CASA: Correlation-Aware Speculative Adders

Gai Liu, Ye Tao, Mingxing Tan, and Zhiru Zhang

Computer Systems Laboratory, Electrical and Computer Engineering
Cornell University, Ithaca, NY
{gl387, yt434, mingxing.tan, zhiruz}@cornell.edu

ABSTRACT

Speculative adders divide addition into subgroups and execute them in parallel for higher execution speed and energy efficiency, but at the risk of generating incorrect results. In this paper, we propose a lightweight correlation-aware speculative addition (CASA) method, which exploits the correlation between input data and carry-in values observed in real-life benchmarks to improve the accuracy of speculative adders. Experimental results show that applying the CASA method leads to a significant reduction in error rate with only marginal overhead in timing, area, and power consumption.

1. INTRODUCTION

Opportunistic computing [3] is an emerging design paradigm to improve the performance and energy efficiency of digital computer systems. By allowing inaccurate or occasionally incorrect results to occur, designers are offered a choice to more easily trade the quality of solution for cost. The central idea of opportunistic computing is to speed up the common-case execution while allowing errors to happen in rare occasions. In applications such as signal processing [8, 7] where inexact results are tolerable, opportunistic computing techniques can improve both speed and energy efficiency. On the other hand, when exact results are desired, appropriate error detection and recovery mechanisms are necessary to correct the errors.

Being one of the essential building blocks of the digital circuits, adders have been recently re-examined and "retooled" in the context of opportunistic computing. The main idea is to exploit the fact that the typical length of the carry propagation is usually much shorter than the full width of the adder. Therefore, an opportunistic adder built with short carry chains can potentially operate at a much faster speed with reasonably accurate results. Broadly speaking, opportunistic adders can be categorized into two classes, namely, approximate adders and speculative adders. Approximate adders [4, 17, 12] are more concerned with minimizing the error amplitude to ensure the accuracy of the output value. Speculative adders [11, 13, 10, 14, 16, 15, 2, 9], on the other hand, primarily aim to improve the speed of the addition by making speculations about the carry-in values. When a misspeculation results in an incorrect output, the error recovery circuit will be triggered to correct the result. Since the error recovery logic usually incurs considerable overhead in latency and energy consumption, a speculative adder is expected to operate with a low error rate (i.e., the number of errors divided by the total number of additions).

ISLPED'14, August 11–13, 2014, La Jolla, CA, USA.

http://dx.doi.org/10.1145/2627369.2627635.

In this work, we focus on speculative adders considering that they are applicable to a wider range of problems. Clearly, minimizing the error rate without significant timing/area/power degradation is one of the key challenges for designing an effective speculative adder. Although a variety of techniques [10, 16, 15, 2] have been proposed to address this issue, the existing speculative adders are mostly tested against uniformly distributed random vectors, which often do not reflect the actual data patterns from the real-life benchmarks. In fact, according to our experiments on MiBench [5], two representative speculative adders exhibit high error rates (above 30%) for multiple benchmarks, suggesting that random test vectors are insufficient for evaluating the effectiveness of existing and new speculation techniques.

In this paper, we address the problem of reducing error rate of speculative adders in the context of real-life applications. Our method intelligently exploits the correlation between the most significant bit of the input operands and the carry-in values to improve the "correctness" of speculative adders. Our major contributions are as follows:

1. We present a systematic study using real-life benchmarks to evaluate the effectiveness of state-of-the-art speculative adders. We show that significant room exists for reducing the error rates of existing techniques.

2. We propose the correlation-aware speculative addition (CASA), which is a generic lightweight extension to existing speculative adders. We show that CASA achieves a significant reduction in error rate with small overhead in timing and area. Compared with a fast parallel prefix adder, CASA can also achieve substantial power reduction with comparable speed.

The rest of the paper is structured as follows: Section 2 reviews the related work on speculative adder designs; Section 3 provides background and preliminaries of the operating principles of the speculative addition; Section 4 presents the quantitative study of two existing speculative adders and motivates our own method; Section 5 present our correlation-aware speculative addition approach; Section 6 reports the experimental results followed by conclusions in Section 7.

2. RELATED WORK

One of the early attempts to build a speculative adder is described in [11] where a long addition is divided into smaller groups of length K, with each group responsible for generating the result only for the most significant bit in that group adder. Since each group adder only handles a window

size of K bits, an error would occur when the actual length of the carry propagation exceeds K. Variable latency speculative adder (VLSA) [13] employs a similar scheme formed by K-bit group adders and further proposes resource sharing techniques to significantly reduce the overall circuit area. Error detection and recovery circuits are also introduced in this work.

In the error-tolerant adder (ETA) [16, 15], a different design is proposed where the original adder is divided into several non-overlapping groups and each group also has a separate carry generation logic. The carry-in value for a group adder depends on the result of the carry generation from the preceding group adder. A different method of carry-in prediction is used in the speculative carry select adder (SCSA) [2], which pre-computes both cases with and without the carry-in. The carry-out value from the preceding group will be used to determine the actual outcome. Although this method can achieve higher speed and accuracy, pre-computations are quite expensive in terms of both area and energy. Accuracy configurable adder (ACA) [10] only uses the result of upper half part of each addition so as to improve accuracy. They also propose a pipelined accuracy-reconfigurable scheme where results are incrementally corrected in different pipeline stages. Since error amplitude of ACA can be limited within certain range by the incremental correction scheme, ACA can also be viewed as a hybrid adder combining the feature of approximate and speculative adders. In [14], a reconfigurable prediction scheme is proposed where carry-in value of current window is predicted by previous K windows, where K is a configurable parameter. This serves as an improvement for ACA achieving accuracy-configurability during compute stage.

3. PRELIMINARIES

In this section, we review the basics of the speculative adders in terms of how they achieve speed improvement with short carry chains and detect errors. In particular, we will examine VLSA and ACA, which are two representative speculative adder designs, in more detail.

3.1 Group Adders

The rationales of the existing speculative adder designs are similar. The key insight is that the typical length of the carry propagation is much shorter than the worst-case scenario. Hence, dividing a long addition into smaller groups and executing them in parallel would lead to smaller delay and potentially lower power consumption.

Existing speculative adders can be roughly categorized into two classes based on how addition groups are defined and how the final sum is composed from the result of each group adder. In the first class [11, 13], each group adder only contributes one bit to the final sum, i.e., only the most significant bit (msb) of each group adder will be used. VLSA is a good representative design for this class of speculative adders. In the second class [10, 14, 16, 15, 2], the addition is divided into multiple windows with each window responsible for generating a range of bits for the final sum. The existing window-based speculative adders are logically similar, but usually exhibit different trade-offs among delay, area and power consumption. In this paper, we focus on ACA [10] in the subsequent discussion since it is a good representative of the window-based speculative adders and has a low area overhead.

Figure 1 illustrates the circuit structures of group adders in VLSA and ACA. In VLSA, only the msb of each group addition will be used in the final result. VLSA can achieve relatively high accuracy because each bit in the final sum is calculated from one dedicated group adder. However, the circuit cost is also relatively high for VLSA because of the large number of group adders needed. In contrast, window-based speculative adders such as ACA organize group additions in a manner of overlapping windows. Area overhead is reduced in this case since less amount of redundant computation is conducted.

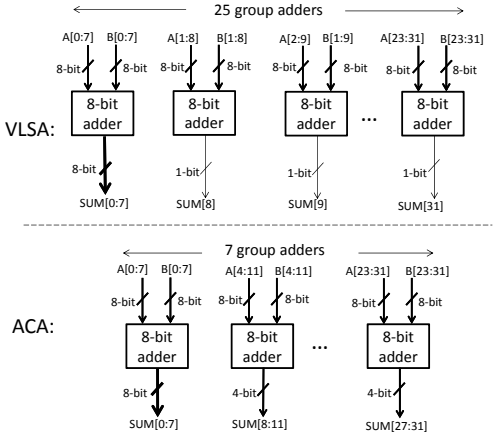

Figure 1: Group adders of VLSA and ACA.

The key design technique to ensure high accuracy is that only certain part (usually left-most bits) of the group sum goes into the final result. In this scenario, even if there exists a carry propagation from the previous group, the result of current group addition may not necessarily be wrong. This is because there is a high probability that carry will be generated or killed inside the lower bits in the group adder, so that carry-in from the previous group will have no impact on the correctness of the current group addition.

3.2 Error Condition

Speculative adders truncate the original addition into group additions of length K, and assume that the carry-in values will not propagate across more than K bit positions. Hence it is not too difficult to derive the error condition of a speculative adder as follows:

$$ErrorFlag = \sum_{i=0}^{n-k-1} p_i p_{i+1} \dots p_{i+k} g_{i+k+1} \quad (1)$$

The definitions g_i and p_i signals in the above equation are described below. We also define the kill signal (k_i) to facilitate the later discussions.

- Generate g_i: $g_i = 1$ if both operands are 1 at position i;
- Kill k_i: $k_i = 1$ if both operands are 0 at position i;
- Propagate p_i: $p_i = 1$ if two operands differ at position i.

It is worth noting that having a long sequence of propagate signals p's does not necessarily result in an error. In fact, only when this sequence is trailed by a g signal (instead of k), the actual carry propagation would occur, resulting in an error.

4. QUANTITATIVE STUDY OF REPRESENTATIVE SPECULATIVE ADDERS

In this section, we present a quantitative study using real-life benchmarks to evaluate the effectiveness of VLSA and ACA.

4.1 Error Rate Evaluation

We have set up a comprehensive evaluation flow based on the GEM5 [1] architectural simulator to assess the effectiveness of existing speculative adders against real-life benchmarks. Specifically, we have modified the arithmetic logic units in a MIPS32 processor and tested benchmarks from the MiBench [5] suite. Results of two representative speculative adders, i.e., VLSI and ACA, are compared against the correct results to obtain error instructions and derive the error rates.

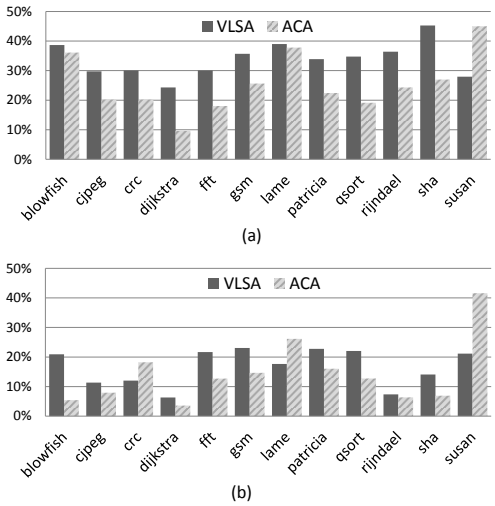

Figure 2: Error rates for VLSA and ACA. (a) K=4; (b) K=8.

The error rates of VLSA and ACA on 12 MiBench benchmarks are shown in Figure 2. Among these benchmarks, the error rates vary widely from below 10% to above 40%. This suggests that the common practice of using uniformly distributed random test vectors is insufficient for evaluating the effectiveness of existing speculative addition techniques. Furthermore, this study shows that significant room exists for reducing the error rates of state-of-the-art speculative adders.

4.2 Diagnosis of Erroneous Additions

In this section, we examine the data patterns in MiBench benchmarks that frequently trigger errors in speculative adders. We compile the benchmark code of MiBench, analyze the assembly, and keep track of the operands of the addition instances during execution.

We identify the most frequently executed and error-prone instructions and examine the characteristic patterns of their

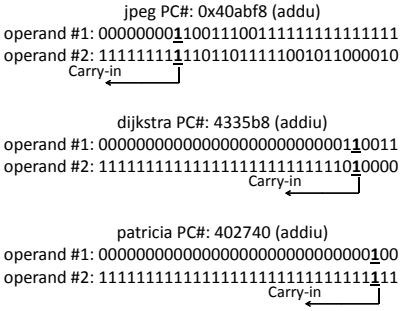

Figure 3: Example of long carry propagation chain causing errors.

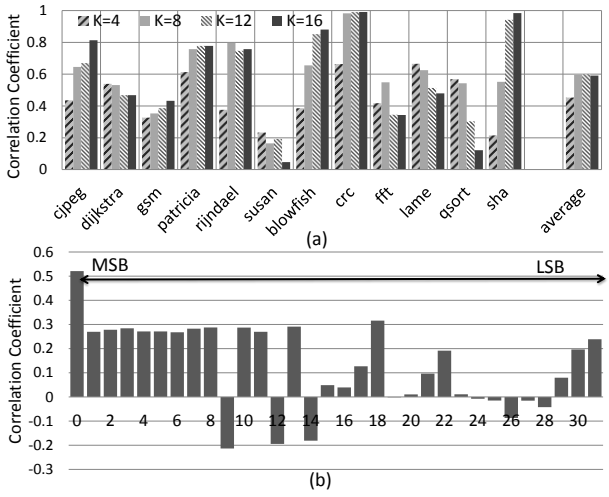

Figure 4: (a) Correlation coefficient for different benchmarks. (b) Correlation coefficient for each of the 32 bits for benchmark *fft*.

source operands. Figure 3 shows several instances of such additions that are executed many times and prone to errors. It can be observed that the carry propagation chains in the above examples are usually long. Any speculative adder that fails to anticipate such long carry propagation will result in an error. In fact, we observe that in most cases these long carry propagation chains arise from sign bit extensions. In other words, long sequence of propagate signals usually happen when adding numbers of different signs. These long carry chains are the major source of errors.

Given this observation, we hypothesize that there exists a correlation between the difference of the most significant bits (i.e. $XOR(msb)$) and the length of the carry propagation chain. We will further justify the correlation in the following section, and show that this correlation-based speculation technique can be used as a lightweight extension to speculative adders, achieving significant reduction in error rate.

4.3 Correlation Analysis

As previously mentioned, we hypothesize that there exists correlation between the length of the carry propagation and the $XOR(msb)$ value. In this section, we will examine this hypothesis based on the well-known Pearson's correlation coefficient for additions from MiBench. To facilitate our test, we define the following two random variables:

- Random variable X: $X = 1$ $if XOR(msb) = 1$; $x = 0$ if $XOR(msb) = 0$.
- Random variable Y: $Y = 1$ if there exists a sequence of propagate signal ending with generate g (i.e. $ppp\ldots g$) whose length is equal to or greater than the group size.

The Pearson's correlation coefficient between X and Y can be defined as:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \quad (2)$$

where cov is the covariance and σ_X and σ_Y are the standard deviation of X and Y respectively. Intuitively, large $\rho_{X,Y}$ indicates strong correlation between X and Y, while small $\rho_{X,Y}$ indicates weak correlation between X and Y.

To investigate the Pearson's correlation coefficient of X and Y, we have sampled 12 benchmarks from MiBench. Figure 4 shows the Pearson's correlation coefficient of X and

Y in these designs. Results show that the correlation coefficient are greater than 0.5 in a majority part of benchmarks, which largely confirms our hypothesis. [1]

5. CASA DESIGN

In this section, we present the design of our correlation-aware speculative addition (CASA) and the corresponding error rate evaluation. CASA is a lightweight carry prediction scheme that can serve as an extension to most existing speculative adders. Here we use ACA adder [10] as our baseline design, while extensions to other speculative adders can be fulfilled in a similar manner.

5.1 Implementation of CASA

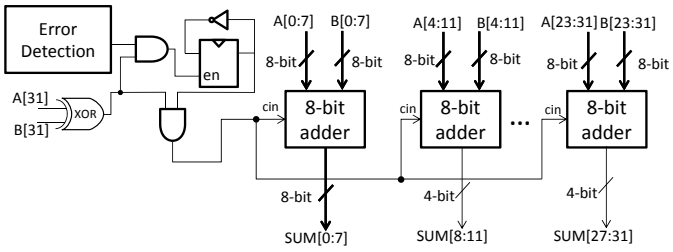

Figure 5: Circuit implementation of CASA.

Figure 5 shows the structural diagram of the CASA extension based on ACA. The first modification to the original ACA adder is an XOR gate connecting the msb of two operands to the carry-in of group adders. When $XOR(msb) = 0$, the carry-in to the group adders will be all zeros, which is the same in the original ACA. As long as the length of carry propagation chain does not exceed K, CASA will always generate the correct result. On the other hand, when $XOR(msb) = 1$, the carry-in to the group adders will be set to one to account for the potential long carry propagation that are frequently encountered when the most significant bits differ. In this case, CASA will achieve higher accuracy compared to the baseline design because it often correctly predicts the carry-in signal generated from previous group adders.

In order to improve the error rate for applications with relatively small correlation coefficient, we also propose a dynamic 1-bit prediction scheme to further reduce error rate for certain benchmarks. The high-level idea of this 1-bit prediction scheme is similar to a simple 1-bit branch predictor. Instead of always predicting carry-in equals one when $XOR(msb) = 1$, we make CASA adaptive by keeping some history information and update the prediction value every time an error occurs. In our case, if the prediction is wrong for the current instruction leading to an error, the prediction outcome will be reversed. This way, the carry prediction unit can dynamically adapt to the characteristics of the input vector, thus further reducing the error rate.

Prediction is realized by one flip-flop enabled by the error detection circuit. We only apply dynamic prediction when $XOR(msb) = 1$ by introducing an AND gate connecting the XOR gate and the output of the flip-flop. Also, to ensure that the state of prediction is only updated when $XOR(msb) = 1$, we implement an enable signal driven by the result of an AND gate that takes as inputs the error flag and output of XOR gate. Then each group adder will conduct addition based on the predicted carry-in and two operands, and the final result will be selected from the group adders accordingly.

Our CASA implementation is lightweight and flexible. CASA only requires one extra XOR gate and one additional flip-flop to conduct correlation-based speculation and dynamic prediction. Although we are extending ACA in this case, CASA can be applied to other speculative adders such as VLSA.

5.2 Error Rate Analysis of CASA under Uniformly Distributed Random Inputs

Although CASA is motivated by observations from real-life input vectors, we can also show that CASA does not significantly degrade the performance under uniformly distributed random input vectors. First of all, we note that for uniformly distributed random inputs, dynamic 1-bit prediction is unbiased and have no impact on the overall error rate. This is confirmed by our Monte Carlo simulation. For simplicity, we restrict our analytical evaluation to CASA with correlation speculation and take VLSA as our baseline design.

Let us first define function $G_n(x)$ as the total number of n-bit long sequences in which the longest run of $pppp\ldots g$ does not exceed x. $G_n(x)$ can be calculated using the following recursive relation:

$$G_n(x) = \sum_{i=1}^{x} 2^i \times G_{n-i}(x) + \sum_{i=x+1}^{n} 2^{i-1} \times G_{n-i}(x) + 2^n \quad (3)$$

with the boundary condition:

$$G_n(x) = 4^n, n \leq x \quad (4)$$

where the first summation corresponds to run of p that is shorter than x and is terminated by g or k; the second summation corresponds to run of p longer than or equal to x but is terminated by k; and the third term corresponds to a run of p whose length is exactly n. Now using $G_n(x)$, error rate of CASA for $XOR(msb) = 0$ and $XOR(msb) = 1$ can be separately calculated as:

$$ErrorRate^{xor=0} = \frac{4^{n-1} - G_n^{xor=0}(x)}{4^{n-1}} \quad (5)$$

$$ErrorRate^{xor=1} = \frac{4^{n-1} - G_n^{xor=1}(x)}{4^{n-1}} \quad (6)$$

where

$$G_n^{xor=0}(x) = G_{n-1}(x) \quad (7)$$

$$G_n^{xor=1}(x) = G_{n-1}(x) - \sum_{i=x+1}^{n-1} 2^i \times 4^{n-i-1} \quad (8)$$

The additional sum in Equation 8 accounts for the fact that when $XOR(msb) = 1$, the msb itself contributes to possible propagate chains starting from msb. The overall error rate for CASA is the expectation of Equation 5 and Equation 6:

$$ErrorRate = \frac{1}{2} \times (ErrorRate^{xor=0} + ErrorRate^{xor=1}) \quad (9)$$

As a reference, the error rate for VLSA can be calculated as:

$$ErrorRate = \frac{4^n - G_n(x)}{4^n} \quad (10)$$

[1] It is worth noting that [6] also observed that long carry propagation chains are more likely to occur when adding small numbers with opposite signs in DCT/IDCT algorithms. Based on this observation, approximate adders with reduced bitwidths are used to improve the area and timing of the design.

Table 1: Theoretical error rate for VLSA and CASA with uniformly distributed random input vectors.

K	4	6	8	10	12
VLSA	37.44%	9.87%	2.33%	0.54%	0.12%
CASA	39.50%	10.29%	2.43%	0.56%	0.13%

Based on above derivations, we provide theoretical error rates for VLSA and CASA under uniformly distributed random input vectors.

From Table 1, we observe that CASA achieves similar accuracy compared to baseline design, confirming the feasibility of CASA even for uniformly distributed random vectors.

5.3 Error Detection for CASA

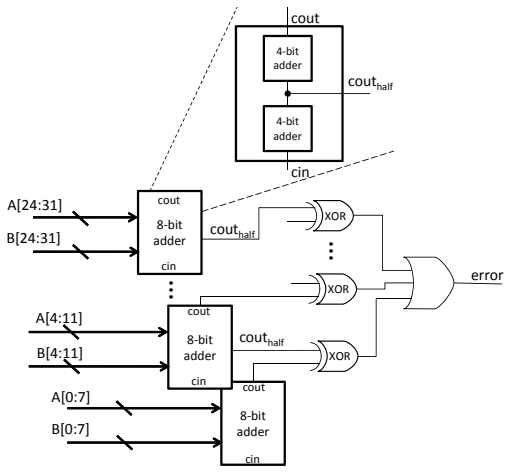

Figure 6: Error detection circuit for CASA. ($cout_{half}$ represents carry-out signal generated by the lower 4 bits in the group adder.)

Our error detection logic is similar to the ACA adder [10] but with slight modification to account for the different carry-in values predicted by the XOR gate. A naive way of implementing error detection circuit is to search in the original input operands for every sequence of $ppp\ldots g$ or $ppp\ldots k$ for $XOR(msb) = 0$ and $XOR(msb) = 1$, respectively. Then the error detection circuit will flag an error if the length of such sequences is longer than the available look-ahead bits in the group adder. Here we introduce a logically equivalent but much simpler version of error detection circuit: For each group addition, error detection circuit compares the carry-out value from the previous window with the carry-out value generated by the lower-half of the current group adder, as shown in Figure 6. If the error detection circuit finds a mismatch between these two values in any of the group adders, it means that the upper-half of the group adder fails to capture the correct carry-in value. Thus the final result will be wrong and error detection circuit will flag an error. Otherwise, if no mismatch is found, the result is guaranteed to be correct.

6. EXPERIMENTAL RESULTS

In this section, we first present error rate reduction of CASA applying to both VLSA and ACA. Then we report the timing, area, and power results of our CASA design.

6.1 CASA Error Rate Evaluation

We evaluate the error rate reduction of using the proposed CASA method on VLSA and ACA, under two different group sizes with $K = 4$ and $K = 8$. Table 2 shows that CASA can significantly reduce the average error rate of VLSA from 16.7% to 3.5% when $K = 8$. For ACA, CASA reduces the average error rate from 14.4% to 2.1%. When $K = 4$, the error rate reductions are also substantial.

Figure 7 shows the breakdown of the error rate reductions by CASA due to prediction and correlation. It is important to note that dynamic prediction alone does not provide sufficient error rate reductions without correlation-based speculation. For example, the error rate for fft only decreases from 12.7% to 11.9% if we employ a naive dynamic prediction scheme where carry-in prediction is always updated regardless of the result of $XOR(MSB)$. In contrast, the final error rate is down to 3.7% with CASA. For *patrica*, the error rate even increases from 16.1% to 17.8% after applying naive dynamic prediction. With CASA, the error rate is as small as 2.0%. From our experiments, coordinated prediction and carry-in speculation leads to most significant error rate reduction.

6.2 Circuit Implementation

We have implemented CASA in behavioral *Verilog* and synthesized the circuit using Synopsys Design Compiler with a Synopsys 90nm technology library. Table 3 compares the timing, area, and power results of parallel prefix adder, ACA, and CASA. Comparing to the original ACA, CASA can lead to significant error rate reduction with only marginal hardware overhead.[2] We also compare CASA with a fast parallel prefix adder provided by the Synopsys DesignWare library. According to Table 3, CASA is able to achieve 25.6% power savings without any compromises in speed.

Table 3: Implementation results for 32-bit ACA and CASA, K=8.

	prefix adder	ACA	CASA
delay (ns)	0.594	0.559	0.560
area (μm^2)	4238	3063	3300
power (mW)	1.29	0.93	0.96
CASA vs. ACA			
timing overhead			0.2%
area overhead			7.7%
power overhead			3.2%
CASA vs. prefix adder			
power savings			25.6%

7. CONCLUSIONS

In this paper, we conduct quantitative study of existing speculative adders with realistic benchmarks and propose a lightweight extension called CASA to significantly reduce the error rate. Our approach is based on the correlation between the MSB of input operands and the carry-in values for the group adders. To validate our observation, we provide detailed correlation analysis based on the input vectors extracted from real-life benchmarks. We further perform circuit implementation of CASA and detailed performance and power analysis to validate the feasibility of our method.

ACKNOWLEDGMENTS

This work was supported in part by NSF Award CCF-1337240 and a research gift from Xilinx, Inc.

[2]Note that we can potentially use a CASA adder with a smaller group size to meet the same accuracy requirement, leading to even higher speed and better energy efficiency than the baseline speculative adders.

Table 2: Error rate reduction after applying CASA.

	VLSA vs. CASA				ACA vs. CASA			
	K=4		K=8		K=4		K=8	
	VLSA	CASA	VLSA	CASA	ACA	CASA	ACA	CASA
blowfish	38.7%	23.4%	20.9%	1.3%	36.1%	5.1%	5.4%	0.1%
cjpeg	29.8%	14.5%	11.4%	3.6%	20.2%	6.4%	7.9%	2.5%
crc	30.1%	14.8%	12.0%	0.1%	20.2%	0.0%	18.2%	0.1%
dijkstra	24.3%	6.7%	6.3%	2.4%	9.6%	3.5%	3.6%	1.8%
fft	30.0%	7.2%	21.7%	2.5%	18.0%	5.2%	12.7%	3.7%
gsm	35.7%	27.2%	23.1%	18.5%	25.6%	17.4%	14.6%	9.0%
lame	39.0%	10.6%	17.6%	1.2%	37.8%	2.7%	26.2%	0.4%
patricia	33.9%	12.4%	22.8%	3.4%	22.4%	4.7%	16.1%	2.0%
qsort	34.7%	7.5%	22.0%	3.9%	19.1%	5.4%	12.7%	3.6%
rijndael	36.4%	13.7%	7.3%	0.9%	24.3%	3.4%	6.4%	0.2%
sha	45.3%	24.6%	14.1%	1.7%	27.0%	6.6%	6.9%	0.9%
susan	27.9%	13.0%	21.2%	2.4%	45.0%	8.3%	41.6%	0.6%
average	33.8%	14.6%	16.7%	3.5%	25.4%	5.7%	14.4%	2.1%

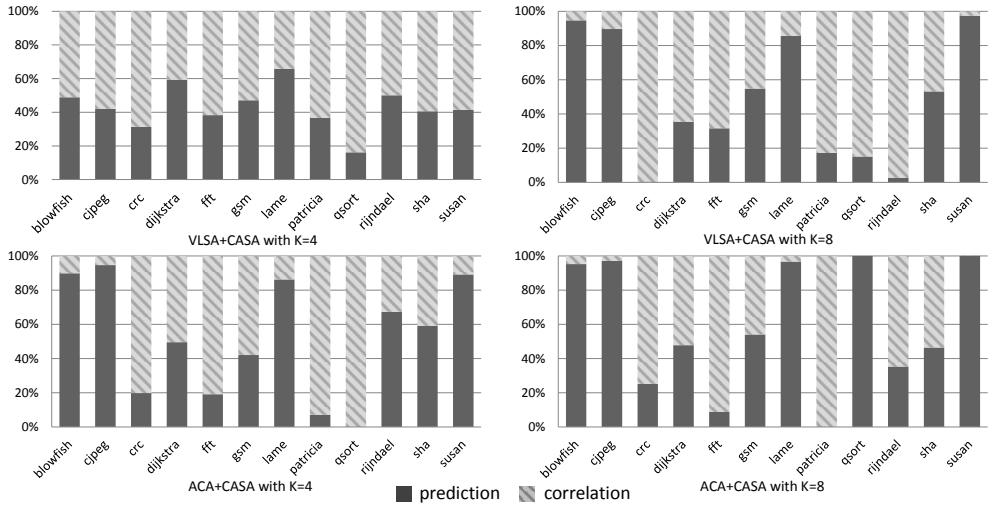

Figure 7: Breakdown of error rate improvements for CASA due to dynamic prediction and correlation-based speculation.

REFERENCES

[1] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2):1–7, 2011.

[2] K. Du, P. Varman, and K. Mohanram. High performance reliable variable latency carry select addition. In *Proceedings of the conference on Design, automation and test in Europe*, pages 1257–1262, 2012.

[3] P. Gupta, Y. Agarwal, L. Dolecek, N. Dutt, R. K. Gupta, R. Kumar, S. Mitra, A. Nicolau, T. S. Rosing, M. B. Srivastava, et al. Underdesigned and opportunistic computing in presence of hardware variability. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 32(1):8–23, 2013.

[4] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy. Low-power digital signal processing using approximate adders. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 32(1):124–137, 2013.

[5] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown. Mibench: A free, commercially representative embedded benchmark suite. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 3–14, 2001.

[6] K. He, A. Gerstlauer, and M. Orshansky. Circuit-level timing-error acceptance for design of energy-efficient dct/idct-based systems. *Circuits and Systems for Video Technology, IEEE Transactions on*, 23(6):961–974, 2013.

[7] R. Hegde and N. R. Shanbhag. Energy-efficient signal processing via algorithmic noise-tolerance. In *Proceedings of the 1999 international symposium on Low power electronics and design*, pages 30–35, 1999.

[8] R. Hegde and N. R. Shanbhag. Soft digital signal processing. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 9(6):813–823, 2001.

[9] J. Huang, J. Lach, and G. Robins. A methodology for energy-quality tradeoff using imprecise hardware. In *Proceedings of the 49th Annual Design Automation Conference*, pages 504–509, 2012.

[10] A. B. Kahng and S. Kang. Accuracy-configurable adder for approximate arithmetic designs. In *Proceedings of the 49th Annual Design Automation Conference*, pages 820–825, 2012.

[11] S.-L. Lu. Speeding up processing with approximation circuits. *Computer*, 37(3):67–73, 2004.

[12] J. Miao, K. He, A. Gerstlauer, and M. Orshansky. Modeling and synthesis of quality-energy optimal approximate adders. In *Proceedings of the International Conference on Computer-Aided Design*, pages 728–735, 2012.

[13] A. K. Verma, P. Brisk, and P. Ienne. Variable latency speculative addition: a new paradigm for arithmetic circuit design. In *Proceedings of the conference on Design, automation and test in Europe*, pages 1250–1255, 2008.

[14] R. Ye, T. Wang, F. Yuan, R. Kumar, and Q. Xu. On reconfiguration-oriented approximate adder design and its application. In *Proceedings of International Conference on Computer-Aided Design*, 2013.

[15] N. Zhu, W. L. Goh, G. Wang, and K. S. Yeo. Enhanced low-power high-speed adder for error-tolerant application. In *Proceedings of the 7th International SoC Design Conference*, pages 323–327, 2010.

[16] N. Zhu, W. L. Goh, and K. S. Yeo. An enhanced low-power high-speed adder for error-tolerant application. In *Proceedings of the 12th International Symposium on Integrated Circuits*, pages 69–72, 2009.

[17] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong. Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 18(8):1225–1229, 2010.