Low-Energy Digital Filter Design Based on Controlled Timing Error Acceptance

Ku He, Andreas Gerstlauer and Michael Orshansky
Electrical and Computer Engineering
The University of Texas at Austin, Austin, TX 78712, USA.
E-mail: {kuhe, gerstl, orshansky}@utexas.edu

Abstract— In signal processing applications, large energy gains can be obtained by accepting some degradation in the output signal quality. Filters are at the core of many such systems. In this paper, we demonstrate the potential of a new paradigm for achieving favorable quality-energy trade-offs in digital filter design that is based on directly accepting timing errors in the datapath under aggressively scaled V_{DD}. In an unmodified design, such scaling leads to rapid onset of timing errors and, consequently, quality loss. In a modified filter implementation, the onset of large errors is delayed, permitting significant energy reduction while maintaining high quality. Specifically, the innovations in the design include techniques for: 1) run-time adjustment of datapath bitwidth, and 2) design-time reordering of filter taps. We tested the new design strategy on several audio and image processing applications. The designs were synthesized using a 45nm standard cell library. Results of SPICE simulations on the entire designs show that up to 70% energy savings can be achieved while maintaining excellent perceived signal-to-noise ratios (SNRs). Compared to a traditional filter design, the area overhead of our architecture is about 2%.

Keywords—Digital filters, Error Tolerant Design, Approximate Computing, Low Power

I. Introduction

Multimedia tasks such as speech, image, and video processing are often responsible for much of the energy consumption in portable electronic devices. Extending battery time requires continued innovation in low-power methods for such systems. In this paper, we propose techniques based on timing error tolerance to significantly reduce energy consumption in digital filter circuits, which are an important building block of many such applications.

Because of their importance, much work has been done in the area of low-power implementation of digital filtering circuits over the previous decades. In general, finite-impulse response (FIR) filters tend to be more power-consuming than infinite-impulse response (IIR) ones [1]. A very incomplete list of approaches to reduce filter power at the architectural level includes techniques such as multirate filtering, subfilter approaches and multiplierless architectures. At the circuit level, optimal selections of filter bitwidths and realizations of adders and multipliers to reduce power consumption have been done either in a static [2] or dynamic [3] fashion. Furthermore, optimally choosing filter parameters for given target metrics such as gain, phase linearity, bandwidth, pass-band ripple or stop-band attenuation for low power has also been investigated [4].

It is widely recognized that voltage scaling is one of the most effective ways to reduce power consumption of any digital system. In [5], this is exploited by implementing the filter using fastest possible filter structures and then using generated timing slack to reduce power via voltage scaling. In the traditional paradigm, V_{DD} scaling is limited by the worst-case delay through any combinational logic. In other words, a conventional methodology guarantees timing correctness of all operations by construction. Scaling of V_{DD} beyond the point of worst-case delays immediately leads to large timing errors and rapidly degrades the output signal quality. This rapid quality drop eliminates the possibility of an efficient tradeoff between quality and energy. In this paper, we describe techniques that allow pushing V_{DD} beyond this point to achieve further energy savings. We show how to achieve a graceful degradation of filter quality by identifying the sources of early and worst timing errors and designing filtering architectures to eliminate such errors.

In developing this approach, we work in the wider framework of error-tolerant low-power design. In previous work, techniques for trading quality for energy in digital filtering and digital signal processing (DSP) applications have been studied at varying levels of abstraction. In [6], the authors propose a technique that dynamically minimizes the order of a digital filter to reduce the switched capacitance and hence the total energy. In [7], the filter computations are restructured such that voltage scaling affects less important filter taps first. Finally, in [8], [9], energy is saved by using lower voltage on the main computing block and running a simplified estimating or error cancelling block at higher voltages to correct timing errors in the main block.

The common feature of prior work is that results produced by blocks subject to timing errors are not directly accepted. By contrast, our strategy allows using the erroneous results directly, provided, of course, that the frequency and magnitude of errors are carefully controlled. The concept of controlled timing error acceptance for low energy DSP applications has been proposed in [10], [11], where a significantly improved quality-energy tradeoff was demonstrated for image processing applications. In this paper, we adapt this approach to develop a general architecture and design strategy for low-power, timing error accepting digital FIR and IIR filters with applications in a wide variety of DSP systems. We evaluate our optimized design method on a range of audio and image filters for which up to 70% energy savings can be achieved with an area overhead of less than 2.1% compared to a standard filter implementation.

The rest of the paper is organized as follows: Section II introduces our techniques for timing error control, Section III presents experiments and results, and Section IV concludes the paper with a summary and outlook on future work.

II. Timing Error Control

This work focuses on developing modifications for generic digital filter implementations to allow them to tolerate timing errors. We base our work on an industry-standard low-overhead, low-power design in which the core multiply-accumulate (MAC) operations are realized using a multiplier and adder that are chained to operate in one clock cycle. This single MAC architecture is common in state-of-the-art general-purpose and digital signal processors. We choose this architecture to demonstrate the applicability of our techniques to widely used DSP implementations. In such an architecture, the critical path is defined by the multiplier-adder chain, where, under timing starvation, the addition at the end of the chain will experience timing errors first. Note that other components in the architecture are not on the critical path and can be treated as timing-error free for the amount of slack and range of voltage scaling considered in our experiments. As such, we focus techniques in this paper on controlling timing errors in the adder.

A. *Error control through bitwidth adjustment*

We first present a technique that exploits the properties of operand statistics to achieve energy savings. When V_{DD} begins to scale down, timing errors impact the results of computation as data moves through the datapath. Timing errors impacting the highly significant bits cause the largest signal quality degradation. Therefore, the objective of our techniques is to prevent such errors. It has been demonstrated before that the early-onset MSB errors are caused largely by processing small opposing-sign additions [10]. This is because in 2's complement code, addition of small opposing-sign operands leads to the longest carry propagation chains and hence worst-case delays into the MSB. Since the actual operands are small, an early termination of the carry propagation results in large errors.

In [10] a method to dramatically reduce the incidence and impact of such timing errors has been introduced. The idea is to statically apply a reduced-width adder for small operands to reduce the length of the longest carry. The necessary condition, of course, is that the bitwidth of a reduced-width adder is large enough to represent the accurate result and avoid overflow. In filters, the input data and coefficient bitwidths are determined by the dynamic range specification, while the bitwidth of the datapath is determined by the gain of the transfer function. In [11], the concept of allowing a dynamic reduction in the bitwidth of adders in an image filter was introduced. In the following, we expand on this idea to develop a general design and optimization technique for arbitrary filter applications.

We exploit the fact that common applications of digital filters operate on data characterized by distributions of a specific type. It is well-known, for example, that speech and music data usually follow a Laplacian distribution [12] However, our techniques are not limited to Laplacian data. We also demonstrate their effectiveness for other distributions, such as pixel data in Section III-C. A key property of typical data distributions is that most values are smaller than the maximum and often even close to zero. Nevertheless, in a traditional design paradigm that does not allow timing errors, the datapath bitwidth has to be designed to be able to process the largest possible inputs, which in fact occur very rarely.

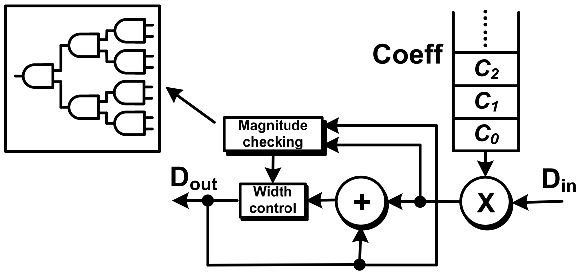
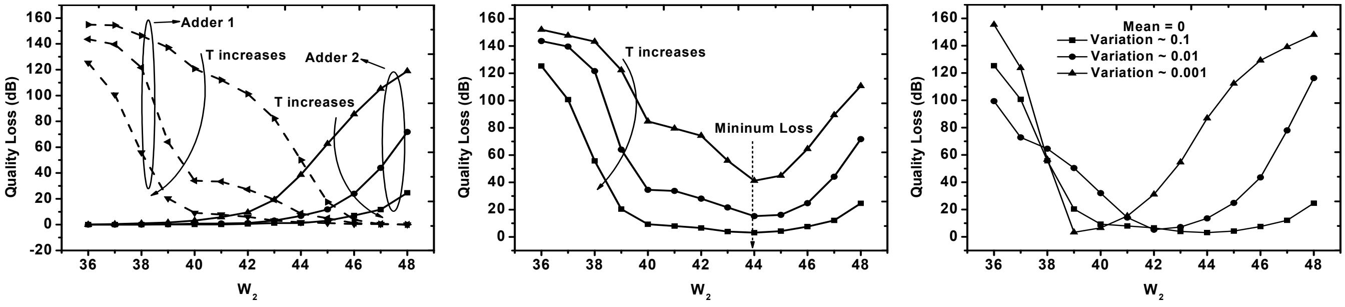


Fig. 1. Dynamic-width adder architecture.

1) Dynamic bitwidth adjustment. The proposed architecture of a filter that uses a dynamic-width adder is shown in Figure 1. The idea is to adjust adder bitwidth dynamically, with the purpose of eliminating the early and large timing errors. The architecture requires checking the magnitude of the input data and processing the operands on an adder with bitwidth sufficient for the particular inputs. To allow the results to be used in downstream computations, we further perform sign extension to the full bitwidth. An actual implementation does not require the bitwidth to be continuously adjustable: according to our experiments, just two bitwidth values are sufficient to enable a significant quality-energy tradeoff.

Operand magnitude-checking logic is activated on each addition. It determines whether the MSBs in both operands are either all 1s or all 0s. If so, the bitwidths of both operands are less than or equal to the reduced width, and can be processed by a reduced-width adder. Otherwise the operands are processed by an adder of regular width. It is assumed that only one physical adder is used. The inputs are first sent to the magnitude-checking logic block, which can be implemented compactly. The checking logic uses *AND* gates to determine whether a specified number of higher-significance input bits are all zeros or all ones. If this condition is true, a width-control logic is activated to perform truncation and sign-extension on the adder output. In essence, each time the magnitude-checking logic initiates a reduced-width addition, a smaller effective configuration of an otherwise full-width adder is used.

The overhead of implementing the described technique includes delay, energy and area costs. The magnitude-checking block operates on 5 to 15 bits with a maximum delay of about $log_2(15)$ gate equivalents. Since it runs in parallel to and is faster than the MAC unit, no overall delay overhead is incurred. The truncation and sign extension logic adds a multiplexer with one gate delay on the critical path. Note that the critical path is defined when operating in full-bitwidth mode, where sign extension is disabled and the multiplexer is in pass-through configuration. For the range of voltage scaling considered, truncation and sign extension logic, which is active only in reduced-width mode, is free of timing errors and otherwise only contributes to the load on the adder. Overall energy and area overhead includes the magnitude-checking block, MUX gates and sign extension



(a) Adder quality loss vs. Adder 2 bitwidth

(b) Total quality loss vs. Adder 2 bitwidth

(c) Optimal Adder 2 width

Fig. 2. Quality loss dependence on Adder 2 bitwidth.

logic, which we quantify in the experimental section. Energy overhead also comes from the cost of switching between a full-width and a reduced-width adder, which needs to occur on a per-sample basis. The frequency of switchings depends on the statistics of input operands. Experiments show that the incurred energy overhead is, in the end, justified because the entire technique enables significantly higher energy savings through an increased potential for voltage scaling.

2) Bitwidth granularity. Because the technique only supports a discrete number of allowed bitwidths, we need to address the question of how to find the optimal bitwidth for the smaller adders. We investigate this question on a practical digital filter design with two bitwidths, which implements a 5th order FIR filter, as further detailed in Section III. We define Adder 1 as the full-width adder and Adder 2 as the reduced-width adder. The widths of Adder 1 and Adder 2 are W_1 and W_2, respectively. In the following discussion, the effective W_1 is set to be 50. Formally, the goal is to find the W_2 which leads to the least quality loss at a given energy budget. We define the following parameters: D_1: the worst-case delay of Adder 1; D_2: the worst-case delay of Adder 2; T: the timing budget of Adder 1 and Adder 2. We assume in this discussion that $T < D_2 \leq D_1$, i.e. timing errors may occur in both Adder 1 and 2. In this following discussion, we assume that the input data is centered around zero (i.e., has a zero mean).

Through simulation we obtain quality-energy profiles for both adders both individually (Figure 2(a)) and jointly (Figure 2(b)). From Figure 2(a) we see that as W_2 increases, quality loss in Adder 2 increases while quality loss in Adder 1 decreases. As a result, we see in Figure 2(b) that there exists a width W_{opt} that results in a minimal total quality loss. The optimal width appears to be largely insensitive to the allotted timing budget. When Adder 2 width is greater than W_{opt}, the overall quality loss grows. This is because error magnitude is a function of adder width, such that errors in Adder 2 become larger and dominate. Conversely, if Adder 2 width is below W_{opt}, a larger fraction of the input data is processed by Adder 1. The result is that more small operands are processed by the full-width adder leading to more frequent and larger errors.

So far, our discussion assumed that we realize a dynamic-width adder with only two possible bitwidths. In principle, it is possible to have a larger number of bitwidths available. We find, however, that increasing the number of bitwidths does not substantially improve the quality-energy tradeoff. In our experiments, compared to a minimum quality loss between 3dB and 15dB in the two-adder case (Figure 2(b)), losses for three or four bitwidths are both in the range of 1dB-7dB.

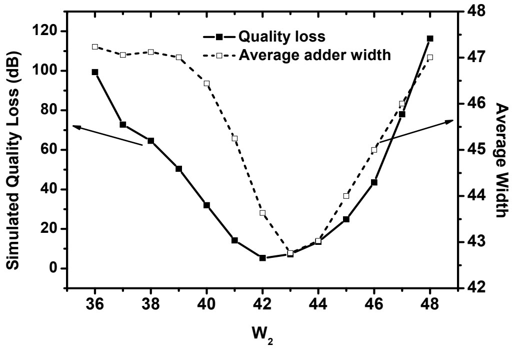


Fig. 3. Quality loss and average adder width.

3) Input-dependent bitwidth optimization. We further investigate the dependence of optimal dynamic bitwidth parameters on operand statistics and, specifically, on the variance of the operand distribution. Quality losses at different W_2 are shown in Figure 2(c) for several values of operand variance. We observe that the optimal Adder 2 width changes notably as the variance of the input data changes. For smaller variance, a larger fraction of data has values that are small, and it is advantageous to make the width of Adder 2 (W_2) smaller.

To better understand the dependence of optimal dynamic-adder widths on input data statistics, we develop an analytical model that allows an estimation of optimal design parameters. Having such a model also removes the need to rely on time-consuming simulation-based analysis. Recall that the maximum error in each adder is proportional to the width of that adder. Thus, the model is based on the intuition that the bitwidth of Adder 2 (W_2) that minimizes quality loss at any given level of variance also minimizes the *average effective adder width.* We define average effective adder width as $W_{avg} = W_1 \times p_1 + W_2 \times p_2$, where p_1 and p_2 are the probabilities of using Adder 1 or Adder 2 respectively. As Figure 3 shows, we observe that quality loss and W_{avg} track well as W_2 is swept and that their minima coincide to a good degree.

Relying on the tracking of the two metrics as established above, we formulate the search for the optimum W_2 as a minimization problem for W_{avg}. The model assumes that input data

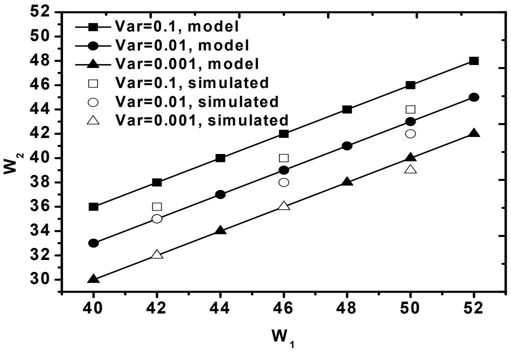


Fig. 4. Predicted and optimal W_2.

distributions are given by Laplacians. We consider the case with only two bitwidths. Let d_1 and d_2 be the two input operands to the adder, and x be the magnitude threshold for determining whether a larger (Adder 1) or smaller (Adder 2) adder is used for processing the operands. Then, the problem of minimizing W_{avg} is given as:

$$\begin{aligned}\min_{\mathrm{x}}: \quad & P(|d_1| < x, |d_2| < x) \cdot W_2 \\ + \quad & [1 - P(|d_1| < x, |d_2| < x)] \cdot W_1\end{aligned}$$

We further make the simplifying assumption that the operands are independent, which allows us to re-write the minimization problem as:

$$\begin{aligned}\min_{\mathrm{x}}: \quad & P(|d_1| < x)P(|d_2| < x) \cdot W_2 \\ + \quad & [1 - P(|d_1| < x)P(|d_2| < x)] \cdot W_1\end{aligned}$$

The probabilities in the above expression can be evaluated under the assumption that the inputs follow a Laplacian distribution. We further assume that distributions are zero-centered, i.e., μ=0, which is true of many practical instances. The probability density function is given by:

$$f(t|\mu, b) = \frac{1}{2b} e^{-\frac{|t-\mu|}{b}},$$

where b is the scale parameter related to variance as $\sigma^2 = 2b^2$. The sought probability can be computed by:

$$P(|d_1| < x) = (\int_{-x}^{x} \frac{1}{2b} e^{-\frac{|t|}{b}} dt)$$

Substituting this probability into the minimization problem:

$$(\int_{-x}^{x} \frac{1}{2b} e^{-\frac{|t|}{b}} dt)^2 log_2 x + [1 - (\int_{-x}^{x} \frac{1}{2b} e^{-\frac{|t|}{b}} dt)^2] W_1$$

Finally, minimizing this function can be reduced to a form:

$$W_1 - (W_1 - log_2 x)(1 - e^{-\frac{x}{b}})^2 \tag{1}$$

The minimum of function (1) can be computed by setting its 1st derivative to zero and solving the equation. The resulting equation is:

$$\frac{1}{ln2 \cdot x}(1 - e^{-\frac{x}{b}})^2 + \frac{2}{b} e^{-\frac{x}{b}} (W_1 - log_2 x)(1 - e^{-\frac{x}{b}}) = 0 \tag{2}$$

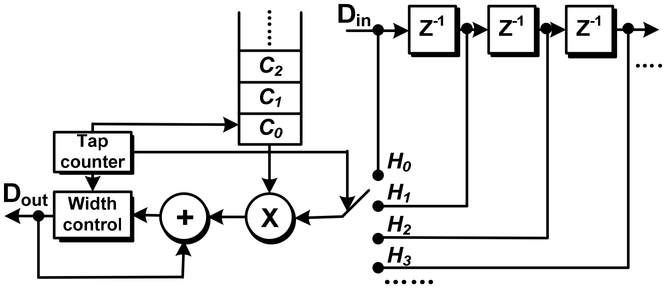


Fig. 5. Technique abstraction of reordering.

We find that the model provides good matching to the simulation-based exploration. For example, optimal simulation-based values of Adder 2 bitwidth that minimize quality loss are 44, 42, and 39 for three levels of input data variance. For the same values of variance, the analytical model described above predicts optimal W_2 to be 46, 43 and 40.

To further test the effectiveness of the model, we compare the optimal W_2 obtained from simulation and predicted by the model under different W_1. Results are plotted in Figure 4. Overall, we can see that our model and simulated results match well at inputs with low variance, which are typical in music and speech (empirically determined to be < 0.01 in our experiments, while the data range is -0.5 to 0.5).

B. Error control through reordering

In the previous Section, we discussed how the onset of large timing errors can be controlled by using a reduced-width adder for small operands. Since errors for large operands are unavoidable, such an approach is most effective if the relative fraction of small operand additions is increased. In this section, we introduce techniques to manipulate the input data distribution of intermediate MAC operations by reordering of filter taps.

In a traditional single MAC unit design, the width of the MAC unit is determined by the maximal bitwidth over all taps, which is generally independent of any intermediate reordering. However, under a timing error acceptance philosophy, such techniques will allow us to statically apply adders of different width to different taps in order to reduce timing errors. Furthermore, in combination with dynamic bitwidth adjustment (Section II-A), reordering can, on average, reduce the magnitude of data in intermediate operations and hence increase the effectiveness of this technique.

The digital filter can be formulated as: $y(n) = \sum_{i=0}^{N} b_i \cdot x(n-i) + \sum_{i=1}^{N} a_i \cdot y(n-i)$. In computing the final output, a filter needs to generate a set of intermediate results, which correspond to a set of intermediate transfer functions H_0-H_{2N}. These transfer functions determine the maximum possible gain over all frequencies at intermediate nodes and, hence, the minimum required bitwidth for each intermediate result in the datapath. Without affecting the output of the filter, intermediate transfer functions vary when the order of filter taps is changed. As such, we can reduce intermediate gains and hence the bitwidth of intermediate operations by optimally reordering the taps. Such a reordering can be done at design time. It allows us to apply an

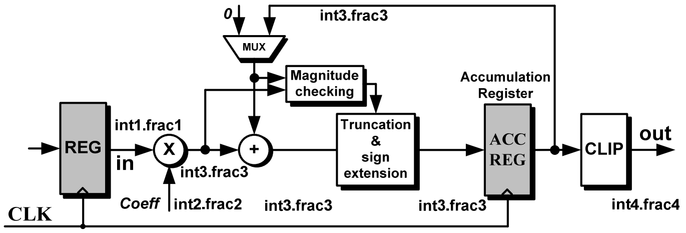


Fig. 6. Single MAC filtering architecture.

adder of smaller width to intermediate taps in order to reduce the timing errors under voltage scaling.

To optimize the order of filter taps, we discuss the cases for FIR and IIR filters separately. For a simpler FIR filter, all a_i paths are removed and the maximum gains at intermediate nodes are determined from the transfer functions as: $G_i = \sum_{j=0}^{i} |b_j|$. Therefore, minimizing gains is achieved by processing of filter taps in ascending order of absolute filter coefficient values (b_0, b_1, ..., b_N).

For IIR filters, obtaining the optimized order of filter taps requires exhaustively searching for all possible orders, which is extremely inefficient. In practice, we divide the IIR filter coefficients into feedforward and feedback sections, where the sets (b_0, b_1, ..., b_N) and (a_1, a_2, ..., a_N) represent the coefficients of the denominator and numerator of the filter transfer function, respectively. We reorder the two coefficient sections separately based on the aforementioned FIR filter optimization method.

The implementation of the reordering technique involves changing the order of arithmetic operations and applying a smaller adder to each tap depending on intermediate filter gains G_i and maximum input data range. For a single MAC architecture, the abstraction of this technique is shown in Figure 5. In the implementation, the tap control logic changes the order in which data and coefficient pairs are fed into the MAC unit. Furthermore, it also truncates and sign-extends results for taps whose gains are small.

Note, however, that reordering can not achieve a smaller gain for intermediate transfer functions in all cases. For example, if taps are already optimally ordered or if all coefficients are the same (as is the case in FIR filters using rectangle windows), no further optimizations are possible, but reduced adder widths may still be applicable.

III. Experiments and Results

We base our implementation of a digital filter architecture with timing error control on a single MAC unit design as shown in Figure 6. Data is processed using a fixed-point format. Coefficients are represented in Q$int2.frac2$ form, i.e. with $int2$ integer and $frac2$ fractional bits. Similarly, input, output and intermediate data is in a Q$int1.frac1$, Q$int4.frac4$ and Q$int3.frac3$ format, respectively. By changing coefficients and data precisions, such a generic architecture can be used in different filtering applications.

We have applied our approach to several filtering examples in audio, speech and image processing. For audio and speech applications, we designed both FIR and IIR filters. For image processing applications, we designed a 2-D sharpening filter based on an FIR architecture. Filter parameters and coefficients were selected and first simulated using Matlab's signal and image processing toolboxes (using FDATool and the $fspecial$ function, respectively).

TABLE I

Energy and area overhead of various techniques.

	Energy Overhead			Area Overhead		
	Width	Reorder	Comb.	Width	Reorder	Comb.
FIR	0.5%	0.7%	0.9%	1.6%	1.1%	1.7%
IIR	0.5%	0.3%	0.7%	1.1%	1.0%	1.2%
Sharpen	0.6%	0.4%	0.9%	1.4%	1.1%	2.1%

TABLE II

Energy savings and performance.

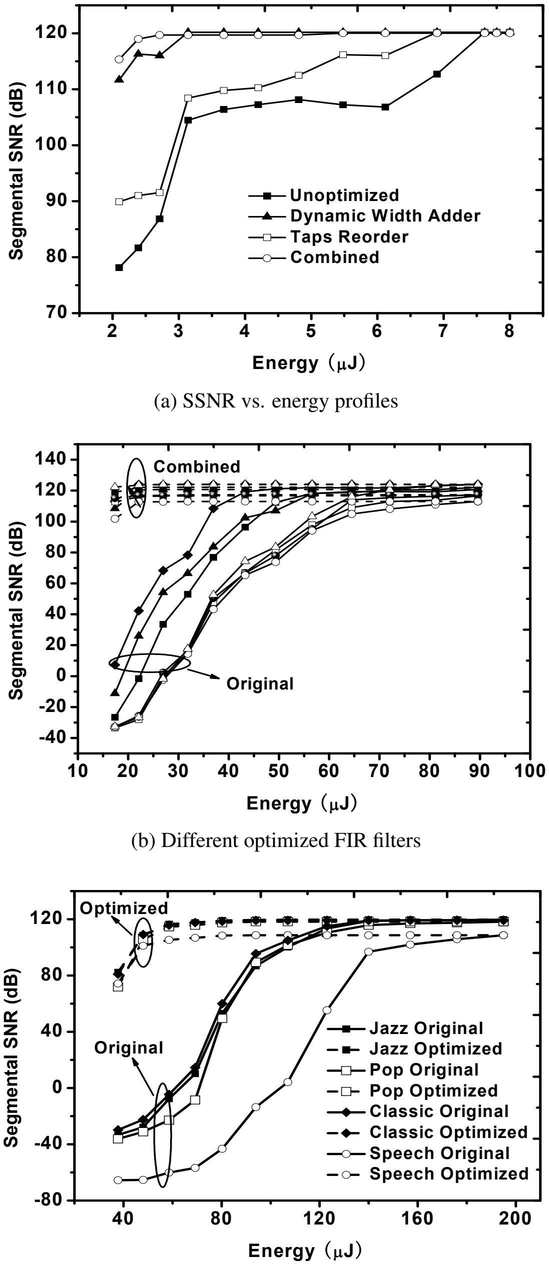
	V_{DD}	SSNR/PSNR	Eng. Saving	Speed
FIR	0.75V	119.7dB	58.8%	207.5MHz
IIR	0.75V	121.7dB	58.1%	207.9MHz
Sharpen	0.70V	23.3dB	69.7%	303.0MHz

All designs were then implemented in Verilog-HDL and synthesized using Design Complier with the OSU 45nm PDK. Timing and energy values were obtained through SPICE-level simulations on the sythesized designs using NanoSim and VCS. Area costs for unoptimized and optimized designs were extracted from the Design Complier synthesis report. Area overhead is further computed by comparing unoptimized and optimized results. In the FIR/IIR cases, we excited unoptimized and optimized versions of each filter with the same 4 seconds of input data. The quality of the filtered output signals is measured using a segmental SNR (SSNR) metric, which is known to be a better estimator of perceived audio quality than regular SNR [13]. SSNR is measured by dividing the output signals into segments of 20 ms and averaging over regular SNR values computed for each segment. For the image sharpening filter, the quality of filtered output images is measured using a standard peak signal-to-noise ratio (PSNR) metric.

In all cases, the nominal voltage of the filters is 1.1V and voltage scaling was applied to maintain iso performance at the fastest clock speed achievable by an unmodified base design. Area and energy overheads of our modifications for each of the filter designs in error-free operation are shown in Table I. Achievable energy savings when applying a combination of our techniques under aggressive voltage scaling are summarized in Table II. Energy levels are thereby measured at commonly accepted good quality levels of around 120dB SSNR and 23dB PSNR, respectively.

A. FIR filter for audio processing

The FIR implemented is a typical 5th-order low-pass filter based on a least-squares design method. The sampling frequency is 22kHz, the pass band ends at 6kHz, and the stop band starts at 7.5kHz. The coefficients are b=(-0.1145, 0.0558, 0.5177, 0.5177, 0.0558, -0.1145). The format for coefficients, input data, intermediate results and final outputs is Q1.21, Q3.29, Q4.50, and Q3.29, respectively. As such, the full adder width is 54 bits, while the reduced-width adder has 39 bits.



(a) SSNR vs. energy profiles

(b) Different optimized FIR filters

(c) Dynamic bitwidth for varying input data

Fig. 7. Quality-energy tradeoffs in FIR filters.

The simulation results for the FIR filter are shown in Figure 7(a). Both dynamic bitwidth adjustment as well as static reordering improve the shape of the quality-energy profile, achieving a graceful quality degradation over a wide energy range. Both techniques can delay the onset of severe timing errors when the voltage is scaled down, but bitwidth adjustment performs significantly better. Combined, significant energy savings can be obtained while maintaining almost perfect signal quality. Most of the savings are due to bitwidth adjustment alone, but with little to no additional overhead, reordering can improve results in the low energy region.

To further test the effectiveness of static reordering, we applied the technique to 7 different FIR filters with orders ranging from $N = 15$ to $N = 81$. $45\mu J$ experiment, we tested 7 Figure 7(b) shows that when pairwise comparing optimized and unoptimized versions of each filter at an energy budget of $35\mu J$, the optimized design consistently leads to more than 10dB quality gain. Gains are even larger at lower energy. Overall, results show that our techniques are effective for a wide range of filter instances. for most FIR cases, the than 20dB better quality compared

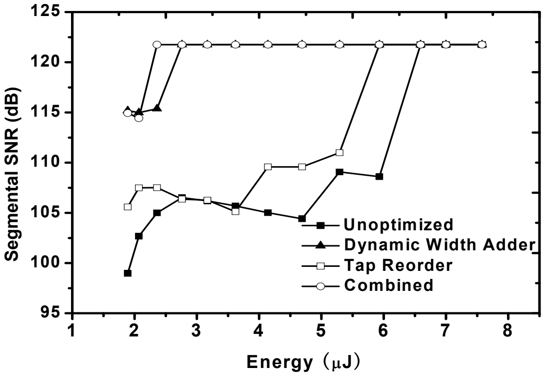


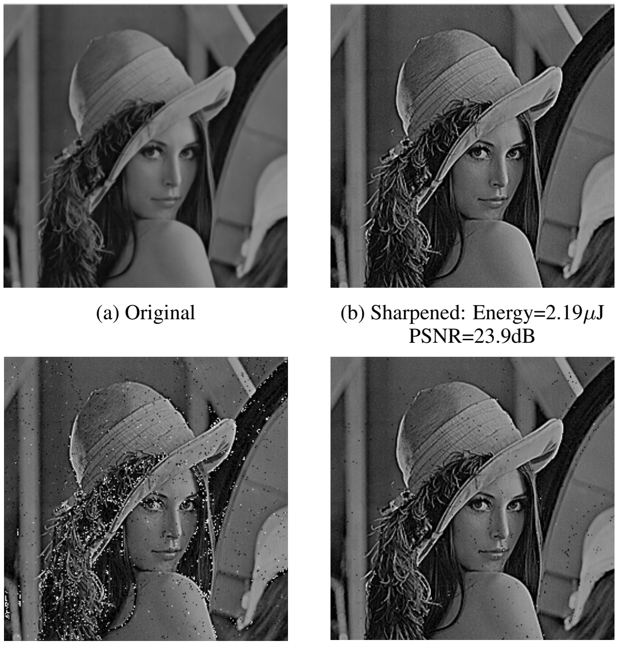
Fig. 8. SSNR vs. energy profiles in IIR filters.

We also tested the sensitivity of dynamic bitwidth adjustment to different types of input data. Figure 7(c) shows the results of feeding jazz, pop and classical music as well as speech audio files into a 63rd-order FIR filter. Results suggest that specifics of speech data trigger worse timing error behavior than in music. This is due to longer segments of silence, which are characterized by small-valued operands triggering early and large timing errors. In all cases, dynamic bitwidth adjustment significantly improves quality-energy behavior, maintaining close to perfect quality over a large energy range. With dynamic adjustment, variations in data magnitudes and their effects on timing errors can be transparently evened out across different characteristics of input data.

B. IIR filter for audio processing

We also implemented a 3rd-order low-pass type II Chebyshev filter as a typical IIR filter example. The sampling frequency is 22kHz and the cutoff frequency is 8kHz. The coefficient sets are b=(0.2282, 0.5612, 0.5612, 0.2282) and a=(-0.1652, -0.3835, -0.0300). The formats for coefficients, input data, intermediate results and final output are Q1.21, Q3.29, Q4.50, and Q3.29. The full-width and reduced-width adders in this case have 54 and 38 bits, respectively.

Results for the IIR filter are shown in Figure 8. It shows the quality-energy profiles when applying dynamic bitwidth adjustment. Both techniques can delay the onset of timing errors, but in the IIR case, dynamic bitwidth adjustment is far more effective than reordering. Furthermore, once errors start to happen, quality drops are overall more severe than in the FIR case. This is due to the feedback loop in the IIR filter, which leads to erroneous results being reused and propagating into subsequent computations. Nevertheless, as shown in Figure 8, the combination of techniques is very effective in improving the timing error behavior of the system.



(a) Original

(b) Sharpened: Energy=2.19μJ PSNR=23.9dB

(c) Unoptimized: Energy=1.27μJ PSNR=19.6dB

(d) Combined: Energy=1.27μJ PSNR=23.2dB

Fig. 9. Image sharpening example.

C. Image sharpening filter

Sharpening of images is used to increase the contrast between bright and dark regions by applying a high-pass FIR filter. We generated a coefficient kernel using MATLAB's $fspecial$ function with the filter option $unsharp$. The filter is usually realized as a 2-D convolution of each pixel with this kernel. We implement a 1-D version on our architecture using the algorithm as a 9th-order FIR filter, where the format for coefficients, input data, intermediate results and final output is Q4.12, Q8.0, Q12.12, and Q8.0. The full and reduced width adders in this case have 24 and 20 bits, respectively. Sample images after applying the sharpening filter with and without our error control are shown in Figure 9. From Figure 9(c) we can see that, compared to a sharpened image at nominal voltage (Figure 9(b)), voltage scaling without error control causes a lot of visually noticeable salt-and-pepper artifacts. By contrast, using our techniques (Figure 9(d)), such noise is significantly reduced and resulting images exhibit good perceived quality at the same reduced energy.

IV. SUMMARY AND CONCLUSIONS

This paper presented techniques that enable architecture-level shaping of the quality-energy tradeoff under aggressively scale V_{DD} through controlled timing error acceptance. The implementation of these techniques is demonstrated on a general digital filtering architecture. Results show that significant energy savings of up to 70% can be achieved while maintaining a constant performance and good SNR/PSNR. The area overhead required to achieve such savings is 2%. Future work will be concerned with extending techniques to other applications and architectures including timing error control in multiplication.

ACKNOWLEDGMENTS

This work was supported by NSF grant CCF-1018075.

REFERENCES

[1] Y. Lian and Y. J. Yu, "Low-power digital filter design techniques and their applications," *Circuits, Systems, and Signal Processing*, vol. 29, no. 2, pp. 1–5, 2010.

[2] H. Choi and W. P. Burleson, "Search-based wordlength optimization for VLSI/DSP synthesis," *VLSI Signal Processing*, pp. 198–207, 1994.

[3] O. Chen, R. Shen, and S. Wang, "A low-power adder operating on effective dynamic data ranges," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 10, no. 4, pp. 435–453, 2002.

[4] T. W. Parks and C. S. Burrus, *Digital filter design*. New York: Wiley, 1987.

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

[6] J. T. Ludwig, S. H. Nawab, and A. P. Chandrakasan, "Low-power digital filtering using approximate processing," *Journal on Solid-State Circuits (JSSC)*, vol. 31, no. 3, pp. 395–400, 1996.

[7] N. Banerjee, J. H. Choi, and K. Roy, "A process variation aware low power synthesis methodology for fixed-point fir filters," in *International Symposium on Low-Power Electronics and Design (ISLPED)*, 2007.

[8] B. Shim, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 12, no. 5, pp. 497–510, 2004.

[9] 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.

[10] K. He, A. Gerstlauer, and M. Orshansky, "Controlled timing-error acceptance for low energy IDCT design," in *Design, Automation and Test in Europe (DATE)*, 2011.

[11] ——, "Low-energy signal processing using circuit-level timing error acceptance," in *International Conference on Integrated Circuit Design and Technology (ICICDT)*, 2012.

[12] S. Gazor and W. Zhang, "Speech probability distribution," *IEEE Signal Processing Letters*, vol. 10, no. 7, pp. 390–399, 2003.

[13] P. Mermelstein, "Evaluation of a segmental SNR measure as an indicator of the quality of ADPCM coded speech," *J. Acoust. Soc. Am.*, vol. 66, no. 8, pp. 1664–1667, 1979.