Experimental analysis of sequence dependence on energy saving for error tolerant image processing

Se Hun Kim, Saibal Mukohopadhyay, and Wayne Wolf
Dept of ECE, Georgia Inst. of technology, *{sehunkim,saibal,wolf}@ece.gatech.edu*

ABSTRACT

We present experimental analysis to exploit the sequence dependence on energy saving in error tolerant image processing. Our analysis shows that the error distributions depend not only on combinational inputs but also on the previous state of the logic. We present a new sequential model for low-power delay faults. Our experimental results demonstrate the importance of considering the state of logic when analyzing errors and its dependence on output quality and energy saving. The results show that different input image type leads to very different output quality. This means that more error tolerant image can be operated at lower voltage. The sequence dependence in output quality and energy saving provide a new perspective to design low power multimedia system.

Categories and Subject Descriptors
B.2.2 [**Performance Analysis and Design Aids**]: Simulation

General Terms Experimentation, Design

Keywords voltage over-scaling, DCT, low power

1. INTRODUCTION

Scaling supply voltage is an efficient technique for low power/energy design since it results in quadratic reduction in dynamic power, as well as reduction in short circuit and leakage power. Conventionally, voltage scaling has been limited to $V_{dd\text{-}crit}$, which ensures correct functionality. Recently, aggressive voltage scaling which reduces supply voltage below $V_{dd\text{-}crit}$ has been considered for further reduction of power/energy dissipation [1]. Operation at voltage lower than $V_{dd\text{-}crit}$ leads to timing failure because of increased delay. The timing failure results in erroneous output and degrades quality of solution. However, the degraded quality may be acceptable for many DSP applications such as image processing and multimedia applications. This is because the quality is mostly evaluated by human perception, which masks small errors in images or sounds. For the reason, aggressive voltage scaling technique becomes attractive for the low power multimedia applications. For such applications, the voltage scalability is limited up to the level which guarantees to meet a certain output quality requirement.

Many previous works in this area tried to compensate the error due to the aggressive voltage scaling. Hedge et. al. proposed a design methodology, algorithmic noise-tolerance, to reduce the degradation in output quality using separate error control logic to compensate the errors [1]. Kurdahi et. al. presented a study of memory behavior under aggressive voltage scaling for DSP/communication applications and provided error control techniques [2]. These approaches try to meet the output quality requirement using additional logic to compensate the error while ensuring reduction in overall energy consumption of the system.

Research in these areas mostly ignored the interaction between input signal/image and voltage scalability of hardware. We provide different perspective to achieve the two contrary goals: output quality and energy saving, by focusing on the relationship between input signal/image and error rate. In the most error tolerant DSP systems, arithmetic units (i.e. adder and multiplier) are a major component that determines the system performance and energy consumption. Under aggressive voltage scaling, excitation of long delay path in arithmetic unit causes erroneous output as a result of timing failure. The error rate is directly related to frequency of the long delay paths excitation. Further, the excitation of the long delay path not only depends on the current input but also the previous input. The voltage scaling generally increase the delay of every path, but both input value and sequence to an arithmetic unit are primary factors to determine actual length of the delay path and the frequency of the excitation.

This paper presents a new sequential model using a finite state machine to obtain accurate delay fault and error rate. Based on the model and explanation, we emphasize that error distributions depend not only on combinational inputs but also on the previous state of the logic. The results of experiments demonstrate the importance of considering the sequence dependence on error rates and energy saving in image processing. Using various test images and conditions for JPEG encoding application, we show that different input images lead to very different output quality under voltage scaling. Based on the fact that the voltage level is determined by the output quality, we discuss the difference in the energy saving is possible if the input image types is considered during voltage scaling. Although the experimental results presented in this paper will vary for different adder architecture or different system, the general methodology in our analyses and experiments establishes fundamental understanding in the relationship between inputs to error tolerant systems and energy saving.

2. BACKGROUND

Our error analysis is based on a simple structure, which two addends from inputs registers are inserted to an *n*-bit ripple carry adder (see figure 1), and calculated value is captured at output registers. We set operating delay, $T_{oper} = 1/f_{clk}\text{-}T_{c\text{-}q}\text{-}T_{setup}$, as maximum delay of the adder without scaling supply voltage, thus

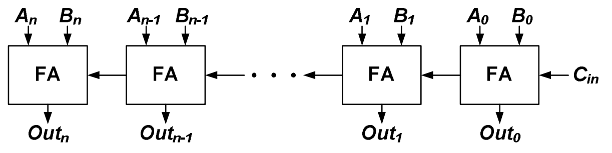


Figure 1: Ripple carry adder.

the adder has zero timing slack at nominal voltage. We define that an output bit has delay fault if the delay at that output bit is greater than the operating delay. Our analysis is based on the following assumption; the output register holds the previous output bit value at presence of delay fault. This assumption is valid when violating set-up time does not put the latch in the metastable condition and the register is not reset between successive additions. Note that the previous output bit value can be same with the correct output bit value. In this case, the output is correct even though the input set cause delay fault under voltage scaling. Therefore, output bit is erroneous only if the captured value is different from the correct value.

3. DELAY-ORIENTED FAULT ANAYSIS

The analysis of carry propagation in addition has been used for many prior works to model path delay [1][3][4]. However, carry propagation does not provide a true value for delay error for continued additions because it does not consider actual delay value based on transition at nodes. Hence, we instead use transition delay based model (we call it 'delay propagation') in order to obtain accurate delay value at each output bit.

3.1 Static path delay based estimation

In the carry propagation models based on ripple carry adder, a bit position generates a carry only if A_i=B_i=1 and propagates a carry if either A_i or B_i is equals to 1. The path delay is based on the longest length of carry chain, which carry '1' is transferred successively from lower order bit (LOB) to higher order bit (HOB). For example, addition of two number *x* and *y*, *x* = *11111111* and *y* = *00000001*, results in carry propagations from LSB to MSB. This longest carry propagation path is considered as a critical path. If *x* = *11101111* and *y* = *00000001*, the carry propagation stops at the 5th bit position and no carry propagates in more significant bit position.

3.2 Transition delay based estimation

For the delay propagation scheme, delay is generated whenever transitions of input cause transition at *CarryOut* and *Sum*. With respect to propagation of delay, we shall say that a full adder does not transfer the generated delay to the adjacent higher order full adder if two current input to a full adder *A* and *B* are equal since *CarryOut* is determined by only *A* and *B* without respect to *CarryIn*. Consequently, if A_i=B_i, *n*-bit ripple carry adder can be divided into independent subsequent parts that can operate in parallel. The number of consecutive '10' or '01' inputs determines the length of delay propagation.

Using an example shown in figure 2, let us explain the difference of two schemes and how the difference leads to very different delay results. When the 2nd combination is current operation, the carry propagation scheme considers it as maximum delay case as we explained in section 3.1. For delay propagation scheme, it is also considered as long delay propagation case (long consecutive '10' inputs), but the actual delay will be very small due to no

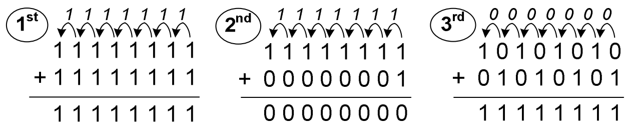


Figure 2: Sequences of 8-bit addition

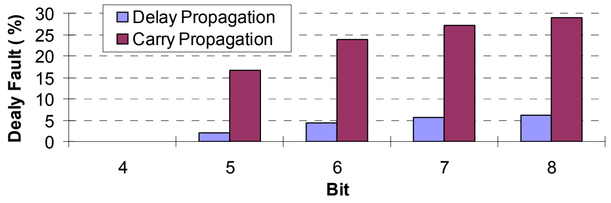


Figure 3: Delay fault comparison of delay propagation and carry propagation schemes (no delay faults at 0~4 bits)

transitions (from 1st to 2nd) at *CarryOut* of each full adder. When 3rd combination is current operation, the carry propagation scheme considers it as minimum delay case. However, it is considered as a long delay propagation case for the delay propagation scheme, and the actual delay may be large to cause delay faults since all *CarryOut* for each full adder make falling transition (from 2nd to 3rd). We also show the difference of two schemes in terms of delay fault result for 9-bit ripple carry adder using 10000 uniform random input combinations (see figure 3).

4. EVENT BASED FSM MODEL

In this section, using a finite state machine (FSM), we present general delay estimation model based on transition event, which is introduced in section 3. Since the delay generation depends on transition from previous to current input, the modeling of the delay can be expressed using transition of states. The each intermediate node (*CarryOut*) is considered as a state, and the input to the adder is the input to the FSM. The FSM shown in figure 4 is for the case that the value of *CarryIn* is 0. If *CarryIn* is 1, the completely opposite FSM is required. The output of the FSM is the transition event (whether the *CarryOut* changes from previous to current operation). The equations that describe functionality of a full adder are included in figure 4. Note that the propagation (*P*) and the value of *CarryOut* are determined by combinational logic, and only generation of delay at *CarryOut* is sequential logic. For example, if the previous operation (t-1th) has 1 for *CarryOut* and input *A* and *B* to a full adder is (0,1), then *CarryOut* at current operation (tth) become 0. Thus, there is a falling transition (T), which means delay is generated. The input combination propagates the generated delay from previous stage (FA_{i-1}) through the full adder (FA_i). Therefore, total delay to

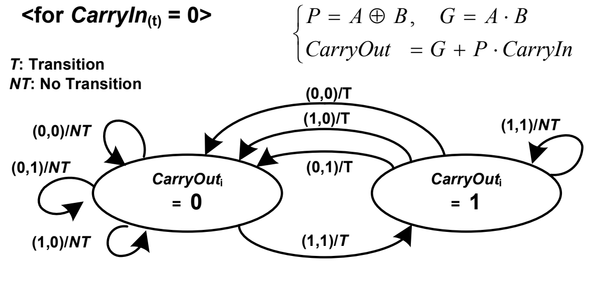


Figure 4: Event based finite state machine

CarryOut is the sum of the generated delay and the delay from the adjacent lower full adder. If *CarryOut* at t-1th operation is 0 state and input is (0,1), *CarryOut* at tth operation remains to 0. In this case, there is no transition (NT) and we do not need to concern propagation.

5. CHARACTERIZING INPUTS

As we discussed in section 3, the delay propagation length is determined by the number of consecutive '01' or '10' inputs to full adders. There are three cases that make long delay propagation length which has high probability to cause delay faults. Figure 5 shows the examples of the cases. First of all, we can intuitively figure out that an input set results in a long delay propagation length if one addend has mostly '0's and the other has mostly '1's. This occurs with (*case 1*) addition of two different signed numbers when the magnitudes of the both numbers are small. This also includes when one of addends is zero. In this case, the magnitude difference of two numbers is inversely proportional to the delay propagation length (e.g. see the difference between figure 5 (a) and (b)). The other case (*case 2*) is addition of two different signed numbers, but magnitudes of the two numbers are large and similar. Two close numbers have similar bit pattern, and the complement of one of them results in opposite bit pattern to the other. In this case, the magnitude difference of two numbers is inversely proportional to the delay propagation length (e.g. see the difference between figure 5 (c) and (d)). The last case (*case 3*) is the addition of two same signed numbers which magnitude difference is very large. The magnitude difference of two numbers is proportional to the delay propagation length in this case (e.g. see the difference between figure 5 (e) and (f)).

It is important to note that the final delay value at each output bit is frequently not large enough to make delay fault even though an input set makes long propagation length. Therefore, relationship between previous and current operation determines actual delay fault rate which is below the rate of long delay propagation length.

6. EXPERIMENT FRAMEWORK

6.1 Simulation setup for adder

We simulated 1-bit full adder (28-T full adder design [5]) with the load of three minimum size inverter using HSpice and PTM 70nm technology. The simulation includes delay and energy consumption of the full adder for all possible input transitions and different voltage levels. We set the voltage scaling factor (κ) in the range of 0.8 ~ 1 (i.e. 0.95V~ 1.2V). A circuit delay calculation based on data dependent gate delays has been proposed in [6][7]. However, gate level delay modeling requires tremendous

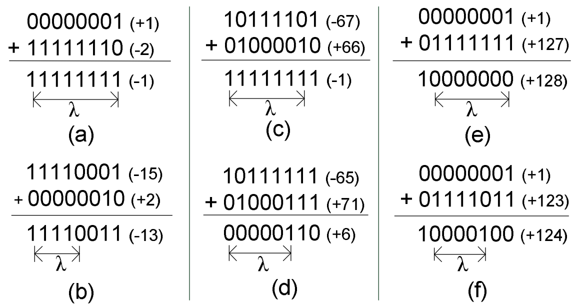
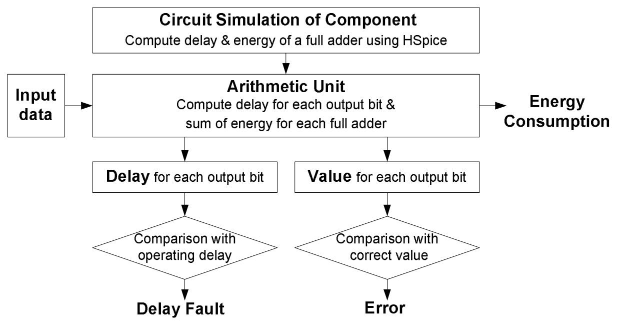


Figure 5: Illustrative examples using 8-bit adder (λ: delay prop. length) for *case 1*: (a), (b), *case 2*: (c), (d), *case 3*: (e), (f)



Circuit Simulation of Component
Compute delay & energy of a full adder using HSpice
Input data
Arithmetic Unit
Compute delay for each output bit & sum of energy for each full adder
Energy Consumption
Delay for each output bit
Value for each output bit
Comparison with operating delay
Comparison with correct value
Delay Fault
Error

Figure 6: Behavioral C simulation

simulation time if the size of circuit is large. Moreover, it is not efficient to examine the timing failure since we only concern the delay at output. Thus, we consider a full adder as smallest component. Three inputs of a full adder make 64 possible input transitions and corresponding delays to each output.

Characterization of delay fault and error behavior of the ripple carry adder was performed through behavioral C simulations based on the HSpice data. This is summarized in the figure 6. The final delay calculation for each output bit is based on the transition delay based estimation explained in section 3. If the calculated final delay value is greater than predefined operating delay, it is counted as a delay fault at the output bit. At the presence of delay fault, the output bit value is set to previous output bit value and compared with correct bit value to determine if the bit position is erroneous or not. This process is repeated for all *n* full adders. In addition, we obtained energy consumptions of the adder for each input set based on the HSpice energy estimation result of each full adder for all 64 input transitions.

6.2 DCT in JPEG

JPEG is the one of the most popular image compression standards. Its lossy compression concept is based on the error tolerance characteristic that human eyes mask the lost in some visual information. In the flow steps for JPEG compression of grayscale images (see figure 7), the 2-D DCT transforms the reformatted data to frequency domain. The 2-D DCT is very computational intensive process. Our analysis is based on the algorithm presented in [8] and the pipelined architecture described in [9]. Excluding multiplication stage, total 10 adders are required for 2-D DCT. Each stage performs maximum eight addition operations. We assume that 15-bit ripple carry adders are used for all stages.

We integrate our simulation tool into the JPEG program included in MediaBench [10]. The modified program performs delay computation for all addition processes in 2-D DCT and then generates erroneous values based on the computed delay results. To compare the quality of images, we used the Mean Structural SIMilarity (MSSIM) index proposed in [11]. MSSIM is expected to be better in assessing perceptual image quality compared to conventional metric, MSE and PSNR. For two visually identical images, MSSIM is 1. Quality degradation of an image results in smaller MSSIM index.

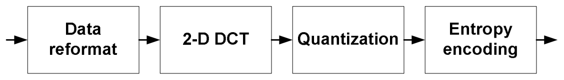


Figure 7: Overview of JPEG encoding

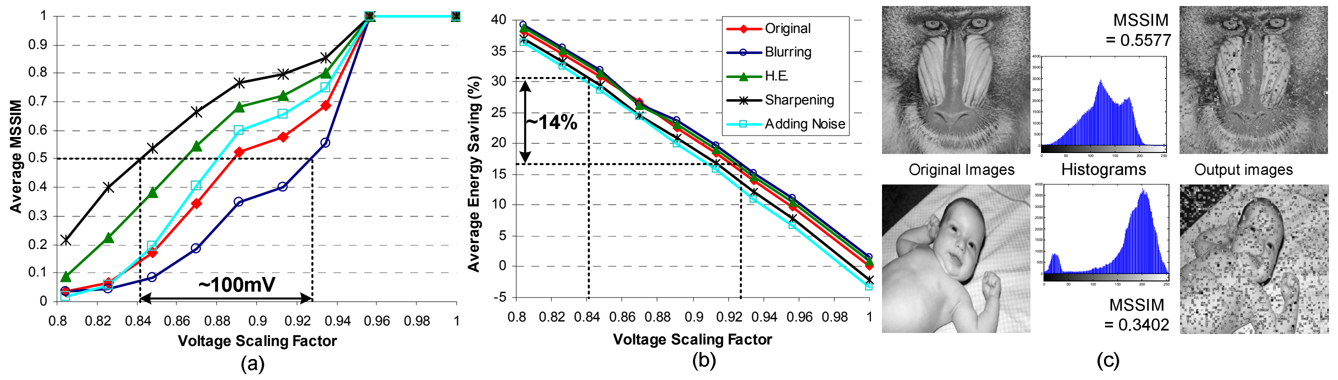


Figure 8: (a) Average MSSIM, (b) Average energy saving in % (c) Comparison of output quality and histogram

7. RESULT AND DISCUSSION

To analyze the dependencies of input (image) to error rate under scaled voltages, we apply common image processing techniques to excite or diminish the conditions that increase delay fault. We consider blurring, sharpening, histogram equalization, and adding white noise. Blurring attenuates high frequency contents, on the contrary adding noise, histogram equalization, and sharpening boost high frequency contents. Matlab functions are used for these processes. These image processing techniques change input distribution, which cause adders in DCT exercise different parts of the input space more frequently.

By observing input type during simulations, we found that the *case 1* was strongly dominant (more than 98%) compared to the others. Therefore, we only focus on the case. As we discussed in section 5, if the addends for additions are small magnitude values or close values, then the probability of error is higher than other input set. These two conditions are related to each other. The addition in DCT algorithm is based on butterfly architectures (i.e. basic butterfly can be expressed as $y_o = x_0 + x_1$, $y_1 = x_0 - x_1$). For such operations, if neighboring pixel values are close, subtractions of the pixel values result in output values with small magnitude. The additions of these small magnitude numbers increase the probability to have long delay propagation length. Blurring process makes neighboring pixels close each other since it assigns each pixel's new value with the average value of the neighboring pixel values. On the other hand, sharpening, histogram equalization, and adding noise to the images lower the probability of error. After increasing information in high frequency components, the differences between neighboring pixel values become larger. Thus, probability of getting small magnitude values or close values becomes lower.

The generated errors at each adder may propagate to next stages or be concealed during quantization process. Though we do not provide detail analysis of the error propagation in this paper, it is obvious that higher error rate for each adder will result in higher errors at final output, which is compressed images for JPEG encoding. Figure 8 (a) shows average MSSIM of the 25 test images for each image type. The results demonstrate the quality of output image highly depends on input image type. In other words, images with more information in high frequency components are more error tolerant under voltage over-scaling. The average energy saving is based on the average energy consumption with nominal voltage. Compared to image degradation at constant voltage, the variation in energy saving for the different image types is trivial (less than 5%, see figure 8 (b)). Therefore, energy saving is primarily dependent on voltage scalability for a certain output quality requirement. For instance, if minimum output image quality is 0.5 in MSSIM and energy saving can be varied by about 14% for different image types. Figure 8 (c) shows the output quality difference of two example images with same scaled voltage.

8. CONCLUSION

This paper provides experimental analysis to exhibit the impact of sequence dependence on energy saving when aggressive voltage scaling is applied to error tolerant DSP applications. Based on experiment results using JPEG encoding, voltage scalability dependent on output quality is varied to different input image type. With analysis of relation between inputs to an arithmetic unit and error rate under voltage scaling, we provide experimental results and discussion for the relationship between voltage scalability and image type. We believe that this sequence dependence on energy saving provides a new perspective for low energy system design.

9. REFERENCES

[1] R. Hegde et. al., "Soft digital signal processing," TVLSI, vol.9, pp. 813-823, Dec. 2001.

[2] F. Kurdahi et. al., "Error-Aware Design," *DSD*, pp. 8-15, 2007.

[3] L. N. B. Charkrapani et. al., "Highly Energy and Performance Efficient Embedded Computing through Approximately Correct Arithmetic," *CASES*, pp. 187-196, 2008.

[4] S. Ghosh et. al., "CRISTA: A New paradigm for Low-Power Variation-Tolerant, and Adaptive Circuit Synthesis Using Critical Path Isolation," *TCAD*, pp.1947-1956, 2007.

[5] K.S Yeo, K. Roy, *Low-Voltage, Low-Power VLSI Subsystems*, McGraw-Hill, New York, 2005.

[6] C.T. Gary et. al., "Circuit delay calculation considering data dependent delays," *Integration, the VLSI Journal,* pp.1-23, 1994.

[7] S. Sun, et. al., "Efficient timing analysis for CMOS circuits considering data dependent delays," *TCAD*, pp. 546-552, 1998.

[8] Y. Arai et. al., "A fast DCT-SQ Scheme for Images," *Transaction of IEICE*, vol. E71, pp. 1095-1097.

[9] L.V. Agostini et. al., "Pipelined fast 2D DCT architecture for JPEG image compression," *Symposium on Integrated Circuits and Systems Design*, pp. 226-231, 2001.

[10] MediaBench, Available: http://euler.slu.edu/~fritts/mediabench.

[11] Z. Wang et. al., "Image Quality Assessment: From Error Measurement to Structural Similiarity," *IEEE Transactions on Image Processing*, vol.13, pp.600-612, 2004.