ACA-CSU: A Carry Selection Based Accuracy Configurable Approximate Adder Design

Alish Kanani
Department of Electrical Engineering
Indian Institute of Technology Jodhpur
Jodhpur, India
kanani.1@iitj.ac.in

Jigar Mehta, Neeraj Goel
Department of Computer Science and Engineering
Indian Institute of Technology Ropar
Ropar, India
2018csb1155, neeraj@iitrpr.ac.in

***Abstract*—Approximate arithmetic circuits can be more effective if their accuracy can be controlled. In this paper, we propose a carry selection based accuracy configurable approximate adder. In the proposed design, longer carry chains provide better accuracy and CSU (Carry Select Unit) gives it better delay properties. The proposed design is generic so that multiple accuracy levels are possible at design time. Our experiments show that our design is considerably more accurate than the already proposed state of the art approximate adders. The latency of the proposed adder is also better than the state of the art adders.**

***Index Terms*—Approximate computing, Configurable accuracy, Carry Select Unit, Fast adder.**

I. Introduction

Approximate computing paradigm provides both high computation speed as well as low power, and are therefore, promising for many domains like machine learning, multimedia, and image processing [1] [2]. Designing circuits that perform approximate computations are the core engines of approximate computing paradigm. Further, in many digital circuits, like multiplier, accumulator, division, floating-point operations, an adder is the basic building block. Therefore, many algorithms for approximation adder have been proposed ([8] – [18]) in past. Most of these adders give a fixed error for a given set of input.

The utility of an approximate circuit depends on the quality requirements of the application. Further, an application may require different levels of accuracy for different operations. Therefore, fixed accuracy adders are not sufficient. ACA [3] and GeAr [4] address this issue by a separate error detection circuit. Additional cycles are required to correct the error in these adders. Another set of accuracy configurable adders (GDA [5], RAP-CLA [6], SARA [7]) use multiplexer based design. Based on control input, these designs multiplex between correct carry and approximate carry.

Accuracy configurable adders have a fixed number of error levels. Based on a control input, error of the adder changes. The lowest level gives the most approximate result, while the highest level gives an *accurate* result. Since the design of such adders is capable of providing approximate as well as accurate results, the proposed design takes a larger area than the corresponding accurate design. In a wider perspective, the proposed adder still saves area since it doesn't require two adders: approximate and accurate.

In this paper, we build upon multiplexer based configurable adders. We propose a carry select unit (CSU) based accuracy configurable approximate adder. The proposed design is generic, thus, can provide any number of accuracy levels. The proposed design has higher accuracy than the previous designs, and an efficient carry select unit increases the speed of the adder. Our experiments also show that our design is considerably better than previously proposed approximate adders in terms of accuracy, and also is better than other adders in terms of latency.

The rest of the paper is organized as follows. In Section II, we have reviewed some previous approximate adders. The detailed algorithm is presented in Section III. Simulation result and comparison data are presented in Section IV. Finally, in Section V the paper is concluded.

II. Related work

This section provides a brief review of some previous works on approximate adders. Some of these adders may be classified as speculative adders, as they speculate carry based on a few previous bits [8] [6]. Due to smaller carry chains, these designs are faster and require less area than accurate adders. RAP-CLA [6] is an accuracy configurable design, that computes speculated carry as well as correct carry, which leads to costlier a design than an accurate adder.

The second type of adders are approximate full adders [9] [10] [11] [12] [13], where lower significant bits are approximately computed and upper significant bits are accurately computed.

In the third type of approximate adders, operands are divided into segments [14] [15] [16] [17] [3] [4] [18]. The sum of each segment is computed independently: some blocks compute sum using accurate methods, some using approximate methods. In accurate design, input carry of a block is based on carry generated by all lower significant blocks. In approximate design, the input carry is predicted and forms the basis of the approximate adder.

The Fourth type of approximate adders are based on Carry-Select Adders [19] [5] [20] [21]. In these adders, every block computes sum assuming carry input equals to '0' and '1' similar to conventional carry select adder, and one of them is selected based on predicted carry rather than accurate carry.

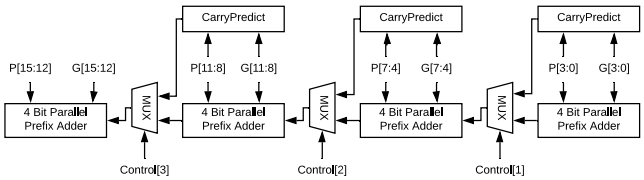


Fig. 1: Block diagram of 16 bit Accuracy Configurable Adder

Apart from the above classification, Liu et al. [22] proposed an approximate adder based on majority logic. [23] proposes to compute the error matrix efficiently using formal methods. [24] develops a method for probability based error analysis of approximate adders.

III. Proposed ACA-CSU Adder

Accurate adders are usually slow because carry-in at i^{th} bit depends on carry generated and propagated from 0 to (i-1) bits. In approximate adders, the carry-in is either predicted statically as zero/one, or predicted based on carry generate of some previous bits. We observe that longer the chain of carry prediction, higher is the accuracy.

A. Accuracy configurable design

We design an N bit adder that takes $A_{N-1:0}$ and $B_{N-1:0}$ as input and generates $S_{N-1:0}$ as outputs. Our proposed adder is divided into L blocks of equal size, where each block has M bits. Each block has two parts: Sum Generators and Carry Predictors. j^{th} Sum Generator takes Propagate $P_{j*M+M-1:j*M}$, Generate $G_{j*M+M-1:j*M}$, and Carry Cin_j as input, and gives Sum $S_{j*M+M-1:j*M}$ and Carry $Cout_j$ as output. Generate and Propagate is calculated as:

$$P_i = A_i \oplus B_i; G_i = A_i \bullet B_i \tag{1}$$

Based on these inputs, sum generators calculate sum using parallel prefix logic. Carry predictor, *CarryPredict*, takes K bits of propagate($P_i s$) and generate($G_i s$) as input and predicts a carry, $CPredict$, using parallel prefix logic. j^{th} CarryPredict takes $P_{j*M-1:j*M-K}$ and $G_{j*M-1:j*M-K}$ as input and gives $CPredict_j$ as output. For example, if K = M = 4, $CPredict_1$ is calculated using the following equation:

$$CPredict_1 = G_3 + G_2 \bullet P_3 + G_1 \bullet P_2 \bullet P_3 + G_0 \bullet P_1 \bullet P_2 \bullet P_3 \tag{2}$$

In overall circuit, based on $Control_j$ either $CPredict_j$ or $Cout_{j-1}$ is selected in carry input to j^{th} block. Thus, accuracy of adder is decided by $Control$, and the adder has L accuracy levels.

Figure 1 shows an example 16 bit ACA-CSU, where L=M=K=4. The four accuracy levels are given by $Control$ value 000, 100, 110, and 111, where first is the most approximate and last is an accurate adder. It may be noted that configuring most significant sum block as approximate, and lower significant sum block as accurate does not make practical sense, thus control values like 010 are not classified as accuracy levels.

The proposed design is generic, as it can have different accuracy levels defined by value of L.

B. Accuracy of ACA-CSU

The maximum carry chain of an m-bit parallel prefix adder is M. In our design Sum Genarator block takes input carry from CarryPredict block in approximate mode. CarryPredict creates a carry chain of K preceding bits. Thus, increasing the accuracy of a sum block to $K + M$ carry chain. It may be noted that K and M can be different.

Accuracy can be further enhanced if we consider $G_{j*M-K-1}$ also to calculate $CPredict_j$. To consider this, we define another signal block propagate for j^{th} CarryPredict block, BP_j, which is calculated as:

$$BP_j = \prod_{i=j*M-K}^{j*M-1} P_i \tag{3}$$

$BP_j = 1$ means carry is not generated by this carryPredict block, thus passing $G_{j*M-K-1}$ as carry, in this case, will increase the accuracy. In other words, this increases the maximum carry chain from $K + M$ to $K + M + 1$.

C. Design of Carry Select Unit

Carry Select Unit (CSU) takes $Control_j$, $CPredict_j$, $Cout_{j-1}$, $G_{j*M-K-1}$, and BP_j as input and generates carry in for j^{th} block as shown in figure 2. By using the semantic meaning of the inputs signals, CSU can be designed optimally. The following points explain the symantic meaning of these terms.

- BP_j = '0' means either carry is generated or killed or both in j^{th} CarryPredict block. Therefore, carry chains of preceding bits (i < j*M-K) will have no impact on carry generated at this stage. Thus, in both approximate and accurate mode Cin_j is equal to $CPredict_j$.
- BP_j = '1' and $Control_j$ = '0' means carry is not generated by j^{th} carryPredict block, thus passing $G_{j*M-K-1}$ in approximate mode carry increases the probability of having correct carry input.
- If BP = '1' and $Control^i$ = '1' that means in accurate mode $Cout_{j-1}$ is the correct carry.

Thus, overall logic of CSU can be written as:

$$\begin{aligned} Cin_j = CPredict_j \bullet \overline{BP_j} + \\ \overline{Control_j} \bullet BP_j \bullet G_{j*M-K-1} + \\ Control_j \bullet BP_j \bullet Cout_{j-1} \end{aligned} \tag{4}$$

D. Delay analysis

In accurate mode, delay of the ACA-CSU is *L*(delay of M-bit parallel prefix adder) + (L-1)* (delay of CSU)*. In most approximate mode, delay of the adder is *(delay of M-bit parallel prefix adder) + (delay of CSU) + (delay of CarryPredict)*. Delay of the CarryPredict depends on K.

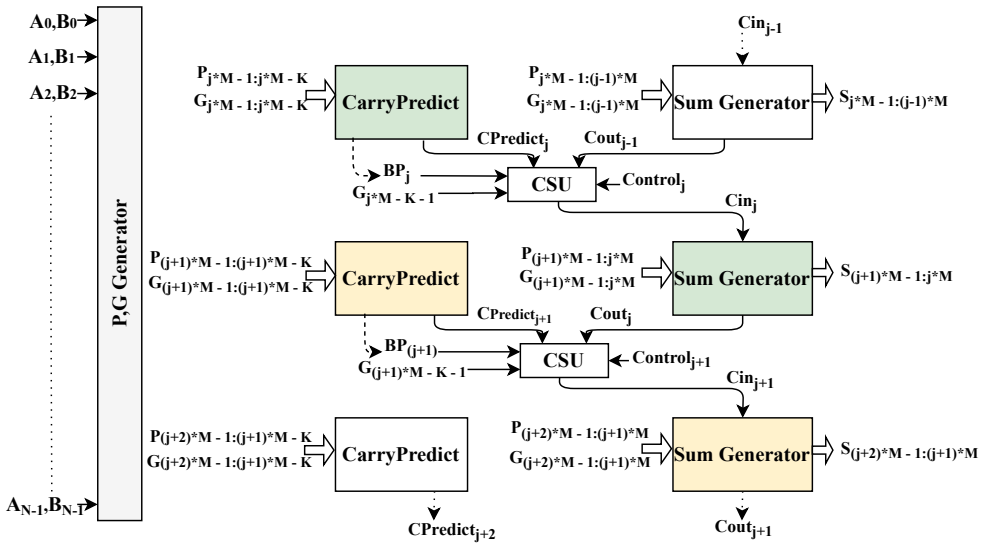


Fig. 2: Schematic of proposed ACA-CSU Adder

E. Theoretical Error Analysis

In this sub-section, we analyze the error probability in uniform distribution for the proposed approximate adder. In approximate mode, a block will give erroneous result if CarryPredict block has predicted wrong carry. In this analysis, block size of Sum Generator(M) and CarryPredict(K) are kept same. If there are $(\frac{N}{K} = l)$ Sum Generator blocks then there will be $(l-1)$ CarryPredict units. The first CarryPredict Unit will always give an accurate output. Therefore, error only occurs due to remaining $(l-2)$ CarryPredict Units. For example, for 16 bit adder and block size of 4 bits, there will be error if either $CarryPredict_2$ is wrong or $CarryPredict_3$ is wrong. Error probability of overall addition can be calculated as (probability $CarryPredict_2$ is wrong) + (probability $CarryPredict_2$ is right) (probability $CarryPredict_3$ is wrong given $CarryPredict_2$ is right).

In general, say, PCR_i represents probability that $CarryPredict_i$ is wrong, $\overline{PCR_i}$ represents probability that $CarryPredict_i$ is right, PCW_i represents the probability $CarryPredict_i$ is wrong given $CarryPredict_{i-1}$, $CarryPredict_{i-2}$, ..., $CarryPredict_1$ are right. Probability that approximate adder gives wrong result, Pr(E) can be calculated as:

$$Pr(E) = PCR_2 + \sum_{i=3}^{l-1}(\prod_{j=2}^{i}\overline{PCR_j})PCW_i \tag{5}$$

Since ACA-CSU has also taken into account $G_{j*M-K-1}$ for each CarryPredict unit, probability that the whole carry predict unit is propagating is $\frac{1}{2^{K+1}}$. The $CarryPredict_j$ unit gives a wrong output only if the carry is generated earlier and propagated to $j*M-K-1^{th}$ bit. So, PCR_2 and PCW_i can be given by equations 6 and 7.

$$\begin{aligned} PCR_2 &= (\frac{1}{2^{K+1}})(\frac{1}{4})(1+\frac{1}{2}+\frac{1}{4}+...+\frac{1}{2^{K-2}}) \\ &= (\frac{2^{K-1}-1}{2^{2K+1}}) \end{aligned} \tag{6}$$

$$\begin{aligned} PCW_i &= (\frac{1}{2^{K+1}})(\frac{1}{4})(1+\frac{1}{2}+\frac{1}{4}+...+\frac{1}{2^{K-1}}) \\ &= \frac{2^K-1}{2^{2K+2}} \end{aligned} \tag{7}$$

Based on equations 6 and 7, probability of error can be given by equation 8. This error probability is verified with simulated result of Error Rate given in section IV-A.

$$\begin{aligned} Pr(E) =& (\frac{2^{K-1}-1}{2^{2K+1}}) + (1-\frac{2^{K-1}-1}{2^{2K+1}})(\frac{2^K-1}{2^{2K+2}}) \\ & (\sum_{i=1}^{l-2}(1-\frac{2^K-1}{2^{2K+2}})^{i-1}) \end{aligned} \tag{8}$$

IV. Experimental results and discussion

In this section, we present accuracy and performance of our proposed adder and compare it with three approximate adders: RAP-CLA [6], BCSA [18], BCSA with Error Reduction Unit (BCSAeru) [18], SARA [7], and accurate adders like Carry Look-ahead adder (CLA), Carry Select Adder, Carry Skip Adder and Brent Kung Adder. Among the three approximate adders only BCSA is not accuracy configurable while SARA and RAP-CLA are accuracy configurable.

A. Accuracy Analysis

For doing accuracy analysis, we did algorithmic implementation of all the mentioned adders using GNU Octave. An adder configuration is represented by *(N, M)*, where N is number of bits in the adder and M is the block size. For comparison, we used 8,16 and 32 bit as N values and 2,4 and 8 block-size. For fair comparison of our proposed adder, block size of Sum Generator(M) and CarryPredict(K) are kept same.

For the comparison, three error metrics – Error Rate (ER), Normalized Error Distance (NED) and Mean Relative Error Distance (MRED) are used. The error metrics are calculated by eq. 9, 10 and 11; where s is correct sum, $\overline{s}$ is approximate sum,

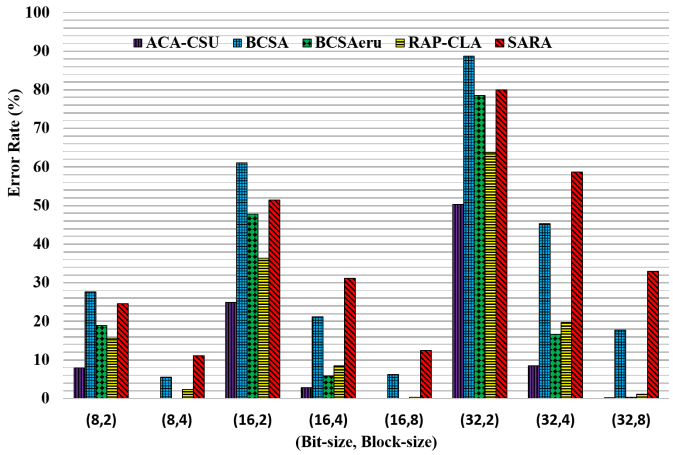


Fig. 3: Error Rate comparison for different configuration

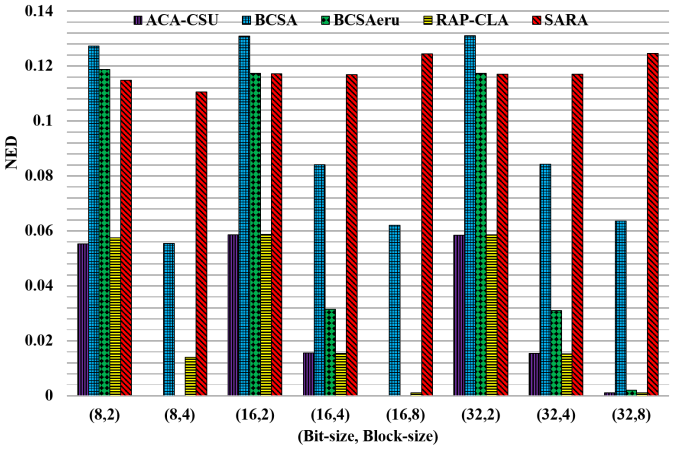


Fig. 4: NED comparison for different configuration

N stands for total number of test-cases, M is total number of incorrect answers and D is maximum error distance.

$$ER = M/N \tag{9}$$

$$NED = \frac{1}{N}\sum_{i=1}^{N}\frac{s-\overline{s}}{D} \tag{10}$$

$$MRED = \frac{1}{N}\sum_{i=1}^{N}\frac{s-\overline{s}}{s} \tag{11}$$

The addition is performed on 10^6 random numbers of appropriate bit-widths. Fig. 3, 4 and 5 shows the comparison of ER, NED, and MRED. It may be noticed that in (8,4) and (16,8) configurations our ACA-CSU and BCSAeru have ER, NED and MRED 0, means they are accurate in those cases. It may be noted that random numbers represent uniform probability distribution of input. Typical numbers in application may not follow uniform probability distribution.

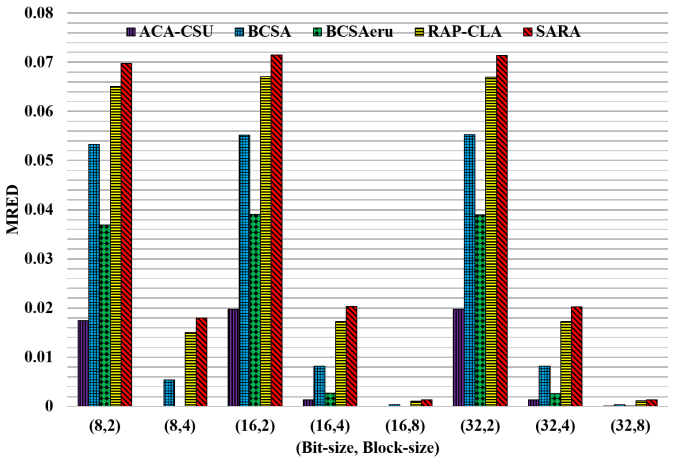


Fig. 5: MRED comparison for different configuration

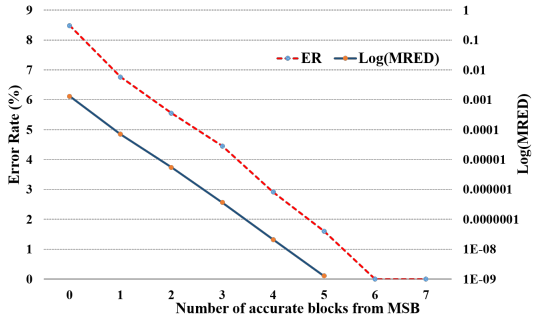


Fig. 6: ER and MRED for different number of accurate blocks

As the carry chains of ACA-CSU are the longest (M+K+1), it results in least ER, NED and MRED among all the studied adders. For example, ACA-CSU has 30%, 61%, and 53% less ER than any other adder for 8-bit, 16-bit and 32-bit adder, respectively. Similarly, our design has 37%, 66%, and 65% less NED than other adder for 8-bit, 16-bit and 32-bit adder, respectively. For MRED comparison, the proposed design has 26% less MRED for 8 bit adders, and 70% less MRED for 16 bit and 32 bit adders. Besides this, accuracy increases with the increase in block size. For example, the ER of our proposed adder is around 24%, 3% and 0% with a block size of 2,4 and 8 respectively.

We compared the experimental error rates of the proposed adder with the theoretically calculated error probability (eq. 8). Both the values were equal for all possible ACA-CSU adders. Therefore, our theoretical model can be used for calculating error rate for any configuration of the proposed method.

The configured accuracy (ER and Log(MRED)) of proposed adder for 32-bit operands and block size of 4 with a different set control bit is shown in Fig. 6. Error rate decreases linearly with increase in the number of accurate blocks while Log(MRED) decreases linearly so we can say that MRED decreases exponentially with the increase in the number of accurate blocks.

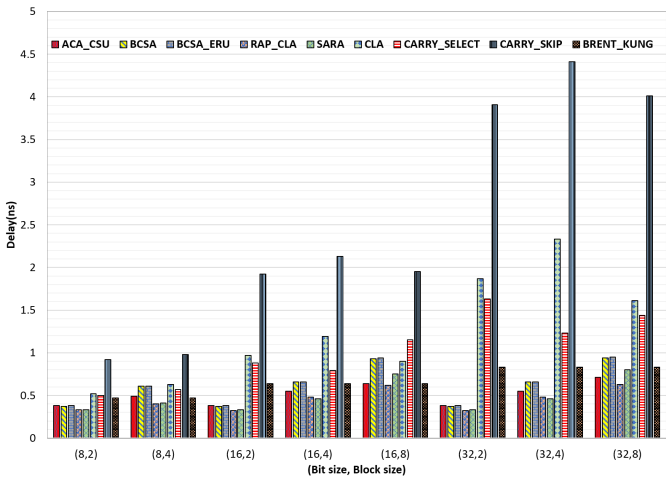


Fig. 7: Delay comparison for different configuration

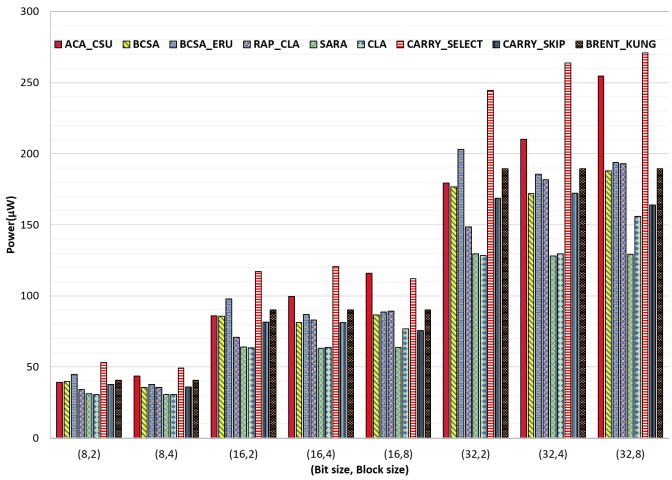
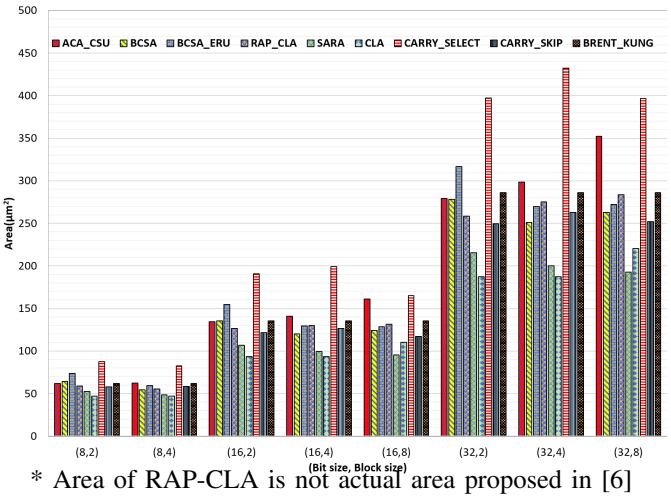


Fig. 9: Power comparison for different configuration



* Area of RAP-CLA is not actual area proposed in [6]

Fig. 8: Area comparison for different configuration

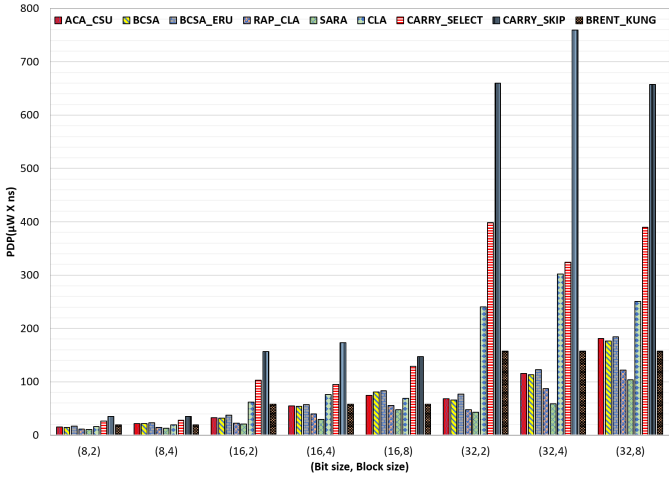


Fig. 10: PDP comparison for different configuration

B. Hardware Implementation

We modelled all the adders using Verilog HDL and synthesized them using Synopsys Design Compiler, and implemented using Nangate Open Cell Library. For ACA-CSU design, $Control$ is assumed as "0" and the size of Sum Generator(M) and CarryPredict(K) are kept the same. Blocks of RAP-CLA, BCSA, and BCSeru are implemented using Carry Look-ahead logic while SARA is implemented using simple Ripple Carry manner. The accurate Carry Look-ahead, Carry Select and Carry Skip Adder are also implemented block-wise and carries rippled through the blocks while Brent Kung Adder is implemented with a bit size of 8, 16 and 32 bit because it can not be implemented block-wise. All the other adders are implemented with a bit size of 8,16 and 32 with a block size of 2, 4 and 8.

In this section, four performance metrics are studied namely area, delay, power and power delay product (PDP). Figure 7 shows the delay comparisons of different adders. The delay of our proposed adder is more than SARA and RAP-CLA but because both adders have a maximum carry chain of M while our proposed adder has the (M+K)=2M length of maximum carry chain. ACA-CSU is on average 36% and 116% faster than Brent Kung(which is the fastest accurate adder among above mentioned accurate adder) and Carry Look-ahead adder respectively. ACA-CSU has comparable (on average 4% less) delay in comparison with other mentioned approximate adders. Notice that ACA-CSU has a much lower error rate compared to these adders. The area of ACA-CSU is comparable (on average 14% more) to all the approximate adders because of parallel prefix logic, the area of RAP-CLA cannot be compared with others because only approximate RAP-CLA is described in hardware, as shown in Fig. 8. In case of power, ACA-CSU has comparable (on average 18% more) power to all other approximate adders because of parallel prefix logic as shown in Fig. 9. In case of PDP, ACA-CSU has on average 29% and 71% less PDP than Brent Kung and CLA respectively, while ACA-CSU has comparable (on average 15% more) PDP in comparison with other mentioned approximate adder as shown in Fig. 10.

From the above studies ACA-CSU has little more PDP than some state of the art adder, but it is much accurate than all the other mentioned adder.

C. Implementation in a Image Processing Application

Fig. 11: Gaussian image smoothing: (a) Original image with noise, (b) Original filter, (c) Filter with rounded fixed point number, PSNR = 37.9761dB & SSIM = 0.95, (d) ACA-CSU, PSNR = 37.9123dB & SSIM = 0.9495,(e) BCSA, PSNR = 33.9086dB & SSIM = 0.9142,(f) BCSAeru, PSNR = 37.8372dB & SSIM = 0.9482,(g) RAP-CLA, PSNR = 29.3660dB & SSIM = 0.7814,(h) SARA, PSNR = 26.79dB & SSIM = 0.7870

In this section, we have used ACA-CSU and other adders in an image processing application of Gaussian Smoothing. In this application, $5X5$ gaussian filter is applied to a noisy image. Since we propose an integer adder, the fractional numbers used in the original filter are converted into fixed-point numbers. Approximate adders are used during convolution operation of Gaussian filter. In rest of the algorithm addition is accurate. We have used 32-bit adders with a block size of 8 in this application and compared SSIM (Structural Similarity Index) and PSNR (Peak Signal to Noise Ratio). All the results are shown in Figure 11. ACA-CSU has almost the same PSNR and SSIM as the accurate adder and highest PSNR, as well as SSIM, compared to all the recent state of the art adders. This shows that for realistic data, our proposed adder performs as good as an accurate adder.

V. Conclusion

In this paper, we have proposed a new carry selection based accuracy configurable adder, which can run in both, approximate and accurate mode. We show that design is generic and can lead to different accuracy levels based on size of carry chains, and number of blocks. In comparison to other state of the art methods, our proposed design shows higher accuracy in approximate mode. Using our proposed adder in an image processing application demonstrates the effectiveness of ACA-CSU.

References

[1] S. Mittal, "A survey of techniques for approximate computing." ACM Computing Surveys (CSUR), vol 48, no 4, 33 pages, 2016.

[2] Q. Xu, T. Mytkowicz and N. S. Kim, "Approximate Computing: A Survey," in IEEE Design & Test, vol. 33, no. 1, pp. 8-22, Feb. 2016.

[3] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," Design Automation Conference (DAC), pp. 820-825, 2012.

[4] M. Shafique, W. Ahmad, R. Hafiz and J. Henkel, "A low latency generic accuracy configurable adder," Design Automation Conference (DAC), pp. 1-6, 2015.

[5] R. Ye, T. Wang, F. Yuan, R. Kumar and Q. Xu, "On reconfiguration-oriented approximate adder design and its application," International Conference on Computer-Aided Design (ICCAD), pp. 48-54, 2013.

[6] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "RAP-CLA: A reconfigurable approximate carry look-ahead adder," IEEE TCAS-II, vol. 65, no. 8, pp. 1089–1093, 2018.

[7] W. Xu, S. S. Sapatnekar and J. Hu, "A Simple Yet Efficient Accuracy-Configurable Adder Design," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 6, pp. 1112-1125, June 2018.

[8] A. K. Verma, P. Brisk and P. Ienne, "Variable Latency Speculative Addition: A New Paradigm for Arithmetic Circuit Design," Design, Automation and Test in Europe (DATE), pp. 1250-1255, 2008.

[9] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie and C. Lucas, "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 4, pp. 850-862, April 2010.

[10] V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, "Low-Power Digital Signal Processing Using Approximate Adders," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124-137, Jan. 2013.

[11] T. Zhang, W. Liu, E. McLarnon, M. O'Neill and F. Lombardi, "Design of Majority Logic (ML) Based Approximate Full Adders," International Symposium on Circuits and Systems (ISCAS), pp. 1-5, 2018.

[12] M. Ramasamy, G. Narmadha and S. Deivasigamani, "Carry based approximate full adder for low power approximate computing," 2019 7th International Conference on Smart Computing & Communications (ICSCC), pp. 1-4, 2019.

[13] N. Zhu, W. L. Goh and K. S. Yeo, "Ultra low-power high-speed flexible Probabilistic Adder for Error-Tolerant Applications," 2011 International SoC Design Conference, pp. 393-395, 2011.

[14] D. Mohapatra, V. K. Chippa, A. Raghunathan and K. Roy, "Design of voltage-scalable meta-functions for approximate computing," Design, Automation and Test in Europe (DATE), pp. 1-6, 2011.

[15] 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," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 8, pp. 1225-1229, Aug. 2010.

[16] Ning Zhu, W. L. Goh and K. S. Yeo, "An enhanced low-power high-speed Adder For Error-Tolerant application," Proceedings of the 2009 12th International Symposium on Integrated Circuits, pp. 69-72, 2009.

[17] D. Celia, V. Vasudevan and N. Chandrachoodan, "Probabilistic Error Modeling for Two-part Segmented Approximate Adders," IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1-5, 2018.

[18] F. Ebrahimi-Azandaryani, O. Akbari, M. Kamal, A. Afzali-Kusha and M. Pedram, "Block-based Carry Speculative Approximate Adder for Energy-Efficient Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs, Vol 67, No 1, pp. 137-141,2020.

[19] K. Du, P. Varman and K. Mohanram, "High performance reliable variable latency carry select addition," Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 1257-1262, 2012.

[20] J. Hu and W. Qian, "A new approximate adder with low relative error and correct sign calculation," Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 1449-1454, 2015.

[21] Y. Kim, Y. Zhang and P. Li, "An energy efficient approximate adder with carry skip for error resilient neuromorphic VLSI systems," International Conference on Computer-Aided Design (ICCAD), pp. 130-137, 2013.

[22] W. Liu, T. Zhang, E. McLarnon, M. O'Neill, P. Montuschi and F. Lombardi, "Design and Analysis of Majority Logic Based Approximate Adders and Multipliers," in IEEE Transactions on Emerging Topics in Computing (Early access).

[23] M. Rezaalipour, M. Rezaalipour, M. Dehyadegari and M. Nazm Bojnordi, "AxMAP: Making Approximate Adders Aware of Input Patterns," in IEEE Transactions on Computers (Early access).

[24] S. Mazahir, M.K. Ayub, O. Hasan, M. Shafique, "Probabilistic Error Analysis of Approximate Adders and Multipliers" In Approximate Circuits, pp 99–120, Springer, Cham, 2019.