

An Ultra-Efficient Approximate Multiplier With Error Compensation for Error-Resilient Applications

Farnaz Sabetzadeh, Mohammad Hossein Moaiyeri¹, *Senior Member, IEEE*, and Mohammad Ahmadinejad

Abstract—Approximate computing is a promising paradigm for trading off accuracy to improve hardware efficiency in error-resilient applications such as neural networks and image processing. This brief presents an ultra-efficient approximate multiplier with error compensation capability. The proposed multiplier considers the least significant half of the product a constant compensation term. The other half is calculated precisely to provide an ultra-efficient hardware-accuracy trade-off. Furthermore, a low-complexity but effective error compensation module (ECM) is presented, significantly improving accuracy. The proposed multiplier is simulated using HSPICE with 7nm tri-gate FinFET technology. The proposed design significantly improves the energy-delay product, on average, by 77% and 54% compared to the exact and existing approximate designs. Moreover, the proposed multiplier's accuracy and effectiveness in neural networks and image multiplication are evaluated using MATLAB simulations. The results indicate that the proposed multiplier offers high accuracy comparable to the exact multiplier in NNs and provides an average PSNR of more than 51dB in image multiplication. Accordingly, it can be an effective alternative for exact multipliers in practical error-resilient applications.

Index Terms—Error compensation, approximate multiplier, neural network, hardware-accuracy trade-off.

I. INTRODUCTION

THE RAPID growth of the digital circuit industry, followed by the increase in density and complexity of digital integrated circuits at the deep nanometer dimensions, has motivated circuit designers to offer new approaches at various levels of design abstraction. One of the practical solutions in designing energy-efficient nanoscale digital circuits is approximate computing [1]. Complete accuracy is not the first concern for many practical applications. Accurate calculations are not always necessary in many error-resistant applications such as artificial neural networks (ANNs), speech and image recognition, and multimedia applications. Such applications can tolerate inaccuracy while producing useful results that, for example, are sufficiently understandable to humans. Accordingly, circuit parameters such as transistor count, power

consumption, delay, and area can be reduced by reasonably reducing the accuracy [1].

Multiplication is one of the essential arithmetic operations in microprocessors and digital signal processing units. Moreover, multipliers are widely used in neural networks. Convolution layers in convolutional neural networks (CNNs) involve a large number of multiplication-accumulation (MAC) operations. Multiplications are much more complex than additions for the MAC operations and consume the most resources. Therefore, reducing multipliers' energy consumption and hardware costs is very important [2], [3], [4].

The most effective solution for trading-off accuracy for hardware efficiency in error-resilient applications is to design efficient approximate multipliers. There are two general approaches for designing approximate multipliers. The first method uses approximate adders and compressors in the structure of the conventional multiplier [1], [2], while the second approach modifies the multiplier structure to reach an approximate design. A practical way to prevent errors in such multipliers is to use error compensation modules (ECMs). By combining efficient ECMs and proper truncation, efficient approximate multipliers can be achieved [6], [7], [8], [9].

This brief proposes a new approximate multiplier with an ultra-efficient error compensation module. The proposed design has an ultra-low-complexity structure and significantly reduces the number of transistors and energy consumption compared to its counterparts. At the same time, it offers a high enough accuracy for error-resilient applications such as neural networks and image processing. The proposed multiplier consists of three main parts, i.e., a constant-truncated region making hardware-accuracy trade-off, a new efficient error compensation module, and an exact part. The negativeness of all the error distances (EDs) resulting from the constant-truncated part is applied to design an ultra-efficient ECM. Moreover, as the absolute ED for each input equals the number of 1's in that input, assuming that the input bits to a multiplier are distributed uniformly in general, an input with a higher error distance has a lower probability of occurrence. Accordingly, we propose an error compensation module to compensate for the errors in cases with high occurrence probability. Our ECM is designed using only two four-input OR gates with 20 transistors.

Moreover, utilization of the proposed approximate multiplier in neural networks and image processing applications is investigated. The results indicate that the proposed design provides an ultra-efficient trade-off between hardware efficiency and accuracy in error-resilient applications.

Manuscript received 15 September 2022; accepted 13 October 2022. Date of publication 17 October 2022; date of current version 9 February 2023. This brief was recommended by Associate Editor Y. Xia. (*Corresponding author: Mohammad Hossein Moaiyeri.*)

The authors are with the Faculty of Electrical Engineering, Shahid Beheshti University, Tehran 1983963113, Iran (e-mail: f_sabetzadeh@sbu.ac.ir; h_moaiyeri@sbu.ac.ir; mo.ahmadinejad@mail.sbu.ac.ir).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCSII.2022.3215065>.

Digital Object Identifier 10.1109/TCSII.2022.3215065

The rest of this brief is organized as follows: Section II briefly reviews the existing designs. Section III describes the proposed approach. The simulation results and comparisons are presented in Section IV. Finally, Section V concludes this brief.

II. RELATED WORK

Several signed and unsigned approximate multipliers have already been presented in the literature [1], [2], [9], [10], [11], [12], [13], [14], [15], [16], [17]. However, in this brief, we have focused on unsigned multipliers because of the higher use of unsigned multiplication in most approximate computing applications, such as image processing and machine vision.

Generally, there are two approaches for designing approximate multipliers. The first group includes those designed without an error compensation module (ECM) [1], [2], [9], [10], [14], [15], [16], [17]. The other one includes designs with ECM that can compensate for the errors to a large extent [5], [6], [7], [8].

In [1], the conventional structure of the 8-bit Dadda multiplier, used as the reference in most previous research, was described. Three hybrid approximate multipliers based on different approximate compressors with different accuracy and performance characteristics were proposed in [2]. Two imprecise 4×4 multipliers were proposed in [9] utilized to construct a high-order multiplier. In [10], compressors of different orders with equal output weights were introduced to be used in approximate multipliers. Approximate recursive multipliers based on high-performance building blocks were presented in [14]. In [15], approximate multipliers using static segmentation with error correction methods were investigated. In [16], logarithmic multipliers for low-power error-tolerant applications were proposed. Dynamic range approximate logarithmic multipliers for machine learning applications were suggested in [17]. Logarithmic approximate multipliers generally have a higher error but lower energy consumption.

In [5], an innovative modification to the approximate 4-2 compressor design and an error recovery module were suggested. In [6], an energy-efficient approximate multiplier using a modified approximate 4:2 compressor was proposed. An error recovery module that can detect erroneous conditions was suggested in [6]. In [7], three novel approximate 4-2 compressors were proposed and utilized in 8-bit multipliers. Meanwhile, an ECM was presented to promote the accuracy of the approximate multiplier. In [8], an unsigned approximate multiplier architecture based on truncation and approximation was proposed to improve hardware saving without significantly sacrificing accuracy. In this design, the partial products in the middle portion were simplified using 4:2 approximate compressors, and the error due to approximation was compensated using an efficient ECM. The most significant portion of the multiplier was implemented using exact logic.

III. PROPOSED APPROXIMATE MULTIPLIER

The partial product reduction unit of the proposed approximate multiplier is shown in Fig. 1. As low-bit

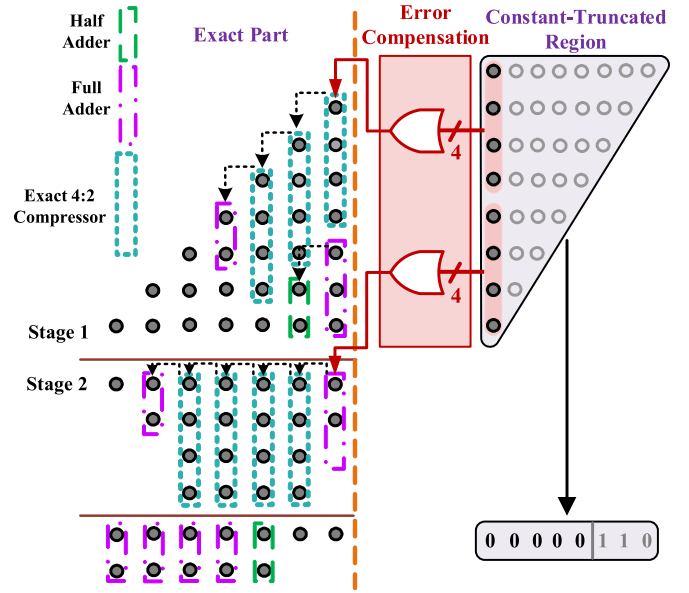


Fig. 1. Partial product reduction of the proposed approximate multiplier.

TABLE I
ERROR DISTANCES FOR ALL INPUT COMBINATIONS

x_4	x_3	x_2	x_1	Sum=0	Carry=0	Error distance
0	0	0	0	0	0	0
0	0	0	1	0	0	-1
0	0	1	0	0	0	-1
0	0	1	1	0	0	-2
0	1	0	0	0	0	-1
0	1	0	1	0	0	-2
0	1	1	0	0	0	-2
0	1	1	1	0	0	-3
1	0	0	0	0	0	-1
1	0	0	1	0	0	-2
1	0	1	0	0	0	-2
1	0	1	1	0	0	-3
1	1	0	0	0	0	-2
1	1	0	1	0	0	-3
1	1	1	0	0	0	-3
1	1	1	1	0	0	-4

multipliers are widely used in machine learning and image multiplication applications [2], [18], we focus on the 8-bit Dadda multiplier without loss of generality. However, our proposed approximate design can be scaled for larger bit-widths with a straightforward extension.

The proposed multiplier consists of a constant-truncated part, an error correction module, and an exact part. The partial products are not generated in the first eight least significant columns of the constant region, and an 8-bit constant product ("00000110") appears in this part. Suppose the left nibble of the constant-truncated region with inaccurate compressors. In that case, it is equivalent to having approximate compressors producing Carry and Sum equal to '0' for all inputs. The errors generated in this section for all inputs are given in Table I.

Although only one of the outputs is accurate, some interesting points in this design result in an ultra-efficient multiplier for a compromise between hardware and accuracy.

First, by zeroing the outputs, the accurate output is generated for the “0000” combination, most likely to occur at the input. Second, the negative EDs for the inaccurate inputs can be well used to design an efficient error compensation module.

Another noteworthy point is that since, in this scheme, the ED corresponding to each input is equal to the number of 1’s in that input, the higher ED a state has, the less likely that state occurs. Assuming the input bits to a multiplier are distributed uniformly in general (considering an equal probability for an input bit to be ‘0’ or ‘1’) [2], [5], [8], [10], the probability that a partial product (an input bit to a compressor), generated by a 2-input AND gate, equals to ‘1’ (‘0’) is 1/4 (3/4). Therefore, the probabilities of inputs having zero, one, two, three, and four 1’s equal 31.5%, 10.5%, 3.5%, 1.1%, and 0.4%, respectively.

Accordingly, the proposed error compensation module is designed to compensate for the errors resulting from the cases with higher occurrence probabilities. Moreover, the output corresponding to the input “0000” is not corrupted.

For this purpose, two 4-input OR gates with a total number of 20 transistors are used. By ORing the four inputs, the errors corresponding to the cells highlighted in Table I are determined. Consequently, by feeding the output of the OR gates as the Carry input to the following stage’s compressors, the impact of the existing negative EDs is compensated to a large extent. It is also worth noting that the existence of only negative EDs is appropriate for neural network applications due to the ReLU activation function [3].

Although the least significant columns are usually truncated to improve hardware efficiency, it can result in a relative accuracy loss and can be compensated. Accordingly, the three LSBs of the products are replaced with the constant correction term of $(6)_{10} = “110”$ to improve the accuracy. This constant correction term is computed as an average value of the input combinations in the least significant columns [8]. This simplification omits 28 two-input AND gates from the partial production stage, and many adder and compressor circuits form the following stages.

As shown in Fig. 1, in the first stage of the reduction part of the proposed multiplier, a half adder, two full adders, and three 4:2 exact compressors are used. Moreover, only two full adders and four 4:2 exact compressors are used in the second stage. Finally, an RCA consisting of two half adders (HA) and four full adders (FA) generates the product. The proposed multiplier’s ultra-low-complexity and efficient structure significantly reduces hardware overhead, delay, and power consumption while providing high accuracy for error-resilient applications such as neural networks.

IV. SIMULATION RESULTS AND COMPARISONS

Several exact and approximate multipliers are simulated and compared in this section, considering various aspects of performance and accuracy. Different multipliers with and without error compensation capability are considered to have a more comprehensive comparison.

TABLE II
PERFORMANCE COMPARISON OF VARIOUS MULTIPLIERS

Multiplier	Delay (ps)	Power (μw)	PDP (fJ)	EDP (fJ.ps)	Transistors	Area (μm ²)
Proposed	187	7.98	1.49	278	748	6.23
[5]	247	13.41	3.31	819	1302	10.85
[6]	280	16.52	4.62	1295	1452	12.08
[7]_1	255	15.34	3.91	999	1364	11.35
[7]_2	255	15.62	3.99	1017	1392	11.59
[7]_3	255	15.28	3.90	995	1360	11.32
[7]_4	255	15.23	3.89	992	1344	11.19
[8]	199	14.49	2.88	574	1296	10.79
[1]	222	10.94	2.43	539	1180	9.8
[2]	188	8.72	1.64	308	826	6.88
[9]	240	13.50	3.24	776	1248	10.39
[10]	237	10.74	2.54	601	1180	9.82
[14]	238	14.56	3.47	825	1190	9.9
[15]	221	10.27	2.27	502	1036	8.6
[16]	218	7.80	1.70	371	706	5.8
[17]	201	7.75	1.56	310	698	5.7
Exact [4]	253	18.80	4.75	1201	1530	12.74

A. Hardware Analysis

The proposed multiplier and the other exact and approximate multipliers have been simulated using HSPICE and the 7nm FinFET model [19]. The simulations are conducted at 0.7V supply voltage and 2GHz frequency. The worst-case critical path delay of each multiplier is reported as the delay. A long stream of random input combinations, considering the switching activity factor, have been generated and used as stimuli to estimate the power consumption of the multipliers.

The simulation results are tabulated in Table II. The results indicate that the proposed multiplier has superior delay, power, power-delay product (PDP), energy-delay product (EDP), and area compared to the existing multipliers, particularly the multipliers with ECM. The low-complexity structure of the proposed multiplier leads to a lower PDP even compared to hardware-efficient logarithmic multipliers. This superiority is due to the ultra-efficient structure and short critical path of our multiplier discussed in Section III. On average, the proposed multiplier improves the delay, power, PDP, EDP, and area by 23%, 34%, 46%, 55%, and 33%, respectively, compared to the existing multipliers listed in Table II.

B. Accuracy Analysis

To evaluate the accuracy of the approximate multipliers, the normalized mean error distance (NMED), the mean relative error distance (MRED), and the number of effective bits (NoEB) metrics [2], [9], [10] have been calculated using MATLAB simulations by applying all of the 65536 possible input combinations. The results are given in Table III.

Although the proposed approximate multiplier shows higher accuracy metrics than the other multipliers with error compensation capability [5], [6], [7], [8], in most cases, it is significantly more accurate than the other counterparts due to its structure and the efficient error compensation module.

Moreover, as shown in Table II, the proposed multiplier significantly improves the hardware metrics especially compared

TABLE III
ACCURACY METRICS OF THE APPROXIMATE MULTIPLIERS

Multipliers	NMED ($\times 10^{-4}$)	MRED ($\times 10^{-2}$)	NoEB
Proposed	17	3.52	9.0
[5]	4.8	1.00	10.3
[6]	4.0	0.74	10.4
[7]_1	8.0	4.61	10
[7]_2	5.0	1.41	10.5
[7]_3	5.8	1.54	10.3
[7]_4	7.2	1.83	10
[8]	4.2	0.70	10.4
[1]	70	4.97	4.1
[2]	40	4.40	6.2
[9]	170	6.50	5.1
[10]	120	4.80	5.1
[14]	52	7.50	7.2
[15]	118	5.49	6.0
[16]	114	4.48	5.8
[17]	90	3.75	6.1

to the multipliers with ECM. It is worth mentioning that the primary purpose of approximate computing is to enhance hardware efficiency while providing an acceptable accuracy for error-resilient applications.

C. Pareto Diagrams

The Pareto diagrams shown in Fig. 2 compare the multipliers considering the accuracy metrics and PDP to identify designs with more effective trade-offs between accuracy and energy efficiency. The multipliers near the bottom-left corner of the Pareto diagrams are more efficient, having lower PDPs and higher accuracies. According to the results, the proposed multiplier makes a more effective trade-off between accuracy and energy efficiency than its counterpart designs with and without ECM.

D. Applications

Neural networks are the most effective solution for many challenging applications, such as speech recognition and image classification. Approximate multipliers with lower complexity can significantly enhance the efficiency of these intrinsically error-tolerant networks [2], [3].

This section evaluates the efficacy of the approximate multipliers with and without ECM in two neural network structures. The first one is a multilayer perceptron (MLP), by which we classify the MNIST (Mixed National Institute of Standards and Technology) dataset [20], and the other is a convolutional neural network (CNN) used to classify the SVHN (Street View House Numbers) dataset [21]. We have classified the MNIST dataset using an MLP network explained in [2], [3]. The LeNet-5 network described in [22] has also been used to classify the SVHN dataset.

It is worth mentioning that negative and positive neuron weights are produced in the training process. Accordingly, we have converted the neuron weights to the sign-magnitude representation after the training process, and accordingly, the multiplications were performed using the unsigned multipliers under study. As the neuron weights are converted once after training, it causes no overhead in the classification process, where the approximate designs are used. Suppose the neural

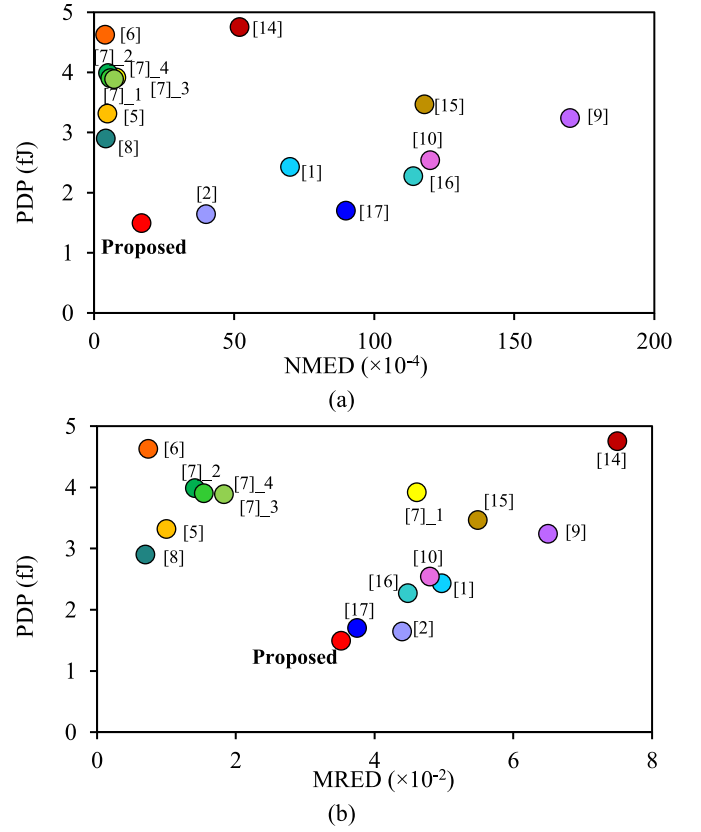


Fig. 2. Pareto diagrams (a) PDP vs. NMED (b) PDP vs. MRED.

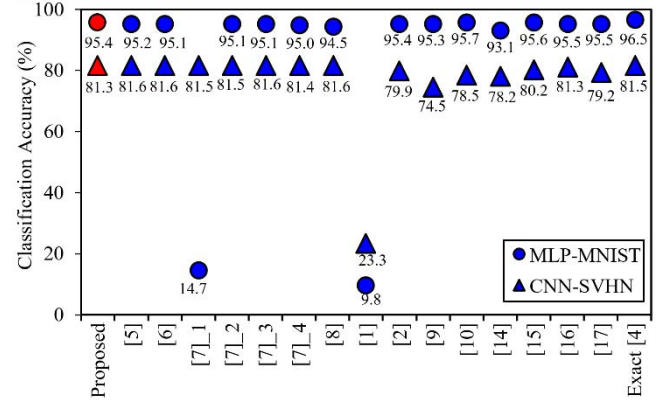


Fig. 3. Classification accuracies using the multipliers under investigation.

network outputs should be converted to 2's complement. In that case, the overhead is negligible compared to the whole network.

These two types of neural networks have been implemented and simulated in the MATLAB environment using the exact and approximate multipliers. The results are shown in Fig. 3. According to the results, the classification accuracies for the MNIST and SVHN datasets using the proposed approximate multiplier are 95.5% and 81.3%, comparable to the exact design. Accordingly, the proposed design can be effectively used to replace exact multipliers in neural network applications.

Image multiplication is considered for image processing applications. Multiplications of various images have been

TABLE IV
AVERAGE PSNR AND MSSIM VALUES FOR APPROXIMATE
IMAGE MULTIPLICATIONS

Multipliers	PSNR (dB)	MSSIM
Proposed	51.6	0.9994
[5]	56.9	0.9997
[6]	58.0	0.9997
[7] 1	55.3	0.9995
[7] 2	57.1	0.9997
[7] 3	56.5	0.9997
[7] 4	55.4	0.9995
[8]	57.8	0.9997
[1]	41.4	0.9890
[2]	41.7	0.9941
[9]	31.7	0.9740
[10]	36.7	0.9932
[14]	31.6	0.9761
[15]	36.0	0.9914
[16]	34.1	0.9832
[17]	33.8	0.9846

performed by the approximate multipliers understudy using MATLAB. The average values for the PSNR and MSSIM quality metrics [2] are tabulated in Table IV. According to the results, the proposed multiplier offers the average PSNR and MSSIM values of more than 51dB and 0.9999, respectively, which are more than enough for image processing applications.

At the same time, our proposed approximate multiplier improves the circuit parameters significantly, as reported in Table II. Hence, our design can be well applied in image processing applications to achieve hardware and energy efficiencies with high quality.

V. CONCLUSION

This brief proposes an ultra-efficient approximate multiplier, including a constant-truncated region, an error correction module, and an exact part. Ignoring the outputs in the truncated region is equivalent to using a 4:2 approximate compressor with outputs equal to zero. Accordingly, it leads to an accurate output for the “0000” input, which is most likely to occur. Moreover, the EDs of the other inputs become negative. As the absolute ED corresponding to each input equals the number 1’s in that input, the input with a higher ED has a lower probability of occurrence. Accordingly, an efficient ECM is proposed to compensate for the errors resulting from the most probable cases to a large extent. Also, the existence of entirely negative EDs can be more efficient for NN applications due to the functionality of the ReLU activation function. The proposed design’s ultra-low-complexity structure reduces EDP by 77% and 54% compared to the existing exact and approximate multipliers. Moreover, the proposed multiplier offers high accuracy and quality in error-resilient applications such as neural networks and image processing. Accordingly, the proposed design can be an effective alternative for exact multipliers in practical error-resilient applications.

REFERENCES

[1] 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.

[2] M. Ahmadijeh and M. H. Moaiyeri, “Energy- and quality-efficient approximate multipliers for neural network and image processing applications,” *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 1105–1116, Apr.–Jun. 2022, doi: [10.1109/TETC.2021.3072666](https://doi.org/10.1109/TETC.2021.3072666).

[3] M. S. Kim, A. A. D. Barrio, H. Kim, and N. Bagherzadeh, “The effects of approximate multiplication on convolutional neural networks,” *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 904–916, Apr.–Jun. 2022, doi: [10.1109/TETC.2021.3050989](https://doi.org/10.1109/TETC.2021.3050989).

[4] A. Arasteh, M. H. Moaiyeri, M. Taheri, K. Navi, and N. Bagherzadeh, “An energy and 4 efficient 4:2 compressor based on FinFET,” *Integr. VLSI J.*, vol. 60, pp. 224–231, Jan. 2018.

[5] 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.

[6] A. G. M. Strollo, D. De Caro, E. Napoli, N. Petra, and G. Di Meo, “Low-power approximate multiplier with error recovery using a new approximate 4-2 compressor,” in *Proc. IEEE Int. Symp. Circuits Syst.*, 2020, pp. 1–4.

[7] H. Pei, X. Yi, H. Zhou, and Y. He, “Design of ultra-low power consumption approximate 4–2 compressors based on the compensation characteristic,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 1, pp. 461–465, Jan. 2021.

[8] U. A. Kumar, S. K. Chatterjee, and S. E. Ahmed, “Low-power compressor-based approximate multipliers with error correcting module,” *IEEE Embedded Syst. Lett.*, vol. 14, no. 2, pp. 59–62, Jun. 2022, doi: [10.1109/LES.2021.3113005](https://doi.org/10.1109/LES.2021.3113005).

[9] M. S. Ansari, H. Jiang, B. F. Cockburn, and J. Han, “Low-power approximate multipliers using encoded partial products and approximate compressors,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 3, pp. 404–416, Sep. 2018.

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

[11] R. Pilipović, P. Bulić, and U. Lotrič, “A two-stage operand trimming approximate logarithmic multiplier,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 6, pp. 2535–2545, Jun. 2021.

[12] 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.

[13] V. Leon, K. Asimakopoulos, S. Xydis, D. Soudris, and K. Pekmestzi, “Cooperative arithmetic-aware approximation techniques for energy-efficient multipliers,” in *Proc. 56th Annu. Des. Autom. Conf. (DAC)*, 2019, pp. 1–6.

[14] H. Waris, C. Wang, C. Xu, and W. Liu, “AxRMs: Approximate recursive multipliers using high-performance building blocks,” *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 1229–1235, Apr.–Jun. 2022.

[15] A. G. M. Strollo, E. Napoli, D. De Caro, N. Petra, G. Saggese, and G. Di Meo, “Approximate multipliers using static segmentation: Error analysis and improvements,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 69, no. 6, pp. 2449–2462, Jun. 2022.

[16] W. Liu, J. Xu, D. Wang, C. Wang, P. Montuschi, and F. Lombardi, “Design and evaluation of approximate logarithmic multipliers for low power error-tolerant applications,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 9, pp. 2856–2868, Sep. 2018.

[17] P. Yin, C. Wang, H. Waris, W. Liu, Y. Han, and F. Lombardi, “Design and analysis of energy-efficient dynamic range approximate logarithmic multipliers for machine learning,” *IEEE Trans. Sustain. Comput.*, vol. 6, no. 4, pp. 612–625, Oct.–Dec. 2021.

[18] H. Afzali-Kusha, M. Vaeztourshizi, M. Kamal, and M. Pedram, “Design exploration of energy-efficient accuracy-configurable dadda multipliers with improved lifetime based on voltage overscaling,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 5, pp. 1207–1220, May 2020.

[19] L. T. Clark et al., “ASAP7: A 7-nm FinFET predictive process design kit,” *Microelectron. J.*, vol. 53, pp. 105–115, Jul. 2016.

[20] Y. LeCun, C. Cortes, and C. Burges. “MNIST handwritten digit database.” AT&T Labs. 2010. [Online]. Available: <http://yann.lecun.com/exdb/mnist>

[21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011, p. 5.

[22] Y. Zhou, S. Song, and N.-M. Cheung, “On classification of distorted images with deep convolutional neural networks,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 1213–1217.