Controlled Timing-Error Acceptance for Low Energy IDCT Design

Ku He, Andreas Gerstlauer and Michael Orshansky
University of Texas at Austin, Austin, TX-78712, USA.
Email:kuhe@mail.utexas.edu, gerstl@ece.utexas.edu, orshansky@mail.utexas.edu

***Abstract*— In embedded digital signal processing (DSP) systems, quality is set by a signal-to-noise ratio (SNR) floor. Conventional digital design strategies guarantee timing correctness of all operations, which leaves large quality margins in practical systems and sacrifices energy efficiency. This paper presents techniques to significantly improve energy efficiency by shaping the quality-energy tradeoff achievable via V_{DD} scaling. In an unoptimized design, such scaling leads to rapid loss of quality due to the onset of timing errors. We introduce techniques that modify the behavior of the early and worst timing error offenders to allow for larger V_{DD} reduction.**

We demonstrate the effectiveness of the proposed techniques on a 2D-IDCT design. The design was synthesized using a 45nm standard cell library. The experiments show that up to 45% energy savings can be achieved at a cost of 10dB peak signal-to-noise ratio (PSNR). The resulting PSNR remains above 30dB, which is a commonly accepted value for lossy image and video compression. Achieving such energy savings by direct V_{DD} scaling without the proposed transformations results in a 35dB PSNR loss. The overhead for the needed control logic is less than 3% of the original design.

I. Introduction

The fast-growing market of portable systems with limited battery life requires continued advances in ultra low-energy design. In this paper, we propose techniques that exploit special properties of digital signal processing (DSP) systems to reduce their energy consumption. In conventional DSP designs, as in other digital design flows, timing correctness of all operations is guaranteed by construction. In static timing analysis-driven design methodologies, every path regardless of its likelihood of excitation, must meet timing. Conversely, any timing violations lead to errors. Since in many DSP applications the best signal quality is not required, it is possible to tolerate some timing errors induced by lower V_{DD}. If aggressive voltage scaling can be made possible with only a small, bounded quality loss, it can lead to significantly reduced energy consumption.

Several efforts in the past have explored the possibility of trading quality in DSP systems for lower energy. In [1], [2], energy is reduced by discarding algorithm steps or iterations that contribute less to the final quality. In [3], adaptive precision of the arithmetic unit output is used to save energy. In [4], [5], energy reduction is enabled by using lower voltage on a main computing block and employing a simpler error-correcting block that runs at a higher voltage and is thus, error-free, to improve the results impacted by timing errors of the main block. The most similar approach to ours is described in [6], [7], [8]. In this work, implementation of combinatorial logic blocks is restructured to enable utilization of intermediate results, which are arranged such that the more important ones, from the quality point of view, are obtained first.

An important distinction between prior work and our strategy is that in other work the results produced by blocks subject to timing errors are not directly accepted. From the point of view of gate-level design, such techniques still guarantee timing correctness of all digital operations. In [4], [5], an estimated value of the result is used in downstream computation in case of timing errors. In [6], [7], computation is terminated early and intermediate results impacted by timing errors are ignored entirely. In contrast, our strategy allows using the erroneous results directly, providing, of course, that the magnitude of error is carefully controlled. Experimental results suggest that our approach may require smaller control and compensation overhead. As a result, we are able to achieve larger energy savings in the low range of quality loss.

We also anticipate that our strategy is extendable to a larger class of algorithms. Our approach does not require changing the algorithm itself, e.g. to allow for early termination. Instead, we directly re-design the implementation to tolerate timing errors. Since we only rely on modifying the implementation at the level of core atomic RTL operations, we expect our strategy to have utility in a wider class of DSP algorithms, with the potentially of being automated. Another difference with [6], [7] is that their approach only allows a discrete set of quality-energy points. By contrast, our technique enables a range of trade-offs along a continuous quality-energy profile.

The proposed strategy for timing-error acceptance is based on a statistical treatment of timing errors: while we give up on guaranteeing the worst-case timing, we have to satisfy timing requirements on average to keep signal quality from severe degradation. We advance architecture-level techniques that significantly reduce algorithm quality loss under V_{DD} scaling, compared to direct V_{DD} reduction. This leads to a superior quality-energy tradeoff profile. Fundamentally, this is enabled by (i) reducing the occurrence of early timing errors with large impact on quality, and (ii) using control and data flow analysis to disallow errors that are spread and get amplified as they propagate through the algorithm.

To address the first goal, we specifically focus on the behavior of timing errors in addition as a fundamental building

This work is supported by NSF grant CCF-1018075

978-3-9810801-7-9/DATE11/© 2011 EDAA

block of most signal processing algorithms. Simple analysis shows that the magnitude of timing errors depends on the values of operands. A specific important class of operands leading to early and large-magnitude timing errors is the addition of small numbers with opposing signs. We develop two distinct techniques at two levels of granularity - one at the operation and one at the block level - to reduce such errors. Note that depending on knowledge about data statistics, both techniques can be applied at design or at run time. For the design chosen in this paper, however, we limit discussions to static operation-level and dynamic block-level optimizations.

Combined across both goals, we present three quality-energy (Q-E) optimizations at the operation, block and algorithm levels. Techniques are introduced and demonstrated on the design of an Inverse Discrete Cosine Transform (IDCT) as a widely used image and video processing kernel. Specifically, the key contributions for architecture Q-E profile shaping are:

1) Controlling large-magnitude timing errors in operations by exploiting the knowledge of statistics of operands. In many cases, we have knowledge of data distributions that can be exploited at design or at run time. Specifically, in the IDCT algorithm, high-frequency coefficients tend to have small magnitude values, often of opposite sign. Our technique is based on the realization that an adder with reduced bitwidth can be used to process such operands. Some operands, of course, require a full-width adder. In the IDCT algorithm, the classification can be done at design time, with higher-frequency components being processed in reduced-width adders while the rest of the matrix components are processed on the regular-width adder.

2) Controlling the frequency of error-generating additions by dynamically re-arranging the sequence of operations, e.g. in accumulation. Similar to the previous technique, this strategy aims at reducing the quality loss in addition stemming from processing of small-valued opposite-sign numbers, but at a level higher than that for a single addition. Specifically, it is targeted at reducing the cumulative quality loss resulting from multiple additions. Such multi-operand addition occurs, for example, in accumulation, which is a key component of many DSP algorithms, and, specifically, of IDCT.

3) Preventing occurrence of errors which can spread and get amplified throughout the algorithm. An important aspect of a design methodology that allows some timing errors is controlling the impact of these errors on output quality from the perspective of the entire algorithm. Specifically, a result impacted by timing errors early in the algorithm can have a dramatic impact on the overall quality by affecting downstream computations through repeated reuse of incorrect data. Therefore, we can not afford to allow errors in certain critical steps, and we propose a technique to avoid such errors based on rescheduling of the algorithm.

The rest of the paper is organized as follows, Section II discusses the principle of timing error management, followed by an introduction of the techniques to control such errors; Section III shows the experiment results, and finally, Section IV concludes the paper with a summary and outlook.

II. Timing Error Management

The 2D-IDCT computation can be represented by $I = C^T \cdot A \cdot C$, where C is the orthogonal type-II DCT matrix and A is the spectrum coefficient matrix. It is customary to implement the 2D-IDCT as a sequence of two 1D-IDCTs. For each 1D-IDCT, the core algorithm is a matrix-vector dot product:

$$T(k) = \frac{c(k)}{2} \cdot \sum_{k=0}^{N-1} x(n) cos[\frac{(2n+1)k}{2N}\pi]$$

$$N = 8, c(0) = 1/2, c(k) = 1, 0 \leq k \leq N-1$$

where $x(n)$ is the data being processed.

A. Error control through knowledge of operand statistics

When V_{DD} is scaled down, large magnitude timing errors happen first for additions of small numbers with opposing sign. Such additions lead to long carry chains and are the timing-critical paths in the adder. The worst case for carry propagation occurs in the addition of -1 and 1. In 2's complement representation, this operation triggers the longest possible carry chain and, thus, experiences timing errors first. Crucially, when a timing error occurs, the apparent result will also have a very large possible numerical error due to carry propagation into the MSBs leading to a large magnitude mismatch compared to the error-free result. For example, in an 8-bit computation, the error magnitude can be up to 64.

In the 2D-IDCT algorithm, the additions that involve small-valued, opposite-sign operands occur in the processing of high-frequency components. This is because the first 20 low-frequency components contain about 85% or more of the image energy [8]. Hence, the magnitude of high-frequency components tends to be small, and coefficients follow a Laplace distribution with high probability densities concentrated in a narrow range [9]. Furthermore, the Laplace distributions are zero-centered, which implies that high frequency components also tend to have opposing signs. As such, a significant amount of quality loss at scaled V_{DD} can be attributed to additions involving such components. The first specific technique we employ is based on the realization that an adder with a bitwidth smaller than required by other considerations can be used to process such operands. Two objectives are achieved by using such adders: the magnitude of quality loss is reduced and its onset is delayed. Large-valued operands, of course, require a regular-width adder. Note that in an actual implementation it is possible to utilize a single adder with variable bitwidth.

In the IDCT algorithm, the classification of matrix elements can be done at design time. This raises the question of (a) how to best perform this classification; and (b) how to identify the optimal bitwidth of the reduced-width adder. In the following, we develop a model to enable such a design optimization. We define Adder 1 as the regular-width adder

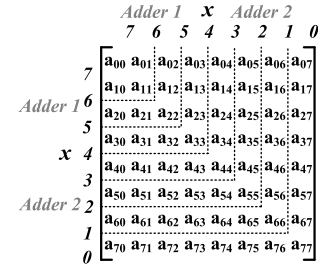
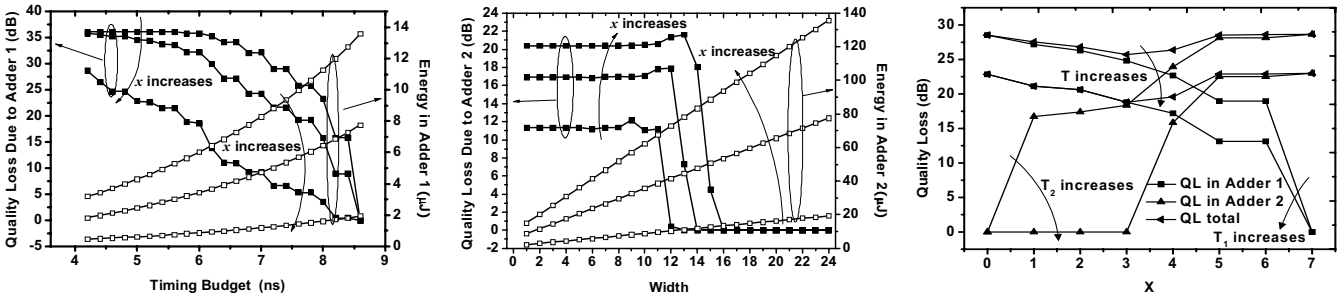


Fig. 1. Partitioning of input matrix.



(a) Energy and quality loss in Adder 1 (b) Energy and quality loss in Adder 2 (c) Quality loss vs. component classification

Fig. 2. Quality-energy tradeoffs in Adder 1 and Adder 2.

and Adder 2 as the reduced-width adder. In classifying the components, we seek to find the boundary, within the data matrix, between the upper-left low-frequency components and the lower-right high-frequency components. We therefore define the following parameters of our model: x: boundary between high-/low-frequency coefficients, where $x = 0$ classifies all inputs as low-frequency processed on Adder 1 (Figure 1); D_1: Worst-case delay of Adder 1; D_2: Worst-case delay of Adder 2; T_1: Timing budget of Adder 1; T_2: Timing budget of Adder 2.

We assume throughout this discussion that $T_2 = D_2$, i.e. that no timing errors are allowed to occur in Adder 2. Furthermore, we assume that $T_1 = T_2$, which implies that both adders are affected by V_{DD} scaling in an identical manner. This assumption is relaxed in Figure 3.

Based on this notation, we can study the Q-E characteristics of the two adders under scaled V_{DD}. By exploring adder characteristics, we are able to identify the optimal partitioning strategy from the point of view of achieving a globally optimal Q-E result. For simplicity, we substitute in this analysis the equivalent notion of timing budget for the value of V_{DD}.

We first study the Q-E relation for the regular width adder, shown in Figure 2(a). The right axis shows the energy value at different timing budgets T_1. As expected, allotting a smaller timing budget, which entails an equivalent lowering of V_{DD}, results in a reduction of energy. Increasing the number of matrix components processed in the reduced-width adder, i.e. increasing x, results in fewer additions performed by Adder 1, and thus a lower energy at the same timing budget. The quality loss (shown on the left axis) is initially low when the allotted timing budget is high and few computations experience error. As T_1 is reduced, however, we begin to observe that the quality loss is smaller for larger x. This corresponds to the scenario in which fewer operations are performed by Adder 1, and thus there is less opportunity for timing errors to occur.

The Q-E behavior of the reduced-width adder is shown in Figure 2(b). We are specifically interested in finding the Q-E behavior as a function of the bitwidth. Note that because no *timing* errors are allowed in Adder 2, an exploration with respect to timing budget, as shown for Adder 1 above, would have no purpose. We see that for large bitwidths of Adder 2, there is no quality loss. A significant reduction in quality occurs with the onset of overflow errors when the magnitude of data being processed is larger than the available adder width.

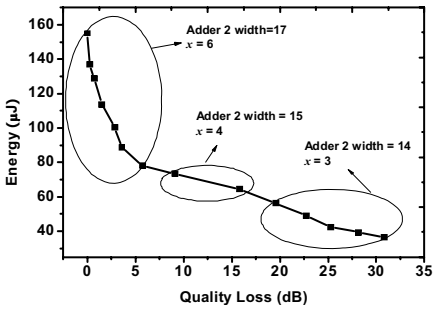


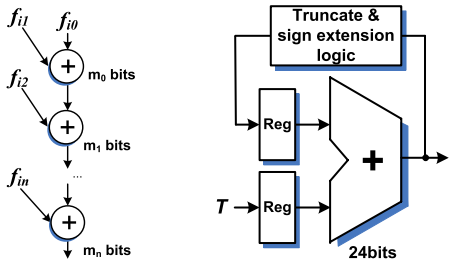
Fig. 3. Energy vs. quality loss Pareto front.

The analysis of the system Q-E behavior combines the behavior of Adder 1 and Adder 2. This enables exploration of the x, D_2, W_2, and T_1 design space in order to find an optimal Q-E solution. The primary trade-off involves the choice of x. From Figure 2(c), we can see that the total quality loss reaches a minimum when x is around 4. For larger values, the quality loss due to Adder 2 becomes excessive. For smaller values, the quality loss is dominated by errors from Adder 1. However, the optimal choice of x also depends on both the total timing budget available as well as the bit-width of Adder 2. The set of optimal design decisions is best represented as a Pareto curve in the energy-quality space as shown in Figure 3. The figure shows the Pareto points, i.e. $min(Q|E)$, that are generated by different choices of x and W_2 at different T_1.

In the implementation, the reduced-width addition is realized using the truncated result of a regular-width adder sharing the same core logic. The combined adder architecture is shown in Figure 4. The indexes of the frequency coefficients are used by the control logic to determine whether to feed them into a full-width or reduced-width addition. The control logic compares the index of the matrix component currently being processed with the predetermined classification constant x. The output of this comparison is used to activate a truncation logic. The truncation logic takes a reduced number of LSBs from the full-width adder output according to the predesigned Adder 2 width, and sign extends them back to the full width and feeds the result back into the destination accumulator.

B. Error control by dynamic reordering of accumulations

The technique introduced in Section II-A is able to delay the onset of large-magnitude errors in individual two-operand additions. The second technique presented in this section is based on reduction of the cumulative quality loss resulting



(a) Technique abstraction (b) Implementation

Fig. 4. Reduced width adder.

from multiple additions, such as accumulations of IDCT. The key observation is that if positive and negative operands are accumulated separately, the number of error-producing operations is reduced to one last addition that involves operands with opposite sign. At the same time, the operands involved in this last addition are guaranteed to be larger in absolute value than any individual opposite-sign operands involved in the original sequence. This guarantees that the reordered accumulation will result in a smaller quality loss under scaled timing.

The difference between optimized and un-optimized sequences is significant. As an example, consider four numbers (-1, 1, -1, 1) being accumulated. There are three possible sequences of accumulation:

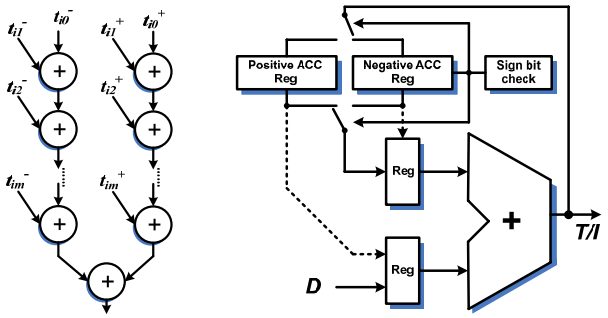
Case 1: 11111111+00000001+11111111+00000001
Case 2: 11111111+11111111+00000001+00000001
Case 3: (11111111+11111111)+(00000001+00000001)

For Case 1, the 1st and the 3rd additions have large delay, each with a carry chain length of 8. For Case 2, the 3rd addition has large delay with a carry chain of 8. For Case 3, only the addition outside the brackets has large delay with a carry length of 7. The total timing budget in Case 3 is roughly half of that of Case 1. Thus, we observe that the order of accumulation can significantly affect the frequency of worst-case delay as well as the length of the longest carry chain.

We now show how the sequence of additions can be changed to reduce overall error. As described above, we first group operands with the same sign. Then, the operands in each group are accumulated and finally the results of two group-accumulations are added. This is akin to the strategy that Case 3 illustrates. Because the best grouping of operands cannot be known at design time, this technique is dynamic and is based on execution-time observation of operand values.

The proposed implementation uses the sign bits in the MSB to separate the positive and negative operands when loading data. The implementation is shown in Figure 5. The control logic checks the sign bits and accumulates positive and negative numbers in separate accumulation registers. Then, in a final step, the results are added together. This final addition can in turn be protected against timing errors using either one of the techniques presented in Section II-A or II-C.

Compared to the original implementation, the reordered accumulation carries extra overhead for the reordering logic and duplicate accumulation registers. Nevertheless, experiments show (Section III) that the technique can significantly improve the quality-energy profile under scaled timing.



(a) Technique abstraction (b) Implementation

Fig. 5. Accumulation reordering architecture.

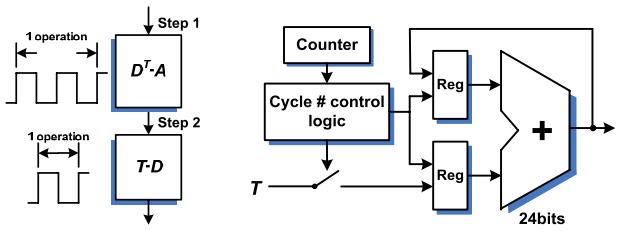
C. Preventing error spread and amplification

In previous sections, we presented techniques for targeting individual error sources at the operation- and block level. With knowledge of the application, we now further focus on control of sources of errors that have the potential to be spread and amplified at the algorithm level. More specifically, we propose a technique using algorithm-level retiming to explicitly prevent errors in critical steps that may have a large impact on downstream results and hence overall quality.

For the 2D-IDCT algorithm, analysis of control and data flow is relatively simple because it consists of two nearly-identical steps: (1) $T = C^T \cdot A$ and (2) $I = T \cdot C$. We address the problem of a timing error in Step 1. Such an error can generate multiple output errors in I because each element of T is used in multiple computations of Step 2. We can model this behavior by introducing an error matrix E, which is added to T such that the two algorithm steps become: (1) $T^{'} = T + E$ and (2) $I = T \cdot C + E^{'}$. Here, $E^{'} = E \cdot C$ is the final error. Although E may have only one non-zero entry, the matrix product results in up to $size(A)$ errors vertically or horizontally in $E^{'}$. As a result, the noise in the decoded image of an unmodified IDCT has a stripe pattern (see Figure 9 in Section III).

Thus, to avoid such wide-spread quality loss, we need to ensure that no errors occur in Step 1. We assume an architecture in which supply voltage can only be scaled uniformly. If timing budgets are allocated to steps based on worst-case analysis, any reduction in V_{DD} would lead to a reduced timing slack in Step 1 and hence un-allowable levels of errors being generated there. We therefore propose a strategy to allocate extra timing margins to critical steps, such as Step 1. Importantly, given overall latency constraints for the design, as is the case for many real-time image or video coding applications, end-to-end algorithm timing must remain constant and performance must not be degraded. Thus, an important element of protecting the early algorithm steps is a re-allocation strategy that shifts timing budgets between steps. Maintaining a constant total time, we show how to borrow computing time from non-critical algorithm steps in order to increase timing margins in critical ones, all while reducing overall quality loss.

To implement such a strategy, we make the timing budget in each step adjustable. The original minimum error-free timing budget for each step is $T_{step1} = N_1 \times T_{clk}$ and $T_{step2} = N_2 \times T_{clk}$, where T_{clk} is the clock period, and N_1 and



(a) Technique abstraction (b) Implementation

Fig. 6. Rescheduling of algorithm steps.

N_2 are the number of cycles in each step. In the original 2D-IDCT implementation, steps are identical and $N_1 = N_2 = N$. To adjust the budget, we need to divide it into multiple parts. A division factor M is used to make $T_{step1} = NM \times T_{clk}/M$, and $T_{step2} = N \times T_{clk}/M$. V_{DD} is then scaled down, increasing the propagation delays. Consequently, T_{clk} is scaled to $T_{clk}^{'}$ such that $2N \times T_{clk}$ is equal to $NM \times T_{clk}^{'}/M + N \times T_{clk}^{'}/M$, i.e. $T_{clk}^{'} = 2T_{clk}/(1+1/M)$. Hence, the new clock frequency is $f_{clk}^{'} = T_{clk}^{'}/M = 2/((M+1)T_{clk})$. Since the total budget is fixed, we disproportionally shift timing budgets under scaled V_{DD} from Step 2 to Step 1. Note, however, that the factor M cannot become too large. Otherwise, the clock frequency would be too high and timing errors would not remain restricted to the adder in Step 2.

The implementation includes logic to allocate different timing budgets to each step (Figure 6). We empirically choose M to be 2 and increase clock frequency accordingly. The control logic includes a 1-bit counter to keep track of the cycle counts for each step. In Step 1, each operation is assigned 2 cycles, while each operation in Step 2 is assigned 1 cycle.

III. Experimental Results

The architecture of our final 2D-IDCT implementation is a folded one [10], where each 1D-IDCT shares the same pipelined arithmetic unit containing an adder and a multiplier. The IDCT data and coefficient matrices A and C have 16-bit and 8-bit resolution, respectively. The multiplier is pipelined and has a width of 8×16 bits. The adder is a ripple-carry adder with a width of 24 bits. Such design restricts entirely the timing errors to adder for acceptable quality loss. Only the Y signal of a Y:Cb:Cr format image is used.

The 2D-IDCT is implemented in Verilog-HDL and synthesized using Design Complier with the OSU 45nm PDK. To enable our experiments, we construct an explicit model of the critical path delays at different V_{DD} values. Since Design Compiler and the PDK only report power at a single V_{DD} value, we are not able to run synthesis at different voltage levels. Instead, we use HSPICE to re-characterize each gate in the cell library and generate the power data of the synthesized design for other V_{DD} values. Design Compiler reports the critical path for V_{DD} = 1.1V. We obtain the delay difference between the nominal case at V_{DD} = 1.1V and delays at different V_{DD} values via the HSPICE simulations. Then, the critical path delay at an arbitrary V_{DD} can be computed as:

$$D_{critical} = \sum_{i=1}^{n} h_i(\Delta V_{DD}) + D(V_{DD} = 1.1V),$$

TABLE I
Energy Saving and Area

	V_{DD}	Energy Saving	Area μm^2
Original	1.0	0%	149948.39
Adder	0.81	35.3%	150482.45
Reorder	0.81	34.6%	154611.35
Step1&2	0.80	35.1%	150073.69
All three	0.75	45.2%	155175.45

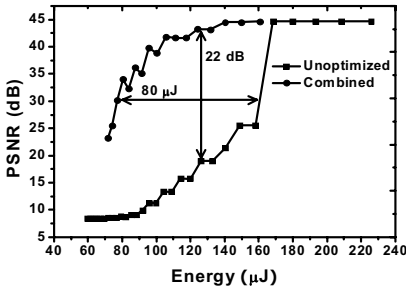


Fig. 8. Combined PSNR vs. energy profile.

where $h_i(\Delta V_{DD})$ is a fitted delay model of a single gate:

$$h(\Delta V_{DD}) = c_2 \cdot \Delta V_{DD}^2 + c_1 \cdot \Delta V_{DD}.$$

We also derive a model to estimate power at different voltage levels. Design Complier reports dynamic power and leakage power at V_{DD} = 1.1V. The dynamic power at other V_{DD} can be estimated as follows:

$$P_{dyn} = \sum_{i=1}^{m} W_i \cdot N_i \cdot f_i(\Delta V_{DD}) + P_{dyn}(V_{DD} = 1.1V),$$

where N_i is the number of min-sized gates of type i, W_i is the total size for a gate of type i, and $f_i(\Delta V_{DD})$ is the quadratic fitted model for the dynamic power component. Similarly, the leakage power can be computed at arbitrary V_{DD} as follows:

$$P_{leak} = \sum_{i=1}^{m} W_i \cdot N_i \cdot g_i(\Delta V_{DD}) + P_{leak}(V_{DD} = 1.1V),$$

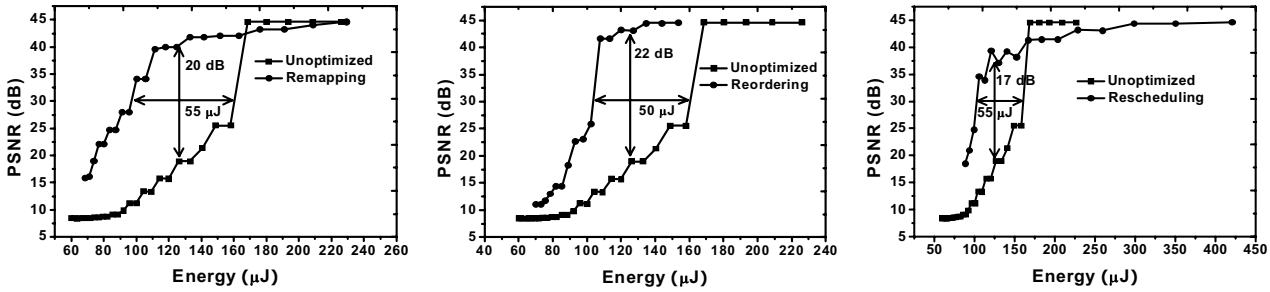
where N_i is the number of min-sized gates of type i, W_i is the total size for a gate of type i, and $g_i(\Delta V_{DD})$ is the fitted model of the leakage power component. Power values under different V_{DD} are estimated based on the fitted models above, and the corresponding energy values are computed as period times power, where the period is 11ms in our case.

Table I shows the energy savings for each technique and their combination. Energy savings are computed at PSNR = 30dB with the processing rate being a constant 11ms per 256×256 frame. The resulting PSNR vs. energy profiles for each technique are shown in Figure 7.

Individual techniques can be combined to achieve maximum energy savings. However, since the described techniques all have varying impact on the different frequency components, their optimal combination is not obvious. Using the technique of Section II-C, a larger timing budget is given to the earlier algorithm step. This change impacts all frequency components. On the other hand, the technique of Section II-A impacts

TABLE II
Energy under different combinations.

Component	0	1	2	3	4	5
Eng(μJ)	159	123	92	96	99	102



(a) Reduced-width adder (b) Accumulation reordering (c) Re-budgeting of two cycles to step 1

Fig. 7. Individual PSNR vs. energy profiles.

mainly the high-frequency components (since they are the components that involve small-valued operands). Finally, the technique of Section II-B impacts operands with opposing sign, no matter if they are low- or high-frequency components.

Based on these observations, we devised the following strategy for selectively applying techniques to different algorithm steps and frequency components: (1) In Step 1, we allocate more cycles only to the low-frequency components while using dynamic reordering and a reduced-width adder to process the high-frequency components; (2) In Step 2, timing errors are not propagated into later steps, so only the reduced-width adder and dynamic reordering are applied. In this combination, the total number of clock cycles needed in Step 1 is smaller than what the technique introduced in Section II-C would require to achieve the same quality level. Hence, under a fixed total time, the adjusted clock period $T_{clk}^{'}$ is larger and there exists more timing slack for energy savings.

The key problem is to determine which low-frequency components in Step 1 require more cycles for their processing after applying techniques from Sections II-A and II-B. Since the size of the frequency coefficient matrix in a 2D-IDCT is small, we can do a brute-force exploration to determine the best assignment. Table II shows the results of such simulations. Results indicate that the smallest energy is obtained when allocating more time (two cycles in our implementation) to the computation of the first two low-frequency components.

The PSNR vs. energy curve for the combination of techniques is shown in Figure 8. A significantly improved trade-off curve is generated by a non-trivial combination of individual techniques. Finally, a set of sample images under scaled V_{DD} is shown in Figure 9. Note that achieving a similar energy reduction by conventional V_{DD} scaling would result in unacceptable degradation of image quality (Figure 9(b)).

IV. Conclusions

This paper presented techniques that enable architecture-level shaping of the quality-energy tradeoff under aggressively scaled V_{DD} through controlled timing error acceptance. We demonstrated the implementation of these techniques on a design of a 2D-IDCT architecture. Results show that significant energy savings can be achieved while maintaining a constant performance and good image PSNR. To further improve the visual quality, error limiting technique can be implemented to reduce image artifacts.

(a) Original: Energy=168μJ PSNR=44.6dB (b) Original: Energy=94μJ PSNR=11.2dB

(c) Proposed: Energy=104μJ PSNR=38.8dB (d) Proposed: Energy=92μJ PSNR=32.3dB

Fig. 9. Image quality under different energy budgets.

References

[1] S. H. Nawab, A. V. Oppenheim, A. P. Chandrakasan, J. M. Winograd, and J. T. Ludwig, "Approximate signal processing," *VLSI Signal Processing*, vol. 15, pp. 177–200, 1997.

[2] J. T. Ludwig, S. H. Nawab, and A. P. Chandrakasan, "Low-power digital filtering using approximate processing," *JSSC*, pp. 395–400, 1996.

[3] A. Sinha and A. P. Chandraksan, "Energy efficient filtering using adaptive precision and variable voltage," *ASIC SOC Conference*, pp. 327–331, 1999.

[4] R. Hedge and N. R. Shanbhag, "Soft digital signal processing," *TVLSIS*, pp. 379–391, 2000.

[5] L. Wang and N. R. Shanbhag, "Low-power filtering via adaptive error-cancellation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 51, no. 2, pp. 575–583, 2003.

[6] J. Park, S. Kwon, and K. Roy, "Low power reconfigurable dct design based on sharing multiplication," *ICASSP*, pp. III–3116–III–3119, 2002.

[7] G. Karakonstantis, D. Mohapatra, and K. Roy, "System level dsp synthesis using voltage overscaling, unequal error protection and adaptive quality tuning," *SIPS*, 2009.

[8] N. Banerjee, G. Karakonstantis, and K. Roy, "Process variation tolerant low power dct artchitecture," *DATE*, pp. 1–6, 2007.

[9] E. Y. Lam and J. W. Goodman, "A mathematical analysis of the dct coefficient distribution for images," *IEEE transaction on image processing*, vol. 9, no. 10, pp. 1661–1666, 2000.

[10] S. Uramoto, Y. Inoue, A. Takabatake, J. Takeda, and Y. Yamashita, "A 100-mhz 2-d discrete cosine transform core processor," *JSSC*, vol. 27, pp. 492–499, 1992.