SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V13I6P115

Volume 13 Issue 6, 187-194, June 2026

Original Article

High-Speed Approximate Adder Architecture for Image Processing Applications

Dande Parameswara Upendranath[1], Krishnanaik Vankdoth[2]

[1]*Department of ECE, Bharatiya Engineering Science and Technology Innovation University (BESTIU), Satya Sai District, Andhra Pradesh, India.*
[2]*Department of ECE, Chaitanya Deemed to Be University, Hyderabad, Telangana, India.*

[1]*Corresponding Author : upendranath.dp@gmail.com*

Received: 17 March 2026 Revised: 16 April 2026 Accepted: 15 May 2026 Published: 27 June 2026

Abstract *- Approximate computing offers a revolutionary way to handle the increasing needs of computationally demanding tasks like video processing and recognition by permitting a certain level of error within acceptable bounds. Enhancing the adder can result in notable decreases in latency, power consumption, and area. This study's strategy improves the accuracy of the adder by approximating the moderately important region. With just a slight loss in accuracy, power efficiency is increased, and additional hardware complexity is decreased by introducing a constant correction term in the constant region. The results show that the proposed design works, with power improvements of up to 67.3% over existing designs.*

Keywords *- Low Power, Approximate Computing, Image Processing Applications.*

1. Introduction

Growing needs in the embedded and portable industries necessitate the employment of more sophisticated design methods to handle complex tasks [1, 2]. The adder is a crucial component of the critical path that influences both power consumption and duration in arithmetic operations. Battery-operated devices rapidly deplete their power due to elevated energy usage. Therefore, to execute these operations, high-speed, low-power adders are necessary [3].

Due to the intrinsic limitations of human sensory modalities, image processing programs can accommodate errors [4-6]. An emerging technique termed approximate computing is employed in these error-tolerant applications to enhance performance with minimal quality degradation. Numerous approximate adder designs that optimize area, delay, and power trade-offs are available in the literature.

Heterogeneous adders, which integrate diverse adder types to compute operand sums, have gained increasing popularity recently. This paper proposes an adder that exhibits superior error metrics than the current approximation adders, while concurrently maintaining an advantageous balance with circuit parameters such as power, area, and latency.

While previous adders offer reduced latency, their accuracy is troublesome. However, there are significant delays for approximate full adders. This work introduced a new approximate full adder that demonstrates enhanced space, speed, and power efficiency compared to the Lower-Part OR Adder (LOA). This study's primary contributions are as follows:

- Proposed approximate adder divided into four portions: constant region, approximate region, moderately significant region, and accurately significant region.
- To reduce the hardware complexity, moderately significant and approximate regions are implemented using the proposed full adder and OR gates, respectively.
- Extensive error and hardware analyses have been conducted to compare the proposed design with current architectures.
- Finally, adders are assessed in image processing applications to ensure that performance and output quality are balanced.

The rest of the paper is set up like this: Section II looks at current adder architectures, whereas Section III talks about the design of the suggested adder architecture. Section IV presents a comprehensive analysis of error and synthesis outcomes. The proposed adder is evaluated against existing designs in Section V, and the study concludes in Section VI.

2. Related Work

Many in exact adders have been recorded to lower the complexity of computations. There are four categories of approximate adders, based on the methods used, which are mentioned in the sections below.

Authors in [7-10] proposed a carry select adder, which is implemented by using the sub-adders and sub-carry engines. The approximate adder named BCSA was proposed by the authors in [7] to enhance precision. Despite resulting in a latency trade-off, select and carry signals are estimated utilizing input bits from both the current and subsequent sub-adders to enhance precision. An enhanced fragmented adder, referred to as HABA, featuring advanced carry inference logic, was presented by Hu and Qian [8]. Each sub-adder in this configuration is implemented using conventional adders. To enhance speed, the authors in [9] employed a carry-participation technique to develop a Simple Reconfigurable Adder (SARA). To enhance precision, the SARA adder is divided into small sub-adders, each constructed using RCA. An error correction unit is integrated into the structure; however, it incurs a higher hardware overhead. An adder comprising a carry anticipation unit, a primary sub-adder, and dual carry engines for managing both carry "0" and carry "1" situations was initially introduced by Lin et al. [10]. This design typically necessitates greater power due to its complex hardware architecture.

Authors in [11-16] proposed an approximate full adder, which is created by dividing the adder into exact and approximate regions. The adder is segmented into approximate and exact sections by the LOA, initially introduced by Mahdiani et al. [11]. The carry-in signal for the exact segment is generated by the ANDing of the higher bits of the approximate component, whereas the approximate section, managed by OR gates, addresses the lower bits. Thus, the cumulative delay between the AND gate and the exact adder influences the latency of the LOA. The approximate section is subsequently separated into two segments named as LOCA [12]. The other segment is executed using the LOA concept, while the total of the lower approximate segment is assigned a duplicate of the input value. As a result, it exhibits a greater error rate but possesses lower hardware complexity compared to the LOA. Balasubramanian et al. [13] introduced three N-bit adders. Each adder comprises two components: an exact segment and an approximation section, the latter possessing a distinct bit count. The 22-10 arrangement, allocating 10 bits for inaccurate components and 22 bits for precise components, is the most efficient. The sum of the important bits is derived by performing an OR operation on the bits, while the least significant eight bits in the approximation portion are linked to Vdd. The carry generated by ANDing corresponding bits is utilized to obtain the greatest significant bit total. The LOA concept calculates the carry-in to a specific place to minimize errors. The Garg [14] presents an approximation of a full adder named AFA through the simplification of the sum and carry Boolean equations. The suggested AFA requires a minimal number of gates and has a low chance of making errors. Also, a new energy-efficient approximation RCA adder architecture that uses the new AFA is introduced. To improve the accuracy of carry anticipation, Patel and Garg presented a reconfigurable adder that was developed using a particular technique. While the lowest part is implemented utilizing the approximation idea, the most important part is dependable and is carried out using a traditional adder [15]. An adder breaks into two equivalent parts, according to Shahrokhi et al., precise adders are used to calculate the total of the most important segment, whereas imprecise adders are used to calculate the sum of the lower segment [17]. A unique N-bit dual adder-based low-latency adder with three distinct segments was proposed by Kim & Hyoju [16]. The sum was calculated using a bitwise OR operation in the approximate region. While the sum in a Lower Sub-Adder (LSA) is computed using exact logic, the carry-in for LSA is established by performing an AND operation on the highest bits of the approximation area. With the introduction of a novel prediction logic in their study, the carry-in for the Upper Sub-Adder (USA) is established, allowing the lower sub-adder LSA and upper sub-adder USA to function simultaneously. For the Upper Sub-Adder (USA), a comparison between accurate carry-in and prediction-based carry-in is made. If they are different, the problem is fixed using a straightforward recovery method. This study [24] introduces a novel design methodology via the creation of an optimized approximate Adder, meticulously engineered for proficient error compensation in unsigned integer operations. The suggested design enhances computing precision while optimising hardware resource efficiency, decreasing power consumption, and minimising critical path delay.

Authors in [18-20] proposed a segmented adder, which is created by dividing the adder into different segments. Mohapatra et al. [18] introduced the N-bit Equal Segmented Adder (ESA), which is separated into N/z segments, where "z" indicates the size of the sub-adder. The carry-in of each sub-adder is disregarded, allowing the sum to be calculated independently, ignoring the distribution of the carry between later sections. Consequently, these adders are faster than the current ones but less precise. Zhu et al. [19] introduced the ETA1 adder. It divides the adder into sections for the Lower Important Bits (LIB) and Most Important Bits (MIB), with a standard adder being used to implement the MIBs. Without using carry prediction logic, addition is done in the LIB from left to right. To increase accuracy, it is recommended to employ a method that computes the sum regularly when both input bits are "0" or unequal. But when both input bits are "1," the checking ends, and the sum bits are set to "1" moving forward. Control logic circuits are used to carry out these operations, which drives up hardware costs. ETA-II has two distinct sections: sum and carry generators. Carry generate blocks are used to estimate the carry, which enhances the hardware circuitry overall. As the carry spreads across succeeding blocks, accuracy improves. There is a longer delay than ETA-I since the carry is sent between adjacent segments. Although this concept is more complicated to implement, it is bigger and more accurate than ETA-I when compared to ESA. Kumar et al. [20] introduced

an improved N-bit segmented adder that speculates the carry for each segment to achieve lower error and minimal area overhead, in contrast to the ESA.

Authors in [21, 22] proposed a speculative adder, which is created by predicting the carry using prior bits. Lu et.al. proposed an adder [21] that predicts carry based on prior bits to decrease delay. As fewer bits are needed to calculate the carry, the delay reduces. As proposed by Verma et al. [22], an adder named ACA creates the carry by using the earlier bits "d." There is a higher area overhead and consumes huge power since (N-d) carries created blocks with a size of "d" bits are required.

3. Proposed Work

While segmented adders such as ESA, ETA, and ETAII offer reduced latency, their accuracy is troublesome. However, there are significant delays for approximate full adders like LOA and LOCA. This work presents a new approximate 1-bit full adder, which is used to design an N-bit approximate adder that consumes less energy than earlier designs. The construction of the suggested adder is represented in Figure 1, where "N" stands for adder size. The architecture is composed of four distinct regions: The Accurate Significant Region (ASR), Moderate Significant Region (MSR), Approximate Region (AR), and Constant Region (CR). While the MSR, LSR, and CR have widths of "P" bits, where P equals N/4-1, the ASR spans (N-3P) bits. The ASR uses precise full adders to ensure precision while approximating the remaining regions. Given that the CR and LSR have a minimal impact on the final output, rather than performing actual computations, which are replaced with constant terms and OR gates, respectively.

Figure 2 depicts the proposed 16-bit adder architecture, which is named the Hybrid Approximate Adder 1(HAA1). For a 16-bit adder, the CR size is 3 bits, as shown in Figure 2, and these three bits are substituted with zeros. Because the CR doesn't require any hardware operations, it significantly reduces both area utilization and power consumption. Since the LSR and MSR have a greater influence on the final result than the CR, calculations in this area are justified. Instead of employing full adders, the LSR uses OR gates to calculate the 3-bit sum while ignoring carry propagation, whereas the MSR uses a proposed approximate full adder, which is described in section III.A. This method minimizes power consumption and improves delay by reducing the critical path.

Further to improve the average error of the proposed adder, a constant correction term was used in the CR region instead of zeros, and it was named HAA2. Based on the methodology described in [23], which uses the average value of all potential input combinations, this constant term is calculated. The computed average correction value is '3'.

3.1. Proposed Approximate Full Adder

A traditional full adder generates two binary outputs, the Sum and the Carry-out, from three binary input signals, usually represented as Y1, Y2, and the carry-in (Cin). The Carry-out takes into consideration any overflow to the next higher bit position, but the Sum output shows the least significant bit of the three inputs added together. Although this precise behavior guarantees accuracy for any possible combination of inputs, it may necessitate more complicated logic, increasing consumption of power, delay, and area.

The suggested approximate full adder, on the other hand, purposefully loosens the exact correctness requirement in order to increase hardware efficiency. In particular, only two clearly defined input circumstances are permitted for mistakes to arise in the design. In one instance, the calculated output is one unit less than the precise value, with an error distance of -1. In the alternative scenario, the output is one unit above the precise value, as indicated by the error distance of +1.

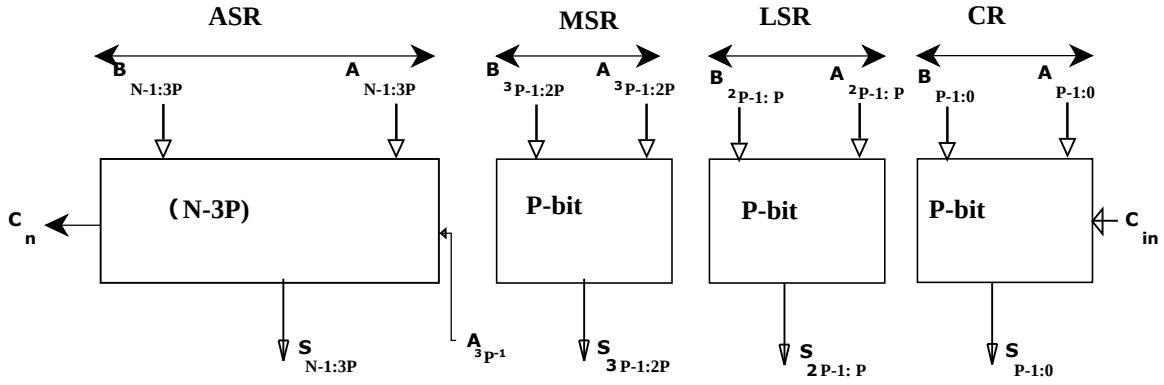

Fig. 1 Generalized proposed approximate adder

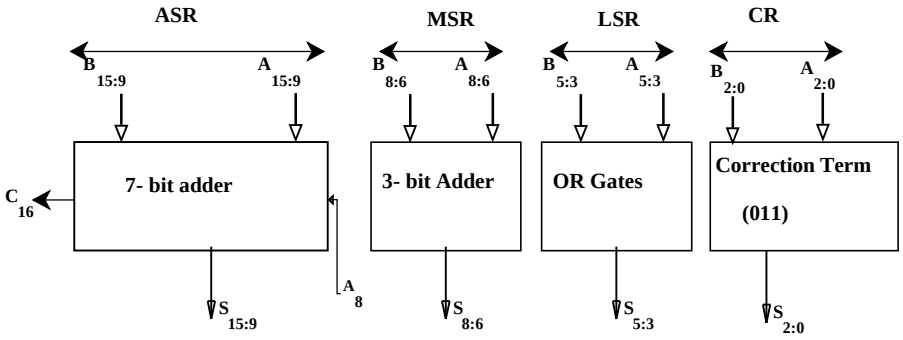

Fig. 2 Proposed 16-bit approximate adder

The approximate adder allows for simplifications in the underlying logic while maintaining adequate numerical accuracy by strictly limiting the magnitude and recurrence of these errors.

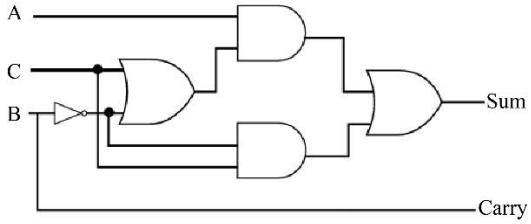

Fig. 3 Proposed 1-bit full adder

Table 1. Truth table of proposed 1-bit full adder

A	B	C	Sum	Carry
0	0	0	0	0
0	1	0	**0**	**1**
0	0	1	1	0
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1
1	0	1	**1**	**0**
1	1	1	1	1

Figure 3 displays the logic diagram that depicts the architecture of the suggested approximate complete adder. It is possible to see the streamlined signal routes and fewer gates. Additionally, the truth table presented in Table 1 summarizes the adder's functional behaviour across all input combinations, highlighting both the exact and approximate output situations. Equations (1) and (2) describe the formal derivation and presentation of the associated Boolean equations that define the Sum and Carry outputs of the approximation design, respectively.

$$Sum = A(C + \bar{B}) + C\bar{B} \quad (1)$$

$$Carry = B \quad (2)$$

4. Results and Discussion

Error and Hardware analysis are used to analyse the suggested design with the previous designs, and the results are mentioned in Sections 4.1 and 4.2.

4.1. Error Analysis

The performance of different adder architectures, including the proposed design, was assessed by a thorough error analysis utilizing two million randomly generated input vectors in the MATLAB environment. Table 2 summarizes the findings of a quantitative evaluation of these architectures' accuracy using common error metrics. By attaining the lowest values of standard metrics such as NMED, MRED, and average error, the comparison analysis shows that the proposed adder consistently performs better than most of the existing designs.

The optimization method used in both the MSR and AR, as well as the addition of a constant correction term, are principally responsible for this increase in accuracy. These methods lessen the total departure from precise computing and successfully minimize the propagation of errors. When taking into account overall design trade-offs, some adder designs are less efficient even when they show slightly lower NMED, MRED, or average error.

However, this is at the cost of lower power savings. When compared to existing methods, the proposed design HAA1 provides significant improvements in error metrics, including gains of up to 92.19% in NMED and 65.39% in MRED.

Additionally, compared to previous designs, an improvement of up to 99.8% in average error is noted, except for the IFA architecture, which has greater power requirements. Delivering greater performance thus depends on the optimization of the MSR and LSR in tandem with the application of a constant corrective term. The evaluation of the 32-bit adder shown in Table 2 further supports this, showing that the suggested design once again has the lowest error metric values among all the studied designs.

4.2. Hardware Analysis

In order to assess the hardware properties of the 16-bit adder architectures, they were synthesized using Cadence RTL Compiler version 7.1 and described using Verilog HDL. Every design was put into practice utilizing the TSMC 90 nm

technology node. Table 3 presents the circuit-level parameters of the reference designs and the proposed adder. The results show that the suggested adder performs competitively in all assessed measures. The lack of specialized hardware for sum generation in the CR and lower hardware complexity in the AR and MSR are the main causes of its increased area efficiency.

Furthermore, a considerable decrease in power consumption results from the use of approximation techniques in the MSR and AR, as well as the insertion of a constant correction term that does not require additional hardware overhead. The proposed architectures (HAA1 and HAA2) deliver power reductions of up to 67.3% when compared to previous adder designs.

The suggested adder offers a better balanced trade-off by outperforming other crucial parameters, including MRED, average error, area, NMED, and power dissipation, even if certain SAAR-based designs show a shorter propagation delay.

Table 2. Error analysis

S.No	Design	16-bit			32-bit		
		NMED(10^{-3})	MRED(10^{-3})	Avg Error	NMED(10^{-3})	MRED (10^{-3})	Avg Error
1	LOCA [12]	0.46	0.14	4.23	0.0029	0.00049	$1.64*10^{4}$
2	BCSA-4 [7]	5.59	0.87	271.4	4.1	0.62	$1.77*10^{7}$
3	HABA [8]	7.69	1.49	1010.1	15.5	1.49	$6.69*10^{7}$
4	SARA [9]	0.19	5.49	-2	16.39	10.63	-13400
5	SAAR [20]	1.5	4	-61.72	0.0033	0.0041	$4.20*10^{6}$
6	ACLA_I [15]	0.239	0.661	-31.366	0.0018	0.0025	-8093
7	IFA [17]	0.544	1.5	0.515	0.0042	0.0059	-24.45
8	HAA1	0.6	1.9	5.07	0.002	0.004	65.32
9	HAA2	0.69	1.9	1.87	0.002	0.005	58.32

Table 3. Synthesis analysis

S.No	Adder	16-bit				32-bit			
		Area (um^2)	Power (uW)	Delay (pS)	PDP (fJ)	Area (um2)	Power (uW)	Delay (pS)	PDP (fJ)
1	LOCA [12]	155.23	5.21	1306	6.80426	373.96	14.61	3443	50.30223
2	BCSA-4 [7]	282.53	7.49	2015	15.1642	612.02	18.91	5349	102.14068
3	HABA [8]	328.39	8.13	1060	8.6072	689.66	17.43	1060	19.04546
4	SARA [9]	568	15.1	1010	15.251	613.16	16.33	7300	119.209
5	SAAR [20]	261	6.75	747	5.04225	553.9	21.4	1265	27.071
6	ACLA_I [15]	275.1	7.1	1088	7.7248	550.36	14.1	1999	28.1859
7	IFA [17]	195.56	6.54	3103	20.31	395.13	13.60623	6232	84.794038
8	HAA1	148.2	4.93	1198	5.89	358.1	13.81	2396	25.6114
9	HAA2	148.2	4.93	1198	5.89	358.1	13.81	2396	25.6114

The benefits of the proposed method also apply to higher-width designs. Compared to current designs, the 32-bit approach uses less power and takes up less space, as indicated in Table 3.

A constant correction term in the carry region and improved approximation techniques in the AR and MSR make these enhancements possible. All things considered, the proposed adder outperforms all other previous adder architectures in terms of power.

5. Applications

Image sharpening is a popular image enhancement technique that enhances perceived visual quality by highlighting high-frequency components, as mentioned in [20]. The approach improves the image's edges, contours, and fine structural features by boosting the contribution of these elements. For a given input image Z, the sharpening procedure highlights intensity changes to provide a more distinct and lucid representation. The following is a description of the algorithm used for this enhancement process:

$$O(u,v) = 2Z(u,v) - W \quad (3)$$

Where

$$W = 1/273 \sum_{r=-2}^{2} \sum_{i=-2}^{2} K(r+3, i+3) Z(u-r, v-i)$$

and K is defined as

$$K = \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Equation (4) uses cumulative additions to estimate the parameter W. The PSNR is calculated for the proposed design and existing adder implementations in order to assess the quality of the suggested adder architecture. Convolution kernels of size 5x5 are used to apply the suggested technique on a group of standard benchmark images for the evaluation.

Table 4 provides a summary of the performance metrics that were obtained for the various adders. Table 4 shows that the proposed adder regularly outperforms all other designs in terms of PSNR. When compared to previous architectures, the proposed adder's reduced NMED and MRED demonstrate its superior error characteristics, which are responsible for this increase in PSNR.

Table 4. PSNR values for various approximate adders

S.No	Adder	PSNR(dB)	
		Lena	Cameraman
1	LOCA [12]	27.49	27.6
2	BCSA-4 [7]	19	19.8
3	HABA [8]	14.2	15.2
4	SARA [9]	13.41	15.3
5	SAAR [20]	30.04	30.83
6	ACLA_I [15]	38.86	40.24
7	IFA [17]	42.38	42.9
8	HAA1	40.13	41.29
9	HAA2	40.48	41.49

(a) Input (b) Exact

(c) Proposed

Fig. 4 (a-c) Output images of exact and proposed designs

Additionally, the images produced using the exact adder and those processed using the proposed approximation adder design are shown in Figures 4(a)-(c).

The close visual similarity between the two image sets shows that, even with computational approximations, the proposed method preserves good perceptual image quality. This discovery emphasizes the proposed method's intrinsic fault tolerance, which makes it ideal for error-resilient applications.

6. Conclusion

The goal of the suggested work is to improve the circuit characteristics of the LOCA-8 adder so that it may be used for image processing. The adder uses an OR-based approximation method in the approximate region and an approximate full adder in the moderately significant region. The suggested adder is better than traditional designs in terms of less area, power use, and latency, as shown by error and hardware analysis. Additionally, image processing results validate the suggested adder's practical utility.

References

[1] Sparsh Mittal, "A Survey of Techniques for Approximate Computing," *ACM Computing Surveys*, vol. 48, no. 4, pp. 1-33, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[2] Srinivasan Narayanamoorthy et al., "Energy-Efficient Approximate Multiplication for Digital Signal Processing and Classification Applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 6, pp. 1180-1184, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[3] Honglan Jiang et al., "A Review, Classification, and Comparative Evaluation of Approximate Arithmetic Circuits," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 4, pp. 1-34, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[4] M. Priyadharshni, and S. Kumaravel, "Design of Imprecise Multipliers by Using Approximate Technique for Error Resilient Applications," *Journal of Circuits, Systems and Computers*, vol. 30, no. 7, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Qiang Xu, Todd Mytkowicz, and Nam Sung Kim, "Approximate Computing: A Survey," *IEEE Design & Test*, vol. 33, no. 1, pp. 8-22, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Walaa El-Harouni et al., "Embracing Approximate Computing for Energy-Efficient Motion Estimation in High Efficiency Video Coding," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Lausanne, Switzerland, pp. 1384-1389, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[7] Farhad Ebrahimi-Azandaryani et al., "Block-Based Carry Speculative Approximate Adder for Energy-Efficient Applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 1, pp. 137-141, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[8] Junjun Hu, and Weikang Qian, "A New Approximate Adder with Low Relative Error and Correct Sign Calculation," *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, France, pp. 1449-1454, 2015. [Google Scholar] [Publisher Link]

[9] Wenbin Xu, Sachin S. Sapatnekar, and Jiang Hu, "A Simple Yet Efficient Accuracy-Configurable Adder Design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 6, pp. 1112-1125, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[10] Ing-Chao Lin, Yi-Ming Yang, and Cheng-Chian Lin, "High-Performance Low-Power Carry Speculative Addition with Variable Latency, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1591-1603, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] H. R. Mahdiani et al., "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 4, pp. 850-862, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[12] Ayad Dalloo, Ardalan Najafi, and Alberto Garcia-Ortiz, "Systematic Design of an Approximate Adder: The Optimized Lower Part Constant-or Adder," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 8, pp. 1595-1599, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[13] Padmanabhan Balasubramanian et al., "An Approximate Adder with a Near-Normal Error Distribution: Design, Error Analysis and Practical Application," *IEEE Access*, vol. 9, pp. 4518-4530, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[14] Bharat Garg, "Energy Efficient Gaussian Filtering for Multimedia Applications using Novel Approximate Adders," *Sadhana*, vol. 46, pp. 1-9, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] Bharat Garg, and Sujit Kumar Patel, "Reconfigurable Carry Look-ahead Adder Trading Accuracy for Energy Efficiency," *Journal of Signal Processing Systems*, vol. 93, pp. 99-111, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Hyoju Seo, and Yongtae Kim, "A Low Latency Approximate Adder Design Based on Dual Sub-Adders with Error Recovery," *IEEE Transactions on Emerging Topics in Computing*, vol. 11, no. 3, pp. 811-816, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Seyed Hossein Shahrokhi et al., "A Novel High-Speed and Low-PDP Approximate Full Adder Cell for Image Blending," *Mathematics*, vol. 11, no. 12, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] Debabrata Mohapatra et al., "Design of Voltage-Scalable Meta-Functions for Approximate Computing," *2011 Design, Automation & Test in Europe*, Grenoble, France, pp. 1-6, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[19] Ning Zhu et al., "Enhanced Low-Power High-Speed Adder for Error-Tolerant Application," *2010 International SoC Design Conference*, Incheon, Korea (South), pp. 323-327, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[20] Uppugunduru Anil Kumar et al., "A High-Speed and Power-Efficient Approximate Adder for Image Processing Applications," *Journal of Circuits, Systems and Computers*, vol. 31, no. 3, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[21] Shih-Lien Lu, "Speeding up Processing with Approximation Circuits," *Computer*, vol. 37, no. 3, pp. 67-73, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ajay K. Verma, Philip Brisk, and Paolo Ienne, "Variable Latency Speculative Addition: A New Paradigm for Arithmetic Circuit Design," *Proceedings of the Conference on Design, Automation and Test in Europe*, Munich Germany, pp. 1250-1255, 2008. [CrossRef] [Google Scholar] [Publisher Link]

[23] U. Anil Kumar, Sumit K. Chatterjee, and Syed Ershad Ahmed, "Low-Power Compressor-Based Approximate Multipliers with Error Correcting Module," *IEEE Embedded Systems Letters*, vol. 14, no. 2, pp. 59-62, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Prateek Goyal, and Sujit Kumar Sahoo, "EOHEAA: Error-Optimized Hardware-Efficient Approximate Adder for Energy-Aware Error-Resilient Applications," *Integration*, vol. 108, 2026. [CrossRef] [Google Scholar] [Publisher Link]