A Compact Low-Power Mitchell-Based Error Tolerant Multiplier

Aly Sultan[§], Ali H. Hassan[*], and Hassan Mostafa[*†]
[§]Department of Electronics and Communication Engineering, American University in Cairo (AUC), New Cairo 11835, Egypt
[*]Electronics and Communications Engineering Department, Cairo University, Giza 12613, Egypt
[†]Nanotechnology and Nanoelectronics Program, Zewail City of Science and Technology, Sheikh Zayed, 12588, Egypt
Email: asultan@aucegypt.edu, ali.h.hassan@ieee.org, and hmostafa@uwaterloo.ca

***Abstract*—Power consumption is a crucial design aspect in multimedia and machine learning applications. Approximate computing offers an energy-efficient approach for both power reduction and area optimization. In this paper, a hybrid approximation methodology based on error tolerant multipliers (ETMs) is introduced. The proposed design splits the approximation process into two parts: (1) approximating the most significant bits (MSBs) using approximate logarithms and (2) approximating the least significant bits (LSBs) using truncation. A prototype of the proposed multiplier is demonstrated with an image processing application (JPEG compression) using a Discrete Cosine Transform (DCT) where the power delay product (PDP) is improved by 1.9X. And the area utilization is reduced by 2.7X with only 20% reduction in the output image peak signal-to-noise ratio (PSNR).**

***Index Terms*—approximate computing, approximate multiplier, low-power, truncation, error-tolerant, power-efficient.**

I. Introduction

Due to the high prevalence of embedded systems based on machine learning, there is a high demand for designing an ultra-low-power, small footprint, and high-performance hardware. Approximate computing satisfies this demand by trading the computational accuracy versus the superior performance, power reduction, and area optimization. Moreover, approximate computing represents a good candidate for multiple applications such as media processing (audio, video, graphics, and image), recognition, machine learning, and data mining [1]. A literature survey [[2], [3], [4], [5], [6], [7], [8], [9] about the different approximate multipliers classifies the approximation methodologies as follows:

1) Approximation in partial product tree [2]
2) Approximation in partial products summation [3], [4]
3) Multiplication using approximate logarithms [5], [6], [7]
4) Approximate hardware generation using a genetic algorithm [8]
5) Error-tolerant multiplication (ETM) [9]

In this paper, a hybrid design based on ETM approximate multiplier (HETM) is presented. The proposed design reduces the utilization area significantly compared to the other architectures. Correspondingly, the power consumption is optimized. A prototype of HETM is evaluated using ZYNQ XCZ 7020-1ClG484 FPGA board. Furthermore, the proposed HETM is tested with JPEG compression of a standard image.

The rest of the paper is organized as follows: in Section II discusses the proposed architecture as well as the energy-efficient approximate multipliers. Then Section III details the error and circuit performance characteristics of the proposed HETM. And in Section VI presents an application for evaluating the different HETMs using JPEG compression. Finally, a conclusion is driven in section V.

II. Proposed Hybrid Error Tolerant Multiplier

A. Error Tolerant Multiplier

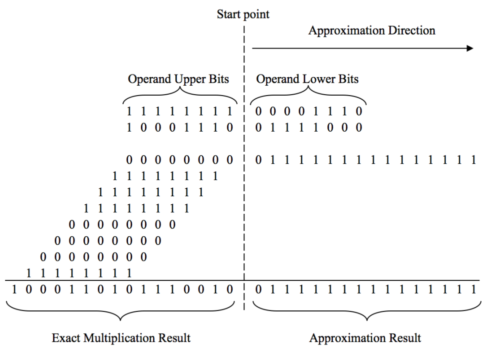

Fig. 1. ETM algorithm

Fig. 1 portrays an illustration for the ETM algorithm [9]. Initially, both the multiplicand $(65295)_{10} = (1111111100001111)_2$ and the multiplier $(36472)_{10} = (1000111001111000)_2$ are segmented into two equal 8-bit segments. When these inputs are fed to the multiplier, the upper parts of both operands are sent to an 8-bit exact multiplier, and the lower parts of both operands are sent to an approximation unit. The 8-bit exact multiplier generates the upper 16-bit of the 32-bit result, and the approximation unit applies an approximation algorithm to generate the lower 16-bit. The adopted algorithm begins at the point where the inputs are segmented into upper and lower parts then it progressively moves to rightmost bits of the input operands. An OR operation is applied to the lower bits of the operands. When the algorithm detects the presence of the first logic '1' output from the OR operation, it begins to approximate the rest of the 16-bit output by setting all bits past the first '1' bit to a logic '1'. In the event where no logic '1' bit is detected when an OR operation is applied to lower bits of both operands, the lower 16 bits of the output are assumed to be logic '0'. If either of the upper parts of both operands is equal to zero, then the lower parts of both operands are sent to the exact multiplier. The starting point for the multiplier is arbitrary. Moving the starting point left reduces the size of the exact multiplier which minimizes hardware but also decreases accuracy. Moving the starting point right increases accuracy at the cost of a larger multiplier [9].

B. Proposed Architecture

Based on the segmentation mechanism of ETM [9], the most significant bits (MSBs) can be evaluated using another approximate algorithm. Correspondingly, this replacement reduces area utilization as well as the power consumption. The upper bits approximate multiplier should be selected carefully to evaluate the MSB of the input operands precisely than the least significant bits (LSBs) of the same operand to limit output inaccuracy. Based on the approximate multipliers review in [10], [11], here are the best candidates to form HETM as follows:

1) Broken Array Multiplier: Mahdini et al. proposed an approximate broken array multiplier (BAM) composed of carry select adder cells (CSA) and an n-bit row of vector merging cells [2]. Depending on the desired reduction in hardware necessary Mahdini proposes the removal of several CSA blocks according to two design parameters, the vertical break line (VBL), and the horizontal break line (HBL). The VBL moves from the right-most CSA block towards the leftmost block. Depending on the position of the line any CSA blocks that fall to the right of it are omitted with their output assumed to be null. Similarly, the HBL moves from top to bottom, and all CSA blocks falling above the HBL are omitted with their output assumed to be null. BAM designs are defined by their VBL and HBL index numbers which define the amount and location of CSA cells to remove.

2) Approximated Partial Product Summation Multipliers: Masadeh et al. proposed several arrays based and tree-based approximate multipliers composed of several approximate full adder cells [11]. The proposed designs are defined by the percentage and type of approximate full adders used. Designs considered in this paper are both array-based, and tree-based multipliers were only the full adders and compressors that contribute to the lowest 50% of the output bits are approximate. The designs considered in this paper are the array based EM4 and compressor based CEM5 that are based on different full adders proposed in [12]. The EM4 multiplier replaces 50% of the full adders in an array multiplier with an AMA4 full adder. The CEM5 multiplier uses approximate compressors based on the AMA5 full adder to evaluate partial production summation for the lower 50% of the partial products.

3) Mitchell-Based Approximate Multipliers: Mitchell Logarithm multiplication converts input operands into approximate logarithms, adds the converted logarithms and then gets the anti-logarithm of the result. This reduces the multiplication of the two operands to simple addition. McLaren proposed a modified Mitchell based logarithm multiplier that uses a look-up table with 64 correction values to improve on Mitchell's original design [5]. McLaren reduced the mean error of the output result from 3% to 0.04% at the cost of increased area utilization due to the addition of the look-up table in the design. While other approaches for improving Mitchells multiplication have been proposed in the literature [13], [7], Mclaren's implementation has been selected for this work due to its superior balance of error and circuit performance.

4) Approximate Multiplier Generation Using Genetic Algorithms: Vojtech et al. developed a library of approximate multipliers generated by a multi-objective Cartesian genetic programming algorithm that attempted to balance error and circuit performance [8]. The library documents each multiplier's error performance metrics in addition to their circuit characteristics. A selection of nine 8bit EVO multipliers with varied error performance metrics was chosen and used in this work from the library provided in [14].

III. Error Performance and Circuit Characteristics

A. Error Performance

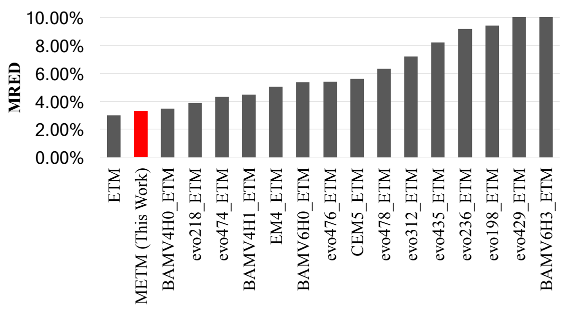

Fig. 2. MRED for different HETMs

Error performance is evaluated using a test-set of 100 million random multiplications. The operands in the test-set were all normally distributed 16-bit numbers. The mean relative error distance (MRED) and normalized mean error distance (NMED) are the primary metrics used to evaluate the approximate multiplier performance [15]. The Error Distance (ED) is defined as the distance between the approximate result of a multiplication $M^{'}$ and the exact result of a multiplication M for each of the evaluated test cases (1). The Relative Error Distance (RED) is the scaling of ED by the exact result M (2). The Mean Error Distance (MED) is the mean of the evaluated ED's for all the test cases (3) where n_t is the number of test cases used. The Mean Relative Error Distance (MRED) is the mean of the evaluated REDs for the given test set (4). The Normalized Mean Error Distance (NMED) is the normalization of MED by the maximum possible output of the design. It is a function of the number of output bits of the multiplier n_i (5). In all of the proposed designs in this paper $n_i = 32$. NMED's purpose is to compare multiplier error performance across different multiplier sizes.

$$ED^{(i)} = |M'^{(i)} - M^{(i)}| \tag{1}$$

$$RED^{(i)} = \frac{ED^{(i)}}{M^{(i)}} \tag{2}$$

$$MED = \frac{\sum_{i=1}^{n_t} ED^{(i)}}{n_t} \tag{3}$$

$$MRED = \frac{\sum_{i=1}^{n_t} RED^{(i)}}{n_t} \tag{4}$$

$$NMED = \frac{MED}{2^{n_i} - 1} \tag{5}$$

According to Fig. 2, the Exact ETM had the lowest MRED and NMED of all the multipliers. BAM error performance progressively declined with the removal of more CSA blocks. The omission of CSA columns has a much greater effect on accuracy in comparison to the omission of CSA rows. BAMV6H3 has the worst error performance in terms of MRED and NMED which is understandable given that it omits 57% of the total available CSA cells. Mitchell's approximate multiplier had a comparable MRED to the original ETM. According to Fig. 3, NMED generally follows MRED for all multipliers. EVO MRED results were similar to those documented by [14] even with the addition of the ETM approximation. However, NMED for the EVO multipliers combined with ETM approximation was worse in comparison to EVO operating as a standalone 8-bit multiplier. CEM5 had a higher MRED than EM4 because the AMA cells used in its compressors have more erroneous test cases and thus a higher probability of error [12].

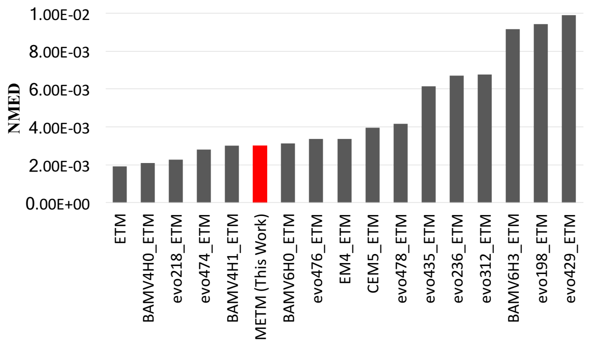

Fig. 3. NMED for different HETMs

B. *Circuit Characteristics*

All HETM are implemented on ZYNQ XCZ 7020-1ClG484 FPGA board. Operand bits in each multiplication have an activity rate of 50%. Power delay product (PDP) and area reduction percentages are evaluated relative to an array based 16-bit exact multiplier implemented on the same FPGA platform. Fig. 4 shows the PDP and area reduction for each of the different multipliers. Area reduction exceeded a factor of 2X for all characterized multipliers which is to be expected given the inclusion of the approximating unit. The original ETM performed poorly in terms of PDP due to the 8-bit x 8-bit exact array multiplier's high delay. The highest reduction in both PDP and area is achieved by the EVO198_ETM, BAMV6H3_ETM, and EVO429_ETM. Significant gains in delay and power reduction can be observed in BAM_ETM based HETMs due to the omission of CSA cells. The movement of the VBL further left reduces delay in the critical path of the circuit by one CSA cell in addition to removing more CSA cells with each step left. EM4 performed well in terms of PDP given the simplicity of the AMA4 full adder cell it is based on. CEM5 while based on an even simpler AMA5 full adder cell performed worse in terms of power consumption due to its internal node structure [16]. All EVO approximate multipliers where generated using a genetic algorithm, therefore no observations based on their structure can be made. However, there is a strong inverse relationship between the superior error performance, PDP, and area reduction for all EVO multipliers. The Mitchell-based error-tolerant multiplier (METM) achieved an area improvement of 2.7X which is comparable to the BAMV6H0_ETM, and a PDP improvement of 1.9X which is also comparable to the BAMV6H3_ETM. The area reduction percentage produced by the METM exceeded the area reduction percentage reported in [5].

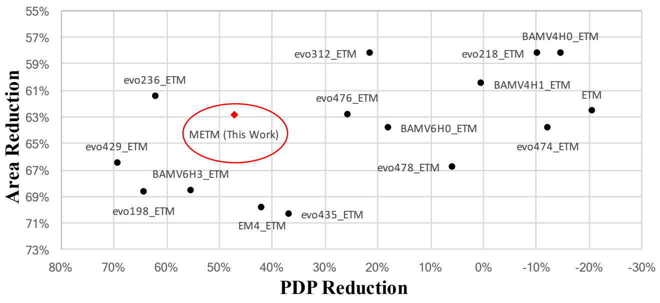

Fig. 4. PDP and Area Reduction Percentages for different multipliers

IV. Image Processing in JPEG Compression

Joint Photographics Experts Group (JPEG) is a lossy compression standard used for images. An integral part of JPEG is DCT [17], which is a mathematical transformation involving matrix multiplications of two 8 x 8 matrices. To perform compression, DCT needs 512 multiplication operations per 8x8 pixel block of any given grayscale image. Approximate multipliers are employed in forward DCT to evaluate their error performance in a real-world application beyond the error metrics discussed in section 4. A standard Lena image is compressed with various HETMs, and the Peak Signal to Noise (PSNR) is evaluated for all compressed images using the uncompressed image as a reference. PSNR is defined as follows:

$$PSNR = 10 \log_{10} \frac{255^2}{MSE} \tag{6}$$

Where 255 is the highest weight of a pixel in a grayscale image.

Fig. 5 shows the original and compressed JPEG images using a selection of different approximate multipliers. The PSNR after compression using an exact 16-bit multiplier, METM, EVO474_ETM, BAMv4H0_ETM, and EVO218_ETM is 32.08 dB, 25.75 dB, 22.89 dB, 19 dB, and 18.07 dB respectively. Fig. 6 shows that the METM was superior in terms of PDP with a PDP improvement factor of 1.9X and an accuracy loss of only 20%, however, it was not the best performer overall in terms of PDP and Area improvement.

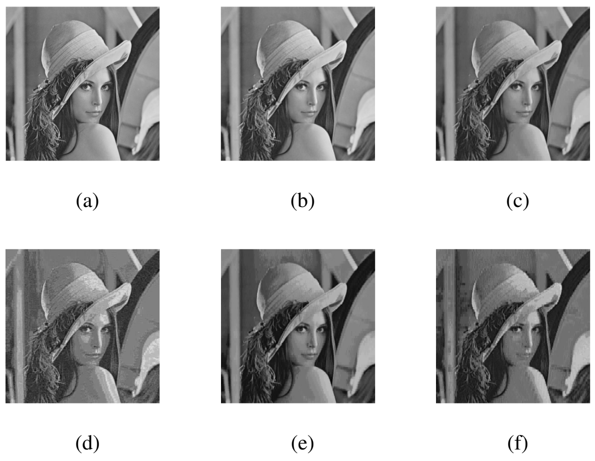

Fig. 5. (a) Original Image, Compressed JPEG images using (b) Exact (c) METM (d) EVO474_ETM (e) BAMV4H0_ETM (f) EVO312_ETM

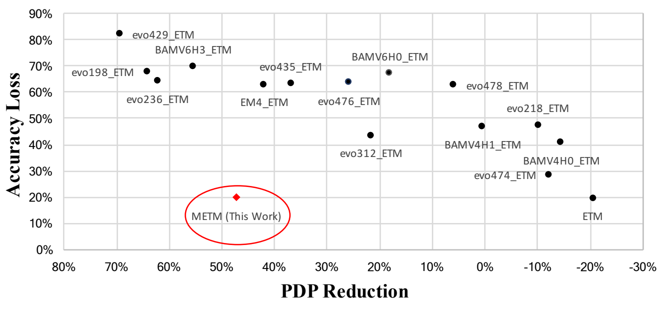

Fig. 6. PDP and Accuracy Loss in JPEG compression for different multipliers

V. Conclusion

In this paper, a low-power compact Mitchell-based error-tolerant multiplier (METM) is discussed. The proposed design is evaluated based on PDP optimization, area reduction, MRED, and NMED. In addition, the accuracy loss in JPEG compression is evaluated using standard Lena image. Finally, the proposed METM has superior performance in comparison with the other designs in terms of PDP and the accuracy loss.

Acknowledgment

This work was partially funded by Mentor Graphics, and ONE Lab at Cairo University and Zewail City of Science and Technology.

References

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

[2] 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 Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 4, pp. 850–862, April 2010.

[3] S. Venkatachalam, H. J. Lee, and S. Ko, "Power efficient approximate booth multiplier," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, Florence, Italy, May 2018, pp. 1–4.

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

[5] D. J. Mclaren, "Improved mitchell-based logarithmic multiplier for low-power dsp applications," in *IEEE International Systems-on-Chip Conference (SOCC)*, Portland, OR, USA, USA, Sept 2003, pp. 53–56.

[6] J. N. Mitchell, "Computer multiplication and division using binary logarithms," *IRE Transactions on Electronic Computers*, vol. EC-11, no. 4, pp. 512–517, Aug 1962.

[7] Z. Babić, A. Avramović, and P. Bulić, "An iterative logarithmic multiplier," *Microprocess. Microsyst.*, vol. 35, no. 1, pp. 23–33, Feb. 2011.

[8] V. Mrazek, R. Hrbacek, Z. Vasicek, and L. Sekanina, "Evoapproxsb: Library of approximate adders and multipliers for circuit design and benchmarking of approximation methods," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, Lausanne, Switzerland, March 2017, pp. 258–261.

[9] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application," in *2010 IEEE International Conference of Electron Devices and Solid-State Circuits (EDSSC)*, Hong Kong, China, Dec 2010, pp. 1–4.

[10] H. Jiang, C. Liu, N. Maheshwari, F. Lombardi, and J. Han, "A comparative evaluation of approximate multipliers," in *2016 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, Beijing, China, July 2016, pp. 191–196.

[11] M. Masadeh, O. Hasan, and S. Tahar, "Comparative study of approximate multipliers," in *Proceedings of the 2018 on Great Lakes Symposium on VLSI (GLSVLSI '18)*. Chicago, IL, USA: ACM, 2018, pp. 415–418.

[12] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 1, pp. 124–137, Jan 2013.

[13] M. S. Kim, A. A. D. Barrio, R. Hermida, and N. Bagherzadeh, "Low-power implementation of mitchell's approximate logarithmic multiplication for convolutional neural networks," in *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jeju, Korea, Jan 2018, pp. 617–622.

[14] "Evoapprox8b l approximate adders and multipliers library." [Online]. Available: http://www.fit.vutbr.cz/research/groups/ehw/approxlib

[15] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Transactions on Computers*, vol. 62, no. 9, pp. 1760–1771, Sept 2013.

[16] M. Masadeh, O. Hasan, and S. Tahar, "Comparative study of approximate multipliers," *CoRR*, vol. abs/1803.06587, 2018. [Online]. Available: http://arxiv.org/abs/1803.06587

[17] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, Jan 1974.