Journal of Advanced Research in Technology and Management Sciences
Volume: 01 Issue: 03 ISSN: 2582-3078 November 2019
Available online at: http://www.jartms.org

A 32-BitArea-Efficient Approximate Parallel Multiplier Design

[1]Harika Gandamaneni, [2]Dr.V.Thrimurthulu
[1]M.Tech (VLSI-SD)Student, [2] M.E., Ph.D,HOD and Professor
[1]ECE Department, Chadalawada Ramanamma Engineering College, Tirupati, A.P, India
[2] ECE Department, Chadalawada Ramanamma Engineering College, Tirupati, A.P, India

Abstract –

Approximate arithmetic has become a prominent choice for applications tolerating inaccurate results. By relaxing accuracy requirements, circuit complexity, delay, and energy consumption can be significantly reduced. In this paper, an approximate parallel multiplier design, based on simplified logic is being proposed by us. This is carried out by calculating product terms, then compressing the adjacent bits of same column based on the required cluster depths and thereby mapping the resulting product terms to achieve a less number of product rows. Thus, a reduction in silicon area is expected to be achieved. Multipliers with varying bit widths viz., 8-bit, 16-bit and 32-bit are designed using Verilog. Post-simulation results done using Xilinx ISE 12.1 tool, show that nearly 50% reduction in silicon area could be achieved compared to accuratemultiplier design. These multipliers could be used in power-constrained computing, multimedia applications, scientific computing etc.

Index Terms: – approximate arithmetic, cluster depth, power-constrained computing

I. INTRODUCTION

There is always a continuous demand to perform higher calculations at lower energy rates. The basic need for approximate calculations is to replace the traditional difficult and energy consuming data processing chunks with less complex ones with reduced logical counts. Thus, energy consumption is reduced at the expense of inaccuracy in the processed data. Applications such as digital signal processing, automation, multimedia, computer vision and statistical analysis have some degree of freedom for these concerns. Multiplier is an important arithmetic component in many of these applications for two important reasons. First, they are specified by a complex logical design, a data processing unit that challenges the energy of the latest microchips. Second, computationally intensive applications find it difficult and time consuming to perform multiple operations to calculate the result. Since the development of multiplier capabilities is expected to have a significant impact on overall structural capabilities, these factors have resulted in close recognition in the study of approximate multiplier designs. In rough circuitry, improvement in multipliers can be done related to either timing or behaviour. First improvement in timing can be achieved using auxiliary contribution voltage measurement development that requires an additional error repayment system to reduce error estimation. Second, useful technology deals with logical compression capabilities and can be manipulated by mitigating the need for accurate Boolean identities in favor of life force and circuit area compression. For example, shortening the multiplier product term can eradicate some of the smaller prominent partial product terms. As more rows are eradicated, more energy is depleted, but there is also a failure. Standard rearrangement using low difficulty linkage logic is another continuous skill. This allows you to develop larger energy capacity multipliers using small rough or inaccurate ones. An important concept in the above research is to achieve a compressed logic difficulty. A N-bit multiplier produces N^2product terms and size of the final product is

2N. Accuracy of the final result is proportional to the correctness of most-significant bits. In our work, we are going to implement the following steps:

a. Implement an approximate multiplier using bits Significance driven logic compression (SDLC) method.
b. The basis of our approach is a configurable logic compression of product terms then map them to reduce the number of product rows.

II. RELATED WORKS

The accurate multiplier design uses three stages –formation of partial product terms, arrangement or accumulation, and carry propagation adder. To perform N-bit multiplication, N^2 AND gates are used in parallel to generate a bit matrix of partial product terms. This matrix is then accumulated column by column to produce the final product using a carry propagation adder.

In the proposed approach, during first stage partial product terms are generated using the same number of AND gates as in traditional multiplication. Before going to the next stage, number of bits in the partial product bit matrix is reduced by performing lossy logic compression. This way the number of rows in the partial product matrix is reduced, thereby realizing less complex hardware before proceeding with accumulation. Figure 1 shows how the existing and proposed multipliers are implemented.

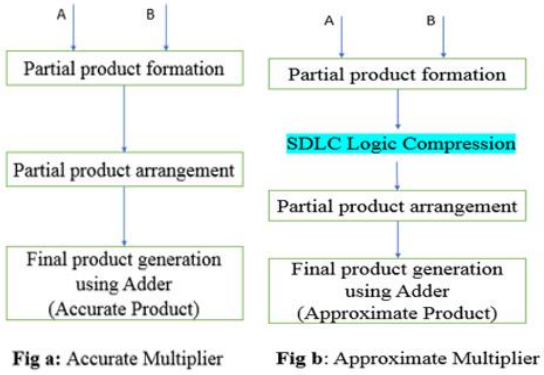


Fig a: Accurate Multiplier **Fig b**: Approximate Multiplier

Fig. 1. Process chart showing the difference between the major stages in:(a) conventional multiplication (b) the proposed approximate multiplication

Then number of rows before the cumulative stage should be reduced. This is achieved by rearranging the partial product terms based on the commutative nature of the bits. That is, bits with the equal weights are placed in the same column. By reducing the number of rows, the critical path delay is significantly reduced. Figure 3 shows the 8x8 multiplier'spartial product bit matrix which is achieved after following the SDLC approach.

III. PROPOSED APRROXIMATE MULTIPLIER DESIGN

Approximate computing devices in digital systems offer high-efficiency solutions by reducing design complexity and improved performance. The technique proposed in this paper explains the concept of bit significance driven logic compression for high-efficiency approximate multipliers. The proposed multiplier concept includes lossy compression of partial products, based on the cluster depths. At last, the total number of product rows is reduced by remapping the resultant partial product terms. Therefore, the

design complexity and critical path length of the multiplier is greatly reduced when compared to a normal exact multiplier. The proposed multiplier design is evaluated in Verilog, synthesized in Xilinx ISE 12.1.

The proposed approach consists of two main steps. First, lossy compression is performed through logic clustering then the remaining product terms are remapped in a way that the total number of product rows are reduced. These steps and variable compression methods are described below –

1) Logic compression: Parallel multiplication designs are usually divided into three stages: partial product generation, accumulation and final product generation using carry propagation adder. In (N × N) multipliers, N^2 AND gates are used in parallel to generate a partial product bit matrix. This matrix is then accumulated column by column to produce the final product using a carry propagation adder. The proposed approach does partial product generation using the same number of AND gates, similar to traditional multiplication. Before proceeding to the next stage, the number of bits in the partial product matrix is reduced by performing lossy logic compression. Thus the number of rows in the partial product matrix are reduced, thereby realizing less complex hardware. Figure 1 shows how the existing and proposed multipliers are implemented. To achieve lossy compression, follow three important principles: "Row group clustering: The proposed multiplier organizes partial product terms using logic clusters of different sizes. Each logical cluster aims at a set of columns with 2 bits starting from the LSB of a contiguous partial product. Each logic cluster, 2 x L, is responsible for two operations:

i) Using 2L AND gates, 2L partial product bits, i.e., vertically aligned L pair bits, in two consecutive rows

ii) minimize these 2L bits in half using L OR gates.

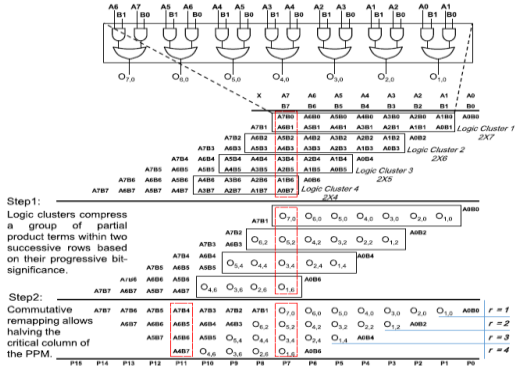


Fig: 2 - Block diagram of 8x8 Approximate Multiplier using SDLC approach

Figure 2 shows the utilization of four different sizes of logic clusters in an 8-bit parallel multiplier, cluster size being varied from 2x7 to 2x4. By doing so, each logical cluster compresses a group of vertically aligned bits in two consecutive partial products. Implementing this logic on any convenient multiplier models such as carry-save array, Wallace tree, Dada tree, etc., you are ready to accumulate a reduced set of preprocessed partial product terms. Theoretically, a two-input AND gate or OR gate outputs same result for all combinations except when both the inputs are 1. The main goal to use SDLC method is to design a multiplier with improved power efficiency that loses little accuracy, so reducing the partial product matrix reduces the size of the logic cluster. The upper bits are progressively processed with higher precision and the lower bits are compressed. This allows you to accumulate the most important product terms on a carry propagation basis, like traditional multipliers. Therefore, the accuracy of critical bits in the final product is not affected. Though the same number of AND gates are used as in exact multiplier, this approach is complicated because of the complex hardware of partial product accumulation, such as the number of compression cells required for column compression multiplication in the case of Wallace and Dadda. Reduce deterministically. Since the number of bits in the cumulative tree is minimized, half-adders and full-adders in carry save arrays are reduced.

2) Commutative Remapping: The logical compression step reduces the number of partial product terms. This step reduces the number of rows before the cumulative stage by rearranging the partial product terms based on the commutative nature of the bits in which the bits with equal weights are collected in the same column. By reducing the number of rows, delay along the critical path is significantly reduced. Variable logical cluster approach can be used to achieve more advanced compression by increasing the depth of the logical cluster.

IV. ANALYSIS

Verilog codes and executes the associated test bench. By reducing complexity, the silicon area could also be reduced by 33.4% -62.9%. Increasing the depth of the logic cluster also reduces the hardware complexity associated with decreasing the number of product lines, so increasing the clustering depth can result in significant savings in all design tradeoffs. The results obtained after synthesis show a significant reduction in run time and even silicon area. The proposed approach is believed to be used with the existing low-power computing units to extract a variety of benefits with minimal loss of output quality.

V. RESULTS

To demonstrate the proposed approach, the algorithm is applied and designed 8-bit, 16-bit and 32-bit multipliers. To arrange the partial product rows during the accumulation stage, accurate ripple adders were used in both accurate and approximate multipliers. The Verilog code was written to generate all synthesizable modules in the multiplier and code was simulated in Xilinx ISE 12.1 tool.

The below figures depict the simulation results of proposed 16-bit & 32-bit approximate multipliers:

Fig: 16-bit

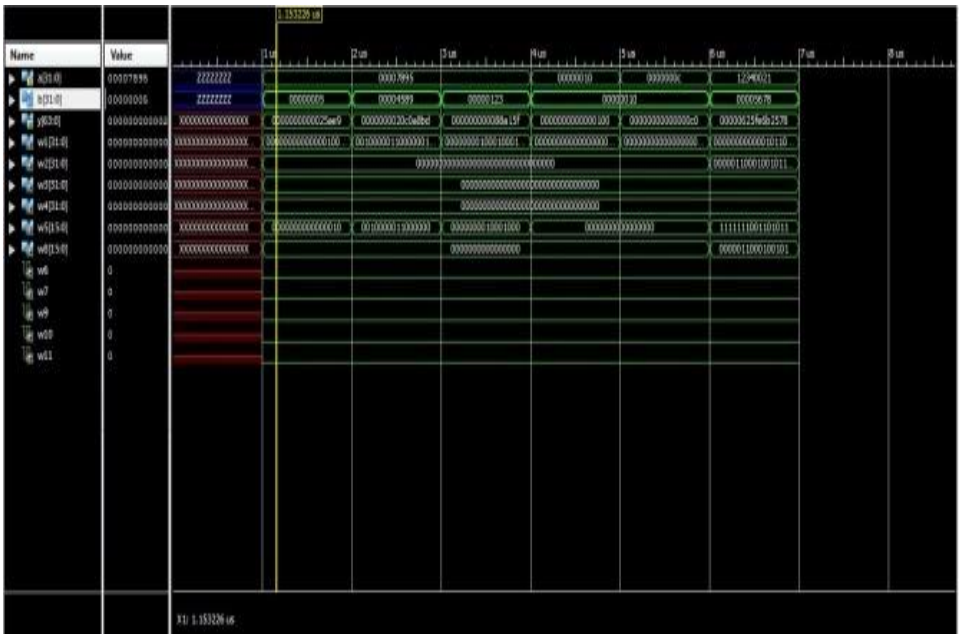


Fig: 32-bit

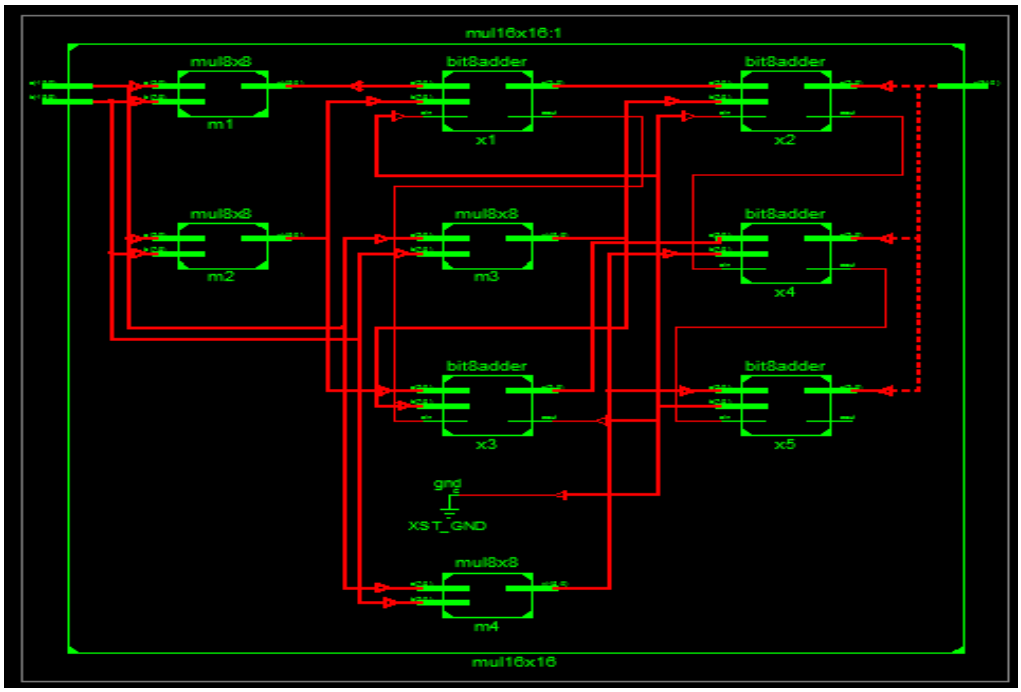
In above results, consider the case when a=7895H,b=0005H then result of the multiplier is y=00025AE9H. Here the result is matching with accurate multiplier value and so error is 0%.

Consider the case, a=7895H, b=4589H then the result is y= 20C0A8BD. The exact value of the multiplication is 20C0B0BD. From this result, it can be observed that, in proposed multiplier the MSB values are preserved and the error is at LSB values. Here also the error percentage is very less.

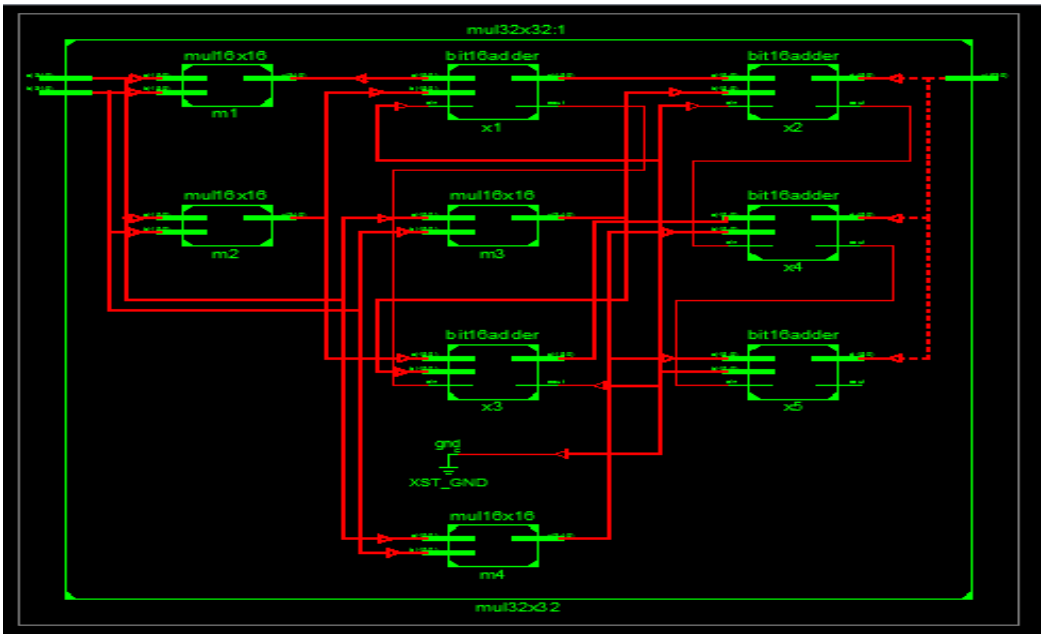
The below table depicts the usage of number of adder cells in proposed multiplier and accurate multiplier:

Size of the Multiplier	**Number of adders required**	
	Array Multiplier	**Proposed Compressed Multiplier**
8-bit	56	30
16-bit	240	160
32-bit	992	720

Synthesis Results:



16-bit



32-bit

REFERENCES

[1] S. Mittal, A survey of techniques for approximate computing, ACMComput. Surv., vol. 48, no. 4, pp. 62-1–62-33, Mar. 2016.
[2] V. De, Energy-efficient computing in nanoscale CMOS, IEEE DesignTest, vol. 33, no. 2, pp. 68–75, Apr. 2016.
[3] M. Shafique, R. Hafiz, S. Rehman, W. El-Harouni, and J. Henkel,Invited: Cross-layer approximate computing: From logic to architectures,in Proc. DAC, Jun. 2016, pp. 1–6.
[4] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in Proc. ETS, May 2013, pp. 1–6.
[5] "Logic Compression for Energy-Efficient Multiplier Design", IEEE Journal on Emerging and selected topics in Circuits and Systems, vol.8, no 3, September 2018.
[6] "Approximate Multiplier Techniques- A Survey", INTERNATIONAL JOURNAL FOR RESEARCH IN EMERGING SCIENCE AND TECHNOLOGY, VOLUME-4, ISSUE-5, MAY-2017
[7] Kai Hwang "Computer Arithmetic: Principles, Architecture, and Design" John Wiley & Sons 1979
[8] S. D. Pezaris "A 40-ns 17-Bit by 17-Bit Array Multiplier", IEEE Trans. on Computers, pp. 442-447,.Abr. 1971
[9] C. Baugh y A. Wooley "A Two's Complement Parallel Array Multiplication Algorithm". IEEE Trans.on Computer, Vol.C-22, Nº12. Dic.1973.
[10] K. Bhardwaj, P. S. Mane, and J. Henkel, "Power- and area-efficient approximate wallace tree multiplier for error-resilient systems," in Proc. ISQED, Mar. 2014, pp. 263–269.
[11] 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.