Computation Error Analysis in Digital Signal Processing Systems With Overscaled Supply Voltage

Yang Liu, *Student Member, IEEE*, Tong Zhang, *Senior Member, IEEE*, and Keshab K. Parhi, *Fellow, IEEE*

***Abstract*—It has been recently demonstrated that digital signal processing systems may possibly leverage unconventional voltage overscaling (VOS) to reduce energy consumption while maintaining satisfactory signal processing performance. Due to the computation-intensive nature of most signal processing algorithms, the energy saving potential largely depends on the behavior of computer arithmetic units in response to overscaled supply voltage. This paper shows that different hardware implementations of the same computer arithmetic function may respond to VOS very differently and result in different energy saving potentials. Therefore, the selection of appropriate computer arithmetic architecture is an important issue in voltage-overscaled signal processing system design. This paper presents an analytical method to estimate the statistics of computer arithmetic computation errors due to supply voltage overscaling. Compared with computation-intensive circuit simulations, this analytical approach can be several orders of magnitude faster and can achieve a reasonable accuracy. This approach can be used to choose the appropriate computer arithmetic architecture in voltage-overscaled signal processing systems. Finally, we carry out case studies on a coordinate rotation digital computer processor and a finite-impulse-response filter to further demonstrate the importance of choosing proper computer arithmetic implementations.**

***Index Terms*—Computation error analysis, computer arithmetic, digital signal processing, voltage overscaling (VOS).**

I. Introduction

VOLTAGE SCALING is an effective technique to reduce the energy consumption in CMOS integrated circuits [1], [2]. In conventional practice, voltage scaling is lower bounded by $V_{\mathrm{dd-crit}}$, under which the critical path delay equals the target clock period. Voltage overscaling (VOS), i.e., overscaling the supply voltage below $V_{\mathrm{dd-crit}}$, can result in transient circuit timing errors, which is generally not allowed in current design practice. It is well known that there is a large degree of discrepancy between the average-case and the worst-case circuit delay in practice, particularly in many computer arithmetic functions. This suggests that, even under an overscaled supply voltage, computer arithmetic functions may have a relatively low probability to produce errors in each clock cycle. Intuitively, this feature may be exploited to enable the use of VOS to reduce the energy consumption of computational datapath, as demonstrated in recent works [3]–[6]. In this context, the key issue is how to maintain the satisfactory functionality in the presence of the transient errors incurred by VOS while ensuring that the overall system energy consumption is reduced. The techniques proposed in [3] and [4] tackle this issue by using a detect/speculate-then-recover mechanism, i.e., first, the occurrence of transient errors is either speculated [3] based on run-time data characteristics or detected [4] using special circuitry, and then, certain system-level operations are executed to recover the errors. This type of approach is suitable to applications such as general-purpose computing that may tolerate the latency overhead due to error recovery operations. However, they may not be applicable to most digital signal processing functions that typically perform real-time continuous data processing, which makes it difficult to support such a detect/speculate-then-recover flow.

Particularly targeting digital signal processing systems, the design methodology proposed in [5] and [6] intends to compensate the signal processing performance degradation incurred by transient errors instead of trying to recover all the errors. It can be justified by the fact that, as pointed out in [5] and [6], most signal processing functions mainly address certain quantitative high-level performance criteria [e.g., signal-to noise ratio (SNR)] and that circuit transient errors may not necessarily make the signal processing performance unacceptable. Techniques for compensating the signal processing performance degradation incurred by VOS have been developed for linear filters [5], [7], [8] and fast Fourier transform [9]. Furthermore, it is not uncommon that signal processing systems are designed to meet a stringent system performance criterion in order to handle the worst-case run-time scenario, which may be far more stringent for average cases. Intuitively, this provides a potential to directly apply VOS for energy reduction without using any performance compensation schemes in some circumstances.

In most signal processing systems, various computer arithmetic functions, particularly addition and multiplication, are major building blocks and typically constitute the critical paths. Therefore, the signal processing performance degradation incurred by VOS heavily depends on the output transient error characteristics of those computer arithmetic functions in response to overscaled supply voltage. All the prior works [5]–[9] on voltage-overscaled signal processing system design assumed the use of carry-ripple adder architecture in the realizations of adders and multipliers. Intuitively, different computer arithmetic architectures (e.g., carry-ripple adder, carry-select adder, and carry-lookahead adder) may respond to overscaled supply voltage differently, leading to different output transient error characteristics, even though they have the same critical path delay. This will further result in different signal processing performance degradation and, hence, different energy saving potentials in voltage-overscaled signal processing systems.

Manuscript received June 17, 2008; revised October 19, 2008 and January 04, 2009. First published June 16, 2009; current version published March 24, 2010.

Y. Liu and T. Zhang are with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: liuy8@rpi.edu; tzhang@ecse.rpi.edu).

K. K. Parhi is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: parhi@ece.umn.edu).

Digital Object Identifier 10.1109/TVLSI.2009.2012863

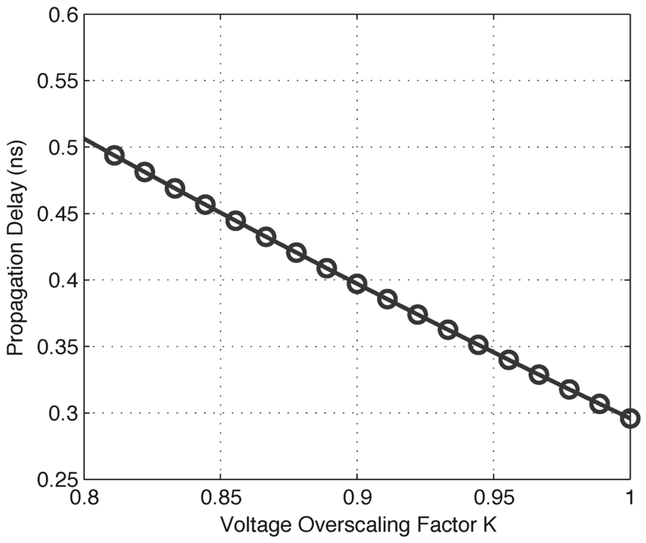


Fig. 1. Simulated propagation delay versus supply voltage for a 1-b full adder.

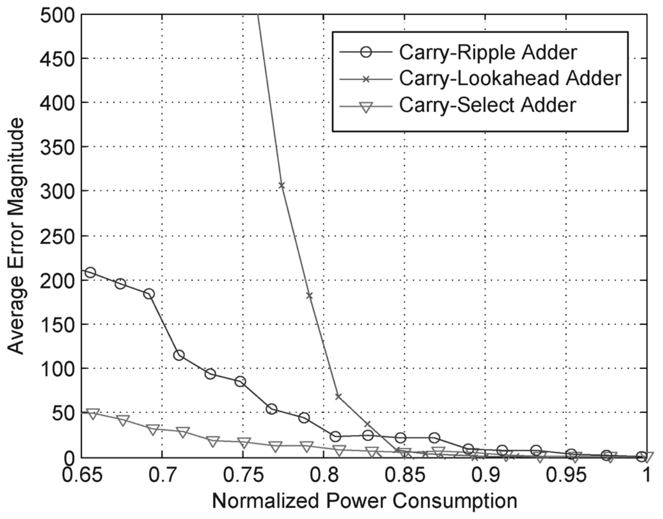


Fig. 2. Simulated average error magnitude versus normalized power consumption for the three 16-b adders operating at 800 MHz.

Therefore, computation error characteristic analysis of computer arithmetic units in voltage-overscaled signal processing systems is an important issue, which nevertheless has not been addressed in the open literature to the best of our knowledge.

This paper attempts to address this open issue. First, using adders as a test vehicle, we empirically demonstrate that, to realize the same computer arithmetic function, different architectures can indeed possibly result in significantly different output transient error characteristics. Then, by leveraging prior works on switching activity estimation, we develop an analytical method that can estimate the average magnitude of computation errors of voltage-overscaled computer arithmetics. Compared with using extensive circuit simulation, this proposed analysis method can be several orders of magnitude faster and meanwhile achieve a reasonable accuracy. This method can be used to effectively analyze the signal processing performance in the presence of overscaled supply voltage and facilitate the selection of appropriate computer arithmetic architectures. Moreover, we carried out case studies on a COordinate Rotation DIgital Computer (CORDIC) processor and an finite-impulse-response (FIR) filter to further demonstrate the importance of choosing proper computer arithmetic implementations. Finally, we note that, for all the experimental studies presented throughout this paper, we use Synopsys DesignWare to generate the arithmetic units without any manual optimization in order to ensure a reasonably fair comparison among different arithmetic architectures.

The rest of this paper is organized as follows. Section II uses 16-b adder design as an example to demonstrate that different architectures can result in significantly different transient error characteristics. Section III presents our ffproposed analytical method for estimating average magnitude of transient errors in voltage-overscaled computer arithmetic units. Section IV presents the case studies on voltage-overscaled CORDIC processor and FIR filter, and conclusions are drawn in Section V.

II. Motivating Example

With 16-b adder design as an example, this section shows that different computer arithmetic architectures may respond to overscaled supply voltage very differently, leading to largely different transient error statistical characteristics. In particular, we considered three adder architectures, including the carry-ripple adder, the carry-lookahead adder, and the carry-select adder. We use Synopsys DesignWare to generate these adders using a TSMC 65-nm CMOS standard cell library without any manual optimizations. The timing constraint is set to 1.25 ns (i.e., 800 MHz) at the normal 0.9-V supply voltage, and all the synthesized adders have the same timing slack. Hence, they have the same critical voltage $V_{\mathrm{dd-crit}}$ of 0.9 V. Power estimations are performed based on designs with back-end place and route information. When operating at 0.9 V and 800 MHz, the carry-ripple adder consumes 0.570 mW, the carry-lookahead adder consumes 0.508 mW, and the carry-select adder consumes 0.578 mW, where both dynamic power and leakage power are taken into account. Therefore, in conventional practice, we may want to choose the carry-lookahead adder. As shown in the following, a different choice may be preferred in the presence of VOS.

We use the Synopsys composite current source (CCS) model [10] to perform simulations under VOS. The CCS model is a complete open-source current-based modeling solution for timing, noise, and power analysis. Because CCS is current based, it can enable both temperature and voltage scaling of the cell behavior and achieve the timing analysis accuracy within 2% of SPICE, which is better than the conventional nonlinear delay and power models [11]. During the simulation, we set VOS factor K in the range of 0.8–1 (i.e., supply voltage ranges between 0.72 and 0.9 V).

First, to evaluate the propagation delay versus supply voltage characteristic of the TSMC 65-nm standard cell library, we carried out simulations on a 1-b full adder (with a load of an inverter with ten times minimum size), as shown in Fig. 1. It shows that the propagation delay is almost linearly proportional to the supply voltage. We further carried out simulations on the aforementioned three 16-b adders under different supply voltages while fixing the frequency as 800 MHz. For each simulation run, we randomly generated 10^6 pairs of 16-b input data, where each bit has equal probability to be 0 or 1 and all the bits are

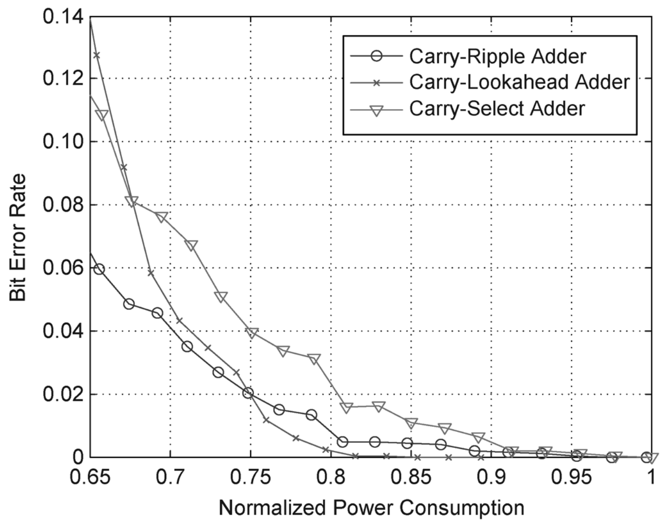


Fig. 3. Simulated bit error rate versus normalized power consumption for the three 16-b adders operating at 800 MHz.

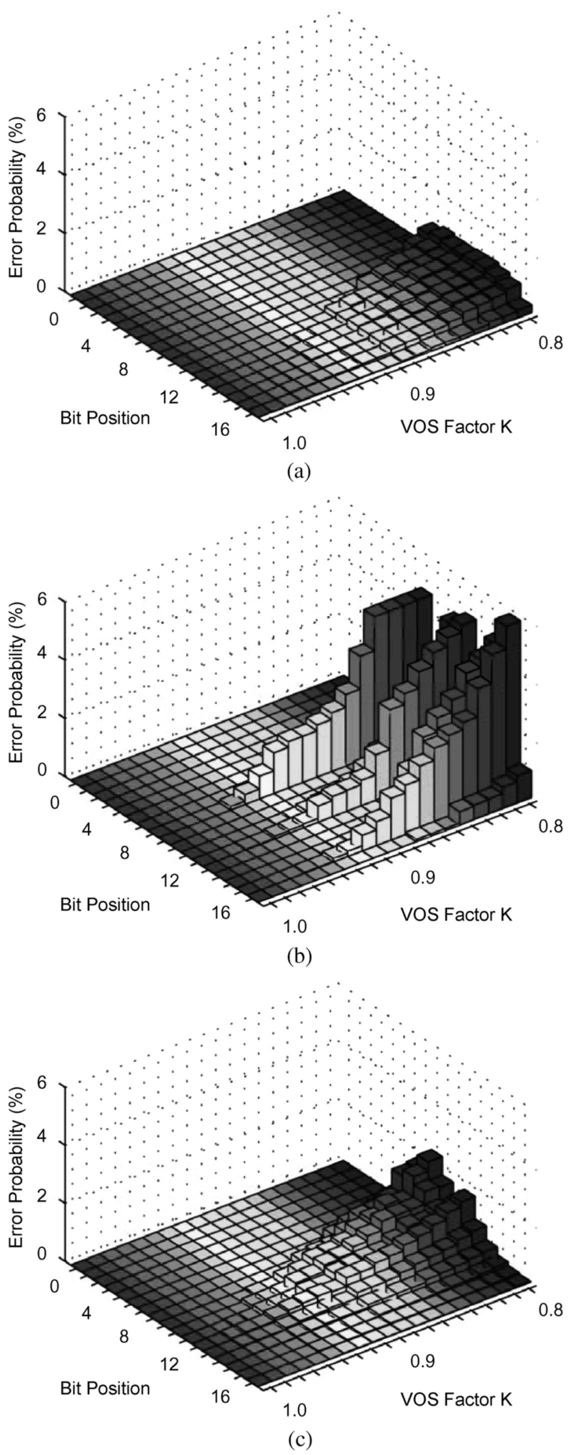


Fig. 4. Error occurrence probability of each bit of the three different adders. (a) Error distributions of the carry-ripple adder. (b) Error distributions of the carry-lookahead adder. (c) Error distributions of the carry-select adder.

randomly generated independent from each other. Fig. 2 shows the simulated average error magnitude versus normalized power consumption, where the average error magnitude is the mean of the computation error magnitude over all the simulated samples and the power consumptions are normalized against the highest power consumption among all the adders under a normal supply voltage. Fig. 3 shows the simulated average bit error rate (i.e., the number of bits that is wrongly computed) versus normalized power consumption. It shows that, under an overscaled supply voltage, the carry-ripple adder has the least bit error rate, while the carry-select adder has the least performance degradation if we take into account the significance of each output bit. Although all the adders have similar bit error rates (i.e., 0.06–0.14), they have very different average error magnitudes, as shown in Fig. 2. Since the performance degradation of signal processing systems under an overscaled voltage heavily depends on the computation error magnitude, the aforesaid results suggest that we should not use mere computation bit error rate as a metric to select the appropriate arithmetic architecture and that we should be able to directly estimate the average error magnitude characteristics for each candidate arithmetic architecture.

Fig. 4 further shows the relations between the error occurrence probability for each bit and the supply voltage for these three adders. Bits 0, 15, and 16 correspond to the least significant bit (LSB), most significant bit (MSB), and adder carry out, respectively. It is clear that Fig. 4 can approximately explain the different average error magnitude versus supply voltage characteristics among the three adders, as shown in Fig. 2. The different error distribution characteristics, as shown in Fig. 4, may be intuitively explained as follows.

1) In a carry-ripple adder, a more significant bit has longer worst-case delay and can hence be more subject to errors. On the other hand, the worst-case-delay paths of a more significant bit tend to be activated less frequently. Due to these two somehow conflicting trends, the error occurrence probabilities of those more significant bits tend to be flat, as shown in Fig. 4(a).
2) A carry-lookahead adder consists of several segments that form a carry-lookahead chain, and within each segment, a carry-ripple adder should be used. As a result, the MSBs within each segment tend to have similar worst-case delay that are the same or close to the overall carry-lookahead-adder critical path. Therefore, these MSBs within all the segments are more subject to errors, as shown in Fig. 4(b).

3) A carry-select adder also consists of several segments. However, the two possible addition results of each segment (corresponding to the two scenarios in which the carry input to this segment is 0 and 1) are precomputed and are simply selected by the carry input. Therefore, the less significant bits within each segment may be more subject to errors compared with the more significant bits in the same segment. Since the adder is generated using DesignWare, the segment partition information is not readily available. From Fig. 4(c), we guess that the bits around bit positions 8 and 12 are LSBs within two adjacent segments.

The aforementioned results suggest that, even though the carry-lookahead adder has the minimum power consumption under a normal supply voltage, the carry-select adder may be preferred in voltage-overscaled digital signal processing system design since the average error magnitude may directly affect the signal processing performance degradation.

III. Error Analysis of Voltage-Overscaled Computer Arithmetics

Since different computer arithmetic implementations may respond to overscaled supply voltage very differently, it is highly desirable to develop a method that can quantitatively reveal the transient error characteristics of various computer arithmetic implementations. Although brute-force circuit simulations can serve this purpose, it suffers from very high computational complexity, even for small circuits (e.g., a 16-b adder). This makes it impossible to explore computer arithmetic design space over a wide range of VOS by relying on circuit simulations and hence demands analytical techniques that can efficiently estimate the transient computation error characteristics while maintaining a reasonable accuracy.

Motivated by the obvious correlation between the circuit switching activity and transient computation error, this paper aims to develop computation error analysis techniques by leveraging prior works on circuit switching activity estimation. Generally speaking, the objective of switching activity estimation is to estimate the switching probabilities of each gate in circuits over time, given the switching probabilities of the input. A variety of switching estimation techniques have been developed with different accuracy versus efficiency tradeoffs, e.g., see [12]–[15].

A first glance may suggest that the estimated switching activities of the computation output bits can be directly used to derive the transient computation error characteristics in a very straightforward manner, which unfortunately turns out to be wrong. For example, let us consider an 8-b unsigned carry-select adder, as shown in Fig. 5, where the critical path lies in the carry-select signal propagation. Suppose that the correct output should be 10000000 at one time, but the output becomes 01110000 because of a critical path violation due to VOS. Clearly, the output error magnitude is $2^4 = 16$ (i.e., $10000000 - 01110000 = 00010000$). However, if we consider the switching of the output bits independently, the magnitude of computation error becomes $2^7+2^6+2^5+2^4 = 240$ since all the four MSBs switch to erroneous values. This example shows that we cannot only rely on the switching activity estimation of computation output bits to

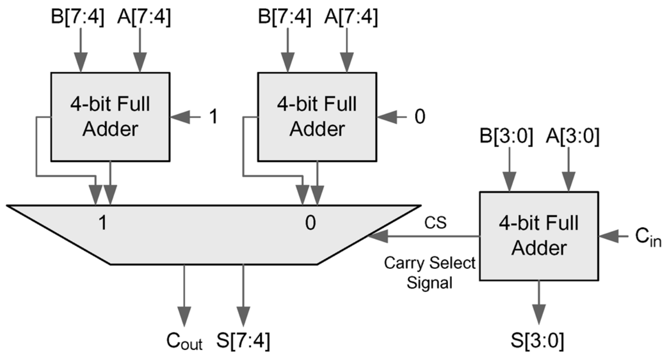


Fig. 5. Structure of an 8-b carry-select adder.

derive the computation output error characteristics because of the correlation among the output-bit switching errors.

Therefore, we must explicitly preserve the correlations among the computation-output-bit switching errors. It is clear that such correlations are due to the fact that each internal signal (or the output of each logic gate) in computer arithmetic circuits may contribute to more than one computation output bit. For example, the carry-select signal in the previous example directly affects all the four most significant output bits. Therefore, the correlations among the computation-output-bit switching errors can be revealed by explicitly observing the contribution of each internal signal's switching to the overall computation output error, which is referred to as the *error significance* of each internal signal. For any internal signal, let W^e denote the maximum computation error magnitude incurred by the switching of this signal. Then, the error significance of this signal depends on both its W^e and switching activity. We call W^e as the error significance factor of this internal signal. In the following, we first discuss how to derive the error significance factor for each internal signal in computer arithmetic circuits and then present how we may estimate the overall computation output error magnitude by combining the error significance factors and switching activities.

A. Error Significance Factor Assignment

The error significance factor associated with each internal signal in computer arithmetic circuits represents the maximum computation output error magnitude that is incurred if the present internal signal cannot propagate to the output bits in time. For example, let us again consider the 8-b unsigned carry-select adder, as shown in Fig. 5. As discussed previously, if a switch on the carry-select signal cannot propagate to the four MSBs in time, the output error magnitude is $2^4 = 16$. Hence, we have that the error significance factor of the carry-select signal is eight. Motivated by this example, we have the following simple rules to determine the error significance factor of each internal signal:

1) If an internal signal only contributes to one computation output bit, then its error significance factor W^e is assigned as the weight of the corresponding output bit.
2) If an internal signal contributes to a group of computation output bits, then its error significance factor W^e is assigned as the minimum one among the weights of the output-bit group.

```
Weight_Assignment (Circuit_Topology)
1   for each output bit from MSB to LSB {
2     trace_back (current signal) {
3       if current signal is a input bit {
4         return (NULL);
5       }
6       else {
7         for each parent signal of the current signal {
8           set parent signal's $W^e$ = current signal's $W^e$;
9           trace_back (parent signal);
10        }
11      }
12    }
13  }
```

Fig. 6. Pseudocode for the error significance factor assignment.

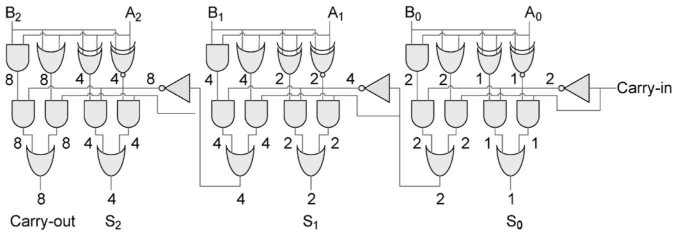


Fig. 7. Example of weight assignment.

In general, the ith computation output bit has a weight of 2^i, where $i = 0$ corresponds to the LSB. Given the computer arithmetic circuit topology, we can use a simple trace-back procedure to automatically derive the error significance factor of each internal signal, where the pseudocode is shown in Fig. 6. The inputs of each gate are called parent signals of its output. Starting from the computation output bit with the highest weight (i.e., the MSB), the weight of each output bit simply propagates backward through all the connected gates, and the later propagated values overwrite the early ones. This is further shown in Fig. 7 for a 3-b adder.

B. Computation Error Magnitude Estimation

Once the error significance factors of all the internal signals in computer arithmetic circuits are determined, they will be combined with the internal signal switching activity information to estimate the average computation output error magnitude. Denote the probability that a signal f_i is logic 1 at a particular time point τ as $P(f_i = 1 : t = \tau) = P_i^1(\tau)$ and the probability that the signal is logic 0 as $P_i^0(\tau) = 1 - P_i^1(\tau)$. Each signal has four possible transitions (i.e., 0→0, 0→1, 1→1, and 1→0) between two adjacent time points, and we denote the corresponding transition probabilities as $P_i^{00}(\tau)$, $P_i^{01}(\tau)$, $P_i^{11}(\tau)$, and $P_i^{10}(\tau)$. The objective of switching activity estimation is to accurately estimate $P_i^{00}(\tau)$, $P_i^{01}(\tau)$, $P_i^{11}(\tau)$, and $P_i^{10}(\tau)$ for all the gates in the circuits at all the discrete time points. The granularity of the discrete-time-point sequence (i.e., the distance between two adjacent time points) depends on the requirement of estimation accuracy and the technology library being used. Readers are referred to [12]–[15] for existing work on switching activity estimation. In this paper, we mainly use the approach addressed in [12] and [14] to calculate these transition probabilities.

Under an overscaled supply voltage, the critical path delay of computer arithmetic circuits will be longer than the target clock period (denoted as T_{clk}). Hence, any switching inside the circuits at time T_{clk} may potentially result in computation error, i.e., the average magnitude of the computation output errors incurred by the switching of internal signal k at T_{clk} can be estimated as

$$E_k = W_k^e \times \left(P_k^{01}(T_{\text{clk}}) + P_k^{10}(T_{\text{clk}})\right) \tag{1}$$

where W_k^e is the error significance factor of internal signal k. Assuming that all the switchings of the internal signals at T_{clk} will independently and additively contribute to the computation output error, we can estimate the average computation error magnitude as

$$E = \sum_{k \in \mathcal{S}} \left(W_k^e \times \left(P_k^{01}(T_{\text{clk}}) + P_k^{10}(T_{\text{clk}})\right)\right) \tag{2}$$

where the set $\mathcal{S}$ contains all the internal signals in computer arithmetic circuits.

We further implemented a software package to realize the aforementioned computation output error estimation approach, where Fig. 8 shows the software flowchart. It extracts the circuit topology from a synthesized gate-level netlist that is further sorted by topological ordering [16] to facilitate the trace-back error significance factor assignment procedure presented earlier. The timing information of the circuits is obtained from the back-annotated stand-delay-format files generated by the Synopsys CCS model under VOS.

For the purpose of evaluation, we apply this proposed approach to the same three 16-b adders discussed in Section II (i.e., the carry-ripple adder, the carry-lookahead adder, and the carry-select adder). Fig. 9 shows the analysis results and the comparison with the results obtained from very time-consuming gate-level circuit simulations over 200 000 random input pairs. Furthermore, we also use Synopsys DesignWare to generate three 16-input two's-complement multipliers that use a carry-ripple adder, a carry-lookahead adder, or a carry-select adder as the final-stage merge adder, respectively. They are denoted as

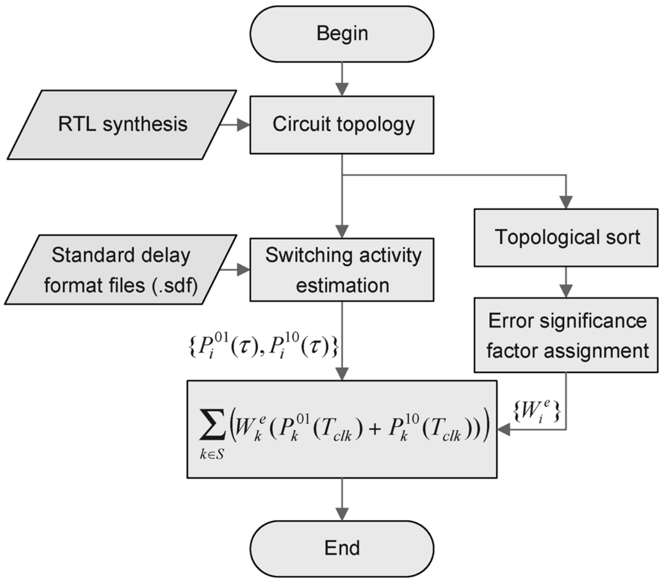

Begin
RTL synthesis
Circuit topology
Topological sort
Standard delay format files (.sdf)
Switching activity estimation
Error significance factor assignment
$\{P_i^{01}(\tau), P_i^{10}(\tau)\}$
$\sum_{k\in S}\left(W_k^e(P_k^{01}(T_{clk}) + P_k^{10}(T_{clk}))\right)$
$\{W_i^e\}$
End

Fig. 8. Flowchart of computation output error estimation.

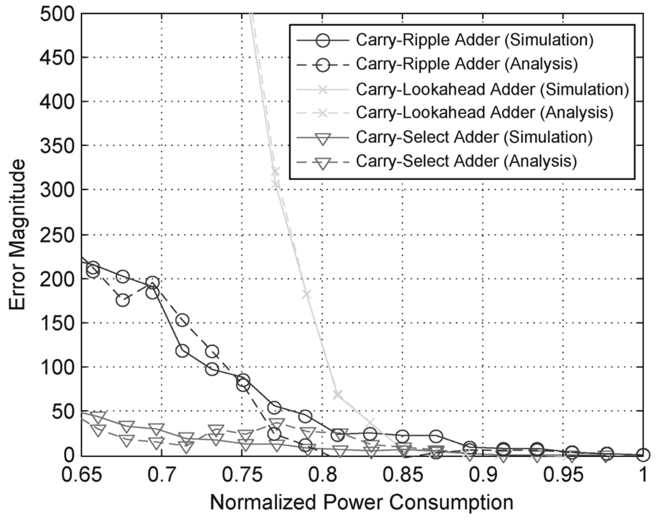


Fig. 9. Estimated and simulated average error magnitudes versus normalized power consumption for the three 16-b adders operating at 800 MHz.

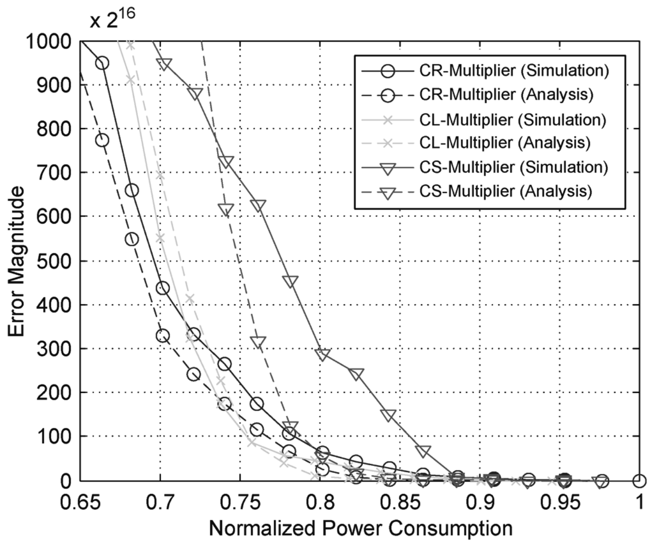


Fig. 10. Estimated and simulated average error magnitudes versus normalized power consumption for the three 16-b multipliers operating at 500 MHz.

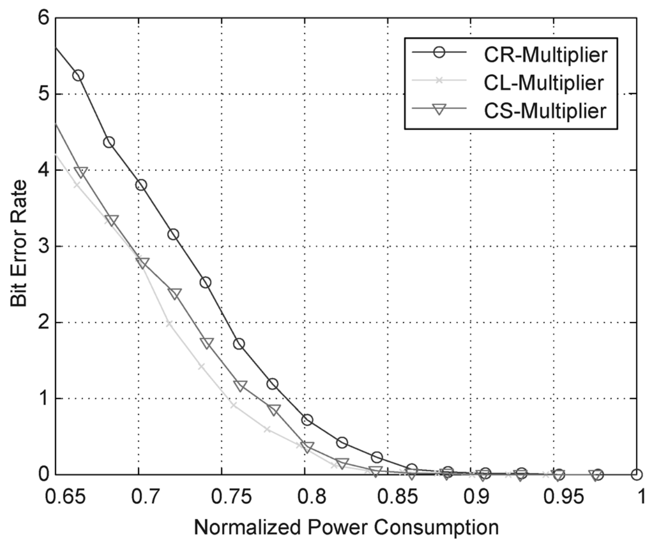


Fig. 11. Bit error rate versus normalized power consumption for the three 16-b multipliers operating at 500 MHz.

CR-Multiplier, CL-Multiplier, and CS-Multiplier, respectively. All the three multipliers have the same timing slack and can run at 500 MHz under the normal supply voltage of 0.9 V. Fig. 10 shows the analysis results for these three multipliers and the comparison with the results obtained from gate-level circuit simulations over 200 000 random input pairs. We note that, in both Fig. 9 and Fig. 10, the absolute values of the estimated and simulated results are somehow different, which is at least partially due to the estimation inaccuracy induced by ignoring those deep correlations, as discussed earlier, but they reveal the same comparative trends. Since our objective is to determine which architecture should be selected, it is the comparative trends, other than the absolute values, that is the most important in this context. Fig. 11 shows the corresponding bit error rate for these three multipliers under an overscaled voltage. They clearly have different trends compared with the results on average error magnitude, as shown in Fig. 10.

In general, for m-input N-bit computation arithmetic circuits, the computational complexity of exhaustive gate-level simulations is on the order of $O(2^{m\cdot N})$, while the computational complexity of this analysis approach is the same as that of switching activity estimation techniques, which is usually on the order of $O(N)$–$O(N^m)$. This suggests orders of magnitude speedup by using the proposed analysis-based approach to estimate the computation error magnitude. For example, for the 16-b adders considered earlier, an exhaustive gate-level simulation may take about 16 h, while the proposed method only takes less than 1 min.

IV. Applications to Digital Signal Processing System Design

This section presents case studies on voltage-overscaled CORDIC processor and FIR filter to further illustrate the impact of various arithmetic unit architectures.

A. Case Study on CORDIC Processor

CORDIC processor is widely used in signal processing systems to calculate hyperbolic, trigonometric, or logarithmic functions without the implementation of multipliers [17]. This work considers using CORDIC processors to realize vector rotation [18], i.e., obtain the vector (x_f, y_f) by rotating a vector (x_0, y_0) with angle θ. CORDIC processors realize the vector rotation in an iterative manner, where the operations during each iteration are

$$\begin{aligned} x_{i+1} &= x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} &= y_i + x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} &= z_i - d_i \cdot \tan^{-1}(2^{-i}) \end{aligned}$$

where $d_i = -1$ if $z_i < 0$ (+1 otherwise) and z_0 is the rotation angle in the binary representation. Fig. 12 shows the structure of an n-stage pipelined CORDIC processor. In this case study, we set $n = 5$ and employ 10-b two's-complement fractional number representation for the data.

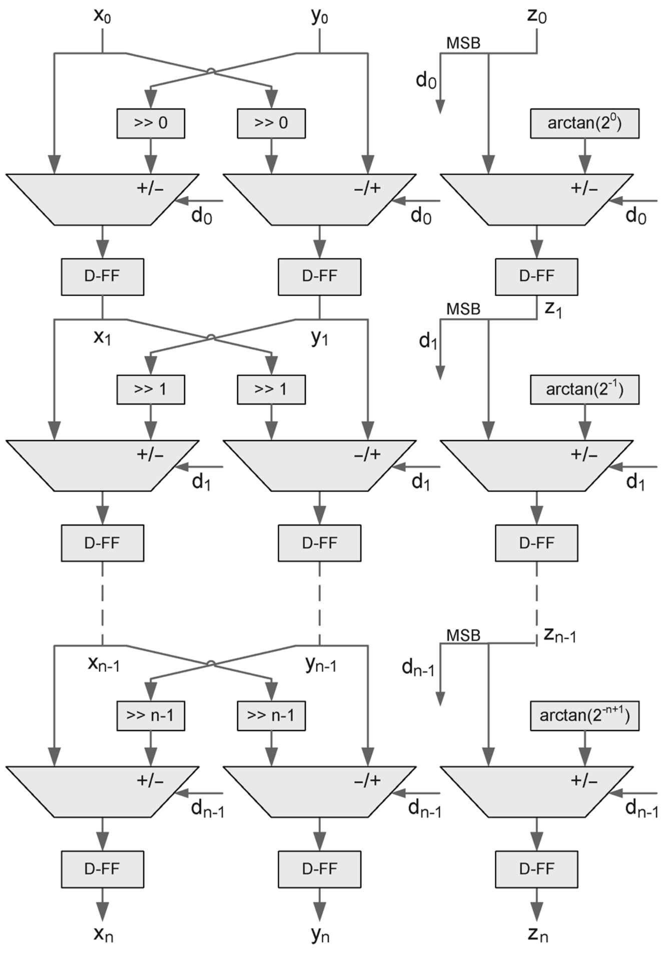


Fig. 12. CORDIC structure.

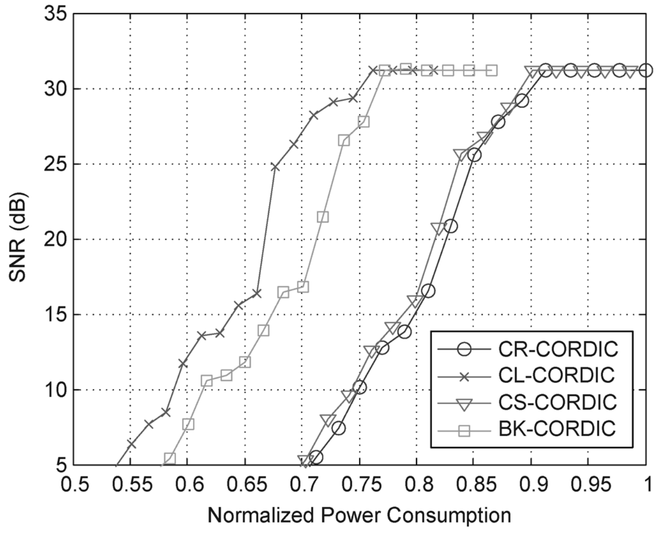


Fig. 13. Estimated CORDIC processor performance under VOS based on the error estimation of each adder.

We studied the impact of different adder architectures on the operation of CORDIC processor under VOS. Using Synopsys DesignWare with the TSMC 65-nm CMOS standard cell library, we obtained four different flip-flop pipelined implementations of the CORDIC processor where addition and subtraction

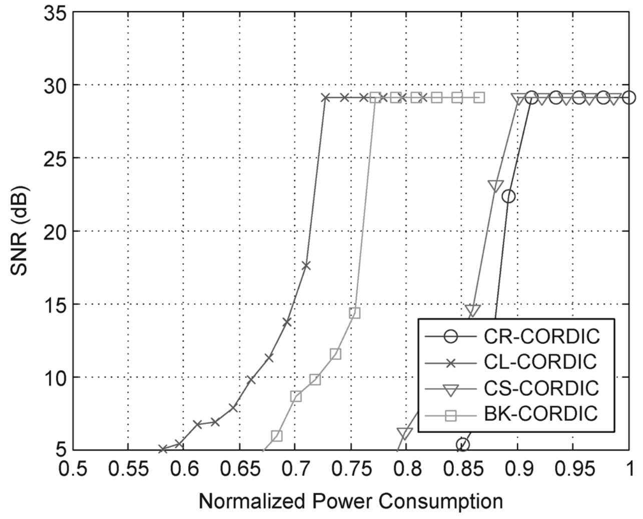


Fig. 14. Simulated CORDIC processor performance under VOS based on the Monte Carlo simulation of the entire CORDIC processor.

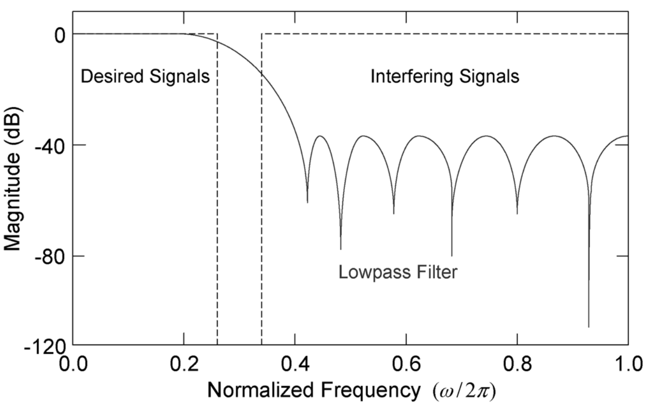


Fig. 15. FIR filter frequency response and run-time environment setup.

are realized by a carry-ripple adder, a carry-lookahead adder, a carry-select adder, and a Brent–Kung (BK) adder, respectively. We denote these four different CORDIC processors as CR-CORDIC, CL-CORDIC, CS-CORDIC, and BK-CORDIC, respectively. All these CORDIC processors have the same critical path of 1.4 ns. The approach developed in Section III is used to evaluate the performance of CORDIC processors under overscaled voltages. We assume that all the primary input bits are statistically independent and have equal probability of being 1 and 0. We first estimate the magnitude of computation errors at each pipeline stage. Similar to conventional quantization error analysis [19], we treat these computation errors at each stage as independent additive error sources, based on which we obtain the overall computation output average error magnitude through Monte Carlo simulations with 10 000 random samples. We assume that the performance of CORDIC processor is measured by

$$SNR = 10\log_{10}\frac{\sigma_s^2}{\sigma_n^2+\sigma_e^2} \quad (3)$$

where $\sigma_s^2 = x_f^2 + y_f^2$ is the signal power of error-free rotated vector (x_f, y_f) that is calculated by Matlab's built-in trigonometric function, σ_n^2 is the variance of errors due to quantization

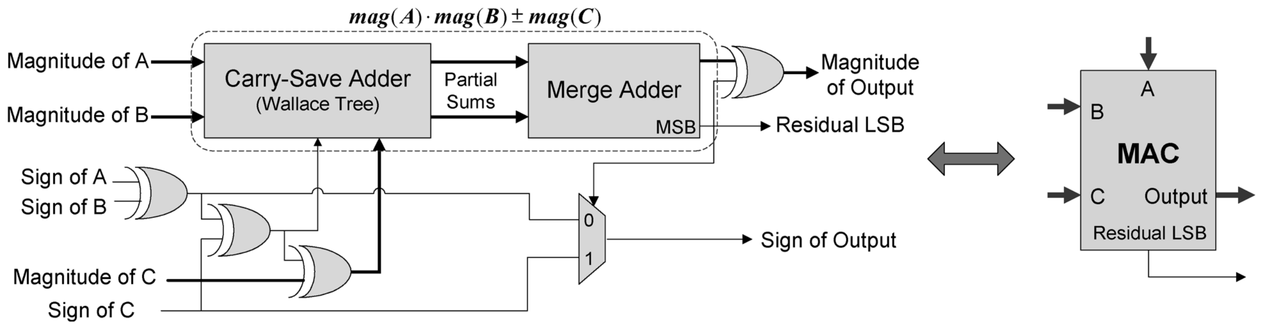


Fig. 16. Architecture of MAC ($A \cdot B + C$) with sign–magnitude data input/output.

and limited number of iterations, and σ_e^2 denotes the variance of computation output errors due to overscaled voltage.

Fig. 13 shows the estimated SNR versus normalized power consumption under different overscaled voltages. Although the four different CORDIC processors have the same critical path, they clearly respond to VOS very differently, and the estimation results suggest that carry-lookahead and BK adders should be selected to achieve better power-performance tradeoff. Moreover, as shown in Fig. 13, even if SNR performance is not allowed to be sacrificed, we may still use VOS to reduce the power consumption by about 25% when either BK-CORDIC or CL-CORDIC is used. To further testify this conclusion, we conducted much more time-consuming simulations on the entire CORDIC processor using these four different arithmetic units, and Fig. 14 shows the simulated SNR versus normalized power consumption, which agrees with the aforementioned conclusion that BK-CORDIC or CL-CORDIC may be preferred. Although the absolute values of estimations (as shown in Fig. 13) and simulations (as shown in Fig. 14) are different, both estimations and simulations reveal the same trends and comparisons among various design options. Hence, the proposed estimation approach can well serve the purpose of selecting appropriate computer arithmetic architecture.

B. *Case Study on FIR Filter*

The second case study is carried out on a 19-tap low-pass FIR filter with $(0 \sim 0.3)\cdot\omega/2\pi$ passband, as shown in Fig. 15. This FIR filter intends to extract the desired signal within $(0 \sim 0.25)\cdot\omega/2\pi$ from the interfering signals within $(0.35 \sim 1)\cdot\omega/2\pi$ and additive white Gaussian noise (AWGN). Its filtering performance is measured in terms of SNR. For the finite word-length configuration, we use 17-b fixed-point representation for the filter input/output, filter coefficients, and all the intermediate data, where the point sits between the MSB and the second MSB.

This FIR filter has a transposed structure, so it only contains one type of arithmetic unit, i.e., MAC at each filter tap. For the design of MAC, as pointed out in prior works [1], [5], using sign–magnitude input data representation may reduce the dynamic power consumption and, more importantly, reduce the error occurrence on more significant bits of the MAC output when VOS is being used. Therefore, we use the sign–magnitude data representation for the input/output of the MACs, as shown in Fig. 16. Let $mag(\cdot)$ and $sign(\cdot)$ represent the magnitude and sign of the data, respectively. As shown in Fig. 16,

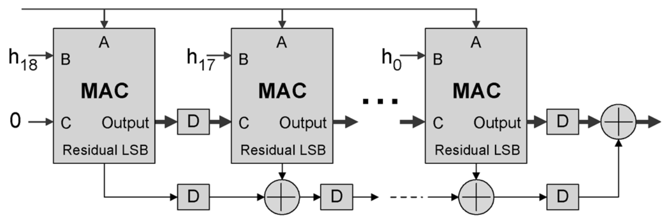


Fig. 17. Structure of the FIR filter.

the carry-save adder, followed by the merge adder, calculates $mag(A)\cdot mag(B) + mag(C)$ if $sign(A\cdot B)$ equals $sign(C)$, or $mag(A)\cdot mag(B) - mag(C)$ otherwise, and the calculation is realized by internally representing $mag(A)\cdot mag(B)$ and $\pm mag(C)$ with two's-complement format. The output of the merge adder is $D = mag(A)\cdot mag(B) \pm mag(C)$ in two's-complement representation. To obtain the magnitude of D, we further XOR its MSB with all the other bits and meanwhile add its MSB to the LSB position. To remove the overhead incurred by such an extra addition in each MAC, we can delay it to the final FIR filter output and refer the MSB of D as the residual LSB output, as shown in Fig. 16. Accordingly, we have the overall FIR filter structure, as shown in Fig. 17.

Again, the TSMC 65-nm standard cell library is used to synthesize the FIR filter. We implemented four different MACs that only differ at the merge adder, where a carry-ripple adder, a carry-lookahead adder, a carry-select adder, or a BK adder is used. We denote these four different MACs as CR-MAC, CL-MAC, CS-MAC, and BK-MAC, respectively. The four MACs have the same critical path of 3.6 ns. We use the same (3) in the previous to calculate the SNR of the FIR filter, except that σ_n now denotes the variance of AWGN. The power of AWGN is -38 dB, corresponding to $\sigma_n^2 = 1.56 \times 10^{-4}$. Since the filtering performance is measured in terms of SNR, the variance of the MAC output error is used in the calculation. Fig. 18 shows the estimated SNR versus normalized power consumption. It suggests that CL-MAC and BK-MAC can achieve modestly better power-performance tradeoff.

We further carried out extensive simulations on the entire FIR filter to evaluate the effects when different MACs are used under VOS. Fig. 19 shows the simulated SNR versus normalized power consumption, which agrees with the previous conclusion that CL-MAC or CR-MAC may be preferred. Again, although the absolute values of estimations and simulations are different,

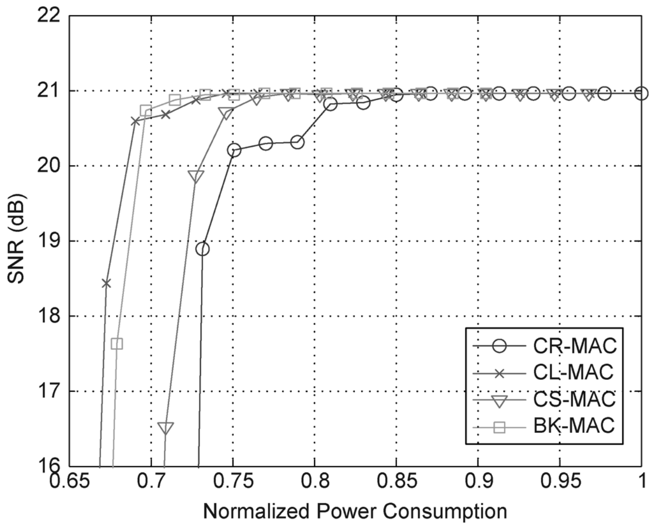


Fig. 18. Estimated FIR performance under VOS based on the error estimation of each MAC.

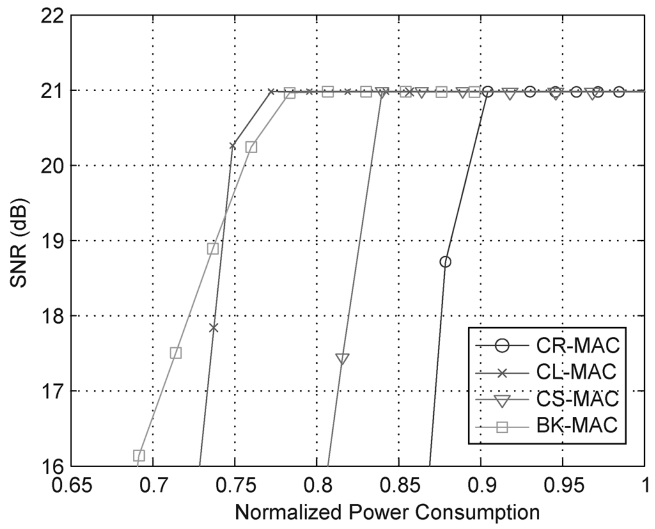


Fig. 19. Simulated FIR performance under VOS based on the Monte Carlo simulation of the entire FIR filter.

they reveal the same trends and comparisons among various design options.

V. Conclusion

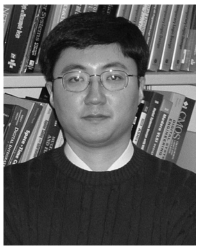
This paper addressed the issue of computation error analysis of computer arithmetic units in voltage-overscaled signal processing systems. For the first time, we showed that different arithmetic unit architectures may respond to VOS very differently, which would result in different energy saving potentials. For the search of proper computer arithmetic architecture, we developed an analytical method that could efficiently estimate the computation error statistics of computer arithmetic units under an overscaled voltage. Moreover, for the purpose of demonstration, we presented case studies on voltage-overscaled CORDIC processor and FIR filter. In the future, we will take efforts on extending our work to support more kinds of errors and taking technology process variations into account.

References

[1] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits," *Proc. IEEE*, vol. 83, no. 4, pp. 498–523, Apr. 1995.
[2] R. Gonzalez, B. M. Gordon, and M. A. Horowitz, "Supply and threshold voltage scaling for low power CMOS," *IEEE J. Solid-State Circuits*, vol. 32, no. 8, pp. 1210–1216, Aug. 1997.
[3] T. Liu and S.-L. Lu, "Performance improvement with circuit-level speculation," in *Proc. 33rd Annu. IEEE/ACM Int. Symp. MICRO*, Monterey, CA, Dec. 2000, pp. 348–355.
[4] S. Das, D. Roberts, S. Lee, S. Pant, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "A self-tuning DVS processor using delay-error detection and correction," *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 792–804, Apr. 2006.
[5] R. Hegde and N. R. Shanbhag, "Soft digital signal processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 6, pp. 813–823, Dec. 2001.
[6] N. R. Shanbhag, "Reliable and energy-efficient digital signal processing," in *Proc. Des. Autom. Conf.*, Jun. 2002, pp. 830–835.
[7] L. Wang and N. R. Shanbhag, "Low-power filtering via adaptive error-cancellation," *IEEE Trans. Signal Process.*, vol. 51, no. 2, pp. 575–583, Feb. 2003.
[8] R. Hegde and N. R. Shanbhag, "A voltage overscaled low-power digital filter IC," *IEEE J. Solid-State Circuits*, vol. 39, no. , pp. 388–391, Feb. 2004.
[9] B. Shim, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 5, pp. 497–510, May 2004.
[10] SYNOPSYS Composite Current Source [Online]. Available: http://www.synopsys.com/
[11] G. Mekhtarian, Composite Current Source (CCS) Modeling Technology Backgrounder Nov. 2005.
[12] K. P. Parker and E. J. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Trans. Comput.*, vol. C-24, no. 6, pp. 668–670, Jun. 1975.
[13] J. Monteiro, S. Devadas, A. Ghosh, K. Keutzer, and J. White, "Estimation of average switching activity in combinational logic circuits using symbolic simulation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 16, no. 1, pp. 121–127, Jan. 1997.
[14] J. C. Costa, J. C. Monteiro, and S. Devadas, "Switching activity estimation using limited depth reconvergent path analysis," in *Proc. Int. Symp.Low Power Electron. Des.* , Aug. 1997, pp. 18–20.
[15] J. H. Satyanarayana and K. K. Parhi, "Power estimation of digital data paths using HEAT," *IEEE Des. Test Comput.*, vol. 17, no. 2, pp. 101–110, Apr.–Jun. 2000.
[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, Ma: MIT Press, 2001.
[17] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Proc. ACM/SIGDA 6th Int. Symp. FPGA*, New York, Aug. 1998, pp. 191–200, ACM Press.
[18] E. Deprettere, P. Dewilde, and R. Udo, "Pipelined CORDIC architecture for fast VLSI filtering and array processing," in *IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 1984, pp. 250–253.
[19] S. Y. Park and N. I. Cho, "Fixed-point error analysis of CORDIC processor based on the variance propagation formula," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 3, pp. 573–584, Mar. 2004.

Yang Liu (S'05–M'09) received the B.S. and M.S. degrees in electrical engineering from Shanghai Jiaotong University, Shanghai, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from the Rensselaer Polytechnic Institute, Troy, NY, in 2008.

Currently, he is an ASIC Design Engineer with Juniper Networks, Inc. His research interests include very low-power VLSI architectures on variation-tolerant signal processing circuits, 3-D integrated signal processing systems, and high-performance communication and signal processing circuits.



Tong Zhang (M'02–SM'08) received the B.S. and M.S. degrees in electrical engineering from the Xian Jiaotong University, Xian, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 2002.

Currently, he is an Associate Professor with the Electrical, Computer And Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY. His current research interests include algorithm and architecture codesign for communication and data storage systems, variation-tolerant signal processing IC design, fault-tolerant system design for digital memory, and interconnect system design for hybrid CMOS/nanodevice electronic systems.

Dr. Zhang currently serves as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.

Keshab K. Parhi (S'85–M'88–SM'91–F'96) received the B.Tech., M.S.E.E., and Ph.D. degrees from the Indian Institute of Technology, Kharagpur, the University of Pennsylvania, Philadelphia, and the University of California, Berkeley, in 1982, 1984, and 1988, respectively.

He has been with the University of Minnesota, Minneapolis, since 1988, where he is currently a Distinguished McKnight University Professor with the Department of Electrical and Computer Engineering. He also holds the Edgar F. Johnson Professorship and is the Director of Graduate Studies of the Electrical Engineering program. His research addresses VLSI architecture design and implementation of physical layer aspects of broadband communications systems, error control coders and cryptography architectures, high-speed transceivers, and ultra wideband systems. He is currently working on intelligent classification of biomedical signals and images, for applications such as seizure prediction, lung sound analysis, and diabetic retinopathy screening. He has published over 450 papers, has authored the text book *VLSI Digital Signal Processing Systems* (Wiley, 1999) and coedited the reference book *Digital Signal Processing for Multimedia Systems* (Marcel Dekker, 1999).

Dr. Parhi was a recipient of numerous awards including the 2004 F.E. Terman award by the American Society of Engineering Education, the 2003 IEEE Kiyo Tomiyasu Technical Field Award, the 2001 IEEE W.R.G. Baker Prize Paper Award, and a Golden Jubilee Award from the IEEE Circuits and Systems Society in 1999. He has served on the editorial boards of the IEEE Transactions on Circuits and Systems, IEEE Transactions on Circuits and Systems II, VLSI Systems, Signal Processing, Signal Processing Letters, and Signal Processing Magazine, and served as the Editor-in-Chief of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS (2004-2005 term), and currently serves on the Editorial Board of the *Journal of VLSI Signal Processing*. He has served as technical program cochair of the 1995 IEEE VLSI Signal Processing workshop and the 1996 ASAP Conference, and as the general chair of the 2002 IEEE Workshop on Signal Processing Systems. He was a distinguished lecturer for the IEEE Circuits and Systems society during 1996–1998. He was an elected member of the Board of Governors of the IEEE Circuits and Systems society from 2005 to 2007.