IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 8, Issue 2, Ver. I (Mar.-Apr. 2018), PP 58-64
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Retime Low Power Approximate Multiplier for Image Sharpening and Smoothing

Jalaja S[1], Tejaswini A[2]
VTU Research Scholar, Assistant Professor Department of ECE, BIT, Bangalore, India
M. Tech, Department of ECE, BIT, Bangalore, India
Corresponding Author: Jalaja S

Abstract: *In DSP systems, due to demand in the higher computational performance in the architecture's is increasing the complexity. In order to enhance the performance, the approximate arithmetic circuits are designed with small errors to increase the speed. The main aim of this work is to approximate the multiplication process with low power consumption. In this paper, a retime low power approximate multiplier is proposed. The approach is based on rounding the operands to the nearest exponent of two. Using this approach, the computational intensive part of multiplication is omitted and thus improving the speed of the multiplier at the cost of small penalty errors. This proposed approach is appropriate for both signed and unsigned operations. The efficiency of the proposed Retime approximate multiplier is analyzed by FIR filter using cross correlation method in image sharpening and smoothing.*

Key Words: *Approximate Multiplier, Retime Rounding number, Finite Impulse Response (FIR)*

Date of Submission: 25-04-2018 Date of acceptance: 14-05-2018

I. Introduction

In any electronic systems, Energy minimization is one of the most important design requirements, especially in the portable ones such as smart phones, tablets and different gadgets. It is greatly desired to reach this minimization with minimal performance (speed) penalty [1]. Digital Signal processing (DSP) blocks are core components of these portable devices for performing various multimedia applications. The computational core of these DSP blocks is the arithmetic logic unit (ALU) where multiplications and additions have the more share among allother arithmetic operations [2]. The multiplications play leading operation in the processing elements which can leads to high consumption of energy and power. Thus, improving the speed and power/energy-efficiency characteristics of multipliers plays an important role in improving the efficiency of processors. Soo many DSP cores implement image and video processing algorithms where end results are either images or videos prepared for human use. This feature enables us to use approximations for improving the speed/energy efficiency. This arises from the limited perceptual abilities of human beings in observing an image or a video. Apart from these image and video processing applications, there are other areas where the exactness of the arithmetic operations is not important to the functionality of the system (see [3], [4]). Due to the use of approximate computing provides the designer with tradeoffs between the accuracy and the speed and also the power/energy consumption [2], [5].

At different design abstraction levels like circuit, logic and architecture levels as well as algorithm and software layers this approximation can be applied [2]. The approximation may be performed using different methods such as enabling some timing violations (e.g., voltage overscaling or overclocking) and function approximation methods (e.g., modifying the Boolean function of a circuit) or a combining both the methods [4], [5]. In the function approximation methods, several approximating arithmetic building blocks, like adders and multipliers, at different design levels have been recommended (see [6]– [8]).

In VLSI Signal Processing there are two types of digital filters i.e. FIR (finite impulse response) and IIR (infinite impulse response). FIR is that the impulses are finite in this filter and phase is kept linear in order to noise distortions and no feedback is used for such filters. FIR is very easy to design when compared with IIR. These FIR filters are used in DSP processors for high speed. The FIR filter is used to check the efficiency of image processing applications.

In this paper, we focus on proposing a Retime low power approximate multiplier appropriate for errortolerance DSP applications. The main contributions of this paper can be outlined as follows:

1) Presenting a new scheme for approximate multiplication by modifying the conventional multiplication approach and retiming approach is used.
2) Efficiency of proposed retime approximate multiplier is evaluated using FIR filter in image sharpening and smoothing.

II. Related Works

Here, the previous works of approximate multiplier are briefly reviewed. In [3], an approximateMultiplier and anadder based on a methodcalled broken-array multiplier (BAM) were proposed. By applying this method of [3] tothe conventional modified Booth multiplier, an approximatesigned Booth multiplier was proposed in [5]. The approximatemultiplier provided power consumption savings from 28% to58.6% and also area reductions from19.7% to 41.8% for differentword lengths in comparison with a regular Booth multiplier.

Kulkarni et al. [6] proposed an approximate multiplier consists of several 2 × 2 inaccurate building blocks thatsaved the power by 31.8%–45.4% over an accurate multiplier. An approximate signed 32-bit multiplier for prediction purposes in pipelined processors was designed in [7]. It was 20% faster than a full-adder-based tree multiplierhavinga probability of error of around 14%. In [8], an error-tolerantmultiplier, which computed the approximate result by dividingthe multiplication into one accurate and one approximate part, was proposed, but the accuracies for different bit widthswere reported. In a 12-bit multiplier, a power savingof more than 50% was reported. In [9], two approximate 4:2compressors weredesigned and analyzed to use in a regular Dadda multiplier.

Using approximate multipliers in image processing applications, the power consumption, delay, and transistor count compared with those of an exactmultiplier design can be reduced. In [10], an accuracy-configurable multiplier architecture (ACMA) was proposed for error-tolerance systems. ACMA used a method named carry-in prediction that worked on a precomputation logic to increase its throughput. The proposed approximatemultiplication resulted in reduction of 50% in the latencyby reducing the critical path when compared with exact one. Bhardwaj et al. [11] suggested an approximate Wallace tree multiplier (AWTM). It also used the carry-in prediction technique to reduce the criticalpath. In this work, using AWTM in a real-time benchmark image application showing about reductions of 40% and 30% in the power and area, respectively, without any loss image quality.

In [12], based on an approximate logarithm of the operands an approximate unsigned multiplication and division havebeen proposed. In the proposed multiplication, the result is determined by the summation of the approximate logarithms. Thus, the multiplication is simplified to some shift and adds operations. In [13], a technique for increasing theaccuracy of the multiplication approach of [12] was suggested. This technique was based on thedecomposition of the input operands. Thismethod improved the average error at the penaltyof doubling the hardware of the approximate multiplier.

In [16], a dynamic segment method (DSM) is proposed, which performs the multiplication operation on an m-bitsegment starting from the leading one bit of the input operands. In [17], a dynamic range unbiased multiplier (DRUM) multiplier, which selects an m-bit segment starting from the leading onebit of the input operands and sets the LSB ofthe truncated values to one, has been proposed. In this method, the truncated values are multiplied and shifted toleft to generate the final result. In [18], an approximate 4×4WTM has been suggested that uses an inaccurate 4:2 counters. Also, an error correction unit for correcting the outputshas been proposed. This 4×4inaccurateWallace multiplier can be used to construct larger multipliers in an array structure.

In this paper, like [12], we propose performing the approximate multiplicationthrough simplifying the operation. The difference between ourwork and [12] is that, although the principles are same for unsigned numbers, butthe mean error of our proposed approach is less.

III. Approximate Multiplier

1. Proposed retime approximate multiplier description

The main aim of proposing this approximate multiplier is to make use of the ease of operation when the numbers are two to the power n (2^n). The detailed operation of the approximate multiplier is denoted as inputs of A and B and the rounded values of inputs A and B are Ar and Br, respectively. The multiplication of A by B may be written as

$$A \times B = (Ar - A) \times (Br - B) + Ar \times B + Br \times A - Ar \times Br \quad (1)$$

The important observation is that the multiplications of Ar×Br, Ar×B, and Br×A may be implemented just by the shift operation. But, the hardware implementation of (Ar− A) × (Br − B), is very complex. The weight of this term in the end result, which depends on differences of the exact numbers from their rounded

ones, is very small. Hence, (Ar− A) × (Br − B) is omitted from (1), Soo that it helps in the simplification of multiplication operation. Therefore, to perform the multiplication operation, the following expression is used:

A x B = Ar x B + Br x A – Ar x Br ---------- (2)

Hence, the multiplication operation can be performed by using three shift and two addition/subtraction operations. In this method, the nearest values for A and B in the form of 2^n should be determined. When the value of A (or B) is equal to the $3 \times 2^{p-2}$ (where p is an arbitrary positive integer larger than one), it has two nearest values in the form of 2n with equal absolute differences that are $2^p$and 2^{p-1}. The accuracy of the suggested multiplier leads to same effect for both the values, selecting the larger one (except for the case of p = 2) leads to a less hardware implementation for determining the nearest rounded value, therefore, it is considered in this paper. It originates from the fact that the numbers in the form of $3 \times 2^{p-2}$ are considered as they do not care in both rounding up and down simplifying the process, and smaller logic expressions may be attained if they are used in the rounding up. But in case of three the nearest value is considered as two in the, proposed approximate multiplier.

Therefore, the advantage of the proposed Retime approximate multiplier exists only for positive inputs because in the two's complement representation, the rounded values of negative inputs are not in the form of 2^n. Hence, before the multiplication operation starts, the absolute values of both inputs and the output sign of the multiplication result based on the inputs signs be determined and then the operation be performed for unsigned numbers and, at the last stage, the proper sign be applied to the unsigned result.

2. Hardware Implementation of Retime approximate multiplier

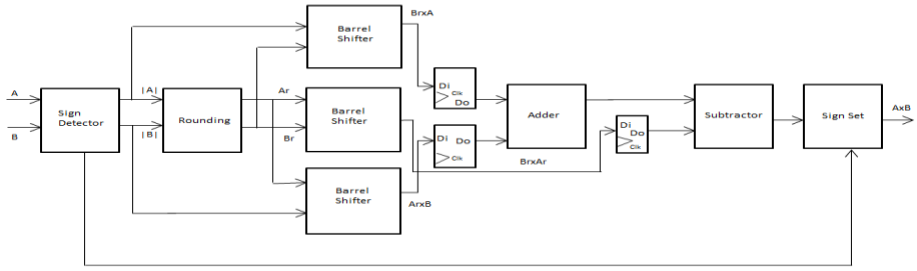

Fig.-1: Block Diagram for Retime approximate Multiplier

The proposed approximate multiplier is applicable to both unsigned and signed operation. From the equation (2), the hardware implementation of the proposed multiplier block diagram is as shown in fig. 1. Here the inputs are represented in 2's complement format. The first thing is the sign of the inputs are determined, if any of the input is negative then the absolute value of that input generated. Then, for each absolute value the nearest value in the form of 2n is extracted from the rounding block. The bit width of the output of rounding block is n (the MSB of the absolute value of an n-bit number in the two's complement format is zero). To determine the nearest value of input A, the operands are rounding off to the power of 2 with the help of rounding criteria.

$$A_r[n-1] = \overline{A[n-1]} \cdot A[n-2] \cdot A[n-3] + A[n-1] \cdot \overline{A[n-2]}$$

$$A_r[n-2] = (\overline{A[n-2]} \cdot A[n-3] \cdot A[n-4] + A[n-2] \cdot \overline{A[n-3]}) \cdot \overline{A[n-1]}$$

$$\vdots$$

$$A_r[i] = (\overline{A[i]} \cdot A[i-1] \cdot A[i-2] + A[i] \cdot \overline{A[i-1]}) \cdot \prod_{i=i+1}^{n-1} \overline{A[i]}$$

$$\vdots$$

$$A_r[3] = (\overline{A[3]} \cdot A[2] \cdot A[1] + A[3] \cdot \overline{A[2]}) \cdot \prod_{i=4}^{n-1} \overline{A[i]}$$

$$A_r[2] = A[2] \cdot \overline{A[1]} \cdot \prod_{i=3}^{n-1} \overline{A[i]}$$

$$A_r[1] = A[1] \cdot \prod_{i=2}^{n-1} \overline{A[i]}$$

$$A_r[0] = A[0] \cdot \prod_{i=1}^{n-1} \overline{A[i]}.$$

In the above equation, $Ar[i]$ is one in two cases. In the first case, $A[i]$ is one and all the bits on its left side are zero while $A[i-1]$ is zero. In the second case, when $A[i]$ and all its left-side bits are zero, $A[i-1]$ and $A[i-2]$ are both one.After finding the rounding values, the products Ar× Br, Ar× B, and Br × A are calculated

by using thethree barrel shifter blocks. Therefore, the amount of shifting is determined based on logAr2 − 1 (or logBr2 − 1) in the case of A (or B) operand. Here, the input bit width of the shifter blocks is n, while their outputs are 2n. By using single Kogge-Stone adder the summation of Ar× B and Br × A is calculated. The output of this adder and one of the barrel shifterresults of Ar× Br are the inputs of the subtractorblock whose output is the absolute value of the output of the proposed multiplier. Finally, using the sign set block if the final output is negative value then the output is negated.

In this paper, proposed retime low power approximate multiplier architecture designed using repositioning of the Flip-flops. Flip-flops are placed at the cutset line of the barrel shifters and are retimed towards the high switching activity factor of different blocks to reduce the power consumption of the design.

3. Image Sharpening and Smoothing

To evaluate the efficiency of the retime low power approximate multiplier in real time applications, the performances of the proposed multiplier is compared in image processing applications i.e. sharpening and smoothing. For both image sharpening and smoothing FIR filter is used to find the efficiency.

3.1. FIR filter implementation

In the FIR system, the impulse response is of finite duration, this means that it has a finite number of nonzero terms. The response of the FIR filter depends only on the present and past input samples. The implementation of an FIR requires three basic building blocks: Multiplication, Addition and Signal delay. FIR filter can be expressed as

$$y[n] = \sum_{k=0}^{N-1} b_k \, x(n - k)$$

Where N represents the filter order, y [n] is the output signal and b_k represents the set of filter coefficients. If x[n] is the input signal applied, x [n - k] terms are referred as taps or tapped delay lines. A cross correlation method is used to find the coefficients of FIR filter.

To implement FIR Filter cross correlation method is used. Consider two waveforms, both sampled at the same rate. The sum of the products of the corresponding pairs of points is represented as a measure of the correlation of the two waveforms. By using cross correlation method, the coefficients of the matrix are found.

3.2. Image Sharpening

Image sharpening is an enhancement technique it highlights edges and fine details in an image. Sharpening an image increases the contrast between bright and dark regions to bring out features. The high pass filter to an image is applied in a sharpening process.

For sharpening, each pixel of the sharp image was extracted from [15], given by

$$Y(i, j) = 2 \cdot X(i + m, j + n) - \frac{1}{273} \sum_{m=-2}^{2} \sum_{n=-2}^{2} X(i+m, j+n) \cdot \mathrm{Mask}_{\mathrm{Sharpening}} \ (m + 3, n + 3)$$

Where the X (i, j) [Y (i, j)] indicates the pixel of the ith row and j th column of input (output) image and Mask $_{\text{sharpening}}$ is an n × n coefficient sharpening matrix given by

$$\mathrm{Mask}_{\mathrm{sharpening}} = \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

3.3. Image Smoothing

Smoothing is used to reduce noise within an image or to produce a less pixelated image. Most smoothing methods are based on low pass filters. Smoothing is also usually based on a single value representing the image, such as the average value of the image or the middle (median) value.

The smoothed output image is given by the following equation [15] is

$$Y(i,j) = \frac{1}{60}\sum_{m=-2}^{2}\sum_{n=-2}^{2} X(i+m, j+n) \cdot \text{Mask}(m+3, n+3).$$

Here, again X (i, j) [Y (i, j)] is the pixel of the i[th] row and j[th] column of input (output) image and Mask Smoothing is an n × n coefficient smoothing matrix given by

$$\text{Mask}_{\text{smoothing}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 4 & 4 & 4 & 4 \\ 1 & 4 & 12 & 4 & 7 \\ 1 & 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

IV. Results and Discussion

Figure 2 shows the simulation results of retime approximate multiplier. The retime approximate multiplier is implemented using Verilog and the simulations are performed by using Xilinx ISE simulator 14.5 and synthesized using cadence RTL compiler.

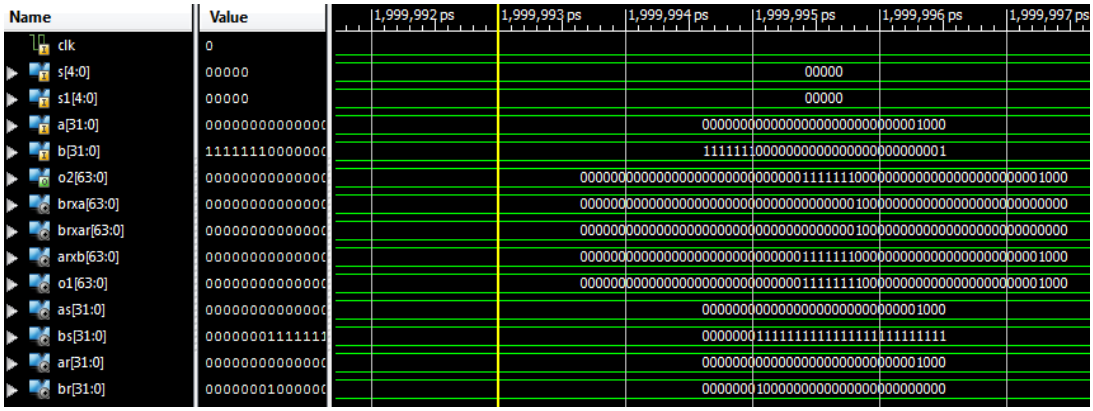

Fig.-2: Simulation results of Retime approximate Multiplier

TABLE-I Synthesis Results

Parameters	**Approximate multiplier using pipeline barrel shifter**	**Retime approximate multiplier using pipeline barrel shifter**
Cells	7321	7181
Leakage Power(nw)	426.636	367.090
Dynamic Power(nw)	1914551.752	214739.779
Total Power(nw)	1914978.388	215106.870

By comparing the results of approximate multipliers with respect to cells and power the following observations are made. From Table I show the cells occupied by retime approximate multiplier are less, because of multiplication operation has been simplified by rounding the values to the nearest power of two. Even though there is a small error in the output, shows reduction in area. After applying the retiming technique to approximate multiplier, the power reduction is less compared to existing design [22].

(a) (b)

Fig.-3: Image sharpening using proposed retime approximate multiplier. (a) Original image (b) Sharpened image utilizing retime approximate multiplier

Consider an example airplane for the sharpening described above for the original image shown in fig.-3(a). By using proposed retime approximate multiplier the sharpened image is shown in fig.-3(b). But this sharpening process cannot be easily recognized by human eyes.

TABLE II PSNR and MSSIM values for Sharpening

Image	Retime approximate multiplier		RoBA multiplier [22]	
	PSNR(dB)	MSSIM	PSNR(dB)	MSSIM
Airplane	54	1.00	39.3	1.00
Girl	44	0.98	49.5	1.00
House	41	1.00	40.6	1.00
Mandrill	54	1.00	44.3	1.00
Peppers	56	1.00	42.8	1.00
Vd-Orig	55	1.00	45.1	1.00
Lena	52	0.99	43.4	1.00

The peak signal-to-noise ratio (PSNR) and mean structural similarity index metric (MSSIM [20]) of the sharpened pictures are shown in Table II. It should be noted that the reported PSNRs are determined based on the sharpened image obtained using the retime approximate multiplier structures. Also, the MSSIM values closer to one indicate higher qualities for the approximate output image.

Table III PSNR and MSSIM values for Smoothing

Image	Retime approximate multiplier		RoBA multiplier [22]	
	PSNR(dB)	MSSIM	PSNR(dB)	MSSIM
Airplane	40	1.00	40.60	0.95
Girl	39	0.99	48.96	1.00
House	41	1.00	41.45	0.99
Mandrill	40	1.00	43.72	1.00
Peppers	43	1.00	43.27	1.00
Vd-Orig	45	1.00	45.53	1.00
Lena	46	0.98	44.47	1.00

The PSNR and MSSIM of the smoothing process is also analyzed using retime approximate multiplier for the images are reported in the Table III. As the results reveal, all the PSNRs (MSSIMs) are higher than 40 (0.99) showing small errors for the proposed approximate multiplier.

V. Conclusion

In this paper, retime low power approximate multiplier is proposed to achieve low power consumption compared to existing design. The approximate multiplier is based on rounding the inputs to 2^n input form. By rounding the inputs, the computationally intensive part of multiplication is ignored. The retime approximate multiplier reduces the total power consumption by repositioning the flip-flops. This method is applicable to both signed and unsigned multiplication. The results show that the proposed retime multiplier shows better performance in terms of area and power. The efficiency of the proposed retime approximate multiplier is further analyzed for image sharpening and smoothing. The results of image sharpening and smoothing results reveal that all the PSNRs and MSSIMs are higher than 40 and 0.99 illustrating small errors for the proposed retime approximate multiplier.

References

[1]. M. Alioto, "Ultra-low power VLSI circuit design demystified and explained: A tutorial," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59,no. 1, pp. 3–29, Jan. 2012.

[2]. V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[3]. H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[4]. R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in Proc.Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.

[5]. F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in Proc. 17th Int. Symp. Comput. Archit. Digit. Syst. (CADS), Oct. 2013, pp. 25–30.

[6]. P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in Proc. 24th Int. Conf. VLSI Design, Jan. 2011, pp. 346–351.

[7]. D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.

[8]. K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application," in Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits (EDSSC), Dec. 2010, pp. 1–4.

[9]. A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," IEEE Trans. Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.

[10]. K. Bhardwaj and P. S. Mane, "ACMA: Accuracy-configurable multiplier architecture for error-resilient system-on-chip," in Proc. 8th Int. Workshop Reconfigurable Commun.-Centric Syst.-Chip, 2013, pp. 1–6.
[11]. K. Bhardwaj, P. S. Mane, and J. Henkel, "Power- and area-efficient approximate wallace tree multiplier for error-resilient systems," in Proc. 15th Int. Symp. Quality Electron. Design (ISQED), 2014, pp. 263–269.
[12]. J. N. Mitchell, "Computer multiplication and division using binary logarithms," IRE Trans. Electron. Comput., vol. EC-11, no. 4, pp. 512–517, Aug. 1962.
[13]. V. Mahalingam and N. Ranganathan, "Improving accuracy in Mitchell's logarithmic multiplication using operand decomposition," IEEE Trans. Comput., vol. 55, no. 12, pp. 1523–1535, Dec. 2006.
[14]. Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available: http://www.nangate.com/
[15]. H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.
[16]. S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.
[17]. S. Hashemi, R. I. Bahar, and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD), Austin, TX, USA, 2015, pp. 418–425.
[18]. C.-H. Lin and I.-C. Lin, "High accuracy approximate multiplier with error correction," in Proc. 31st Int. Conf. Comput. Design (ICCD), 2013, pp. 33–38.
[19]. A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in Proc. 49th Design Autom. Conf. (DAC), Jun. 2012, pp. 820–825.
[20]. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004.
[21]. J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," IEEE Trans. Comput., vol. 62, no. 9, pp. 1760–1771, Sep. 2013.
[22]. Reza Zendegani, Mehdi Kamal, Milad Bahadori, Ali Afzali- Kusha and MassoudPedram," RoBAMultiplier: A Rounding-Based Approximate Multiplier for High-Speed yet Energy Energy- Efficient Digital Signal Processing" IEEEtransactions on very large scale integration (VLSI)systems syst., pp.1-9, July 2017.

Jalaja S "Retime Low Power Approximate Multiplier For Image Sharpening And Smoothing "IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) , vol. 8, no. 2, 2018, pp. 58-64