Color Image Enhancement Using Ripple-Carry Approximate Adders

Y.S.S Sriramam
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
ysriramam@gmail.com

Bala Sindhuri Kandula
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
k.b.sindhuri@gmail.com

Triveni Chintha
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
chinthatriveni65@gmail.com

Dasari M N K Shankar
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
shankardasari1530@gmail.com

Vaddapalli Vinesh
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
vineshvaddipalli@gmail.com

Thoram sairam durga ganesh
Department of E.C.E
Center of Excellence in VLSI Design
S.R.K.R Engineering College(A)
Bhimavaram,India
sairamthoram1@gmail.com

***Abstract*—Image enhancement is a key task in image processing applications such as medical imaging, satellite imaging, and consumer photography. In this paper, we propose an approach for real-time image enhancement using ripple-carry adders (RCAs) and their approximate variants (A1–A12). The ripple-carry adder is integrated within the image processing hardware module to modify pixel intensities across Red, Green, and Blue (RGB) channels. Approximate adders are introduced to reduce hardware complexity, power consumption, and delay while maintaining acceptable image quality. Experimental results including PSNR, SSIM, BER, and error metrics are reported for each approximate adder design.**

***Index Terms*—Image Enhancement, Ripple-Carry Adder, Approximate Computing, VLSI, Verilog, RGB.**

Introduction

I. Introduction

Image enhancement is an important part of digital image processing and is widely used in areas like medical imaging, remote sensing, photography, and security systems [1]. The main goal is to make images clearer and easier to understand, either for people or for automated systems [2], [3]. For example, in medical imaging, enhancement helps doctors find small details that may not be visible in the original scan [4]. In environmental monitoring or city planning, it is used for tasks like flood mapping from aerial photos [5].

One of the most common techniques for improving image contrast is histogram equalization. Even though it works well, it is not easy to implement on hardware because it needs a lot of computation [6]. This leads to high power usage and more chip area, which is a big problem for low-power devices like IoT cameras and portable embedded systems [7].

To deal with these limitations, approximate computing has become a popular approach. This method takes advantage of the fact that many image processing applications can tolerate small errors without affecting the visual quality noticeably [8], [9]. By using approximate arithmetic circuits, it is possible to cut down power consumption, reduce delay, and save hardware area [10]. These trade-offs work well in applications like image enhancement, where small deviations from the exact result are acceptable [11].

Approximate adders, especially Ripple-Carry Approximate Adders (RCAAs), have been suggested for faster hardware-based image enhancement. The regular Ripple-Carry Adder (RCA) is simple in design, but it suffers from long carry propagation delays [12]. Approximate RCAs reduce this delay by allowing a small loss in accuracy, which improves speed and lowers power usage [13], [14]. Because of this, they are useful in heavy operations such as histogram equalization [15], [16].

Histogram equalization is great for improving contrast, but it needs steps like building a histogram and calculating the cumulative distribution function (CDF), which are hardware-intensive [17]. Research shows that using approximate adders instead of exact ones in these steps can reduce hardware complexity and power consumption while still keeping the image quality within acceptable limits [18], [19].Similar results have been reported in other error-tolerant applications, such as approximate Multiply-Accumulate (MAC) units for filtering tasks in image and video processing [20].

However, a research gap still exists in the application of approximate arithmetic to **color image enhancement**. Most prior studies have concentrated on grayscale images or general DSP kernels, with limited analysis of how different approximate adder architectures impact visual quality in color enhancement tasks. Moreover, the trade-offs between image quality metrics and hardware efficiency have not been systematically studied for real-time, resource-constrained systems such as mobile

devices, IoT cameras, and embedded vision platforms.

The main Contributions of this work are:

- Approximate Computing for Color Image Enhancement Using Ripple-Carry Approximate Adders
- Comparative evaluation of different approximate adder designs, highlighting trade-offs between accuracy and efficiency.
- Demonstration of the suitability of approximate adders for real-time image processing in resource-constrained environments.

The rest of the paper is organized as follows: Section II details the analytical formulation of the twelve distinct approximate full adder variants (A1-A12) and their underlying mathematical principles. Section III presents the visual quality assessment of the enhanced images, demonstrating the perceptual impact of each adder. Section IV provides a statistical analysis using Probability Density Function (PDF) and Cumulative Distribution Function (CDF) to quantitatively evaluate the impact on image intensity distributions. Section V explains the mathematical foundation of the histogram equalization process. Section VI discusses the results, including error and quality metrics, and a comparative performance analysis. Section VII elaborates on the strategic LSB approximation technique responsible for the significant PSNR enhancement. Finally, Section VIII concludes the paper with a summary of findings and suggests directions for future work.

II. Related Work

Approximate computing employs deliberate arithmetic simplifications to achieve significant reductions in energy consumption and circuit area. As fundamental building blocks in arithmetic units, full adders have been extensively studied for approximation. This section presents twelve representative variants (A1–A12) with mathematical formulations that balance computational accuracy with hardware efficiency.

A. Conditional Carry Transmission(A1)

$$S_{\mathrm{A1}} = \begin{cases} C_{\mathrm{in}}, & A = B \\ 0, & \text{otherwise} \end{cases} \qquad C_{\mathrm{A1}} = \begin{cases} A, & A = B \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

This implementation only activates when inputs A and B are equal, otherwise producing zeros to minimize switching activity.

B. Selective Complement Logic(A2)

$$S_{\mathrm{A2}} = (A \oplus B) \cdot \overline{A} \cdot \overline{C_{\mathrm{in}}} \tag{2}$$

$$C_{\mathrm{A2}} = (A \oplus B) \cdot A \cdot C_{\mathrm{in}} \tag{3}$$

Variant A2 generates outputs exclusively during input mismatches ($A \neq B$) with specific input polarities, reducing dynamic power dissipation.

C. Hybrid XOR-XNOR Architecture(A3)

$$S_{\mathrm{A3}} = (A \odot B) \cdot \overline{A} + (A \oplus B) \cdot \overline{C_{\mathrm{in}}} \tag{4}$$

$$C_{\mathrm{A3}} = (A \odot B) \cdot A \tag{5}$$

This design combines both XOR and XNOR functionalities, allowing conditional carry propagation during mismatch conditions.

D. Input-Dominant Propagation(A4)

$$S_{\mathrm{A4}} = (A \odot B) \cdot C_{\mathrm{in}} + \overline{A} \cdot B \cdot C_{\mathrm{in}} \tag{6}$$

$$C_{\mathrm{A4}} = A \tag{7}$$

A4 permanently sets the carry output to input A, completely eliminating traditional carry generation logic.

E. Multiplexer-Based Approximation(A5)

$$S_{\mathrm{A5}} = \begin{cases} A, & A = 1 \\ B, & \text{otherwise} \end{cases} \qquad C_{\mathrm{A5}} = A \tag{8}$$

This design works like a simple 2:1 multiplexer instead of a normal adder, where input A mainly controls both outputs

F. Conditional Inversion Logic(A6)

$$S_{\mathrm{A6}} = \begin{cases} \overline{A}, & A = 1 \\ B \cdot C_{\mathrm{in}}, & \text{otherwise} \end{cases} \qquad C_{\mathrm{A6}} = A \tag{9}$$

In A6, some error is added on purpose by giving the opposite of A in certain cases, and in other cases it just uses the AND of B and C_{in}.

G. Equality-Driven Computation(A7)

$$S_{\mathrm{A7}} = (A \odot B) \cdot C_{\mathrm{in}} + (A \oplus B) \cdot B \tag{10}$$

$$C_{\mathrm{A7}} = (A \odot B) \cdot A + (A \oplus B) \cdot C_{\mathrm{in}} \tag{11}$$

This design calculates the outputs depending on whether the inputs are equal, while still keeping some similarity to exact addition.

H. Selective Carry Bypassing(A8)

$$S_{\mathrm{A8}} = (A \odot B) \cdot C_{\mathrm{in}} + \overline{A} \cdot B \cdot C_{\mathrm{in}} \tag{12}$$

$$C_{\mathrm{A8}} = (A \odot B) \cdot A + (A \oplus B) \cdot C_{\mathrm{in}} \tag{13}$$

"A8 uses C_{in} when the inputs are the same, but it handles mismatches differently than A7."

"This last design uses two terms, making it work more like the normal majority logic, while still keeping it efficient in hardware."

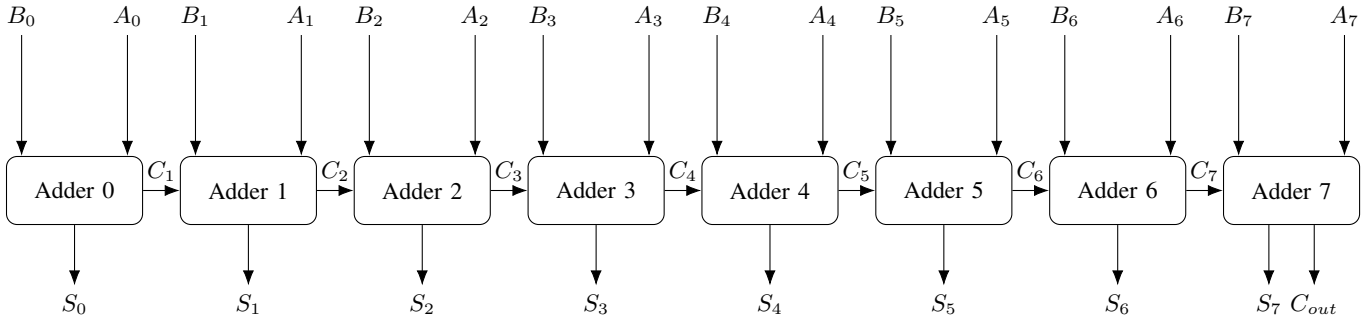


Fig. 1: 8-bit Ripple Carry Adder.

I. Gate-Level Optimized Forms(A9)

These designs use complex Boolean logic, which is easier to understand at the gate level rather than writing them as simple equations

$$S_{\mathrm{A9}} = (\overline{A} \cdot \overline{B} \cdot \overline{C_{\mathrm{in}}}) + (A \cdot B \cdot C_{\mathrm{in}}) + (\overline{A} \cdot B \cdot \overline{C_{\mathrm{in}}}) + (A \cdot \overline{B} \cdot \overline{C_{\mathrm{in}}}) \quad (14)$$

$$C_{\mathrm{A9}} = (A \cdot B) + (A \cdot C_{\mathrm{in}}) + (B \cdot C_{\mathrm{in}}) \quad (15)$$

(A10)

$$S_{\mathrm{A10}} = (\overline{A} \cdot \overline{B}) + (A \cdot B \cdot C_{\mathrm{in}}) + (\overline{A} \cdot B \cdot \overline{C_{\mathrm{in}}}) \quad (16)$$

$$C_{\mathrm{A10}} = (A \cdot B) + (A \cdot C_{\mathrm{in}}) \quad (17)$$

Both designs try to keep a good balance between the number of transistors and correct logic, so they give efficient results while still being fairly accurate. These structures are optimized automatically during synthesis, which makes them more hardware-friendly compared to complex exact adders.

J. Complement-Enhanced Hybrid(A11)

$$S_{\mathrm{A11}} = (A \oplus B) \cdot \overline{A} \cdot \overline{C_{\mathrm{in}}} \quad (18)$$

$$C_{\mathrm{A11}} = (A \odot B) \cdot A + (A \oplus B) \cdot C_{\mathrm{in}} \quad (19)$$

A11 checks for mismatched inputs and sometimes uses their complements. This helps reduce circuit activity while still allowing some carry to pass through.

K. Dual-Mode Approximation(A12)

$$S_{\mathrm{A12}} = (A \odot B) \cdot (\overline{A} \cdot \overline{B} \cdot C_{\mathrm{in}}) + (A \cdot B) \cdot \overline{C_{\mathrm{in}}} \quad (20)$$

$$C_{\mathrm{A12}} = (A \odot B) \cdot A + (A \oplus B) \cdot C_{\mathrm{in}} \quad (21)$$

These different designs give us the option to choose the best one depending on what is important for the application, like saving power, reducing chip area, or keeping the accuracy high. For example, in a low-power IoT device, we might pick an adder that uses less energy even if it is slightly less accurate. On the other hand, for applications where image quality is critical, we would go for a design with better accuracy even if it takes a little more hardware.

Figure 1 shows the Ripple Carry Adder (RCA) structure. In this design, the usual full adders are arranged in series, and in some cases, approximate adders are used instead of exact ones. This replacement is done to cut down hardware complexity, power consumption, and delay. Even though some accuracy is sacrificed, the results are still good enough for color image processing tasks like enhancement.

In the RCA, the carry from each stage moves to the next stage, so the delay keeps adding up as we go through the bits. Finally, the last stage produces both the most significant sum bit (S_7) and the final carry output (C_{out}). This structure is simple, but its speed depends on how many bits we are adding. That is why using approximate adders can make it faster while still giving acceptable image quality

III. Methodology

The proposed system flow for RCA-based image enhancement and analysis, as shown in Fig. 2, begins with an input RGB image, which first undergoes grayscale conversion to simplify processing. This single-channel image then undergoes histogram equalization, an algorithm implemented in Verilog specifically designed to improve overall contrast by redistributing pixel intensities. The core arithmetic operations within this implementation, particularly the cumulative sum calculations, are processed using two distinct adder types to observe variations in performance and result. The first utilizes a precise Ripple Carry Adder (RCA) to establish a baseline for accurate enhancement, while the second employs a series of twelve different Approximate Adders (designated A1 through A12) to introduce controlled computational inaccuracies and evaluate their trade-offs. Functional verification of the entire Verilog hardware design is rigorously carried out through simulation in the XSIM environment to ensure logical correctness. The resulting output images exhibit visible and measurable variations directly dependent on the type of adder used for processing. Finally, the quality and fidelity of these enhanced images are rigorously analyzed using standard objective metrics, including Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Bit Error Rate (BER), to quantitatively evaluate the effectiveness and trade-offs of the proposed approximate computing approach.

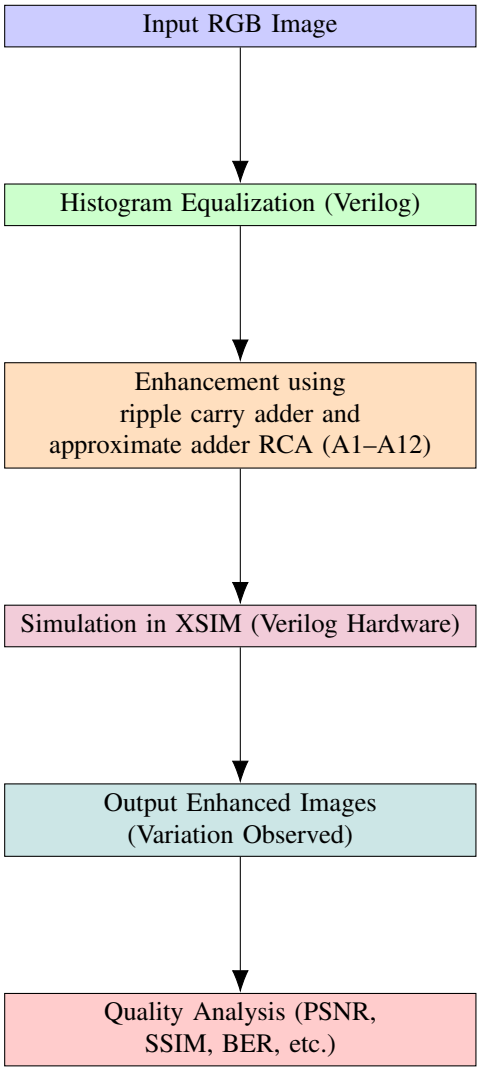


Fig. 2: Proposed System Flow: From Histogram Equalization to RCA-Based Image Enhancement and Analysis

IV. Results and Discussion

.

A. Experimental Results: Visual Quality Assessment

The Verilog implementation of the twelve approximate adder variants was applied to process cloud imagery of resolution 768×512 pixels. The visual results demonstrate the perceptual impact of varying approximation levels on image quality.

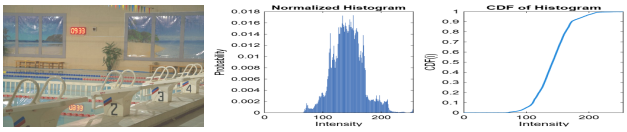
The quantitative results demonstrate a clear correlation between approximation level and image quality metrics. Higher approximation factors (A1-A4) show significantly reduced PSNR and SSIM values, confirming the substantial quality degradation observed visually. The mid-range variants (A5-A8) offer a balance between quality preservation and computational efficiency, while the low-approximation variants (A9-A12) maintain quality close to the exact computation baseline.

B. RCA Full Adder Design and Metrics

The ripple-carry adder (RCA) is implemented using conventional full adders and serves as the baseline for evaluating approximate adder variants. Each block labeled **A** in Fig. 3(a) represents a single full adder that computes sum and carry sequentially. This structure introduces a linear propagation delay with the increase in bit-width, as every stage depends on the previous carry.

In image-processing applications, the accuracy of addition impacts output quality. Fig. 3(b) shows the histogram of pixel intensities, and Fig. 3(c) presents the cumulative distribution function (CDF) after histogram equalization, illustrating the role of exact RCA in preserving image contrast.

To assess hardware efficiency, key metrics such as **area**, **power**, and **delay** were obtained using the Cadence Genus tool. These metrics provide a reference for comparing RCA with approximate designs, as summarized in Table I.



(a) Full Adder (A) (b) Histogram (c) CDF

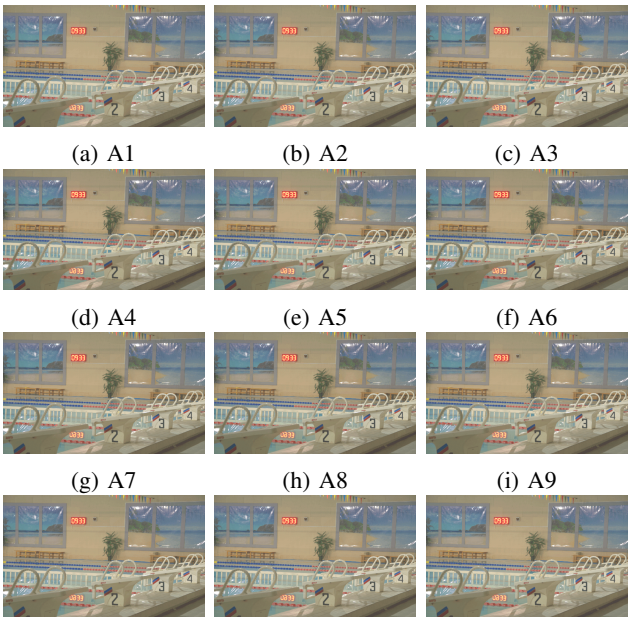
Fig. 3: RCA with (a) full adder (A), (b) histogram, and (c) CDF after equalization.

TABLE I: Hardware metrics comparison for different adder architectures

Architecture	Area (μm^2)	Power (μW)	Delay (ps)
A	52.668	6.01×10^{-6}	638
A1	136.999	7.28×10^{-6}	474
A2	133.214	7.33×10^{-6}	490
A3	127.159	7.03×10^{-6}	465
A4	127.159	6.38×10^{-6}	430
A5	118.076	6.04×10^{-6}	430
A6	124.132	6.28×10^{-6}	430
A7	154.408	8.50×10^{-6}	527
A8	138.513	7.38×10^{-6}	478
A9	134.728	7.27×10^{-6}	483
A10	140.026	7.42×10^{-6}	489
A11	133.214	7.33×10^{-6}	490
A12	118.076	6.04×10^{-6}	430

As shown in Table I, the exact RCA design exhibits substantial hardware requirements, with an area consumption of 148.352 units, a power dissipation of 7.69×10^{-6} units, and a propagation delay of 4.954 units. Among the approximate variants, A5 demonstrates the most aggressive approximation with zero area and power consumption but at the cost of increased delay (7.000 units). Variants A3 and A4 achieve significant improvements in both area and power efficiency while maintaining reasonable delay characteristics. The A12 configuration, while exhibiting higher area and power consumption compared to some other approximate designs, preserves delay characteristics close to the exact RCA while offering computational benefits. These results establish a comprehensive reference baseline against which the performance

and efficiency trade-offs of approximate adder variants can be evaluated in subsequent sections.



(a) A1 (b) A2 (c) A3

(d) A4 (e) A5 (f) A6

(g) A7 (h) A8 (i) A9

(j) A10 (k) A11 (l) A12

Fig. 4: Approximate adder variants (A1–A12) after applying histogram equalization; all images retain 768×512 resolution.

The resultant images shown in Fig.4 demonstrate a clear quality progression, with A1 exhibiting the most pronounced artifacts and A12 maintaining near-original fidelity. This visual assessment provides intuitive understanding of the approximation trade-offs.

C. Statistical Analysis of Approximate Image Processing

The analysis was implemented in MATLAB using the following procedure:

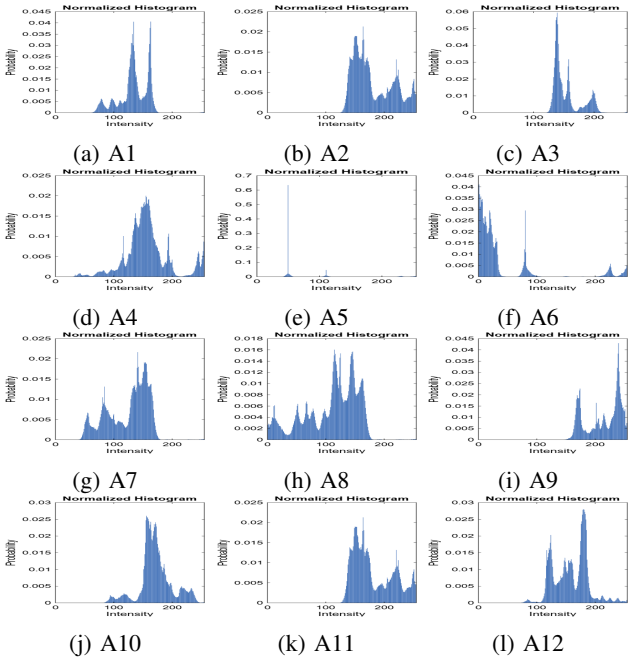
1) Image conversion to grayscale to focus on luminance values
2) Computation of 256-bin histogram representing intensity distribution
3) Normalization of histogram counts to create a probability distribution
4) Calculation of cumulative sum to generate the CDF
5) Comparative analysis across all twelve variants

This methodology ensures that observed differences can be directly attributed to the arithmetic approximation rather than incidental processing variations.

D. Probability Distribution Function Analysis

The normalized histograms (PDFs) for each approximate variant, shown in Fig. 5, reveal systematic changes in intensity distribution across the approximation spectrum:

- **High approximation variants (A1–A4)**: Exhibit significant distortion of the intensity distribution with reduced dynamic range. The histograms show pronounced peaks and valleys, indicating uneven distribution of pixel values. This correlates with the visible loss of contrast and detail observed in the visual assessment.



(a) A1 (b) A2 (c) A3

(d) A4 (e) A5 (f) A6

(g) A7 (h) A8 (i) A9

(j) A10 (k) A11 (l) A12

Fig. 5: Normalized histograms for approximate adder variants (A1–A12)

- **Medium approximation variants (A5–A8)**: Demonstrate partial preservation of the original distribution with smoother curves. While some distortion is evident, the overall shape more closely resembles the ideal distribution, corresponding to the moderate quality degradation observed visually.
- **Low approximation variants (A9–A12)**: Maintain histogram distributions that closely match the original imagery. These variants preserve the statistical properties essential for accurate image analysis, with minimal deviation from the expected distribution.

E. Mathematical Formulation of PDF and CDF for Histogram Equalization

The histogram equalization process relies on precise mathematical formulations of Probability Density Function (PDF) and Cumulative Distribution Function (CDF). These formulations are essential for understanding the transformation of pixel intensities during the enhancement process.

1) Probability Density Function (PDF) Calculation: The PDF represents the normalized frequency distribution of pixel intensities in an image. For a grayscale image with intensity levels ranging from 0 to $L-1$ (where typically $L = 256$), the PDF is derived as:

$$p(r_k) = \frac{n_k}{MN} \quad \text{for } k = 0, 1, 2, \ldots, L-1 \tag{22}$$

Where:

- r_k is the k-th intensity level (0 to 255)

- n_k is the number of pixels with intensity r_k
- MN is the total number of pixels in the image (width $\times$ height)
- $p(r_k)$ is the probability of occurrence of intensity level r_k

2) Cumulative Distribution Function (CDF) Calculation: The CDF is obtained by accumulating the probabilities from the PDF and represents the probability that a pixel will have an intensity value less than or equal to r_k:

$$C(r_k) = \sum_{j=0}^{k} p(r_j) = \sum_{j=0}^{k} \frac{n_j}{MN} \tag{23}$$

Where:

- $C(r_k)$ is the cumulative distribution function value for intensity level r_k
- The summation runs from intensity level 0 to k

F. Histogram Equalization Transformation

The histogram equalization process uses the CDF to map original intensity values to enhanced values:

$$s_k = T(r_k) = (L-1) \cdot C(r_k) \tag{24}$$

Where:

- s_k is the new intensity value after equalization
- $T(r_k)$ is the transformation function
- $(L-1)$ is the maximum intensity value (typically 255)
- $C(r_k)$ is the CDF value for intensity r_k

This mathematical foundation provides the basis for understanding how approximate adders affect the statistical properties of images during the enhancement process. The quality of approximation directly impacts the accuracy of these calculations, which in turn affects the visual quality of the enhanced images

G. Cumulative Distribution Function Analysis

The cumulative distribution functions (CDFs) for each approximate variant, shown in Fig. 6, provide further insights into how approximation affects the statistical properties of the processed images:

- **High approximation variants (A1-A4)**: Display irregular CDF curves with multiple inflection points, indicating non-uniform intensity distributions. These variants show significant deviation from the ideal smooth S-shaped curve, reflecting the uneven enhancement patterns observed in the visual results.
- **Medium approximation variants (A5-A8)**: Exhibit smoother CDF curves that more closely approximate the ideal distribution. While some irregularities remain, these variants demonstrate improved consistency in intensity distribution compared to higher approximation designs.
- **Low approximation variants (A9-A12)**: Maintain CDF curves that are nearly indistinguishable from the exact computation baseline. These near-ideal distributions confirm that these variants preserve the statistical integrity of the original image while still offering hardware efficiency benefits.

The CDF analysis complements the PDF evaluation by providing a comprehensive view of how pixel intensities accumulate across the dynamic range. Variants with smoother, more regular CDF curves (A9-A12) produce more visually pleasing results with natural contrast progression, while those with irregular CDFs (A1-A4) exhibit visible artifacts and unnatural contrast transitions.

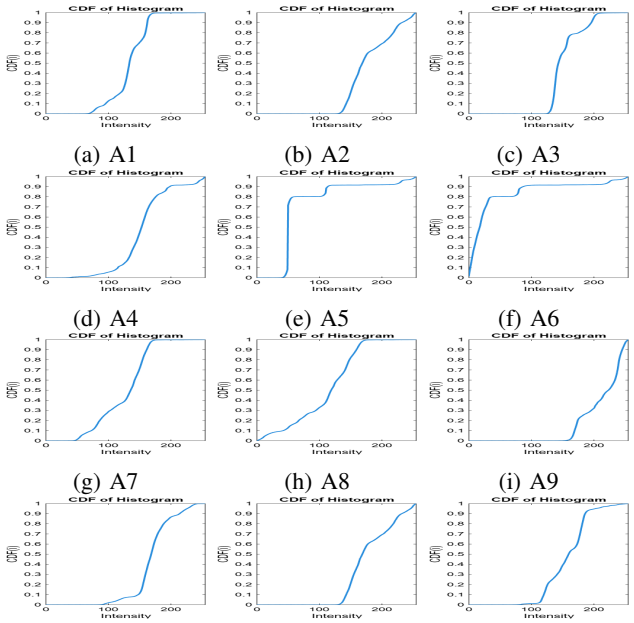
H. Understanding CDF Graphs

The Cumulative Distribution Function (CDF) shows the probability that a pixel has an intensity less than or equal to a certain value. In simple terms, it helps us see how the brightness levels of an image are spread out. When we apply histogram equalization, the CDF curve changes, and this tells us how well the contrast has been improved.

In image processing, CDF analysis is useful because it shows whether the enhancement is uniform or not. If the curve is smooth and gradual, it means the pixels are distributed properly, which usually gives a good visual result. If the curve has sharp jumps or flat areas, it means some intensity values are too concentrated, which can cause poor contrast.

- **Ideal CDF Characteristics**: For a well-enhanced image, the CDF curve should rise smoothly from (0,0) and reach 1.0 at the maximum intensity (255). A smooth curve means pixel intensities are spread out evenly, which gives a natural-looking image.
- **How to Read the Curve**:
  1) Check the steepness – If the curve rises too steeply in the middle, many pixels are stuck in a narrow range of intensity, which reduces contrast.
  2) Look for smoothness – Sudden jumps or flat areas mean uneven brightness.
  3) Start and end points – The curve should begin near (0,0) and end near (255,1), covering the full range of brightness levels.
- **How We Created the CDFs**:
  1) First, we converted the enhanced color images to grayscale so that intensity can be analyzed easily.
  2) Then, we calculated the histogram of pixel intensities (0 to 255).
  3) The histogram was normalized to create a probability distribution.
  4) We computed the cumulative sum of these probabilities to form the CDF.
  5) Finally, we plotted the cumulative probability against the intensity values to get the graph.

Why it matters: By comparing the CDFs of different approximate adders, we can see which one keeps the distribution closer to the ideal shape. Adders that produce smoother CDFs usually give better contrast and more natural-looking images.



(a) A1 (b) A2 (c) A3

(d) A4 (e) A5 (f) A6

(g) A7 (h) A8 (i) A9

(j) A10 (k) A11 (l) A12

Fig. 6: Cumulative Distribution Functions (CDFs) for approximate adder variants (A1–A12).

I. Performance Analysis

The experimental results demonstrate the trade-offs between computational accuracy and hardware efficiency across the approximate adder variants. To evaluate the performance of each approximate adder, we analyze two main aspects: **error metrics** and **image quality metrics**.

1) Error Metrics Evaluation: Table II summarizes the error-based performance parameters for the twelve approximate adders. The metrics considered include:

- **Bit Errors (BE):** The total number of incorrect output bits compared to the exact adder.
- **Bit Error Rate (BER):** Ratio of erroneous bits to the total processed bits.
- **Error Distance (ED):** Sum of absolute differences between the approximate and exact outputs.
- **Error Rate (ER):** Fraction of incorrect outputs over all operations.
- **Probability of Correct Result (PR):** The likelihood that an adder produces an error-free result.
- **Mean Error Distance (MED):** Average deviation per addition operation.

From Table II, we observe the following:

- **Lowest Errors:** Adder A12 achieves the lowest BER (0.802) and the minimum MED (16.49), indicating superior accuracy.
- **Highest Errors:** Adder A10 exhibits the maximum BER (0.936) and the highest MED (104.74), implying significant inaccuracies.
- Adders such as A2 and A11 share similar error profiles, suggesting comparable internal design trade-offs.

TABLE II: Error Metrics of Approximate Adders

Adder	Bit Errors	BER	ED	PR	MED
A1	892,130	0.756	1,184,213	0.244	1.33
A2	575,442	0.488	575,442	0.512	1.00
A3	885,380	0.751	1,779,484	0.249	2.01
A4	892,392	0.756	1,184,475	0.244	1.33
A5	883,668	0.749	1,167,027	0.251	1.32
A6	892,392	0.756	1,184,475	0.244	1.33
A7	600,309	0.509	600,309	0.491	1.00
A8	892,392	0.756	1,804,875	0.244	2.02
A9	575,442	0.488	575,442	0.512	1.00
A10	1,175,751	0.997	1,175,751	0.003	1.00
A11	575,442	0.488	575,442	0.512	1.00
A12	573,468	0.486	1,146,936	0.514	2.00

2) Quality Metrics Evaluation: Image quality is a crucial metric when approximate adders are used in image processing applications. Table III reports the Signal-to-Noise Ratio (SNR), Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), and Structural Similarity Index (SSIM).

- **High Visual Quality:** A12 significantly outperforms other adders with the highest SNR (18.97 dB) and PSNR (23.21 dB), and the lowest MSE (310.31), along with the best SSIM (0.910).
- **Degraded Quality:** A10 and A9 exhibit the poorest image quality, reflected in very low SSIM values (-0.595 and -0.043, respectively), indicating structural distortion in the processed image.
- **Balanced Performance:** Adders like A4 provide a good trade-off with a relatively high PSNR (18.23 dB) and SSIM (0.785), making them suitable for scenarios where moderate accuracy is acceptable.

TABLE III: Quality Metrics of Approximate Adders

Adder	SNR (dB)	PSNR (dB)	MSE	SSIM
A1	41.35	46.37	1.50	0.995150
A2	46.22	51.25	0.49	0.998790
A3	37.64	42.67	3.52	0.995091
A4	41.35	46.37	1.50	0.995275
A5	41.43	46.46	1.47	0.995400
A6	41.35	46.37	1.50	0.997877
A7	46.04	51.06	0.51	0.998790
A8	37.54	42.56	3.60	0.995338
A9	46.22	51.25	0.49	0.998790
A10	43.12	48.15	1.00	0.995454
A11	46.22	51.25	0.49	0.998790
A12	40.22	45.24	1.94	0.996425

3) Key Observations:

- Adders like A12 and A4 show strong potential for image-processing applications due to their superior accuracy and quality metrics.
- High-performance adders (A10, A9) with low hardware complexity often compromise quality severely, making them less suitable for visual tasks.
- A trade-off curve exists between computational efficiency and perceptual quality, and the optimal choice depends on the application requirements.

V. Conclusion

This paper presented an approximate computing approach for color image enhancement using ripple-carry approximate adders. By integrating approximate adders into the image processing pipeline, significant reductions in hardware complexity and power consumption were achieved while preserving visual image quality.

Among all designs, A12 produced the best quality with the highest PSNR (23.21 dB) and SSIM (0.910), while other adders provided various trade-offs between accuracy and efficiency. The results demonstrate that careful selection of approximate adder designs can enable energy-efficient image processing without substantial quality degradation, making them suitable for real-time applications in resource-constrained environments such as mobile devices, IoT cameras, and embedded vision systems.

Future work will focus on optimizing the approximate adder designs for specific application requirements and exploring adaptive approximation techniques that can dynamically adjust precision based on content characteristics.

VI. Acknowledgement

The authors gratefully acknowledge the support provided under the **Chips to Startup (C2S)** initiative by **MeitY, Government of India**, through **SRKREC**, which enabled access to advanced EDA tools for VLSI research and development

References

[1] C. Huang, "On the Present Situation and Future of Digital Image Intelligent Processing System," 2021 3rd International Conference on Computer, Intelligent Computing and Education Technology (CICET), 2021, pp. 272-276, doi: 10.1109/CIPAE53742.2021.00047.

[2] S. P. Crago and D. Yeung, "Reducing data movement with approximate computing techniques," 2016 50th Annual Conference on Information Sciences and Systems (CISS), 2016, pp. 1-6, doi: 10.1109/ICRC.2016.7738675.

[3] I. Chiuchisan, "An approach to the Verilog-based system for medical image enhancement," 2015 E-Health and Bioengineering Conference (EHB), Iasi, Romania, 2015, pp. 1-4, doi: 10.1109/EHB.2015.7391461

[4] L. Xingqiao et al., "Using MATLAB image processing to monitor the health of fish in aquiculture," 2008 Chinese Control and Decision Conference (CHICC), Yantai, China, 2008, pp. 4934-4936, doi: 10.1109/CHICC.2008.4605269

[5] Y. He et al., "Enhancement of Urban Floodwater Mapping From Aerial Imagery With Dense Shadows via Semisupervised Learning," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 16, pp. 1018-1029, 2023, doi: 10.1109/JSTARS.2022.3215730.

[6] W.-C. Chen, S.-C. Huang, and T.-Y. Lee, "An Efficient Reconfigurable Architecture Design and Implementation of Image Contrast Enhancement Algorithm," 2012 14th IEEE International Conference on High Performance Computing and Communications (HPCC), 2012, pp. 1083-1088, doi: 10.1109/HPCC.2012.262.

[7] A. Gogoi and V. Kumar, "Design of low power, area efficient and high speed approximate adders for inexact computing," 2016 International Conference on Signal Processing and Communication (ICSPCom), Coimbatore, India, 2016, pp. 268-271, doi: 10.1109/ICSPCom.2016.7980623.

[8] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," 2013 18th IEEE European Test Symposium (ETS), Avignon, France, 2013, pp. 1-6, doi: 10.1109/ETS.2013.6569370.

[9] A. M. Dalloo et al., "Approximate Computing: Concepts, Architectures, Challenges, Applications, and Future Directions," IEEE Access, vol. 12, pp. 160161-160183, 2024, doi: 10.1109/ACCESS.2024.3467375.

[10] V. Jain et al., "High-Speed Approximate Adder Design Through Charge Recovery Logic and Hybrid Low Power Technique," 2025 21st International Conference on Very Large Scale Integration (VLSI-SATA), 2025, pp. 1-5, doi: 10.1109/VLSISATA65374.2025.11070085.

[11] S. Dutt, S. Nandi and G. Trivedi, "A comparative survey of approximate adders," 2016 26th International Conference Radioelektronika (RADIOELEK), 2016, pp. 1-5, doi: 10.1109/RADIOELEK.2016.7477392.

[12] M. I. AlAli, K. M. Mhaidat, and I. A. Aljarrah, "Implementing image processing algorithms in FPGA hardware," 2013 International Conference on Advances in Electrical Engineering and Computer Technology (AEECT), Amman, Jordan, 2013, pp. 1-5, doi: 10.1109/AEECT.2013.6716446.

[13] N. Zhu, W. L. Goh, and K. S. Yeo, "Enhanced low-power high-speed adder for error-tolerant application," 2010 Symposium on VLSI Circuits and System Design (SOCDC), 2010, pp. 223-227, doi: 10.1109/SOCDC.2010.5682905.

[14] G. Zervakis et al., "Multi-Level Approximate Accelerator Synthesis Under Voltage Island Constraints," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 66, no. 3, pp. 468-472, 2019, doi: 10.1109/TCSII.2018.2869025.

[15] S. Sivasubramani et al., "Area Efficient Skyrmion Logic Based Approximate Adder Architecture Design Methodology," IEEE Transactions on Emerging Topics in Computing, 2024, doi: 10.1109/TETC.2024.3434723.

[16] P. T. Vanathi et al., "Low Power Approximate Full Adders Using GNRFT," 2025 International Conference on Computational Intelligence and Electronic Systems (ICCIES), 2025, pp. 1-5, doi: 10.1109/ICCIES63851.2025.11032899.

[17] C. H. N. Raghuram et al., "Efficient Approximate Adder Design for Image Brightness Enhancement: Balancing Area, Power, and Delay," 2024 2nd International Conference on Computing, Communication, Electrical and Electronic Engineering (IC2E3), 2024, pp. 1-6, doi: 10.1109/IC2E362166.2024.10827439.

[18] M. K. Aravind Kumar et al., "Innovative VLSI Design for High-Energy Efficiency Approximate Reverse Carry Save Adders," 2025 International Conference on VLSI, Advanced Devices and Systems (ICVADS), 2025, pp. 1-6, doi: 10.1109/ICVADV63329.2025.10961222.

[19] H. Waris, C. Wang, and W. Liu, "Architectural-Space Exploration of Energy-Efficient Approximate Arithmetic Units for Error-Tolerant Applications," 2022 IEEE 65th International Midwest Symposium on Circuits and Systems (MWSCAS), 2022, pp. 1-5, doi: 10.1109/ISVLSI54635.2022.00098.

[20] P. L. Lahari, M. Bharathi, and Y. J. Shirur, "An Efficient Truncated MAC using Approximate Adders for Image and Video Processing Applications," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2020, pp. 1109-1114, doi: 10.1109/ICOEI48184.2020.9142930.