



# Design and Test Strategies for Microarchitectural PostFabrication

## Citation

Liang, Xiaoyao, Benjamin Lee, Gu-Yeon Wei, and David Brooks. 2008. Design and Test Strategies for Microarchitectural PostFabrication. Harvard Computer Science Group Technical Report TR-06-08.

## Permanent link

<http://nrs.harvard.edu/urn-3:HUL.InstRepos:24019782>

## Terms of Use

This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA>

## Share Your Story

The Harvard community has made this article openly available.  
Please share how this access benefits you. [Submit a story](#).

[Accessibility](#)

# Design and Test Strategies for Microarchitectural PostFabrication

Xiaoyao Liang, Benjamin Lee, Gu-Yeon Wei  
and David Brooks

TR-06-08



Computer Science Group  
Harvard University  
Cambridge, Massachusetts

# Design and Test Strategies for Microarchitectural Post-Fabrication Tuning

Xiaoyao Liang, Benjamin Lee, Gu-Yeon Wei, David Brooks

School of Engineering and Applied Sciences

Harvard University, Cambridge MA 02138

## Abstract

Process variations are the major hurdle for continued technology scaling. Both systematic and random variations will affect the critical delay of fabricated chips, causing a wide frequency and power distribution. Tuning techniques are capable of adapting the microarchitecture to mitigate the impact of variations at post-fabrication testing time. Most of the existing techniques ignore testing cost or simply assume a naive exhaustive testing scheme. But testing has associated costs, which might be prohibitively expensive for a large space of post-fabrication tuning configurations. This paper proposes a new post-fabrication testing framework that accounts for testing costs. This framework uses on-chip canary circuits to capture systematic variation while using statistical analysis to estimate random variation. Regression model is applied to predict the chip performance and power. These techniques comprise an integrated framework that identifies the most energy efficient post-fabrication tuning configuration for each chip. The testing cost for the proposed framework is low, usually converging with fewer than two rounds of tests. At low cost, the proposed test framework can achieve 93 to 96 percent of  $BIPS^3/W$  optimal tuning results, even under very large variations. Furthermore, the test framework fits into existing test flows without major changes to testing facilities.

## 1 Introduction

Moore's Law has driven drastic and fundamental advances in computing. Enabled by regular and predictable transistor scaling, computing capacities and capabilities have benefited from smaller devices. Process variations, a side-effect of Moore's Law scaling in nanoscale CMOS technologies, jeopardize the significant performance and power advances from scaling. In modern, aggressively-scaled technologies of 45nm and beyond, the expected 30 percent performance improvement per process generation is no longer certain [1]. Previously a problem only for analog circuits that require high precision, variations in nanoscale technologies now threaten the performance and power of digital systems implemented in static CMOS logic with full swing signals. The scope and impact of process variations is significantly greater in the digital domain. Digital designers strive to balance delays across paths through combinatorial logic blocks such that no single path defines

the critical path delay. In the presence of process variations, digital designers are less certain of their delay balancing efforts since one marginal set of transistors could potentially compromise performance for the entire system.

Although statistical timing analysis attempts to model process variations at design time, the true impact of process variations is unknown until the chip returns from fabrication. Thus, strategies to mitigate process variations often include a post-fabrication component. Post-fabrication tuning configures a chip to compensate for realized process variations. Back-body bias might be used to alter the effective threshold voltage. Supply voltages might also be tuned using localized voltage domains implemented with on-chip regulators or voltage interpolation. Latencies for microarchitectural blocks might be tuned such that blocks with long delays use an extra pipeline stage instead of lengthening the critical path delay. Finally, microarchitectural structures might be resized to exclude slower elements, thereby improving their maximum operating frequency. Collectively, these tuning techniques mitigate variations at the microarchitectural level. Analysis at this level is necessary to fully account for variations' impact on microarchitectural performance (IPC) and power.

These variation mitigating techniques expose a large space of tunable parameters, presenting significant optimization challenges. Tuning techniques are typically evaluated by exhaustively searching the space to identify optimal configurations. However, the search space size grows combinatorially with the number of tuning techniques, the tunable parameters within each technique, and the number of tuned microarchitectural structures. Thus, exhaustive search is naive and prohibitively expensive in a post-fabrication tune and test framework. For high-volume parts, tune and test time must be minimized to manage costs. Exhaustive search simply does not achieve economies of scale.

Addressing these fundamental challenges in post-fabrication tuning, we propose an enhanced testing framework. This framework implements the standard test flow, which includes wafer-level test/repair, packaging, and chip-level test/bin. We enhance the standard flow to characterize process variations, to estimate the impact of those variations, and to optimize tunable parameters to best mitigate that impact. Such a framework addresses two fundamental challenges in mitigating process variations. First, post-fabrication tuning is required since the true impact of variations are unknown pre-fabrication. Secondly, intelligent modeling and optimization is required to tractably explore the space of tunable parameters.

The following summarizes the contributions of the proposed framework:

- **Mitigates Process Variations:** The framework implements the state-of-the-art in post-fabrication tuning (e.g., variable latency, voltage interpolation, structure resizing) to mitigate both systematic and random process variations. (Section 2)
- **Fits in Standard Test Flow:** The framework fits into existing test flows, including wafer-level test/repair, packaging, chip-level test/repair. (Section 3)
- **Enhances Standard Test Flow:** The framework characterizes process variations from canary circuit measurements to obtain a *variation fingerprint*. Regression on pre-characterized chips produces *predictive models*, which express the optimization metric,  $BIPS^3/W$ , as an empirically derived function of tunable parameters. Post-fabrication *statistical*

*delay analysis* uses the variation fingerprint to estimate the delay impact of measured variations. Lastly, *constrained optimization* uses the computationally efficient predictive models to explore a large multi-billion point tuning space, identifying optimal per chip configurations subject to delay targets from statistical delay analyses. (Section 4)

- **High Efficiency, Low Cost:** In the presence of large systematic and moderate random variation, the framework delivers 93-96 percent of ideal  $BIPS^3/W$  efficiency while controlling testing costs by using fewer than two rounds of tests. Furthermore, we examine a variety of trade-offs between performance objectives and testing cost. (Section 5)

Collectively, the proposed testing framework effectively mitigates process variations, delivering high performance and power efficiency at low testing cost.

## 2 Motivation and Background

The techniques described in this paper are at the intersection of microarchitectural design and test. As such, this section provides the necessary background in these subareas. We first briefly describe challenges related to variations and several post-fabrication tuning techniques proposed to overcome these challenges.

### 2.1 Process Variations

While process variation exists at several scales (wafer to wafer, die to die), within-die (WID) variation is particularly important for nanoscale technologies. Variations impact device feature sizes and threshold voltages, which cause delay variations. For example, variations lead to a spread in logic gate speed in different parts of a chip. To meet target speed for the overall system, a typical design must accommodate the worst-case portion of the chip by either reducing the frequency or increasing the voltage. Both approaches create extra timing or voltage margins in faster parts of the chip, leading to a less-efficient system overall.

Process variation includes both systematic variations due to lithographic irregularities and random variations due to varying dopant concentrations. Systematic variation exhibits strong spatial correlation among device features for structures located close together. Canary circuits placed near a circuit of interest exploit these correlations yielding insights into a chip’s systematic variations. In contrast, adjacent circuits are uncorrelated for random variations. While it is difficult to directly measure the random variation, statistical timing analysis provides a framework for reasoning about their impact. We combine canary circuits and statistical timing analysis to enable new capabilities in post-fabrication tuning.

This paper uses delay and power data are derived from Hspice circuit simulations at the 32nm technology node using Predictive Technology Models (PTM) [2]. We rely on a Monte-Carlo simulation framework, which is similar to prior approaches [3, 4, 5]. Ideally, we would use real silicon data to verify the proposed scheme. But acquiring silicon data about process variation is extremely difficult because this data is considered secret by many vendors. Thus, we apply Monte-Carlo simulation to model real silicon production. This framework considers both die-to-die (D2D) and within-die (WID) variations, handling systematic variations related to floorplans using a multi-level quad tree method. Recent experimental

results evaluate the accuracy of the quad-tree method, demonstrating errors of 5% relative to fabricated chips [6]. Such error rates are sufficiently low for our architectural studies. We modeled both random and systematic fluctuations at the transistor level. We assume  $\sigma L/L_{nominal} = 7\%$  for gate-length variations and  $\sigma V_{th}/V_{th_{nominal}} = 15\%$  for threshold voltage variations. These assumptions are comparable to the data forecast in prior work [1, 7].

## 2.2 Post-Fabrication Tuning Techniques

Vendors must perform detailed chip testing and tuning to address design uncertainty due to variations. Traditionally, most chip testing is performed directly after fabrication before systems move into the field. In contrast, built-in self-test (BIST) mechanisms perform tests at runtime to test the correctness of circuits under long-term reliability conditions. BIST approaches can be broadly grouped into two distinct approaches: those that check timing constraints at run-time with shadow (double) latches in pipelined combinatorial logic [8, 9], and those that perform runtime BIST checking by periodically taking cores off-line and executing test patterns that are stored close to the CPU core [10]. While both approaches require additional overheads in terms of latch area and power, storage for test vectors, and runtime performance overheads, if aging becomes a major problem in future CMOS technologies vendors may begin to adopt these techniques. In this work, since our focus is on static process variations, we consider tuning mechanisms that are configured at test time to compensate for performance and power penalties from process variations, but our post-fabrication tuning approach would be equally applicable to systems with runtime test capabilities. We defer this aging analysis for future work.

After diagnosing variations at test-time, traditional approaches apply global voltage/frequency tuning to sort fabricated chips into different speed-power bins. However, these global approaches are often too coarse-grained to effectively compensate for within-die variations. To address variations on a finer scale, many researchers have proposed fine-grained post-fabrication tuning techniques to customize individual processors in response to their variation profile.

Device tuning techniques seek to optimize delay by modifying the threshold voltage, supply voltage, or logic delays. For example, adaptive body biasing adjusts the threshold voltage for individual blocks of transistors to mitigate fluctuations in threshold voltages arising from the manufacturing process [11, 12, 13]. Mani, et al., make the case for coordinated optimization of gate sizes at design time and adaptive body bias post-fabrication [14]. “Voltage interpolation” technique provides the ability for different groups of logic gates within a block to switch between two static supply voltages (high and low) to accommodate gates that deviate from nominal delays [15, 16]. In both approaches, devices are partitioned into groups or blocks at design time, but the block level tuning occurs after fabrication and during test.

Several post-fabrication tuning knobs have been proposed at the architectural block level. Sylvester, et al. show the microarchitecture plays an essential role in post-fabrication tuning [17], surveying the many issues related to self-healing systems. We further this prior work by providing the necessary detailed design and testing solutions for post-fabrication tuning in high-end microprocessors. Most microarchitectural knobs tune either latency or resource sizes. Both latency and size tuning seek to reduce worst-case logic delays for blocks that suffer variations. Several researchers have proposed

architectural post-fabrication tuning, extending the latency of long pipelines to accommodate critical path delay outliers [18, 3, 19]. Researchers have also proposed register file resizing [3], cache resizing [20, 21], and criticality-aware prefetching and resource sizing [22].

This paper takes delay as the circuit’s timing performance and latency as the number of stages used to pipeline the circuit. To meet delay targets, we focus on three post-fabrication tuning techniques: variable latency (VL) [16], voltage interpolation (VI) [16], and resizing [20]:

- **Variable Latency (VL):** VL provides two possible latency settings for each architecture unit. If the delay of one unit exceeds the target delay, the latency of that unit can be extended without reducing the global frequency. Extending the latency of architectural blocks often lead to an IPC loss and a detailed study of post-fabrication tuning configurations is required to guarantee a net performance gain when using this technique.
- **Voltage Interpolation (VI):** VI can provide fine-grained voltage tuning for each architecture unit. It can provide effective voltage levels by interpolating two global supply voltages (VddH and VddL). Depending on the number of cuttings of the logic blocks, we can have a different number of “effective” voltage levels, which we refer to the as VI points. These VI points are obtained by cutting logic into multiple voltage domains and assigning each domain to a low or high voltage. A VI point is a particular combination of low and high voltages across these domains. For example, a logic block divided in  $D$  voltage domains will have at most  $2^D$  unique VI points, which give the overall logic block an effective voltage achieved through interpolation across the  $D$  domains.
- **Resizing:** Resizing adjusts array sizes of key microarchitectural structures, which significantly impact system performance. If the delay cannot fit into the target frequency, we can reduce the array size by turning off part of the array that operates at a slow speed. This technique should be applied with caution since the size of key architecture queues is very important to the system performance. This technique trades off IPC with target frequency.

A common theme across the space of post-fabrication tuning schemes is an exacerbation of testing challenges. All techniques, in effect, require a per-chip customization of various resources at microarchitectural block granularities. Although many of the schemes appear promising, advances in testing strategies will be necessary to enable these schemes in practice.

### 3 Standard Test Flow

The standard test flow for commodity microprocessors fulfills a variety of important objectives. We define the test flow as the sequence of steps from wafer fabrication to product shipment. A variety of tests are conducted at both wafer and packaged-chip levels, stuck-at fault checks, IDDQ measurements, at-speed functional tests, AC scan, etc. [23]. Test time directly translates into cost and, consequently, any post-fabrication tuning technique must minimally impact the test flow. This subsection provides a basic overview of a test flow for commodity microprocessors to show how our proposed testing strategy for post-fabrication tuning fits into the overall flow.

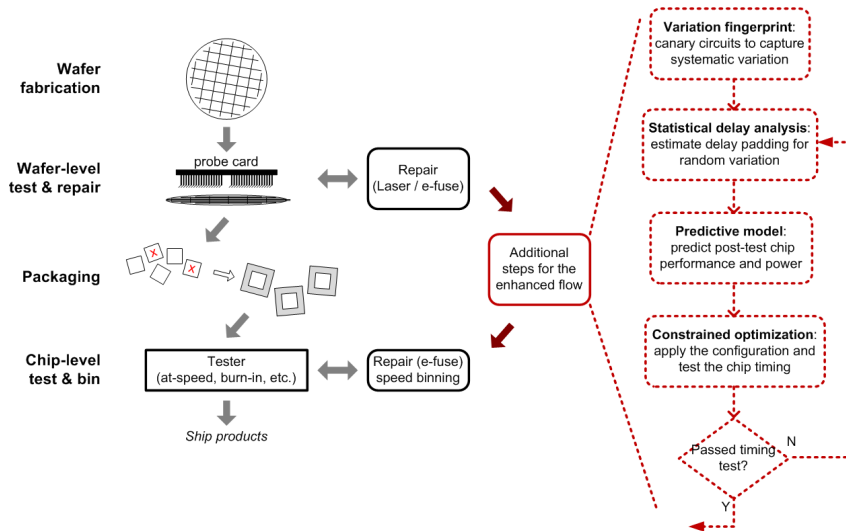


Figure 1: Illustration of standard microprocessor test flow with additional steps for proposed post-fabrication tuning.

Figure 1 presents a simple illustration of a standard test flow for microprocessors. Preliminary tests at the wafer level pre-screen defective parts early in the flow. After wafer-level test and sort, wafers are sent to the assembly house to be diced up and only parts that pass wafer-level tests are packaged. We assume a relatively long latency between chip packaging and the next test phase, especially if the fab and assembly houses are not co-located. Packaged parts are put through another round of rigorous tests and binning before they are shipped. Additional steps in the flow added for post-fab tuning, identified on the right, are described in Section 4.

**Wafer-Level Test and Repair.** Once a wafer comes out of the fab, wafer-level tests are performed by powering up and driving signals to the device under test via tips on a probe card. For example, built-in self test checking of on-chip memories are often run at the wafer level to identify bad bits that are flagged to be corrected by row and column redundancy circuits. Such memory array fixes are implemented via laser-cut fuses and/or on-chip electrical fuses (e-fuses). More generally these fuse technologies enable per-chip customization during post-fabrication tuning. Laser fuses offer simpler on-chip circuitry, but impose minimum area overheads for each fuse sight due to laser spot-size limitations. In contrast, e-fuses require additional circuitry for addressing and to set the fuses. However, e-fuses can be smaller and entirely embedded within chips. Set after packaging, these fuses offer an additional degree of tuning flexibility. In this paper, we assume that e-fuses are available for standard test flows and for post-fabrication tuning with low overheads.

**Packaging.** After wafer sort, bad parts are flagged (X in the figure) and good parts are sent for packaging. Fabrication and wafer-level test is often geographically separated from packaging and package-level test. For example, fabrication might occur in North America while the assembly house is located in Asia. As a result, long latencies between wafer-level and package-level tests are possible. If needed, these long latencies might be exploited for more thorough wafer or chip characterization, modeling, and optimization.

**Chip-Level Test and Bin.** After packaging, chips go through another round of rigorous tests to further sort out bad



chips. Since packaged chips encompass the environmental conditions (thermal and electrical) seen by the chips out in the field, extensive at-speed and burn-in tests are often run at this point. Again, modifications to the chips can be made via e-fuses, which can include settings for array fixes, clock distribution tuning [24], etc. In addition to package sort, chips are tested for binning such that higher performance (or more energy-efficient) parts can be sold at a price premium.

Cost is a critical component of the test flow as it directly affects the bottom line. As a device progresses further along the flow, it consumes increasing investments in terms of time and capital expense, making it more costly to throw out. Parts that pass wafer sort incur the added cost of packaging and further tests. A device that fails out in the field represents the worst-case scenario due to penalties for replacing them and damage to the manufacturer’s reputation. However, testing is expensive and test times must be minimized to preserve profit margins.

## 4 Enhanced Test Flow

Due to rising process variability in modern nanoscale process technologies, several post-fabrication tuning techniques have emerged as potential solutions to maximize performance and energy efficiency. However, for such techniques to be viable, they require low-cost test solutions that efficiently set tuning knobs without incurring large overheads. This section describes a generally-applicable testing framework for post-fabrication tuning with minimal impact on testing latency and time. Referring back to Figure 1, the right-most sequence of steps illustrate the additions made to the standard test flow. The proposed scheme relies on scan-enabled on-chip process monitoring circuits, also known as canary circuits, scattered throughout the chip to provide a “fingerprint” of on-chip process variability. Based on the variation fingerprint, we statistically estimate the true critical path delay. This estimated critical path delay is combined with regression models to predict and optimize performance and power responses as a function of various tunable parameters. We then apply the parameters identified optimal by regression models to chips and test to verify their timing behavior. If a chip passes the delay test, regression-predicted optima for tuning knobs are applied to the packaged part for shipment.

### 4.1 Variation Fingerprint

The variation fingerprint of a chip characterizes its systematic variations using measurements of canary circuits.<sup>1</sup> Canary circuits are on-chip process monitoring circuits, commonly used to profile chip characteristics. Since canary circuits are physically located next to the critical paths of an architectural block, the effects of systematic variations on canary circuits and the average path are highly correlated. Ideally, delay measurements from canary circuits will perfectly match the average path delay. The design and use of canary circuits is an area of ongoing research. Without loss of generality, we use ring oscillators to demonstrate our proposed framework. A ring oscillator consists of a loop of inverters where the number of inverters is determined by logic depth. For example, a pipelined logic block with 30 FO4 delays per stage will require a ring oscillator with 15 inverters each of which drives an inverter sized 4x larger. Although we use ring oscillators to capture

---

<sup>1</sup>Canary circuits are named after canaries in coalmines that helped determine the amount of breathable air in a coalmine. Thus, canaries are a metaphor for tools providing early measurement and detection.

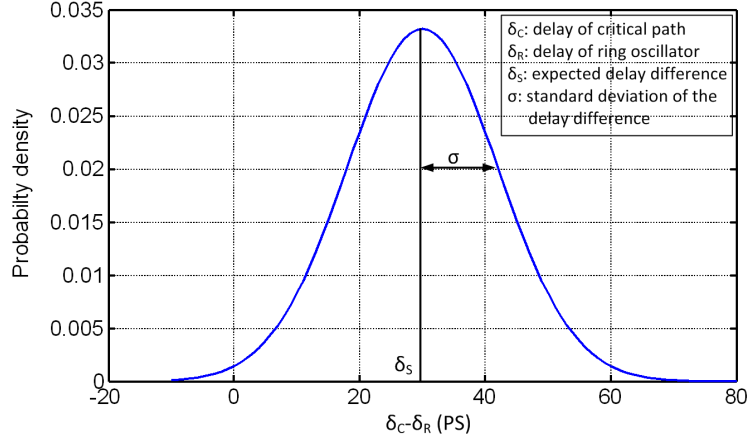


Figure 2: The PDF plot of the delay difference between the ring oscillator and critical path delay in the instruction decoder where  $\delta_R - \delta_C \sim N(\delta_S \approx 30ps, \sigma \approx 12ps)$ , a representative architectural block.

systematic variations, our methodology does not depend on this particular canary circuit and other canary circuits can be used, for example, for dynamic logics.

In practice, canary delays are imperfect proxies of path delays since canary circuits only capture systematic variations and do not account for random variations. These random variations will determine measurement differences between the canary circuit and the average path. Furthermore, canary circuits are representative of the *average* path and not the *critical* path. To illustrate this difference, we consider a hypothetical scenario where we know the delay for both ring oscillators and critical paths. To quantify delay differences, we implement a Monte Carlo experiment with one thousand instances of the instruction decoder, a logic-dominated architectural block, each instance with a slightly different oscillator and critical path delay due to process variations.

Figure 2 illustrates the distribution of delay differences between the oscillator delay ( $\delta_R$ ) and the worst-case critical path delay ( $\delta_C$ ) in the decoder block from Monte Carlo simulations. The delay difference follows approximately a Normal distribution with mean  $\delta_S$  and standard deviation  $\sigma$ . Intuitively,  $\delta_S$  quantifies the expected difference between the measured ring oscillator delay and true critical path delay ( $\delta_S = E[\delta_R - \delta_C]$ ). If we consider a large number of paths through a logic block, the expected difference between measured ring oscillator delay and true *average* path delay is zero. But for a fully synchronous digital system, the true *critical path* delay determines the operating frequency. Given this difference between average and critical path delays and noting that ring oscillators are designed to capture only average delays, critical path delays are estimated from ring oscillators with an additional delay shift  $\delta_S$  that accounts for this difference.

The standard deviation  $\sigma$  captures variation in differences between measured oscillator delay and true worst-case critical path delay. This  $\sigma$  is primarily due to random variation on the critical paths, which produces a delay distribution for fabricated chips. We apply statistical analysis based on this distribution to estimate the worst-case critical path delay of each architecture block. In practice, we would fully pre-characterize  $\delta_S$  and  $\sigma$  by measuring an initial lot of fabricated chips

during wafer-level test and repair as described in Figure 1. In this work, we assume 1,000 chips are pre-characterized early in the production cycle as the product ramps up. The overhead of measuring 1,000 sample chips is relatively low compared with the future mass production of millions of chips. This characterization might be refined as more chips become available for measurement.

## 4.2 Statistical Delay Analysis

In contrast to statistical timing analysis at design time, our testing framework implements post-fabrication statistical delay analysis. A post-fabrication test defines a target for critical path delay and identifies all chips that satisfy that target. Because worst-case critical path delay is unknown, the testing methodology must rely on a combination of a measured ring oscillator delay ( $\delta_R$ ), the expected difference between measured oscillator and true critical path delays ( $\delta_S = E[\delta_R - \delta_C]$ ), and extra delay padding ( $\delta_P$ ) that provides an error margin to the estimate of critical path delay. Padding  $\delta_P$  is needed since tests are evaluated with measured oscillator delays that may not accurately capture true critical path delay. Thus, the estimated critical path delay for a microarchitectural block is given by  $\hat{\delta}_C = \delta_R + \delta_S + \delta_P$  and the estimate is made more conservative by increasing  $\delta_P$ .  $\hat{\delta}_C$  is an estimate for the true delay  $\delta_C$ .

Post-fabrication delay tests are evaluated with estimated worst-case critical path delay  $\hat{\delta}_C$  but results may differ if tests were evaluated with true critical path delay  $\delta_C$ . We define the *block pass rate* ( $PR_{block}$ ) as the number of blocks that pass the same test under  $\hat{\delta}_C$ . Intuitively, the pass rate is a measure of confidence in the estimate  $\hat{\delta}_C$ . A high pass rate means  $\hat{\delta}_C$  is a conservative estimate and is likely at least as large as  $\delta_C$ . If  $\hat{\delta}_C \geq \delta_C$ , then any tuning configuration that satisfies the delay test for  $\hat{\delta}_C$  (a more difficult constraint) will also satisfy the delay test for  $\delta_C$  (a less difficult constraint) leading to a high pass rate.

To further explore this relationship between *block pass rate* and average chip  $BIPS^3/W$  efficiency, we consider a range of *chip pass rates* ( $PR_{chip}$ ) and define delay tests such that the pass rate is achieved. We translate chip pass rate to block pass rate using elementary probability theory for  $B$  blocks:  $(PR_{block})^\alpha = PR_{chip}$ . If  $B$  microarchitectural block delays are perfectly correlated, the chip-level pass rate must be satisfied by every block and  $\alpha = 1$ . If the  $B$  microarchitectural block delays are completely independent, each of the  $B$  blocks must have pass rate of  $(PR_{chip})^{1/B}$  and  $\alpha = B$ . In practice, blocks are neither fully correlated nor fully independent, implying  $1 \leq \alpha \leq B$ . In the absence of any variation, all blocks are perfectly correlated by design. Systematic variations reduce this correlation at coarse granularity and random variations reduce this correlation at fine granularity. The exact impact on correlation must be quantified by measurement or simulation. As described in Section 4.4, we empirically determine the proper value  $\hat{\alpha}$ , our best estimate of the true  $\alpha$ .

$$PR_{block} = (PR_{chip})^{1/\hat{\alpha}} \quad (1)$$

$$\delta_P = Q(PR_{block}) \times \sigma \quad (2)$$

Given  $\hat{\alpha}$ , we compute the block pass rate from the desired chip pass rate as shown in Equation 1. The block pass rate, in turn, defines the block-level delay padding as show in Equation 2. Specifically, we compute a particular quantile of the

Configuration Parameter	Value	Configuration Parameter	Value
Issue Width	4 instructions	Issue Queues	20-entry INT, 15-entry FP
Load Queue	32-entries	Store Queue	32-entries
Reorder Buffer	80-entry	I-Cache, D-cache	64KB, 4-way Set Associative
Instruction TLB	128-entry Fully-Associative	Data TLB	128-entry Fully-Associative
Integer Functional Units	4 FUs	Floating Point Functional Units	2 FUs
L2 Cache	2MB 4-way	Branch Predictor	21264 Tournament Predictor

Table 1: Baseline processor configuration.

Arch unit	Latency choices	Array size choices	VI points
DEC	3,4-cycle	-	0.8:0.02:1.2
MAP	3,4-cycle	32, 64, 96, 128	0.8:0.02:1.2
RF	3,4-cycle	32, 64, 96, 128	0.8:0.02:1.2
IQ	3,4-cycle	10, 20, 30, 40	0.8:0.02:1.2
FXU	3,4-cycle	-	0.8:0.02:1.2
FPU	4,5-cycle	-	0.8:0.02:1.2

Table 2: Post-fab tuning knobs.

block’s oscillator delay distribution and multiply by its standard deviation. For example, if we wish to achieve a block pass rate  $PR_{block} = 0.997$ , then  $Q(PR_{block}) = 3$ , since 99.7% of the probability distribution for estimates of critical path delay is located below  $\delta_S + 3\sigma$  (shown in the normal distribution of Figure 2). Both the quantile function  $Q$  and the standard deviation  $\sigma$  are known once we pre-characterize 1,000 chips to get a variation fingerprint during wafer-level test and repair as described in Figure 1.

### 4.3 Predictive Model

We consider a space of configurations achievable by post-fabrication tuning that is defined by six microarchitectural parameters and three tuning techniques. The space contains hundreds of billions of possible configurations, which exhibit significant diversity in performance and power. To tractably identify optimal configurations for each chip, we must construct predictive models to capture the relationship between performance, power, and tuning parameters. Computationally efficient predictive models enable comprehensive optimization across the large configuration space.

We assume a baseline processor, which is comparable to the Alpha 21264, with parameters listed in Table 1. Our framework is not specific to this configuration space and might be applied more generally. To control the number of simulations in this work, we use 8 SPEC2000 benchmarks and rely on Sim-Point for sampling [25]. Phansalkar et al. show that 8 benchmarks (*crafty*, *applu*, *fma3d*, *gcc*, *gzip*, *mcf*, *mesa*, *twolf*) can adequately represent the entire SPEC2000 benchmark suite [26]. For each benchmark, 100 million instructions are simulated after fast forwarding to specific checkpoints. In this paper, single number results of power or performance correspond to the harmonic mean of all simulated benchmarks.

We study six architectural units: instruction decoder (DEC), register rename table (MAP), physical register file (RF), issue queue (IQ), fixed-point unit (FXU), and floating-point unit (FPU). We apply three post-fabrication tuning techniques to

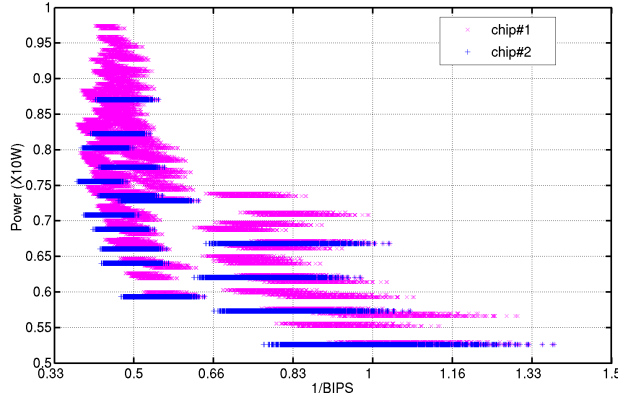


Figure 3: Power-BIPS for different settings for two chips.

these structures: voltage interpolation (VI), variable latency (VL), and structural resizing. The possible tuning parameters are listed in Table 2. With voltage interpolation, every unit can have 20 effective voltage settings from 0.8V to 1.2V in increments of 0.02V. These 20 settings are derived from 20 corresponding combinations of high-low voltage settings. With variable latency, every unit can have two latency choices. Array structures take one of four sizes. Additionally, we allow eight frequency choices for each chip. This yields a large design space of 282 billion post-fabrication configurations, which cannot be exhaustively evaluated in simulation or measurement. Again, our framework is not specific to this configuration space and might be applied more generally.

Figure 3 illustrates the power and performance characteristics for two chips experiencing different process variations. For each chip, the figure considers a representative subset of the 240 billion configurations. Even for this reduced subset of configurations, we find different configurations produce very different power and performance values. Across these configurations, we find performance ranges between 0.39 and 1.39 BIPS, a factor of  $3.6\times$ . Similarly, we find power ranges between 0.53 and 0.98, a factor of  $1.86\times$ . Given the size of the configuration space and observing that each chip might have different optimal configurations, an exhaustive exploration of these post-fab tuning configurations is intractable and we require more computationally efficient methods.

Techniques in statistical inference reveal performance and power trends from sparsely measured configuration samples, enabling performance and power analysis for much larger, comprehensive tuning spaces. In particular, we apply spline-based regression models, which predict a performance or power response as a function of design parameter values [27]. Within this framework, interactions between predictors are captured by products terms specified in the models' functional form using domain-specific knowledge. For example, cache sizes for adjacent levels in the memory hierarchy should interact (i.e., the optimal L1 cache size depends on the L2 cache size and vice versa, thereby requiring joint optimization). Non-linearity is captured by cubic spline (ie. piecewise polynomial models) transformations on the predictors. Given sparsely measured configurations from the space, a multi-dimensional curve fit is performed to capture the relationships between

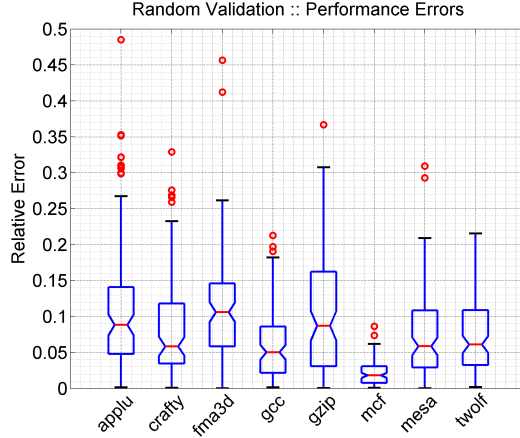


Figure 4: Distribution of prediction errors for 100 random validation configurations.

design metrics of interest and tunable parameters. Model construction is computationally efficient and may be reduced to a series of cubic transformations followed by a linear solve (highly-optimized matrix operations). Model evaluation, expressed as matrix multiplication, is also highly efficient. Hundreds or thousands of predictions per second are possible. This computational efficiency allows us to tractably explore a large space of microarchitectural structural, voltage, and latency configurations.

We train regression models with 500 configurations sampled uniformly at random from the space of 282 billion points. Such a sparse sampling is used to construct unbiased models that weight every part of the configuration space equally. These training configurations are measured at the beginning of a production cycle, incurring the one time cost of constructing these models. Figure 4 illustrates model accuracy when validated against simulation for 100 randomly selected and separate validation points, demonstrating median errors of 7.4 percent for performance. Similar models have been applied to and validated for practical design studies, demonstrating accuracy sufficient for early stage design optimization [28]. This prior work also found outlier errors tend to occur near design space boundaries where models are extrapolating instead of interpolating. Since joint performance and power optimization typically identifies optima well within space boundaries, outliers are unlikely to significantly affect our analysis. However, outliers might be mitigated with more advanced sampling techniques. For example, Latin Hypercube sampling ensures evenly spaced samples and weighted sampling emphasizes training configurations from regions of the space modeled less accurately. To further reduce error, we might continuously refine these models as chip manufacturing provides additional training samples across production cycles.

Our enhanced test flow relies on these efficient predictive models to capture the relationship between  $BIPS^3/W$  efficiency and post-fab tuning parameters. Since test time and costs have a direct impact on profit margins, statistical inference and other modeling methodologies are imperative. The computational efficiency of regression models enables previously intractable modeling and optimization for the space of structure sizes, latencies, and voltages. The results of regression model optimization determine the configuration applied in post-fabrication tuning.

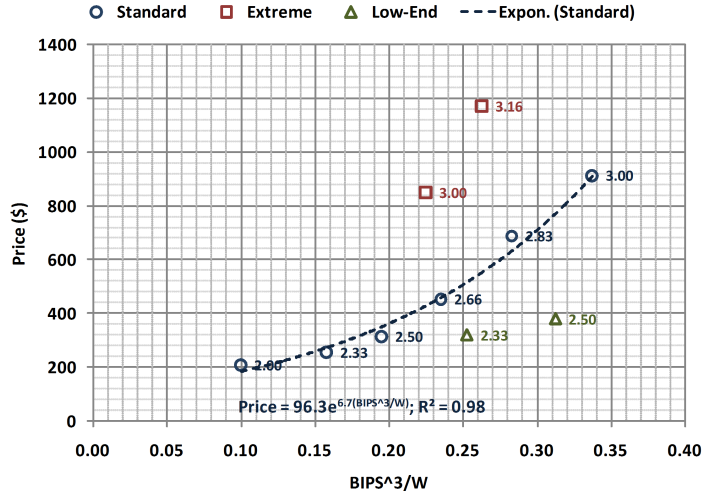


Figure 5:  $BIPS^3/W$  and price for representative Intel Xeon processors (12MB L2, 1333MHz FSB, 45nm) [30]. Data series annotated with products’ frequencies in GHz.

#### 4.4 Constrained Optimization

Constrained  $BIPS^3/W$  optimization identifies optimal tuning configurations. Delay constraints are provided from analyses of Section 4.2 while  $BIPS^3/W$  estimates are provided by predictive models of Section 4.3.

**Optimization Objective.** Current microprocessors are binned with respect to maximum achievable clock speed or power efficiency as power often constrains performance in modern designs. Given the direct trade-off between frequency and power, we differentiate chips via the  $BIPS^3/W$  metric [29]. Derived from the cubic relationship between power and voltage/frequency, this metric is voltage and frequency invariant. However,  $BIPS^3/W$  is sensitive to parameters in the post-fabrication tuning techniques studied throughout this work. In the absence of detailed cost and pricing models, which are typically closely guarded industrial secrets, we use  $BIPS^3/W$  as a proxy for price and evaluate the proposed framework with respect to trade-offs between  $BIPS^3/W$ , yield and test time.

Published product data sets, although lacking detailed price and performance components, suggest a strong relationship between  $BIPS^3/W$  and price. For example, Figure 5 plots price against  $BIPS^3/W$  as reported by data sheets for server class processors. The figure illustrates ten frequency bins for a particular server product and the price differentiation across these bins. This data captures only the frequency contribution to  $BIPS$  and its associated power cost. Despite analyzing this subset of  $BIPS$  contributors (other contributors include latency, structure sizes), we observe material relationships between  $BIPS^3/W$  and price. An exponential relationship is fit very closely to ( $R^2=0.98$ ) to six of ten *standard* products. We would expect similar relationships if latency and size contributions to  $BIPS$  were included. Thus, we take  $BIPS^3/W$  as our proxy for price without loss of generality. Price is likely a function of  $BIPS^3/W$  and, although we illustrate an exponential function, the exact function is orthogonal and independent of the proposed methodology. Any other price function might be used in our testing framework.

Four of ten products are outliers, which Intel classifies for different market segments. Intel grades the two products above the exponential trend as *extreme* processors for their high frequency and significant power costs. Despite their low  $BIPS^3/W$ , these processors are sold at much higher prices. Similarly Intel grades the two products below the trend as *low-end* processors for their low frequency and exceptionally low power costs. Despite their high  $BIPS^3/W$ , these processors are sold at much lower prices. Thus, although most processors exhibit an exponential relationship between  $BIPS^3/W$  and price, the current market for microprocessors prefers high performance over low power. In such a market, processors with high frequency and exceptionally high power costs are sold at much higher prices than processors with low frequency and exceptionally low power costs. As power becomes increasingly important in the microprocessor market, prices of the four outliers will likely gravitate toward the exponential trend.

**Optimization Constraints.** The framework maximizes  $BIPS^3/W$  of the overall chip while ensuring each architectural block satisfies the constraint of Equation 4. The estimated critical path delay consists of three components. Ring oscillator  $\delta_R$  is measured and known for each block. The delay shift  $\delta_S$  is pre-characterized for a small number of chips and is included in the variation fingerprint. Lastly,  $\delta_P$  is delay padding computed for a desired block pass rate from Equation 2. For each block, the estimated critical path delay  $\hat{\delta}_C$  is constrained to be less than the delay of the tuning configuration. A block's delay is a function of its configuration, which includes configured voltage  $V_{cfg}$  relative to some nominal voltage  $V_{nom}$ , configured latency  $L_{cfg}$ , and configured frequency  $f_{cfg}$ .  $V_{cfg}$  affects the constraint as a higher voltage  $V_{cfg}$  reduces critical path delay and allows the constraint to be more easily satisfied.

$$\hat{\delta}_C = \delta_R + \delta_S + \delta_P \quad (3)$$

$$\hat{\delta}_C \times \frac{V_{nom}}{V_{cfg}} \leq \frac{L_{cfg}}{f_{cfg}} \quad (4)$$

The space of post-fabrication tuning configurations is defined by combinations of  $V_{cfg}$ ,  $L_{cfg}$ ,  $f_{cfg}$ . The optimization relies heavily on the computational efficiency of our predictive regression models. We exhaustively evaluate regression equations to predict the performance and power efficiency of every configuration. We repeat this optimization for every chip, obtaining chip-specific measurements for  $\delta_R$  and identifying chip-specific optimal values for  $V_{cfg}$ ,  $L_{cfg}$ ,  $f_{cfg}$ .

**Calibrated Optimization.** Recall the analysis of Section 4.2 assumes an empirically derived  $\hat{\alpha}$ . To empirically determine the measure of block-level correlation  $\hat{\alpha}$ , we repeat the above optimization for varying values of  $\alpha$ ,  $1 \leq \alpha \leq B$ . For each value of  $\alpha$ , we characterize pass rates using the estimated critical path delay  $\hat{\delta}_C$  and the true critical path delay  $\delta_C$ . The empirically derived  $\hat{\alpha}$  is chosen such that pass rates are equal for both analyses. This empirical calibration effectively identifies the degree to which blocks are correlated or independent. The calibration of  $\hat{\alpha}$  is a one-time cost, requiring detailed measurements of true critical paths for 1,000 sample chips early in the manufacturing process.

The chip pass rate directly influences the performance of chips passing the delay test. Figure 6 quantifies this relationship, plotting  $BIPS^3/W$  efficiency against chip pass rate. A high pass rate leads to lower average efficiency since the high pass rate is achieved by more delay padding  $\delta_P$  and conservative estimates of  $\hat{\delta}_C$ , which effectively increase the delay a



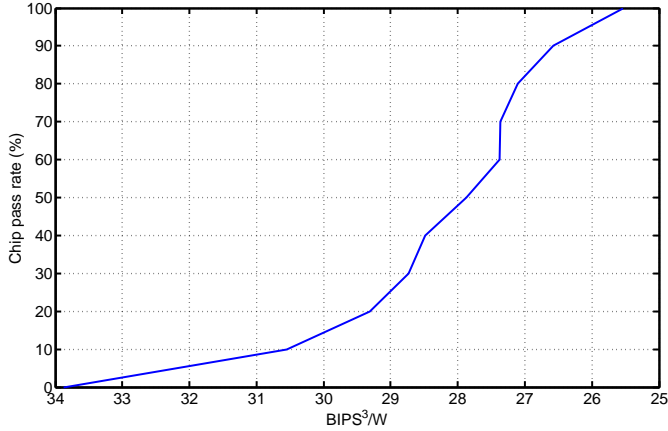


Figure 6: The relationship between average  $BIPS^3/W$  and the chip pass rate.

chip can deliver and still pass. Searching the space of post-fabrication tuning configurations under such conservative delay estimates will produce configurations with higher latencies, lower frequencies, and higher voltages. The net effect is lower efficiency. In contrast, if we consider a low pass rate, less delay padding  $\delta_P$  is required and more  $BIPS^3/W$  efficient configurations are identified. However,  $\hat{\delta}_C$  is a less conservative estimate, which increases estimation error ( $\hat{\delta}_C - \delta_C$ ) and causes more chips fail delay tests than when evaluated under the true critical path delay  $\delta_C$ .

## 5 Analysis and Evaluation

We evaluate the proposed testing framework by assessing trade-offs the delivered  $BIPS^3/W$  and various measures of testing cost, including the number of tests and the density of canary circuits. We find our framework robust against higher systematic variation, fewer tunable knobs, and differences across workloads. Lastly, we evaluate our post-fabrication tuning framework with several testing and tuning heuristics, finding fundamental trade-offs between  $BIPS^3/W$  and testing cost.

### 5.1 Tuning with Multiple Tests

There is an inherent trade-off between  $BIPS^3/W$  and pass rate for a single test iteration. As shown in Figure 6, the pass rate directly influences the average  $BIPS^3/W$  efficiency of chips passing the test. Average efficiency decreases with increasing pass rates. Further exploring the relationship between pass rate and average efficiency, we consider multiple tests and their ability to deliver greater efficiency. Multiple tests stratify chips by their  $BIPS^3/W$  efficiency to improve average efficiency. Suppose, for example, we implement two tests. The first test is defined to achieve a low pass rate. Although a small fraction of chips pass this first test, each passing chip will achieve high  $BIPS^3/W$  efficiency as shown in Figure 6. The second test is defined to achieve a high pass rate, effectively a catch-all for chips that fail the first test. Due to this higher second pass rate, chips that fail the first but pass the second test will achieve on average lower  $BIPS^3/W$  than those that

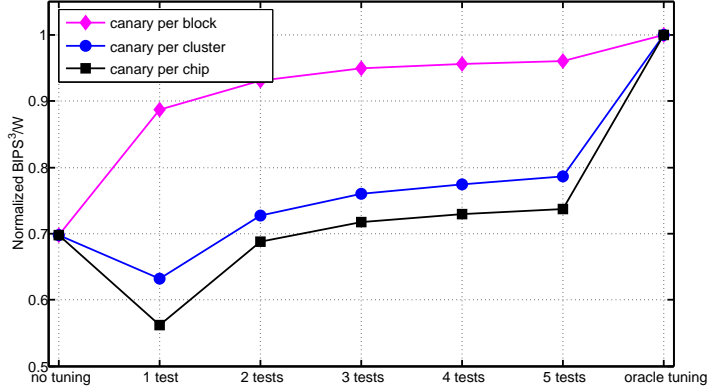


Figure 7: The relationship between the averaged  $BIPS^3/W$  and the number of tests.

pass the first test.

Given this intuitive understanding of multiple tests, we consider a range of test counts and assess its impact on delivered efficiency. Figure 7 illustrates efficiency trends as the number of tests increases. We consider a continuum between *no-tuning* and *oracle-tuning*. No-tuning is the baseline method which does not implement post-fabrication tuning techniques for architecture flexibility, voltage interpolation, and variable latency. Thus, in the no-tuning case, the operating frequency is defined by the slowest critical path on the chip. In contrast, oracle-tuning assumes the true critical path is estimated perfectly with no error. Given this perfect estimate where  $\hat{\delta}_C = \delta_C$ , the enhanced test flow fully utilizes architectural flexibility, voltage interpolation, and variable latency to maximize  $BIPS^3/W$  efficiency while guaranteeing constraints for the true critical path delay are satisfied. The efficiency results of oracle-tuning are not achievable in practice since it would require perfect knowledge of each architectural block’s critical path.

To explore intermediate values between no-tuning and oracle-tuning, we consider varying test counts between one and five. We assume a final pass rate of 100 percent, which means all chips must eventually pass a delay test and the total pass rate across  $T$  tests must sum to 100 percent:  $\sum_{t=1}^T PR_t = 100$ . If only one test is used ( $T = 1$ ),  $PR_1 = 100$  and passing chips will achieve low average efficiency. If multiple tests are used ( $T > 1$ ), we explore all combinations of pass rates that satisfy  $\sum_{t=1}^T PR_t = 100$  to identify the combination that maximizes average efficiency. For example, if  $T = 3$ , we examine all combinations of  $(PR_1, PR_2, PR_3)$  to maximize efficiency. Thus, our analysis considers the best achievable efficiency for each test count.

The diamond line (canary-per-block) of Figure 7 illustrates efficiency trends as the number of tests varies. No-tuning is 30 percent less efficient than oracle-tuning, indicating significant efficiency gains are possible from post-fabrication tuning. Thus, we demonstrate the importance of adding these tuning capabilities to mitigate process variations. With these capabilities, even a modest number of tests drastically improves delivered efficiency. A single test improves efficiency by 1.27x, increasing normalized efficiency from 70 to 89 percent of oracle-tuning. Efficiency increases monotonically with the

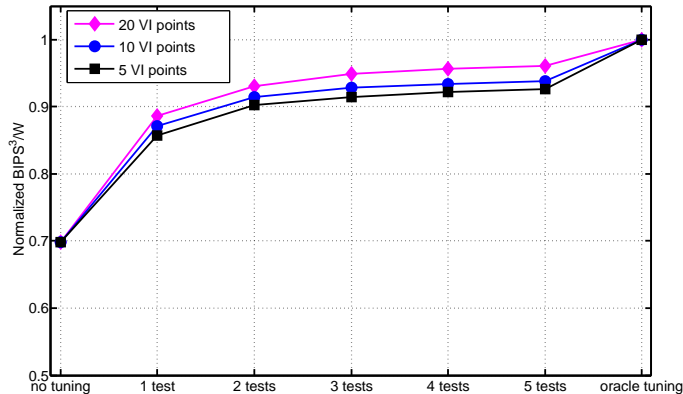


Figure 8: The relationship between the averaged  $BIPS^3/W$  and number of VI points.

number of tests. However, we observe diminishing marginal returns in efficiency. Two post-fabrication tests are sufficient to achieve 93 percent of oracle-tuning whereas five post-fabrication tests achieve 96 percent. Overall, this analysis shows the effectiveness of using canary circuits to guide post-fabrication tuning. These canary circuits characterize the variation fingerprint by capturing systematic variations and are the basis for our statistical estimates of critical path delay.

## 5.2 Canary Circuit Density

The efficiency gains from our enhanced test flow are driven by canary circuits and their characterization of variation fingerprints. The effectiveness of this characterization depends on the density of canary circuits. Additional canary circuits would provide more detailed information about systematic variations simply by increasing the granularity of on-chip delay measurements. Furthermore, these additional canary circuits would further the characterization of random variations. If too few canary circuits are used, systematic variation is imperfectly characterized and might be mistaken for random variation. Treating systematic variation as random variation directly reduces our confidence in our critical path delay estimates by increasing the standard deviation  $\sigma$  in Figure 2. These estimation errors, in turn, will impact the test flow’s ability to correctly identify optimal post-fabrication tuning configurations. Despite the potential benefits of additional canary circuits, we must ensure these circuits are used judiciously to control measurement times in the test flow.

Figure 7 illustrates efficiency trends under different canary circuit (e.g., ring oscillator) densities. We consider three scenarios in order of decreasing canary density: (1) canary-per-block, (2) canary-per-cluster, and (3) canary-per-chip. The canary-per-block scenario is the baseline scenario considered in the previous analysis, illustrating trends for the greatest canary density where each of the six architectural blocks contains a ring-oscillator. For comparison, we define a cluster of three architectural blocks and consider a canary-per-cluster scenario where the three blocks share a single ring oscillator. In this scenario, the test flow attempts to capture systematic variations for the three blocks using a single measurement. Similarly, we also consider a canary-per-chip where all six architectural blocks share a single-ring oscillator. Thus, we

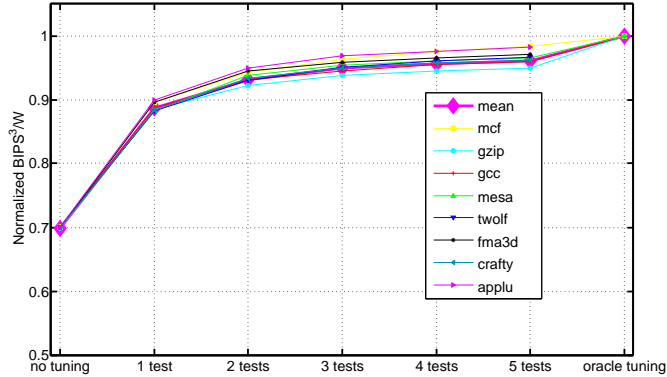


Figure 9: The relationship between the averaged  $BIPS^3/W$  and different benchmarks.

explore trade-offs between fewer ring oscillator measurements and  $BIPS^3/W$  efficiency.

As shown in Figure 7, canary density significantly impacts achieved efficiency from our test flow. Post-fabrication tuning is less effective when fewer ring oscillators are available. In particular, one post-fabrication test is worse than no-tuning under canary-per-cluster or canary-per-chip scenarios. These scenarios deliver only 63 and 71 percent of canary-per-block efficiency. Canary circuits are designed to capture the impact of process variations for a small, localized on-chip area, but the canary-per-cluster and canary-per-chip scenarios attempt to generalize variation estimates from these localized measurements to much larger cluster or chip areas. This generalization is ineffective, providing a misleading variation fingerprint that leads to a failure of testing and optimization schemes. Efforts to identify optimal chip-level post-fabrication tuning configurations from a small number of localized measurements leads to sub-optimal configurations. Although efficiency increases with more tests, canary-per-cluster and canary-per-chip scenarios are disadvantaged because they perform these additional tests with an incomplete characterization of systematic variation that also leads to inaccurate estimates of random variations.

### 5.3 VI Points and Benchmark Sensitivity

We assess the sensitivity our enhanced test flow to the resolution of our tuning configuration space. In particular, Figure 8 plots efficiency trends for a varying number of voltage interpolation (VI) points. These VI points are obtained by cutting logic into multiple voltage domains and assigning each domain to a low or high voltage. A VI point is a particular combination of low and high voltages across these domains as described in Section 2.2. Figure 8 suggests our test flow is insensitive to the number of VI points; 5 VI points per block deliver efficiency within 5 percent of 20 VI points per block. Considering the significant design complexity and test overhead of additional VI points, we conclude 5 VI points are sufficient for our enhanced test flow. This modest number of required VI points improves the speed and efficiency of tuning and optimization by reducing the size of the configuration space.

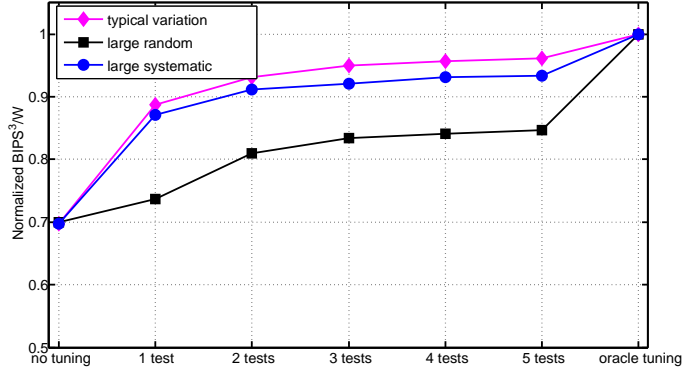


Figure 10: The relationship between the averaged  $BIPS^3/W$  and the variations.

To further demonstrate the general applicability of our enhanced test flow, we assess the sensitivity of efficiency gains to choice of benchmark. Figure 9 illustrates negligible efficiency differences across the benchmark suite with similar trends as the number of tests increases. Thus, our enhanced test flow is application insensitive and might be considered a general strategy for microprocessor testing.

#### 5.4 Sensitivity to Process Variation

Figure 10 illustrates the impact of process variations on our enhanced test flow. We consider three scenarios: (1) typical variation, (2) large random, and (3) large systematic. Typical variation assumes gate length coefficient of variation  $\sigma L/L_{nom} = 7\%$  and threshold voltage coefficient of variation  $\sigma V_{th}/V_{th_{nom}} = 15\%$ . Large random variation considers greater gate length variation with  $\sigma L/L_{nom} = 14\%$  and large systematic variation considers greater threshold voltage variation with  $\sigma V_{th}/V_{th_{nom}} = 30\%$ . This analysis considers canary-per-block measurements and assesses the impact of greater variations on our test flow.

As shown in Figure 10, the enhanced test flow is more sensitive to random variations. Under a canary-per-block scenario, ring oscillators effectively capture the effects of large systematic variations and our test flow delivers efficiency comparable to that delivered under typical variation. We observe negligible efficiency differences between two and three percent when we consider greater systematic variation. In contrast, we observe significant efficiency losses under large random variation. Ring oscillators cannot capture increased random variation, which increases errors in estimates of critical path delay by increasing the spread  $\sigma$  in Figure 2. As the spread between canary-derived estimates and true critical path delays increases, our test flow requires greater delay paddings  $\delta_P$  to guarantee desired chip-level pass rates. These paddings reduce average  $BIPS^3/W$ , thereby hindering our tuning strategy.

## 5.5 Alternative Test Schemes

Given the sensitivity of our enhanced test flow to large random variations, we propose and evaluate several test schemes and attempt to mitigate these large random variations. In particular, we evaluate the following five methods with large gate length variations with  $\sigma L/L_{nom} = 14\%$  and canary-per-block measurements.

- **Method 1 (No Tuning):** The baseline method which does not implement post-fabrication tuning techniques for architecture resizing, voltage interpolation, and variable latency. Operating frequency is defined by the slowest critical path on the chip and is identified by beginning at the lowest frequency and progressively increasing frequency until the delay test is failed.
- **Method 2 (VI with Relaxed Latency/Size):** Optimizes only voltage while fixing latencies and sizes to *relaxed* values. These relaxed values are longer latencies and larger sizes. Voltage optimization, implemented with voltage interpolation (VI), begins at the lowest effective voltage and progressively increases effective voltage until delay test is passed.
- **Method 3 (VI with Popular Latency/Size + Method 2):** Optimizes only voltage while fixing latencies and sizes to *popular* values. These popular values are found effective for a representative number of pre-characterized chips early in the manufacturing process. Voltage optimization begins at the lowest effective voltage and progressively increases effective voltage until delay test is passed. If a chip cannot pass the delay test under popular latencies and sizes, the scheme reverts to Method 2.
- **Method 4 (Tuning with Block-Level Pass Feedback):** Applies *dependent* tests with feedback from the first test into subsequent tests. The first test defines a low pass rate to maximize efficiency by tuning latency, size, and voltage. Block-level pass rate is measured, identifying blocks that pass and fail. Voltage is progressively decreased for passing blocks and increased for failing blocks.
- **Method 5 (Tuning with Two Tests):** Applies two *independent* tests. Tests define pass rates to maximize efficiency by tuning latency, size and voltage (described in Section 4.2).

Figure 11 compares average normalized efficiency and average number of tests required by each method. Method 5 achieves an average efficiency of 0.81 using only 2 tests. In contrast, the baseline Method 1 achieves 0.70 using, on average, 4 tests to identify the highest operating frequency without tuning latencies, sizes, and voltages. This data is consistent with Figure 10. Methods 2-4 deliver greater efficiency than Method 5 at greater cost. Methods 2-4 deliver efficiency between 0.84 to 0.97, improvements of 3.7 to 19.7 percent over Method 5. However, these methods require a greater number of tests and do not deliver efficiency in a cost-effective manner. Method 2 requires, on average, 4.5x the number of tests (9 versus 2) to deliver a 3.7 percent increase in efficiency (0.84 versus 0.81). Even less cost effective, Method 3 requires, on average, 10x the number of tests (20 versus 2) to deliver a 13.6 percent increase in efficiency (0.92 versus 0.81).

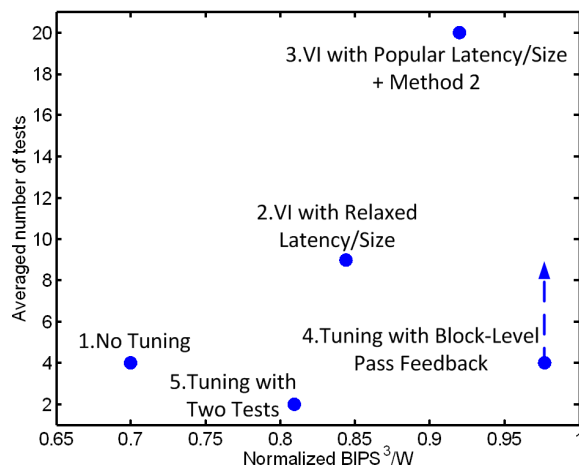


Figure 11: Comparison of testing schemes.

Method 4 warrants particular attention. Although its delivered efficiency is 19.7 percent greater than Method 5, its costs are less clear. Method 4 requires at least 1 test that attempts to maximize efficiency by tuning size, latency, and voltage. It then requires, on average, 3 additional tests that adjust per block voltages depending on whether a block passes or fails. However, Method 4 relies on knowledge of per block pass and fail rates, which might require additional per block measurements per test and increases the cost of every test. Furthermore, Method 4 requires state that is saved in the first test and used in subsequent tests, complicating the test flow.

Thus, Methods 2-5 deliver a range of  $BIPS^3/W$  efficiency and cost trade-offs. The best method depends on the particular testing and reliability requirements of each manufacturer. Manufacturers with low volume production might be willing to run more tests, incurring higher costs for higher performance and power efficiency. In contrast, high volume manufacturers might find marginal costs of additional tests are not justified by the marginal benefits in  $BIPS^3/W$  efficiency. Thus, post-fabrication tuning is the strategy we advocate, but the specific implementation of the strategy depends on cost-benefit analyses particular to each manufacturer.

## 6 Conclusion

Process variations has become an increasingly important issue for future microprocessor designs in nanoscale technologies. Various post-fabrication tuning techniques have been proposed recently to adapt the microarchitecture and circuit to different degrees of variations. But most of these techniques ignore a key question—the testing issue—since it will be prohibitive for efficient testing if the post-fabrication testing space is huge. This paper proposes the use of on-chip canary circuits to capture the correlated systematic variation, combined with statistical analysis and regression models to estimate the random variation and find the best post-fabrication settings for all chips. Experiments show the testing cost for the proposed approach is relatively low and can fit well into existing test flows with minimal overheads. Simulations shows the proposed test method can achieve close-to-optimal post-fabrication tuning results in terms of  $BIPS^3/W$ , even under very large

systematic variation and under reasonable assumption of random variation.

## References

- [1] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration", *Journal of Solid-State Circuits*, vol. 37, no. 2, February 2002.
- [2] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm design exploration", in *IEEE International Symposium on Quality Electronic Design*, 2006.
- [3] X. Liang and D. Brooks, "Mitigating the impact of process variations on processor register files and execution units", in *39th IEEE International Symposium on Microarchitecture*, December 2006.
- [4] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations", in *International Conference on Computer-Aided Design*, November 2003.
- [5] K. Meng and R. Joseph, "Process variation aware cache leakage management", in *International Symposium on Low Power Electronics and Design*, october 2006.
- [6] B. Cline, K. Chopra, and D. Blaauw, "Analysis and modeling of CD variation for statistical static timing", in *International Conference on Computer-Aided Design*, November 2006.
- [7] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variation and impact on circuits and microarchitecture", in *40th Design Automation Conference*, June 2003.
- [8] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, K. Flautner, C. Ziesler, D. Blaauw, T. Austin, and T. Mudge, "Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation", in *MICRO'03*, 2003.
- [9] M. Agarwal, B. Paul, M. Zhang, and S. Mitra, "Circuit failure reduction and its application to transistor aging", in *IEEE VLSI Test Symposium*, 2007.
- [10] Y. Li, S. Makar, and S. Mitra, "CASP: Concurrent autonomous chip self-test using stored test patterns", in *Design Automation and Test in Europe*, 2008.
- [11] J. Tschanz, J. Kao, and S. Narendra, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage", in *Journal of Solid-State Circuits*, Vol. 37, No. 11, November 2002.
- [12] R. Teodorescu et al., "Mitigating parameter variation with dynamic fine-grain body biasing", in *International Symposium on Microarchitecture*, December 2007.
- [13] J. Gregg and T. W. 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*, vol. 15, no. 3, pp. 366 – 376, 2007.
- [14] M. Mani, A. Singh, and M. Orshansky, "Joint design-time and post-silicon minimization of parametric yield loss using adjustable robust optimization", in *International Conference on Computer Aided Design*, 2006.
- [15] X. Liang, D. Brooks, and G. Wei, "A process-variation-tolerant floating-point unit with voltage interpolation and variable latency", in *IEEE International Solid-State Circuits Conference*, February 2008.
- [16] X. Liang, G. Wei, and D. Brooks, "ReVIVaL: Variation tolerant architecture using voltage interpolation and variable latency", in *International Symposium on Computer Architecture*, June 2008.
- [17] D. Sylvester, D. Blaauw, and E. Karl, "ElastiC: An adaptive self-healing architecture for unpredictable silicon", *IEEE Design and Test of Computers*, vol. 23, no. 6, Nov 2006.
- [18] A. Tiwari, S. R. Sarangi, and J. Torrellas, "Recycle: Pipeline adaptation to tolerate process variation", in *Proceedings of the International Symposium on Computer Architecture*, 2007.
- [19] N. Soundararajan, A. Yanamandra, C.A. Nicosopolous, N. Vijaykrishnan, A. Sivasubramaniam, and Mary Jane Irwin, "Analysis and solutions to issue queue process variation", in *International Conference on Dependable Systems and Networks(DSN)*, June 2008.
- [20] 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.
- [21] S. Ozdemir, D. Sinha, G. Memik, J. Adams, and H. Zhou, "Yield-aware cache architectures", in *39th IEEE International Symposium on Microarchitecture*, December 2006.
- [22] B. F. Romanescu, S. Ozev, and D. J. Sorin, "Quantifying the impact of process variability on microprocessor behavior", in *2nd Workshop on Architectural Reliability*, 2006.
- [23] P. C. Maxwell, "Wafer-package test mix for optimal defect detection and test time savings", in *Design and Test of Computers*, *IEEE vol. 20 (5) pp. 84 - 89*, 2003.
- [24] T. Fischer, J. Desai, B. Doyle, S. Naffziger, and B. Patella, "A 90-nm variable frequency clock system for a power-managed itanium architecture processor", *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, January 2006.
- [25] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, "Automatically characterizing large scale program behavior", in *International Conference on Architectural Support for Programming Languages and Operating Systems*, October 2002.
- [26] A. Phansalkar, A. Joshi, L. Eeckhout, and L. K. John, "Measuring program similarity: Experiments with SPEC CPU benchmark suites", in *IEEE International Symposium on Performance Analysis of Systems and Software*, March 2005.
- [27] B.C. Lee and D.M. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction", in *ASPLOS: International Conference on Architectural Support for Programming Languages and Operating Systems*, October 2006.
- [28] B.C. Lee and D.M. Brooks, "Illustrative design space studies with microarchitectural regression models", in *HPCA: International Symposium on High-Performance Computer Architecture*, February 2007.
- [29] D. Brooks and et. al., "Power-aware microarchitecture: Design and modeling challenges for next-generation microprocessors", *IEEE Micro*, vol. 20, no. 6, pp. 26–44, Nov/Dec 2000.
- [30] Intel Corporation, "Intel processor pricing. effective july 20, 2008".