

# AppAdapt: Opportunistic Application Adaptation in Presence of Hardware Variation

Aashish Pant, Puneet Gupta, and Mihaela van der Schaar, *Fellow, IEEE*

**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 30% points at 0% hardware overdesign), a reduction in overdesign (by as much as 8% points at 80% yield) as well as application quality improvements (about 2.0 dB increase in average peak-signal-to-noise ratio at 70% yield). Further, we investigate implications of limited information exchange (i.e., signature quantization) on yield and quality. 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.

**Index Terms**—Application adaptation, hardware-software interface, manufacturing yield, overdesign, process variation.

## I. 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 technology scaling to finer geometry devices, the impact of manufacturing variations is getting worse [2], [3]. For high-performance microprocessors in 180 nm technology, measured variation is found to be as high as 30% in performance and 20 times in chip level leakage within a single wafer [4]. According to the International Technology Roadmap for Semiconductors (ITRS) [5], this trend is expected to get worse (see Fig. 1).

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 design flow in an attempt to maximize the expected manufacturing yield, under the constraint that a certain minimum performance level is satisfied [3]. These methods often result in the creation of designs that are high on resources and designer effort. Other techniques like [6] and [7] rely on post manufacturing tuning of the hardware. For example, threshold voltage of the gates on the critical path can be lowered after manufacturing in order to make the design run

Manuscript received August 12, 2010; revised January 02, 2011; accepted August 12, 2011. Date of publication October 18, 2011; date of current version July 27, 2012. This work was supported in part by the National Science Foundation (NSF) under Grant CCF-1029030 and Grant CNS-0831549.

The authors are with the Department of Electrical Engineering, University of California Los Angeles, Los Angeles, CA 90095 USA (e-mail: apant@ucla.edu; puneet@ee.ucla.edu; mihaela@ee.ucla.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TVLSI.2011.2167360

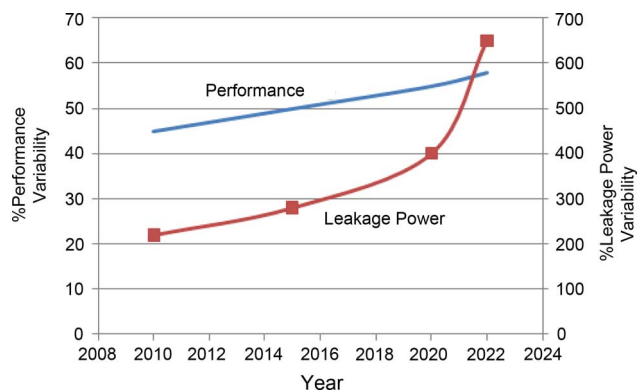


Fig. 1. ITRS projection of variability.

faster (forward body biasing) for slower chips. For extra leaky chips, the threshold voltage can be raised to reduce leakage (reverse body biasing). Often, these techniques require that the designs support requisite tuning knobs, thus making them complex. Moreover, tuning needs to be done on a chip by chip basis and this results in an increased test time. Performance-power optimization techniques like Dynamic Voltage Scaling (DVS) have been used to take process variations into account as in Razor [8].

While process variability is increasing, the basic approach to designing and operating complex systems has remained unchanged. 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 costs. 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. Therefore, there is motivation to think of systems that have a flexible hardware-software interface.

In this paper, we seek to build a flexible hardware-software interface paradigm by proposing the notion of hardware instance guided software adaptation for performance constrained applications. The broad idea is indicated in Fig. 2 where the actual hardware state guides application adaptation on a die specific basis. We show that, by adapting the application to the post manufacturing hardware characteristics (hardware signatures) across different die, it is possible to compensate for application quality losses that might otherwise be significant in

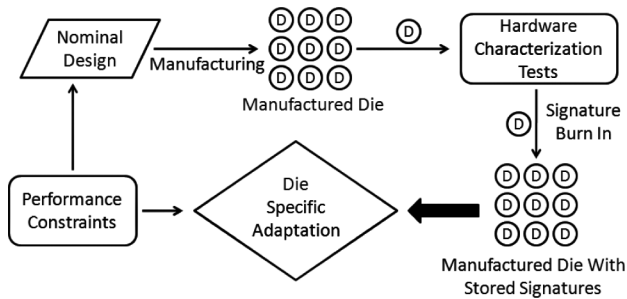


Fig. 2. Proposed application adaptation model.

the presence of process variations. This in turn results in improved manufacturing yield, relaxed requirement for hardware overdesign, and better application quality.

Our work is motivated by the following two observations.

- 1) A plethora of modern applications are reconfigurable and adaptive, e.g., video encoding and decoding, multimedia stream mining, gaming, embedded sensing [9], etc. They 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 overdesign have become expensive [10] and may benefit from being complemented by alternate software-level strategies.

Communication and wireless systems provide an excellent analogy [11]. Communication systems adapt based on the underlying physical communication fabric which is dynamic (for instance [12]–[14]). Therefore, instead of designing protocols with rigid speed and power constraints, an approach that is flexible and allows for tradeoffs is used and it has been proven to be far more effective. In the same way, a system can also adapt to the underlying variation-affected hardware layer.

The idea of modifying the nonhardware layer to suit the underlying hardware (for process variations or otherwise) is not entirely new. In a recent work [15], the authors propose a method to optimize the power management policy of a System-On-Chip (SOC) statistically across all chips taking process variations into account and its effect on leakage power. Further, they suggest approaches to adapt the policy on a chip by chip basis. Software fault tolerance schemes [16] detect hardware faults using methods like Error Correcting Codes (ECCs) and correct them on the fly in the software layer. In a recent work [17], a new low-power motion estimation framework is proposed in which the supply voltage is purposely lowered, occasionally triggering some timing faults which are then corrected using software fault tolerance techniques. To handle supply voltage variations, some authors [18] have proposed the use of a voltage sensor, error recovery hardware, and runtime modification of the compiled software to prevent such voltage variations to get triggered again. Software thermal management techniques [19] perform scheduling in a multitasking scenario to ensure that thermal constraints are met. Error resilience

inherent in applications has also been leveraged to improve defect-limited hardware yield (see [20], [21]).

Most approaches either treat hardware inadequacy or malfunctioning as emergencies by modeling them as transient faults or rely on the inherent error tolerance of specific applications. Moreover, these techniques are employed when the hardware faults happen and some of them require special hardware for correction. For process variations, software adaptation can utilize the application algorithm’s quality or performance tradeoffs to achieve error free operation in the functional sense in presence of permanent manufacturing variations.

Designing a robust and dependable hardware is indispensable in the presence of manufacturing variations. Statistical design and post silicon tuning significantly help to improve overall manufacturing yield. We believe that incorporating die-specific adaptation at the software layer can ease the burden off expensive robust-hardware design methodologies. This is because adaptation is much better informed of application quality tradeoffs at the software layer. In this context, the main contribution of our work is the following.

- To the best of our knowledge, this is the first work to discuss application adaptation based on process-variation affected manufactured hardware.
- Using an H.264 encoder, we show that implementing die-specific software adaptation increases manufacturing yield, improves overall application quality and thereby allows for underdesign of hardware.
- We consider the implications of limited hardware-software information exchange and die test time by presenting methods to compute optimal signature quantization points.

This paper is organized as follows. In Section II, we introduce the concept of hardware signature-based adaptation in the context of applications that are performance constrained. In Section III, we apply this methodology to an H.264 encoder and demonstrate its benefits. In Section IV, we discuss the effects of signature quantization and present an algorithm to compute optimal signature measurement points. We conclude in Section V.

## II. HARDWARE SIGNATURE-BASED ADAPTATION

In this section, we describe the use of hardware signatures for software adaptation in performance constrained applications.

### A. Hardware Guided Adaptation: Formulation for Performance Constrained Applications

Consider a system that comprises of an application running on a generic or dedicated hardware. The application can be tuned to run in different software configurations denoted by set  $\mathcal{S}$ . These configurations are associated with varying performance and quality trends. Note that, if the application is not adaptive,  $\mathcal{S} = \phi$ . Also, in this discussion, we assume the hardware to be static but the idea can easily be extended to reconfigurable hardware.

At this point, it might be worth noting that most applications can be made to support software tuning knobs. Even something as simple as sorting can have a variety of implementations to choose from. As an example, while one implementation requires

a lower runtime, another one might have a less memory footprint, etc. Our focus in this paper is on multimedia applications which inherently provide ample knobs for adaptation.

Adaptation attempts to find the optimal software operating configuration  $c^{\text{opt}} \in \mathcal{S}$ . Note that, the definition of optimality strictly depends on context and will differ from application to application. In our discussion, optimal software operating configuration is one that maximizes output application quality  $\mathbf{Q}$ , while satisfying application execution time constraints,  $ET_{\text{max}}$ . Examples of such systems include but are not limited to audio/video compression applications, gaming, stream mining, graphics processing, etc. The notion of quality and configurations depends on the application. For audio/video compressions applications, quality can be the Peak-Signal-to-Noise Ratio (PSNR) of the encoded bitstream and the configurations can be different modes of operation of some block, say motion estimation.

Note that application execution time strongly depends on the underlying hardware characteristics (maximum operating frequency, memory, etc.). Conventionally, this dependence is assumed to be implicit and worst-case (or expected post manufacturing) hardware characteristics are used to solve for  $c^{\text{opt}}$ . Therefore, regardless of the true post manufacturing hardware characteristics, the same  $c^{\text{opt}}$  is chosen as being optimal for all die. Because of the impact of process variations on hardware performance, post manufacturing hardware characteristics may differ significantly from idealized expectations and also from one die to another (because of die-to-die variations). Hence, the choice of  $c^{\text{opt}}$  may not be truly optimal for all die. We propose the inclusion of these hardware characteristics into the optimization problem as

$$\begin{aligned} c^{\text{opt}}(v) &= \arg \max_{c \in \mathcal{S}} Q(c, v) \\ &\text{under the constraint that} \\ ET(c, v, h) &\leq ET_{\text{max}}. \end{aligned} \quad (1)$$

In this equation,  $\mathbf{v}$  is the input to the application,  $\mathbf{c}$  is an operating configuration,  $\mathbf{Q}$  is the quality, and  $ET$  is the execution time that depends on the configuration  $\mathbf{c}$  and the input  $\mathbf{v}$ .  $\mathbf{h}$  represents the hardware characteristics.

If the underlying hardware consists of more than one functional blocks or more than one independently fabricated components, each block can be affected by process variations in different ways (because of within-die process variations). The hardware characteristic  $\mathbf{h}$  should therefore include the state of every functional block. Consequently, an application can knowledgeably adapt and redistribute the effort of computation among the hardware functional blocks to achieve the same desired performance given the manufactured hardware. A die that does not currently satisfy the performance constraint can be made usable by adapting the operating configuration to give the same performance at a small tolerable loss in output quality using (1).

### B. Hardware Signatures: Representing True Hardware Characteristics

Equation (1) assumes that the application is aware of the exact hardware characteristics on a die specific basis. We call these die specific hardware characteristics made known to the soft-

ware application as *hardware signatures*. Hardware signatures are potentially different for different die and different functional blocks within the same die.

Choice of signature content depends on the particular system objectives. For systems that pose strict constraints on timing (real-time applications), signature could comprise of the maximum operating frequency of individual functional blocks of hardware. System memory along with speed of the CPU-memory interface can be an important metric to include if memory intensive and computation intensive techniques are choices for application configuration. Indeed, exploiting space-time tradeoff has been a major focus of research in algorithms. By knowing the exact frequency and memory characteristics of the hardware at hand, these algorithms can make decisions optimal for that particular hardware. For systems where low-power operation is a concern, the exact value of leakage power and maximum switching current are valid signatures contents. Knowing the exact values of leakage and switching power can aid power management policies like Dynamic Voltage and Frequency Scaling (DVFS) to make optimal die specific power-performance tradeoffs. High leakage variability [4] indicates tremendous potential for power savings through adaptation.

Hardware signatures can be measured once post-fabrication and written into a nonvolatile software readable<sup>1</sup> memory element on-chip or on-package. Signature characterization can be done in software as well with some hardware support (e.g., mechanisms to detect errors and control frequency). This is likely more expensive though with a benefit of runtime characterization. Well-known parametric tests such as FMAX (performance) and IDDQ (leakage power) can yield such signature values. Signatures can also be measured at regular intervals during system operation to account for ambient voltage/temperature fluctuations and wearout mechanisms such as Time Dependent Dielectric Breakdown (TDDB) and Negative Bias Temperature Instability (NBTI). At-speed logic and memory built-in self test (BIST) techniques can be employed for faster and any time computation of such signatures. Approximations using on-chip monitors (e.g., ring oscillators or monitors such as [23]) 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.

### C. Q-C Plot and Modeling Hardware Signatures

The behavior of a performance constrained application can be represented by a Quality-Complexity (Q-C) plot [24]–[26] (see Fig. 3). Every valid operating configuration (for a fixed input, process and environmental condition)  $c$  is represented by a point  $(x_c, y_c)$  on the Q-C plot, where the  $Y$ -coordinate ( $y_c$ ) represents Quality ( $Quality(c)$ ) and  $X$ -coordinate ( $x_c$ ) represents execution time of the application ( $ET(c)$ ) in that configuration. Operating configurations with larger execution times (under constant input, process and environmental condition assumption) are usually associated with higher quality as the application gets more time to process its input and therefore, can do a better job.

For illustrative purposes, we do not show the dependence on input  $\mathbf{v}$ . Various recent works deal with the problem of cap-

<sup>1</sup>Most modern chips already contain several such EEPROM or NVRAM components for storing hardware IDs, time, etc. (e.g., see [22])

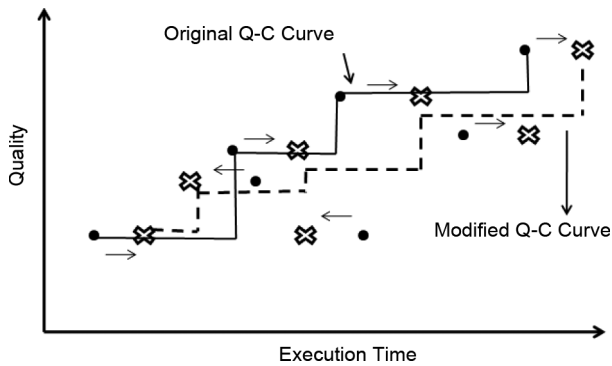


Fig. 3. Q-C plot changes with hardware.

turing input dependence. Classification and machine learning [27], [28] is a recent example in which a set of training data is first computed by executing the application on various kinds of inputs and operating environments. Subsequently, relevant and easy to compute features are extracted from this training data. At runtime, input features are matched to the features computed offline. Various other ad-hoc solutions [29], [30] have been proposed in the same area. This is a well researched topic and is not the focus of this work. We urge the interested reader to refer to [27] and [28] for details. We also assume constant environmental conditions for our analysis.

The behavior of the system as formulated in (1) can be translated to the problem of finding the configuration with maximum quality that lies to the left of the vertical line  $x = ET_{\max}$ , where  $ET_{\max}$  is the application tolerated execution time constraint for the system. Therefore, over the range of such execution time constraints, the optimal operating points (the points of quality upperbound) form an envelope or a Q-C curve. Note that these operating configurations are discrete and not continuous. Therefore, a particular operating configuration will be optimal for a range of execution time constraints. A Q-C curve will therefore typically look like a staircase function.

The Q-C plot implicitly depends on hardware state. By making this dependence explicit as formulated in (1), every die will have its own Q-C plot. Specifically, an application configuration will have different execution time ( $ET(c)$ ) for different die depending on the process variation scenario, i.e., the configuration undergoes a horizontal shift in position on the Q-C plot. Therefore, the envelope or the operational Q-C curve also changes. The magnitude of the configuration point shift on the Q-C plot 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. Fig. 3 demonstrates this Q-C curve change.

Hardware signatures make the application aware of such die specific Q-C plot perturbations. By knowing the exact die specific Q-C curve, the application is better equipped to make optimal  $c^{opt}$  selections. This results in improved manufacturing yield as systems may now successfully operate in die specific optimal configurations instead of being simply discarded for not satisfying the specified performance constraints or minimum quality levels. This also translates to a smaller performance

guardband requirement to achieve the same manufacturing yield.

Note that the presence of a quality-performance tradeoff is essential for the above methodology to work. There is a large class of modern day applications that fall under this category. For example, video encoding, multimedia stream mining, gaming, and embedded sensing [9] are examples of such applications. The class of RMS applications proposed in [31] are all conducive to this kind of tradeoff. As shown in the next section, incorporating power into the optimization framework opens up yet another broad class of applications that can benefit from this strategy. Evaluating all such applications is out of scope of this paper. We concentrate on H.264 encoding for our analysis.

### III. PROOF OF CONCEPT: H.264 ENCODING

In this section, we apply the proposed adaptation to an H.264 encoding scheme [32], [33]. We assume that motion estimation (M.E), DCT transform (T.X), and entropy coding (E.C) modules are the three independent functional blocks of the encoder. Quality  $Q$  is given by the PSNR of the encoded video. The encoder is required to maximize the output PSNR subject to bitrate ( $R_{\max}$ ) and frame processing delay ( $ET_{\max}$ ) constraints. Please refer to Table I for details. The optimization in (1) can be rewritten as

$$c^{opt}(v) = \arg \max_{c \in S} \text{PSNR}(c, v)$$

under the constraint that

$$R(c, v) \leq R_{\max}$$

$$ET(c, v, h) \leq ET_{\max}. \quad (2)$$

In (2),  $ET(c, v, h)$  is the sum of the execution times of the three functional blocks and  $S$  is the set of all available encoder configurations. Table II shows the various representative H.264 encoder tuning knobs used in our experiments. Please refer to [34] for a detail description of these tuning parameters. Operating configurations are obtained by permuting the values of the knobs. These configurations range from algorithmically simple to more complex ones with varying levels of performance and quality tradeoffs. Fig. 4 shows the Q-C plot for the encoder using the above configuration set.<sup>2</sup> This plot is constructed from the data obtained from profiling the encoder running on a representative video sequence. In our experiments, we use the mobile video sequence because of its strong texture and complex motion content. Note that, from our discussion on input classification, in practical systems, every input type will have its own associated Q-C curve and some online learning technique may be employed to map to the correct input type at runtime. Hardware signatures are taken to be the independent frequency deviations of the three functional blocks (refer to Table I).

We also consider and demonstrate improvements for over-designed hardware in our experiments, where the percentage of overdesign is varied from  $-20\%$  to  $+20\%$ . Overdesign

<sup>2</sup>In this context, it should be noted that a PSNR difference of 0.5 to 1 dB is significant and is perceivable to the human eye.

<sup>3</sup>In this context, a negative value of overdesign simply means an under-designed hardware.

TABLE I  
EXPERIMENT SPECIFICATIONS

Number of Frames	192
$ET_{max}$	0.03 seconds
Bitrate	800 kbps
Frames per second	33
Frequency Variation	I.I.D Gaussian Distributed Mean=0, $3\sigma=30\%$ of Nominal Frequency

TABLE II  
ENCODER CONFIGURATIONS USED IN EXPERIMENTS

1	Enable/Disable sub-pixel motion estimation
2	Enable/Disable bi-prediction sub-pel motion estimation
3	FFT transform window size: 8x8, 4x4 or combination of both
4	Run length encoding: CABAC [35] or CAVLC
5	Enable/Disable 8x8, 8x16, 16x8, 8x4, 4x8 motion estimation search window
6	Enable/Disable 8x8, 8x16, 16x8, 8x4, 4x8 bi-prediction motion estimation search window

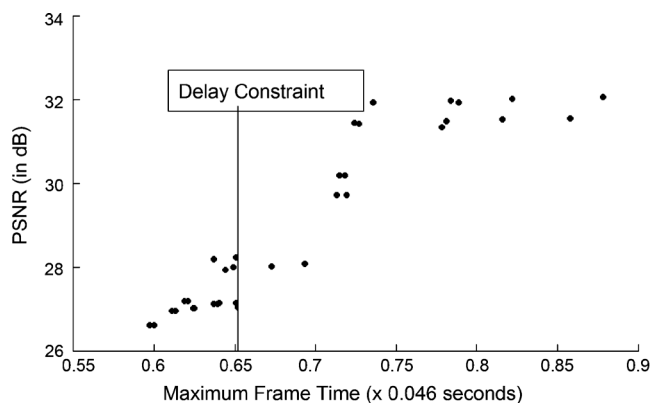


Fig. 4. Operating configurations for the H.264 encoder.

provides for a guardband/margin in hardware performance. In other words, the hardware is intentionally designed to achieve a higher performance than is required. This is done to overcome potential degradation in performance due to process variations to regulate manufacturing yield.<sup>3</sup> However, this overdesign has significant penalties in terms of area, power, cost and turnaround time [10]. In our experiments, overdesign (i.e., faster hardware) is handled by relaxing the input processing time constraint  $ET_{max}$ .

### A. Results and Discussion

In Fig. 5, we show the change in encoder PSNR as the operating frequency varies.<sup>4</sup> When encoding is done at nominal frequency (0% frequency variation), both the nonadaptive (red dashed line) and adaptive (blue solid line) cases have the same PSNR. This is because, they are operating in the same base configuration with no frame loss.

As frequency reduces, the PSNR of the nonadaptive encoder differs from the adaptive one. This is because, the non adaptive encoder operates in the same base configuration and starts

<sup>4</sup>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.

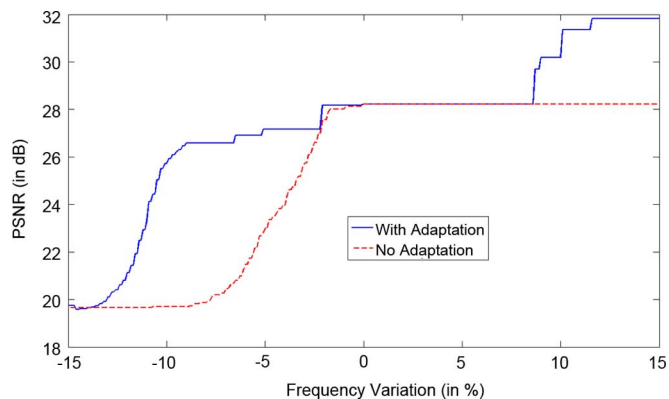


Fig. 5. Hardware guided adaptation improves PSNR (for samples of video sequences encoded using adaptive and nonadaptive methods, please see <http://nanocad.ee.ucla.edu/Main/Codesign>).

dropping frames rapidly. Consequently, its PSNR falls sharply.<sup>5</sup> On the other hand, the adaptive encoder tries to adapt to a configuration that ensures maximum quality with no frame loss. Consequently, it is able to achieve a higher PSNR than the nonadaptive case. For example, when frequency decreases from nominal, the adaptive encoder adapts from a configuration with PSNR of 28.28 dB to a less complex configuration with PSNR of 28.18 dB, thus avoiding frame loss and achieving a better overall quality.

When frequency increases, the adaptive encoder shifts to a more complex configuration (still with zero frame loss) and achieves a PSNR higher than nominal, while the nonadaptive encoder is not able to utilize the faster hardware. Consequently, its PSNR stays flat over the higher frequency range, i.e., hardware-aware adaptation achieves the same desired PSNR with a lower frequency of operation, in turn implying that such a system can tolerate process variations to a greater extent.

We perform Monte Carlo simulation on 1000 die samples (Table I). The Q-C curve perturbation for every die sample is estimated and optimal operating configuration  $c^{opt}$  is found using (2). We plot the results by varying hardware overdesign. Overdesign provides a guardband in performance to counter the effect of process variations after manufacturing. We define manufacturing yield as the percentage of die that undergo no frame loss (i.e., a jitter constraint).

Fig. 6 demonstrates significant yield improvements with adaptation. At 0% overdesign, yield of the nonadaptive encoder is 50% (intuitively, half of the manufactured die lie on either side of the nominal hardware under normal frequency distribution). When the encoder adapts according to the manufactured hardware, it operates in a configuration with minimal frame loss and yield increases to 80%. This trend is seen over the entire span of positive or negative overdesign. An important point to observe is that, given enough available configurations (scalable encoding), application adaptation can ensure almost constant quality by trading off work needed for different components. From Fig. 6, we can also conclude that hardware-aware adaptation relaxes the requirement of overdesign to achieve the

<sup>5</sup>We handle lost frames by replacing them with the previously known good frame and computing the output PSNR as is usually done in real-time multimedia decoders.

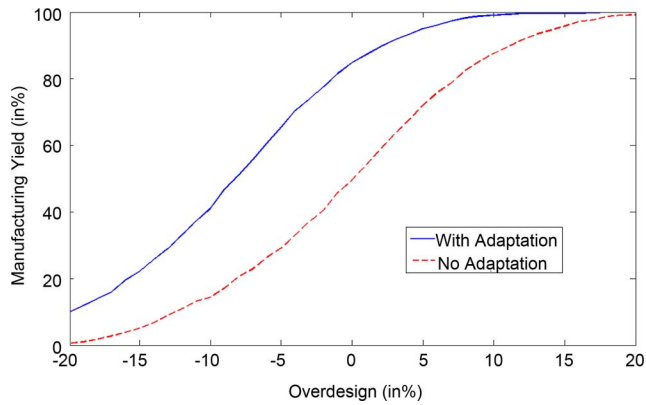


Fig. 6. Hardware guided adaptation improves manufacturing yield.

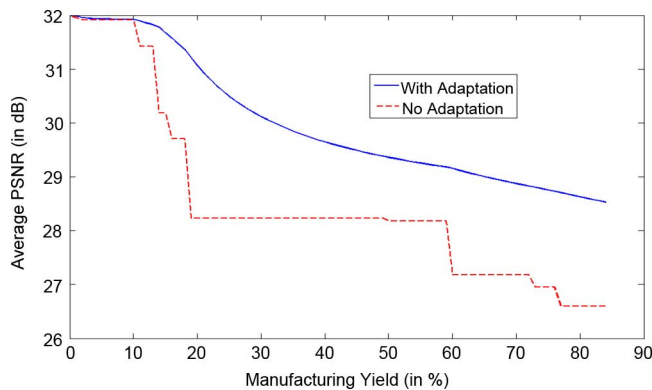


Fig. 7. Hardware guided adaptation improves overall application quality.

same manufacturing yield. For example, to ensure 80% yield, adaptation relaxes the overdesign requirement by 8%.

Fig. 7 shows the variation of average PSNR across all passing die with manufacturing yield for both hardware adaptive and nonadaptive cases. We only show the plot for 0% overdesign, i.e., nominal design as the data for other overdesign 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.<sup>6</sup> At 70% yield, average PSNR for hardware adaptive case is higher by 2.0 dB. For the nonadaptive 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 graceful degradation in PSNR when improving yield, as operating configurations can change on a die-by-die basis.

### B. 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 allotted 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 Fig. 8 which shows the dependence of frequency and

<sup>6</sup>For the adaptive case, the highest quality die are used to match the nonadaptive case for the same yield.

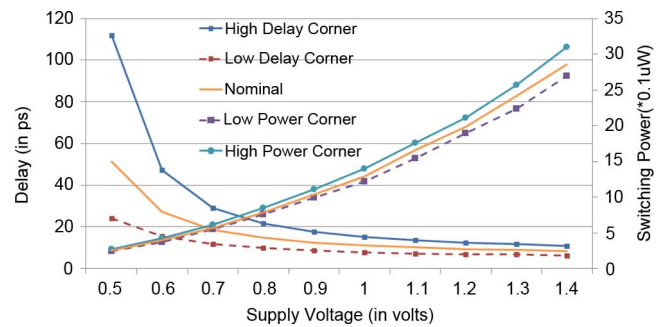


Fig. 8. Variation of frequency and power with supply voltage under process variations.

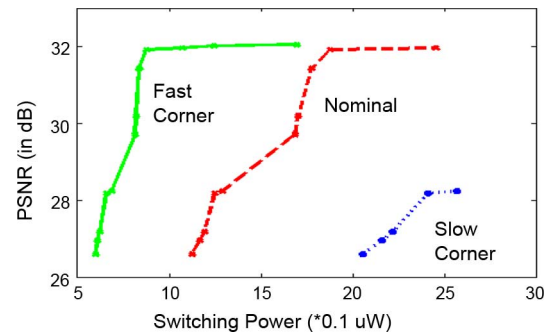


Fig. 9. Variation space of the PSNR versus power curves for nominal/slow/fast corners under process variations for H.264 encoder.

switching power<sup>7</sup> on supply voltage for a simple 4 stage FO-4 inverter chain<sup>8</sup> under process variations (varying transistor length and threshold voltage by  $\pm 10\%$ ) 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 die is significantly different and so is power dissipation, resulting in a wide power-performance band. For example, at supply voltage of 1 V, there is a variation of 64% in delay and 16% in switching power across the nominal. By knowing the exact power-performance trade-off specific to a die, adaptation algorithms like DVS 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 estimate the returns that one can expect, we scale supply voltage to achieve the same performance for various sample die affected by process variations. As a result, power consumption of these sample die changes according to Fig. 8. Using the Q-C curve of Fig. 4, we construct the PSNR versus power curves of the H.264 encoder for these sample die in Fig. 9. Specifically, we show results for the fast corner, slow corner, and the nominal. Intuitively, the power consumption of the faster corner is lower because it can operate at a lower supply voltage to achieve same performance. These curves show that different die have different power requirement levels to achieve the same performance (and quality) and this gives us a potential scope of improvement using signature-based adaptation.

<sup>7</sup>In this analysis, switching power is estimated at constant frequency of operation i.e., the variation in switching power is essentially the same as that of switching energy. This variation is mainly due to change in gate capacitance.

<sup>8</sup>45 nm PTM models have been used for these simulations.

Hardware signature for such a system will consist of a lookup table that specifies the operational voltage (e.g., a lookup table-based method is proposed in [36] and [37] 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 algorithms like *DVS* know of the exact operational PSNR-power curve specific to that die.

#### IV. HARDWARE SIGNATURE MEASUREMENT TRADEOFFS

Size (i.e., how many functional blocks and how many parameters per block) and quantization (e.g., discretization of performance into frequency bins) of the signature affects the potential benefit that can be derived from signature-based adaptation. Signature quantization influences storage and more importantly, post manufacturing test complexity. In this section, we focus on determining optimal signature quantization scheme. The problem is very similar to the concept of data compression using quantization in signal theory. Quantization results in an associated distortion during signal reconstruction. The choice of signal quantization levels is therefore very important to minimize distortion. For this analysis, we assume operating frequency as the hardware signature. Therefore, we focus on the problem of determining what frequencies to test (and store as signatures) to ensure minimum quality loss, given the maximum permitted number of such frequency tests.

Consider a hardware system with  $N$  independent components. When the system operates in software configuration  $c$ , a certain number of average execution cycles are spent in each component. Thus, every configuration  $c$  can be represented by the load distribution row vector  $LD^c$

$$LD^c = [m_1^c \quad m_2^c \quad \dots \quad m_N^c]$$

where  $m_k^c$  are the number of execution cycles spent in component  $k$  when executing in configuration  $c$ . Let  $t_k^c$  be the time spent in component  $k$  and  $t^c$  be the total input processing time when the system is operating in configuration  $c$ . We have

$$\sum_{k=1}^N t_k^c = t^c.$$

Because of frequency quantization, let  $X_k^i$  be the  $i^{th}$  frequency quantization point ( $i \in 1 \dots s_k$ ) for component  $k$  ( $k \in 1 \dots N$ ) which need to be determined. Further, assume that

$$X_k^i < X_k^j \text{ for } i < j \text{ and } k \in 1 \text{ to } N. \quad (3)$$

For some die, let the maximum frequency of component  $k$  for  $k \in 1 \dots N$  be quantized to  $X_k^{i_k}$ . When the component operates at this frequency, then  $t_k^c = m_k^c / X_k^{i_k}$ , and therefore

$$t^c = \sum_{k=1}^N \frac{m_k^c}{X_k^{i_k}} = LD^c \times p$$

where  $p$  is the quantized hardware signature of the die and is given by

$$p^T = \left[ \frac{1}{X_1^{i_1}} \quad \frac{1}{X_2^{i_2}} \quad \dots \quad \frac{1}{X_N^{i_N}} \right].$$

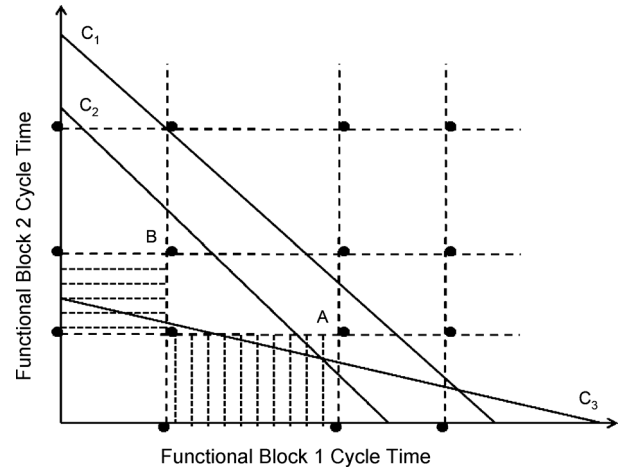


Fig. 10. Two component signature quantization.

The optimal configuration is the one which meets the input processing time constraint and has the maximum output quality (refer (1)). The input time constraint is met when

$$t^c = LD^c \times p \leq ET_{\max}.$$

The output quality of the application executing on the die is therefore

$$Q_{die} = \max_{c \in S} (Q^c \times u(ET_{\max} - (LD^c \times p)))$$

where  $Q^c$  is the output quality when the application operates in configuration  $c$  and  $u(\cdot)$  is the standard unit step function. Note that there are a total of  $\prod_{k=1}^N s_k$  possible quantized signatures. Let these quantized signatures be denoted by set  $\mathbf{P}$ . At this point, it is helpful to visualize the sampling process as an  $N$ -dimensional space where an axis corresponds to the cycle time of a unique component. The signature set  $\mathbf{P}$  is therefore represented by a set of points which divide the  $N$ -dimensional space into  $N$ -dimensional hyper-rectangles. The hyper-volume of such an  $N$ -dimensional hyper-rectangle encompasses all those die that will be quantized to one signature. This is shown in Fig. 10 for two components. Note that the frequencies of components 1 and 2 are quantized at three points giving a total of nine signature quantization points. Every configuration ( $c_1$ ,  $c_2$  and  $c_3$  in the figure) is represented as a slanted line  $x^c + y^c = ET_{\max}$ . Therefore, all die that lie to the bottom left of a configuration line can operate at that configuration to meet performance constraints. The vertically (horizontally) shaded hardware space in the figure will be quantized to the signature on the upper right corner, i.e., point A (B). Hence, while the hardware enclosed in the vertically shaded region can only operate at  $c_1$ , those that belong to the horizontally enclosed region will operate at  $\max(Q(c_1), Q(c_2))$ .

Let  $f(x_1, x_2, \dots, x_N)$  be the joint probability distribution function of the frequency variations of the hardware components. For signature  $p \in \mathbf{P}$ , let  $V_p$  denote the

probability weighted hyper-volume of the  $N$ -dimensional hyper-rectangle that is quantized to  $\mathbf{p}$

$$V_p = \int_{X_1^{i_1-1}}^{X_1^{i_1}} \cdots \int_{X_N^{i_N-1}}^{X_N^{i_N}} f(x_1, \dots, x_N) dx_1 \dots dx_N.$$

If the frequency variations of the hardware components are independent, then

$$V_p = \prod_{k=1}^N \int_{X_k^{i_k-1}}^{X_k^{i_k}} f(x_k) dx_k.$$

For a given signature quantization scheme, the total quality benefit of having a signature at point  $\mathbf{p}$  is given by  $V_p \times Q_p$ . For the most optimal signature quantization, this quality needs to be maximized. Therefore, the signature quantization problem can be formulated as an optimization problem

$$\text{Maximize QuantGain} = \sum_{p \in P} V_p \times Q_p$$

where

$$V_p = \int_{X_1^{i_1-1}}^{X_1^{i_1}} \cdots \int_{X_N^{i_N-1}}^{X_N^{i_N}} f(x_1, \dots, x_N) dx_1 \dots dx_N$$

$$Q_p = \max_{c \in C} (Q^c \times u(ET_{\max} - (LD^c \times p)))$$

$$X_k^i < X_k^j \text{ for } i < j \text{ and } k \in 1 \text{ to } N. \quad (4)$$

The above formulation is very similar to that of scalar quantization in multiple dimensions commonly encountered in signal compression theory. The problem is to determine the signature quantization values, given the maximum number of such values so that expected quality is maximized. However, the expression for quantization error in (4) is significantly different from the standard minimum mean square (MMSE) quantization error formulation in signal compression theory.

Equation (4) is a generic optimization problem and therefore, various ad-hoc techniques can be employed. For our experiments, we use the cyclic coordinate descent approach to solve the optimization problem in (4). Specifically, we employ an iterative strategy, where at each iteration step, we determine the best location of one signature quantization point, given the location of all other signature quantization points. We perform this analysis for all quantization points. We iterate until the quality benefit of performing another iteration is less than a certain threshold. The global range of quantization point variation is  $-3\sigma$  to  $3\sigma$  of frequency variation. Note that, in this process, we arrive at a locally optimal solution that depends on the initial starting point. By repeating the process for different starting points, a sufficient amount of solution space can be covered.<sup>9</sup>

<sup>9</sup>We believe that for practical problems, this strategy is manageable. Moreover, specific ad-hoc methods can always be employed to perform this iteration efficiently. For example, in our H.264 encoder, we had 3 components and 34 configurations. We carried out the iteration procedure by initially starting with big iterator steps and then reducing the step size gradually. This helps in two ways. It ensures quick convergence near the optimal through big movements in signature quantization parameters when we are searching far away from the optimal solution in the solution space. On getting to the near optimal space, we reduce the iterator step size to fine tune the location of the signature quantization parameters. On a 2.5 Ghz Xeon processor, this analysis took 18 s for five quantization points per component in MATLAB.

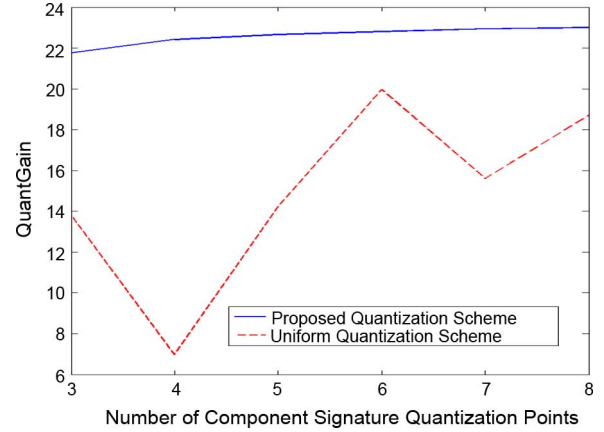


Fig. 11. QuantGain versus total number of signature quantization points per component for 0% overdign.

In the next section, we will show how the optimal solution can be obtained for the special case of one-component hardware ( $N = 1$ ). At this point, it is worth noting that computational complexity of solving (4) is not critical as deciding on a signature quantization scheme needs to be done just once for a product.

For the Q-C plot of the H.264 encoder (see Fig. 4), we compare our proposed signature quantization scheme with a uniform quantization scheme.<sup>10</sup> Figs. 11 and 12 shows QuantGain and yield loss respectively as the number of signature quantization points per component is varied. Note that our proposed signature quantization scheme results in a higher QuantGain (and therefore a higher PSNR) as well as improved yield over the uniform quantization scheme. Uniform quantization scheme is not able to capture the sensitivity of expected quality (and yield) to the location of the quantization points and hence a rippling behavior is observed as the number of signature quantization points increases. For the proposed scheme, expected quality monotonically increases with the number of signature quantization points.

Note that, if the probability distribution  $f(x_1, x_2, \dots, x_N)$  is not known in closed form, Lloyd's algorithm [38] for vector quantization can be employed to solve the problem using representative hardware samples.

#### A. Special Case: One Component Hardware

It is interesting to think of the signature quantization problem of the previous section in the special case of one-component hardware. The objective is to find optimum signature quantization points  $X^j$  ( $j \in 1 \dots s$ ) for maximum quality, where  $s$  is the maximum number of available quantization points and  $|C|$  is the number of configurations. We have dropped the subscript  $k$  from the notation because there exists just one component. Note that,  $N = 1$  can be plugged into (4) and similar techniques as in the previous section can always be employed. Here, we will analytically solve the signature quantization problem for one-component hardware using Q-C curves. This is important as there exists a definite solution to the optimization problem in the one-component case.

<sup>10</sup>In the uniform quantization scheme, component signature quantization is done at equal frequency intervals lying between  $-3\sigma$  to  $3\sigma$  of frequency variation.



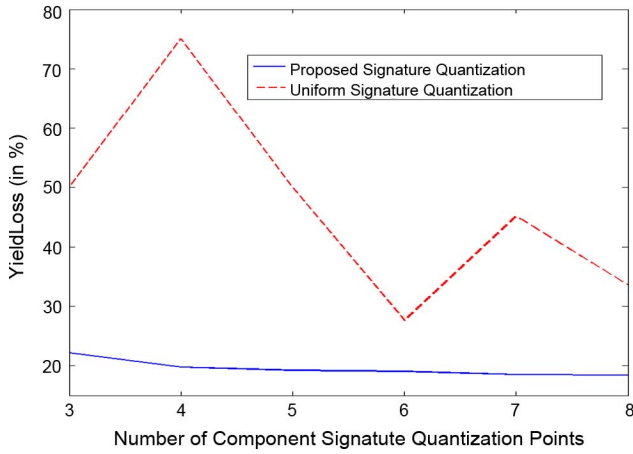


Fig. 12. Yield loss versus total number of signature quantization points per component for 0% overdesign.

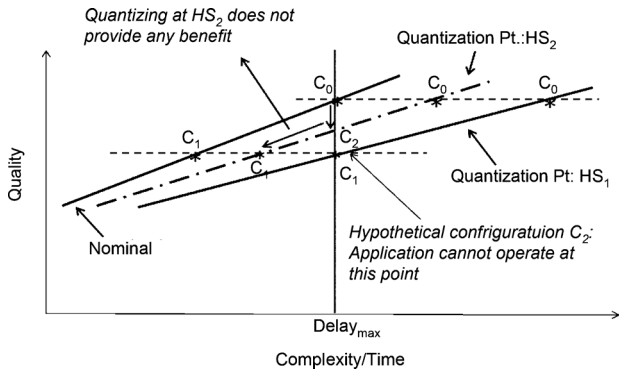


Fig. 13. Signature measurement for one-component hardware.

We will start by developing an intuition into the solution. Consider Fig. 13.  $c_0$  and  $c_1$  are two operating configurations. The Q-C curve for 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  $HS_2$ ,  $c_2$  (that lies on the  $ET_{max}$  line) is not a valid physically existing operating configuration. So, the application has to operate at  $c_1$  for  $HS_2$ . For  $HS_1$ ,  $c_1$  lies on the  $ET_{max}$  line and the application operates at  $c_1$ . Therefore,  $HS_2$  and  $HS_1$  are equivalent from this perspective. Every die slower than the nominal but faster than  $HS_1$  will operate on  $c_1$ . From the above, it makes sense to quantize signature at  $HS_1$ , but no additional benefit is obtained by having a quantization point between  $HS_1$  and nominal. This result is important as it limits the potential solution space of the problem.

Therefore, when  $s \geq |C|$ , the optimum location of signature quantization points correspond to those hardware which have their Q-C curves intersecting the  $ET_{max}$  line at valid operating configuration points on the Q-C plot. Any additional number of signature quantization points over the number of configurations are redundant and will not improve quality.

When  $s < |C|$ , a brute-force search technique would require  $\binom{|C|}{s}$  operations to get to the optimal quantization set. As previously mentioned, computational complexity is not an issue. However, a clever technique that uses graph shortest path algorithm [39] can be used to solve this without brute-force.

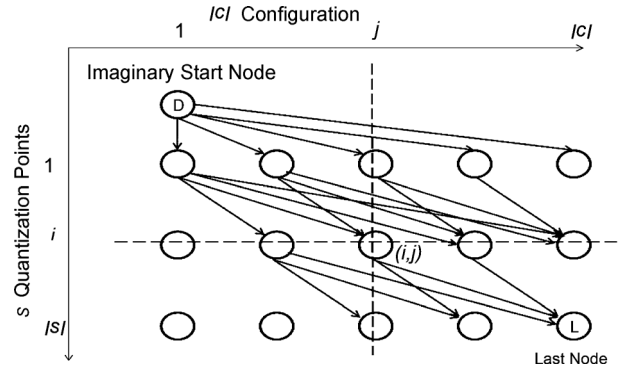


Fig. 14. Shortest path approach to find optimal signature quantization.

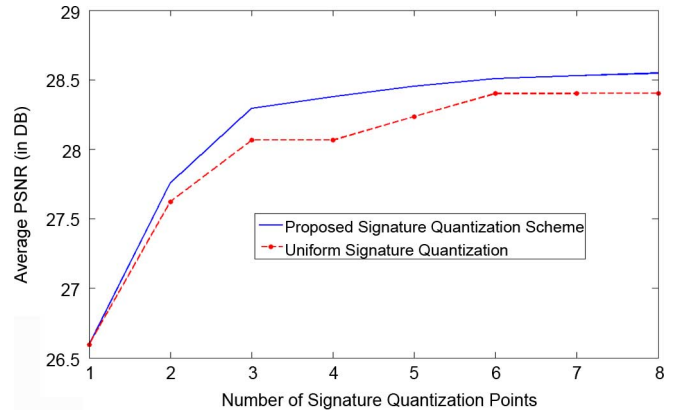


Fig. 15. Improvement in PSNR with finer signature granularity.

Consider Fig. 14. Let  $Q^{c_j}$  denote the quality corresponding to configuration  $c_j$  and let  $X^j$  be the corresponding signature quantization location. The number of nodes in the graph is  $|C| \times s$  (arranged as a matrix) and the cost of an edge from node  $(i1, j1)$  to  $(i2, j2)$  ( $cost_{(i1,j1)}^{(i2,j2)}$ ) is the quality loss incurred by having signature quantization point at configurations  $j1$  and  $j2$  and no quantization point between them (note that all nodes in column  $j$  have same quality  $Q^{c_j}$  and  $Q^{c_{j1}} > Q^{c_{j2}}$  for  $j1 < j2$ ). If  $f(x)$  is the probability distribution of the frequency variations of the hardware, then

$$cost_{(i1,j1)}^{(i2,j2)} = \begin{cases} \infty, & \text{if } j2 \leq j1 \\ \infty, & \text{if } i2 \neq i1 + 1 \\ \sum_{l=j1+1}^{j2} (Q^{c_l} - Q^{c_{j2}}) \int_{X^{l-1}}^{X^l} f(x) dx, & \text{otherwise.} \end{cases}$$

Every path from node  $D$  (imaginary node corresponding to having a quantization point at  $\infty$ ) to node  $L$  (last signature quantization location corresponding to the maximum tolerable variation) will consist of  $s$  nodes. The quality loss minimization problem maps to finding the shortest path from  $D$  to  $L$ . Nodes in the path correspond to the quantization points.

We perform this analysis for the Q-C curve of the H.264 encoder shown in Fig. 4 for different values of  $s$  and the results are compared to a naive signature quantization approach, where quantization is done at uniform intervals. From Fig. 15, it can be observed that the proposed signature quantization results in higher PSNR than the uniform quantization approach.

## V. CONCLUSION

In this work, we have built on the notion of a flexible hardware-software interface by proposing the use of hardware instance guided software adaptation for performance constrained applications. With increasing variability, there is a need to shift from the basic approach of designing and operating complex system with a rigid hardware-software interface. With more and more applications being adaptive by nature, we show that variation-aware software adaptation can ease the burden of strict power-performance constraints in design. Hardware signatures (once measured and stored) can be used to guide software adaptation to handle variability on a die by die basis. For an H.264 encoder, we illustrate that this approach can lead to an improvement in manufacturing yield, relaxed requirement for overdesign and an overall better application quality. Specifically, for the H.264 encoder:

- Manufacturing yield improves by 30% points at 0% overdesign.
- For an objective yield of 80%, adaptation relaxes the need for overdesign by 8%.
- Encoding quality is better by 2.0 dB over the non adaptive case with an objective yield of 70%.

We discuss the implications and cost of signature test and present methods to quantize signatures in an optimized way. We do this analysis for a generic multicomponent hardware and then discuss the special case of one-component hardware. Overall, we believe that adaptation is better informed of application quality tradeoffs at the application layer rather than the hardware layer. Therefore, it is easier and cheaper to implement adaptation at the software layer as compared to designing a robust and dependable hardware.

We plan to extend, implement and show the improvements of this methodology for various other application scenarios. Dynamic voltage scaling is a potential application as was briefly hinted in this paper. Further, we will investigate signature-based adaptation policy perturbations in already adaptive applications. In the future, it would also be interesting to compare hardware-level variation mitigation approaches with proposed opportunistic software approaches.

## REFERENCES

- [1] A. Pant, P. Gupta, and M. van der Schaar, "Software adaptation in quality sensitive applications to deal with hardware variability," in *Proc. 20th Great Lakes Symp. VLSI, GLSVLSI'10*, New York, pp. 85–90.
- [2] Y. Cao, P. Gupta, A. Kahng, D. Sylvester, and J. Yang, "Design sensitivities to variability: Extrapolations and assessments in nanometer VLSI," in *Proc. 15th Annu. IEEE Int. Conf., ASIC/SOC'02*, 2002, pp. 411–415.
- [3] S. Nassif, "Modeling and forecasting of manufacturing variations," in *Proc. Asia and South Pacific Design Autom. Conf., ASP-DAC'01*, 2001, pp. 2–10.
- [4] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proc. 40th Annu. Design Autom. Conf., DAC'03*, New York, 2003, pp. 338–342.
- [5] International Technology Roadmap for Semiconductors, "Process integration, devices and structures," 2007. [Online]. Available: <http://public.itrs.net>
- [6] J. W. Tschanz, J. T. Kao, S. G. Narendra, R. Nair, D. A. Antoniadis, A. P. Ch, S. Member, and V. De, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1396–1402, Nov. 2002.
- [7] S. Sen, V. Natarajan, R. Senguttuvan, and A. Chatterjee, "Pro-visor: Process tunable virtually zero margin low power adaptive RF for wireless systems," in *Proc. 45th Annu. Design Autom. Conf., DAC'08*, New York, 2008, pp. 492–497.
- [8] D. Ernst, N. S. Kim, S. Das, S. Pant, T. Pham, R. Rao, C. Ziesler, D. Blaauw, T. Austin, T. Mudge, and K. Flautner, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th Int. Symp. Microarchitecture, IEEE CS Press*, 2003, pp. 7–18.
- [9] L. Wanner, C. Apte, R. Balani, P. Gupta, and M. B. Srivastava, "A case for opportunistic embedded sensing in presence of hardware power variability," in *Proc. Workshop on Power Aware Comput. Syst., HotPower'10*, 2010, pp. 1–8.
- [10] K. Jeong, A. B. Kahng, and K. Samadi, "Quantified impacts of guard-band reduction on design process outcomes," in *Proc. 9th Int. Symp. Quality Electron. Design*, Washington, DC, 2008, pp. 790–797.
- [11] N. Shanbhag, "A mathematical basis for power-reduction in digital VLSI systems," *IEEE Trans. Circuits Syst. II: Analog Digit. Signal Process.*, vol. 44, pp. 935–951, Nov. 1997.
- [12] P. Bhat, V. Prasanna, and C. Raghavendra, "Adaptive communication algorithms for distributed heterogeneous systems," in *Proc. 7th Int. Symp. High Performance Distrib. Comput.*, Jul. 1998, pp. 310–321.
- [13] S. Sampei, S. Komaki, and N. Morinaga, "Adaptive modulation/TDMA scheme for personal multimedia communication systems," in *Proc. Global Telecommun. Conf., Communications: The Global Bridge, GLOBECOM'94*, 1994, pp. 989–993.
- [14] X. Qiu and K. Chawla, "On the performance of adaptive modulation in cellular systems," *IEEE Trans. Commun.*, vol. 47, no. 6, pp. 884–895, Jun. 1999.
- [15] S. Chandra, K. Lahiri, A. Raghunathan, and S. Dey, "System-on-chip power management considering leakage power variations," in *Proc. 44th Annu. Design Autom. Conf., DAC'07*, New York, 2007, pp. 877–882.
- [16] G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. I. August, "Swift: Software implemented fault tolerance," in *Proc. 3rd Int. Symp. Code Generation and Opt.*, 2005, pp. 243–254.
- [17] G. V. Varatkar and N. R. Shanbhag, "Energy-efficient motion estimation using error-tolerance," in *Proc. Int. Symp. Low Power Electron. Design, ISLPED'06*, New York, 2006, pp. 113–118.
- [18] V. Reddi, M. Gupta, M. Smith, G. Y. Wei, D. Brooks, and S. Campanoni, "Software-assisted hardware reliability: Abstracting circuit-level challenges to the software stack," in *Proc. 46th ACM/IEEE Design Autom. Conf., DAC'09*, 2009, pp. 788–793.
- [19] J. Choi, C.-Y. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal-aware task scheduling at the system software level," in *Proc. Int. Symp. Low Power Electron. Design, ISLPED'07*, New York, 2007, pp. 213–218.
- [20] M. Breuer and H. Zhu, "An illustrated methodology for analysis of error tolerance," *Design Test of Computers*, vol. 25, pp. 168–177, Mar.–Apr. 2008.
- [21] I. S. Chong and A. Ortega, "Hardware testing for error tolerant multimedia compression based on linear transforms," in *Proc. 20th IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. DFT'05*, Oct. 2005, pp. 523–531.
- [22] [Online]. Available: <http://docs.sun.com/source/816-5772-11/funct.html>
- [23] S. Mukhopadhyay, K. Kang, H. Mahmoodi, and K. Roy, "Reliable and self-repairing SRAM in nano-scale technologies using leakage and delay monitoring," in *Proc. IEEE Int. Test Conf., ITC'05*, 2005, pp. 10–1135.
- [24] B. Foo, Y. Andreopoulos, and M. van der Schaar, "Analytical rate-distortion-complexity modeling of wavelet-based video coders," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 797–815, Feb. 2008.
- [25] Z. He, Y. Liang, S. Member, L. Chen, I. Ahmad, S. Member, S. Member, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 5, pp. 645–658, May 2005.
- [26] L. Su, Y. Lu, F. Wu, S. Li, and W. Gao, "Complexity-constrained H.264 video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 4, pp. 477–490, Apr. 2009.
- [27] M. van der Schaar and Y. Andreopoulos, "Rate-distortion-complexity modeling for network and receiver aware adaptation," *Multimedia, IEEE Transactions on*, vol. 7, no. 3, pp. 471–479, 2005.

- [28] M. van der Schaar, D. Turaga, and R. Wong, "Classification-based system for cross-layer optimized wireless video transmission," *IEEE Trans. Multimedia*, vol. 8, no. 5, pp. 1082–1095, Oct. 2006.
- [29] A. Majumda, D. Sachs, I. Kozintsev, K. Ramchandran, and M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 524–534, Jun. 2002.
- [30] Q. Li and M. van der Schaar, "Providing adaptive QoS to layered video over wireless local area networks through real-time retry limit adaptation," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 278–290, Apr. 2004.
- [31] P. Dubey, "A Platform 2015 Workload Model Recognition, Mining and Synthesis Moves Computers to the Era of Tera," Intel Corporation, White paper, 2005.
- [32] "Joint video team reference software JM 15.0." [Online]. Available: <http://iphome.hhi.de/suehring/tml>
- [33] G. J. Sullivan and T. Wiegand, "Video compression—from concepts to the H.264/AVC standard," *Proc. IEEE*, pp. 18–31, 2005.
- [34] "H.264/MPEG-4 AVC reference software manual." [Online]. Available: [http://iphome.hhi.de/suehring/tml/JM%20Reference%20Software%20Manual%20\(JVT-X072\).pdf](http://iphome.hhi.de/suehring/tml/JM%20Reference%20Software%20Manual%20(JVT-X072).pdf)
- [35] D. Marpe, H. Schwarz, G. Blttermann, G. Heising, and T. Wieg, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.
- [36] M. Elgebal, A. Fahim, I. Kang, and M. Sachdev, "Robust and efficient dynamic voltage scaling architecture," in *Proc. SOC Conf.*, 2003, pp. 155–158.
- [37] M. Elgebal and M. Sachdev, "Variation-aware adaptive voltage scaling system," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 5, pp. 560–571, May 2007.
- [38] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.
- [39] E. W. Dijkstra, "A note on two problems in connection with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.



**Aashish Pant** received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Delhi, in 2006 and the M.S. degree in electrical engineering from the University of California, Los Angeles, in 2010.

He is currently working as an R&D Engineer in the Design-to-Silicon Division at Mentor Graphics. His research interests include computer-aided design of VLSI circuits and systems, design for manufacturing, lithography and resolution enhancement techniques.



**Puneet Gupta** received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Delhi, in 2000 and the Ph.D. degree from the University of California, San Diego, in 2007.

He is currently a faculty member of the Department of Electrical Engineering, University of California, Los Angeles (UCLA). He co-founded Blaze DFM, Inc. (acquired by Tela, Inc.), in 2004, and served as its Product Architect until 2007. He has authored over 70 papers, ten U.S. patents, and a book chapter. His research has focused on building

high-value bridges across application-architecture-implementation-fabrication interfaces for lowered cost and power, increased yield, and improved predictability of integrated circuits and systems.

Dr. Gupta is a recipient of the NSF CAREER Award, the ACM/SIGDA Outstanding New Faculty Award, the European Design Automation Association Outstanding Dissertation Award, and the IBM Ph.D. Fellowship. He has given tutorial talks at DAC, ICCAD, International VLSI Design Conference, and the SPIE Advanced Lithography Symposium. He has served on the Technical Program Committee of DAC, ICCAD, ASPDAC, ISQED, ICCD, SLIP, and VLSI Design. He served as the Program Chair of the IEEE DFM&Y Workshop 2009, 2010, and 2011.

**Mihaela van der Schaar** (M'99–SM'04–F'10) is Chancellor's Professor of Electrical Engineering at the University of California, Los Angeles. Her research interests include dynamic multiuser networks and system designs, online learning, network economics and game theory, multimedia networking, communication, processing, and systems, and multimedia stream mining. She holds 33 granted US patents.

Dr. van der Schaar is a Distinguished Lecturer of the Communications Society for 2011–2012, the Editor-in-Chief of the IEEE TRANSACTIONS ON MULTIMEDIA and a member of the Editorial Board of the IEEE JOURNAL ON SELECTED TOPICS IN SIGNAL PROCESSING. She received an NSF CAREER Award (2004), the Best Paper Award from the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2005), the Okawa Foundation Award (2006), the IBM Faculty Award (2005, 2007, 2008), the Most Cited Paper Award from EURASIP: *Image Communications Journal* (2006), the Gamenets Conference Best Paper Award (2011), and the 2011 IEEE Circuits and Systems Society Darlington Award Best Paper Award. She received three ISO awards for her contributions to the MPEG video compression and streaming international standardization activities. For more information about her research visit: <http://medianetlab.ee.ucla.edu/>