Overclocking Datapath for Latency-Error Tradeoff

Kan Shi, David Boland, George A. Constantinides
Department of Electrical and Electronic Engineering
Imperial College London, UK
Email: {k.shi11, david.boland03, g.constantinides}@imperial.ac.uk

***Abstract*— Relaxing constraints of 100% accuracy in datapath can provide the freedom to create designs with better performance or energy efficiency. This paper develops probabilistic models, which enable us to explore these trade-offs for key arithmetic primitives. We show that because specific input patterns are required to cause timing violations and that these patterns arise rarely, a lower expected error can be attained by allowing some timing variations to occur, instead of reducing the precision of a circuit to meet a target latency. Experiments show that a mean reduction of $5.6\times \sim 36.7\times$ in error expectation and an improvement of $7.2dB \sim 19.7dB$ in signal-to-noise ratio can be obtained for practical applications.**

I. Introduction

The continuous scaling of CMOS technology places progressively stringent requirements on circuit performance. Recently we have seen that tuning a circuit to meet a desired accuracy criteria can help to meet these demands, because in general it is rarely to create a completely error free design as any fixed or floating point representation introduces quantisation errors. Since a simplified number representation can lead to performance benefits, there has been significant recent research into the area of optimising the precision throughout a datapath to meet a specific accuracy requirement [1].

However, the choice of precision is not the only source of error in a datapath, and parallel streams of research have explored alternative methods to trade accuracy for performance or energy efficiency. Ernst et. al. introduced a voltage overscaling technique which monitored the error rate using additional circuitry when overscaling the supply voltage [2]. This work demonstrated that greater power efficiency could be obtained because the errors rarely occurred, in which case the voltage could be overscaled. Similarly, a non-uniform voltage scaling technique was discussed in [3], which analysed the link between probability of output correctness and energy saving. While the voltage scaling literature takes advantage of the fact that only specific input patterns could cause timing errors, research on imprecise architectures take this one step further by taking advantage of errors only occurring with specific input patterns to design a simplified circuit. This includes a simplified multiplier unit [4], whilst the authors in [5] discussed the link between accuracy and clock frequency for approximation circuits which employed a simplified datapath to mimic and speculate the original logic functions. Alternatively, an adder architecture was described in [6] which provided the flexibility to trade accuracy for energy by providing the option to only utilise part of the adder or enable additional circuitry to correct for errors for high accuracy requirements.

In this paper, we attempt to bring these two strands of research together, by evaluating the probabilistic behavior of basic arithmetic primitives with different datapath precisions, when operating beyond the deterministic region. Although pipelining could be used to increase operating frequency, this technique is typically employed to meet throughput constraints and does not reduce circuit latency. Therefore we use truncation as a more appropriate comparison metric to our proposed alternative under the latency constraint environment. We initially present probabilistic models of errors generated in this process, and subsequently we test this design methodology on real DSP examples. Both our models and experimental results demonstrate that performance benefits can be achieved in comparison to the traditional situation where the target latency is limited by choice of precision.

The contributions of this paper are:

- the first combination of datapath function with overclocking in a holistic framework to trade accuracy for latency,
- probabilistic models of errors resulting from overclocking arithmetic primitives under different frequencies,
- analytical and empirical results which demonstrate that allowing rare timing violations to occur results in less error than truncating a datapath to meet timing.

II. Ripple Carry Adder

A. Generation of Overclocking Error

Adders serve as a key building block for arithmetic operations, and other major arithmetic operators such as multipliers can be implemented using adders. Since the ripple carry adder (RCA) is the most widely used one among numerous adder structures, particularly in FPGA technology, the philosophy of our approach is exemplified with the analysis of a RCA. In this work, we analyse the errors originating from two scenarios. The first is a traditional circuit design approach where operations occur without timing violations. To this end, the word-length of the input signal is truncated in order to meet the timing requirement. This process results in truncation or roundoff error. In our proposed new scenario, circuits are implemented with greater word-length, but are clocked beyond the safe region so that timing violations sometimes occur. This process generates "overclocking error".

An n-bit RCA is composed of n serial-connected full adders (FAs) as shown in Fig. 1. Typically the critical path delay of RCA (μ_{RCA}) is determined by the longest carry propagation. We assume that carry propagation delay ($\mu_i,\ 0 \leq i < n$) of each FA is a value μ, and hence $\mu_{RCA} = n\mu$.

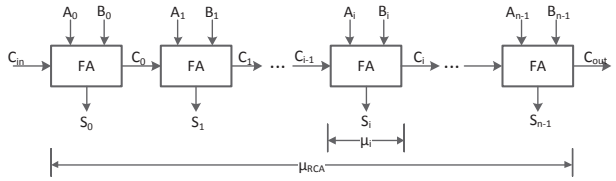


Fig. 1. Ripple Carry Adder.

For an n-bit RCA, if the sampling period T_S is greater than μ_{RCA}, correct results will be sampled. If, however, $T_S < \mu_{RCA}$, intermediate results will be sampled, potentially generating overclocking error. For a given T_S, the maximum length of error-free carry propagation is described by (1) where f_S denotes the sampling frequency.

$$b := \left\lceil \frac{T_S}{\mu} \right\rceil = \left\lceil \frac{1}{\mu \cdot f_S} \right\rceil \tag{1}$$

However, since the length of an actual carry chain during execution is dependent upon input patterns, in general, the worst case may occur rarely. As a result, in order to determine when this timing constraint is not met, and the size of the error in this case, we make use of standard results which decide carry generation, propagation and annihilation, as well as corresponding summation results [7]:

- If $A_i = B_i = 1$, carry chain is generated at bit i, $S_i = C_{i-1}$;
- If $A_i \neq B_i$, carry propagates for this chain at bit i, $S_i = 0$;
- If $A_i = B_i$, current carry chain annihilates at bit i, $S_i = 1$.

We then model the errors assuming that all bits in A and B are mutually independent and uniformly distributed. However we relax this assumption in Section IV where the predictions are verified using real data.

1) Absolute Value of Overclocking Error: For an n-bit RCA, let C_{tm} denote the carry chain generated at bit S_t with the length of m bits. For a certain f_S, the maximum length of error-free carry propagation, b, is determined through (1). The presence of overclocking error requires $m > b$. Since the length of carry chain cannot be greater than n, parameters t and m are bounded by (2) and (3):

$$0 \leq t \leq n - b \tag{2}$$

$$b < m \leq n + 1 - t \tag{3}$$

For C_{tm}, correct results will be generated from bit S_t to bit S_{t+b-1}. Hence the absolute value of error seen at the output, normalized to the MSB (2^n), is given by (4), where $\hat{S}_i$ and S_i denote the actual and error-free output of bit i respectively.

$$e_{tm} = \frac{\left| \sum_{i=t+b}^{n} (S_i - \hat{S}_i) \cdot 2^i \right|}{2^n} \tag{4}$$

We can calculate S_i and $\hat{S}_i$ using the equations from the previous discussion. In the error-free case, the carry will propagate from bit S_t to bit S_{t+m-1}, and we know that $S_{t+b} = S_{t+b+1} = \cdots = S_{t+m-2} = 0$ for carry propagation, and $S_{t+m-1} = 1$ for carry annihilation. However, when a timing violation occurs, carry will not propagate through all these bits, instead, $\hat{S}_{t+b} = \hat{S}_{t+b+1} = \cdots = \hat{S}_{t+m-2} = 1$ and $\hat{S}_{t+m-1} = 0$. Note that we assume all internal bits are set to 0 initially. Substituting these values into (4) yields (5), from where interestingly we see that the value of overclocking error has no dependence on the length of carry chain m.

$$e_{tm} = \left| 2^{t+m-1} - 2^{t+m-2} - \cdots - 2^{t+b} \right| / 2^n = 2^{t+b-n} \tag{5}$$

2) Probability of Overclocking Error: The carry chain C_{tm} occurs when there is a carry generating at bit t, a carry annihilating at bit $t+m-1$ and carry propagates in between. Consequently, its probability P_{tm} is given by (6). Altogether, under the assumption that A and B are mutually independent and uniformly distributed, we have $P_{(A_i=B_i=1)} = 1/4$, $P_{(A_i \neq B_i)} = 1/2$ and $P_{(A_i=B_i)} = 1/2$, P_{tm} can be obtained by (7). Note that (7) takes into account the carry annihilation always occurs when $t + m - 1 = n$.

$$P_{tm} = P_{(A_t=B_t=1)} P_{(A_{t+m-1}=B_{t+m-1})} \cdot \prod_{i=t+1}^{t+m-2} P_{(A_i \neq B_i)} \tag{6}$$

$$P_{tm} = \begin{cases} (1/2)^{m+1} & \text{if } t+m-1 < n \\ (1/2)^m & \text{if } t+m-1 = n \end{cases} \tag{7}$$

3) Expectation of Overclocking Error: Expectation of overclocking error can be expressed by (8). Using P_{tm} and e_{tm} from (5) and (7) respectively, E_O can be obtained by (9).

$$E_O = \sum_t \sum_m P_{tm} \cdot e_{tm} \tag{8}$$

$$E_O = \begin{cases} 2^{-b} - 2^{-n-1}, & \text{if } b \leq n \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

B. Probabilistic Model of Truncation Error

If the input signal of a circuit is k bits, truncation error occurs when the input signal is truncated from k bits to n bits. Under this premise, the mean value of the truncated bits at signal input (E_{Tin}) is given by (10).

$$E_{Tin} = \frac{1}{2} \sum_{i=n+1}^{k} 2^{-i} = 2^{-n-1} - 2^{-k-1} \tag{10}$$

Since we assume there are two mutually independent inputs to the RCA, the overall expectation of truncation error for the RCA is given by (11).

$$E_T = \begin{cases} 2^{-n} - 2^{-k}, & \text{if } n < k \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

C. Comparison between Two Scenarios

In the traditional scenario, for a given f_S, the word-length of RCA must be truncated to $n = b - 1$ in order to meet the required timing constraint. The error expectation of the traditional scenario is then given by (12).

$$E_{trad} = 2^{-b+1} - 2^{-k} \tag{12}$$

In the new scenario, we allow overclocking errors, so set the word-length of RCA to be equal to the input word-length k, giving (13) according to (9).

$$E_{new} = 2^{-b} - 2^{-k-1} \tag{13}$$

Comparing Eq.(13) and Eq.(12), we have (14). This equation indicates that by allowing timing violations, the overall error expectation of RCA outputs drops by a factor of 2 in comparison to traditional scenario. This provides the first hint that our approach is useful in practice.

$$\frac{E_{new}}{E_{trad}} = \frac{2^{-b} - 2^{-k-1}}{2^{-b+1} - 2^{-k}} = \frac{1}{2} \tag{14}$$

III. Constant Coefficient Multiplier

Another key primitive of arithmetic operations, the constant coefficient multiplier (CCM), can be implemented using RCA and shifters. For example, operation $B = 9A$ is equivalent to $B = A + 8A = A + (A << 3)$, which can be built using one RCA and one shifter. We will focus on this single RCA and single shifter structure in the rest of this paper, since complex structures consisting of multiple RCAs and multiple shifters can be built in accordance with this baseline structure.

In this CCM structure, let the two inputs of the RCA be denoted by A_S and A_O respectively, which are both two's complement numbers. A_S denotes "shifted signal", with zeros padded after LSB, while A_O denotes "original signal" with MSB sign extension. For an n-bit input signal, it should be noted that an n-bit RCA is sufficient for this operation, because no carry will be generated or propagated when adding with zeros, as shown in Fig. 2.

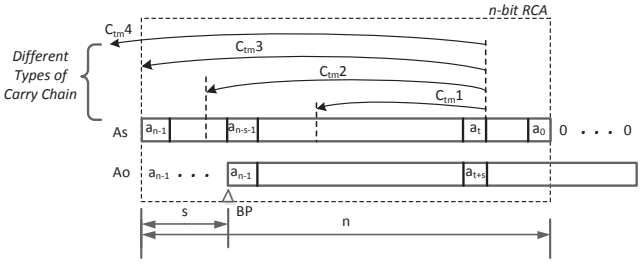


Fig. 2. Different Types of Carry Chain in Constant Coefficient Multiplier.

A. Probabilistic Model of Overclocking Error

1) Absolute Value of Overclocking Error: For a CCM, the absolute value of overclocking error of carry chain C_{tm} is increased by a factor of 2^s due to shifting, compared to RCA. Therefore e_{tm} in CCM can be modified from (5) to give (15).

$$e_{tm} = 2^{t+b-n+s} \tag{15}$$

2) Probability of Overclocking Error: Due to the dependencies in a CCM, carry generation requires $a_t = a_{t-s} = 1$, propagation and annihilation of a carry chain is best considered separately for four types of carry chain generated at bit t. We label these by $C_{tm}1$ to $C_{tm}4$ in Fig. 2, defined by the end region of the carry chain. For $C_{tm}1$, we have:

- Carry propagation: $a_i \neq a_{i+s}$, where $i \in [t+1, n-s-2]$;
- Carry annihilation: $a_j = a_{j+s}$, where $j \in [t+1, n-s-1]$.

Similarly for $C_{tm}2$, we have:

- Carry propagation: $a_i \neq a_{n-1}$, where $i \in [n-s-1, n-3]$; or $a_i \neq a_{i+s}$, where $i \in [t+1, n-s-2]$;
- Carry annihilation: $a_j = a_{n-1}$, where $j \in [n-s-1, n-2]$.

If all bits of input signal are mutually independent, then the probability of carry propagation and annihilation for $C_{tm}1$ and $C_{tm}2$ is $1/2$, and the probability of carry generation is $1/4$. If we substitute this into (6), we obtain (16).

$$P_{tm} = (1/2)^{m+1}, \quad \text{if } t+m-1 \leqslant n-2 \tag{16}$$

For carry annihilation of $C_{tm}3$, $a_{n-1} = a_{n-1}$, which is always true. Thus the probability of $C_{tm}3$ is given by (17).

$$P_{tm} = (1/2)^{m}, \quad \text{if } t+m-1 = n-1 \tag{17}$$

$C_{tm}4$ represents carry chain annihilates over a_{n-1}, therefore carry propagation requires $a_{n-1} \neq a_{n-1}$. This means $C_{tm}4$ never occurs in a CCM. Altogether, P_{tm} for a CCM is given by (18).

$$P_{tm} = \begin{cases} (1/2)^{m+1} & \text{if } t+m-1 < n-1 \\ (1/2)^{m} & \text{if } t+m-1 = n-1 \end{cases} \tag{18}$$

3) Expectation of Overclocking Error: For a CCM, since the carry chain will not propagate over a_{n-1}, the upper bound of parameter t and m should be modified from (2) and (3) to give (19) and (20).

$$0 \leqslant t \leqslant n-b-1 \tag{19}$$

$$b < m \leqslant n-t \tag{20}$$

Finally, by substituting (18) and (15) with modified bounds of t and m into (8), we obtain the expectation of overclocking error for a CCM to be given by (21).

$$E_O = \begin{cases} 2^{s-b-1} - 2^{s-n-1}, & \text{if } b \leq n-1 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

B. Probabilistic Model of Truncation Error

In order to meet the target frequency, we assume the input signal is truncated before entering CCM. Let E_{Tin} and E_{Tout} denote the expectation of truncation error at the input and output of CCM respectively, then we have (22), where c denotes the coefficient of CCM.

$$E_{Tout} = |c| \cdot E_{Tin} \tag{22}$$

IV. Case Study: FIR Filter

A. Experimental Setup and Model Verification

The benefits of the proposed methodology are demonstrated by using an FIR filter, as shown in Fig. 3. The results are obtained through timing simulations. In order to achieve the desired latency between input and output, the word-length of the input signal is truncated through the quantizer (Q) in the traditional scenario. However, in our proposed new scenario, the operating frequency is over-scaled while maintaining the original input word-length. We explore the best trade-off between latency and error based on these two scenarios. Two types of input signal are employed in our experiments: 8-bit data sampled from a uniform distribution, which we refer to as "uniform independent inputs", and "real inputs", which denote 8-bit pixel values of several 512×512 images.

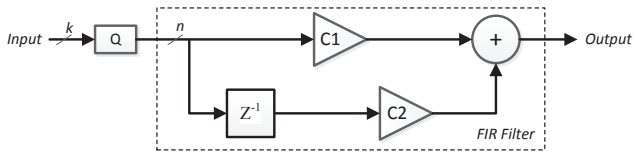


Fig. 3. A Simple FIR Filter.

B. Expectation of Error

We first assess the accuracy of our proposed models of error. The amount of input signal truncation for the traditional scenario varies from 1 bits to 7 bits. When the circuit is truncated, it allows the circuit to operate at a higher frequency than the rated frequency, up to $2.75\times$, which corresponds to the maximum frequency required for 1-bit input word-length

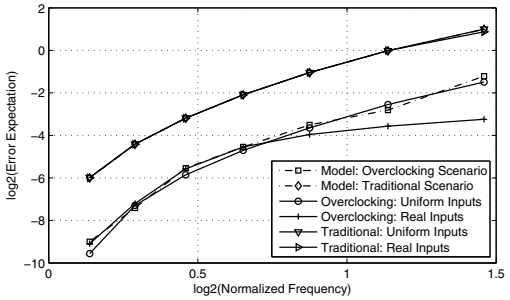
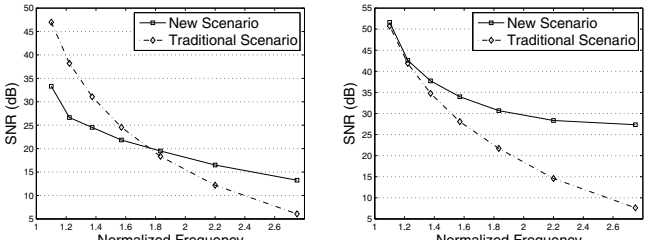


Fig. 4. Verification of Proposed Model.



(a) Uniform Independent Inputs (b) Real Inputs (Tiffany Image)

Fig. 5. Comparison between New Scenario and Traditional Scenario: SNR.

according to our experiments. Results in Fig. 4 demonstrate that our models for both overclocking error and truncation error match well with the uniform case. However, we observe a small deviation when using real inputs, since the real data does not exactly follow the uniform distribution or the independent assumption. In Fig. 4 the real inputs are the results for the "Lena" benchmark image.

Table I summarizes the experimental results obtained by uniform independent inputs together with 4 benchmark images. For all input types, error expectation is reduced in the new scenario, as expected by our model, with the geometric mean of reduction varying from $5.6\times$ to $36.7\times$. In addition, we see that in practice, larger differences of error expectation are achieved. This is because for real data, long carry chains typically occur with even smaller probabilities.

TABLE I
RATIO OF ERROR EXPECTATION: TRADITIONAL OVER NEW.

Inputs	**Normalized Frequency**							**Geo.**
	1.10	1.22	1.38	1.57	1.83	2.20	2.75	**Mean**
Uniform	7.11	4.17	5.06	5.35	5.86	5.92	5.83	5.55
Lena	8.55	6.93	5.20	5.45	7.54	11.74	17.30	8.24
Sailboat	35.31	15.19	11.41	9.70	9.39	10.92	11.69	13.24
Pepper	17.91	10.79	8.84	7.67	8.13	10.39	16.49	10.90
Tiffany	204.5	50.52	28.78	21.94	18.57	22.14	33.69	36.74

C. Signal-to-Noise Ratio

Fig. 5 demonstrates SNR for uniform independent inputs and the "Tiffany" image with increasing frequencies. For the former input type, higher SNR is obtained in the traditional scenario where f_s is increased to $1.1\times \sim 1.6\times$ of rated frequency, while the new scenario outperforms when f_S is increased to over $1.8\times$ of rated frequency. This is because SNR is inversely proportional to square of the error. In the traditional scenario, small f_S corresponds to limited truncation of LSBs, which in turn lead to small noise power. In the new scenario, overclocking error is generated in the MSBs, therefore large noise power is expected at the early stage of overclocking. However, as f_S is further increased, the corresponding amount of truncation rises, resulting in smaller SNR compared to the new scenario.

For real inputs, SNR can be higher for all frequencies under the new scenario, since long carry chains occur rarely. This information could be used to achieve better performance in the new scenario. For example, suppose a user only require an SNR of 30dB or less. In this case, one could operate at a frequency of $2.1\times$ over rated frequency under the new scenario but only $1.5\times$ under the traditional scenario.

Table II presents the differences of SNR (dB) obtained by the two input types. Similar to error expectation, we also see that the improvement of SNR is higher for real image data than uniform data.

TABLE II
DIFFERENCE OF SNR: NEW OVER TRADITIONAL.

Inputs	**Normalized Frequency**						
	1.10	1.22	1.38	1.57	1.83	2.20	2.75
Uniform	-13.68	-11.57	-6.56	-2.72	1.14	4.30	7.18
Lena	-12.90	-9.20	-6.78	-2.71	2.83	9.05	14.41
Sailboat	-6.75	-5.11	-2.06	1.04	4.56	8.76	11.33
Pepper	-9.71	-6.99	-3.71	-0.56	3.46	8.29	14.67
Tiffany	0.89	0.70	2.96	5.90	8.97	13.75	19.71

V. CONCLUSION

This paper has explored the probabilistic behavior of key arithmetic primitives when allowing timing violations. We have developed models for errors generated due to both overclocking and truncation of inputs. These models indicate that it may be preferable to allow timing violations to occur, under the knowledge that they will only occur rarely. We support this hypothesis with experiments that demonstrate a geometric mean reduction of error expectation of $5.6\times \sim 36.7\times$, a maximum improvement of SNR of $7.2dB \sim 19.7dB$ can be achieved in real applications over the conventional scenario. This information can in turn be used to achieve better performance for a given budget of error or SNR.

ACKNOWLEDGMENT

This work is supported by the EPSRC (Grants EP/I020557/1 and EP/I012036/1).

REFERENCES

[1] G. Constantinides, N. Nicolici, and A. Kinsman, "Numerical data representations for FPGA-based scientific computing," *IEEE Design Test of Computers*, vol. 28, no. 4, pp. 8 –17, 2011.

[2] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *IEEE/ACM Int. Symp. on Microarchitecture*, 2003, pp. 7–18.

[3] Z. Kedem, V. Mooney, K. Muntimadugu, and K. Palem, "An approach to energy-error tradeoffs in approximate ripple carry adders," in *Int. Symp. on Low Power Electronics and Design*, 2011, pp. 211 –216.

[4] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in *Int. Conf. on VLSI Design*, 2011, pp. 346–351.

[5] S. Lu, "Speeding up processing with approximation circuits," *IEEE Computer*, vol. 37, no. 3, pp. 67–73, 2004.

[6] A. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," *Proc. Design Automation Conf.*, pp. 820–825, 2012.

[7] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital integrated circuits: a design perspective (2nd edition)*. Prentice-Hall, 2003.