International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 7, ISSN No. 2455-2143, Pages 365-370
Published Online November 2019 in IJEAST (http://www.ijeast.com)

DESIGN AND IMPLEMENTATION OF ROBA MULTIPLIER IN DSP SYSTEMS

P. Ramya,
PG student, Master of Technology (VLSI),
Dept. Of Electronics and Communication Engineering
Audisankara College Of Engineering And Technology (Autonomous), Gudur, A.P

K. Dhanunjaya
Professor
Dept. Of Electronics and Communication Engineering
Audisankara College Of Engineering And Technology (Autonomous), Gudur, A.P

ABSTRACT - In this paper, we propose an approximate multiplier that is high speed yet energy efficient. The approach is to round the operands to the nearest exponent of two. This way the computational intensive part of the multiplication is omitted improving speed and energy consumption at the price of a s mall error. The proposed approach is applicable to both signed and unsigned multiplications. We propose three hardware implementations of the approximate multiplier that includes one for the unsigned and two for the signed operations. The efficiency of the proposed multiplier is evaluated by comparing its performance with those of some approximate and accurate multipliers using different design parameters. In addition, the efficacy of the proposed approximate multiplier is studied in two image processing applications, i.e., image sharpening and smoothing.

Key Words: ROBA Multiplier, DSP system, Image processing.

I. INTRODUCTION

Multipliers are one of the most significant blocks in computer arithmetic and are generally used in different digital signal processors. There is growing demands for high speed multipliers in different applications of computing systems, such as computer graphics, scientific calculation, and image processing and so on. Speed of multiplier determines how fast the processors will run and designers are now more focused on high speed with low power consumption. The multiplier architecture consists of a partial product generation stage, partial product reduction stage and the final addition stage. The partial product reduction stage is responsible for a significant portion of the total multiplication delay, power and area. Therefore, in order to accumulate partial products, compressors usually implement this stage because they contribute to the reduction of the partial products and also contribute to reduce the critical path which is important to maintain the circuit's performance.

This is accomplished by the use of 3-2, 4- 2, 5-2 compressor structures. A 3-2 compressor circuit is also known as full adder cell. As these compressors are used repeatedly in larger systems, improved design will contribute a lot towards overall system performance. The internal structure of compressors is basically composed of XOR- XNOR gates and multiple xers. The XOR-XNOR circuits are also building blocks in various circuits like arithmetic circuits, multipliers, compressors, parity checkers, etc. Optimized design of these XOR-XNOR gates can improve the performance of multiplier circuit. In present work, a new XOR- XNOR module has been proposed and 4-2 compressor has been implemented using this module. Use proposed circuit in partial product accumulation reduces transistor count as well as power consumption.

Addition and multiplication are widely used operations in computer arithmetic; for addition full-adder cells have been extensively analyzed for approximate computing Liang et al. has compared these adders and proposed several new metrics for evaluating approximate and probabilistic adders with respect to unified figures of merit for design assessment for inexact computing applications. For each input to a circuit, the error distance (ED) is defined as the arithmetic distance between an erroneous output and the correct one.

However, the design of approximate multipliers has received less attention. Multiplication can be thought as the repeated s um of partial products; however, the straightforward application of approximate adders when designing an approximate multiplier is not viable, because it would be very inefficient in terms of precision, hardware complexity and other performance metrics. Several

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 7, ISSN No. 2455-2143, Pages 365-370
Published Online November 2019 in IJEAST (http://www.ijeast.com)

approximate multipliers have been proposed. Most of these designs use a truncated multiplication method; they estimate the least significant columns of the partial products as a constant. An imprecise array multiplier is used for neural network applications by omitting some of the least significant bits in the partial products (and thus removing some adders in the array). A truncated multiplier with a correction constant is proposed.

II. LITERATURE SURVEY

In this section, some of the previous works in the field of approximate multipliers are briefly reviewed. In, an approximate multiplier and an approximate adder based on a technique named broken-array multiplier (BAM) were proposed. By applying the BAM approximation method of to the conventional modified Booth multiplier, an approximate signed Booth multiplier was presented in. The approximate multiplier provided power consumption savings from 28% to 58.6% and area reductions from 19.7% to 41.8% for different word lengths in comparison with a regular Booth multiplier. Kulkarni et al. suggested an approximate multiplier consisting of a number of 2 × 2 inaccurate building blocks that saved the power by 31.8%–45.4% over an accurate multiplier. An approximate signed 32-bit multiplier for speculation purposes in pipelined processors was designed in. It was 20% faster than a full-adder-based tree multiplier while having a probability of error of around 14%.

In, an error-tolerant multiplier, which computed the approximate result by dividing the multiplication into one accurate and one approximate part, was introduced, in which the accuracies for different bit widths were reported. In the case of a 12-bit multiplier, a power saving of more than 50% was reported. The use of approximate multipliers in image processing applications, which leads to reductions in power consumption, delay, and transistor count compared with those of an exact multiplier design, has been discussed in the literature. An accuracy - configurable multiplier architecture (ACMA) was suggested for error-resilient systems. To increase its throughput, the ACMA made use of a technique called carry-in prediction that worked based on a pre-computation logic. When compared with the exact one, the proposed approximate multiplication resulted in nearly 50% reduction in the latency by reducing the critical path. Also, Bhardwaj et al. presented an approximate Wallace tree multiplier (AWTM). Again, it invoked the carry-in prediction to reduce the critical path. In this work, AWTM was used in a real-time benchmark image application showing about 40% and 30% reductions in the power and area, respectively, without any image quality loss compared with the case of using an accurate Wallace tree multiplier (WTM) structure.

III. EXISTING SYSTEM

A. VEDIC MULTIPLIER

The hardware architecture of 2X2, 4x4 and 8x8 bit Vedic multiplier module are displayed in the below sections. Here, "Urdhva- Tiryagbhyam" (Vertically and Crosswise) sutra is used to propose such architecture for the multiplication of two binary numbers. The beauty of Vedic multiplier is that here partial product generation and additions are done concurrently. Hence, it is well adapted to parallel processing. The feature makes it more attractive for binary multiplications. This in turn reduces delay, which is the primary motivation behind this work.

B. VEDIC MULTIPLIER FOR 8x8 BIT MODULE

The 8x8 bit Vedic multiplier module as shown in the block diagram in Fig. 6 can be easily implemented by using four 4x4 bit Vedic multiplier modules as discussed in the previous section

Lets analyze 8x8 multiplications, say A= A_7 A_6 A_5 A_4A_3 A_2 A_1 A_0 and B= B_7 B_6 $B5B_4$ B_3 B_2 B_1B_0.

The output line for the multiplication result will be of 16 bits as – S_{15} S_{14} $S_{13}S_{12}$ S_{11} S_{10} S_9 S_8 S_7 $S_{6b}S_5S_4$ S_3 S_2 S_1 S_0. Let's divide A and B into two parts, say the 8 bit multiplicand A can be decomposed into pair of 4 bits AH-AL. Similarly, multiplicand B can be decomposed into BH-BL.

The 16 bit product can be written as:

Using the fundamental of Vedic multiplication, taking four bits at a time and using 4 bit multiplier block as discussed we can perform the multiplication. The outputs of 4x4 bit multipliers are added accordingly to obtain the final product. Here total three 8 bit Ripple -Carry Adders are required as shown in figure 1

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 7, ISSN No. 2455-2143, Pages 365-370
Published Online November 2019 in IJEAST (http://www.ijeast.com)

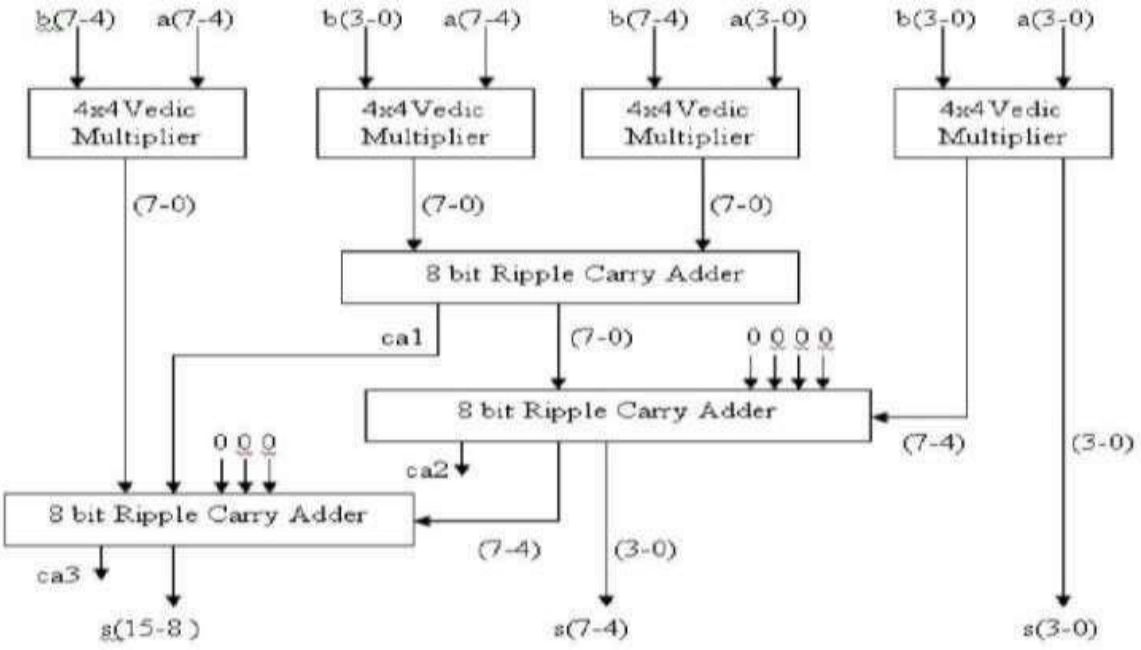

Fig. 3.3.1 Block Diagram of 8x8 bit Vedic Multiplier

we can generalize the method as discussed in the previous sections for any number of bits in input. Let, the multiplication of two N-bit binary numbers (where N = 1, 2,3…N, must be in the form of 2^N) *A* and *B* where A = A AN

A_2 A_1 and B = B_N ….B_3 B_2 B_1. The final multiplication result will be of (N + N) bits as S = S (N + N) S_3 S_2 S_1.

Step 1: Divide the multiplicand A and multiplier B into two equal parts, each consisting of [N to (N/2) + 1] bits and [N/ 2 to 1] bits respectively, where first part indicates the MSB and other represents LSB.

Step 2: Represent the parts of A as A_M and A_L, and parts of B as B_M and B_L. Now represent A and B as A_M A_L and B_M B_L respectively.

Step 3: For A X B, we have general format as shown in Fig.3.3.2

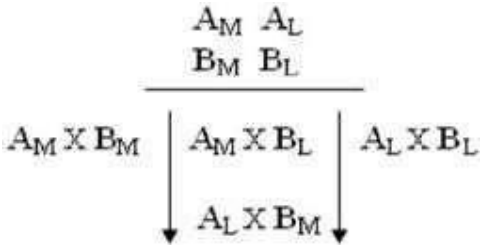

Fig. 3.3.2 General Representation for Vedic multiplication.

IV. PROPOSED SYSTEM

In this paper, we focus on proposing a high-speed low power/energy yet approximate multiplier appropriate for error resilient DSP applications. The proposed approximate multiplier, which is also area efficient, is constructed by modifying the conventional multiplication approach at the algorithm level assuming rounded input values. We call this rounding-based approximate (RoBA) multiplier.

The proposed multiplication approach is applicable to both signed and unsigned multiplications for which three optimized architectures are presented. The efficiencies of these structures are assessed by comparing the delays, power and energy consumptions, energy- delay products (EDPs), and areas with those of some approximate and accurate (exact) multipliers. The contributions of this paper can be summarized as follows: 1) presenting a new scheme for Ro BA multiplication by modifying the conventional multiplication approach; 2) describing three hardware architectures of the proposed approximate multiplication scheme for sign and unsigned operations.

A. MULTIPLICATION ALGORITHM OF ROBA MULTIPLIER

The main idea behind the proposed approximate multiplier is to make use of the ease of operation when the numbers are two to the power n (2n). To elaborate on the operation of the approximate multiplier, first, let us denote the rounded numbers of the input of A and B by Ar and Br, respectively. The multiplication of A by B may be rewritten as

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 7, ISSN No. 2455-2143, Pages 365-370
Published Online November 2019 in IJEAST (http://www.ijeast.com)

$$A \times B = (A_r - A) \times (B_r - B) + A_r \times B + B_r \times A - A_r \times B_r.$$

The key observation is that the multiplications of Ar × Br, Ar ×B, and Br ×A may be implemented just by the shift operation. The hardware implementation of (Ar − A) × (Br − B), however, is rather complex. The weight of this term in the final result, which depends on differences of the exact numbers from their rounded ones, is typically small. Hence, we propose to omit this part, helping simplify the multiplication operation. Hence, to perform he multiplication process, the following expression is used:

$$A \times B \cong A_r \times B + B_r \times A - A_r \times B_r$$

In this approach, the nearest values for A and B in the form of 2n should be determined. When the value of A (or B) is equal to the $3 \times 2^{p-2}$ (where p is an arbitrary positive integer larger than one), it has two nearest values in the form of 2n with equal absolute differences that are 2^p and 2^{p-1}. While both values lead to the same effect on the accuracy of the proposed multiplier, selecting the larger one (except for the case of p = 2) leads to a smaller hardware implementation for determining the nearest rounded value, and hence, it is considered in this paper.

It originates from the fact that the numbers in the form of $3 \times 2^{p-2}$ are considered as do not care in both rounding up and down simplifying the process, and smaller logic expressions may be achieved if they are used in the rounding up. The only exception is for three, which in this case, two is considered as its nearest value in the proposed approximate multiplier.

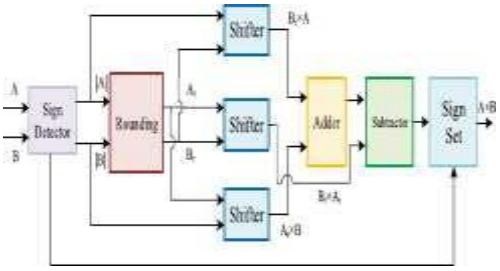

Fig.2: Block diagram for the hardware implementation of the ROBA multiplier.

It should be noted that contrary to the previous work where the approximate result is smaller than the exact result, the final result calculated by the RoBA multiplier may be either larger or smaller than the exact result depending on the magnitudes of Ar and Br compared with those of A and B, respectively. Note that if one of the operands (say A) is smaller than its corresponding rounded value while the other operand (say B) is larger than its corresponding rounded value, then the approximate result will be larger than the exact result. This is due to the fact that, in this case, the multiplication result of (Ar − A) × (Br − B) will be negative.

Since the difference between them is precisely this product, the approximate result becomes larger than the exact one. Similarly, if both A and B are larger (or) both are smaller than Ar and Br, then the approximate result will be smaller than the exact result. Finally, it should be noted the advantage of the proposed RoBA multiplier exists only for positive inputs because in the two's complement representation, the rounded values of negative inputs are not in the form of 2n.

Hence, we suggest that, before the multiplication operation starts, the absolute values of both inputs and the output sign of the multiplication result based on the inputs signs be determined and then the operation be performed for unsigned numbers and, at the last stage, the proper sign be applied to the unsigned result.

V. SIMULATION RESULTS

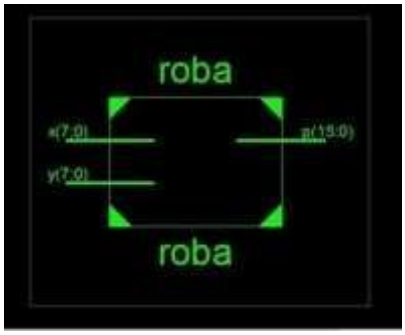

Fig 3:RTL Schematic

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 7, ISSN No. 2455-2143, Pages 365-370
Published Online November 2019 in IJEAST (http://www.ijeast.com)

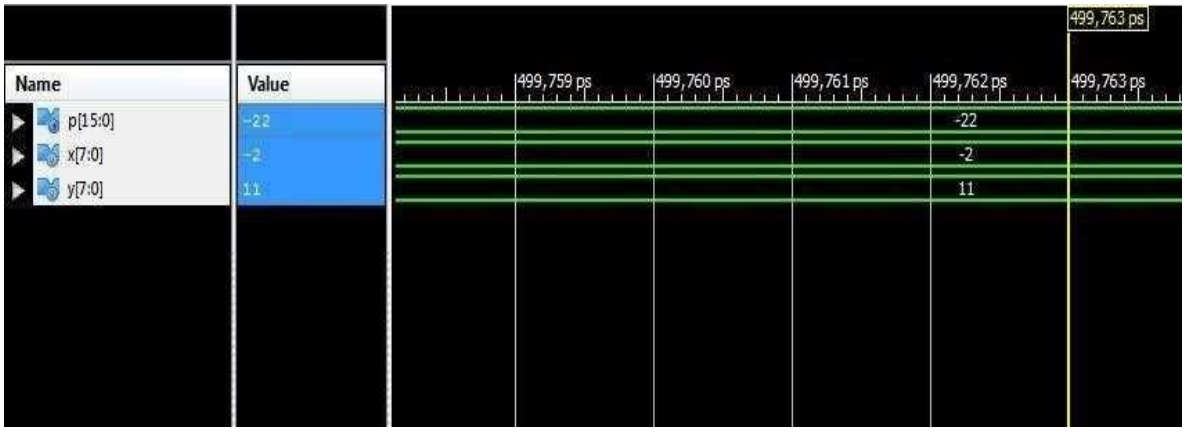

Fig4: Simulation Results

TABLE I

COMPARISON TABLE

PARAMETER	EXISTING SYSTEM	PROPOSED SYSTEM
DELAY(ns)	22.950	21.41
NO.OF LUT'S	190	156

VI. CONCLUSION

In this paper, we proposed a high-speed yet energy efficient approximate multiplier called RoBA multiplier. The proposed multiplier, which had high accuracy, was based on rounding of the inputs in the form of 2n. In this way, the computational intensive part of the multiplication was omitted improving speed and energy consumption at the price of a small error. The proposed approach was applicable to both signed and unsigned multiplications. Three hardware implementations of the approximate multiplier including one for the unsigned and two for the signed operations were discussed. The efficiencies of the proposed multipliers were evaluated by comparing them with those of some accurate and approximate multipliers using different design parameters. The results revealed that, in most (all) cases, the RoBA multiplier architectures outperformed the corresponding approximate (exact) multipliers.

This work is extended to implement fir filter using roba multiplier. It offers great advantage in the reduction of delay.

VII. REFERENCES

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

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

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

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

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

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

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

[8] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-

power high-speed multiplier for error-tolerant application," in Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits (EDSSC), Dec. 2010, pp. 1–4.

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

[10] K. Bhardwaj and P. S. Mane, "ACMA: Accuracy-configurable multiplier architecture for error-resilient system-on-chip," in Proc. 8th Int.Workshop Reconfigurable Commun.-Centric Syst.-Chip, 2013, pp. 1–6.

[11] K. Bhardwaj, P. S. Mane, and J. Henkel, "Power- and area-efficient appro ximate wallace tree multiplier for error-resilient systems," in Proc. 15th Int. Symp. Quality Electron. Design (ISQED), 2014, pp.263–269.

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

[13] V. Mahalingam and N. Ranganathan, "Improving accuracy in Mitchell's logarithmic multiplication using operand decomposition," IEEE Trans. Comput., vol. 55, no. 12, pp. 1523– 1535, Dec. 2006.

[14] Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available: http://www.nangate.com/

[15] H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.

[16] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, " Energy-efficient approximate multiplication for digital s ignal processing and classification applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun.2015.

[17] S. Hashemi, R. I. Bahar, and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD), Austin, TX, USA, 2015, pp. 418–425.

[18] C.-H. Lin and I.-C. Lin, "High accuracy approximate multiplier with error correction," in Proc. 31st Int. Conf. Comput. Design (ICCD), 2013, pp. 33–38.

[19] A. B. Kahng and S. Kang, "Accuracy - configurable adder for appro ximate arithmetic designs," in Proc. 49th Design Autom. Conf. (DAC), Jun. 2012, pp. 820–825.

[20] Z. Wang, A. C. Bovik, H. R. Sheikh, and E P. Simoncelli, " Image quality assessment: From error visibility to structural similarity," IEEE Trans. Image Process., vol. 13, no. 4, pp. 600– 612, Apr. 2004.


IJEAST