Software Adaptation in Quality Sensitive Applications to Deal With Hardware Variability

Aashish Pant, Puneet Gupta, and Mihaela van der Schaar
Department of Electrical Engineering, University of California Los Angeles
Los Angeles, California, USA
apant@ee.ucla.edu, puneet@ee.ucla.edu, mihaela@ee.ucla.edu

ABSTRACT

In this work, we propose a method to reduce the impact of process variations by adapting the application's algorithm at the software layer. We introduce the concept of hardware signatures as the measured *post manufacturing hardware characteristics* that can be used to drive software adaptation across different die. Using H.264 encoding as an example, we demonstrate significant yield improvements (as much as 40% points at 0% over-design), a reduction in over-design (by as much as 10% points at 80% yield) as well as application quality improvements (about 2.6dB increase in average PSNR at 80% yield). Further, we investigate implications of limited information exchange (i.e. signature measurement granularity) on yield and quality. We show that our proposed technique for determining optimal signature measurement points results in an improvement in PSNR of about 1.3dB over naive sampling for the H.264 encoder. We conclude that hardware-signature based application adaptation is an easy and inexpensive (to implement), better informed (by actual application requirements) and effective way to manage yield-cost-quality tradeoffs in application-implementation design flows.

Categories and Subject Descriptors: B.8.1

General Terms: Design, Performance, Reliability

Keywords: Variability, Hardware Software Interface, Adaptation

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.

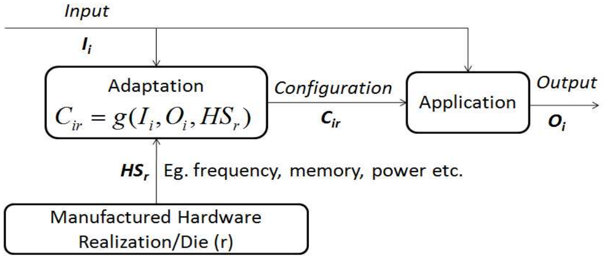

Input
I_i
Adaptation
$C_{ir} = g(I_i, O_i, HS_r)$
Configuration
C_{ir}
Application
Output
O_i
HS_r
Eg. frequency, memory, power etc.
Manufactured Hardware Realization/Die (r)

Figure 1: Simplified application adaptation model with hardware signatures. Hardware signatures are the post manufacturing power-performance numbers of the die made known to the software application for adaptation. $g()$ is the adaptation function.

In this work, we propose to mitigate the impact of process variations through software adaptation for quality sensitive applications as shown in Figure 1. We show that, by adapting the application to the post manufacturing performance-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 hard-

GLSVLSI'10, May 16–18, 2010, Providence, Rhode Island, USA.
Copyright 2010 ACM 978-1-4503-0012-4/10/06 ...$10.00.

ware 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, we introduce the concept of hardware signatures based adaptation in quality sensitive applications and then describe the general idea. In section 3, we apply this methodology to an H.264 encoder and demonstrate the improvements. In section 4, we discuss the effects of signature discretization and present an algorithm to determine optimal signature measurement points. We conclude in section 5.

2. HARDWARE SIGNATURE BASED ADAPTATION

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

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,

$$\begin{aligned} & c_{best} = argmax_c(Q_c) \\ & where\ c \in \ \{\ set\ of\ all\ configurations\ \} \\ & T_c \leq T_{MAX} \end{aligned} \tag{1}$$

We model the behavior of such a system by means of a *Quality-Complexity (Q-C) curve* (see Figure 2(a)) (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.

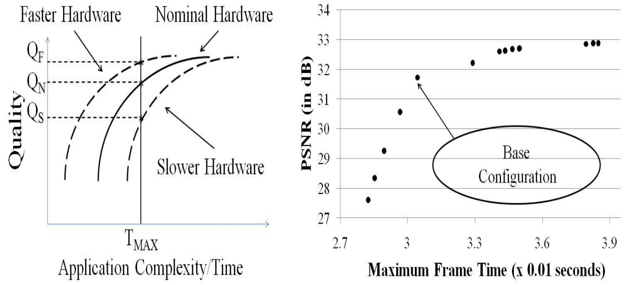
Because of process variations, every manufactured die is different. The direction and magnitude of each point shift on the Q-C graph depends on the relative contribution of various constituent functional blocks in that application configuration and the magnitude of process variations for each of these 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(a)).

If the underlying application is unaware of such Q-C graph perturbations (as in present day systems), the way it solves (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 1 pictorially depicts the proposed hardware-aware adaptation model. Next, we present a generalized description of the idea.

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

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



(a) (b)

Figure 2: (a) Q-C curve for a one-component hardware undergoes a scaled horizontal shift with frequency variations, (b) Q-C curve for H.264 encoder showing the various operating configurations.

redistribution of workloads among hardware components according to variation map) or at a slight loss in quality. (We demonstrate these benefits in section 3 in Figure 3(a) 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).

2.3 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 computation 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.

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.

Table 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

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 (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 $\{t_{M.E}, t_{T.X}, t_{E.C}\}$.

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

3.2 Results

In Figure 3(a), we show how the encoder PSNR changes with variation in operating frequency[6]. As frequency reduces, the non adaptive encoder violates the time constraint

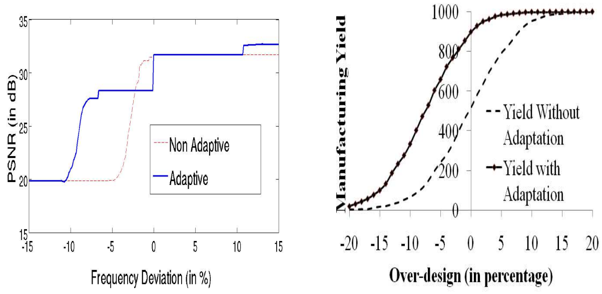
[2] Most modern chips already contain several such EEPROM or NVRAM components for storing hardware IDs, time, etc (e.g., see [21])

[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

[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



(a) (b)

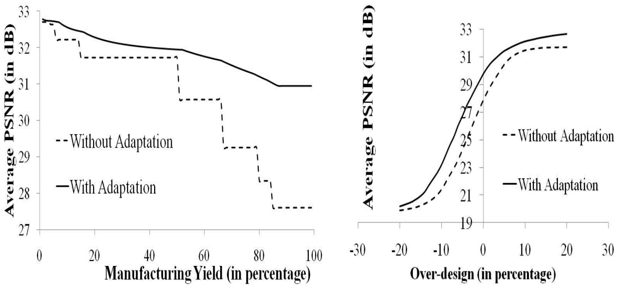
Figure 3: (a) Signature based adaptation achieves better PSNR at a given frequency of operation compared to the non-adaptive case, (b) Manufacturing yield is defined as the percentage of die that ensure no frame loss.

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 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 3(b) 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 3(b), 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%.



(a) (b)

Figure 4: (a) PSNR Vs. Yield for 0% over-design, (b) Average PSNR over all die samples.

Figure 4(a) shows how average PSNR across all die falls as one aims for a 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 4(b), 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 almost the entire over-design range.

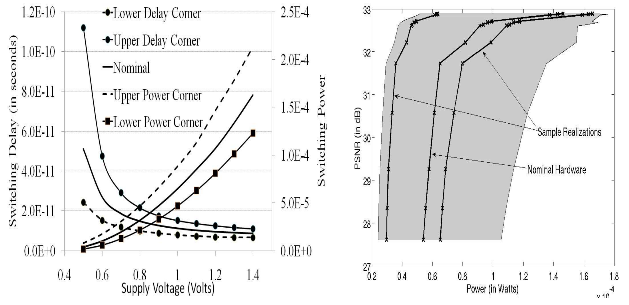
3.3 DVS: Power and Voltage as Hardware Signatures

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 5(a) 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 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,

[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

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

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



(a) (b)

Figure 5: (a) Variation of frequency and power with supply voltage under process variations, (b) Variation space of the PSNR vs power curves for some sample hardware realizations under process variations for H.264 encoder.

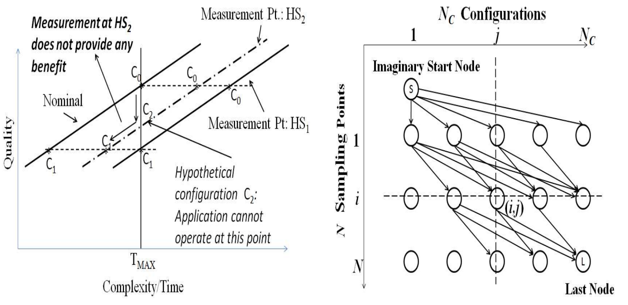
to achieve the same switching delay of $20ns$, the switching power spans from 13μW to 25.5μW (i.e. 68% of the nominal power of 18.21μW at $20ns$). By knowing the exact power-performance numbers for a die, adaptation algorithms like DVS (dynamic voltage scaling) that try to optimize on a combined performance-power-quality 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 5(b) plots the PSNR vs power trade-off for various hardware realizations for the encoder configurations of Figure 2(b). 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.

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.



(a) (b)

Figure 6: (a) Signature measurement point analysis using Q-C curves, (b) Mapping the optimal signature location problem to a shortest path problem.

4.1 Optimal Signature Measurement

In Figure 6(a), 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.

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

$$cost^{(i2,j2)}_{(i1,j1)} = \infty \;\; for\ j2 \leq j1\ or\ i2 \neq i1 + 1$$

$$= \sum_{l=j1+1}^{j2} ((Q_l - Q_{j2}) \int_{X_{l-1}}^{X_l} p(x)\, dx), \;\; otherwise$$

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

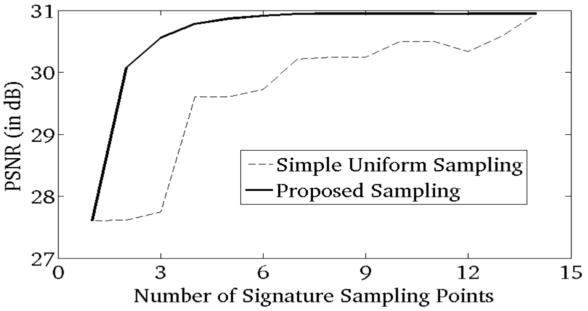


Figure 7: Improvement in PSNR with finer signature granularity.

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.

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(b) 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 7, 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.

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.

6. REFERENCES

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

[2] 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.

[3] "Process Integration, Devices and Structures, ITRS," 2007.

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

[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.

[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.

[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.

[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, pp. 935–951, Nov 1997.

[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.

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

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

[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.

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

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

[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.

[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.

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

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

[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.

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

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

[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.

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

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

[25] "H.264/MPEG-4 AVC Reference Software Manual." http://iphome.hhi.de/suehring/tml/JM%20Reference%20Software%20Manual%20%(JVT-X072).pdf/.

[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.

[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.

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

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

[30] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, 1959.