University of California

Los Angeles

Hardware-Software Interface in the Presence of Hardware Manufacturing Variations

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Electrical Engineering

by

Aashish Pant

2010

The thesis of Aashish Pant is approved.

University of California, Los Angeles

2010

To my parents, Bhasha and Prabhat Pant...

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my research advisor, Prof. Puneet Gupta for the constant motivation, guidance and support throughout this work. Completing this thesis would have been impossible without his help. I am also thankful to Prof. Mihaela van der Schaar for helping me understand the basics of multimedia encoding algorithms and to Prof. Rakesh Kumar from UIUC for guiding me in the area of computer architecture. I would also like to thank John Sartori from UIUC and Nicholas Mastronarde for helping me in setting up the experiments for my work. My sincere acknowledgements are due to my lab-mates: Tuck Boon Chan, Rani S. Ghaida, Amarnath Kasibhatla, Viswakiran Popuri, Abde Ali Kagalwalla and John Lee for giving suggestions, participating in various discussions related to my project and supporting me during those tough paper submission times. My special thanks to UCLA Fall-08 group for making my stay at UCLA, a memorable one. I would like to express my deep gratitude towards my parents, family, and friends for their constant love, encouragement, and support.

Abstract of the Thesis

Hardware-Software Interface in the Presence of Hardware Manufacturing Variations

by

Aashish Pant

Master of Science in Electrical Engineering

University of California, Los Angeles, 2010

Professor Puneet Gupta, Chair

Variations in manufacturing process are increasingly affecting the performance (speed, power) of systems, both across multiple instances of a design and in time over its usage life. Even in these conditions, the basic approach to designing and operating complex systems that embeds a hardware-software interface that is rigid and deterministic has largely stayed the same which coupled with the objective to achieve a good manufacturing yield often leads to systems being overdesigned and power hungry. In this work, we present the motivation to think of systems that have a flexible hardware-software boundary. We show that, by adapting the software application to the post manufacturing performance-power characteristics of the hardware across different die, it is possible to compensate for the application quality losses that might otherwise be significant. This in turn results in improved manufacturing yield, relaxed requirement for hardware over-design and a better application quality. In another experiment, we show that performance binning for multi-core processors should be based on the throughput of the multi-core system which depends on the workload that is run. Such binning metrics can aid in the graded pricing of multi-core processors to maximize profit.

CHAPTER 1

Introduction

Variations in manufacturing process are increasingly affecting the performance (speed, power) of systems, both across multiple instances of a design and in time over its usage life. With finer geometry technologies, manufacturing variations have become very important. For high performance microprocessors in 180nm technology, measured variation is found to be as high as 30% in performance and 20x in chip level leakage within a single wafer [1]. According to the authoritative International Technology Roadmap for Semiconductors [ITRS] [2] projections on performance and power variability for next 10 years shown in Figure 1.1 and various other research surveys [3, 4], this trend is expected to worsen.

A number of approaches have been proposed to handle the variability associated with the manufacturing process. Most of these approaches statistically model and forecast the effect of variations early in the circuit design flow. These approaches fall under the broad category of *manufacturing aware design techniques*. In these set of approaches, the effect of process variations is statistically incorporated in the design flow and the circuit is designed keeping in mind that the expected manufacturing yield is maximized, under the constraint that a certain minimum performance level is satisfied. Although such techniques solve the problem to some extent, they often lead to the creation of designs that waste power and are high on designer effort, often resulting in a longer time to market.

Some of the most widely used industrial techniques rely on fine tuning of the

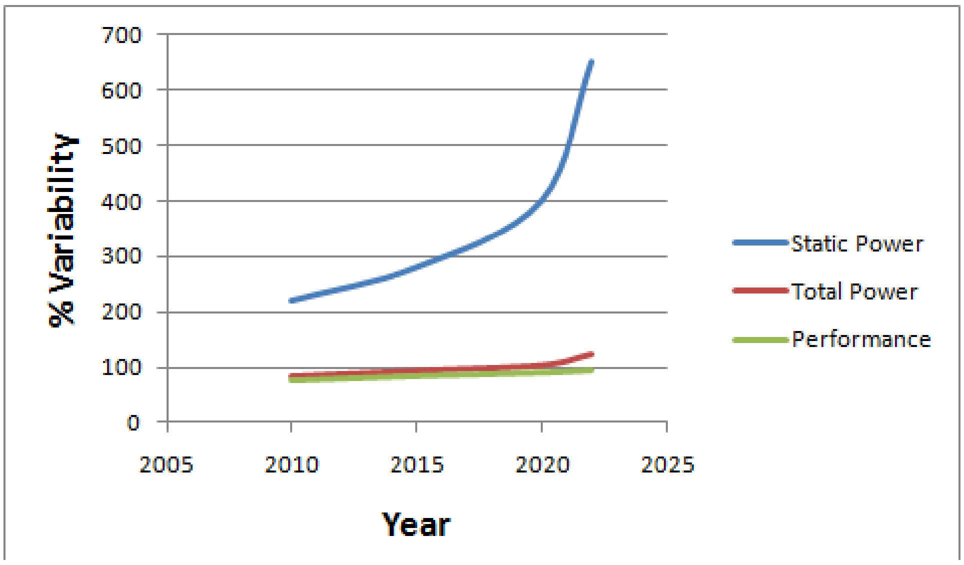


Figure 1.1: ITRS Projection of Variability.

manufactured hardware. For example, for slower chips, the threshold voltage of the gates on the critical path can be lowered after manufacturing in order to make the circuit run faster by using forward body biasing. For extra leaky chips, the threshold voltage can be increased to reduce leakage. Often, these techniques, though effective, result in high power dissipation and complex designs that support post-manufacturing tuning. Moreover, the tuning needs to be done on a chip by chip basis and this results in an increased chip test time.

In another set of methods, the final design is subjected to a pre-manufacturing step where it is changed in a way that guarantees the final chip to be operating near the expected nominal. For example, methods like optical proximity correction fall under this category. These methods are indispensable for a reliable yield and significantly complicate the process and mask making steps.

While process variability is significantly increasing, the basic approach to designing and operating complex systems has remained unchanged. While the structure of these complex systems has evolved from being general purpose to specialized, and from single core to multi-core, the notion of a hardware-software interface that is rigid and deterministic has largely stayed the same. Software has always assumed the hardware to deliver a certain minimum level of performance, which the hardware designers try hard to meet without leveraging software's flexibility.

This rigid hardware-software paradigm coupled with the objective to achieve a good manufacturing yield often leads to systems being overdesigned relative to their specification by addition of certain guardbands. Getting the last bit of performance incurs serious power and area overheads, thus increasing overall design cost. It also leaves enormous performance and energy potential untapped as the rigid software has to assume lower hardware performance than what a majority of the instances of that system deliver most of the time. Therefore, there is a motivation to think of systems that have a flexible hardware-software boundary. The idea is very similar to modern wireless systems where instead of designing protocols with rigid speed and power constraints, an approach that is flexible and allows for trade-offs is used and it has been proven to be far more effective.

In this work, we seek to rethink the area of hardware-software interaction. We observe that it is much cheaper to make software adaptable than it is to make hardware robust and adaptable. In Chapter 2, we show that, by adapting the software application to the post manufacturing performance-power characteristics of the hardware across different die, it is possible to compensate for the application quality losses that might otherwise be significant. This in turn results in improved manufacturing yield, relaxed requirement for hardware over-design and a better application quality.

In Chapter 3, we demonstrate how hardware can be graded into different categories based upon the application that runs on top of them by taking the example of multi-core frequency binning. We show that performance binning for multi-core processors should be based on the throughput of the multi-core system which depends on the workload that is run. Such binning metrics can aid in the graded pricing of multi-core processors.

Thus, this work researches on a set of approaches that aim at changing the traditional notion of a rigid hardware-software interface to one that has more flexibility thus ensuring high manufacturing yields, improved profits and better system quality.

CHAPTER 2

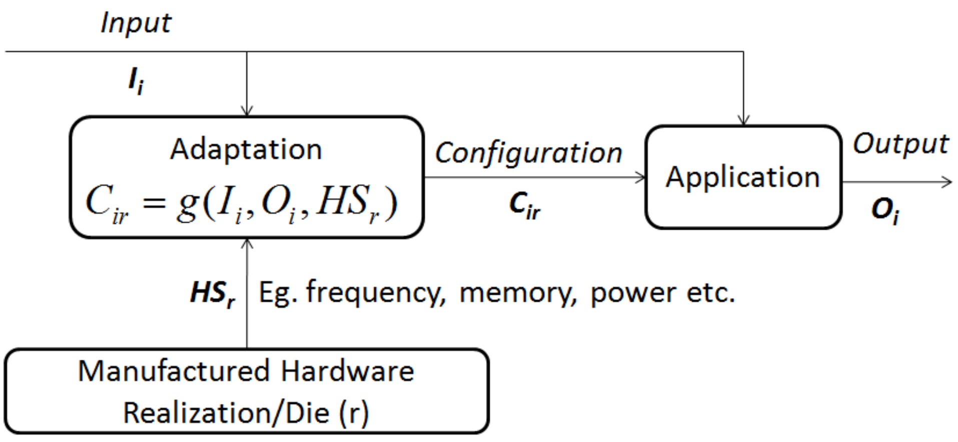
Software Adaptation in Quality Sensitive Applications to Deal With Hardware Variability

2.1 Introduction

Variations in manufacturing process play a very important role in determining end circuit functionality. For high performance microprocessors in 180nm technology, measured variation is found to be as high as 30% in performance and 20x in chip leakage within a single wafer [1] and with technology scaling, the impact is getting worse [2–4].

A number of approaches have been proposed to handle the variability associated with the manufacturing process. While some of these approaches statistically model and forecast the effect of variations early in the circuit design flow [4], others like [5][6] rely on post manufacturing tuning of the hardware. Performance-power optimization techniques like DVS have been used to take process variations into account as in Razor [7]. However, over-designing hardware is the most commonly used industry mechanism to regulate manufacturing yield. Over-design comes at a significant cost, power, turnaround time and designer overheads.

In this work, we propose to mitigate the impact of process variations through software adaptation for quality sensitive applications as shown in Figure 2.1. We show that, by adapting the application to the post manufacturing performance-



Input
I_i
Adaptation
$C_{ir} = g(I_i, O_i, HS_r)$
Configuration
C_{ir}
Application
Output
O_i
HS_r
Eg. frequency, memory, power etc.
Manufactured Hardware Realization/Die (r)

Figure 2.1: Simplified application adaptation model with hardware signatures.

power characteristics of the hardware (we refer to these characteristics as hardware signatures) across different die, it is possible to compensate for the application quality losses that might otherwise be significant. This in turn results in improved manufacturing yield, relaxed requirement for hardware over-design and better application quality.

Our work is motivated by the following two observations:

1. A plethora of modern applications are quality sensitive, e.g. video encoding, stream mining etc. These applications are capable of operating in various configurations by adapting to certain input or environmental conditions in turn producing similar or different quality of service. This notion can be extended to let variation-affected hardware drive application adaptation.

2. Process variation is increasing and hence, the conventional methods of incorporating variation resistant design techniques, post manufacturing hardware tuning or hardware over-design can be too expensive to use.

Communication systems provide an excellent analogy[8]. Communication systems adapt based on the underlying physical communication fabric which is dynamic (for instance [9–11]). In the same way, a system can also adapt to the underlying variation-affected hardware layer. Increased hardware variation and a plethora of adaptation friendly applications motivate the use of this idea.

The idea of modifying the software to suit the underlying hardware (for process variations or otherwise) is not entirely new. In a recent work[12], the authors propose a method to optimize the power management policy of an *SOC* statistically across all chips taking process variations into account and its effect on leakage power. Further, they suggest approaches to tweak the policy on a chip by chip basis. Software fault tolerance schemes like [13] fall under a related category where hardware faults are detected (using methods like ECC) and corrected in the software layer. [14] proposed the design of a low power motion estimation framework in which the supply voltage is purposely lowered triggering some timing faults which are then corrected using software fault tolerance techniques. [15] proposes an approach to handle supply voltage variations using a voltage sensor, error recovery hardware and runtime modification of the compiled software to prevent such voltage variation triggering. Software thermal management techniques like [16] perform scheduling in a multitasking scenario to maintain thermal constraints. The work presented in [17][18] uses application error resilience in hardware test. A recent work in [19] proposes soft architectures designs that fail gracefully, thus allowing reliability/performance trade-offs upto the level which can be tolerated by the application at hand.

All such previous software level approaches either model the hardware inadequacy or malfunctioning as transient faults and treat them as emergencies or rely on the inherent error tolerance of some applications. Moreover, these techniques are triggered when the so called faults happen and some of them require special hardware. For process variations, software adaptation can utilize the application algorithm's quality or performance tradeoffs to achieve error free operation in presence of permanent manufacturing variations.

Adaptation is easier and cheaper (to implement) as well as better informed at the application software layer rather than hardware. Contributions of our work include the following.

- A general framework to discuss application adaptation based on process-variation affected manufactured hardware.
- Using an H.264 encoder, we show that the use of hardware realization-based software adaptation increases manufacturing yield, improves overall application quality and thereby allows for under-design of hardware.
- We present novel methods to compute optimal signature measurement points.

This paper is organized as follows. In section 2.2, we introduce the concept of hardware signatures based adaptation in quality sensitive applications and then describe the general idea. In section 2.3, we apply this methodology to an H.264 encoder and demonstrate the improvements. In section 2.4, we discuss the effects of signature discretization and present an algorithm to determine optimal signature measurement points. We conclude in section 2.5.

2.2 Hardware Signature Based Adaptation

In this section, we describe the use of hardware signature based software adaptation for quality sensitive applications.

2.2.1 Quality Sensitive Applications: Q-C Curves

Consider a quality sensitive application that can operate in different software configurations to maximize a certain quality metric under the constraint that the input is processed in time T_{MAX}. If the input processing time is T_c under configuration c and Q_c is the corresponding output quality, the job of the adaptation algorithm is to find the configuration c_{best} such that,

$$\begin{aligned}
&c_{best} = argmax_c(Q_c) \\
&where\ c \in \ \{\ set\ of\ all\ configurations\ \} \\
&T_c \leq T_{MAX} \qquad (2.1)
\end{aligned}$$

We model the behavior of such a system by means of a *Quality-Complexity (Q-C) curve* (see Figure 2.2) (e.g.[20]). Any point on the *Q-C graph* denotes some application operating configuration and Q-C curve is the envelope or the curve connecting the points of quality upper bound for every complexity point. Note that *complexity* (x-axis) is synonymous to *processing time* in our case and we shall use the latter in this discussion.

Clearly, the Q-C graph (and hence the Q-C curve) changes with the underlying hardware realization. A more complex algorithm can be run on a faster hardware to satisfy the same time constraint with improved application quality. In general, an application configuration point maps to different time to process values to achieve the same quality on the Q-C graph for faster/slower hardware i.e. the point undergoes a respective horizontal left/right shift in position on the Q-C graph. Therefore, the envelope or the operational Q-C curve also changes.

Because of process variations, every manufactured die is different. The direction and magnitude of each point shift on the Q-C graph depends on the

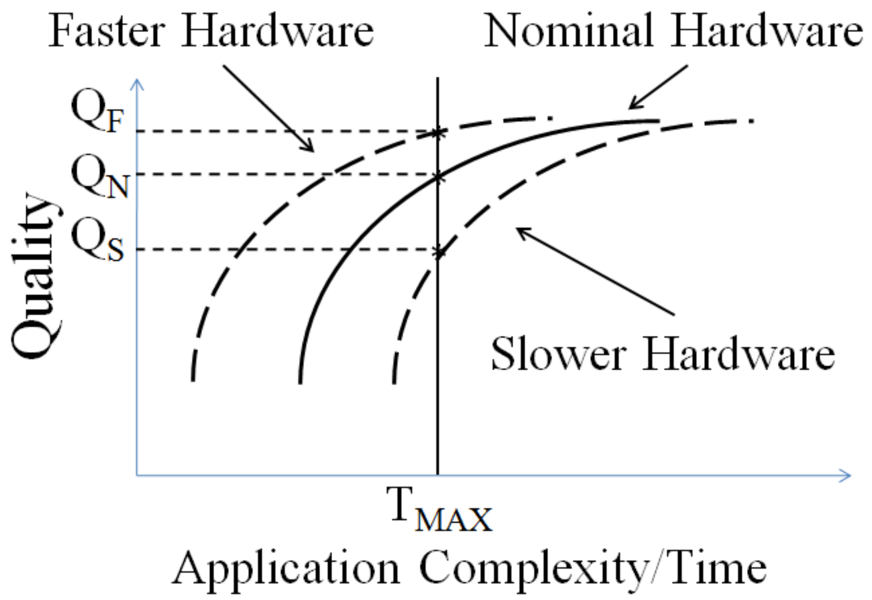


Figure 2.2: Q-C curve for a one-component hardware undergoes a scaled horizontal shift with frequency variations.

relative contribution of various constituent functional blocks in that application configuration and the magnitude of process variations for each of these functional blocks. For the special case of a one-component hardware, every point on the Q-C graph shifts horizontally by the same percentage amount and the result is a simple scaled horizontal shift of the Q-C curve (see Figure 2.2).

If the underlying application is unaware of such Q-C graph perturbations (as in present day systems), the way it solves (2.1) and the resultant configuration selection cannot be optimal. This results in a loss of manufacturing yield as systems that cannot meet the specified timing or quality constraints because of manufacturing variations are simply discarded. We propose that such Q-C graph perturbations can be captured by storing actual *hardware signatures*. For quality sensitive applications, frequency deviation of the hardware from the nominal is the hardware signature. Figure 2.1 pictorially depicts the proposed hardware-aware adaptation model. Next, we present a generalized description of the idea.

2.2.2 Signatures and Adaptation

Hardware signatures are the post manufacturing hardware performance-power numbers that are communicated to the software for adaptation. Apart from the fact that they differ from one hardware realization to another (die to die variations), they might also differ from one functional block to the other within the same die (because of within die process variations). The hardware signature then consists of the independent block level signatures[1]. An important consequence is that an application can knowledgeably adapt and redistribute the effort of computation among its hardware components to achieve the same desired performance given the manufactured hardware i.e. a chip that will be discarded in the current setup can be made usable by changing the application's algorithm to give the same performance (by redistribution of workloads among hardware components according to variation map) or at a slight loss in quality. (We demonstrate these benefits in section 2.3 in Figure 2.4 in the context of an H.264 encoder, where under hardware-aware adaptation, a slower hardware results in the same PSNR of the encoded video as the nominal hardware without adaptation).

2.2.3 Signature Choice and Measurement

Choice of signature values depends on system objectives. For a system that poses strict constraints on timing (like real time quality sensitive applications), signature could comprise of the frequency deviations of the individual functional blocks of the hardware. System memory along with the speed of CPU-memory interface can also be important metric to include if memory intensive and com-

[1]This assumes that different blocks can be clocked at different frequencies

putation intensive techniques are choices for application configuration. For low power applications that try to trade-off performance for power, leakage power dissipation values and maximum switching current can be stored as signatures.

Signatures can be measured once post-fabrication and written into a non-volatile memory element on-chip or on-package. These memory elements need to be software-readable[2]. They may also be measured at regular intervals during operation to account for wearout mechanisms such as TDDB and NBTI as well as ambient voltage/temperature fluctuations. Well-known parametric tests such as FMAX (performance) and IDDQ (leakage power) can yield signature values. At-speed logic and memory built-in self test (BIST) techniques can be employed as well for faster and any time computation of the signatures. Approximations using on-chip monitors (e.g., ring oscillators or monitors such as [22]) can work as well. Since signature measurement involves using test techniques with well understood overheads, in this work we do not discuss these methods in more detail.

2.3 Proof of Concept: H.264 Encoding

We demonstrate the benefits realized through hardware-aware adaptation using Q-C curves for an H.264 encoder. Maximum permitted frame encoding time is T_{MAX} and the quality metric Q is the PSNR (peak signal to noise ratio) of the encoded video at a constant bit-rate. If this time deadline is not met, the frame is dropped, affecting the PSNR of the output video and manufacturing yield of the hardware.

[2]Most modern chips already contain several such EEPROM or NVRAM components for storing hardware IDs, time, etc (e.g., see [21])

Table 2.1: Experiment Specifications

Video Source	Mobile Sequence
Number of Frames	250
Encoder Tuning Knobs Used	Motion estimation accuracy (Full-Pixel, Sub-Pixel), Transform window sizes (4x4, 8x8, 4x4 & 8x8), Entropy coding algorithm: CAVLC, CABAC[26], Number of reference frames for Motion Estimation, Motion Estimation Search Range, Quantization Parameters
T_{MAX} (not accounting for over-design)	0.03 seconds
Bitrate	1 Mbps
Frequency Variations	I.I.D Gaussian Distributed Mean = 0 SD: 6.66%
Monte-Carlo Samples	1000

2.3.1 Experiment Setup

We use a H.264 software encoder [23][24] for our experiments. The three critical components of H.264 encoder are motion estimation (M.E), DCT transform (T.X) and entropy coding (E.C). The encoder is tunable through various configuration parameters[25]. Problem of adaptation is therefore, to solve (2.1) for configuration c_{best}, $T_c = t_{M.E} + t_{T.X} + t_{E.C}$ where $t_{M.E}$, $t_{T.X}$ and $t_{E.C}$ is the time taken by M.E, T.X and E.C units respectively. Note that the hardware signatures are frequency variations of these components represented by the triplet $\{t_{M.E}$, $t_{T.X}$, $t_{E.C}\}$.

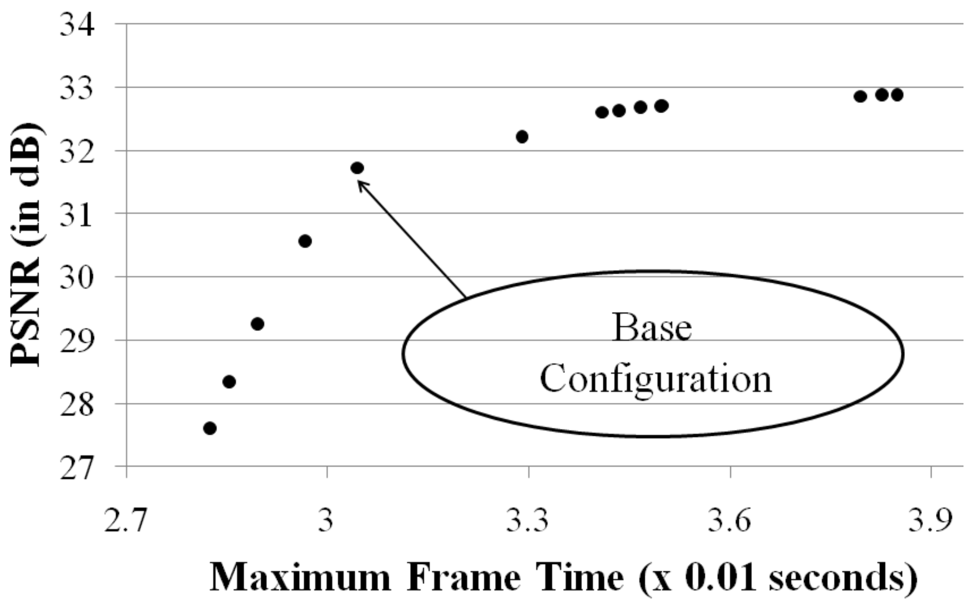


Figure 2.3: Q-C curve for H.264 encoder showing the various operating configurations.

We profile the encoder, measure output PSNR[3] and time taken by M.E, T.X and E.C units on a per frame basis for encoding the standard *mobile video sequence*[4] for the chosen encoder configurations. The specifics are indicated in Table 2.1[5]. This data is used to construct the Q-C curve for the H.264 encoder at nominal hardware which is shown in Figure 2.3. Base configuration is the one for which the nominal hardware is designed. Further, we vary hardware over-design from -20% to +20%. Overdesign provides a buffer/guardband in performance to take care of process variations after manufacturing. This over-design has significant penalties in terms of area, power, cost and turnaround time [27]. Over-design buffer is added to the the maximum frame time for the base

[3]In this context, it should be noted that a PSNR difference of 0.5 to 1 dB is significant and is visible

[4]Note that the results will vary with video sequences and in practical systems, some online learning techniques may be employed to adapt to the sequence/workload characteristics

[5]The profiled runtimes are scaled to ensure 33 fps video

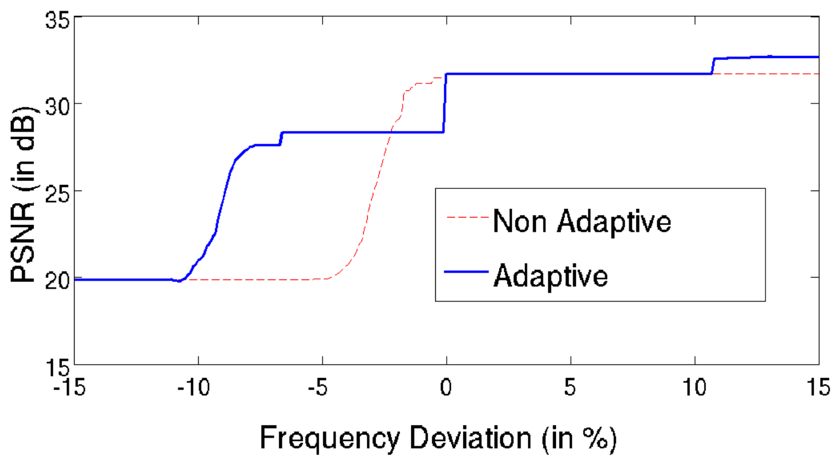


Figure 2.4: Signature based adaptation achieves better PSNR at a given frequency of operation compared to the non-adaptive case.

configuration, and the resulting sum is taken as T_{MAX}.

2.3.2 Results

In Figure 2.4, we show how the encoder PSNR changes with variation in operating frequency[6]. As frequency reduces, the non adaptive encoder violates the time constraint for certain complex frames which eventually get dropped, resulting in a significant PSNR loss[7]. With hardware-aware adaptation, the encoder adapts and operates in a configuration that results in minimum frame loss, eventually giving a high PSNR output. *In other words, hardware-aware adaptation achieves the same desired PSNR with a lower frequency of operation, which in turn implies that such a system can tolerate variations to a greater extent.* Note that, a small part of the curve where PSNR for adaptive case is lower than that of non adaptive

[6]For this analysis, all three hardware components are assumed to have the same variation so that the results can be shown on a 2-D plot

[7]We handle lost frames by replacing them with the previous known good frame and computing the output PSNR as is usually done in real time multi-media decoders

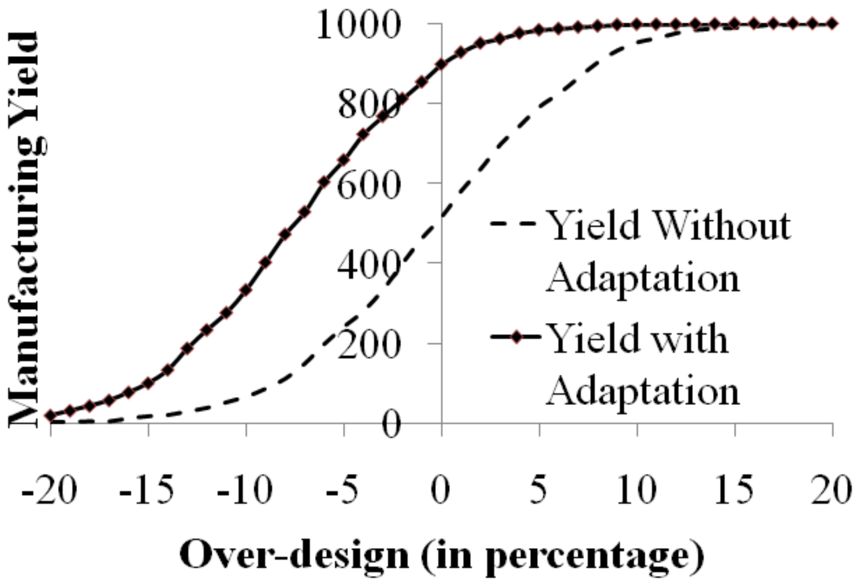


Figure 2.5: Manufacturing yield is defined as the percentage of die that ensure no frame loss.

case, is because in our experiments, adaptation is guided to achieve no frame loss rather than minimum PSNR.

We generate 1000 Monte-Carlo samples of percentage delay variations for the three components assuming them to be i.i.d. gaussian distributed random variables with mean 0 and standard deviation 6.66% (3σ=20%). Actual frame processing times are calculated by applying these variation samples over the nominal values and the Q-C curve perturbation is estimated. For the non adaptive case, frames with processing times exceeding T_{MAX} (i.e., the corrected maximum permitted time after taking over-design into account) in base configuration are dropped resulting in yield loss. Adaptation is guided to select a configuration that has minimum frame loss for the given hardware. In our experiments, we define manufacturing yield as the percentage of die that ensure no frame loss (i.e., a jitter constraint).

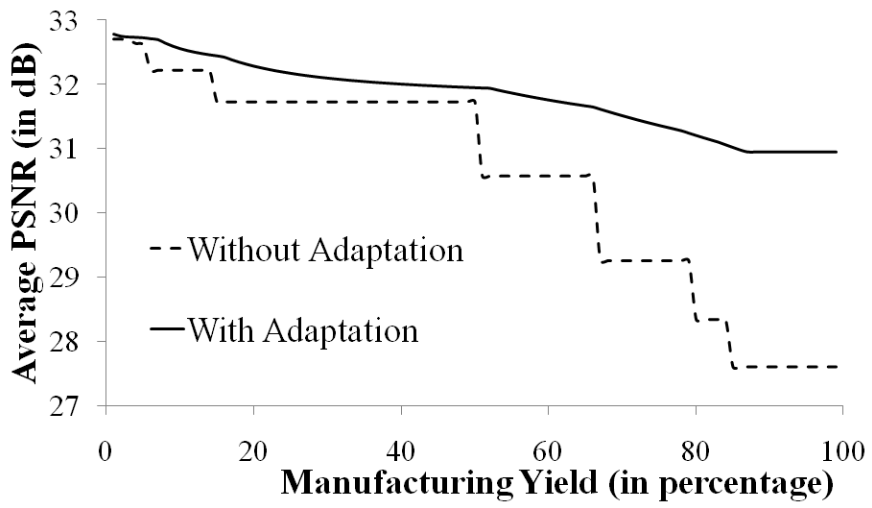


Figure 2.6: PSNR Vs. Yield for 0% over-design

Figure 2.5 demonstrates significant yield improvements with hardware adaptation. At 0% over-design, yield of the non-adaptive encoder is 50% (intuitively, half of the die lie on either side of the nominal hardware realization with normal frequency distribution). When the encoder adapts to manufactured hardware, it operates in a configuration with minimal frame loss and yield increases significantly to 90%. This trend is seen over the entire span of over-design or under-design values. *An important point to observe is that, given enough available configurations, application adaptation can ensure almost constant quality by trading off work needed for different components.* Nevertheless, some hardware realizations do show a slight PSNR degradation since yield is defined to ensure no frame loss.

From Figure 2.5, we can also conclude that hardware-aware adaptation relaxes the requirement of over-design to achieve the same manufacturing yield. For example, to ensure 80% yield, adaptation reduces the over-design requirement by 10%.

Figure 2.6 shows how average PSNR across all die falls as one aims for a

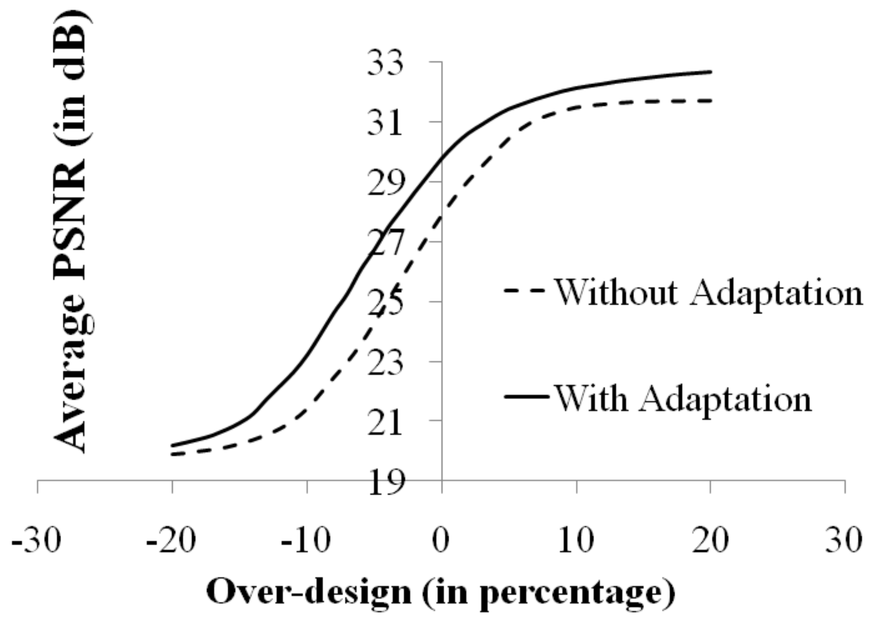


Figure 2.7: Average PSNR over all die samples.

higher manufacturing yield for both hardware adaptive and non-adaptive cases. We only show the plot for 0% over-design as the data for other over-design values follows the same trend. From the figure, it is observed that adaptation results in a higher average PSNR over the entire range of manufacturing yield[8]. At 80% yield, averare PSNR for hardware adaptive case is higher by 2.6dB. For the non-adaptive encoder, increase in yield comes at significant PSNR penalty because the encoder has to ensure a low enough complex configuration (for all die) that satisfies the required yield and hence a staircase PSNR waveform is observed. *However, adaptation allows for a graceful degradation in PSNR when improving yield, as operating configurations can change on a die-by-die basis.*

In Figure 2.7, we show the behavior of average PSNR over *all* die samples with varying over-design values. An improvement of about 1.4dB is seen over

[8]For the adaptive case, the highest quality realizations are used to match the non adaptive case for the same yield

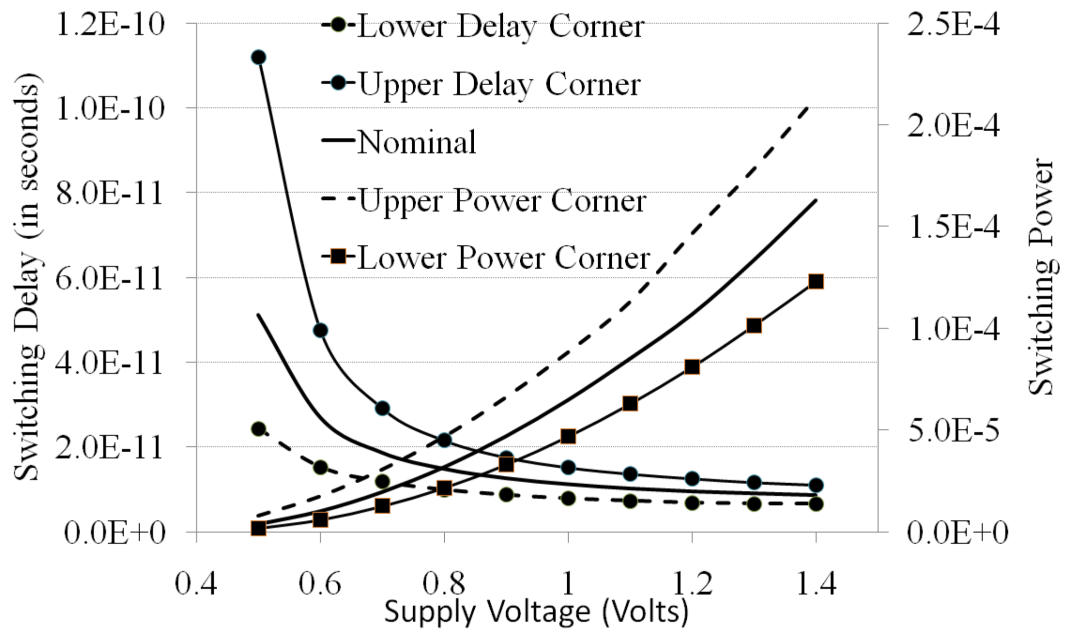


Figure 2.8: Variation of frequency and power with supply voltage under process variations.

almost the entire over-design range.

2.3.3 DVS: Power and Voltage as Hardware Signatures

In the above discussion, we considered a system where quality (PSNR) was maximized under the constraint that the input was processed within the alloted time. Frequency deviations from the nominal values were the hardware signatures in this case. For energy constrained systems, power dissipation is an important quality metric to include in the adaptation process. Consider Figure 2.8 which shows the dependence of frequency and power on supply voltage for a simple 4 stage FO-4 inverter chain[9] under process variations (varying transistor length, width and threshold voltage by +-10%) using HSPICE. The curves indicate the

[9]45nm PTM models have been used for these simulations

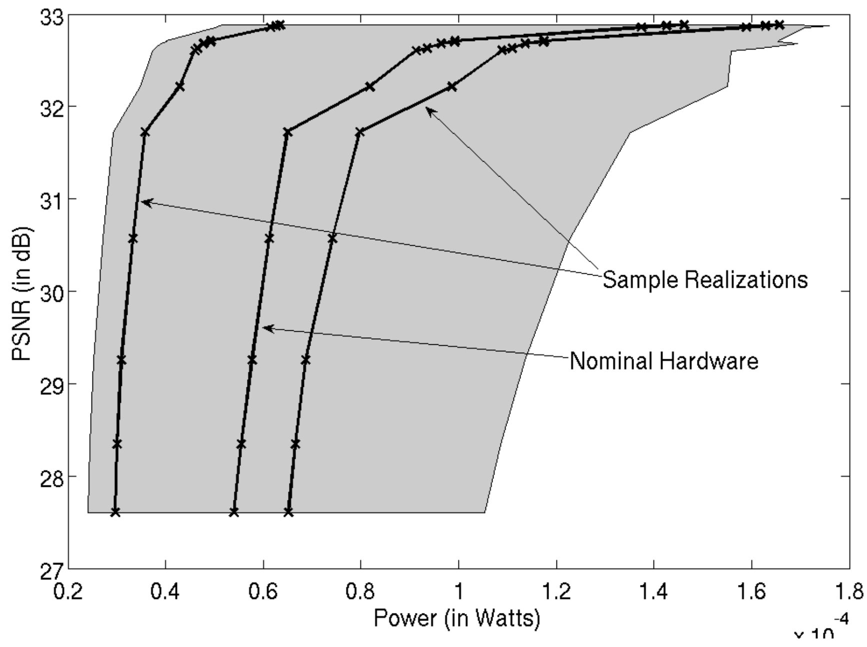


Figure 2.9: Variation space of the PSNR vs power curves for some sample hardware realizations under process variations for H.264 encoder.

nominal and the fast/slow delay/power envelopes. It can be seen that the supply voltage required to achieve the same frequency for different hardware realizations is significantly different and so is power dissipation, resulting in a wide power-performance band. For example, at supply voltage of 1V, there is a variation of 64% in delay and 63% in switching power across the nominal. More interestingly, to achieve the same switching delay of $20ns$, the switching power spans from 13μW to 25.5μW (i.e. 68% of the nominal power of 18.21μW at $20ns$). By knowing the exact power-performance numbers for a die, adaptation algorithms like DVS (dynamic voltage scaling) that try to optimize on a combined performance-power-quality metric can do a much better job by adapting in a manner specific to the die. This motivates the inclusion of power as a possible signature metric for such systems.

To further motivate this work and estimate the returns that one can expect, Figure 2.9 plots the PSNR vs power trade-off for various hardware realizations for the encoder configurations of Figure 2.3. For a given T_{MAX}, every encoder operating configuration is associated with a minimum operating frequency requirement (to achieve that T_{MAX}) and let us assume that these are the frequencies that DVS can make the system operate on. Intuitively, to achieve the same frequency, different realizations need different voltages and hence have different switching power dissipation. The figure indicates significant PSNR-power curve differences across different realizations.

Hardware signature for such a system will consist of a look up table that specifies the operational voltage ([28, 29] proposed a look-up table based method to store and track frequency-voltage relationships across process and temperature variations) and power dissipation as well for each frequency of operation. This information will let the application (DVS) know of the exact operational PSNR-Power curve specific to that die.

2.4 Hardware Signatures: Granularity Tradeoffs

Size (i.e., how many functional blocks and how many parameters per block) and granularity (e.g., discretization of performance into frequency bins) of the signature affects the potential benefit that can be derived from signature-based adaptation. Signature granularity influences test as well as storage complexity. In this section, we focus on determining optimal signature measurement points from Q-C curves for one-component hardware or multiple components with perfectly correlated variation. We will show that there is no benefit in having more number of hardware signature measurement points than the number of available configurations.

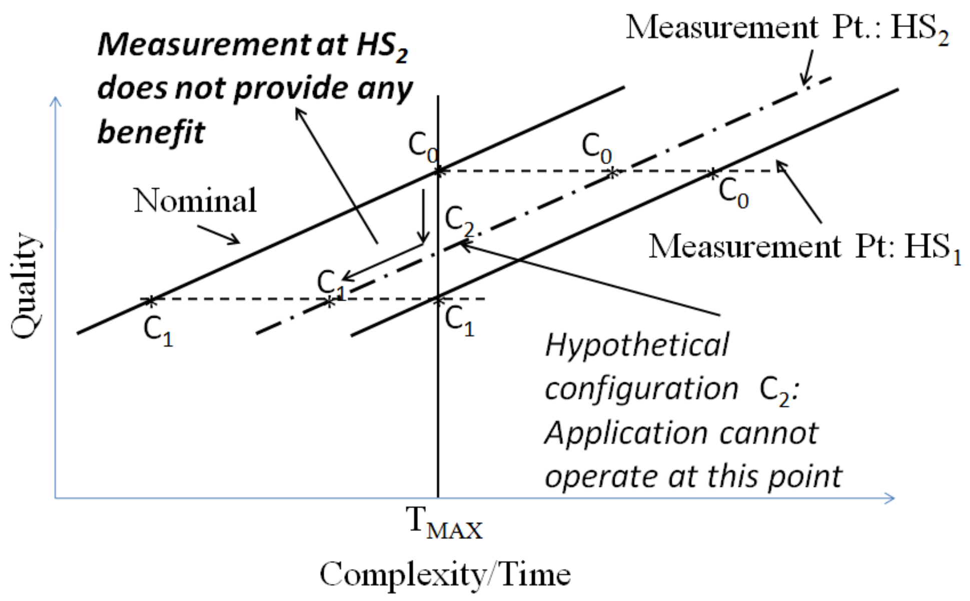

Measurement at HS2 does not provide any benefit
Measurement Pt.: HS2
Nominal
C0
C0
C0
C2
C1
C1
C1
Measurement Pt: HS1
Quality
Hypothetical configuration C2: Application cannot operate at this point
TMAX
Complexity/Time

Figure 2.10: Signature measurement point analysis using Q-C curves.

2.4.1 Optimal Signature Measurement

In Figure 2.10, C_0 and C_1 are two operating configurations. The Q-C curve at nominal hardware and also for two slower hardware, HS_1 and HS_2 is shown, where hardware HS_1 is slower than hardware HS_2. For hardware HS_2, C_2 (that lies on the T_{MAX} line) is not a valid physically existing operating configuration. So, the software operates at C_1. For hardware HS_1, C_1 lies on the T_{MAX} line and the software operates at C_1. Therefore, hardware HS_2 and the hardware HS_1 are equivalent. This equivalance arises because the operating points are discrete.

Therefore, every hardware slower than the nominal but faster than the hardware at HS_1 will operate on C_1. Hence, signature measurement is only required to be done at HS_1. In general, the maximum number of signature measurement points for optimum gain are the number of configurations. These measurement points correspond to those hardware which have their Q-C curves intersecting the T_{MAX} line at valid operating point.

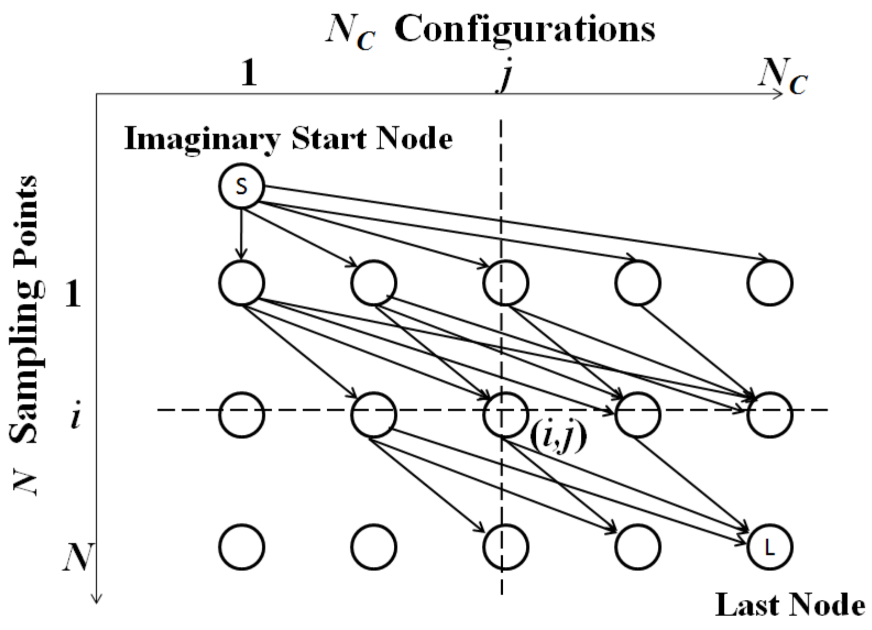


Figure 2.11: Mapping the optimal signature location problem to a shortest path problem.

If the available number of hardware measurement points, N are less than the number of configurations, N_C, a brute force search technique would require $\binom{N_C}{N}$ operations to get to the optimal measurement set. We map the optimal measurement set problem to a graph shortest path problem and solve it using Dijkstra's algorithm[30]. Consider Figure 2.11. For notational convenience, to have a measurement point at configuration c is to have a signature measurement point at that hardware which has its Q-C curve intersecting the T_{MAX} line at configuration c. Now, let Q_j denote the quality corresponding to configuration C_j and let X_j be the corresponding measurement location. The number of nodes in the graph is $N_C * N$ (arranged as a matrix) and the cost of an edge from node $(i1, j1)$ to $(i2, j2)$ ($cost^{(i2,j2)}_{(i1,j1)}$) is the quality loss incurred by having signature measurement points at configurations $j1$ and $j2$ and no measurement point between them (note that all nodes in column j have same quality Q_j and $Q_{j1} > Q_{j2}$ for

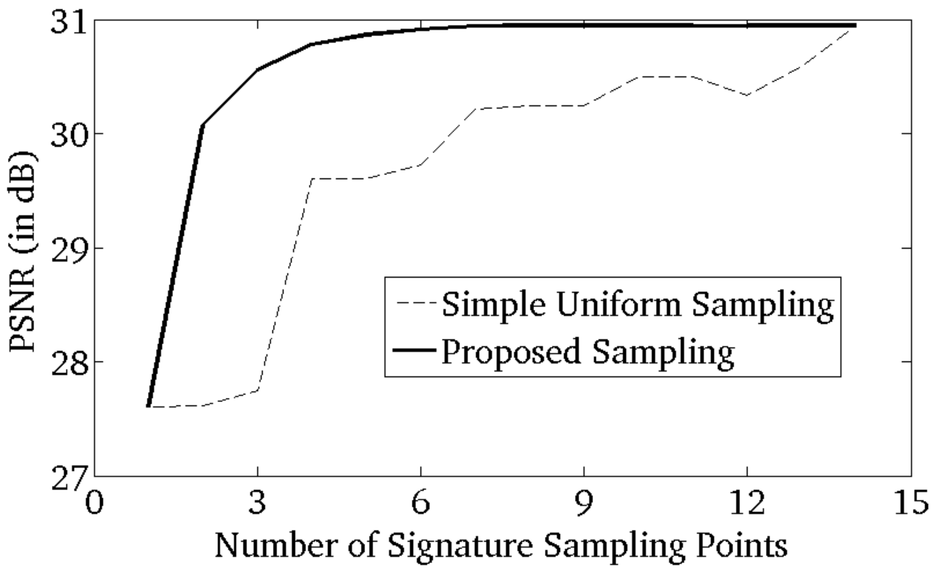


Figure 2.12: Improvement in PSNR with finer signature granularity.

$j1 < j2$). If *p(x)* is the probability distribution of the frequency variations of the hardware, then

$$
\begin{aligned}
cost_{(i1,j1)}^{(i2,j2)} &= \infty \;\; for\; j2 \leq j1 \; or \; i2 \neq i1+1 \\
&= \sum_{l=j1+1}^{j2} ((Q_l - Q_{j2}) \int_{X_{l-1}}^{X_l} p(x)\, dx), \;\; otherwise
\end{aligned}
$$

Every path from node S (imaginary node corresponding to having a signature at ∞) to node L (signature measurement location X_N corresponding to the maximum tolerable variation) will consist of N nodes. The quality loss minimization problem maps to finding the shortest path from S to L. Nodes in the path correspond to the measurement locations.

2.4.2 H.264 Encoding: Granularity Analysis

We derive optimal signature measuring locations on the Q-C curve of the H.264 encoder shown in Figure 2.3 using the proposed shortest path based strategy and the results are compared with a naive uniform signature measurement based approach. Monte-Carlo analysis is performed with 1000 die samples where all components have the same variation. From Figure 2.12, it can be observed that the proposed signature measurement method results in higher PSNR than the naive approach. Also, as we increase the number of available measurement points, the marginal benefit of adding another signature sample decreases. For six available measurement points, the improvement in PSNR with the proposed approach is about 1.3dB. Granularity analysis for a generic multi-component hardware with independent variations is part of our ongoing work.

2.5 Conclusion

In this work, we have proposed a method to reduce the impact of process variations by adapting the application's algorithm at the software layer. With increasing process variations and applications being adaptive and quality sensitive, we show that variation-aware software adaptation can ease the burden of strict power-performance constraints in design. Hardware signatures or the post manufacturing power-performance numbers of the hardware, can be used to guide software adaptation. Using the concept of Q-C curves and Monte-Carlo analysis on an H.264 encoder, we illustrate that this approach can lead to an improvement in manufacturing yield, relaxed requirement for over-design and an overall better application quality. Specifically, we show that, for the H.264 encoder

- Manufacturing yield improves by 40% points at 0% over-design.

- To achieve the same yield of 80%, adaptation relaxes the need for over-design by 10%.

- Encoding quality is better by 2.6dB over the non adaptive case for 80% yield.

We also derive strategies to determine optimal hardware signature measurement points and analyze the effects of signature granularity on application quality for one-component hardware or multiple components with perfectly correlated variation. Specifically, we show that our proposed approach for determining optimal signature measurement points results in an improvement in PSNR of about 1.3dB over naive sampling for the H.264 encoder.

As part of our ongoing work, we extend signature granularity analyses to multi-component (possibly pipelined), multi-application systems (possibly mediated by an operating system). We are also pursuing other application scenarios such as DVS (already hinted at in this paper) and optimal signature-dependent adaptation policy perturbations for adaptive applications.

CHAPTER 3

Frequency Binning of Multi-core Processors

3.1 Introduction

Performance (or speed) binning refers to test procedures to determine the maximum operating frequency of a processor. It is common practice to speed bin processors for graded pricing. As a result, even in the presence of manufacturing process variation, processors can be designed at the typical "corners", unlike ASICs, which are designed at the worst-case corners. Binning a processor also sets the expectations for the consumer about the performance that should be expected from the processor chip.

In the case of uniprocessors, the performance of a processor is strongly correlated with its frequency of operation. As a result, processors have traditionally been binned according to frequency [31]. However, for chip multiprocessors, the appropriate binning metrics are much less clear due to two main considerations.

1. If binning is done according to the highest common operating frequency of all cores (one obvious extension to the uniprocessor binning metric), good performance correlation of the binning metric would only be observed when the maximum operating frequencies of all cores are very similar. We speculate that this assumption will not hold true in the future based on the following observations.

- The transition from multi-core to many-core would mean several tens to hundreds of cores on a single die. In this case, all the cores are unlikely to have similar maximum safe operating frequencies.
- With scaling, technology process variation is increasing. There is no obvious process solution to variability in sight. ITRS [32] predicts that circuit performance variability will increase from 48% to 66% in the next ten years. Moreover, many-core die sizes may scale faster than geometric technology scaling [33], facilitated by future adoption of 450mm wafers and 3D integration. As a result, core-to-core frequency variation is likely to increase in coming technology generations.

2. The second reason why binning metrics may need to be re-evaluated for multi-core processors is that a good binning metric should not only correlate well with the maximum performance of the chip (in order to maximize producer profits and consumer satisfaction), but should also have acceptable time overhead for the binning process. As we show in this paper, different binning metrics have different binning overheads, and therefore, the tradeoff between correlation to performance and timing overhead should be evaluated carefully.

In the simplest and most general form of speed binning, speed tests are applied and outputs are checked for failure at different frequencies [34]. The testing may be structural or functional in nature [31, 35, 36]. The total test time depends on the search procedure, the number of speed bins, and the frequency distribution of the processor. To the best of our knowledge, this is the first work discussing speed binning in the context of multi-core processors.

In this paper, we make the following contributions.

- We explore, for the first time, speed binning in the context of multi-core processors.
- We propose two multi-core binning metrics and quantify their correlation with absolute performance as well as their testing time overheads for various kinds of workloads.
- We demonstrate that leveraging data from the process variation model can have a significant impact on binning efficiency and propose several variation-aware binning strategies.

Our results show that variation-aware binning strategies can reduce testing time significantly with little or no degradation in performance correlation.

3.2 Variation Model

An accurate, physically justifiable model of spatial variability is critical in reliably predicting and leveraging core-to-core variation in the binning process. Though most design-end efforts to model spatial variation have concentrated on spatial correlation (e.g., [37,38]), recent silicon results indicate that spatial dependence largely stems from across-wafer and across-field trends [39]. [40] assumes the source of core-to-core variation to be lithography-dependent across-field variation. Though a contributor, across-field variation is smaller compared to across wafer variation [41] (even more so with strong RET and advanced scanners). In light of these facts, we use a polynomial variation model [42] for chip delay, similar to those proposed in [39,41,43], having three components: (1) systematic (bowl-shaped) across wafer variation[1], (2) random core-to-core variation (arising from

[1]Example physical sources of across-wafer bowl-shaped variation include plasma etch, resist spin coat, post exposure bake [42].

random within-die variation); and (3) random die-to-die variation (e.g., from wafer-to-wafer or lot-to-lot variation).

$$\begin{aligned} V_d(x, y) =& A(X_c + x)^2 + B(Y_c + y)^2 + C(X_c + x) + \\ & D(Y_c + y) + E(X_c + x)(Y_c + y) + F + R + M \end{aligned} \tag{3.1}$$

where $V_d(x, y)$ is the variation of chip delay at die location x, y; X_c, Y_c are the wafer coordinates of the center of the die ($(0, 0)$ is center of wafer); x, y are die coordinates of a point within the die; M is the die-to-die variation and R is the random core-to-core variation. A, B, C, D, E, F are fitted coefficients for systematic across-wafer variation. We use a fitted model as above based on real silicon data from a 65nm industrial process [42][2]. The goal of the binning process is to accurately classify a chip into one of n bins (where n is decided based on business/economic reasons) in light of the above variation model.

3.3 Binning Metrics

Traditional uniprocessor binning strategies, which sort chips according to maximum operating frequency, may fail to adequately characterize multicore processors, in which within die process variation given by Equation 1 can be substantial. In this section, we propose and discuss two simple binning metrics that recognize the frequency effects of process variation. We assume that individual cores are testable and runnable at independent operating frequencies [44–49] though our discussion and analysis would continue to hold in other scenarios.

[2]For this model, mean = 4GHz, $\sigma_{bowl} = 0.128$GHz, $\sigma_R = 0.121$GHz, $\sigma_M = 0.09$GHz.

3.3.1 *Min-Max* and Σf

Min-Max stands for the minimum of the maximum safe operating frequencies for various cores of a chip multiprocessor. The *Min-Max* metric is computed using equation 3.2, where n represents the number of frequency bins (f_i is the frequency of bin i), m represents the number of processor cores, and f_{ij} is a successful test frequency or 0 if core j fails the i^{th} frequency test.

$$\begin{aligned} \textit{Min-Max} &= \min[\max[f_{ij}|_{i=1}^{n}]|_{j=1}^{m}] \quad \text{where} \\ f_{ij} &= 0 \quad \text{if processor } j \text{ fails the } i^{th} \text{ frequency test} \\ &= f_i \quad \text{otherwise} \end{aligned} \tag{3.2}$$

The second binning metric that we evaluate is Σf. While frequency represents the primary means of increasing the performance of uniprocessors, new conventional wisdom dictates that the performance of multiprocessors depends on increasing parallelism [50]. Thus, ranking processors according to maximum attainable aggregate throughput represents a fitting binning strategy. Ideally, aggregate throughput should be maximized when every core operates at its maximum frequency. Consequently, we calculate the Σf metric using equation 3.3.

$$\begin{aligned} \Sigma f &= \sum_{j=1}^{m} \max[f_{ij}|_{i=1}^{n}] \\ f_{ij} &= 0 \quad \text{if processor } j \text{ fails the } i^{th} \text{ frequency test} \\ &= f_i \quad \text{otherwise} \end{aligned} \tag{3.3}$$

3.3.2 Correlation to Throughput

In terms of correlation of the metric with the throughput of the chip, *Min-Max* is conservative and therefore, should demonstrate good correlation only for work-

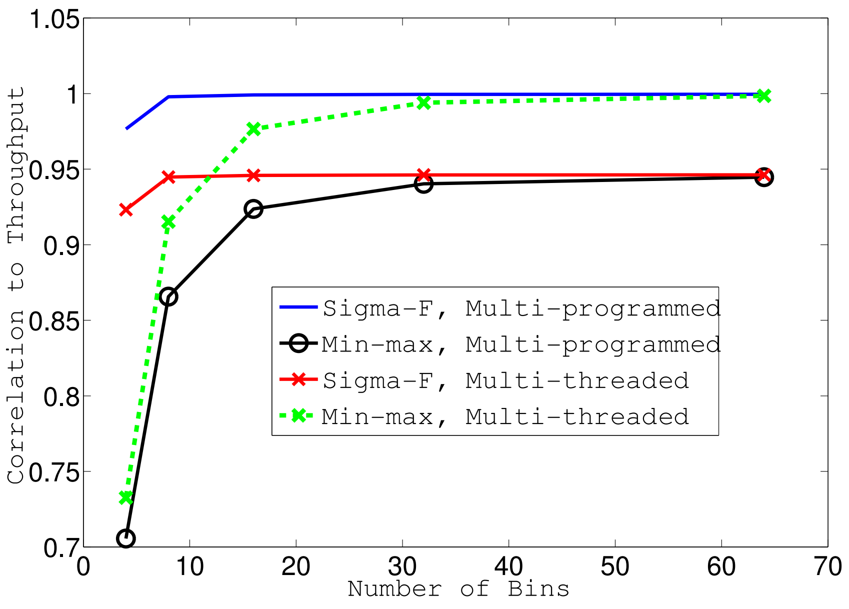


Figure 3.1: Correlation of *Min-Max* and Σf to throughput for multi-programmed and multi-threaded workloads.

loads with regular partitioning (parallel or multi-threaded workloads) in which the load is distributed evenly between all cores. For other workloads that have inherent heterogeneity (multi-programmed workloads), Σf should demonstrate good correlation, especially when runtimes are designed to take advantage of the heterogeneity inherent in systems and thread characteristics. In fact, for multi-programmed workloads, the magnitude of miscorrelation between actual throughput and Σf depends on the extent of disparity between the workloads that run on various cores. One drawback of Σf is that it may increase the binning overhead, although we show in this paper that utilizing knowledge of variation trends can help to keep the overhead in check.

Figure 3.1 compares the correlation of *Min-Max* and Σf to actual throughput for multi-programmed and multi-threaded workloads using Monte-carlo simulations on 100,000 dice, each die being a 64 core processor in 65nm technology on a 300mm wafer (please refer to section 3.5 for further details on experimental setup). It is evident that Σf is a better metric for multi-programmed workloads while *Min-Max* performs better for multi-threaded workloads for moderate to large number of bins. This is because the performance of multi-threaded benchmarks depends on the speed of the slowest executing thread (because of thread synchronizations in the benchmarks) which is nicely captured by *Min-Max*. Also, the correlation of Σf and *Min-Max* to the throughput of multi-programmed and multi-threaded workloads respectively, converges to 1 asymptotically with the number of bins. This is because, finer binning granularity leads to more precise estimation of maximum core frequencies. Conversely, when the number of bins is small, we observe rather poor performance correlation for the metrics.

To compare the two metrics, consider the asymptotic case of very large n and m and completely random core-to-core variation (i.e., A, B, C, D, E, F, M all equal zero in equation 1). In this simplified case, Σf converges to $m \times mean\_frequency$ while *Min-Max* converges to $(E(Min_{i=1...\infty} f_i) = 0$, i.e., for multi-programmed workloads, we expect the *Min-Max* to be a progressively worse metric as the number of cores in a die increases or the variation increases.

3.3.3 Binning Overhead

The binning overhead depends on the specific testing methodology that is used. On one extreme lies the case where individual cores are tested one at a time and on the other extreme is the case where all cores are tested simultaneuosly in parallel. While the latter reduces test time compared to the former, it results in higher power consumption of the circuit during test. With ever increasing number of cores within a multiprocessor, parallel testing of all cores leads to very high test power. Hence, testing is usually performed by partitioning the design into blocks and testing them one at a time [51–53]. For our work, we assume that cores are tested one at a time. Note that the analysis is also extensible to cases where a group of cores are tested together in parallel.

To calculate the binning overhead for *Min-Max* on a processor with n frequency bins and m cores, we use binary search[3] (i.e. frequency tests are applied in a binary search fashion) to find f_{max} for every core. However, the search range will reduce progressively. The worst case arises when f_{max} for every core is 1 bin size less than the f_{max} found for the previous core. In this case, the worst-case number of tests that need to be performed can be computed as $(log(n!) + m - n)$ (assuming $m \geq n$). The best case binning overhead for *Min-Max* would be m tests.

To fully evaluate the Σf metric, the maximum operating frequency of each core must be learned. Using binary search, this process performs, at worst, $m \times logn$ tests[4]. The best case is still m tests. We will show the average case runtime results of both these testing strategies using monte-carlo analysis.

It should be noted from the above discussion that the binning overhead for Σf is always equal to or higher than that of *Min-Max* and this remains true even when simple linear search (i.e. frequency tests are applied in a simple linear fashion, which is the case with most industrial testing schemes) is used instead of binary search. Moreover, the disparity between binning times for Min-Max and Σf is never higher for binary search than for linear search. For *Min-Max*, the worst case overhead is on the order of n^2 and the best case is m tests. For Σf,

[3]In this work, we assume that if a core works at a certain frequency, it is guaranteed to work at all lower frequencies. This stems from the specific case of using binary search in conjunction with the minmax metric. The constraint can be easily avoided by adding one more test per core (i.e., testing it at the minmax frequency)

[4]Note that this expression and the expressions corresponding to Min-Max ignore the bias introduced in binary search by the probability distribution of the frequencies themselves.

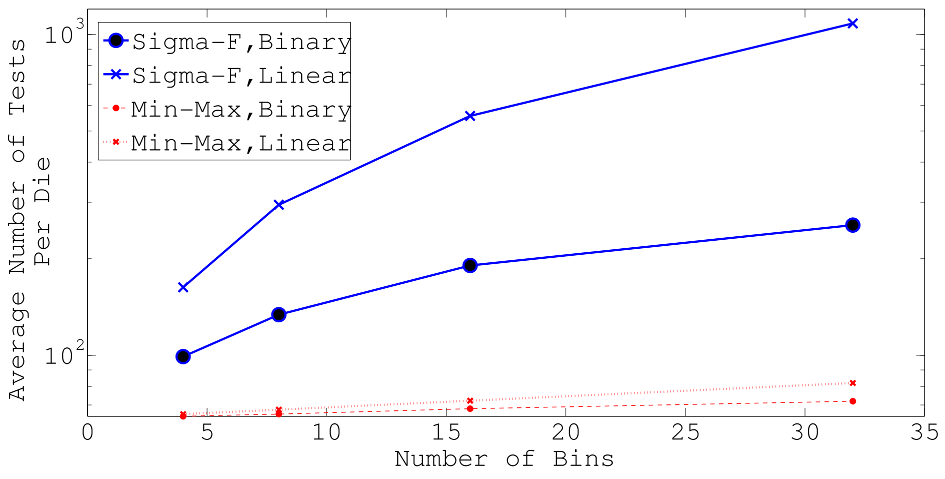


Figure 3.2: Frequency binning overhead (linear and binary search) for Σf and *Min-Max*.

the worst case number of tests is on the order of $m \times n$ and the best case is m tests. This is also shown in Figure 3.2 by performing Monte-Carlo simulations on a 64 core multi-processor in 65nm technology with a 300mm wafer. In this work, we use binary search for comparing test time overheads of various binning strategies but as explained above, our proposed analysis and results will hold for linear search as well.

3.4 Using the Variation Model to Reduce Binning Overhead

The binning metrics described above, as well as the binning strategies for those metrics, are agnostic of the process variation model. The overhead of binning using those metrics, however, depends strongly on the process variation model. In this section, we advocate the use of variation-aware binning strategies. We argue that the overhead of binning can be considerably reduced by making the binning strategies variation model-aware. The maximum safe operating frequency (f_{max}) of a core can be strongly predicted (i.e. mean with standard deviation around it) based on the process variation model. Therefore, the process variation model can give a smaller frequency range within which the search should be performed.

3.4.1 Curve Fitting

We propose curve fitting as a technique for reducing testing time overhead by trimming the range of frequencies at which a core must be tested. The curve-fitting strategy involves using the variation model (equation 1) to approximate the expected frequency (in GHz) as well as the standard deviation ($= \sqrt{(\sigma_M^2 + \sigma_R^2)}$) of a core, given its location within a die and die location within the wafer. Therefore, we can identify the center (= mean) as well as the corners ($= +/-k\sigma$) of a new, tighter search range. If the core falls outside of this range (decided by k), we assign the core to the lowest frequency bin. Curve fitting reduces both the average and worst-case testing time for each core.

3.4.2 Clustering

Another strategy for reducing the binning overhead can be to create a hybrid metric which incorporates the advantages of each of the original metrics – namely, the low testing overhead of Min-Max and the high performance correlation of Σf. This behavior can be achieved by clustering the cores in a chip multiprocessor and then using Min-Max within the clusters (low binning overhead advantage) while using Σf over all clusters (high correlation to maximum throughput advantage). To further reduce the overhead of binning, a process like curve fitting can be applied, where the process variation model is used to identify the search range for f_{max} of a core. We refer to this combination of clustering and curve fitting as *smart clustering*

In order to improve the performance correlation within the cluster and minimize the binning overhead (especially when across-wafer variations are high), clusters can be chosen intelligently to minimize frequency variation (and hence loss of correlation) within a cluster. To this end, the cluster size can be set to be inversely proportional to the spread of frequency mean (calculated from the bowl-shape in equation 1) within the cluster. In general, the dice close to the center of the bowl (typically close to the center of the wafer) will see large cluster sizes, while clusters are smaller for the dice closer to the edge of the wafer. We do not evaluate variable clustering in this paper due to the relatively low across-wafer variations that our current process variation models suggest.

3.5 Methodology

We model chip multiprocessors with various numbers of cores on the die for different technologies. Each core is a dual-issue Alpha 21064-like in-order core with 16KB, 2-way set-associative instruction cache and data cache. Each core (1 mm^2 at 65nm) on a multiprocessor has a private 1MB L2 cache ($0.33MB/mm^2$ at 65nm). We assumed a gshare branch-predictor [54] with 8k entries for all the cores. The various miss penalties and L2 cache access latencies for the simulated cores were determined using CACTI [55]. We model the area consumption of the processors for different technologies using the methodology in [56].

We considered two types of workloads – multi-programmed workloads and multi-threaded workloads. Table 3.1 lists the ten benchmarks used for constructing multi-programmed workloads and the three multi-threaded benchmarks. The benchmarks are chosen from different suites (SPEC, IBS, OOCSB, and Media-

Table 3.1: Benchmarks used

Program	Description
ammp	Computational Chemistry (SPEC)
crafty	Game Playing: Chess (SPEC)
eon	Computer Visualization (SPEC)
mcf	Combinatorial Optimization (SPEC)
twolf	Place and Route Simulator (SPEC)
mgrid	Multi-grid Solver: 3D Potential Field (SPEC)
mesa	3-D Graphics Library (SPEC)
groff	Typesetting Package (IBS)
deltablue	Constraint Hierarchy Solver (OOCSB)
adpcmc	Adaptive Differential PCM (MediaBench)
CG	Parallel Conjugate Gradient (NAS)
FT	Parallel Fast Fourier Transform (NAS)
MG	Parallel Multigrid Solver (NAS)

bench) for diversity. The parallel applications (CG, FT, MG) are chosen from the NAS benchmark suite and run to completion. The class B implementations have been used.

Multi-programmed workloads are created using the sliding window methodology in [57]. For multi-programmed workloads, the performance of a multiprocessor is assumed to be the sum of the performance of each core of the multiprocessor, derated by a constant factor. The methodology is accurate for our case, where each core is assumed to have a private L2 cache and a memory controller [56]. The methodology was shown to be reasonable for our benchmarks even for processors with shared L2 [56], due to the derating factor.

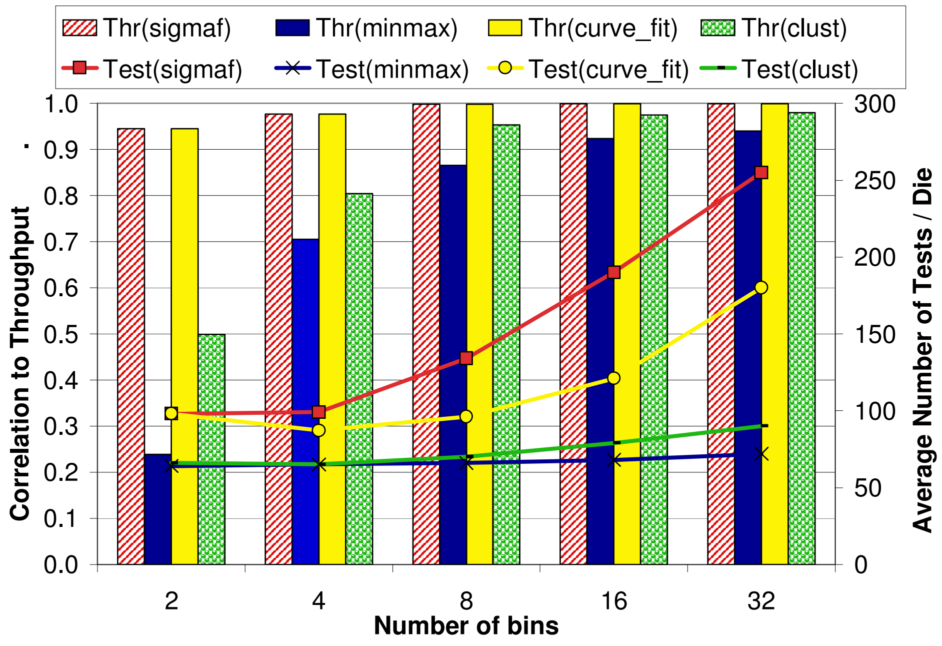
After fast-forwarding an appropriate number of instructions [58], multi-programmed simulations are run for 250 million cycles. As mentioned before, parallel applications are run to completion. The frequency of each core is determined by the variation model. Simulations use a modified version of SMTSIM [57].

3.6 Analysis of Results

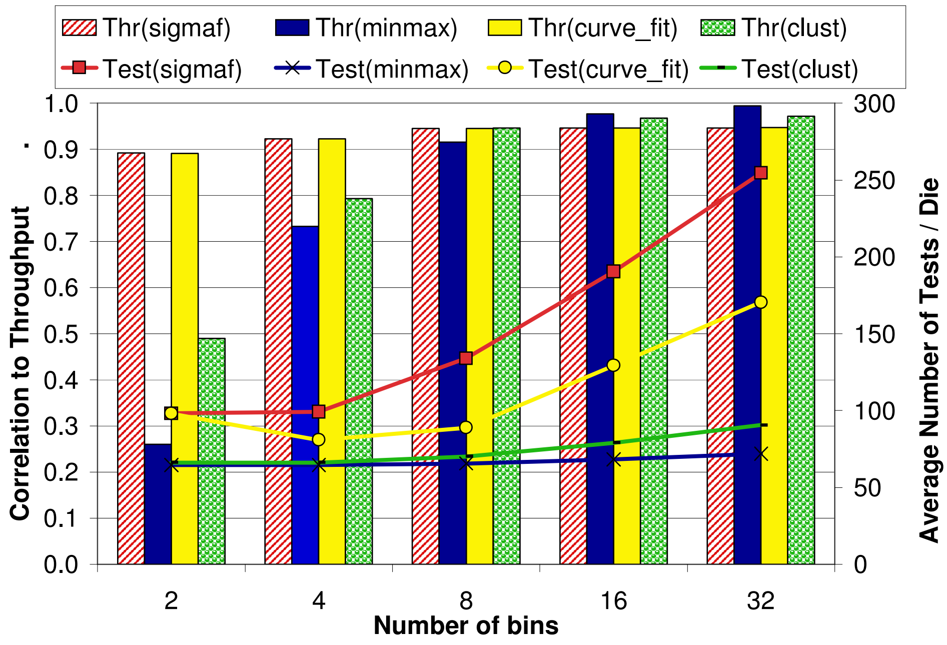
In this section, we compare the binning metrics and the various evaluation strategies in terms of their overheads as well as their correlation to throughput. We run Monte-Carlo simulations using 100,000 dice. Unless specified otherwise, each die is a 64-core processor (256 mm^2) in a 65nm technology 300mm wafer, binned using 8 frequency bins. *Curve fitting* and *smart clustering* use a search range of $\pm 3\sigma$ (where σ accounts for the random die to die and within die variations), while Σf and the baseline *clustering* approach search the entire frequency range for f_{max}. We use the process variation model as described by Equation 1, with $\sigma_{bowl} = 0.128$GHz, $\sigma_R = 0.121$GHz, $\sigma_M = 0.09$GHz, based on a fitted model from a 65nm industrial process.

3.6.1 Dependence on Number of Bins

Figure 3.3 shows how binning overhead and throughput correlation vary with the number of frequency bins for multi-programmed (Fig. 3.3(a)) and multi-threaded (Fig. 3.3(b)) workloads. Using 100,000 data points (processor dice), we calculate correlation between the average of the maximum throughput of the various workloads on a processor (where cores run at different frequencies dictated by the variation model) and the value of the metric when following a given binning strategy. Note that performance of a thread often does not vary



(a) multi-programmed



(b) multi-threaded

Figure 3.3: Correlation of various binning metrics to actual throughput and their binning overhead for varying number of bins.

linearly with frequency due to pipeline hazards, memory accesses, etc., so it is unlikely that correlation will be 1 for any binning metric.

There are several things to note in these graphs.

- *First, Σf achieves significantly better correlation to throughput than* Min-Max *for multi-programmed workloads.* This is not surprising, considering that the throughput of a thread often depends on the frequency of the core it is running on, and for multi-programmed workloads, every thread execution is independent. *Min-Max* fails to account for variation in frequency (and therefore, average throughput) between individual cores.

- While the correlation of Min-Max to throughput suffers for multi-programmed workloads, *Min-Max actually surpasses Σf for multi-threaded benchmarks as the number of bins increases.* This is due to the fact that synchronization in the parallel benchmarks causes performance to be constrained by the slowest thread, since faster threads must wait at synchronization points until all threads have arrived.

- Correlation is especially low for a small number of frequency bins. This is because the binning process picks an overly conservative frequency as f_{max} for a die in that case. Even the relative performance of *Min-Max* (as compared to Σf) worsens as the number of frequency bins is decreased.

- *In terms of binning overhead,* Min-Max *is significantly faster than Σf, especially for large number of bins (70% faster for 32 bins).* This is because while Σf involves doing binary search over the full frequency range (over all frequency bins) for every core, *Min-Max* progressively reduces the search range and requires very few tests per core, on average. *minmax* and Σf

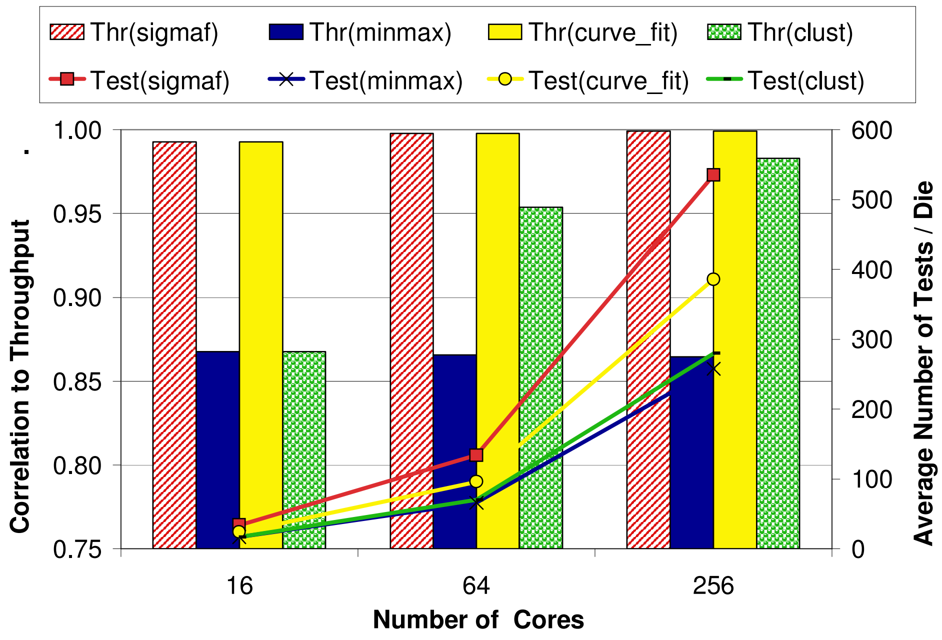
have comparable overheads for small number of bins since the search range is reduced.

- The graph also shows that $curve\_fit$ *(the approach of using variation model aware curve fitting to approximate* Σf*) has performance correlation to throughput that is equivalent to that of* Σf. This is because a range of 6σ ($\pm 3\sigma$) is searched for $curve\_fit$, which is often big enough to allow the discovery of the true f_{max} of a core. *In terms of binning overhead,* $curve\_fit$ *is significantly faster than* Σf *(36% for our baseline architecture).* This is because the range of frequencies that are searched for $curve\_fit$ is directed by the variation model and is therefore, relatively small. Overhead is greater than that for *Min-Max* because of the need to estimate the f_{max} for every core.

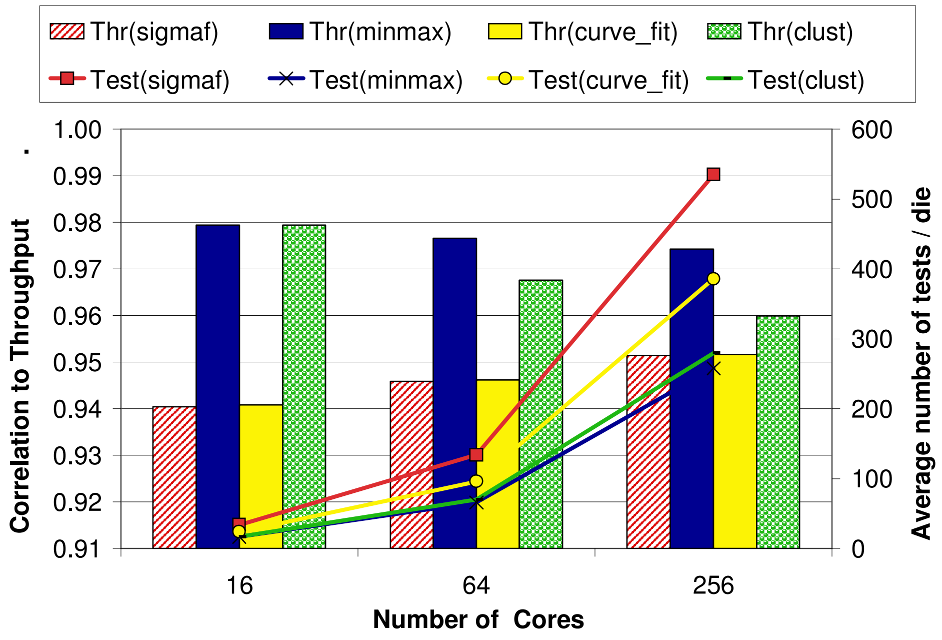
- *Clustering-based strategies (the approach of using clustering to approximate* Σf*) result in a smaller binning overhead than* $curve\_fit$ *(26% for the baseline, results are shown for a cluster size of 16).* Clustering that relies on the variation model to reduce the search range for f_{max} of the cores (*smart clust*) is faster than the naive approach that performs search over the full range for all cores (6% improvement in test time for the baseline case). In terms of correlation to throughput, clustering-based strategies lie between Σf and *Min-Max* for both types of workloads. This is not surprising, considering that clustering represents a hybrid between the two schemes.

3.6.2 Dependence on Number of Cores

Figure 3.4 shows how correlation and binning overhead change with the number of cores on the processor dice. The results are shown for 16 frequency bins. There are several things to note from these graphs.



(a) multi-programmed



(b) multi-threaded

Figure 3.4: Correlation of various binning metrics to actual throughput and their binning overhead for varying number of cores in the multi-processor.

- For multi-programmed workloads, the correlation to throughput increases with the number of cores for both clustering-based strategies. Better correlation with more cores is a result of having a fixed cluster size, which results in a larger number of clusters per chip (note that with more clusters, the granularity of clustering becomes finer). To confirm this, we also performed experiments to see how the correlation and binning overhead change when the number of cores per cluster (and, therefore, the number of clusters) is changed for a fixed sized chip (with 64 cores). Figure 3.5 shows the results. We indeed observe that the binning overhead of clustering decreases with increasing number of cores per cluster. Similarly, the correlation to throughput decreases for multi-programmed workloads with increasing cores per clusters.

- Interestingly, the roles of the metrics are reversed for multi-programmed and multi-threaded workloads. While Σf and curve fitting do well for multi-programmed workloads, Min-Max and clustering do better for multi-threaded workloads. This reversal can be explained by the fact that Σf and curve fitting (a close approximation) characterize the maximum throughput of a die, which is strongly correlated to performance for multi-programmed workloads. However, when workload performance correlates more strongly to the performance of the weakest core, Min-Max wins out. Since clustering uses the Min-Max metric as its backbone, trends for clustering are similar to those for Min-Max.

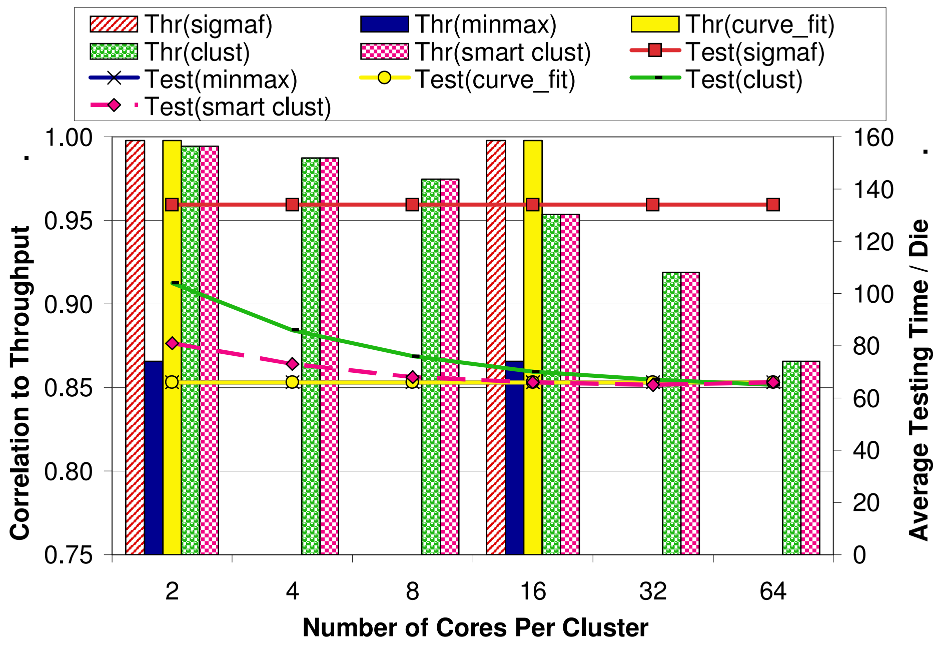
- As the number of cores per cluster increases, we see an interesting difference between the two types of clustering for multi-threaded benchmarks. *For clustering that bounds the search range based on perceived variation (smart clust), throughput correlation levels off and begins to decrease as the number* *of cores per cluster becomes large.* This is because the limited search range may not be wide enough to capture the variation range in a large cluster. However, when the entire search range is considered, correlation continues to increase even as the number of cores per cluster increases. This is because performance is correlated to the performance of the slowest core on the die for our multi-threaded benchmarks, and larger clusters result in less over-estimation of performance for a processor running such benchmarks.

3.6.3 Dependence on Search Range for Variation-Model Aware Approaches

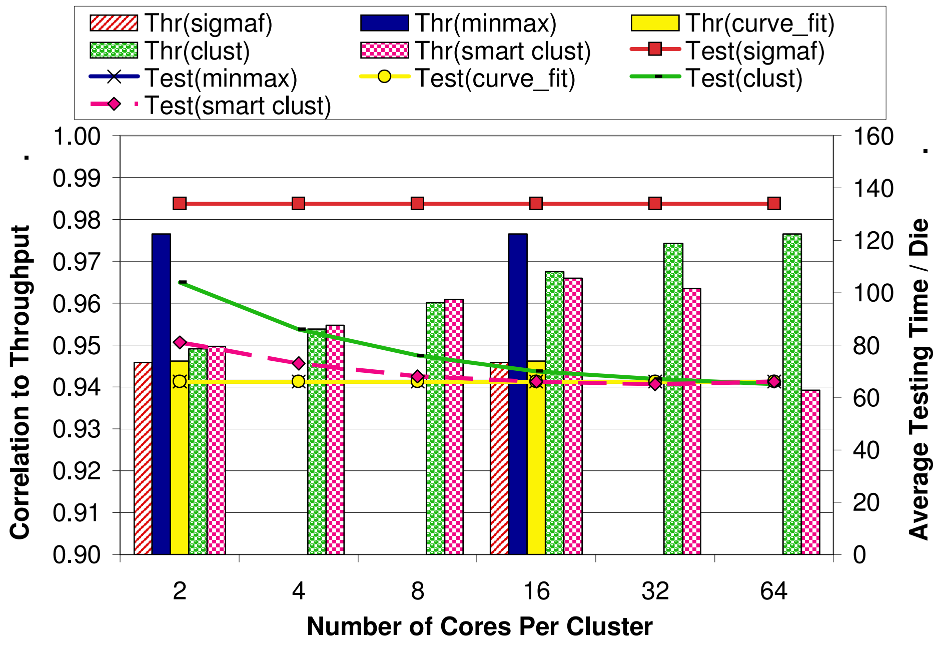
Figure 3.6 shows how performance correlation and binning overhead change as the search range is varied for 8 and 64 frequency bins (we only show the results for multi-programmed workloads as multi-threaded benchmarks behave similarly). Both techniques that rely on the variation model to come up with aggressive search ranges ($curve\_fit$ and *smart clust*) have better correlation as the search range is increased. The improvement is higher for larger number of frequency bins. For example, when moving from 2σ to 3σ, correlation to throughput for curve fitting improves by 30% for 64 bins but just by 6% for 8 bins. However, the increase in binning overhead is also higher for a larger number of bins. *Therefore, unless the variation is large enough to justify an increase in the bin count, fixed search range of 2σ or 3σ is good enough.*

3.6.4 Dependence on Nature of Variations

In Figure 3.7, we show the effect that the nature of variations has on binning metrics and their evaluation. The four cases: *baseline* (incorporates all variation model components), *only inter-core random*, *only inter-die random*, and

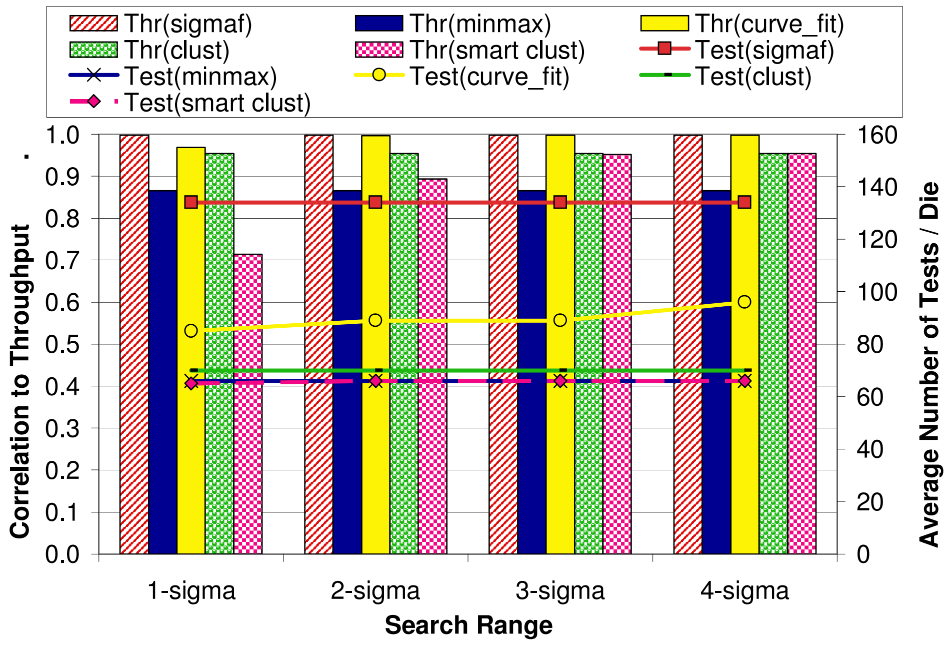


(a) multi-programmed

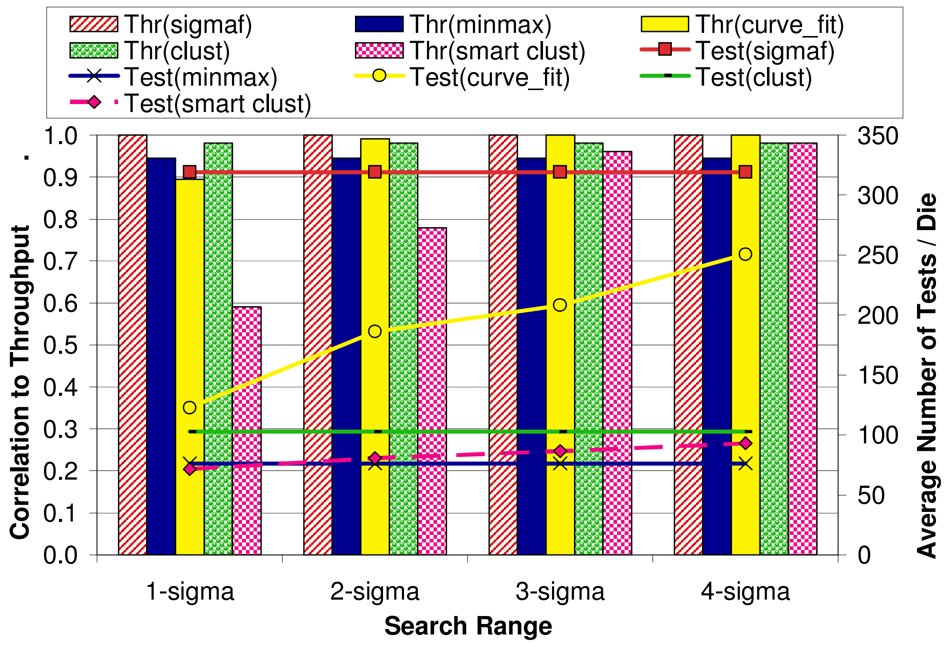


(b) multi-threaded

Figure 3.5: Correlation of various binning metrics to actual throughput and their binning overhead for varying number of cores per cluster.

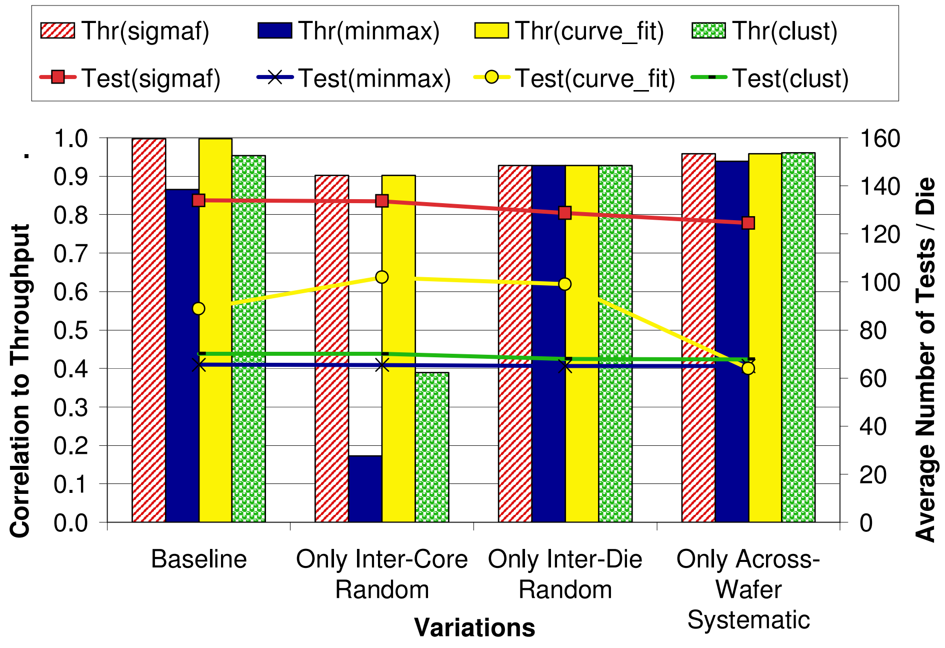


(a) 8 frequency bins

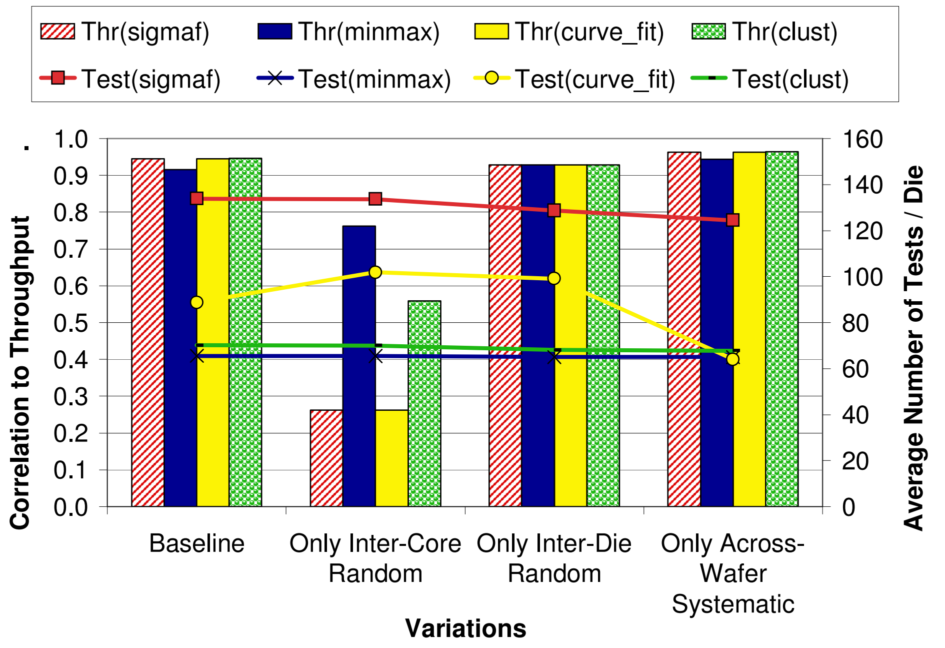


(b) 64 frequency bins

Figure 3.6: Correlation of various binning metrics to actual throughput and their binning overhead for varying search range.



(a) multi-programmed



(b) multi-threaded

Figure 3.7: Correlation of various binning metrics to actual throughput and their binning overhead for different process variation scenarios.

only across-wafer systematic (i.e., the bowl-shaped variation) all have the same variance. As within-die (i.e. core-to-core) variation increases, the correlation of *Min-Max* to the throughput of multi-programmed workloads decreases, since it grossly underestimates throughput (because it takes the minimum f_{max} of all cores). However, for multi-threaded workloads, Σf shows poor performance correlation when inter-core variation dominates, since it overestimates the throughput of the processor. *Therefore, increase in random core to core variation magnifies the difference between the two metrics with the workload types. This implies that in such a variation scenario, choice of metric will strongly depend on the expected workload type.* Note that variation-aware binning strategies that use the variation model for prediction (i.e., curve fitting) achieve maximum reduction of binning overhead in cases where there is systematic variation (*baseline* and *only across-wafer systematic*).

3.7 Conclusion

In this paper, we have studied for the first time, speed binning for multi-core processors. We have compared two intuitive metrics – *Min-Max* and Σf – in terms of their correlation to actual throughput for various kinds of workloads as well as their testing overheads. Furthermore, we have proposed binning strategies which leverage the extent of variation (clustering) as well as the partially systematic nature of variation (curve fitting). From our analysis, we conclude the following

- In terms of correlation to actual throughput, Σf is an overall better metric except for two cases where *Min-Max* performs well: 1) multi-threaded benchmarks, with large number of bins (larger than 8) and, 2) multi-threaded benchmarks when within-die variations are dominant. However, *Min-Max* has a significantly lower binning overhead than Σf (lower by as much as 70%).

- Clustering based strategies which are a hybrid of Σf and *Min-Max* reduce the binning overhead by as much as 51% with a small loss (5% points for 8 bins) in correlation to throughput.

- Variation-model aware strategies help in reducing the binning overhead significantly with the same correlation to throughput as Σf. Variation aware curve fitting reduces the binning overhead by as much as 36%.

Our overall conclusion is that uniprocessor binning methods do not scale well for multi-core processors in the presence of variations. Multi-core binning metrics and testing strategies should be carefully chosen to strike a good balance between goodness of the metric and time required to evaluate it. Most importantly, the efficiency of speed binning can be improved significantly by leveraging process variation knowledge to optimize the binning procedure.

In some cases, power and memory/cache size are also important binning metrics. For low power embedded applications where power is an equally important metric as performance, the same notion of binning can be employed to categorize processors. The variation model can be used to bin processors based on power dissipation. The concept of voltage binning [59] [34] can be extended for multicore processors by making use of similar techniques as suggested in this paper. This is part of our ongoing work on efficient characterization of multicore processors.

CHAPTER 4

Conclusions

In this work, we have shown that the notion of a flexible hardware-software interface can significantly help in combating the effects of manufacturing variations. Traditional approaches have always assumed software to be fixed, thus demanding a certain minimum level of performance out of the hardware. Meeting these minimum performance levels in presence of process variations results in a significant design overhead and resource wastage. We show that, by adapting the software application to the post manufacturing performance-power characteristics of the hardware across different die, it is possible to relax design constraints with the same manufacturing yield and application quality. In another area, we show that binning multi-core processors depending on the kind of workload and levaraging the information from the variation model can help in maximizing the profit with the added benefit of a much lower test time.

This work demonstrates the benefits of software adaptation on a single application system. Most systems today run multiple applications possibly mediated by an operating system. Therefore, it is quite intuitive to extend the notion of adaptation to multi-application scenarios. This scenario will further benefit from the trade-offs that will exist among individual applications, thus providing a greater potential to impact performance, power and quality. Extending this notion of adaptation to operating system power management policies can dramatically improve power savings by knowing the exact chip specific power-performance trade-offs.

This work also demonstrates how manufacturing profits can be maximized by binning multi-core processors accurately, that depends on the workload that they execute. In this work, we touched upon the traditional notion of frequency aspects of binning. However, multi-core performance is strongly related to other aspects like memory. Moreover, power dissipation is also an important performance metric. Both memory and power are subject to process variability and therefore, it is but natural to include these quantities during binning. This indicates an area of potential future research.

Another potential area of impact of this hardware-software interaction is related to the design cycle of complex systems. The manufacturing process model that is used during design matures over course of time. Therefore, the models that are used during architectural design are not the same that are used during placement and routing. Therefore, the optimum set of decisions taken at the architectural level may not remain optimum at the routing level. As part of our future work, we are investigating a soft architectural constraints guided synthesis, place and route system that tries to optimize the design flow at every stage knowing the most recent input models.

References

[1] S. Borkar, "Parameter Variations and Impact on Circuits and Microarchitecture," C2S2 Marco Review, 2003. 1, 2.1

[2] "Process Integration, Devices and Structures, ITRS," 2007. 1, 2.1

[3] Y. Cao, P. Gupta, A. Kahng, D. Sylvester, and J. Yang, "Design Sensitivities to Variability: Extrapolations and Assessments in Nanometer VLSI," *ASIC/SOC Conference,15th Annual IEEE International*, 2002. 1, 2.1

[4] S. R. Nassif, "Modeling and Forecasting of Manufacturing Variations," in *Fifth International Workshop on Statistical Metrology*, 2000. 1, 2.1

[5] J. Tschanz, "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," *ISSCC*, 2002. 2.1

[6] S. Sen, V. Natarajan, R. Senguttuvan, and A. Chatterjee, "Pro-VIZOR: Process Tunable Virtually Zero Margin Low Power Adaptive RF for Wireless Systems," in *Design Automation Conference*, 2008. 2.1

[7] D. Ernst, N. S. Kim, S. Das, S. Pant, T. Pham, R. Rao, C. Ziesler, D. Blaauw, T. Austin, and T. Mudge, "Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation," in *Micro Conference*, 2003. 2.1

[8] N. Shanbhag, "A Mathematical Basis for Power-Reduction in Digital VLSI Systems," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 44, no. 11, pp. 935–951, Nov 1997. 2.1

[9] P. B. Bhat, V. K. Prasanna, and C. Raghavendra, "Adaptive Communication Algorithms for Distributed Heterogeneous Systems," *Intl. Symposium on High-Performance Distributed Computing*, 1998. 2.1

[10] S. Sampei, S. Komaki, and N. Morinaga, "Adaptive Modulation/TDMA Scheme for Personal Multimedia Communication Systems," in *GLOBECOM*, 1994. 2.1

[11] X. Qiu and K. Chawla, "On the Performance of Adaptive Modulation in Cellular Systems," *Communications, IEEE Transactions on*, 1999. 2.1

[12] S. Chandra, K. Lahiri, A. Raghunathan, and S. Dey, "System-on-Chip Power Management Considering Leakage Power Variations," in *DAC*. New York, NY, USA: ACM, 2007. 2.1

[13] G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. I. August, "SWIFT: Software Implemented Fault Tolerance," in *CGO*, 2005. 2.1

[14] G. V. Varatkar and N. R. Shanbhag, "Energy-efficient Motion Estimation Using Error-tolerance," in *ISLPED '06*. ACM, 2006. 2.1

[15] V. J. Reddi, M. S. Gupta, S. Campanoni, M. D. smith, G. Wei, and D. Brooks, "Software-Assisted Hardware Reliability: Abstracting Circuit-level Challenges to the Software Stack," in *DAC*, 2009. 2.1

[16] J. Choi, C.-Y. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal-aware Task Scheduling at the System Software level," in *ISLPED '07*. ACM, 2007. 2.1

[17] H. Chung and A. Ortega, "Analysis and Testing for Error Tolerant Motion Estimation," in *DFT*, Oct. 2005. 2.1

[18] M. A. Breuer, "Intelligible Test Techniques to Support Error-Tolerance," *Asian Test Symposium*, vol. 0, 2004. 2.1

[19] A. B. Kahng, S. Kang, R. Kumar, and J. Sartori, "Designing a Processor from the Ground Up to Allow Voltage/Reliability Trade-offs," *HPCA*, 2010. 2.1

[20] B. Foo, Y. Andreopoulos, and M. van der Schaar, "Analytical Complexity Modeling of Wavelet-Based Video Coders," 2007. 2.2.1

[21] http://docs.sun.com/source/816-5772-11/funct.html. 2

[22] H. M. Saibal Mukhopadhyay, Kunhhyuk Kang and K. Roy, "Reliable and Self-Repairing SRAM in Nanoscale Technologies using Leakage and Delay Monitoring," in *IEEE International Test Conference*, 2005. 2.2.3

[23] "Joint Video Team Reference Software JM 15.0," http://iphome.hhi.de/suehring/tml/. 2.3.1

[24] G. Sullivan and T. Wiegand, "Video Compression - From Concepts to the H.264/AVC Standard," *Proceedings of the IEEE*, 2005. 2.3.1

[25] "H.264/MPEG-4 AVC Reference Software Manual," http://iphome.hhi.de/suehring/tml/JM(JVT-X072).pdf/. 2.3.1

[26] D. Marpe, H. Schwarz, and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, 2003. 2.1

[27] K. Jeong, A. B. Kahng, and K. Samadi, "Quantified Impacts of Guardband Reduction on Design Process Outcomes," *Quality Electronic Design, International Symposium on*, vol. 0, pp. 790–797, 2008. 2.3.1

[28] M. Elgebaly, A. Fahim, I. Kang, and M. Sachdev, "Robust and Efficient Dynamic Voltage Scaling Architecture," *SOC Conference*, 2003. 2.3.3

[29] M. Elgebaly and M. Sachdev, "Variation-Aware Adaptive Voltage Scaling System," *VLSI Sys., IEEE Tran. on*, 2007. 2.3.3

[30] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, 1959. [Online]. Available: http://jmvidal.cse.sc.edu/library/dijkstra59a.pdf 2.4.1

[31] D. Belete, A. Razdan, W. Schwarz, R. Raina, C. Hawkins, and J. Morehead, "Use of dft techniques in speed grading a 1ghz+ microprocessor," in *ITC '02: Proceedings of the 2002 IEEE International Test Conference.* Washington, DC, USA: IEEE Computer Society, 2002, p. 1111. 3.1, 3.1

[32] D. P. Darcy and C. F. Kemerer, "The international technology roadmap for semiconductors," *Semiconductor Industry Association*, p. 19, 1999. 1

[33] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, no. 4, pp. 23–29, 1999. 1

[34] S. Paul, S. Krishnamurthy, H. Mahmoodi, and S. Bhunia, "Low-overhead design technique for calibration of maximum frequency at multiple operating points," in *ICCAD '07: Proceedings of the 2007 IEEE/ACM international conference on Computer-aided design.* Piscataway, NJ, USA: IEEE Press, 2007, pp. 401–404. 3.1, 3.7

[35] B. D. Cory, R. Kapur, and B. Underwood, "Speed binning with path delay test in 150-nm technology," *IEEE Des. Test*, vol. 20, no. 5, pp. 41–45, 2003. 3.1

[36] J. Zeng, M. Abadir, A. Kolhatkar, G. Vandling, L. Wang, and J. Abraham, "On correlating structural tests with functional tests for speed binning of high performance design," in *ITC '04: Proceedings of the International Test Conference on International Test Conference.* Washington, DC, USA: IEEE Computer Society, 2004, pp. 31–37. 3.1

[37] J. Xiong, V. Zolotov, and L. He, "Robust extraction of spatial correlation," in *ISPD '06: Proceedings of the 2006 international symposium on Physical design.* New York, NY, USA: ACM, 2006, pp. 2–9. 3.2

[38] F. Liu, "A general framework for spatial correlation modeling in vlsi design," in *DAC '07: Proceedings of the 44th annual Design Automation Conference*. New York, NY, USA: ACM, 2007, pp. 817–822. 3.2

[39] P. Friedberg, W. Cheung, and C. J. Spanos, "Spatial modeling of micron-scale gate length variation," I. Emami, K. W. Tobin, and Jr., Eds., vol. 6155, no. 1. SPIE, 2006, p. 61550C. [Online]. Available: http://link.aip.org/link/?PSI/6155/61550C/1 3.2

[40] E. Humenay, D. Tarjan, and K. Skadron, "Impact of process variations on multicore performance symmetry," in *DATE '07: Proceedings of the conference on Design, automation and test in Europe*. San Jose, CA, USA: EDA Consortium, 2007, pp. 1653–1658. 3.2

[41] K. Qian and C. J. Spanos, "A comprehensive model of process variability for statistical timing optimization," V. K. Singh and M. L. Rieger, Eds., vol. 6925, no. 1. SPIE, 2008, p. 69251G. [Online]. Available: http://link.aip.org/link/?PSI/6925/69251G/1 3.2

[42] L. Cheng, P. Gupta, C. Spanos, K. Qian, and L. He, "Physically justifiable die-level modeling of spatial variation in view of systematic across wafer variability," in *DAC '09: Proceedings of the 46th Annual Design Automation Conference*. New York, NY, USA: ACM, 2009, pp. 104–109. 3.2, 1, 3.2

[43] B. E. Stine, D. S. Boning, and J. E. Chung, "Analysis and decomposition of spatial variation in integrated circuit processes and devices," *IEEE Transactions on Semiconductor Manufacturing*, vol. 10, pp. 24–41, 1997. 3.2

[44] M. C. S. J. N. B. D. W. M. B. S. M. E. F. J. Dorsey, S. Searles and R. Kumar, "An integrated quad-core opteron processor," in *Proceedings of the International Solid-State Circuits Conference (ISSCC 2007)*, 2007. 3.3

[45] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *ISLPED '07: Proceedings of the 2007 international symposium on Low power electronics and design*. New York, NY, USA: ACM, 2007, pp. 38–43. 3.3

[46] C. Isci, A. Buyuktosunoglu, C. yong Cher, P. Bose, and M. Martonosi, "An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget," in *in Proc. Intl Symp. Microarch. (MICRO*, 2006, pp. 347–358. 3.3

[47] P. Juang, Q. Wu, L. shiuan Peh, M. Martonosi, and D. W. Clark, "Coordinated, distributed, formal energy management of chip multiprocessors," in

In ISLPED 05: Proceedings of the 2005 International Symposium on Low Power Electronics and Design, 2005, pp. 127–130. 3.3

[48] J. Sartori and R. Kumar, “Proactive peak power management for many-core architectures,” 2009. 3.3

[49] ——, “Three scalable approaches to improving many-core throughput for a given peak power budget,” 2009. 3.3

[50] K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, and K. A. Yelick, “The landscape of parallel computing research: a view from berkeley,” Electrical Engineering and Computer Sciences, University of California at Berkeley, Tech. Rep. UCB/EECS-2006-183, December 2006. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf 3.3.1

[51] Y. Bonhomme, P. Girard, C. Landrault, and S. Pravossoudovitch, “Test power: a big issue in large soc designs,” in *DELTA ’02: Proceedings of the The First IEEE International Workshop on Electronic Design, Test and Applications (DELTA ’02)*. Washington, DC, USA: IEEE Computer Society, 2002, p. 447. 3.3.3

[52] P. Girard, “Survey of low-power testing of vlsi circuits,” *IEEE Des. Test*, vol. 19, no. 3, pp. 82–92, 2002. 3.3.3

[53] N. Nicolici and B. Al-Hashimi, *Power-Constrained Testing of VLSI Circuits.* Kluwer Academic, 2003. 3.3.3

[54] M. S., “Combining branch predictors,” Digital Western Research Laboratory, Tech. Rep., 1993. 3.5

[55] S. J. E. Wilton and N. P. Jouppi, “Cacti: An enhanced cache access and cycle time model,” *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 677–688, 1996. 3.5

[56] R. Kumar and D. M. Tullsen, “Core architecture optimization for heterogeneous chip multiprocessors,” in *International Conference on Parallel Architectures and Compilation Techniques, PACT*. ACM Press, 2006, pp. 23–32. 3.5, 3.5

[57] D. M. Tullsen, “Simulation and modeling of a simultaneous multithreading processor,” in *Int. CMG Conference*, 1996, pp. 819–828. 3.5

[58] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, "Automatically characterizing large scale program behavior," in *ASPLOS-X: Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*. New York, NY, USA: ACM, 2002, pp. 45–57. 3.5

[59] J. Tschanz, K. Bowman, and V. De, "Variation-tolerant circuits: circuit solutions and techniques," in *DAC '05: Proceedings of the 42nd annual Design Automation Conference*. New York, NY, USA: ACM, 2005, pp. 762–763. 3.7