2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS) | 978-1-6654-0521-8/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICACCS51430.2021.9441960

Fast Approximate Matrix Multiplier based on Dadda Reduction and Carry Save Ahead Adder

Arun Kumar
Dept. of Electronics and Communication Engineering
Delhi Technological University
New Delhi,India
arunkumar.06185@gmail.com

Asif Ansari
Dept. of Electronics and Communication Engineering
Delhi Technological University
New Delhi,India
aasifansari1811@gmail.com

Ayush Srivastava
Dept. of Electronics and Communication Engineering
Delhi Technological University
New Delhi,India
ayushsrivastavakv8@gmail.com

Kriti Suneja
Dept. of Electronics and Communication Engineering
Delhi Technological University
New Delhi,India
kritisuneja@dtu.ac.in

***Abstract*—Exact computation techniques require a high amount of resources and time. But the need of today's Electronics world is a system that is faster and requires less area. Approximate Computing has emerged as a technique that plays with the trade-off between the requirement of exact computing to improve area requirement, speed, and power performance. In this paper, a speed-efficient model for approximate matrix multiplier is proposed with the use of the Dadda compression technique followed by the Carry-Save Ahead (CSA) adder. Hardware design is modeled in Verilog. Artix-7 xc7a100t-3csg324 is used as a target device in Xilinx ISE Design Suite 14.7. ISIM simulation results and device utilization summary have been presented. Synthesis results show that the proposed design is faster, requires less area, consumes less power than published designs.**

Keywords— approximate computing, Dadda reduction tree, Carry Save Ahead (CSA) Adder, Look Up Table (LUT)

I. INTRODUCTION

Approximate Computing (AC) has emerged as an efficient model that leverages the error tolerance feature to improve the overall resource efficiency of a digital system in terms of area and power requirement. To reduce design complexity, the logic is simplified by replacing traditional complex and energy wasteful computational blocks with the low complex approximate units that can perform the same function to much extent without making any significant impact on the result [3]. Domains such as computer vision, machine learning, data mining, etc. are few examples of domains where error tolerance in computations is a common factor [2] [4]. Trivial errors in computations do not result in significant performance degradation in these applications [2]. Hence approximate computing can be considered as an opportunity for energy-efficient system design which provides faster results at the cost of accuracy. Arithmetic circuits such as adders, multipliers are crucial arithmetic units responsible for the applications which are computationally intensive and thus substantially influence the overall area and power requirement of the entire system. Therefore, optimization/simplification of these modules can greatly reduce energy requirements and enhance the performance efficiency of the overall system [3].

A great amount of research has been done related to multipliers and adders over the years to find the best multipliers in different scenarios. Our paper focuses on approximate matrix multiplication which is faster and consumes less area. In [1] an approximate matrix multiplier was proposed in which after generating the partial products, they were divided into two halves, henceforth referred to as Gupta's model or previous model. One was an Accurate Part that was used for MSB's part calculation and the other was an Inaccurate Part that was used for LSB's part calculation. The Inaccurate part utilizes OR gate approximation to get the final product for the LSB part. And the Accurate Part, first, utilizes the Wallace reduction technique on the upper half columns of partial product. Followed by RCA to get complete approximate multiplication. Elements of the result matrix are calculated with the summation of approximate multiplier results with approximate adder which utilizes the same approximation technique as approximate-multiplier [5].

In this paper an approximate matrix multiplier is proposed which is faster, delay follows more linear nature with doubling the operand size and utilizes nearly equal or less area as compared to the model proposed in [1].

The organization of the paper is as follows: Section 2 briefly discusses the proposed architecture, section 3 shows the simulation and results and Section 4 concludes the paper.

978-1-6654-0521-8/21/$31.00 ©2021 IEEE

II. PROPOSED ARCHITECTURE

The most common method for designing an approximate circuit is dividing the circuit into 2 parts i.e. definite part and indefinite part. The definite part is used for MSB's calculation while the Indefinite adder accounts for LSB's Calculation. The proposed architecture is divided into two halves. The lower half calculation is done by using OR gate approximation. The upper half calculation is carried out by first multiplication of operands with Dadda compression technique for the definite part and OR gate approximation for the lower indefinite part, then for the definite part, Carry Save Ahead (CSA) adder is used to add the reduced stages and finally, Ripple Carry adder is used to get the final definite part of the result. Whereas, for indefinite-part, OR gate approximation is utilized for addition. Generally, the length of a definite and indefinite part depends upon the application but in this case, lengths of definite and indefinite part have been taken nearly equal [6-7].

Multiplication involves two procedural steps:

- Generation of Partial Products.
- Summation of Partial Products to generate the final result.

The first step is common for both the definite and the indefinite-part.

A. Lower Part OR-Based Approximate Adder

For operand of N bit, summation for lower l bits is done by OR gate approximation whereas the remaining higher N-l bits utilize accurate adder like RCA. The value of l determines the approximate part in the result as well as the error percentage in the final result. Thus, the value of l is application-dependent.

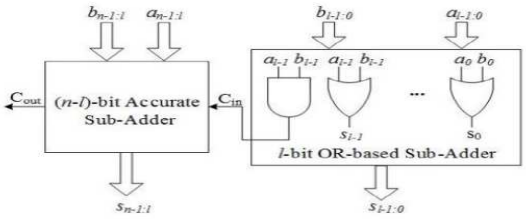

Fig. 1. Lower Part OR-Adder [5]

B. Lower Part OR-Based Dadda Reduction Tree

Approximate multiplier utilizes approximate summation of partial product. In the proposed reduction tree, the partial product is divided into two parts, accurate-block for the MSBs and inaccurate-block for LSBs. Dadda reduction scheme is used in accurate-block to compress the partial products into two rows of equal width (row-0, row-1). Figure 2 shows how the reduction is done.

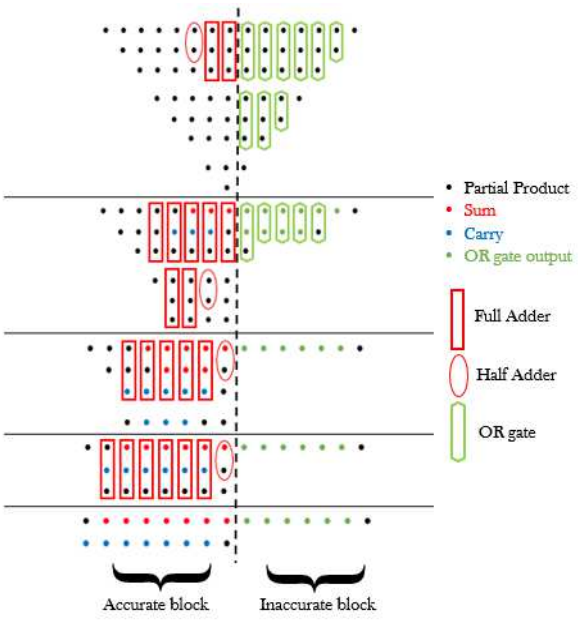

Fig. 2. Dadda Reduction scheme for Proposed Architecture

These rows then directly use as operands in the CSA adder thus saving the delay of RCA at cost of twice the operand in CSA adder. The Inaccurate block employs OR-gate for the reduction of partial products. Fig. 3 below shows the complete model of the proposed architecture for n-bit 2x2 matrix multiplication.

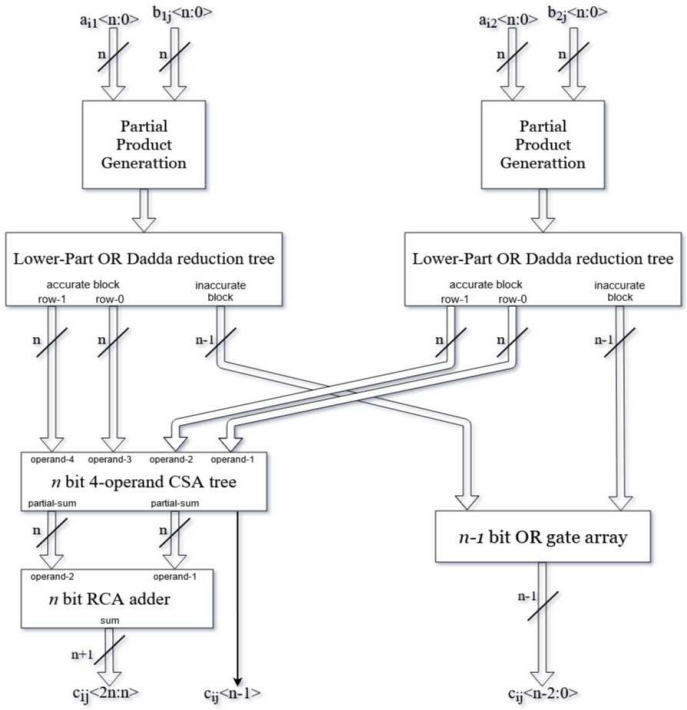

Fig. 3. Block diagram of approximate cij calculation for n-bit 2x2 matrix.

In Fig.3, ai1,ai2 are the elements of a 2x2 multiplicand matrix 'a' and b1j,b2j are the elements of 2x2 multiplier matrix 'b' with i,j could be either 1 or 2 for 2x2 matrix. The result of the given module cij is the element of the product matrix 'c'.

III. RESULTS AND ANALYSIS

A 3x3 8bit unsigned matrix multiplier of the proposed matrix multiplier was implemented using Verilog HDL for comparison of results with Exact Multiplier (utilizing Wallace Tree Multiplier and Ripple Carry Adder) and previously existing Approximate Multiplier published in [1]. It was verified that the proposed design has a reduction in time and consumes less Area. Comparisons for other bit sizes and matrix size were also done by varying the sizes using Verilog HDL, and Xilinx ISE 14.7 was used for logic simulations.

Table I shows the reduction in delay and area for the proposed design by introducing a negligible amount of error in calculations. Tables II and III show the comparison of delay and Area (Slice LUT's) respectively for different operand and matrix sizes. Fig. 4 and Fig. 5 show the comparison graphically, it can be observed that the delay of the proposed model shows a more linear nature with doubling the operand size with nearly the same LUT utilization as Gupta's Matrix multiplier model.

TABLE I. COMPARISON OF 3X3 8BIT MATRIX MULTIPLIER

MATRIX MULTIPLIER	AREA (LUT)	DELAY (ns)	ERROR %
Exact Multiplier	3177	10.873	0
Gupta's Matrix Multiplier	1809	8.377	0-1%
Proposed Approximate Multiplier	2196	7.154	0-1%

TABLE II. DELAY COMPARISON OF PREVIOUS WORK IN [1] VS PROPOSED MODEL FOR DIFFERENT DATA AND MATRIX SIZE

MATRIX SIZE	DELAY (ns)					
	4 bit		*8 bit*		*16 bit*	
	Gupta's model	*proposed model*	*Gupta's model*	*proposed model*	*Gupta's model*	*proposed model*
2x2	4.612	3.87	7.099	6.097	14.282	9.077
3x3	5.438	4.815	8.377	7.154	15.902	9.888
4x4	5.449	5.469	7.987	7.797	15.85	10.032

TABLE III. AREA COMPARISON OF PREVIOUS WORK IN [1] VS PROPOSED MODEL FOR DIFFERENT DATA AND MATRIX SIZE

MATRIX SIZE	SLICE LUTs					
	4 bit		*8 bit*		*16 bit*	
	Gupta's model	*proposed model*	*Gupta's model*	*proposed model*	*Gupta's model*	*proposed model*
2x2	176	146	524	624	2371	2144
3x3	588	531	1809	2196	8153	7406
4x4	1469	1279	4400	5296	19391	17639

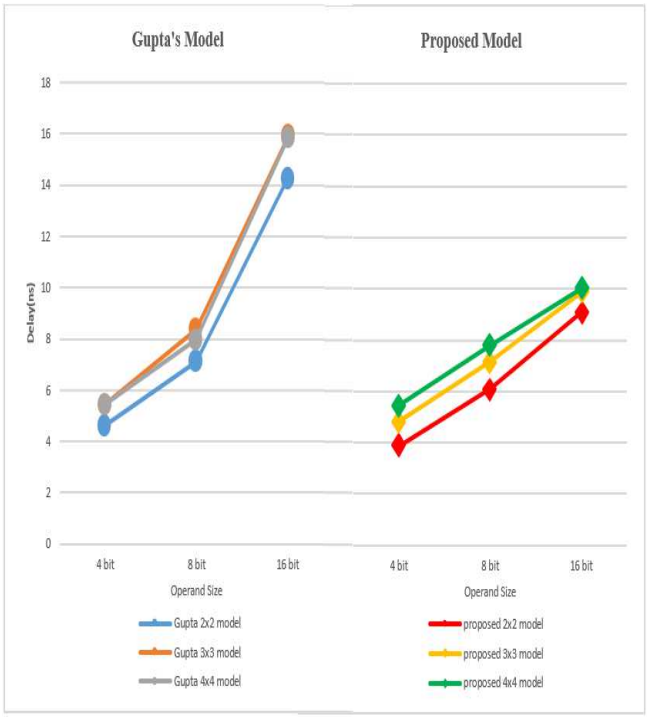

Fig. 4. Delay comparison

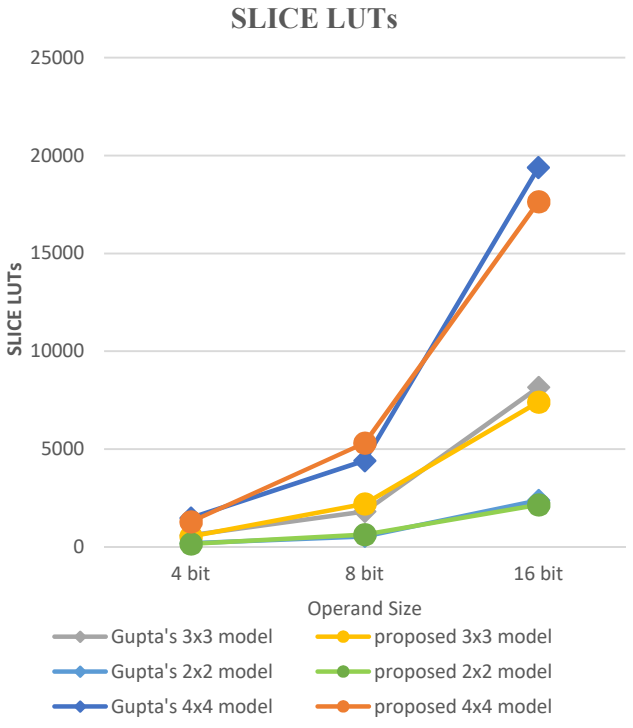

Fig. 5. LUTs comparison

IV CONCLUSION

In this paper, a novel approximate matrix multiplier has been proposed and results show that it is faster than previously published work with nearly the same SLICE LUT utilization. Results show that the delay of the proposed model shows a more linear nature with doubling the operand size as compared to the model in [1]. The speed of the matrix multiplier is traded off with precision. Introducing a negligible error in the multiplication result a significant amount of improvement can be achieved in terms of speed and area. So, the proposed multiplier can be used to get faster results in applications like Data Mining, Computer vision, Machine learning, etc.

REFERENCES

[1] A. Gupta and K. Suneja, "Hardware Design of Approximate Matrix Multiplier based on FPGA in Verilog," 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2020, pp. 496-498.

[2] Jie Han, Michael Orshansky, "Approximate Computing: An Emerging Paradigm For Energy-Efficient Design", design," in Proc. European Test Symposium (ETS), pp. 1-6, May 2013.

[3] H. Anandakumar and K. Umamaheswari, "Supervised machine learning techniques in cognitive radio networks during cooperative spectrum handovers," Cluster Computing, vol. 20, no. 2, pp. 1505–1515, Mar. 2017.

[4] H. Anandakumar and K. Umamaheswari, "A bio-inspired swarm intelligence technique for social aware cognitive radio handovers," Computers & Electrical Engineering, vol. 71, pp. 925–937, Oct. 2018. doi:10.1016/j.compeleceng.2017.09.016

[5] Qiqieh, R. Shafik, G. Tarawneh, D. Sokolov, S. Das and A. Yakovlev, "Energy-efficient approximate wallace-tree multiplier using significance-driven logic compression," 2017 IEEE International Workshop on Signal Processing Systems (SiPS), Lorient, 2017, pp. 1-6.

[6] Honglan Jiang, Cong Liu, Leibo Liu, Fabrizio Lombardi, and Jie Han," A Review, Classification, and Comparative Evaluation of Approximate Arithmetic Circuits". J. Emerg. Technol. Comput. Syst. 13, 4, Article 60 (August 2017), 34 pages.

[7] Mohammad Saeed Ansari , Student Member, IEEE, Honglan Jiang,Bruce F. Cockburn , Member, IEEE, and Jie Han , Senior Member, IEEE, " Low-Power Approximate Multipliers Using Encoded Partial Products and Approximate Compressors", IEEE.