A Low Power JPEG2000 Encoder With Iterative and Fault Tolerant Error Concealment

Mohammad A. Makhzan (Avesta Sasan), Student Member, IEEE, Amin Khajeh, Student Member, IEEE, Ahmed Eltawil, Member, IEEE, and Fadi J. Kurdahi, Fellow, IEEE

Abstract—This paper presents a novel approach to reduce power in multimedia devices. Specifically, we focus on JPEG2000 as a case study. This paper indicates that by utilizing the in-built error resiliency of multimedia content, and the disjoint nature of the encoding and decoding processes, ultra low power architectures that are hardware fault tolerant can be conceived. These architectures utilize aggressive voltage scaling to conserve power at the encoder side while incurring extra processing requirements at the decoder to blindly detect and correct for encoder hardware induced errors. Simulations indicate a reduction of up to 35% in encoder power depending on the choice of technology for a 65-nm CMOS process.

Index Terms—Design, iterative, fault tolerant, JPEG2000, low power, multimedia, process variation, random dopant fluctuation (RDF), SRAM, wavelet.

I. INTRODUCTION

JPEG2000 is an example of many multimedia applications that require significant processing power yet must be implemented in an increasingly power-thrifty and cost-conscious context in order to exist in handheld devices. The push to reduce power and cost have necessitated increasing levels of integration, leading to systems-on-chip (SoC) that integrate interface, processing, and storage on a single silicon chip. These SoCs, tend to have significant amounts of memory needed to store input, output, and intermediate forms of media (e.g., images, audio, or video) during the encoding and/or decoding process. The increasingly demanding requirements on image and display resolution has resulted in the need for large memories to store such images, to the point where memory is already the dominant part in most advanced SoCs today—a trend that is expected to increase for the foreseeable future.

The increasing amount of on-chip memory, while beneficial in reducing cost, has a two-fold shortcoming: 1) power consumption metrics become dominated by the metrics of the embedded memory and 2) yield of the overall integrated circuit is negatively impacted since the defect density of memories is roughly twice as much as logic [18]. The problem becomes compounded in advanced CMOS technologies where random dopant fluctuation (RDF) [1]–[3] is starting to become the dominant factor causing intra-die variations. Intra-die variations shift the process parameters of different transistors in a die in different directions, which can result in significant mismatch between neighboring transistors leading to memory cell instability and failure [1]–[4]. These failures are manifested as either an increase in the cell access time or unstable read and write operations. In this paper, we will focus on parametric failures, which are failures caused by process variations that can be lumped into an effective change in the individual transistor threshold voltage. Since for a single memory cell the transistors are in close proximity, RDF will be the primary cause of mismatches between the transistors. To counter the effect of RDF induced faults designers typically raise the supply voltage of the circuit to assure the masking of these errors. This approach in turn creates an undesired relation between yield and power consumption, where power consumption (voltage supply) must be high to ensure a high yield. This statement counters the well known technique of supply scaling to reduce power consumption, where in CMOS, the exponential relation of the leakage power with the supply voltage [5] and the square relation of the dynamic power with $V_{dd}$ [5], are typically used as one of the most effective means of reducing power consumption.

An in-depth view of these two issues namely power and yield, indicates that they are really two facets of the same problem. If memory dominated systems can be designed to be fault tolerant, they will also be low-power by definition. While it is true that some applications cannot function unless the data processed is 100% correct (such as processor instruction code), there exists a broad family of applications that are inherently fault tolerant such as wireless applications and multimedia systems. The authors have shown in previous publications [10], [11] that maintaining 100% correctness in such embedded systems is not always the optimum power saving approach. Furthermore, with technology scaling, it is becoming increasingly difficult to meet 100% correctness even at nominal operation condition. In fact, for applications such as multimedia, the system is designed to be inherently resilient to errors introduced by the transmission medium. Furthermore, the constraints imposed on the quality of the output are soft and statistical in nature. The data itself also possesses a significant amount of spatial and temporal redundancy that greatly facilitates the recovery of errors. This algorithmic resilience to errors can be exploited and codesigned with the hardware circuitry in mind to provide resilience not only to transmission faults but also to hardware-induced faults. Resilience to hardware-induced faults makes power reduction techniques such as aggressive $V_{dd}$ scaling feasible, while increasing the effective yield of the chip via masking errors. As features sizes shrink, the rate of operation condition induced errors becomes significantly higher [19] and as a result the cost of

Manuscript received April 02, 2007; revised August 08, 2007. First published April 17, 2009; current version published May 20, 2009.

The authors are with the Electrical Engineering and Computer Science Department, University of California, Irvine, CA 92697-2625 USA (e-mail: mmakhzan@uci.edu; kurdahi@uci.edu; akhajehd@uci.edu; aeltawil@uci.edu).

Digital Object Identifier 10.1109/TVLSI.2009.2016714

1063-8210/$25.00 © 2009 IEEE
correcting them increases significantly. Fig. 1 illustrates the relation between supply voltage and probability of bit error $P(e)$ for an SRAM memory cell in 65-nm CMOS technology when the “safe” operating point is at 0.8 V [19]. It can be seen that by reducing the supply voltage a corresponding increase in $P(e)$ is observed. The question that we are investigating in this paper is: If we take the application in consideration (JPEG2000), can we design a circuit architecture that can achieve the same applications metrics (PSNR, etc.) while running at a lower than nominal voltage?

In cases where chip-level errors are relatively large, it is possible to introduce some system (or algorithmic) level modifications to restore the output to a level of quality similar to the one achieved using a perfect chip. This way we could use partially defective memory chips to increase the production yield. In addition, since memory is the dominating area of the chip, and also the dominating power consumption component, we could lower the memory supply voltage in the encoder (increasing number of defective cells), and deal with the effects of the defective cells at the system level and at the decoding time. Furthermore, one can utilize the disjoint nature of encoding and decoding, where typically the encoder is embedded in mobile power sensitive devices (cell phone, digital cameras, etc.), while the decoder is a viewing device such as a laptop or a screen which are much less power sensitive. The complexity of detecting and correcting for these errors is shifted to the decoder side which as stated previously is typically less sensitive to power levels.

This paper is organized as follows. Section II presents an overview of the JPEG2000 encoding process, while Section III analyzes the impact of memory errors on the output of the system in terms of performance metrics such as peak signal to noise ratio. Section IV discusses the error detection algorithm at the decoder, while Section V discusses the mapping of a specific error pattern to a unique coefficient. Section VI, then discusses how to pinpoint the exact bit that is faulty. Simulation results are presented in Section VII and power consumption savings are illustrated in Section VIII. Finally, appropriate conclusions are drawn in Section IX.

II. JPEG2000 ENCODER SOC ARCHITECTURE

JPEG2000 is the latest compression standard for still images [8]. Due to the adaptations of the discrete wavelet transform (DWT) and the adaptive binary arithmetic coder (Tier-1), JPEG2000 provides a rich set of features not available in its predecessor JPEG. In particular, the core algorithm of the standard addresses some of the shortcomings of baseline JPEG by supporting features like superior low bit-rate performance, lossy to lossless compression, multiple resolution representation, embedded bit-stream, and so forth. Several JPEG2000 encoder architectures have been implemented [6]–[9]. Additionally, Motion JPEG2000 is being touted as an alternative to temporal-based MPEG as a means of achieving high quality video compression schemes [20], [21]. This is especially the case in devices where both still and moving image recording and playback is needed such as digital cameras and cellular phones.

The flow of steps in the JPEG2000 encoding scheme are shown in Fig. 2. The decoding process is symmetric to the encoding but in the reverse direction. Following is a quick overview of the encoding steps, for the details; the reader is referred to the standard [8]. During encoding, the image is partitioned into rectangular and non-overlapping tiles which are processed separately. Next, unsigned sample values are shifted to make them symmetric around zero. Counting the optional Inter Component Transform (ICT), those procedures are summarized as preprocessing. The dyadic DWT is applied on the tile repeatedly to de-correlate it into different decomposition levels (resolutions). For lossy compression, the wavelet coefficients are fed to a uniform quantizer with central dead zone. Each sub-band is further divided into rectangular blocks namely, code blocks, which are entropy-coded independently. In JPEG2000, entropy coding is two-tiered. Tier-1 is a context-based adaptive arithmetic coder composed of bit-plane coder (BPC) and binary arithmetic coder (MQ-coder). It accepts the quantized wavelet coefficients along with their corresponding probability estimates generated by the BPC, and produces a highly compressed code-stream. This code-stream is carefully organized by Tier-2 coder, constructing a flexible formatted file. New terms and techniques like precinct, packet, tag-tree, and rate control enable features like random access, region of interest coding and scalability.

Based on the profiling results [6] hardware/software partitioning schemes found in typical architectures allocate DWT, Quantizer and Tier-1 coder into hardware while the rest runs as software on the host processor. Fig. 3 shows the architecture of a JPEG2000 encoder SoC. Due to the large image sizes involved in today’s imaging appliances (e.g., digital cameras) it is often the case that the tile memory is too large to be on chip and intermediate computations are typically stored in off-chip memory (e.g., SRAM/
SDRAM). With more advanced processes and the maturity of embedded memories, it is possible to integrate this memory on chip and therefore reduce power consumption and component count in an imaging appliance. Indeed in [13] it is shown that embedded memory can reduce power consumption by 22% compared to a multi-chip solution. The latest JPEG 2000 Codec chip from Analog Device, the ADV202 [16] contains over 1 MB of image buffer memory. However, such additional memory (which would be in the 5–10 MB range when a complete high resolution image is to be stored on-chip) would adversely affect the overall SoC’s yield and power consumption as explained in Section I. The proposed approach of fault tolerance can be readily applied here in order to reap the benefits of the on-chip memory while mitigating the disadvantage of lower chip yield and higher power consumption. The first step in developing such an approach is to assess the impact of memory errors on the encoder output. In order to simulate a defective memory and analyze the effect of the errors we designed an error injection method mimicking the error model of a typical memory cell. Section III explains the error injector design and comparison metrics that were used.

III. ANALYZING THE IMPACT OF MEMORY ERRORS ON THE SYSTEM OUTPUT

A. Simulation Setup

To analyze how using a defective memory affects the quality of the JPEG image, we identified when and how the image data is saved in memory. Then we corrupted the memory image data at that point. Based on currently developed SoC architectures of JPEG2000 [6]–[9], the image data is once saved at the initiation of the encoding process and then after the discrete wavelet transformation “DWT” step (and before quantization). At the initiation of the encoding process, data is in a simple color format, such as RGB or YCrCb. Storing the image data in a defective memory as presented in Fig. 4(c) will result in changes in the RGB pixel values. This will result in salt and pepper like noise. Salt and pepper noise is well understood in the image processing fields and a number of filters, (e.g., median) have been devised as solutions to nearly eliminate such noise [13]. In this paper, we concentrate on developing solutions to compensate for the corruption of intermediate data when stored in the defective memory. To our knowledge this is the first concealment algorithm that provides detection, and correction capabilities for errors introduced after DWT step during the write and read to/from embedded memory.

After the DWT step, as shown in Fig. 4(c) data is saved in memory in the form of a wavelet, with each coefficient in the wavelet being 16 bits long. Each defective bit corrupts the coefficient that contains that bit. Corruption of one coefficient, as we will explain later in this paper, will result in corruption of all color values in the sampling window of that coefficient.

In this work, we assumed that the source of memory defects is parametric errors resulting from variation in process parameters in small geometry devices. The distribution of such parametric errors is random and uniform throughout the memory [1]–[3]. Therefore, in the error injection step we randomly and uniformly corrupt the coefficient values by producing stuck at zero/one errors. In addition to error injection we need a comparison metric to compare the quality of decoded images with the original image. To achieve this, we have used the peak signal to noise ratio (PSNR) which is defined as following:

\[
\text{MSE} = \frac{\sum_{M,N} (I_1[m,n] - I_2[m,n])^2}{M \times N}
\]

(1)

\[
\text{PSNR} = 10 \times \log_{10} \left( \frac{R^2}{\text{MSE}} \right)
\]

(2)

In (2), \(M\) and \(N\) are the number of rows and columns in the input images, respectively, and \(R\) in the PSNR equation is the maximum pixel value in the image.

B. Effect of Injected Errors

Consider the flow of encoding in Fig. 2. Following the 2-D DWT step, the image is divided into subbands. A 2-D DWT could be obtained by cascaded application of two one dimensional DWT steps, first in the vertical and then in the horizontal direction. Vertical application of 1-D-DWT generates high frequency (H) and low frequency (L) coefficients. The following horizontal 1-D-DWT produces the areas LL, HH, HL, and LH.
is the change in the center H (ill) matrix of color values in the decoded
tinction of different combination of high and low pass filters. Coef-
830 IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 17, NO. 6, JUNE 2009
ficients could also be arranged in an alternative way interleaving
the coefficient of each quadrant as shown in Fig. 5.
The LL sub-band could be further transformed into smaller
sub-bands by going through another 2-D-DWT producing
LL4H, LLHL, LLHL, LLLL sub-bands. This process could con-
tinue to any number of levels of transformation. As a tradeoff
between quality and compression ratio, the 2-D-DWT could be
either lossy (9/7) or lossless (5/3) depending on the choice of
wavelet filters [8].
The wavelet coefficients resulting from the DWT step are
stored in memory as shown in Fig. 3. If a memory location is
defective, the coefficient mapped to that location will be defec-
tive and will therefore the transmitted stream will be altered.
On the decoder side, the inverse DWT is made possible by
the application of two cascaded inverse DWTs one in hori-
zontal (HIDWT) followed by another in the vertical direction
(VIDWT). Let us consider the case that the HIDWT is applied
followed by VIDWT. In the first step, a defect in a coefficient
will propagate to all the coefficients which depend on the
defective coefficient. We call these ill coefficients since their
values are altered due to the defective coefficient. Depending on
the length of the high or low pass filter a number of generated
coefficients will be altered. In the next step, the coefficients
are taken through VIDWT. Just like the previous case, any
regenerated color/coefficient value that uses one of the ill
coefficients will also be altered. This process is illustrated in
Fig. 6(a)–(c) for a defect in HH area. Equations (3) and (4) are
the inverse DWT equations for the interleaved presentation of
the (9,7) DWT.

With the following coefficient values:

\[
\begin{align*}
C_{10} &= 1.1150870524 \\
C_{12} &= -0.057543262 \\
C_{13} &= -0.0168641184 \\
C_{h0} &= -0.629490182 \\
C_{h1} &= 0.5912717631 \\
C_{h2} &= 0.0782232665 \\
C_{h3} &= -0.0912717631 \\
C_{h4} &= -0.0267487574 .
\end{align*}
\]

As an example, we consider the case of how a defective HH
sub-band propagates through IDWT. Following HIDWT, a de-
fect in an HH coefficient will generate nine ill coefficients in the
same row centered at the location of the defective coefficient.
These nine ill coefficients are represented as a 1 × 9 Matrix M1.
Another 1 × 9 matrix, M0 holds the coefficients that were gen-
erated if that HH coefficient was not defective. The difference
matrix, \(M_1 - M_0(5)\) accounts for the changes induced by
the defective HH coefficients. Using the inverse DWT (3), (4) we
obtain the individual components of \((M_1 - M_0)\)

\[
\begin{align*}
(M_1 - M_0)[1] &= X_{2n+1} \delta \Delta c = C_{h1} \Delta c \\
(M_1 - M_0)[2] &= X_{2n+1} \delta \Delta c = C_{h3} \Delta c \\
(M_1 - M_0)[3] &= X_{2n+1} \delta \Delta c = C_{h2} \Delta c \\
\cdots
\end{align*}
\]

\[
(M_1 - M_0)[9] = X_{2n+1} \delta \Delta c = C_{h4} \Delta c
\]

\[
M_1 - M_0 = [C_{h1}, C_{h3}, C_{h2}, C_{h1}, C_{h0}, C_{h1}, C_{h2}, C_{h3}, C_{h4} \Delta c .
\]

(5)

In which \(\Delta c\) is the change in the value of the original HH co-
efficient. So after inverse DWT in the horizontal direction, one
defective HH coefficient generates nine ill H coefficients. After
going through VIDWT, each one of these ill coefficients will
result in nine altered color values vertically centered at the ill
coefficient. Since the same equations is used for VIDWT, the
matrix of changes in the vertical direction for each column is
\([M_1 - M_0]^T \Delta h\) in which \(\Delta h\) is the change in the center H (ill)
coefficient. Thus, a 9 × 9 matrix of color values in the decoded
image are altered by the defective HH coefficient. The 9 × 9 ma-
trix SS could be written as

\[
SS = [M_1 - M_0] \times [M_1 - M_0]^T \Delta c .
\]

(6)

To illustrate how this matrix looks like, we scaled it 10,000 times
and truncated the digits after decimal point. The result is shown
in Table I. Such alteration in color values translates into appear-
ance of artifacts in the image. The artifact corresponding to an
HH defect is shown in Fig. 7 which is well accounted for by
looking at the matrix of altered color values given in Table I.

The amount by which each coefficient is altered depends on:
1) the distance from the location of defective coefficients and 2)
the bit position of the defect within the coefficient. A defect in
the most significant bit causes much larger visibility/alteration
than a defect in LSB. In fact, presence of defects in LSB bits,
when a lossy (9,7) DWT is used, is negligible. The degradation
is negligible since in any case we loose some minimal data when
using lossy DWT transformation. This allows us to focus on
detection and correction of defects in the MSB bits.
TABLE I

| Location of Defective HH Coefficient, Values are Multiplied by 10,000 and Rounded |
|---|---|---|---|---|---|---|---|
| 7 | 5 | -21 | -71 | 161 | -71 | -21 | 5 | 7 |
| 5 | 3 | -13 | -45 | 102 | -45 | -13 | 3 | 5 |
| 21 | -13 | 61 | 209 | -472 | 209 | 61 | -13 | -21 |
| -71 | 71 | 209 | 712 | -1609 | 712 | 209 | -45 | -71 |
| 161 | 102 | -1609 | 3635 | -1609 | -472 | 102 | 161 |
| -71 | -45 | 209 | 71 | -1609 | 712 | 209 | -45 | -71 |
| -21 | -13 | 61 | 209 | 21 | 209 | 61 | -13 | -21 |
| 5 | 3 | -13 | -45 | 102 | -5 | -13 | 3 | 5 |
| 7 | 5 | -21 | -71 | 161 | -71 | -7 | 5 | 7 |

IV. ERROR DETECTION BASED ON PATTERN RECOGNITION

In the first part of this paper, we hypothesized that we can construct an encoder that uses memories that are very low cost and consume very low power by allowing these memories to exhibit a certain bit error rate. The problem is now translated to the decoder that has to blindly identify the location of these errors and correct for them. The key to this ability is the fact that errors in different sub-bands have unique signatures in the corresponding domains.

Identifying these signatures ultimately leads to identifying which coefficient is faulty and ultimately recovering the right value. Since comparison based pattern recognition techniques are highly expensive, we developed a faster and smarter technique that recognizes the patterns by exploring their curvature changes. We developed two diagnosis filters, one to detect positive curvature changes and the other to detect negative and no curvature changes. Fig. 8 illustrates the proposed positive and negative curvature change detection filters. These filters are specific to (9,7) lossy transformation. For different transformation waveforms other filters could be developed. In order to use these filters, a linear preprocessing stage is performed. In the preprocessing stage, a unidirectional second derivative of the image is taken using the following linear formula:

$$\frac{d^2}{dx^2} P_i = 2P_i - P_{i-1} - P_{i-2}. \quad (7)$$

A small threshold value is used to binarize the image. The second derivative filter produces a spike whenever it detects a curvature change. The direction of curvature change dictates the sign of the spike. Fig. 6(d) presents what is produced after second derivative and binarization step for a defective HH coefficient. Binarized second derivative (BSD) information, in conjunction with the knowledge of the size and kind of waveforms.

---

Fig. 6. Propagation of a defect in an HH coefficient to defect pattern in a 9/7 lossy inverse DWT.
that are used for the DWT makes it possible to uncover a pattern of curvature change specific to corruption of coefficients in the different sub-bands. Detection is simply done by sliding a detection filter that is designed based on the characteristics of patterns produced for errors in each sub-band. In our proposed technique, a positive curvature change filter has “1” in locations that it anticipates a positive curvature change and “0” in other locations. The negative curvature change filter has “1” in all locations that it expects a negative curvature change and “0” in all other locations.

The positive or negative filter is applied as a sliding window to the BSD image, and the convolution of the filter and the BSD image is obtained. If the picture is binarized to \{0, 1\}, and the total sum resulted from convolution of filter and BSD image in some point is equal to the sum of all values in the filter, adequate positive or negative curvature changes for an error pattern are found. Please note that both positive and negative curvature change filters should match before we conclude that we have discovered a possible error pattern. Only one of the filters slides through the image until there is a match and then the second filter is applied for confirmation. When an error pattern is recognized, we know that an error exist in some coefficient. But to realize which coefficient, we need to map the error pattern back to the defective coefficient. This mapping is performed using the mapping algorithm which is explained in Section V.

V. MAPPING AN ERROR PATTERN TO A COEFFICIENT

After uncovering an error pattern in the BSD image, we need to identify which coefficient in the wavelet is corrupted. Mapping from RGB channel to a corrupted coefficient is not a straightforward direct map. From each color channel we could go to three different wavelets. In each wavelet, there are M sub-bands, and in each sub-band there are N coefficients.

Coefficients might not be sampled from RGB channels, but from intermediate color presentation (ICP), such as YCrCb, channels. Therefore, if one of the coefficients in the wavelet which is driven from an ICP channel is corrupted, the dependency of RGB color values to the ICP values spreads the error artifact to all three RGB channels. Although this intermediate step makes the mapping slightly more difficult, it has an advantage for the error detection algorithm; if an error in one of the planes is detected, it could be verified by querying whether that error also exist in the other two planes. If wavelets are sampled directly from RGB channels, the mapping to wavelet planes becomes trivial. But still we need to find the sub-band and corrupted coefficient within the sub-band.

A. Mapping to the Coefficient Sub-band

As explained in previous sections, a defect in each sub-band translates to a signature pattern unique to that sub-band (e.g., a plus sign pattern for defective HH coefficients). Therefore, we developed filters to look for such patterns. When a filter matches a pattern in addition to discovering the error, the type of the filter
that has discovered the error reveals the defective coefficient sub-band. For example, if an HH template has matched the error pattern, the error is coming from the HH sub-band.

B. Mapping to the Exact Coefficient

In order to find the exact location of each corrupted coefficient, we use the equations provided in Table II. These equations linearly map the center of defective pattern (CDP) to the coefficient which its center of sampling window is the same as the CDP. Before using these equations, one needs to know the sub-band that the corrupted coefficient resides obtained as described in Section V-A and the Cartesian location of the center pixel of the artifact pattern in the BSD image.

C. Mapping to the Right Coefficient Plane

As mentioned previously, if coefficient wavelets are directly obtained from RGB channels, the mapping is trivial; the R color channel maps to the R coefficient wavelet and so on. But if ICP transformation is used, we need to determine which wavelet plane is hosting the defective coefficient (hosting wavelet). For this purpose we utilized the delta function as shown in (8)

$$
\Delta(C[i_0, j_0]) = \sum_{i=1}^{1} \sum_{j=1}^{1} (C[i_0, j_0] - C[i_0 + i, j_0 + j]).
$$

(VI. BLIND ITERATIVE ERROR CONCEALMENT ALGORITHM)

As previously indicated, we are focusing on memory parametric errors as the source of memory defects. Distribution of parametric errors is random and uniform. Due to the small probability of failure per memory cell, chances of more than one of the coefficient’s MSBs being corrupted is negligible. Therefore, we assume that only one of the most significant bits in each corrupted coefficient is defective. As we stated in (6), a defect pattern strength is directly related to the change in value of corrupted coefficient. Therefore corruption of one of the MSB bits will greatly deteriorate the decoded image quality whereas a defect in LSB bits will be negligible. The number of MSB bits that are detected and corrected as we will explain later in this section depends on the sensitivity of the binarization threshold as well as the number of iterations. Each iteration corrects one of the MSB bits. Since 100% correctness is not a necessity especially in lossy transformation, we ignore the defects in the lower significant bits without noticeable loss in PSNR or image quality. This assumption is the key to a novel blind iterative algorithm to identify which bit is in fault. At any given time, one bit of all defective coefficients is flipped; the image is decoded and analyzed at the error sites to see if the error pattern is again produced. To iteratively decode the image, we should either save the coefficients (if extra memory is available) or each time generates them (adding a half encoder). Fig. 9 illustrates the iterative error concealment flow. After decoding the image for the first time, the second derivative of the image is taken and binarized. Then the error detection and mapping algorithm is activated.
The detection step unravels and maps each error pattern to a defective coefficient. In this step defective coefficient values along with information on their location in RGB and coefficient wavelet are saved in a data base. If it is not the first decoding iteration, there is no need for error detection and mapping since the data base has already stored this information. At the end of each iteration, the error analysis algorithm takes the BSD image and queries the data base for the location of previously recognized defect sites. Analyzing these locations and determines if the error patterns exist. The error analysis algorithm is very similar to error detection in the sense that it uses the BSD filters to qualify the presence of an error pattern. The only difference between the error analysis algorithm and error detection algorithm is that the error detection algorithm slides through the entire image, but the error analysis algorithm queries the data base and examines only the locations that are reported defective. Furthermore, the error detection algorithm examines the BSD image for all possible error patterns whereas the error analysis algorithm only examines the image for one BSD error detection filter. If a defective coefficient is fixed in the previous iteration, its associated error patterns will not be generated. In this case, the data base will be updated to avoid further changes to the repaired coefficient. In the correction step, for each corrupted coefficient, the (MSB - # iteration +1) bit of the coefficient is flipped, and the (MSB - # iteration +2) bit, which in the last iteration that was flipped, is flipped back. In the first iteration only the MSB is flipped. In the last iteration only the (MSB - # iteration +2) bit is flipped back to retrieve the original value of coefficients that are not yet fixed or miss-detected. Coefficient replacement algorithm then takes charge. This algorithm obtains the location of defective coefficients from the data base and replaces their values with values suggested by the correction algorithm. Then data is submitted to the decoder half way through the decoding process, and the iterations continue. If we are only concerned with defects in the “N” MSBs of the coefficient data, the iteration should be done “N + 1” times. The first iteration will detect all the errors and correct the errors in 1st MSB. The next “N - 1” iterations will go through the “Nth-MSB” to “Vth-MSB +1” bits and the last iteration retrieves the original value of the coefficients that are not yet fixed, or miss-detected and decodes the final image.

Another interesting observation is that when a low power encoder is coupled with an iterative decoder after decoding a few images, there is no further need for the error detection step since the location of the errors are fixed. Therefore, as the marriage of encoder and decoder ages, the decoder better understand the defective behavior of the encoder. At the same time the need for long executing iterations for fixing the errors goes away as the decoder based on previous iterations on previous images realizes which bit in each defective coefficient are defective and with only one iteration (to try both bit values), determine the right value of the defective bit. Therefore, the expense of iterative error concealment decreases as the marriage between the encoder and decoder ages.

A. Practical Considerations

When two defective coefficients are spatially close to one other, their produced error patterns coincide. This sometimes results in detection of the stronger artifact and missing the weaker one. After fixing the coefficient that had caused the stronger artifact, the neighboring weaker artifact could easily be discovered. Reapplication of the proposed iterative method unravels and fixes the weaker artifact as well and further improves the image.

The unique characteristic of the proposed iterative error concealment methodology is that it does not estimate a value for a defective coefficient, but it actually retrieves its original value. Therefore, it could be used along with other error concealment methodology as the first concealment method applied to the decoded image. Furthermore, since the error detection algorithm has detected more errors than what is corrected, (possibly in LSBs), the iterative error concealment could return an error map for the coefficients in “MB-N” (with N being the number of iterations) in RGB channels and wavelet channels to be used by other filtering schemes.

VII. SIMULATION RESULTS

The simulation setting of the proposed JPEG2000 encoder is illustrated in Table III. Table IV summarizes the iterative error concealment simulation results with one pass. For each image, the average PSNR value for the color channels R, G, and B over different error rates is provided. The table illustrates the number of defective coefficients (injected errors), number of errors detected and the number of fixed errors for each error percentage. As the error percentage increases, the algorithms’ ability to correct the errors deteriorates because with an increase in the number of errors the possibility of two or more neighboring coefficients being defective increases.

Fig. 10 compares the average PSNR obtained after applying the iterative error concealment with multiple passes. As discussed before, reapplication of the iterative error concealment,
further improves the visual quality of the image by detecting and correcting additional defects. Fig. 10 highlights a tradeoff between error rates in the memory and the expected PSNR. For example, if a target PSNR of 40 dB is required, the designer can choose to implement an error free encoder working at nominal voltage. Alternatively, to achieve a low power solution, the designer can choose an encoder that allows up to 0.28% errors in the memory of the encoder with the knowledge that these errors can be detected and corrected at the decoder side to provide greater than 40 dB PSNR with just one iteration. To quantify the power savings, one can refer back to Fig. 1 and identify that 0.28% memory error can be associated with a reduction of the supply voltage down to 0.65 v rather than the nominal 0.9 V. The power savings associated with Vdd reduction are illustrated in Fig. 11 which shows that a reduction of Vdd from 0.9 to 0.65 V will result in 50% savings in dynamic power and 60% savings in leakage power. These savings are associated with the memory section of the decoder. In Section VIII, we will discuss the power reduction impact on the entire system (of which the memory is a subsystem) using two commercially available JPEG2000 encoders as case studies.

Fig. 12 illustrates the extra effort for a decoder which is supporting the iterative error detection and concealment technique (IDC) described earlier. An encoder with a defective memory containing 0.2% defective cells is assumed at the encoder. A series of images are encoded and provided to the decoder which runs the IDC algorithm. The total number of assembly instruction executed in the decoder at the end of each iteration for each image is obtained and recorded. The average number of instructions for different images is then compared to the average number obtained using a typical receiver without IDC. The first iteration require noticeably more effort compared to other MSB because it is the only step that the detection algorithm executes and detection filters are applied for every single pixel in the image. In the next iterations, the detection filters are only used in the previously detected spots to confirm the resolution or persistence of the defects as a part of analysis algorithm. Each MSB also require less work compared to previous MSB because, in each iteration, some of the defects are corrected and in the next iteration there is no need to run the filters for those spots.

VIII. POWER CONSUMPTION GAIN

We use two existing implementations of JPEG2000 encoders to validate our claims of power reduction. The first one is an intellectual property (IP) block from BARCO-SILEX that is described in [15]. The second system is the ADV202 JPEG2000 codec chip from Analog Devices [16]. The BARCO-SILEX IP implements the JPEG2000 encoding but requires additional tile memory of 512 kb (64 kB) and another 135 kb of additional data storage. In our approach, the tile memory is error resilient and can be supplied at aggressively lower Vdd than the rest of the blocks. We used both TSMC 65-nm LP process and G process standard cell libraries to estimate the power consumption of the core, $P_{Vdd}$. Additionally, we used CACTI [22] to estimate the tile memory’s dynamic power consumption at the system’s required operating frequency and 0.9 V supply, $P_{mem}$. Unfortunately, CACTI’s estimates of leakage are not reliable at technologies below 90 nm so we estimated the leakage at 180 nm and scaled it based on TI’s predictions by $10^x$ [17] to obtain the memory leakage power, $P_{mem}^{leak}$. Fig. 11 shows how $P_{mem}$ and $P_{mem}^{leak}$ scale with Vdd. Since the power savings come only
from scaling to the tile memory, the power savings percentage ratio, \( r \), is obtained by

\[
  r = \frac{P_{\text{mem}}(V_{\text{dd0}}) + P_{\text{mem}}(V_{\text{dd1}})}{P_{\text{mem}}(V_{\text{dd0}}) + P_{\text{mem}}(V_{\text{dd1}}) + P_{\text{logic}}(V_{\text{dd1}})} \tag{9}
\]

where \( V_{\text{dd0}} \) is the nominal supply voltage and \( V_{\text{dd1}} \) is the low supply voltage applied to the tile memories. As we can see the total savings, \( r \) can reach about 17% (25%) when the memory supply is at 0.7 V and 25% (35%) when it is lowered to 0.6 V in the LP process (G process).

The second system we considered is a complete JPEG 2000 codec SoC. The ADV 202 chip from analog devices includes all the memories, interfaces and control processor needed for encoding and decoding JPEG 2000 images. The tile memory is about 512 kB. The chip was manufactured in 180 nm technology so we scaled the logic power to 65 nm. Similarly, we also used CACTI to estimate the memory’s dynamic power and TI’s leakage forecast to estimate its leakage power. Using the same method and (9), the estimated savings are computed and shown in Fig. 14. The savings in total system power are at 10% (16%) when the memory’s \( V_{\text{dd0}} \) is scaled to 0.7 V and 15% (23%) when scaled to 0.6 V in the LP (G) process.

IX. CONCLUSION

In this paper, we presented algorithmic system level techniques to overcome the imperfections in fabricating multimedia chips. We focused on JPEG2000 as a case study. We illustrate that blind techniques could be used at the decoder to detect and correct for encoder induced errors. Relaxation of the 100% correctness bound allows for the implementation of low cost, ultra low power encoders with savings of up to 35% in power possible for 65-nm process technologies.

REFERENCES

Mohammad A. Makhzan (Avesta Sasan) (S’05) received the B.S. degree (summa cum laude) in computer and electrical engineering and the M.S. degree in electrical and computer engineering from the University of California Irvine, Irvine, in 2005 and 2006, respectively, where he is currently pursuing the Ph.D. degree in electrical and computer engineering.

He is currently a Research Staff Member with Jazz Semiconductor, Newport Beach, CA. His research interests include design of low power, fault tolerant and high performance circuits, and architectures for nanoscale technologies with a focus on the effects of parametric process variation.

Dr. Makhzan was a recipient of the Town and Gown Scholarship from UCI in 2004 and the CPCC fellowship in 2006 and 2007 from the Center for Pervasive Communications and Computing, CA.

Amin Khajeh (S’98) was born in Birmingham, U.K., in 1981. He received the Bachelor of Science in electrical engineering and communication from Shiraz University, Iran, in 2002 and the Master of Science in electrical engineering from University of Texas, Arlington, in 2005. He is currently pursuing the Ph.D. degree in electrical and computer engineering from the University of California Irvine, Irvine.

He was an intern with the Research and Development Division, Siemens Company, in the Summer of 2002, where he later became a Research Staff Member from 2002 to 2003. His research interests include design of low power high performance circuits for communication and multimedia applications.

Mr. Khajeh was the recipient of the UT/TeXTEC Scholarship from UT Arlington in 2004 and the CPCC fellowship in 2006 and 2007 from the Center for Pervasive Communications and Computing, CA.

Ahmed M. Eltawil (M’97) received the B.Sc. and M.Sc. degrees with honors from the Electronics and Communications Department, Cairo University, Cairo, Egypt, in 1997 and 1999, respectively, and the doctorate degree from the University of California, Los Angeles, in 2003, where the focus of his research was on the development of highly flexible low power VLSI architectures for wireless broadband communication systems.

He is an Assistant Professor with the Electrical Engineering and Computer Science Department, University of California, Irvine (UCI). Currently at UCI, his research interests are in advanced digital circuit and signal processing architectures for communication systems with a focus on physical layer design for software defined radios. Prior to joining UCI, he was the director of ASIC Engineering at Innovics Wireless, where he worked on diversity enabled third generation W-CDMA mobile transceiver system design.

Dr. Eltawil was a recipient of several awards and patents in his field including being the Henry Samueli faculty fellow of Engineering at the University of California, Irvine.

Fadi J. Kurdahi (M’87–SM’03–F’05) was born in Beirut, Lebanon. He received the Bachelor of Engineering from the American University of Beirut, Beirut, Lebanon, in 1981, the M.S.E.E. and Ph.D. degrees from the University of Southern California, Los Angeles, in 1982 and 1987, respectively.

Since 1987, he has been a faculty member in the Electrical Engineering and Computer Science and Information and Computer Science Departments, University of California, Irvine, where he conducts research in VLSI CAD and reconfigurable computing with applications in communication and multimedia systems.

Prof. Kurdahi was the corecipient of the 2001 IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS Best Paper Award and the 2006 ISQED Best Paper Award, as well as three. He served as program chair and general chair of the 1999 and 2000 International Symposium of System Synthesis. He was an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS in 1993–1995, and Topic Area Editor for reconfigurable computing for IEEE Design and Test (2000-present). He also served on executive committees, steering committees, and technical program committees of numerous IEEE conferences.