Design and Analysis of Area and Power Efficient Approximate Booth Multipliers

Suganthi Venkatachalam, Elizabeth Adams, *Student Member, IEEE*, Hyuk Jae Lee, *Member, IEEE*, and Seok-Bum Ko, *Senior Member, IEEE*

Abstract—Approximate computing is an emerging technique in which power-efficient circuits are designed with reduced complexity in exchange for some loss in accuracy. Such circuits are suitable for applications in which high accuracy is not a strict requirement. Radix-4 modified Booth encoding is a popular multiplication algorithm which reduces the size of the partial product array by half. In this paper, three Approximate Booth Multiplier Models (ABM-M1, ABM-M2, and ABM-M3) are proposed in which approximate computing is applied to the radix-4 modified Booth algorithm. Each of the three designs features a unique approximation technique that involves both reducing the logic complexity of the Booth partial product generator and modifying the method of partial product accumulation. The proposed approximate multipliers are demonstrated to have better performance than existing approximate Booth multipliers in terms of accuracy and power. Compared to the exact Booth multiplier, ABM-M1 achieves up to a 23 percent reduction in area and 15 percent reduction in power with a Mean Relative Error Distance (MRED) value of 7.9×10^{-4}. ABM-M2 has area and power savings of up to 51 and 46 percent respectively with a MRED of 2.7×10^{-2}. ABM-M3 has area savings of up to 56 percent and power savings of up to 46 percent with a MRED of 3.4×10^{-3}. The proposed designs are compared with the state-of-the-art existing multipliers and are found to outperform them in terms of area and power savings while maintaining high accuracy. The performance of the proposed designs are demonstrated using image transformation, matrix multiplication, and Finite Impulse Response (FIR) filtering applications.

Index Terms—Approximate booth arithmetic, radix-4 partial product generators, inexact computing

1 Introduction

Approximate computing is a concept applied to error-tolerant applications in which the accuracy of an operation is reduced to improve other measures of circuit performance. Approximate computing leverages the innate ability of some applications to tolerate error. Relaxed accuracy requirements are typically acceptable in applications such as digital signal processing, image processing, data mining, and pattern recognition. In these applications, multipliers make a notable impact on power consumption and they stand to benefit from new inexact multiplier designs with high performance. Use of approximate circuits in such applications allow for substantial improvements in performance measures such as power, area, and/or delay [1], [2].

Arithmetic units such as adders and multipliers are extensively used in digital signal processing applications. Approximation schemes for addition are widely discussed in the literature [3], [4], [5]. Approximation in carry-select adders based on speculation with error detection and recovery is proposed in [3]. An error-tolerant adder based on segmentation is analyzed in [4]. In [5], several imprecise adders are designed by reducing the number of transistors and are utilized in digital signal processing applications.

Multiplication is most commonly implemented using either AND-array multipliers or Booth multipliers. For a $n \times n$ multiplication, AND-array multipliers involve the use of AND-gates for partial product generation to produce a partial product matrix with n rows. Booth encoding is introduced in [6] and in [7], Booth multipliers involve recoding the input combination for use in partial product generators to produce signed and plural values of the multiplicand, thereby reducing the number of rows in partial product accumulation matrix. Truncation schemes are a widely-used traditional method of decreasing circuit complexity in fixed-width multipliers in exchange for some loss in accuracy as in [8], [9], [10], [11], where the term fixed-width indicates a multiplier that produces a n-bit output given two n-bit inputs. A post-truncated fixed-width Booth multiplier designed using a compensation vector is discussed in [8]. In [9], quantization error is compensated with approximate carry values. An error compensation circuit composed of simplified sorting networks is proposed in [10]. An adaptive estimator based on conditional probability theory is studied in [11]. In order for fixed-width multipliers to obtain high accuracy, such compensation strategies require additional hardware resources. Approximation provides an alternative method of achieving varying degrees of accuracy in multipliers without compensation circuits.

Approximation in multipliers has been widely discussed in recent years [12], [13], [14], [15], [16], [17], [18], [19], [20]. Many of these works focus on applying approximation to the partial product accumulation stage of the multiplier [13], [14], [15], [16], [17]. Approximate counters and compressors are investigated in [13], [14], where partial product accumulation is performed using approximate counters and compressors rather than exact models. In [13], an inaccurate counter is proposed and used in a Wallace tree structure of a 4×4 multiplier. In [14], two approximate 4-2 compressors are proposed and used in a Dadda tree partial product accumulation. In [15], partial products are altered and approximate arithmetic units are proposed according to the probability of the modified partial products being equal to one. In the partial product perforation multiplier from [16], approximation is achieved by reducing the number of rows in the partial product accumulation circuit in AND-array multipliers and Booth multipliers. In [17], a broken Booth multiplier with vertical breaking levels is introduced, where the elements of partial product matrix to the right of the breaking level are made zero. The authors of these works mostly analyze the effect of applying approximation to multipliers in the partial product accumulation stage. However, Booth multipliers make use of a more complex partial product generation circuit in order to reduce the total number of partial products generated. While substantial work has been performed on approximating partial product accumulation, additional exploration is needed into techniques that apply approximation to partial product generation in Booth multiplication.

There are few existing works investigating approximation in partial product generation [18], [19], [20]. In Booth multipliers, a higher radix corresponds to a decrease in the number of rows of the partial product matrix. For instance, in radix-4 Booth multipliers, partial product generation produces values of 0, ± 1, and $\pm 2 \times$ multiplicand and reduces the size of the partial product matrix by nearly half. Similarly, radix-8 multipliers further reduce the number of rows in partial product matrix where the encoding signals are 0, ± 1, ± 2, ± 3, and ± 4 multiplicand. In [18], the complexity of radix-4 partial product generation is reduced via the modification of truth tables to produce two approximate Booth partial product generators

- *S. Venkatachalam, E. Adams, and S.B. Ko are with the Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, SK S7N 5A9, Canada. E-mail: {suganthi.venkat, lizzie.adams, seokbum.ko}@usask.ca.*
- *H.J. Lee is with the Department of Electrical and Computer Engineering, Seoul National University, Korea. E-mail: hjlee@capp.snu.ac.kr.*

Manuscript received 4 Sept. 2018; revised 16 June 2019; accepted 22 June 2019. Date of publication 1 July 2019; date of current version 16 Oct. 2019.
(Corresponding author: Seok-Bum Ko).
Recommended for acceptance by G. Constantinides.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2019.2926275

TABLE 1
Recoding of Multiplier Bit Groups and Corresponding Operation in Exact Radix-4 Multiplier

b_{2i+1}	b_{2i}	b_{2i-1}	neg_i	two_i	$zero_i$	PP_i
0	0	0	0	0	1	0
0	0	1	0	0	0	$+A$
0	1	0	0	0	0	$+A$
0	1	1	0	1	0	$+2A$
1	0	0	1	1	0	$-2A$
1	0	1	1	0	0	$-1A$
1	1	0	1	0	0	$-1A$
1	1	1	0	0	1	0

each exhibiting $4/32$ and $8/32$ altered truth table entries respectively. In [19], approximation is applied in the generation of partial products for radix-8 Booth multipliers. An approximate 2-bit adder composed of a 3-input XOR-gate is used to generate the $\pm 3\times$ multiplicand term. [20] makes use of a hybrid encoding technique in which exact radix-4 encoding is used to generate the most-significant partial products and approximate higher-radix encoding is used to produce the less-significant bits.

In this work, three approximate Booth multipliers models (ABM-M1, ABM-M2, and ABM-M3) based on radix-4 Booth encoding are proposed. The ABM-M1 multiplier makes use of an approximate Booth partial product generator that replaces $\pm 2\times$ multiplicand terms with $\pm 1\times$ multiplicand terms, producing error in 4 out of 32 cases. The same approximate partial product generator is used in ABM-M2, but the multiplicand input to the generator is consolidated by replacing a set of partial products in every row with a single reduced partial product. ABM-M3 makes use of a second proposed partial product generator that produces a partial product according to the zero-values of a single encoded signal and multiplicand.

This paper is an extension of our conference work [21]. The main improvements and novel contributions of this paper include:

1) Error distance (the absolute difference between actual value and approximate value) of the partial product generator in ABM-M1 multipliers is discussed and analyzed using 16-bit multipliers models.
2) ABM-M2 multipliers are introduced, where partial product generation and accumulation is further simplified based on a consolidated value of the multiplicand and replacing a set of partial product generators with a single partial product generator.
3) A partial product generator based on zero-values of the multiplicand and encoded signal is proposed. The proposed partial product generator is utilized in ABM-M3 multipliers.
4) An approximation factor m is used to implement and analyze the proposed designs with varying degrees of applied approximation. In each design, approximation factor m refers to the number of columns in the partial product matrix to which approximation is applied, in order of increasing significance. As m increases, a higher number of columns make use of the approximate partial product generator, and the inexactness of the multiplier increases. Approximation factors are chosen such that the error metrics of the designs for all models are similar and therefore comparable.

Models 1 and 3 make use of a rectangular replacement scheme in which all partial products with significance less than m are replaced with approximate partial products. Specifically, models 1 and 3 implement approximation factors $m = N/4$, $N/2$, $3N/4$, and N. Model 2 makes use of a diagonal replacement scheme in which approximation factor specifically indicates that, for each row, m exact partial products are compressed into a single approximate partial product. Model 2 implements approximation factors $m = N/8$, $N/4$, $3N/8$, and $N/2$. Smaller approximation factors are used in model 2 because a diagonal replacement scheme is used, meaning that a larger total number of exact partial product generators are replaced with approximate partial product generators than in the rectangular replacement scheme used in models 1 and 3. In all proposed multipliers, the partial product accumulation is performed using a Dadda tree structure composed of exact 4-2 compressors, full-adders, and half-adders. The exact, proposed, and existing approximate multipliers are evaluated with applications including image transformation, matrix multiplication, and Finite Impulse Response filtering.

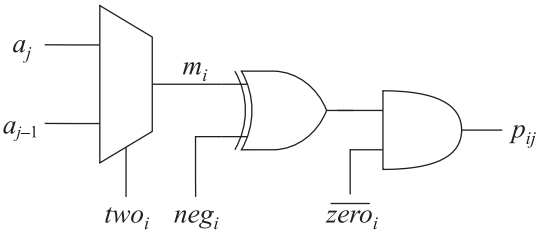

Fig. 1. Circuit schematic for the partial product generator in radix-4 encoding.

The rest of the paper is organized as follows. In Section 2, radix-4 Booth multipliers are discussed. In Section 3, three approximate designs ABM-M1, ABM-M2, and ABM-M3 are presented. In Section 4, design and error metrics of proposed and existing approximate Booth multipliers are compared and analyzed. In Section 5, approximate multipliers are used in practical applications. Section 6 concludes the paper.

2 Radix-4 Booth Multipliers

The output of Booth multiplication can be given as the multiplication of two signed inputs A and B of length N resulting in output P_{out} of length $2N$. The inputs and outputs of the multiplication in two's complement representation can be given as

$$\begin{aligned} A &= -a_{N-1}2^{N-1} + \sum_{n=0}^{N-2} a_n 2^n, \\ B &= -b_{N-1}2^{N-1} + \sum_{n=0}^{N-2} b_n 2^n, \\ P_{out} &= -P_{2N-1}2^{2N-1} + \sum_{n=0}^{2N-2} P_n 2^n. \end{aligned} \tag{1}$$

The input B is grouped in sets of bits $\{b_{2i+1}, b_{2i}, b_{2i-1}\}$ which corresponds to one of the values from 0, ± 1, ± 2. The radix-4 Booth encoder encodes these bit groupings into three signals neg_i, two_i, and $zero_i$ which are used to express the value 0, ± 1, or ± 2 as shown in Table 1. neg_i indicates the sign of each partial product operation, two_i signifies whether the generated partial product is to be shifted, and $zero_i$ specifies whether the partial product is a zero or non-zero value. Based on the signals neg_i, two_i, and $zero_i$, the corresponding row-wise partial product PP_i is selected from the set $\{-2A, -1A, 0, +1A, +2A\}$. For 16-bit signed inputs, eight partial products rows for i ranging from 0 to 7 are generated using radix-4 encoding. Each element in the partial product row is outputted from the partial product generator as shown in Fig. 1. The logic equation of the partial product element p_{ij} produced by the partial product generator is given by

$$\begin{aligned} m_i &= (a_j \cdot \overline{two_i}) + (a_{j-1} \cdot two_i), \\ p_{ij} &= \overline{zero_i} \cdot (neg_i \oplus m_i), \end{aligned} \tag{2}$$

where m_i is the output of the multiplexer and a_j is the multiplicand input row. The partial product matrix for a 16-bit exact Booth multiplier after sign-extension elimination [22] is shown in Fig. 2.

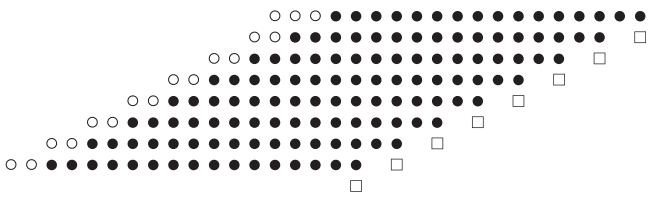

Fig. 2. Partial product matrix of a 16-bit radix-4 Booth multiplier (•: a partial product, ○: a sign-extension term, □: a correction term).

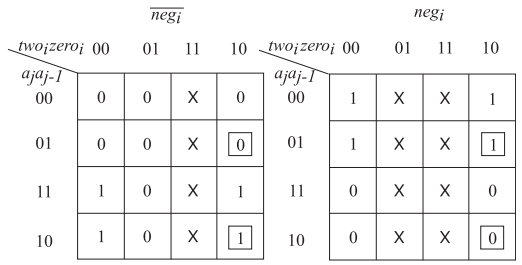

Fig. 3. K-map of approximate partial product generator.

3 Approximation in Booth Multipliers

Booth multipliers are suitable candidates for applying approximation in both partial product accumulation and partial product generation. An exact radix-4 partial product generator requires all three signals neg_i, two_i, and $zero_i$, to generate the partial product. For ABM-M1 and ABM-M2, an approximate partial product generator is designed using only two of the three signals, namely neg_i and two_i. In ABM-M3, a partial product generator is proposed which uses only the signal $zero_i$. In ABM-M1, approximation is applied in partial product accumulation by combining the correction term to its respective row in the partial product matrix and thereby reducing the depth of the matrix. In ABM-M2 and ABM-M3, approximation in generation is achieved by replacing a set of partial product generators with a single approximate partial product generator, thereby reducing the number of elements in accumulation.

In the following sections, 16-bit Booth multipliers with different approximation factors are discussed. ABM-M1 and ABM-M3 employ column wise-approximation and, given an input of length N = 16, designs with approximation factors $m : N/4 = 4$, $N/2 = 8$, $3N/4 = 12$, and N = 16 are implemented. ABM-M2 employs diagonal-wise approximation thereby approximating more elements for a given approximation factor compared to ABM-M1 and ABM-M3. For ABM-M2, designs with $m : N/8 = 2$, $N/4 = 4$, $3N/8 = 6$, and $N/2 = 8$ are implemented.

3.1 ABM-M1 Approximate Multipliers

The K-map corresponding to the partial product generator circuit in Fig. 1 is approximated by modifying 4 of 32 entries as shown in Fig. 3, where 1 represents a change from '0' to '1' and 0 represents a change from '1' to '0'. This results in an approximate partial product generator based on two signals, neg_i and $zero_i$, subsequently referred to as PPG-2S.

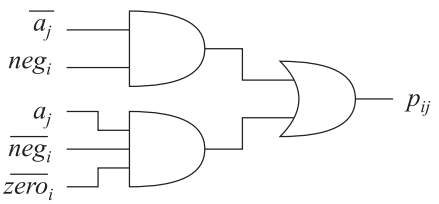

Fig. 4. Circuit schematic for approximate two-signal partial product generator PPG-2S.

The circuit schematic for this approximate partial product generator is shown in Fig. 4 and can be given as

$$p_{ij} = \overline{a_j} \cdot neg_i + a_j \cdot \overline{neg_i} \cdot \overline{zero_i}\,. \tag{3}$$

When compared to the exact partial product generator, the PPG-2S circuit does not require a multiplexer nor an XOR-gate and the output can be expressed in terms of AND and OR-gates. The error distance between the exact partial product generator and PPG-2S is given in Table 2. Since the two_i signal is absent from PPG-2S, the $+2A$ and $-2A$ cases are replaced with $+1A$ and $-1A$, respectively, which results in two cases with an error distance of ± 1. Although error distance of approximate values deviates from actual values by 50 percent, it happens only in 4 out of 32 cases resulting in small error. Errors complement each other and designing an error compensation takes more resourses and negates the performance benefits of approximation.

The proposed approximate partial product generator PPG-2S is utilized in the ABM-M1 multipliers. The partial product matrix for a 16-bit ABM-M1 multiplier with m = 12 is shown in Fig. 5. In the ABM-M1 design, all partial products with a significance less than m are generated using the approximate PPG-2S circuit and all remaining partial products are generated using the exact partial product generator. To reduce the height of the matrix, each correction term is combined with a partial product in its respective column using an OR-gate, as shown in Fig. 5.

3.2 ABM-M2 Approximate Multipliers

The ABM-M2 multipliers differ from ABM-M1 such that, in every row, the m least-significant exact partial product generators are replaced with a single PPG-2S. The least-significant m bits of input A are summed to produce a_{sum}. Based on a_{sum}, a value $a_{\forall j \in (0,m-1)}$ is found as given in Equation (4). The single approximate partial product $p_{i,\forall j \in (0,m-1)}$ for each row is generated using PPG-2S where $a_{\forall j \in (0,m-1)}$ is inputted as the a_j signal.

TABLE 2
Error Distance of Proposed Approximate Partial Product Generator Based on Two Signals

neg_i	two_i	$zero_i$	Exact pp_{ij} for $a_j a_{j-1}$				Exact OP	Approx. pp_{ij} for $a_j a_{j-1}$				Approx. OP	Error
			00	01	10	11		00	01	10	11		
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	$+1A$	0	0	1	1	$+1A$	0
0	0	0	0	0	1	1	$+1A$	0	0	1	1	$+1A$	0
0	1	0	0	1	0	1	$+2A$	0	0	1	1	$+1A$	1
1	1	0	1	0	1	0	$-2A$	1	1	0	0	$-1A$	-1
1	0	0	1	1	0	0	$-1A$	1	1	0	0	$-1A$	0
1	0	0	1	1	0	0	$-1A$	1	1	0	0	$-1A$	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0

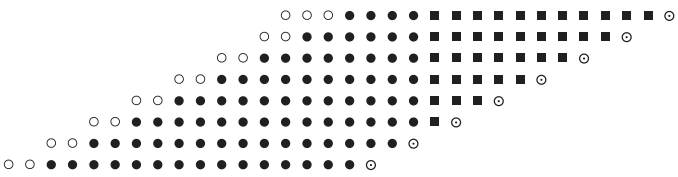

Fig. 5. Partial product matrix of a 16-bit ABM-M1 multiplier with m = 12 (•: a partial product, ○: a sign-extension term, ■: an approximate partial product generated with PPG-2S, ⊙: term resulting from OR-ing the least-significant bit of a partial product with its correction term).

$$a_{sum} = \sum_{j=0}^{m-1} a_j \,, \qquad a_{\forall j \in (0,m-1)} = \begin{cases} 1, & a_{sum} > m/2 \\ 0, & a_{sum} \leq m/2 \,. \end{cases} \tag{4}$$

ABM-M2 for approximation factor m = 8 is illustrated in Fig. 6, which depicts the transformation of the exact partial product matrix to a reduced approximate partial product matrix. For m = 8, the least-significant 8 bits of input A are summed to produce a_{sum}. From a_{sum}, a value $a_{\forall j \in (0,7)}$ is found as per Equation (4), which is then used to generate for each row i the approximate partial product $p_{i,\forall j \in (0,7)}$ using the signals neg_i, $zero_i$, and $a_{\forall j \in (0,7)}$ as inputs to PPG-S2.

3.3 ABM-M3 Approximate Multipliers

The ABM-M3 multiplier utilizes an even further simplified approximate partial product generator PPG-1S which takes into account only the zero-values of the multiplicand A and signal $zero_i$. For approximation factor m, all partial products with significance less than m are reduced to a single approximate partial product. Considering the exact partial product matrix in Fig. 2, for a row i, let l be the number of bits with a significance less than m. For a row i, $a_{\forall j \in (0,m-(2i+1))}$ is generated by OR-ing the l least-significant bits of A. The approximate partial product for the row i is then generated by the use of PPG-1S as shown in Fig. 7. PPG-1S takes in the result of the OR operation as well as the signal $zero_i$ to produce the approximate partial product for that row.

An example with m = 12 is given in Fig. 8. For the fourth row with $i = 3$, a ranging from 0 to $m - 7$ are given to the $m - 6$ input OR-gate. The output of this OR-gate $a_{\forall j \in (0,m-7)}$ and the signal $\overline{zero_3}$ are inputted to the AND-gate. Based on these values, the partial product $p_{i,\forall j \in (0,m-7)}$ is found. Similarly, for the first row, input a from 0 to $m - 1$ and $zero_0$ are considered. In the second row, input a from 0 to $m - 3$, and $zero_1$ are considered. In the third row, input a from 0 to $m - 5$ and $zero_2$ are considered, and so on.

4 Result Analysis and Discussion

The proposed approximate multipliers (ABM-M1, ABM-M2, and ABM-M3) and existing approximate multipliers are implemented in Verilog HDL. A SystemVerilog testbench and a Python script were used to compare the output of the proposed multipliers with the result of the exact multiplication operation for one million uniformly-distributed random pairs of 16-bit signed inputs ranging from −32768 to +32767. The multipliers are then synthesized with the Synopsys Design Compiler using the TSMC 65 nm library. The Design Compiler simulations use an operating voltage of 1 V and an operating temperature of 25 °C. The largest critical path delay of all designs, i.e., exact radix-8 multiplier was measured to be 1.2 ns and that delay was then used as the timing constraint for all other simulations.

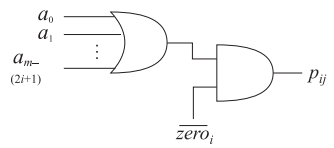

Fig. 7. Circuit schematic for approximate single-signal partial product generator PPG-1S.

The error characteristics of the proposed multipliers are reported in Table 3, along with the error metrics of other contemporary multipliers used as comparison designs. Mean relative error distance (MRED) and normalized mean error distance (NMED) as defined in [23] were calculated for all multipliers. Error distance is the difference between the exact value and approximate value, whereas relative error is the value of error distance divided by the exact value. The MRED metric is used to evaluate the average impact of the approximation on accuracy and is calculated as the mean relative error for all possible values. The NMED metric involves the normalization of the average error distance to the maximum possible output value, and therefore is effective in expressing the impact on accuracy irrespective of the model scale. Table 4 reports the area, power, and area-power product (APP) metrics of the proposed multipliers and comparison models as well as the percentage reduction in APP compared to the exact radix-8 multiplier.

Comparison multipliers include the ABM1 and ABM2-C9 designs proposed by Jian et al. in [19], R4ABM multiplier proposed by Liu et al. in [18] and RAD256 of [20]. ABM1 of [19] utilizes an approximate recoding adder with 8 approximated bits with error recovery. ABM2-C9 of [19] employs truncation in addition to an approximate recoding adder with 8 approximated bits with error compensation. In the R4ABM design of [18], the multiplier bits are grouped into sets of three bits and are used in exact and approximate partial product generation. The exact partial product generator used in [18] takes as inputs the grouped input bits a_j, a_{j-1}, b_{2i+1}, b_{2i}, and b_{2i-1}, rather than using recoded signals such as neg_i, two_i, and $zero_i$ as done in this paper. In [18], to reduce the logic complexity of the exact partial product generator, $\pm 2\times$ multiplicand values are replaced by zero. The RAD256 model from [20] uses a hybrid of exact radix-4 and approximate radix-256 Booth encoding.

All three proposed designs achieve significant area and power savings over the exact radix-8 multiplier. The proposed ABM-M1 designs allow for APP savings in the range of 13 to 35 percent, and the ABM-M3 designs provide APP savings in the range of 9 to 76 percent. ABM-M2 exhibits the most substantial APP reduction, with m values of 2, 4, 6 and 8 corresponding to APP savings of 35, 48, 65, and 74 percent, respectively.

ABM-M1 with m = 4,8,12 and 16 have better error characteristics and area-power product compared to R4ABM [18] with m = 4, 8, 12

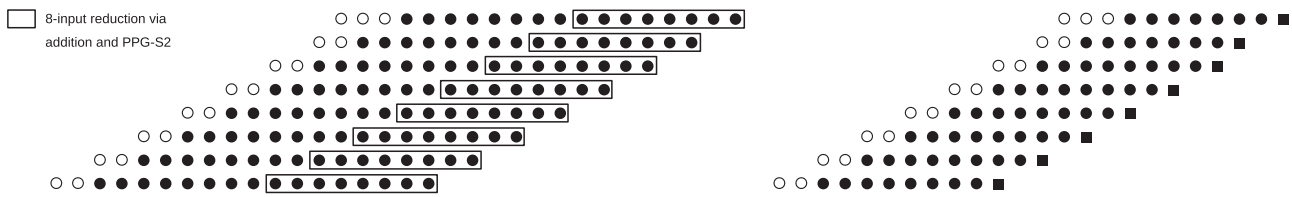

Fig. 6. Partial product matrix of a 16-bit ABM-M2 multiplier with m = 8. The width of the matrix is reduced by adding the m least significant bits of each partial product row, comparing the result to m, and then using the resulting 1-bit or 0-bit as an input to PPG-2S (•: a partial product, ○: a sign-extension term, ■: approximate partial product generated using PPG-2S).

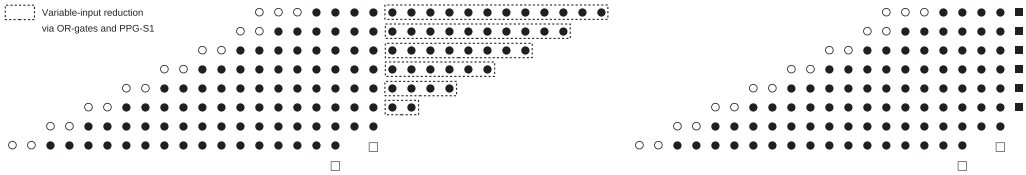

Fig. 8. Partial product matrix of a 16-bit ABM-M3 multiplier with m = 12. The width of the matrix is reduced by OR-ing together all bits with a significance less than m and then using the result of the OR operation as an input into PPG-1S (●: a partial product, ○: a sign-extension term, □: a correction bit, ■: approximate partial product generated using PPG-1S).

and 16 respectively. ABM-M1 with m = 8 and 12 have better area-power product, MRED, and NMED than ABM1. ABM-M2 with m = 2 performs better than ABM1 [19] in terms of both area-power product and NMED while having the same MRED. ABM-M2 with m = 4, 6 and 8 shows drastic reduction in area and power, with increased error values as the approximation factor increases.

ABM-M3 has excellent error characteristics for m = 4 and 8, with slight improvement in area-power product. The reason behind the high accuracy is that the correction terms are not approximated to reduce the depth of the matrix, whereas in ABM-M1, correction terms are ORed with a element in their corresponding column to reduce the depth. ABM-M3 with m = 12 has better error and area power product performance when compared to ABM1 and ABM2-C9. ABM-M3 with m = 16 has better NMED and area-power product performance when compared to RAD256, while RAD256 [20] has comparatively better MRED. Overall, the proposed multipliers show good improvement in accuracy and hardware performance over the existing approximate Booth multipliers. They also achieve high performance with respect to exact Booth multipliers.

5 Application Level Case-Studies

In this section, the proposed and existing approximate multipliers are tested using practical and relevant applications. Three popular applications related to signal processing are chosen—image transformation, matrix multiplication, and FIR filter. For ABM-M1 and ABM-M3, designs with $m : N/2 = 8$ and $N = 16$ are tested. For ABM-M2, designs with $m : N/4 = 4$ and $N/2 = 8$ are tested. MATLAB is used to collect the input data for signal processing applications which is then inputted to exact and approximate multipliers simulated in the ModelSim environment. The multiplication results produced by the simulations are collected for processing by MATLAB which is used to produce results and assess the quality of difference between the designs.

TABLE 3
MRED and NMED Values of Proposed and Existing Approximate Multipliers

Design	m	MRED (10^{-2})	NMED (10^{-6})
ABM-M1	4	0.011	5.079
	8	0.012	5.089
	12	0.016	5.491
	16	0.079	20.800
ABM-M2	2	0.040	15.250
	4	0.165	64.626
	6	0.663	266.100
	8	2.689	1087.270
ABM-M3	4	0.00007	0.005
	8	0.001	0.106
	12	0.020	2.002
	16	0.340	36.150
R4ABM [18]	4	0.012	5.714
	8	0.013	5.835
	12	0.038	8.430
	16	0.473	64.190
ABM1 [19]	–	0.040	19.360
ABM2-C9 [19]	–	0.089	44.500
RAD256 [20]	–	0.277	167.800

5.1 Image Transformation

The multipliers are additionally applied to image transformation. The pixel values of a 16-bit image are uniformly shifted from the range $[0, 65535]$ as to occupy the range $[-32768, 32767]$. The shifted image is then multiplied by a constant on a pixel-by-pixel basis to perform image brightening. The results of image transformation using exact Booth multipliers, comparison approximate Booth multipliers, and the proposed multipliers are processed using MATLAB. Peak signal-to-noise ratio (PSNR) is used to measure and compare the output image quality for various models.

The images produced by the multipliers and their corresponding PSNR values are shown in Fig. 9. For all models, the image quality deteriorates with increasing approximation factor. The results from this application show that ABM-M3 with approximation factors $m = 8, 16$ outperform R4ABM with equivalent approximation factors, ABM1, and ABM2-C9. Similarly, ABM-M2 exhibits better PSNR values for approximation factors m = 4. ABM-M1 with $m = 8, 16$, ABM-M3 with $m = 8, 16$, and ABM-M2 with $m = 4$

TABLE 4
Area, Power, and Area-Power Product Values of Proposed and Existing Approximate Multipliers under Uniform Delay 1.2 ns

Design	m	Area (μm^2)	Power (mW)	APP (μm$^2\cdot$ ns)
Exact radix-4	–	3851	1.3	5006
Exact radix-8	–	4116	1.3	5351
ABM-M1	4	3578	1.3	4651 (–13%)
	8	3419	1.2	4103 (–23%)
	12	3218	1.2	3862 (–28%)
	16	3160	1.1	3476 (–35%)
ABM-M2	2	3148	1.1	3463 (–35%)
	4	2797	1.0	2797 (–48%)
	6	2333	0.8	1866 (–65%)
	8	2015	0.7	1411 (–74%)
ABM-M3	4	3747	1.3	4871 (–9%)
	8	3563	1.3	4632 (–13%)
	12	2669	1.0	2669 (–50%)
	16	1830	0.7	1281 (–76%)
R4ABM [18]	4	4037	1.3	5248 (–2%)
	8	4008	1.3	5210 (–3%)
	12	3640	1.2	4368 (–18%)
	16	3362	1.1	3698 (–31%)
ABM1 [19]	–	3589	1.2	4307 (–20%)
ABM2-C9 [19]	–	2820	1.0	2820 (–47%)
RAD256 [20]	–	1977	0.7	1384 (–74%)

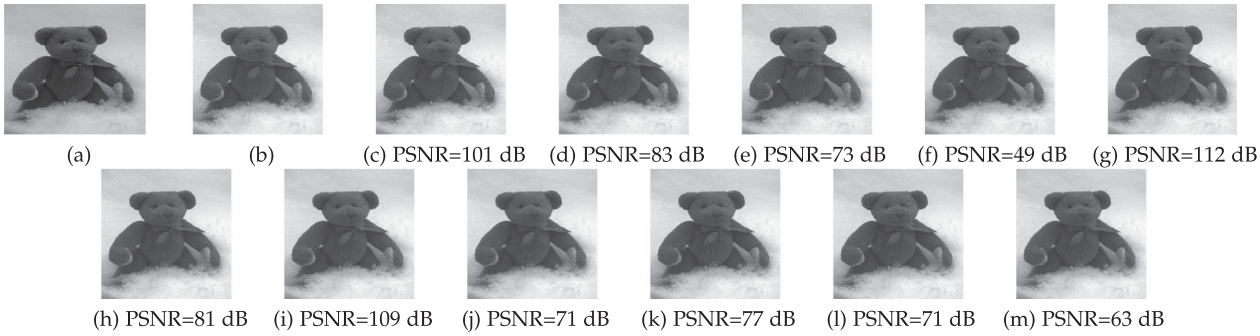

Fig. 9. (a) Input image. Images after image multiplication using (b) exact multiplier, the approximate multipliers (c) ABM-M1 ($m = 8$) (d) ABM-M1 ($m = 16$) (e) ABM-M2 ($m = 4$) (f) ABM-M2 ($m = 8$) (g) ABM-M3 ($m = 8$) (h) ABM-M3 ($m = 16$) (i) R4ABM [18] ($m = 8$) (j) R4ABM [18] ($m = 16$) (k) ABM1 [19] (l) ABM-C9 [19] (m) RAD256 [20].

are suitable candidates for image processing applications which require a negligible loss in image quality.

5.2 Matrix Multiplication

Matrix multiplication is one of the most critical kernel operations in applications such as convolution, deep learning, multimedia processing, computer graphics, and robotics. In this work, 5×5 matrices are chosen for matrix multiplication. MRED and NMED are used as error metrics to assess the quality of output for various multipliers. Error values are based on the absolute difference between the matrix multiplication results of the approximate multiplier and the exact multiplier. The MRED and NMED values are listed in Table 5. It can be seen that the proposed multipliers perform better than existing approximate Booth multiplier designs. The findings are consistent with MRED and NMED values discussed in Table 3.

TABLE 5
MRED and NMED Values of Proposed and Existing Approximate Multipliers Used in Matrix Multiplication Application

Approximate Design (16-bit)	m	MRED (10^{-3})	NMED (10^{-5})
ABM-M1	8	0.308	0.51
	16	0.337	1.006
ABM-M2	4	3.274	6.074
	8	51.774	97.726
ABM-M3	8	0.004	0.005
	16	1.186	1.747
R4ABM [18]	8	0.382	0.577
	16	4.233	6.499
ABM1 [19]	–	1.858	1.617
ABM2-C9 [19]	–	2.447	3.341
RAD256 [20]	–	3.140	8.531

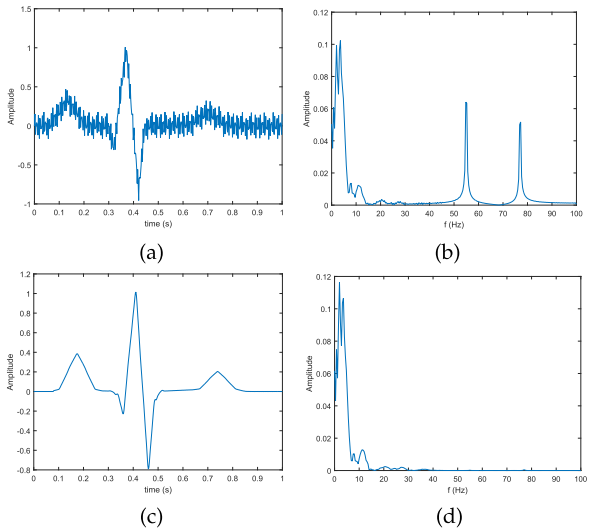

Fig. 10. (a) and (b) Input ECG with its frequency spectrum (c) and (d) Time and frequency plot of ECG after low pass FIR filter.

5.3 FIR Filter

Filtering is a type of signal processing, widely utilized to remove unwanted noise from a signal. Finite impulse response filters are used to generate a desired frequency response. A low pass filter attenuates signals above the cut-off frequency allowing contents less than the cut-off frequency. In this application, we have chosen a electrocardiograph (ECG) wave with high-frequency noise to pass through a 41-tap low-pass FIR filter with a cut-off frequency of 45 Hz and sampling frequency of 200 Hz. The input and filtered ECG signal and its frequency components are given in Fig. 10. FIR filtering makes use of multiplication and addition operations.

To evaluate the approximate multipliers, exact multipliers are substituted with the approximate designs. Mean square error (MSE) is the metric taken to measure filter performance based on the difference between the low-pass filtering results of the approximate multipliers and the exact multiplier. The MSE values of low-pass filter multiplications are listed in Table 6. ABM-M3 performs better than R4ABM with equivalent approximation factors, ABM2-C9 and RAD256. ABM-M1 of $m = 16$ has better performance than R4ABM of the same approximation factor, ABM1, ABM2-C9 and RAD256. ABM-M2 with $m = 4$ performs better than R4ABM with $m = 16$ and RAD256. ABM-M3 with $m = 8$ has best MSE of all designs.

TABLE 6
MSE Values of Proposed and Existing Approximate Multipliers Used in FIR Low Pass Filter Application

Approximate Design (16-bit)	m	MSE
ABM-M1	8	2.546×10^{-9}
	16	6.328×10^{-8}
ABM-M2	4	1.044×10^{-6}
	8	2.673×10^{-4}
ABM-M3	8	3.570×10^{-12}
	16	2.391×10^{-7}
R4ABM [18]	8	7.747×10^{-11}
	16	1.819×10^{-5}
ABM1 [19]	–	7.291×10^{-8}
ABM2-C9 [19]	–	2.897×10^{-7}
RAD256 [20]	–	1.919×10^{-6}

6 Conclusion

Three models of approximate Booth multipliers are proposed, in which approximation is focussed on partial product generation. A partial product generator that makes use of only two signals is used in ABM-M1 and ABM-M2, and the partial product generator in ABM-M3 is further reduced to use only one recoded signal. An approximation factor m is used to indicate the imprecision of each model.

Area-power product and error metrics of the proposed multipliers are compared with exact multipliers and existing state-of-the-art approximate Booth multipliers. When compared to the exact Booth multiplier, ABM-M1 exhibits APP savings of up to 35 percent with a corresponding MRED value of 7.9×10^{-4}. Similarly, ABM-M2 reduces APP by up to 74 percent corresponding to an MRED value of 2.7×10^{-2}, and ABM-M3 provides APP savings of up to 76 percent with an MRED value of 3.4×10^{-3}. When compared with state-of-the-art approximate Booth multipliers, the proposed Booth multipliers exhibit better accuracy and outperform the existing designs in terms of area-power product. Furthermore, the proposed multipliers are tested in practical applications such as image transformation, matrix multiplication, and FIR filtering and are found to have better performance than existing works.

Increasing approximation factor for ABM-M1 results in gradually-reduced area-power product with relatively little deterioration in accuracy, making it a good candidate for use in applications that cannot tolerate significant error. ABM-M2 multipliers provide excellent hardware performance with increasing approximation factor. ABM-M2 exhibits the largest error among the proposed models but also the most significant performance improvement, making it a good fit for applications that can tolerate higher error rates.

ABM-M3 multipliers have excellent accuracy when approximation factor is low, but accuracy drops below that of ABM-M1 as approximation factor increases and area-power delay improves. The nature of a given application and its specific requirements will determine which of the proposed multiplier models is best suited to the task. However, the three models together have a good variety in terms of accuracy-performance tradeoff and therefore should provide useful results in a variety of applications.

In the future, the proposed approximate multipliers and their accuracy distribution impact are to be studied and analyzed in deep learning applications which exhibit good error-esiliency.

Acknowledgments

This work was supported by the NSERC, R&D program of MOTIE/KEIT [10077609, Developing Processor-Memory-Storage Integrated Architecture for Low Power, High Performance Big Data Servers], and University of Saskatchewan. This paper is an extension of work originally presented in 2018 ISCAS [21].

References

[1] J. Han and M. Orshansky, "Approximate Computing: An Emerging Paradigm for Energy-Efficient Design," *Proc. 18th IEEE Eur. Test Symp.*, 2013.
[2] S. Venkataramani, S. T. Chakradhar, K. Roy and A. Raghunathan, "Computing approximately, and efficiently," *Des. Autom. Test Eur. Conf. Exhib.*, 2015, pp. 748–751.
[3] K. Du, P. Varman and K. Mohanram, "High performance reliable variable latency carry select addition," *Des. Autom. Test Eur. Conf. Exhib.*, 2012, pp. 1257–1262.
[4] Ning Zhu, W. L. Goh, and K. S. Yeo, "An enhanced low-power high-speed adder for error-tolerant application," in *Proc. 12th Int. Symp. IC*, 2009, pp. 69–72.
[5] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 32, no. 1, pp. 124–137, Jan. 2013.
[6] A. D. Booth, "Signed binary multiplication technique," *Quart. J. Mech. Appl. Math.*, vol. 4, pp. 236–240, Jan. 1951.
[7] O. L. Macsorley, "High-Speed arithmetic in binary computers," in *Proc. IRE*, vol. 49, no. 1, pp. 67–91, Jan. 1961.
[8] S. J. Jou, M. Tsai, and Y. Tsao, "Low-error reduced-width Booth multipliers for DSP applications," *IEEE Trans. Circuits Syst.*, vol. 50, no. 11, pp. 1470–1474, Nov. 2013.
[9] K. J. Cho, K. C. Lee, J. G. Chung and K. K. Parhi, "Design of low-error fixed-width modified Booth multiplier," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 12, no. 5, pp. 522–531, May 2004.
[10] J. Wang, S. Kuang and S. Liang, "High-accuracy fixed-width modified booth multipliers for lossy applications," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 19, no. 1, pp. 52–60, Jan. 2011.
[11] Y. Chen and T. Chang, "A high-accuracy adaptive conditional-probability estimator for fixed-width booth multipliers," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 59, no. 3, pp. 594–603, Mar. 2012.
[12] P. Kulkarni, P. Gupta, and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture," *J. Low Power Electron.*, vol. 7, no. 4, pp. 490–501, 2011.
[13] C. H. Lin and C. Lin, "High accuracy approximate multiplier with error correction," in *Proc. IEEE 31st Int. Conf. Comput. Des.*, 2013, pp. 33–38.
[14] 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.
[15] S. Venkatachalam and S. B. Ko, "Design of power and area efficient approximate multipliers," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 5, pp. 1782–1786, May 2017.
[16] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 24, no. 10, pp. 3105–3117, Oct. 2016.
[17] F. Farshchi, M. S. Abrishami and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in *Proc. 17th CSI Int. Symp. Comput. Archit. Digit. Syst.*, 2013, pp. 25–30.
[18] 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.
[19] 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.
[20] V. Leon, G. Zervakis, D. Soudris and K. Pekmestzi, "Approximate hybrid high radix encoding for energy-efficient inexact multipliers," in *Proc. IEEE Trans. Very Large Scale Integr. Syst.*, vol. 26, no. 3, pp. 421–430, Mar. 2018.
[21] S. Venkatachalam, H. J. Lee and S. B. Ko, "Power efficient approximate booth multiplier," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2018, pp. 1–4.
[22] N. Burgess, "Removal of sign-extension circuitry from Booth's algorithm multiplier-accumulators," *Electron. Lett.*, vol. 26, no. 17, pp. 1413–1415, Aug. 1990.
[23] 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.