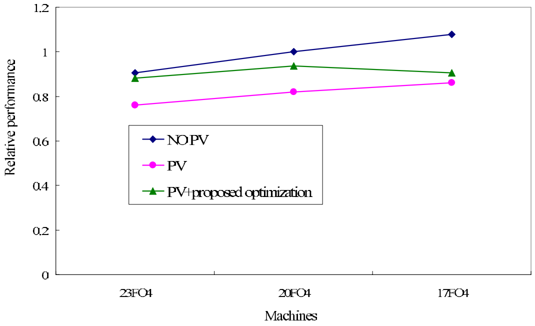


Fig. 9. Performance of 3 sample machines under different conditions.

is not tight. Thus, there is still substantial headroom in the 23FO4 machine for further SRAM-optimizations for process variation. Further optimization of SRAM structures in the 23FO4 machine is useless because the logic delay clamps the frequency of the chip, because FBB for logic can only improve the delay by at most 10% -20%. This can be seen in the delay distribution within the figure, as the floating point execution unit is limiting further optimization.

On the contrary, the final frequency of the 17FO4 machine is limited by the SRAM structures such as the register file and caches. In order to meet the relatively tight nominal delay requirement of the 17FO4 machine, the original design has used up some of the potential for SRAM sub-blocking. This leaves little optimization headroom for these structures under process variation and most of the techniques cannot be applied to 17FO4 machines. The non-SRAM logic blocks in this machine can still be accelerated via FBB techniques, but this does not significantly help because the SRAM structures eventually limit the frequency.

Applying the optimization techniques to the 20FO4 machine causes the delay distribution of logic and SRAM structures to converge to the same point. This allows the design to maximize the potential of both types of structures.

Fig. 9 shows the mean system performance of the three machines under different conditions, with the base frequency normalized to the nominal frequency of the 20FO4 machine without process variation and the IPC value normalized to the same machine. Without any process variation, the 17FO4 machine is the best choice because the nominal frequency of this machine is higher and can compensate for the IPC loss due to deeper pipelining. Under process variation, but without any optimizations, the 17FO4 machine still has the best system performance, although the advantage is reduced due to the larger frequency loss from PV for deeper pipelines (primarily due to reduced PV-canceling from shorter logic depth and more critical paths with more stages). However, when the machines are optimized using our approach, the best design point shifts to the 20FO4 machine. The 23FO4 machine has a higher IPC (due to the shallower pipeline) and more room for delay optimization, but suffers from large nominal delays. The 17FO4 machine has small nominal delay, but lower IPC (deeper pipeline) and less room for delay optimization. The 20FO4 machine is the best choice among the three machines, because it can settle to reasonable values for all the parameters allowing it to achieve the best performance.

This example demonstrates that the traditional processor design flow must be modified to take statistical performance analysis into account when making key decisions for critical parameters like processor pipeline depth. A caveat of this example is that the best point found in this section is only the best point under our design configuration and optimization method for the three machines. For designs with much deeper pipelines, the designs will have different delay and performance parameters and thus the best design point may change. This illustrates the complexity of determining the best design choice under process variation.

VI. Conclusion

In this paper we propose to use simple techniques to mitigate the impact of PV on system performance of high-performance microprocessors. We demonstrate an optimization procedure that selectively applies these techniques to various microarchitectural structures, and we show that for a typical machine, the approach can achieve good performance improvements compared to the original machine designed in a PV-unaware manner. Finally, we show the these optimization methods scale with different pipeline depths and architectural design choices. We hope that this work can encourage architects to begin to integrate statistical performance analysis and optimization into the early architecture design phase.

There is future research that can be done in the area of architectural PV-modeling including tighter integration with power and temperature modeling and variation. Furthermore, there are many additional circuit and architectural PV-optimization techniques that can be developed that would fit within the proposed optimization framework.

Acknowledgments

This work is supported by NSF grants CCF-0048313 (CAREER), CCF-0429782, Intel, and IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF, Intel or IBM.

References

[1] 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," *IEEE Journal of Solid-State Circuits, Vol. 37, No. 2*, February 2002.

[2] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. "Parameter Variation and Impact on Circuits and Microarchitecture," *Proceedings of the 40th Conference on Design Automation*, June 2003.

[3] A. Agarwal, B. C. Paul, H. Mahmoodi, A. Datta, and K. Roy. "A Process-Tolerant Cache Architecture for Improved Yield in Nanoscale Technologies," *IEEE Transactions on Very Large Scale Integration Systems, Vol. 13, No. 1*, January 2005.

[4] N. S. Kim, T. Kgil, K. Bowman, V. De, and T. Mudge. "Total Power-Optimal Pipelining and Parallel Processing under Process Variations in Nanometer Technology," *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, November 2005.

[5] A. Datta, S. Bhunia, S. Mukhopadhyay, N. Banerjee, and K. Roy. "Statistical Modeling of Pipeline Delay and Design of Pipeline under Process Variation to Enhance Yield in sub-100nm Technologies," *Proceedings of Design, Automation and Test in Europe Conference*, March 2005.

[6] A. Agarwal, D. Blaauw, S. Sundareswaran, V. Zolotov, M. Zhou, K. Gala, and R. Panda. "Path-based Statistical Timing Analysis Considering Inter and Intra-die Correlations," *In Proceedings of TAU*, June 2002.

[7] 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, Vol. 37*, November 2002.

[8] D. Marculescu and E. Talpes. "Variability and Energy Awareness: A Microarchitecture-Level Perspective," *Proceedings of the 42nd Conference on Design Automation*, June 2005.

[9] R. E. Kessler. "The Alpha 21264 microprocessor," *IEEE MICRO, Vol. 19, No.2*, March/April 1999.

[10] R. Desikan, D. Burger, S. Keckler, and T. Austin. "Sim-alpha: a Validated, Execution-Driven Alpha 21264 Simulator," *Technical Report TR-01-23, Department of Computer Sciences, University of Texas at Austin*, 2001.

[11] A. Keshavarzi, S. Narendra, B. Bloechel, S. Borkar, and V. De. "Forward Body Bias for Microprocessors in 130-nm Technology Generation and Beyond," *IEEE Journal of Solid-State Circuits, Vol. 38, No. 5*, May 2003.

[12] P. Shivakumar and N. P. Jouppi. "CACTI 3.0: An Integrated Cache Timing, Power, and Area Model," *WRL Research Report 2001/2* ,2001.

[13] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu. "Impact of Spatial Intrachip Gate Length Variability on the Performance of High-Speed Digital Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 21, No. 5*, May 2002.

[14] Berkeley Predictive Technology Model, UC Berkeley Device Group. http://www-device.eecs.berkeley.edu/ ptm/

[15] H. Chang and S. S. Sapatnekar. "Statistical Timing Analysis Considering Spatial Correlations Using a Single PERT-like Traversal," *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, November 2003.

[16] C. S. Amin, N. Menezes, K. Killpack, F. Dartu, U. Choudhury, N. Hakim, and Y. Ismail. "Statistical Static Timing Analysis: How simple can we get?," *Proceedings of the 42nd Conference on Design Automation*, June 2005.

[17] M. Pan, C. C. Chu and H. Zhou. "Timing Yield Estimation Using Statistical Static Timing Analysis," *In Proceedings of IEEE International Symposium on Circuits and Systems*, 2005.

[18] X. Li, J. Le, L. T. Pileggi, and A. Strojwas. "Projection-based performance modeling for inter/intra-die variations," *IEEE/ACM International Conference on Computer Aided Design*, 2005.

[19] D. Sinha and H. Zhou. "A Unified Framework for Statistical Timing Analysis With Coupling and Multiple Input Switching," *IEEE/ACM International Conference on Computer Aided Design*, 2005.

[20] L. Zhang, W. Chen, Y. Hu, J. Gubner, C. Chen. "Correlation-Preserved Non-Gaussian Statistical Timing Analysis with Quadratic Timing Model," *Proceedings of the 42nd Conference on Design Automation*, June 2005.