National Conference on Research Challenges and Opportunities in Digital and Cyber Forensics
International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2020 IJSRCSEIT | Volume 4 | Issue 11 | ISSN : 2456-3307 (www.ijsrcseit.com)

A New Approximate Adder with Block-based Carry Speculation for High- Speed Applications

Dr. Boda Saroja*[1], Rajesh Gundlapalle[2]
[1]Department of Electronics and Communication Engineering, Vemu Institute of Technology, Chittoor, A.P, India.
[2]Department of Electronics and Communication Engineering, New Horizon College of Engineering, Bangalore, Karnataka, India

ABSTRACT

In any digital signal processing operation adders play an important role. Here, a high speed adder based on block-based carry speculation is proposed. Its structure is dependent on the separation of the additive from other summer blocks that are not integrated into their properties which can be selected from various additives such as broadcast carriers or parallel prefix adders. Here, the acquisition output for each block is considered based on the installation of the block itself and that of the next block. The block extension used in this case is a modified flexible extension with less space and delay compared to the Parallel Supplement connector circuit. In this case, we recommend high performance but low power / power based block-carry that carries a limited design structure called BCSA Adder. To minimize the risk, we suggest a way to predict the exit of a block based on its features and on the next block.

Keywords : Parallel Prefix Adder, BCSA Adder, Power/energy Consumption.

I. INTRODUCTION

The basic operators of approximate adders in performing arithmetic operations are deliberated in this paper. Approximate/estimated adders have been acquired numerous consideration by the designers. In the cutting edge estimated adders, where a large portion of them depend on the convey spread structures, the vitality and speed gains have been accomplished by equipment controlling, rationale improvement, and voltage over scaling. Although a portion of the adders depended on a configurable precision, others had a fixed accuracy level. The precision configurability forced a few overheads regarding postponement, region, and force which could restrict their utilization in certain applications where such re-configurability isn't required.

In this paper, a high-performance but low-power structure is proposed that produces a standardized structure called the BCSA adder. In this setting, the adder is divided into several non-interlocking blocks, which, in the worst case, exit the block depends on the exit of the previous block. To further reduce the critical approach, we propose a way to predict block release on the basis of its characteristics and the next block. The structure has low hardware weight, which leads to low latency (on average, about one block) and very high quality. In order to achieve lower accuracy losses, a debugging and retrieval system is recommended, which significantly reduces operating error rate. The efficiency of this additive is compared to that of other bees. Finally, the efficiency of the adder is analysed.

The previous works in the improving the adder structures by approximation methods are briefed as Carry Look Ahead adders, Carry Skip adders and Block based Prediction of Carry adders viz RAPCLA [1], ACSA [2], ACAA [3], HABA [4] SARA[5]. These structures suffers from critical path delay and high power consumption at times compromised with area. However, it has suffered from high delay although it has a high output precision. In most of the approximate adders, the carry input of each block is selected based on the input of the previous blocks. In this work, however, we propose an approximate adder that the carry input is speculated.

II. METHODS AND MATERIAL

A. Adder with Error Recovery Unit:

Fig.1 illustrates the generalized approximate adder with carry prediction unit. The improvement in accuracy is achieved by increasing the output error recovery. The error is as much as reduced by appropriate selection of carry input on the succeeding block. Hence, the carry output is selected with the highest possible accuracy in each of the every case. The idea behind the carry output selection for the four cases (K_0^{i+1}= 0 and G_{l-1}^{i}= 0), (K_0^{i+1}= 0 and G_{l-1}^{i}= 1), (K_0^{i+1}= 1 and G_{l-1}^{i}= 0) and (K_0^{i+1} = 1 and G_{l-1}^{i} = 1) are considered. The speculations in carry speculated carry (C_{Prdt}^{i}) is correct for the three cases are well predicted except for the first case. Thereby, shortening the critical path, C_{Prdt}^{i} is selected as the carry output of the ith block. Therefore, in the proposed approach, C_{Prdt}^{i}. is selected as the C_{in}^{i+1}.

Between these cases, the carry in the first stage is distributed in two blocks. In the third case, although the block input is executed and no longer distributed, the carry input is used to obtain the first block measurement. Therefore, if the input in this case is incorrect, it affects the accuracy of the sum output, Likewise, for all other 3 cases the average length of the carry distribution is close to one, illustrated in Fig. 2.

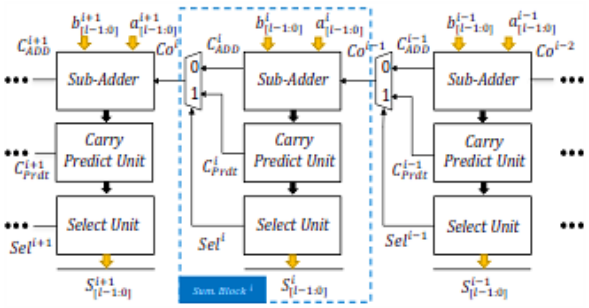

Figure 1 Generalized Approximate Adder with Carry Prediction.

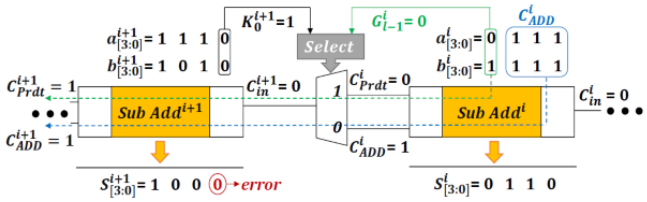

Figure 2 Approximate adder without Error Recovery Unit (ERU).

Hence, to improve the accuracy of the proposed connector, an error detection unit is suggested Fig. 3, that brings the first bit of the *ith* block S_0^i by,

$$S_0^{i+1} = \left(K_0^{I+1}.C_{Add}^{i}\right) + \left(P_0^{i+1} \oplus C_{in}^{i+1}\right) \qquad (1)$$

The first bit of (i+1)th block's summation output is predicted by the equation,

$$P_0^{i+1} \oplus C_{in}^{i+1}\left(C_{in}^{i+1} = C_{Prdt}^{i}\right) \qquad (2).$$

The ERU is will not be in the path of addition process. This improves the accuracy of Error Recovery Unit (ERU) without increasing the further delay

B. Proposed adder with new PPA:

In this adder, ripple carry adder is used in the sub adder. So instead of using ripple carry adder if we use parallel prefix adder, reduce the delay of the adder. Hence to improve the delay use Parallel Prefix Adder (PPA) like Brent-kung adder and new parallel prefix adder.

C. Error Recovery Unit:

The transmitted message is vulnerable to noise or data corruption. The additional error detection codes details given to the digital message helps in detecting the occurrence of errors during message transmission. A simple example of an error detection code is a unity check.

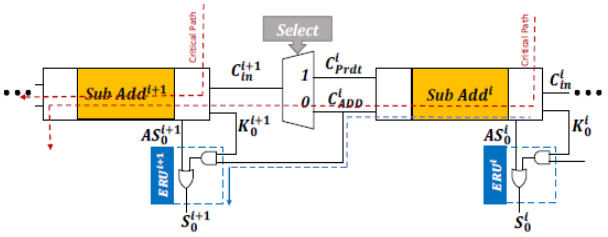

Figure 3 Proposed adder with new PPA with Error Recovery Unit (ERU)**.**

In debugging codes, unity testing has an easy way to find errors and a complex way to find a fraudulent location. Once the corrupted bit is found, its value is restored (from 0 to 1 or 1 to 0) to receive the first message.

Therefore, the error recovery unit is used to detect and correct errors using different techniques.

D. Brentkung adder:

Brentkung adder of 8-bit width are used in the sub adders. Four 8-bit Brentkung adders are used in order to implement the 32-bit adder. Intermediate prefixes in small groups are computed in parallel prefix adders and then find the large group prefixes, until all the carry bits are computed. Parallel prefix addition of the operands 'A' and 'B' of width 'n' is done in pre-processing, carry generation and post processing stages.

In preprocessing stage, carry input is generated from propagated signals for each adder. These propagation signals are given by the equation 3 & 4.

$$Pi = Ai \oplus Bi \quad (3)$$

$$Gi = Ai.\,Bi \quad (4)$$

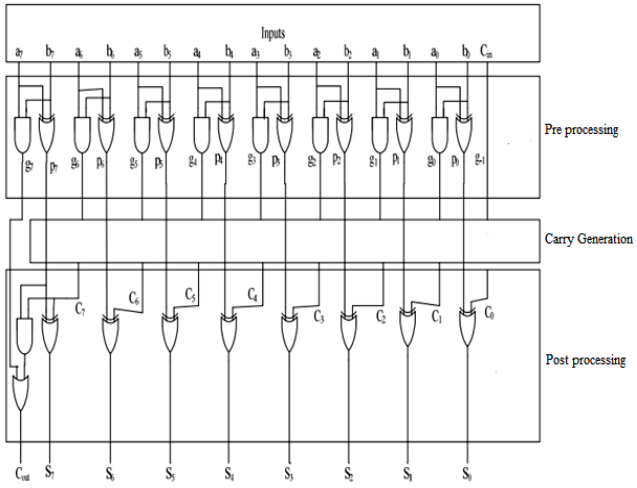

Figure 4 Overall architecture of PPA

In carry generation stage, the carries corresponding to each bit are generated. Execution is done in parallel form. After the counting of the same carriers, these are divided into smaller pieces. It uses generation and propagation as intermediate signals given by numbers 5 & 6. After the computation of carries in parallel they are divided into smaller pieces.

$$P_{(i:k)} = P_{(i:j)} \cdot P_{(j-1:k)} \quad (4)$$

$$G_{(i:k)} = G_{(i:j)} + \left(G_{(j-1:k)} \cdot P_{(i:j)}\right) \quad (5)$$

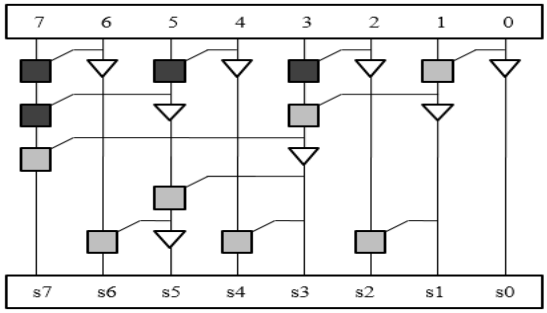

Figure 5 Carry Generation Design of 8-bit Brent kung adder**.**

This carry is generated by using different cell structures called Gray cell, Black cell and Buffer cell shown in Fig.6. By using this cell structures, final carry is calculated and are same for all parallel prefix adders but the design of carry generation is different.

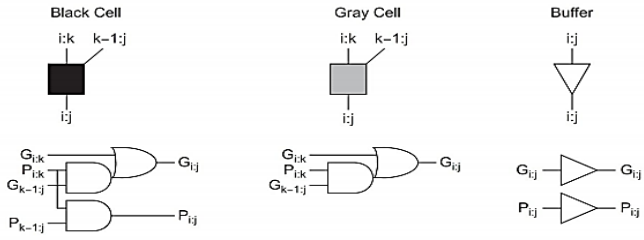

Figure 6 Black Cells, Gray Cells and Buffer cell used for carry generation stage.

The Post Processing Stage created the sum bits either by utilizing straightforward XOR entryways or by the utilization of restrictive entirety adders. In restrictive aggregate adders, for each bit position, two tentative sum will be produced and the right one will be chosen when the relevant carry for that bit arrives.

E. New parallel prefix adder:

A parallel prefix adder is used to improve the delay when of the adder when compare to the existing adder. This parallel prefix adder also contains three stages. Processing, carry generation stage and post processing stage. it is similar to the Brent kung adder but the carry generation stage is different. In this adder also contains black cells and gray cells but the design is different. The 8 bit proposed parallel prefix adder is shown in the Fig. 7.

Hence by using parallel prefix adder like Brent kung and proposed parallel prefix adder in place of ripple carry adder we can reduce the delay when compare to the existing adders

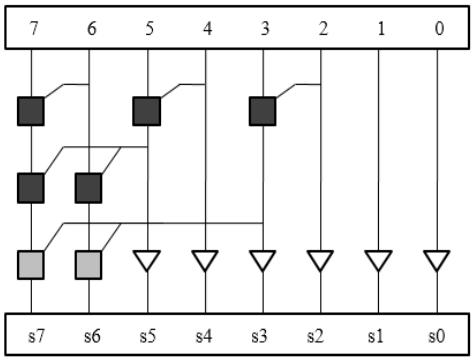

Figure 7 Proposed Parallel Prefix Adder

III. RESULTS AND DISCUSSION

For experimentation, a random 32-bit data's are experimented on 2 and 4 block sizes. The simulation is performed in Xlinx IDE. The RTL schematic is shown in figure 8. The layout design for the proposed architecture is viewed in figure 9. The simulation output is shown in figure 10.

The error rate(ER) metric is considered for analysing the proposed scheme more specifically Relative Error Distance Rate [RED rate]. The RED for 32-bit Adder with two separate block sizes 2 and 4 are projected in TABLE I.

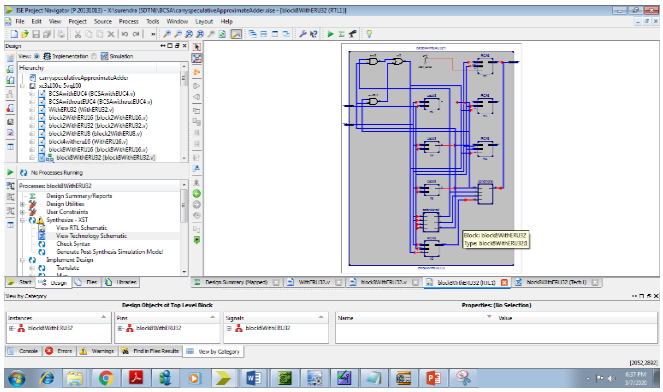

Figure 8 RTL Schematic

The metrics used for analysing error rate with 2 and 4 block sizes are depicted in TABLE I, despite of the study being performed on different block sizes viz., 2, 4 ,8 and 16. The error rate is extracted for 2^{16} unvarying random 8-bit numbers (16- and 32-bit) adders shown in figure 11. Accuracy rises with the increase in block size for adders. The percentages of the summation results with RED rate for 8-bit adders with the block size of 4 is depicted in TABLE II.

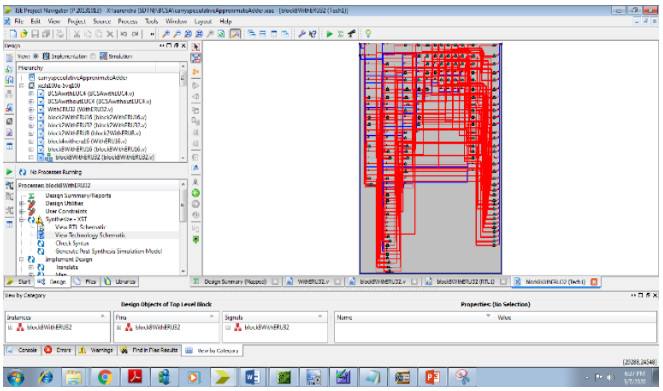

Figure 9 Layout Schematic

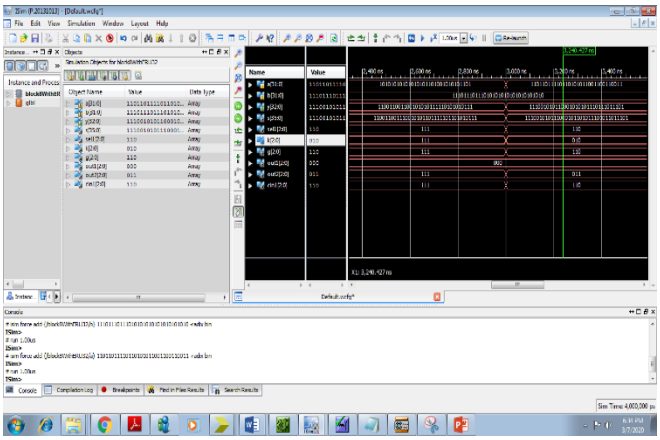

Figure 10 Simulation output

The RED is extracted by employing without '1/|*N*|' term. As the values of TABLE II show the outputs of the BCSA without ERU and with ERU, which produced 100% accuracy except for 5% and 10% RED rate.

Table 1 Accuracy Results for 32-Bit Adders

Proposed BCSA		32-bit date
Adder Type	Block Size	ER (%)
BCSA$_{ERU}$	2	74.66
BCSA$_{ERU}$	4	16.66
BCSA	2	87.62
BCSA	4	45.94

Table 2 Results of 32- bit Adder with RED

RED(%)	≤ 5%	≤ 10%	≤ 20%	≤ 50%	≤ 100%
BCSA WITHOUT ERU	94.50 %	94.50 %	100%	100%	100%
BCSA WITH ERU	100%	100%	100%	100%	100%

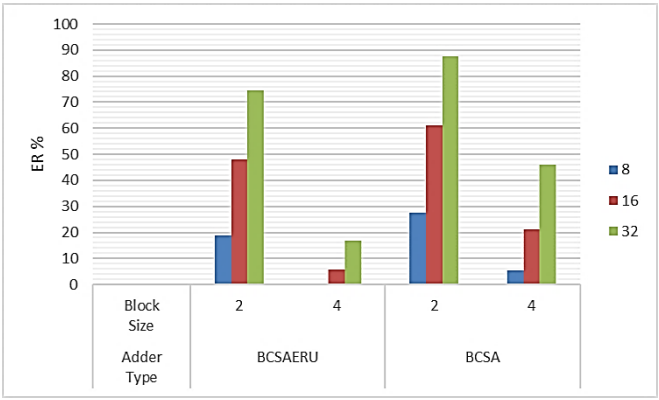

Figure 11 ER for 8 bit, 16 bit and 32 bit adders.

IV. CONCLUSION

In this case, we proposed a hypothetical block-based (BCSA), which was based on separating the adder directly into unconnected blocks that work in parallel. Each block can be named after any desired type of adders. In this extension, the length of the bearing chain was reduced and used to measure the bearing. The designated logic was suggested to look at the acquisition input of each block based on the other operator insertion of current and subsequent block components. In addition, to reduce the delay we use the novel structure of the parallel startup compound where the delay and location are better compared to other similar start additives. By reducing the loss of accuracy, the error detection method and recovery method have been suggested.

V. REFERENCES

[1]. M. Pashaeifar, M. Kamal, A. Afzali-Kusha, and M. Pedram, “Approximate Reverse Carry Propagate Adder for Energy-Efficient DSP Applications,” IEEE TVLSI, vol. 26, no. 11, pp. 2530-2541, 2018.

[2]. 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.

[3]. H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural Acceleration for General-Purpose Approximate Programs, " In Proc. of Micro, pp.449-460, 2012.

[4]. M. Bilal, S. Masud, and S. Athar, “FPGA Design for Statistics-Inspired Approximate Sum-of-Squared-Error Computation in Multimedia Applications,” IEEE TCAS-II, vol. 59, no. 8, pp. 506-510, 2012.

[5]. A. B. Kahng, S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," In Proc. DAC, pp.820-825, 2012.

[6]. M. Samadi, J. Lee, D.A. Jamshidi, A. Hormati and S. Mahlke, “SAGE: self-tuning approximation for graphics engines”. Proc. Micro, pp.13-24, 2013.

[7]. Y. Kim, Y. Zhang, and P. Li. “An energy efficient approximate adder with carry skip for error resilient neuromorphic VLSI systems,” In Proc. ICCAD, pp. 130–137, 2013.

[8]. R. Ye, T. Wang, F. Yuan, R. Kumar and Q. Xu, “On Reconfiguration-Oriented Approximate Adder Design and Its Application,” In Proc. ICCAD, pp. 48-54, 2013.

[9]. B. K. Mohanty, S. K. Patel, "Area–delay–power efficient carry select adder", IEEE Trans. Circuits Syst. II Exp. Briefs, vol. 61, no. 6, pp. 418-422, Jun. 2014.

[10]. M. Kamal, A. Ghasemazar, A. Afzali-Kusha, and M. Pedram, “Improving efficiency of extensible processors by using approximate custom instructions,” in Proc. DATE, 2014. 11J. Hu and W. Qian, "A new approximate adder with low relative error and correct sign calculation," In Proc. IEEE DATE, pp. 1449-1454, 2015.

[11]. M. Shafique, W. Ahmad, R. Hafiz and J. Henkel, "A low latency generic accuracy configurable adder," In Proc. DAC, pp. 1-6, 2015.

[12]. "NanGate - The Standard Cell Library Optimization Company", 2016, Online]. Available: http://www.nangate.com/.

[13]. H. Jiang, C. Liu, L. Liu, F. Lombardi and J. Han, “A review, classification and comparative evaluation of approximate arithmetic circuits,” ACM JETCAS, vol. 13, no. 4, Article no. 60, 2017.

[14]. W. Xu, S. S. Sapatnekar, and J.Hu. "A Simple yet Efficient Accuracy Configurable Adder Design,” In Proc. ISLPED, 2017.