IJAREEIE

e-ISSN: 2278-8875
p-ISSN: 2320-3765

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

Volume 13, Issue 6, June 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.317

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

Modified TOSAM: Truncation and Rounding-Based Scalable Approximate Multiplier for High-Speed for Energy-Efficient Digital Signal Processing

Advaiet B Mishra[1], Dr Prabhat Sharma[2]

PG Scholar, Dept of Electronics and Communication Engineering, OIST, Bhopal, MP, India[1]

HOD, Dept of Electronics and Communication Engineering, OIST, Bhopal, MP, India[2]

ABSTRACT: The TOSAM method reduces the number of partial products by truncating each input process based on the leading unit position, resulting in an elastic approximate multiplier. This approach involves shifting, adding, and limiting multiplier operations of a given length, leading to significant improvements in energy efficiency and resource usage compared to traditional multipliers. The input process is rounded to the nearest value to enhance overall precision, and the multiplier can be scaled according to the truncated positions of the input operands. With lower precision in input operand width, the multiplier remains scalable, and further improvements are possible if design parameters such as area and power consumption decrease. The design parameters of the proposed approximate multiplier are compared with those of an accurate multiplier and other recently suggested approximate multipliers to evaluate performance. The results indicate that the proposed multiplier, with an absolute error ranging from 11% to 0.3%, significantly reduces time delays, area, and energy usage by 90%, 98%, and 41%, respectively. Various approximate multipliers often exhibit similar improvements in area and energy usage. The proposed multiplier features a Gaussian error distribution with a mean value close to 0, making it suitable for tasks such as writing, sharpening, and organizing JPEG encoders. Tests demonstrate a minor improvement in output quality. Additionally, a configurable TOSAM accuracy feature allows for adjusting energy usage based on the required precision during propagation.

KEYWORDS: Configurable Accuracy, Approximate Multiplier, Area-Efficient, Low-Energy, Scalable, Truncation-Based.

I.INTRODUCTION

Power utilization is a key architectural requirement in digital network design. Calculation approximation (CA) is one strategy to enhance energy efficiency. CA can be particularly useful in error-resilient applications, where exact outcomes are not always necessary. These applications include audio and image processing, machine learning, and data mining. In many signal processing tasks, arithmetic operations account for a significant portion of energy consumption, such as up to 75% in the fast Fourier transform system.Approximate multipliers are therefore well-suited for error-tolerant signal processing systems. Typically, a multiplication cycle consists of three stages: generating partial products from input operations, reducing these partial products to two rows, and then using an adder to combine these rows. Strategies to improve this process involve generating partial products more efficiently or reducing the complexity of their addition, which can reduce delays and power consumption.

One method to achieve this is through the use of approximate multipliers that employ truncation and rounding. In the proposed approach, inputs are truncated to a certain number of bits based on their leading bits. The error introduced by truncation is then mitigated by rounding, resulting in a more accurate approximation. This simplification leads to a smaller and more energy-efficient multiplier.

The accuracy of this method is primarily determined by the truncation and rounding parameters, which do not significantly affect the input operand distance, thus ensuring scalability. The proposed multiplier has several key features:

- Identifies leading multiplier positions and applies truncation and rounding to enhance accuracy.
- Explores truncation (t) and rounding (h) parameters to balance accuracy, delay, and energy usage.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

- Implements a hardware version of the approximate multiplier (TOSAM) for both signed and unsigned operations.
- Evaluates the multiplier's specifications for image processing and other grading demands.

The paper is organized as follows: Section II reviews previous research on approximate multipliers. Section III describes the architecture of the new approximate multiplier, and Section IV details its hardware implementation and error analysis.

II.REVIEW OF LITERATURE SURVEY

This section reviews various research efforts on designing approximate multipliers. In the dynamic segment (DSM) method, input operands are truncated to m bits based on the leading element, resulting in a fixed-width multiplication. This truncation method often produces outputs smaller than the actual values, leading to a negative mean relative error, which is undesirable for applications requiring high signal-to-noise ratios, like optical signal processing with Gaussian error distribution.

In the complex DRUM structure, the least significant bit of the truncated input is set to "1" to bring the mean relative error to zero. The LETAM structure truncates input operands and ignores half of the partial products, reducing delays and power consumption compared to DSM and DRUM systems. RoBA multipliers round input operands to the nearest power of two, simplifying the multiplication, addition, and subtraction processes, thus enhancing energy efficiency and speed.

Other methods have removed the least significant bits of partial products to increase multiplier speed and size. Partial products can be generated through logical AND operations or encoded in higher radixes. However, as radix decreases, the encoding complexity increases. Approximate encoders can generate partial products with this complexity. Approximate radix-4 booth multipliers, radix-9 multipliers, and partial product accumulation techniques have been explored.

In some approaches, the most significant bits of the multiplier are encoded with radix-4 encoding, while the least significant bits use higher approximate radix encoding. Various estimated compressors, such as 4:2 and 5:3 compressors, have been proposed to improve multiplier accuracy. A design algorithm for efficient approximate multipliers using these compressors has also been suggested.

Other methods involve modifying the counting scheme to a logarithmic one, which increases multiplication speed. This method computes the logarithm of input operands, sums them, and performs an antilogarithm operation to obtain the final result. The accuracy of these multipliers depends on the precision of the logarithm and antilogarithm steps. Mitchell proposed a simple procedure for logarithmic and antilogarithmic computations, which has been the basis for subsequent logarithmic multiplier designs.

This paper introduces a novel multiplier design that truncates and rounds the leading bits of input operands, applies certain adjustments, and performs small fixed-width multiplication operations to achieve the desired output.

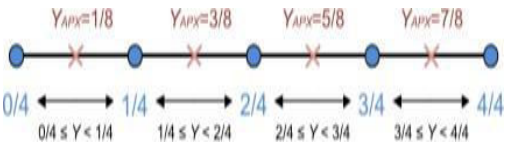


Fig. 1. Amount of YAPX according to the amount of Y for the case of S =4.

III. PROPOSED APPROXIMATE MULTIPLIER

A. TOSAM Each positive integer number (N) can be represented as

$$N = \sum_{i=0}^{k} 2^i x_i \qquad (1)$$

where k denotes the position of its leading one bit and xi is the ith bit of N. By factoring 2k from (1), it is

$$N = 2^k \left(\sum_{i=0}^{k} 2^{i-k} x_i \right) = 2^k \times X \qquad (2)$$

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

rewritten as

where X is a fractional number between 1.0 and 2.0. Based on (2), the result of multiplying A by B may be calculated as

$$A \times B = 2^{k_A + k_B} \times X_A \times X_B. \quad (3)$$

X_A and X_B widths are the same as A and B for the correct period and power usage amount of X_A perX_B. We propose that the average sum of this word be determined based on X_A and X_B fractional sections. They reflect the fraction of X in the remainder of this article, as Y is from

$$Y = X - 1. \quad (4)$$

For example, assume that X =(1.1101)2. In this case, Y =(0.1101)2. To generate the approximate value of Y, we divide this range (0.0–1.0) into S equal segments where S is a power of two represented by

$$S = 2^h \quad (5)$$

where h denotes an arbitrary positive integer which is one of our design parameters. It is obvious that the length of each segment is equal to 1/S. We propose to generate the approximate value of Y as

$$Y_{\mathrm{APX}} = \frac{2m-1}{2S} \quad \text{if } \frac{m-1}{S} \le Y < \frac{m}{S}, \quad m = 1, 2, \ldots, S. \quad (6)$$

The estimated sums of Y for the situation in which S equals 4 are seen in Figure for a greater example. 1. In order to locate Y_{APX}, only h most significant pieces of Y must be remembered. For eg, if S = 4(h = 2) is zero if there are two big bits of Y, it implies 0 is < 1/4. So, as $Y_{\mathbf{APX}}$, we chose 1/8 = (0.001)2. If two of the most

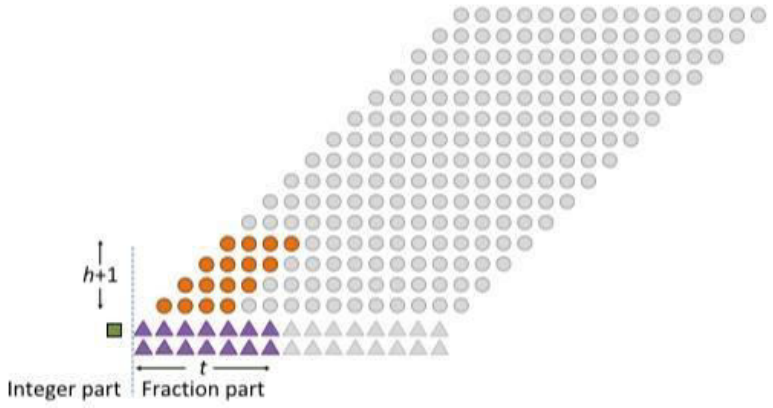


Fig. 2. Dot diagram of term $1+(Y_A)_t+(Y_B)_t+(Y_A)_{\mathrm{APX}} \times (Y_B)_{\mathrm{APX}}$ where $t = 7$ and $h = 3$.

significant Y pieces are "10," indicating 2/4 of a new Y < 3/4, Y_{APX} is then approximated to 5/8=(0.101)2. In other words, Y_{APX} 's meaning is obtained by actually splitting Y into h bits and adding a "1" bit on the right side. The range of YAPX thus corresponds to h+1 bits. (4), (3) shall be revised as

Now, the approximate of (7) may be expressed as

$$A \times B \approx 2^{k_A + k_B} \times (1 + Y_A + Y_B + (Y_A)_{\mathrm{APX}} \times (Y_B)_{\mathrm{APX}}). \quad (8)$$

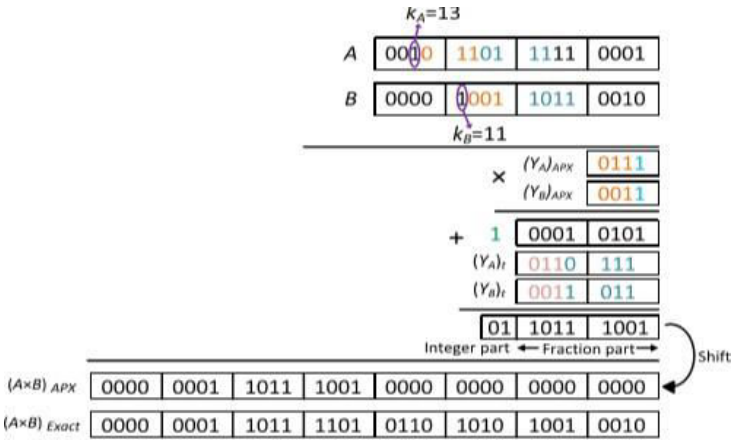
To improve the speed of calculation, we truncate $Y_A$and Y_B to t bits, where in the rest of this paper, we denote by $(Y_A)_t$and $(Y_B)_t$. Hence, we modify (8) as

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

$$A \times B \approx (A \times B)_{APX} = 2^{k_A+k_B} \times \left(\begin{matrix} 1 + (Y_A)_t + (Y_B)_t + \\ (Y_A)_{APX} \times (Y_B)_{APX} \end{matrix} \right) \quad (9)$$



where the width of $(Y_A)_{APX}$ ($(Y_B)_{APX}$) is h+1 bits. To have a better understanding, the dot diagram of the proposed algorithm for the case where t = 7 and h = 3 compared to that of an exact 16-bit multiplier is depicted in Fig. 2. The green square shows the "1" bit in the term $1+(Y_A)_t+(Y_B)_t+ (Y_A)_{APX}\times(Y_B)_{APX}$. Orange circles denote partial products of $(Y_A)_{APX}\times(Y_B)_{APX}$, whereas purple triangles show the bits of $(Y_A)_t$ and $(Y_B)_t$. Gray circles and triangles are omitted and are not considered in the calculations. As shown in Fig. 2, in the exact 16-bit multiplier, the number of partial products is equal to 256, which must be summed to generate the final result while in the proposed method, only 31 of the partial products are kept (which amounts to ~88% partial products reduction). This reduction rate will rise as the bit length of the multiplier input operands increases. As an example, the steps of multiplying A by B for the case of t = 7 and h = 3 are depicted in Fig. 3. In the rest of this paper, we denote our proposed structures by TOSAM (X, Y) where Xand Y correspond to h and t. The accuracy of the proposed approach depends on the values of the parameters t and h. Therefore, in the error analysis section (Section V), we will find a relationship between t and h parameters to achieve an almost high accuracy while having an acceptable speed and energy consumption. Finally, the proposed multiplication approach is feasible for the case of unsigned operands. To use it for signed multipliers, one may find the absolute value of the input operands, multiply them by the proposed algorithm, and fix the sign of the final result according to the sign of the input operands. Finding the exact absolute value of the input operands may degrade the speed of calculation and, hence, we produce it according to the method presented in [7].

Fig. 3. Numeric example of 16-bit TOSAM(3, 7) with A = 11761 and B=2482. The approximate result $[(A\times B)_{APX}]$ is equal to 28 901 376 while the exact result [(A× B)Exact] is equal to 29 190 802. In this case, the absolute error is 289 426 which is about 0.99% of the exact output (the error is less than 1% in this case).

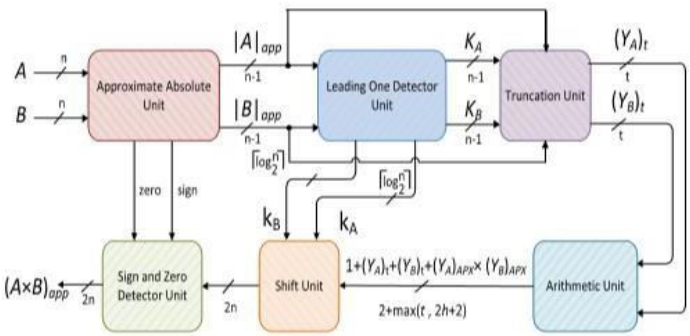


Fig. 4. Block diagram of the proposed approximate signed multiplier.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

IV.HARDWARE IMPLEMENTATION

The block diagram of the proposed signed approximate multiplier is depicted in Fig. 4. First, the approximate absolute value of the input operands (|A|app,|B|app) is determined using the Approximate Absolute Unit, similar to the one exploited in [7]. In this unit, the bits of the input are inverted if the input is negative and they are not changed if the input is positive. |A|app and |B|app are injected to the Leading-One Detector Unit [25] and the positions of their leading one bits are found using

$$K[i] = \left(\bigwedge_{j=i+1}^{n-2} \overline{I[j]}\right) \wedge I[i] \text{ for } 0 \le i \le n-2 \qquad (10)$$

where I can be either|A|app or|B|app. Only one bit of the signal K is "1" revealing the position of the input leading one bit. By using the KA and KB signals in a lookup table, kA and kB signals needed for (7) can be generated. The schematic of the Leading- OneDetector Unit for 8-bit input operandsis depicted in Fig. 5. For example, assume that |A|app = (011001)2, in this case K_A = (010000)2 and k_A =(100)2 =4. Signals $|A|_{app}$, $|B|_{app}$, K_A, andK_B are then applied to the Truncation Unit [25] to produce $(Y_A)_t$ and (YB)t. Assume that the input and output of this unit are I and (Y)t. In this case, the i^{th} bit of the output can be generated using

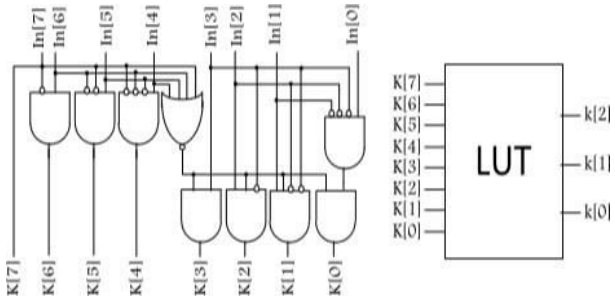


Fig. 5. Schematic of the Leading-One Detector Unit for 8-bit input operands.

$$(Y)_t[i] = \bigvee_{j=0}^{n-2} (K[j] \wedge I[j+i-t]) \text{ for } i < t. \qquad (11)$$

Signals $(Y_A)_t$ and $(Y_B)_t$ are then exerted to the Arithmetic Unit to calculate the term 1 + $(Y_A)_t$ + $(Y_B)_t$ + $(Y_A)_{APX}$ × $(Y_B)_{APX}$. It should be noted that the h most significant bits of $(Y_A)_{APX}$ and $(Y_B)_{APX}$ are the same as the h most significant bits of $(Y_A)_t$ and $(Y_B)_t$ whose rightmost bits are always "1." Hence, there is no need to add extra hardware to generate $(Y_A)_{APX}$ and $(Y_B)_B$ signals which are produced by simple wiring. In the Shift Unit, the output of the Arithmetic Unit is shifted to left by kA+kB to produce the term $2k_A+k_B$ ×(1+$(Y_A)_t$ + $(Y_B)_t$ +$(Y_A)_{APX}$ ×$(Y_B)_{APX}$) [see (9)]. In the Sign and Zero Detector Unit, the output's sign is determined by the sign of the multiplier input operands, and the output is set to zero if at least one of the inputs is zero. For unsigned multipliers, the Approximate Absolute Unit should be omitted, and the Sign and Zero Detector Unit should be replaced with a Zero Detector Unit.

TOSAM can be implemented in an accuracy-configurable structure. To achieve this, all units of TOSAM should be designed to accommodate the largest desired values of h and t, allowing the design to function in all operation modes. We propose a configurable TOSAM structure with three different operating modes: T2, T6, and T9, corresponding to TOSAM(0, 2), TOSAM(2, 6), and TOSAM(5, 9), respectively. The Truncation and Shift Units of the configurable TOSAM should be designed for the maximum values of t and h (h = 5 and t = 9 in this case).

In the Arithmetic Unit, some adders and logical AND gates should be power-gated based on the operating mode to enhance power efficiency. The reduction levels of the partial products based on the operating modes are illustrated in Fig. 6. In the final level, a fast 9-bit adder is used, and to decrease its switching activity, some

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

inputs are set to "0" using a transmission gate (TG), depending on the operating mode.

In T2 mode, only the purple partial products are accumulated, only the purple adders are active (not power-gated), and all inputs of the 9-bit adder are set to "0." Additionally, the 10 least significant bits of the result are set to "0.".

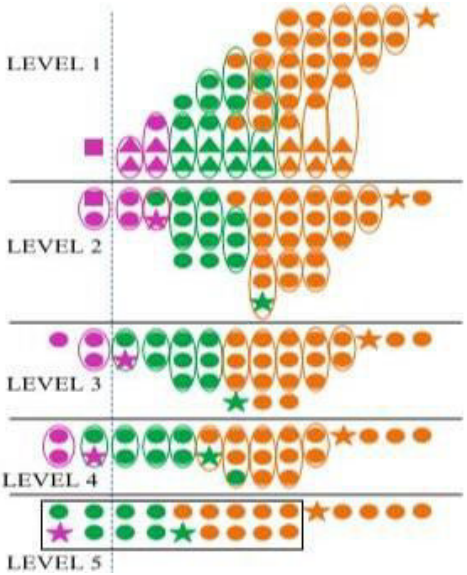


Fig. 6. Reduction levels of accuracy configurable TOSAM with three different operating modes.

TGs and purple stars are used to generate the four most significant bits of the output. In T6 mode, only the green and purple partial products are generated and summed to form the final output, with the orange adders power-gated and their inputs set to "0." Additionally, the two orange circles in the eighth column of LEVEL1 should be set to "0" by TGs in T6 mode. Here, the six least significant bits of the result are set to "0," the green stars pass through TGs to produce four intermediate bits, and the 9-bit adder generates the four most significant bits of the result.

In T9 mode, all components are active. The orange stars pass through the TGs to generate the five least significant bits of the result, with the remaining bits produced by the 9-bit adder. The least significant bits of (Y A)APX and (Y B)APX, which depend on the operating mode, should be rounded (set to "1"). This is achieved by performing a logical OR operation on the corresponding bit and the operating mode. For instance, when the T6 signal is "1," the logical OR operation sets the corresponding bit of (Y A)APX and (Y B)APX to "1.".

V. CONCLUSION

Table: Comparisons between TOSAM and Proposed Modified TOSAM

TITLE	DELAY(ns)
Previous work (TOSAM)	23.363
Proposed work (Modified TOSAM with energy efficient high speed approximate multiplier)	22.389

In this paper, we proposed a low-energy and area-efficient approximate multiplier. The input operations were truncated to two different lengths, t and h, and then rounded to the nearest odd numbers to minimize truncation errors. Compared to a standard Wallace Multiplier, the proposed 32-bit multiplier improved energy efficiency by an average of 95% and reduced area usage by 85%. However, the multiplication delay and power consumption increased by 4% to 41% and 89% to 97%, respectively.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

The enhancements in speed, size, and energy efficiency of the new multiplier became more pronounced as the multiplier size increased, thanks to its simple and scalable calculation core. The high accuracy of the proposed multiplier makes it a strong candidate for applications such as image analysis and classification.

REFERENCES

[1] 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.

[2] S. Xu and B. C. Schafer, "Exposing approximate computing optimizations at different levels: From behavioral to gate-level," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 11, pp. 3077– 3088, Nov. 2017.

[3] A. Raghunathan and K. Roy, "Approximate computing: Energyefficient computing with good- enough results," in Proc. IEEE 19th Int. On-Line Test. Symp. (IOLTS), Chania, Greece, Jul. 2013, p. 258.

[4] 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, Nov. 2015, pp. 418–425.

[5] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali- Kusha, and M. Pedram, "RoBa multiplier: A rounding-based approximate multiplier for high- speed yet energy-efficient digital signal processing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 2, pp. 393–401, Feb. 2017.

[6] V. Leon, G. Zervakis, D. Soudris, and K. Pekmestzi, "Approximate hybrid high radix encoding for energy-efficient inexact multipliers," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 3, pp. 421–430, Mar. 2018.

[7] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Dual-quality 4:2 compressors for utilizing in dynamic accuracy configurable multipliers," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 4, pp. 1352–1361, Apr. 2017.

[8] M. Ha and S. Lee, "Multipliers with approximate 4–2 compressors and error recovery modules," IEEE Embedded Syst. Lett., vol. 10, no. 1, pp. 6–9, Mar. 2018.

[9] R. Marimuthu, Y. E. Rezinold, and P. Mallick, "Design and analysis of multiplier using approximate 15–4 compressor," IEEE Access, vol. 5, pp. 1027–1036, 2017.

[10] D. Esposito, A. G. M. Strollo, E. Napoli, D. De Caro, and N. Petra, "Approximate multipliers based on new approximate compressors," IEEETrans. Circuits Syst. I, Reg. Papers, vol. 65, no. 12, pp. 4169–4182, Dec. 2018.

[11] D. Esposito, D. De Caro, E. Napoli, N. Petra, and A. G. M. Strollo, "On the use of approximate adders in carry-save multiplieraccumulators," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Baltimore, MD, USA, May 2017, pp. 1–4.

[12] H. J. Ko and S. F. Hsiao, "Design and application of faithfully rounded and truncated multipliers with combined deletion, reduction, truncation, and rounding," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 5, pp. 304–308, May 2011.

[13] A. Lingamneni, A. Basu, C. Enz, K. V. Palem, and C. Piguet, "Improving energy gains of inexact DSP hardware through reciprocative error compensation," in Proc. 50th ACM/EDAC/IEEE Design Automat. Conf. (DAC), Austin, TX, USA, May/Jun. 2013, pp. 1–8.

[14] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, and F. Lombardi, "Design of approximate radix-4 booth multipliers for error-tolerant computing," IEEE Trans. Comput., vol. 66, no. 8, pp. 1435–1441, Aug. 2017.

[15] S. Venkatachalam, H. J. Lee, and S.-B. Ko, "Power efficient approximate booth multiplier," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Florence, Italy, May 2018, pp. 1–4.

[16] H. Jiang, J. Han, F. Qiao, and F. Lombardi, "Approximate radix-8 booth multipliers for low- power and high-performance operation," IEEE Trans. Comput., vol. 65, no. 8, pp. 2638–2644, Aug. 2016.

[17] I. Alouani, H. Ahangari, O. Ozturk, and S. Niar, "A novel heterogeneous approximate multiplier for low power and high performance," IEEE Embedded Syst. Lett., vol. 10, no. 2, pp. 45–48, Jun. 2018.

[18] K. H. Abed and R. E. Siferd, "CMOS VLSI implementation of a lowpower logarithmic converter," IEEE Trans. Comput., vol. 52, no. 11, pp. 1421–1433, Nov. 2003.

[19] M. Combet, H. Van Zonneveld, and L. Verbeek, "Computation of the base two logarithm of binary numbers," IEEE Trans. Electron. Comput., vol. EC- 14, no. 6, pp. 863–867, Dec. 1965.

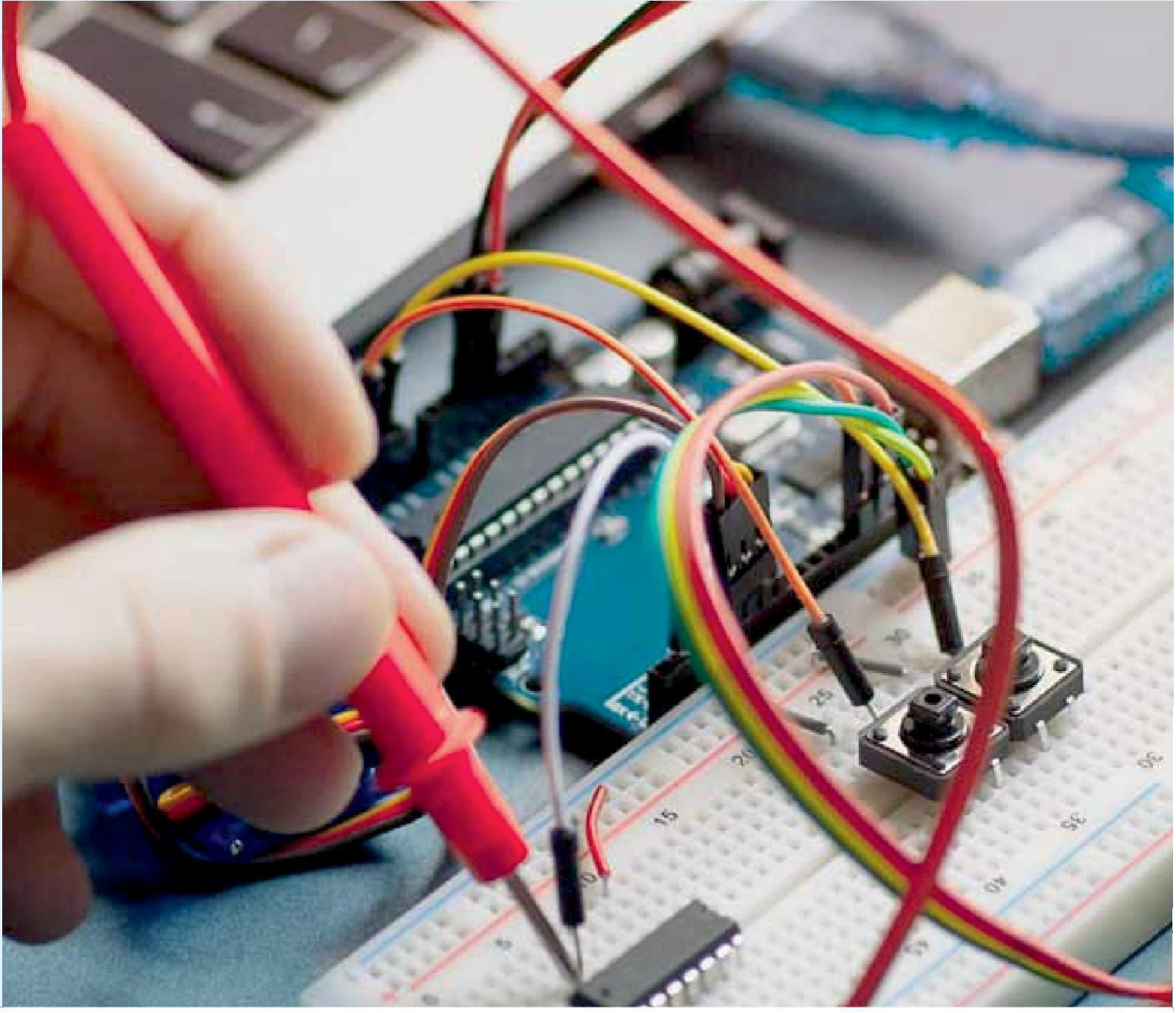
[20] J. N. Mitchell, "Computer multiplication and division using binary logarithms," IRE Trans. Electron. Comput., vol. EC-11, no. 4, pp. 512–517, Aug. 1962.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 8.317|| A Monthly Peer Reviewed & Referred Journal |

||Volume 13, Issue 6, June 2024||

|DOI:10.15662/IJAREEIE.2024.1306009|

[21] Z. Babic, A. Avramovic, and P. Bulic, "An iterative Mitchell's algorithm based multiplier," in Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT), Sarajevo, Serbia, Dec. 2008, pp. 303–308.
[22] M. S. Kim, A. A. Del Barrio, R. Hermida, and N. Bagherzadeh, "Lowpower implementation of Mitchell's approximate logarithmic multiplication for convolutional neural networks," in Proc. 23rd Asia South Pacific Design Automat. Conf. (ASP- DAC), Jeju, South Korea, Jan. 2018, pp. 617–622.
[23] S. Vahdat, M. Kamal, A. Afzali-Kusha, M. Pedram, and Z. Navabi, "TruncApp: A truncation- based approximate divider for energy efficient DSP applications," in Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE), Lausanne, Switzerland, Mar. 2017, pp. 1635–1638.
[24] 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.
[25] Nangate. 45 nm Open Cell Library. Accessed: 2012. [Online]. Available: http://www.nangate.com
[26] The USC-SIPI Image Database. Accessed: Jun. 2017.[Online]. Available: http://sipi.usc.edu/database
[27] 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.
[28] H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.
[29] Y. LeCun, C. Cortes, and C. J. C. Burges. (1998).



INNO SPACE
SJIF Scientific Journal Impact Factor

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

International Journal of Advanced Research

in Electrical, Electronics and Instrumentation Engineering

9940 572 462 6381 907 438 ijareeie@gmail.com

Scan to save the contact details