Robust and Energy-Efficient DSP Systems via Output Probability Processing

Rami A. Abdallah and Naresh R. Shanbhag
Coordinated Science Laboratory/ECE Department
University of Illinois at Urbana-Champaign, Urbana, IL
email: [rabdall3,shanbhag@illinois.edu]

***Abstract*— This paper proposes to employ error statistics of nanoscale circuit fabrics to design robust energy-efficient digital signal processing (DSP) systems. Architectural level error statistics are exploited to generate probability or the reliability of each output bit of a DSP kernel. The proposed technique is referred to here as bit-level a posteriori probability processing (BLAPP). Energy efficiency and robustness of a 2D discrete cosine transform (2D-DCT) image codec employing BLAPP is studied. Simulations in a commercial $45nm$ CMOS process show that BLAPP provides up to $14X$ improvement in robustness, and 25% power savings over conventional 2D-DCT codec design.**

I. Introduction

Moore's Law, the driving force behind the global semiconductor industry for the last 50 years, is under threat today from artifacts of nanoscale dimensions. Process, voltage and temperature (PVT) variations, leakage, soft errors, and noise in sub-45nm process technologies [1] are conspiring to make it difficult to reap the benefits of feature size scaling due to reliability concerns. A parallel trend is the growing functional complexity and power of next generation applications. A large class of the next generation of applications can be categorized into recognition, mining and synthesis (RMS), where massive amounts of potentially media-intensive data needs to be processed. Such applications rely heavily on digital signal processing (DSP) kernels. The result is a power and reliability problem in nanoscale systems-on-a-chip (SOCs). Reliability and power are interlinked problems viewed by the semiconductor industry as the key inhibiters of Moore's Law. Not surprisingly, since 2001, the International Technology Roadmap for Semiconductors (ITRS) [2] has stated the achievement of reliability and energy-efficiency as two of the important challenges facing the semiconductor industry. Thus, the design of robust and energy-efficient DSP systems is an important area of research.

Robust system design techniques such as *N-modular redundancy* (NMR) (see Fig. 1(a)) are general but employ N-way replication of computation, and thus incur a large power overhead. *Algorithmic noise-tolerance* (ANT) [3]–[5] (see Fig. 1(b)) is a DSP-specific robust system design technique. ANT avoids replication by exploiting the knowledge of signal statistics, statistical signal processing techniques, and the use of statistical performance metrics such as signal-to-noise ratio (SNR) and bit error-rate (BER) in order to detect and correct errors in DSP kernels. Recently, [11] proposed *soft NMR* where error statistics, i.e., the probability mass function (PMF) $P_\eta(\eta)$ is employed to enhance the robustness of NMR. Soft NMR was shown to be highly effective in enhancing robustness while providing energy-savings. Other architectural approaches include RAZOR [6] rely on logic-level timing error-detection circuitry followed by recomputation. The main source of hardware errors in [3], [5], [6], [11] are from timing violations due to *voltage overscaling* (VOS) [3], whereby the supply voltage V_{dd} reduced below $V_{dd-crit}$, the critical supply voltage, i.e., the minimum supply voltage required to avoid any timing violations. VOS offers a well-quantified trade-off between power and error-rates.

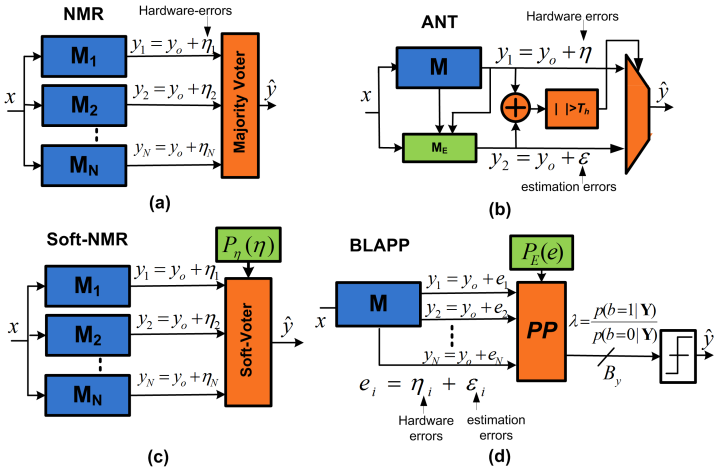

Fig. 1. Architecture-level error resilient techniques: (a) NMR, (b) algorithmic noise-tolerance, (c) soft-NMR, and (d) proposed technique: bit-level a posteriori probability processing (BLAPP).

In this paper, we propose the technique of *bit-level a posteriori probability processing* BLAPP (see Fig. 1(d)) whereby the output and intermediate signals from a single DSP kernel $\mathbf{M}$ are employed to generate multiple observations $\mathbf{Y} = (y_1, y_2, \ldots, y_N)$, where y_i is an estimate of the correct output y_o. Thus, BLAPP avoids hardware replication inherent in NMR and soft NMR. BLAPP employs error statistics, i.e., $P_E(e)$, to compute the bit-level *a posteriori* probability (APP) ratio $\lambda_j = P(b_j = 1|\mathbf{Y})/P(b_j = 0|\mathbf{Y})$ for each output bit b_j $(j = 0, .., B_y - 1)$ of the B_y-bit outputs y_i of the $\mathbf{M}$ block. The slicer then thresholds λ_j to obtain a hard

estimate $\hat{b}_j$. BLAPP is shown to be effective in enhancing robustness and providing power savings. BLAPP is superior to NMR and soft-NMR in that replication overhead is not incurred. BLAPP is an improvement over NMR and ANT in that it exploits error statistics to enhance robustness. Thus, BLAPP combines the best features of soft NMR (exploiting error statistics) and ANT (avoid hardware replication). In addition, we show that the complexity of BLAPP is only a function of output precision, By, and is independent of the main block complexity.

A number of logic level techniques for robust system design have been proposed. Design of general-purpose reliable systems from unreliable components dates back to von Neumann [8] in 1950, where logic level replication with majority voting was proposed to increase resiliency. Logic level approaches such as error-sequential detection [10] and random Markov field [7] have also been proposed. *Stochastic logic* [9], which operates on soft/probabilistic input values at the logic level assuming error-free computational fabric. These logic-level techniques can be complementary to the system-level technique presented in this paper.

The paper is organized as follows: Section III-B presents a motivation example demonstrating the principle behind BLAPP. Section III formally describes BLAPP and its architecture. Section IV demonstrates BLAPP benefits in terms of robustness and energy efficiency in the design of 2D discrete cosine transform (2D-DCT) image codec. Finally, section V concludes and discusses future work.

II. A Unified Framework for Error-resiliency in DSP Systems

This section presents a unified framework (Fig. 1) for describing error-resiliency techniques, in order to relate the proposed BLAPP technique to existing work. The erroneous output y_i of any computational block $\mathbf{M}_i$ is given by:

$$y_i = y_o + \eta_i + \epsilon_i = y_o + e_i \tag{1}$$

where y_o is the correct (error-free) output, η_i and ϵ_i are the hardware and estimation errors, respectively, and $e_i = \eta_i + \epsilon_i$ is the composite error. The set of all possible outputs of $\mathbf{M}_i$ is referred to as the *output space* $\mathcal{Y}$, i.e., y_i, y_o, and $e \in \mathcal{Y}$. We assume that the error statistics are available in the form of a probability mass function (PMF) $P_E(e)$, e.g., as shown in Fig. 2 for a 2-bit output. Note: the topic of error characterization of a DSP kernel is an interesting one in its own right, and can be done in many ways, both off-line as well as via in-situ calibration using typical inputs.

Most system-level robust design techniques can be described by:

1) *Observation vector:* $\mathbf{Y} = (y_1, y_2, \ldots, y_N)$
2) *Decision rule:* $\mathcal{R}$ to generate a corrected/final output $\hat{y}$

For example, NMR (see Fig. 1(a)) employs N-way replication of computation ($\epsilon_i = 0$) to generate $\mathbf{Y}$, and majority or other forms of voting strategies as decision rule. NMR is effective if η_i's are independent, and is described as follows:

1) $\mathbf{Y}_{NMR} = (y_1, y_2, \ldots, y_N)$
2) $\mathcal{R}_{NMR}: \ \hat{y} = Maj\left(\mathbf{Y}_{NMR}\right)$

where $maj()$ is the majority operator. ANT [3] employs an *estimator*/approximate version of the main computation $\mathbf{M}$ to generate an estimate y_2 of y_o as shown in Fig. 1(b). The estimator is a low-complexity version of the $\mathbf{M}$-block and is designed to be error-free of hardware errors ($e = \epsilon$). ANT is described as:

1) $\mathbf{Y}_{ANT} = (y_1 = y_o + \eta, y_2 = y_o + \epsilon)$
2) $\mathcal{R}_{ANT}: \ \hat{y} = \begin{cases} y_1, & \text{if } |y_1 - y_2| < T_h \\ y_2, & \text{otherwise} \end{cases}$

Soft NMR in Fig. 1(c) employs the same observation vector as NMR. However, unlike NMR, soft NMR exploits the hardware error probability mass function (PMF), $P_\eta(\eta)$, to implement a decision rule based on word-level maximum-likelihood (ML) principle to enhance system robustness. Statistics of timing errors due to voltage overscaling were obtained in [11] via a one-time characterization of the $\mathbf{M}$-block. Soft NMR is described as:

1) $\mathbf{Y}_{SNMR} = (y_1, y_2, \ldots, y_N)$
2) $\mathcal{R}_{SNMR}: \ \hat{y} = \underset{y_o}{argmax}\, P\left(y_o | \mathbf{Y}_{SNMR}\right)$

In this paper, we propose BLAPP (see Fig. 1(d)) which consists of the main computational block $\mathbf{M}$, a *a probability processor* (**PP**) and a *slicer*. The main block $\mathbf{M}$ is permitted to exhibit both hardware and estimation errors, i.e., its output is given by (1). Intermediate signals from the $\mathbf{M}$ are employed to generate the observation vector $\mathbf{Y}$, thereby avoiding hardware replication inherent in NMR and soft NMR. The **PP** block employs the composite error PMF $P_E(e = \eta + \varepsilon)$ to compute the a posteriori probability (APP) ratio $\lambda_j = P(b_j = 1|\mathbf{Y})/P(b_j = 0|\mathbf{Y})$ for each output bit b_j ($j = 0, .., B_y - 1$) of the B_y-bit outputs y_i. The slicer then thresholds λ_j to obtain $\hat{b}_j$. Next section, formally describes BLAPP framework.

III. Proposed Technique: Bit-level A Posteriori Probability Processing (BLAPP)

In this section, we present BLAPP in its most general form, illustrate it through an example, and then describe an efficient architecture for the **PP** block in Fig. 1(d).

A. The BLAPP Algorithm

Consider a general computational block $\mathbf{M}$ in Fig. 1(d), whose correct output y_o is represented with B_y bits, $y_o = \{b_j\}_{j=1}^{B_y}$, an output space $\mathcal{Y} = \left\{0, 1, \ldots, 2^{B_y} - 1\right\}$, and an observation vector $\mathbf{Y} = \{y_i\}_{i=1}^{N}$ consisting of N outputs corrupted by hardware and estimation errors $\{e_i = \eta_i + \varepsilon_i\}_{i=1}^{N}$ according to PMFs $\{P_{E_i}(e_i)\}_{i=1}^{N}$, respectively. In most DSP applications, multiple independent output observations can be generated at low overhead by exploiting spatial and temporal correlations.

For each output bit b_j ($j = 1, \ldots, B_y$), we compute the APP ratio λ_j as follows:

$$\lambda_j = \frac{P(b_j = 1|\mathbf{Y})}{P(b_j = 0|\mathbf{Y})} = \frac{P(b_j = 1|\mathbf{Y})}{1 - P(b_j = 1|\mathbf{Y})} \tag{2}$$

In the decision step, we consider the log-domain APP ratio instead of λ_j as it simplifies the algorithm and implementation, and is given by:

$$\Lambda_j = \log \lambda_j = \log \frac{P(b_j = 1|\mathbf{Y})}{P(b_j = 0|\mathbf{Y})} \tag{3}$$

The log-domain APP ratio reflects the confidence in our decision on b_j. From (3), we see that $\Lambda_j = 0$ when $P(b_j = 1|\mathbf{Y}) = 0.5$ and the farther $P(b_j = 1|\mathbf{Y})$ is from 0.5, the larger the magnitude of Λ_j, i.e., $|\Lambda_j|$ indicates our confidence in deciding the value of $\hat{b_j}$, the j^{th} corrected bit. The BLAPP framework for each output bit b_j is described as:

1) $\mathbf{Y}_{BLAPP} = (y_1, y_2, \ldots, y_n)$
2) $\mathcal{R}_{BLAPP}: \quad \hat{b_j} = \begin{cases} 1, & \text{if } \Lambda_j = \log \lambda_j \geq 0 \\ 0, & \text{otherwise} \end{cases}$

Next, we describe how **PP** block computes λ_j or Λ_j in the log-domain. Applying Bayes rule to (2), we obtain:

$$P(b_j = k|\mathbf{Y}) = \frac{P(\mathbf{Y}|b_j = k)P(b_j = k)}{P(\mathbf{Y})} \tag{4}$$

where $P(\mathbf{Y})$ is the probability of observing the vector $\mathbf{Y}$ and $k \in \{0, 1\}$. Substituting (4) in (2), we obtain:

$$\lambda_j = \frac{P(\mathbf{Y}|b_j = 1)P(b_j = 1)}{P(\mathbf{Y}|b_j = 0)P(b_j = 0)} \triangleq \frac{p_{j,1}}{p_{j,0}} \tag{5}$$

where we denoted $p_{j,k} = P(b_j = k|\mathbf{Y})P(b_j = k)$ for $k = 0$ or 1. For each output bit, b_j, the probabilities $\{p_{j,k}\}_{k=0}^{1}$ are generated by a bit-level to word-level mapping in the output space $\mathcal{Y}$ as follows:

$$p_{j,k} = \sum_{y_o \in \mathcal{Y}_{j,k}} P(\mathbf{Y}|y_o)\, P(y_o) \tag{6}$$

where $\mathcal{Y}_{j,k} = \{y_o \in \mathcal{Y} \mid b_j \ \ of\ y_o = k\}$, i.e., the set of all possible outputs that have $b_j = k$, and $P(y_o)$ is the prior probability, i.e., the distribution of the error-fee output.

We assume here errors are independent. Though this is not required, it simplifies decomposition of the joint error PMFs. In DSP applications, independent output observations will generally exhibit independent errors. Assuming error independence, we write

$$\begin{aligned} P(\mathbf{Y}|y_o) &= \prod_{i=1}^{N} P(y_i|y_o) \\ &= \prod_{i=1}^{N} P(e_i = y_i - y_o) = \prod_{i=1}^{N} P_{E_i}(e_i) \end{aligned} \tag{7}$$

Substituting (7) into (6) results in

$$p_{j,k} = \sum_{y \in \mathcal{Y}_{j,k}} \prod_{i=1}^{N} [P_{E_i}(e_i = y_i - y_o)] P(y_o) \tag{8}$$

Substituting in (5), we get

$$\lambda_j = \frac{\sum_{y_o \in \mathcal{Y}_{j,1}} \left[\prod_{i=1}^{N} P_{E_i}(e_i = y_i - y_o)\right] P(y_o)}{\sum_{y_o \in \mathcal{Y}_{j,0}} \left[\prod_{i=1}^{N} P_{E_i}(e_i = y_i - y_o)\right] P(y_o)} \tag{9}$$

Next, we illustrate the BLAPP algorithm with an example.

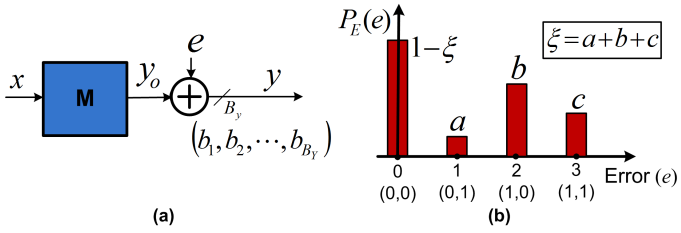

Fig. 2. Error modeling: (a) erroneous DSP block, and (b) a 2-bit sample error PMF.

B. Motivational Example

Consider the DSP kernel **M** in Fig. 2(a) with a $B_y = 2$-bit output $y = (b_1, b_2)$ corrupted by error e according to the PMF in Fig. 2(b), where the *component probability of error* $\xi = P_E(e \neq 0)$. Assume that we rely on a single output observation $y = (b_1, b_2) = (1, 0)$, i.e., $N = 1$, and $\mathbf{Y} = (y = (1, 0))$.

The **PP** block computes $p_{1,1}$ by considering all possible outputs $y_o \in \mathcal{Y}_{1,1} = \{y_o = (1, b_2) \in \mathcal{Y}\}$ as follows:

$$\begin{aligned} p_{1,1} &= \sum_{y_o \in \mathcal{Y}_{1,1}} P(\mathbf{Y}|y_o)\, P(y_o) \\ &= P(\mathbf{Y}|y_o = (1,0))\, P(y_o = (1,0)) \\ &+ P(\mathbf{Y}|y_o = (1,1))\, P(y_o = (1,1)) \end{aligned} \tag{10}$$

Assuming all outputs $y_o \in \mathcal{Y}$ are equally likely to occur, we have $P(y_o = (1,0)) = P(y_o = (1,1)) = 0.25$ in (10). Employing the error PMF $P_E(e)$ in Fig. 2(b), we write (10) as

$$\begin{aligned} p_{1,1} &= P(y = (1,0)|y_o = (1,0)) + P(y = (1,0)|y_o = (1,1)) \\ &= P(e = (0,0)) + P(e = (1,1)) \\ &= (1 - \xi) + c = 0.4 \end{aligned}$$

assuming that the component error probability $\xi = 0.6$, $a = 0.7\xi$, $b = 0.3\xi$, and $c = 0$. Similarly from (6), we get $p_{1,0} = a + b = 0.6$. Thus, the APP ratio of first bit $\lambda_1 = 0.67$ and $\Lambda_j = -0.4$ which is less than 0, and thus the slicer in Fig. 1(d) generates $\hat{b}_1 = 0$. Similarly, one can show that $p_{2,1} = a + c = 0.7\xi = 0.42$, $p_{2,0} = (1 - \xi) + b = 0.58$, $\lambda_j = 0.72$ and $\lambda_j = -0.33$ hence $\hat{b}_2 = 0$. Thus, the **PP** block output followed by the slicer generates the final output $\hat{y} = (0, 0)$ even though the **M** block output is $y = (b_1, b_2) = (1, 0)$, i.e., there is a high probability that b_1 is in error given the knowledge of the error PMF $P_E(e)$.

The benefits of BLAPP are more pronounced over conventional design when multiple output observations are derived from the **M** block. This is easily obtained in DSP kernels because of spatio-temporal correlations present in a typical DSP data-flow graph. Thus, intermediate and past outputs can be employed as additional observations of the correct output y_o. Here, we assume that the observation vector $\mathbf{Y}$ in Fig. 1(d) consists of $N = 3$ outputs $\mathbf{Y} = \{y_1, y_2, y_3\}$. Without any loss of generality, we assume that y_1, y_2, and y_3 are corrupted by independent identically distributed (iid) errors e_1, e_2, and e_3, respectively. In other words,

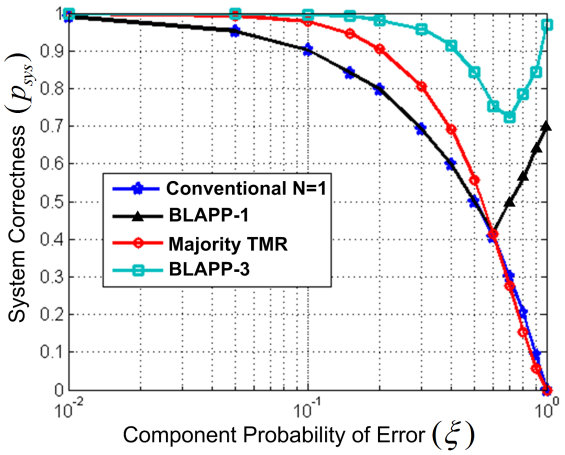

Fig. 3. System correctness of a 2-bit output at different ξ.

the errors e_1, e_2, and e_3 are independent and follow the same error PMF $P_E(e)$ in Fig. 2(b), i.e., $P_{E_1}(e_1) = P_{E_2}(e_2) = P_{E_3}(e_3) = P_E(e)$. If the observation vector $\mathbf{Y} = (y_1 = (1,0), y_2 = (1,0), y_3 = (0,1))$, then TMR selects $\hat{y} = (1,0)$ via a majority vote. On the other hand, a smart voter with the knowledge of error statistics would realize that the correct output $y_o \neq (1,0)$ since $e_3 = y_3 - y_1 = (1,1)$ but $P_{E_3}(e_3 = (1,1)) = c = 0$ (see Fig. 2(b) at $e = 3$ when $c = 0$). For example, in BLAPP, with $N = 3$ (BLAPP-3), the computation of $P(\mathbf{Y}|y_o = (1,1))$ in (10) is given by

$$\begin{aligned} P(\mathbf{Y}|y_o=(1,1)) &= P(e_1=(1,1), e_2=(1,1), e_3=(1,0)) \\ &= P(e_1=(1,1))P(e_2=(1,1))P(e_3=(1,0)) \\ &= (c\xi)^2(b\xi) = bc^2\xi^3 \end{aligned}$$

Similarly, $P(\mathbf{Y}|y_o = (1,0)) = c\xi(1-\xi)^2$. Assuming $\xi = 0.5$, $a = 0.7\xi$, $b = 0.3\xi$, $c = 0$, and equal priors, (10) results in $P(b_1 = 1|\mathbf{Y}) = 0$, i.e., $\lambda_i = 0$ indicating our confidence in the decision $\hat{b}_1 = 0$ is high. Similarly, one can show for the second bit b_2, $p_{2,1} = (a\xi)^2(1-\xi) = 0.07$, $p_{2,0} = (b\xi)^2(a\xi) = 0.019$, $\lambda_j = 3.4$, and $\Lambda_j = 1.22$ and hence $\hat{b}_2 = 1$, resulting the corrected output as $\hat{y} = (0,1)$. We also observe that the log-domain APP ratio moved farther away from 0 with $N = 3$ than that with $N = 1$. This indicates that multiple observations increases our confidence in the decision $\hat{y} = (0,1)$. The effectiveness of BLAPP over conventional design can be calculated for this example by injecting errors at the output of a 2-bit computational kernel according to the PMF in Fig. 2(b) at various component probability of error ξ. BLAPP uses $P_E(e)$ to generate probabilities for b_1 and b_2 from y followed by a slicer to produce a hard estimate $\hat{y}$, while a conventional design directly uses y. The ***system correctness metric***, defined as $P(\hat{y} = y_o) = 1 - P(\hat{y} \neq y_o) = p_{sys}$, is employed to compare BLAPP to conventional design.

From Fig. 3, we observe that BLAPP-3 outperforms TMR for all values of ξ. Second, p_{sys} for both BLAPP-1 and BLAPP-3 increases with ξ for $\xi \geq 0.6$ and $\xi \geq 0.7$, respectively. This unusual outcome is because BLAPP understands that the observations in $\mathbf{Y}$ are unreliable for high values of ξ, and thus tends to choose outputs from $\mathcal{Y}$ that do not belong to $\mathbf{Y}$. Finally, for $\xi \geq 0.6$, p_{sys} for TMR falls below even BLAPP-1 and conventional $N = 1$ system, because the probability of two or more identical errors becomes larger, and hence the majority voter selects the wrong values more often. On the other hand, BLAPP exploits the knowledge of the error distribution in Fig. 2(b), i.e., different error magnitudes have different error probabilities, to correct for errors.

C. The Probability Processor (PP) Architecture

We use the log domain of probabilities to simplify the implementation of the **PP** block. The **PP** computes log-APP ratio $\Lambda_j = \log \lambda_j$. Taking the logarithm (base 2) of (8), we obtain:

$$\log p_{j,k} = \log \sum_{y_o \in \mathcal{Y}_{j,k}} 2^{\left[\sum_{i=1}^{N} \log P_{E_i}(e_i = y_i - y_o)\right] + \log P(y_o)} \tag{11}$$

Using the *log-max* approximation, (11) is approximated as

$$\begin{aligned} \log p_{j,k} &\simeq \max_{y_o \in \mathcal{Y}_{j,k}} \left[\sum_{i=1}^{N} \log P_{E_i}(e_i = y_i - y_o) + \log P(y_o)\right] \\ &= \max_{y_o \in \mathcal{Y}_{j,k}} \Gamma(y_o) + \log P(y_o) \end{aligned} \tag{12}$$

where $\Gamma(y_o)$ is referred to as the *word-metric* and is defined for each $y_o \in \mathcal{Y}$ to be

$$\Gamma(y_o) = \sum_{i=1}^{N} \log P_{E_i}(e_i = y_i - y_o) \tag{13}$$

From (5) and (12), the log-APP ratio Λ_j for bit b_j is computed as follows:

$$\begin{aligned} \Lambda_j &= \log p_{j,1} - \log p_{j,0} \\ &\simeq \max_{y_o \in \mathcal{Y}_{j,1}} [\Gamma(y_o) + \log P(y_o)] - \max_{y_o \in \mathcal{Y}_{j,0}} [\Gamma(y_o) + \log P(y_o)] \end{aligned} \tag{14}$$

where $\Gamma(y_o)$ is given by (13).

The **PP** architecture implementing (14) is shown in Fig. 4. The look-up tables (LUTs) in Fig. 4 store the output prior, $\log P(y_o)$, and error PMFs, $\log P_E(e)$. The **PP** generates Λ_j after 2^{B_y} clock-cycles. During each clock cycle, a $y_o \in \mathcal{Y}$ is fed into the metric unit (MU). The MU uses N-fast latches to compare y_o to each of the N y_i observations and generate $\Gamma(y_o)$ in one clock-cycle. For each b_j, there are two recursive compare-select (CS) units to keep track of the maximum values over $\mathcal{Y}_{j,1}$ and $\mathcal{Y}_{j,0}$, respectively, according to (14).

Architectural Complexity: The complexity of **PP** depends only on the output precision (B_y) and number of observations N and is independent of the main DSP **M** complexity. Thus, as the complexity of **M** increases, BLAPP overhead constitutes a smaller portion of the total system complexity resulting in higher energy and robustness benefits. The storage of prior and error PMF requires storing 2×2^{B_y} values. The **PP** architecture in (Fig. III-C) needs 2^{B_y}

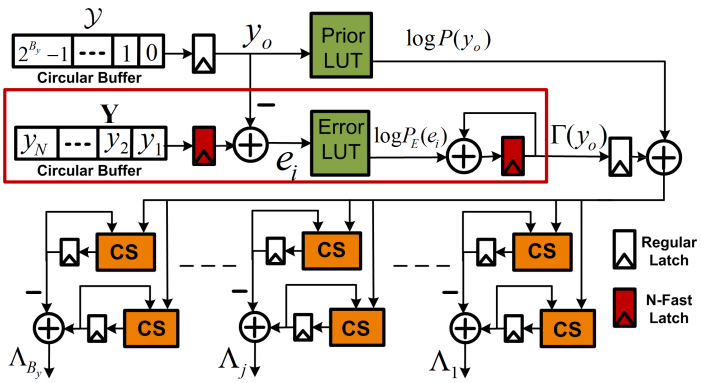

Fig. 4. The **PP** architecture for a B_y-bit output with N observations (MU: metric unit and CS: compare select).

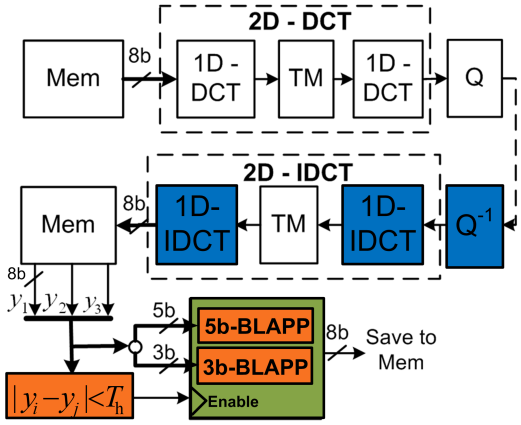

Fig. 5. An 8-bit output BLAPP-based 2D-DCT/IDCT Codec.

clock-cycles to compute Λ_j. Parallelization by a factor of L reduces the number of clock-cycles to $2^{B_y}/L$ but increases hardware complexity. An L-parallel **PP** requires $3L + B_y$ adders and $2LB_y$ 2-operand CS units.

The complexity of a **PP** block can be reduced via *bit-level sub-grouping* BLAPP is applied independently to m groups of $B'_y = B_y/m$ outputs bits. Thus, m B'_y-bit **PP**s are needed instead of a single B_y-bit **PP**, thereby reducing the storage and computational complexity of a fully-parallel **PP** block ($L = 2^{B_y}$) from 2^{B_y} to $m2^{B'_y}$. However, as m increases, the system-level correctness p_{sys} of BLAPP will be reduced due to the loss in information that occurs when error PMFs are characterized over smaller subspaces of the larger output space. In addition, the power overhead can be reduced by activating **PP** only when there is a large difference between the observations y_i, which indicates the presence of a large error.

IV. Simulation Results

We demonstrate benefits of BLAPP in terms of robustness and energy-efficiency in the design of a two-dimensional inverse discrete-cosine transform (2D-IDCT) image codec subject to PVT errors. We exploit application level characteristics to provide different observations to the same output at low-overhead and use the proposed BLAPP techniques to correct for PVT errors and save power.

The DCT-IDCT transform in Fig. 6(a) is applied on 256×256 8-bit pixel images, stored initially in memory (Mem), in blocks of 8×8 pixels using Chen's algorithm [12]. Each 2D transform employs two 1D transforms: the first is applied column-wise on the input block, and the second is applied row-wise on the output of the first. Transposition memory (TM) is used to swap the data between rows and columns. The quantizer (Q) and inverse quantizer (Q^{-1}) employs JPEG quantization table for compression. Only the receiver computational kernels (Q^{-1} and IDCT blocks) are subject to hardware errors. The error-free DCT-IDCT codec achieves a peak signal-to-noise ratio ($PSNR$) of $33dB$, where the $PSNR$ is defined as

$$PSNR = 10log_{10}(\frac{(255)^2}{E[(y_o - \hat{y})^2]}) \quad (15)$$

A. Architectural Set-up

In BLAPP, though spatial independence of errors is not required, it is desirable in order to decrease the storage requirements of the error PMFs. In this work, we resort to application-level data correlations to generate multiple observations with independent hardware and estimation errors. In the IDCT output, adjacent pixels have similar values. The observation vector $\mathbf{Y} = (y_1, y_2, y_3, y_4, y_5)$ is generated by choosing pixels y_1, y_2, y_3, y_4, and y_5, with row and column coordinates (r_i, c_i), (r_{i-1}, c_i), (r_{i-2}, c_i), (r_{i+1}, c_i) and (r_{i+2}, c_i), respectively. Hardware errors in pixels in different rows are made independent since the 1D-IDCT is applied row-wise on the image in the final step.

Given $\mathbf{Y}$, the **PP** block employs a 2^{B_y}-parallel version of the architecture in Fig. 4 which was discussed in section III-C so that it requires one clock-cycle to generate the output APP ratios. To reduce BLAPP complexity and power overhead, bit-level sub-grouping is applied (see section III-C) with $m = 2$, i.e., two **PP**s are employed as shown in Fig. 5. The first **PP** processes the 5 most-significant bits and the second **PP** processes the 3 least-significant bits of the $B_y = 8$-bit output. Sub-grouping results in a transistor-complexity overhead of the **PP**s to be $0.4X$ with respect to the receiver computational kernels (Q^{-1} and 2D-IDCT). To reduce BLAPP power overhead further, the **PP**s are activated only the maximum difference between y_1 and the other four observations in $\mathbf{Y}$ exceeds $T_h = 4$ (see Fig. 5). Also, the **PP**s are operated at their critical supply voltage of $V_{dd-crit} = 0.7V$, to ensure correct operation while consuming minimum power.

B. Error Characterization and Simulation Procedure

Soft-output processing requires statistical characterization of output errors of the DSP computation engine (DCT-IDCT codec). Accurate modeling of errors increases resiliency at the expense of increased storage requirement and search space of BLAPP. Generally, hardware errors are a function of the PVT settings, the input space, and the architecture. Modeling the dependence of error on the input is complex due to the large input space. Alternatively, a training input data I_t can be used to statistically characterize the output er-

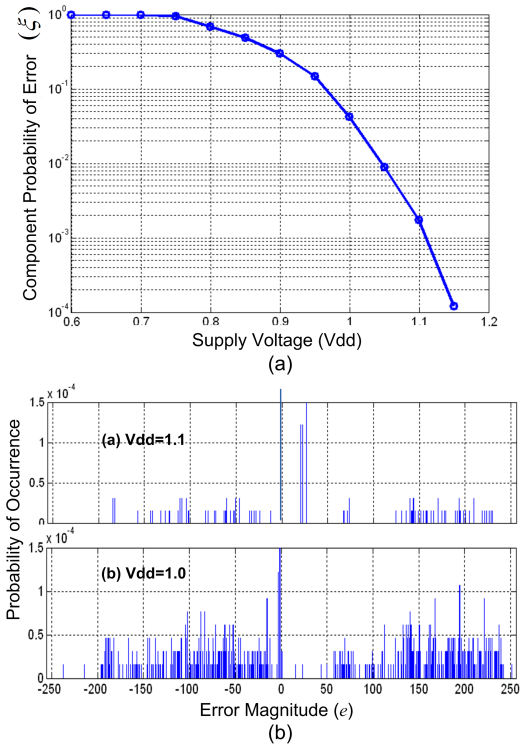

Fig. 6. VOS errors in 2D DCT-IDCT :(a) component probability of error ξ and (b) error PMFs for 2D DCT-IDCT at $V_{dd} = 1.1V$ and $V_{dd} = 1V$.

ror profile at each PVT corner. This captures the dependence of hardware errors on the architecture, indirectly considers the dependence on the input x, and provides the **PP** with a good estimate of the actual output errors. This training phase can be performed either off-chip or on-chip.

We evaluate the robustness and energy efficiency of the proposed soft-output technique under timing violations caused by PVT variations. *Voltage overscaling* (VOS) is used to emulate these timing violations using a commercial 45nm CMOS process. Keeping the frequency fixed, the supply voltage V_{dd} is reduced beyond a critical design voltage $V_{dd-crit}$ so that intermittent timing errors appear. The simulation methodology involves two steps:

1) **Training phase**: An error PMF $P_{E_t}(e_t)$ is obtained via an RTL simulation of main DSP kernel **M** employing a *training* input data-set I_t as follows:
   - Circuit simulations are employed to characterize the worst-case delay vs. V_{dd} relationship of basic gates (1-bit adder, and-gate, or-gate, inverter).
   - At each V_{dd}, an RTL-level structural Verilog model of the PE is simulated at a fixed frequency, using individual gate delays obtained from step 1 and employing data-set I_t. This step generates the erroneous output $y[n]$ in (1).
   - Error PMFs $P_{E_t}(e_t)$s are obtained at each V_{dd} by comparing the $y_o[n]$ and $y[n]$ as shown in (1).
2) **Operational phase**: The kernel **M** operates under VOS on an actual data-set I_a, which is different from I_t, and exhibits I_a dependent-errors. The **PP** employs $P_{E_t}(e_t)$ obtained during the training phase for error correction.

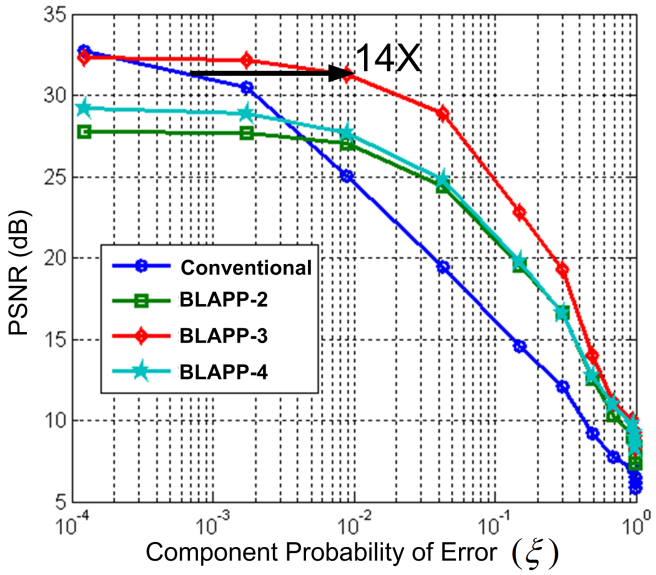

Fig. 7. System Robustness of 2D DCT-IDCT

Figure 6(a) shows the component probability of error ξ at the output of 2D-IDCT as V_{dd} is reduced from 1.2V to 0.6V. Each point on the curve is characterized by an error PMF. For example, error PMFs for the 8-bit output 2D-IDCT at 1.1V and 1.0V are shown in Fig. 6(b). As voltage is reduced, more spread in error values is observed as more circuit paths begin to fail.

C. System Performance

The $PSNR$ for the conventional IDCT codec output and with BLAPP is shown in Fig. 7 under different component probability of error ξ corresponding to different V_{dd}. Figure 6(a) can be used to relate ξ to V_{dd}. We consider BLAPP with N=2, 3, and 4 observations and denote the corresponding designs as BLAPP-2, BLAPP-3, and BLAPP-4, respectively. In each design, the N-element observation vector is obtained as explained in section IV-A. In Fig. 7, as ξ increases, BLAPP-3 is able to provide a $PSNR$ of $31dB$ but in the presence of a $14X$ increase in component error probability ξ over the conventional IDCT. In addition, BLAPP-2 performs better than conventional design but only when $\xi > 0.005$. This is because for $\xi < 0.005$, estimation errors dominate and the **PP** is not able to determine which of the two estimate is correct. BLAPP-4 shows worse performance than BLAPP-3 since estimation errors increases as more adjacent pixels are employed. Thus, the performance of BLAPP depends upon the relative contribution of estimation and hardware errors to the $PSNR$. Figure 8 shows that at $\xi = 0.04$, BLAPP-3 shows a $9dB$ improvement in $PSNR$, and provides a perceptually superior image.

D. Power Savings

HSPICE is used to estimate the power consumption of the gate library at different V_{dd}'s in a commercial 45nm CMOS

(a) (b) (c) (d)

Fig. 8. Output image: (a) original image, (b) error-free IDCT ($\xi = 0, PSNR = 33dB$), (c) erroneous IDCT ($\xi = 0.04, PSNR = 20dB$), and (d) BLAPP-3 IDCT ($\xi = 0.04, PSNR = 29dB$)

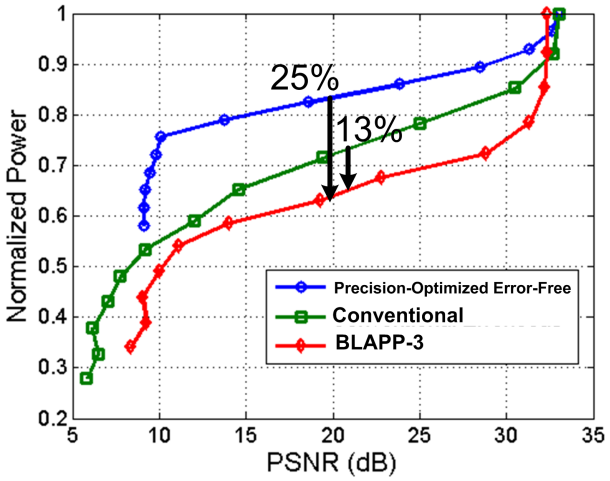

Fig. 9. BLAPP power savings in 45nm process

process. The total power for the computational kernel (2D-IDCT) and **PP**s is obtained by summing up the individual power of constituent gates (for **PP** see the architecture in Fig. 4 and the block diagram in Fig. 5). Since the **PP**s are activated only when a large difference is observed among the observations (see Fig. 5), BLAPP power overhead is scaled by an activation factor of $1-(1-\xi)^N$. Figure 9 shows the power consumption at each $PSNR$ in Fig. 7 for three architectures: 1) precision-optimized error-free (POEF) architecture, 2) conventional, and 3) BLAPP-3. The POEF architecture has its precision reduced in order to operate at the same $PSNR$ as the other two, but in an error free manner. The conventional architecture is error-free only at $PSNR = 33dB$. It is clear that BLAPP achieves a 13% and 25% power savings when compared to conventional and POEF architectures, respectively. These power savings apply for a wide range of $PSNR$s ranging from $32dB$ down to $15dB$, indicating that error-resilient systems are, indeed, efficient in saving power and increasing robustness.

V. Conclusions and Future Work

We presented a technique to design robust systems by exploiting error statistics. Techniques from detection and estimation theory were employed to generate reliability information on each output bit. This work opens up a number of interesting problems to explore such as: a) complexity vs. robustness trade-offs in BLAPP, b) impact of error PMF profiles on robustness and engineering desirable error PMFs, and c) developing iterative/turbo versions to the proposed technique where different probability processors exchange their APP ratios.

VI. Acknowledgments

The authors acknowledge the support of Texas Instruments and the Gigascale Systems Research Center (GSRC), one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity.

References

[1] S. Borkar, V. De and T. Karnik,"Sub-90nm technologies: challenges and opportunities for CAD," in *Proc. of Int. Conf. on CAD*, Nov. 2002, pp. 203-206.

[2] International Technology Roadmap for Semiconductors (www.itrs.net).

[3] N. Shanbhag, "Reliable and energy-efficient digital signal processing," in *Proc. of Design Automation Conf.*, Jun. 2002, pp. 830835.

[4] F. Kurdahi et al., "System-level SRAM yield enhancement," in *Proc. Int. Sym. Quality Electronics Design*, March 2006, pp. 179-184.

[5] G. Karakonstantis, N. Banerjee, K. Roy and C. Chakrabarti, "Design methodology to trade off power, output quality and error resiliency: application to color interpolation filtering," in *Proc. of Int. Conf. on CAD*, Nov. 2007, pp. 199-204.

[6] T. Austin, V. Bertacco, D. Blaauw and T. Mudge, "Opportunities and challenges for better than worst-case design," in *Proc. of Asia-South Pacific Design Automation Conf.*, Jan. 2005, pp. 2-7.

[7] R. Bahar, J. Mundy and J. Chen, "A probabilistic-based design methodology for nanoscale Computation," in *Proc. of Int. Conf. on CAD*, Nov. 2003, pp. 480-486.

[8] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," pp. 43-98 in Automata Studies, Princeton Univ. Press, Princeton, N.J. (1956).

[9] W. Qian, M. Riedel, K. Barzagan and D. Lilja, "The synthesis of combinational logic to generate probabilities," in *Proc. of Int. Conf. on CAD*, Nov. 2009, pp. 367-374.

[10] J. Tschanz, K. Bowman et al, "Resilient circuits enabling energy-efficient performance and reliability," in *Proc. of Int. Conf. on CAD*, Nov. 2009, pp. 71-73

[11] E. Kim, R. Abdallah, and N. Shanbhag, "Soft NMR: exploiting statistics for energy-efficiency," in *Proc. Int. Sym System-on-Chip*, Oct. 2009.

[12] W.-H. Chen, C. Smith, and S. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Tran. on Comm.*, vol. 25, no. 9, pp. 1004-1009, Sep. 1977.

[13] J. Rabaey, D. Burke, K. Lutz and J. Wawrzynek, "Workloads of the future," *IEEE Design and Test of Computers*, vol. 25, no. 4, pp. 358-365, July-August 2008.