ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Design of High Performance 64 bit MAC Unit

T.Bhavani
M.Tech Student,
Department of ECE,
KITS for Women's, kodad, T.S, India.

Ms.K.Anuradha
Associate Professor,
Department of ECE,
KITS for Women's, kodad, T.S, India.

ABSTRACT:

This paper presents a high speed binary floating point multiplier based on Dadda Algorithm. To improve speed multiplication of mantissa is done using Dadda multiplier replacing Carry Save Multiplier. The design achieves high speed with maximum frequency of 526 MHz compared to existing floating point multipliers. The floating point multiplier is developed to handle the underflow and overflow cases. To give more precision, rounding is not implemented for mantissa multiplication. The multiplier is implemented using Verilog HDL and it is targeted for Xilinx Virtex-5 FPGA. The multiplier is compared with Xilinx floating point multiplier core.

Keywords:

Dadda Algorithm; Floating point; multiplication; FPGA, Verilog HDL;

I.INTRODUCTION:

Unit is an inevitable component in many digital signal processing (DSP) applications involving multiplications and/or accumulations. MAC unit is used for high performance digital signal processing systems. The DSP applications include filtering, convolution, and inner products. Most of digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transforms (DWT). Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic determines the execution speed and performance of the entire calculation. Multiplication –and -accumulate operations are typical for digital filters. Therefore, the functionality of the MAC unit enables high-speed filtering and other processing typical for DSP applications.

Since the MAC unit operates completely independent of the CPU, it can process data separately and thereby reduce CPU load. The application like optical communication systems which is based on DSP, require extremely fast processing of huge amount of digital data. The Fast Fourier Transform (FFT) also requires addition and multiplication. 64 bit can handle larger bits and have more memory. A MAC unit consists of a multiplier and an accumulator containing the sum of the previous successive products. sixth section and finally obtained results is discussed in seventh and the conclusion is made in the eight section.

II. MAC OPERATION:

The Multiplier-Accumulator (MAC) operation is the key operation not only in DSP applications but also in multimedia information processing and various other applications. As mentioned above, MAC unit consist of multiplier, adder and register/accumulator. In this paper, we used 64 bit modified Wallace multiplier. The MAC inputs are obtained from the memory location and given to the multiplier block. This will be useful in 64 bit digital signal processor. The input which is being fed from the memory location is 64 bit. shown below in figure 1.The function of the MAC unit is given by the following equation [1]:

$$F=\sum Pi\ Qi \qquad (1)$$

The output adder and accumulator block is 129 bit i.e. one bit is for the carry (128bits+ 1 bit). Then, the output is fed back to the same adder and accumulator block. The figure 1 shows the new architecture of MAC unit.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

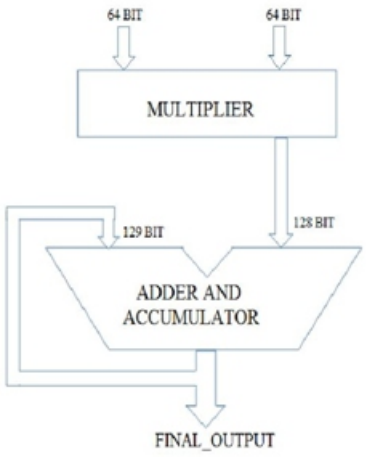

Figure 1 : New architecture of MAC unit.

III. 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.

A.Vedic Multiplier for 2x2 bit Module:

The method is explained below for two, 2 bit numbers A and B where A = a1a0 and B = b1b0 as shown in Fig. 1. Firstly, the least significant bits are multiplied which gives the least significant bit of the final product (vertical). Then, the LSB of the multiplicand is multiplied with the next higher bit of the multiplier and added with, the product of LSB of multiplier and next higher bit of the multiplicand (crosswise). The sum gives second bit of the final product and the carry is added with the partial product obtained by multiplying the most significant bits to give the sum and carry. The sum is the third corresponding bit and carry becomes the fourth bit Of the final product[10]. Figure 1 shows the block diagram of 2x2 bit Vedic Multiplier which is further used for the implementation of the 4x4 bit vedic multiplier and further 8x8 vedic multiplier is implemented.

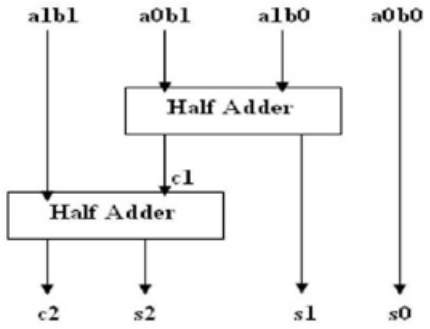

Figure 2[10]:Block Diagram of 2x2 bit Vedic Multiplier.

B.Vedic Multiplier for 4x4 bit Module The proposed Vedic multiplier can be used to reduce delay. Early literature speaks about Vedic multipliers based on array multiplier structures. On the other hand, we proposed a new architecture, which is efficient in terms of speed. The arrangements of RC Adders shown in Fig. 2, helps us to reduce delay. Interestingly, 8x8 Vedic multiplier modules are implemented easily by using four 4x4 multiplier modules.

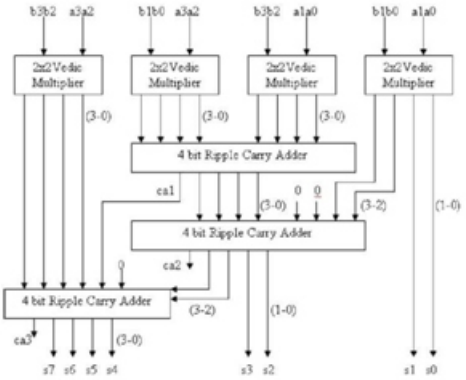

Figure 3[10]: Block Diagram of 4x4 bit Vedic Multiplier.

C.Vedic Multiplier for 8x8 bit Module:

The 8x8 bit Vedic multiplier module as shown in the block diagram in Fig. 3 can be easily implemented by using four 4x4 bit Vedic multiplier modules as discussed in the previous section Let"s analyze 8x8 multiplications, say A= A7 A6 A5 A4 A3 A2 A1 A0 and B= B7 B6 B5B4 B3 B2 B1B0. The output line for the multiplication result will be of 16 bits as – S15 S14 S13 S12 S11 S10 S9 S8 S7 S6bS5S4 S3 S2 S1 S0. 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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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 Fig.3.Since the 64 bit vedic multiplier is difficult to represent, a typical 8-bit block diagram is shown in figure 4 for understanding.

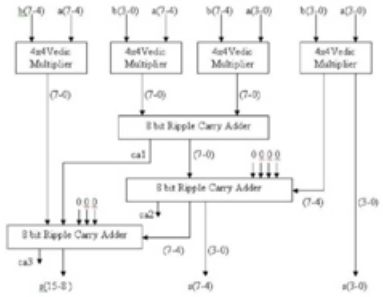

Figure 4[10]: Block Diagram of 8x8 bit Vedic Multiplier.

IV. DADDA MULTIPLIER:

Dadda multipliers are the refinement of parallel multipliers first presented by Wallace in 1964. In contrast to the Wallace reduction Dadda multiplier perform the least reduction at each stage[2]. The maximum height of each stage is determined by working back from final stage which consists of two rows of partial products.

The height of each stage should be in the order 2,3,4,6,9,13,19,28,42,63 etc. An 8 bit Dadda multiplier reduction is shown in figure 5.For Dadda multipliers the number of full adders and half adders required depends on the value of N.

No of Full Adders = N 2- 4N+3

No of Half Adders = N-1

Since the 64 bit dadda multiplier is difficult to represent, a typical 8-bit by 8-bit reduction shown in figure 5 for understanding.

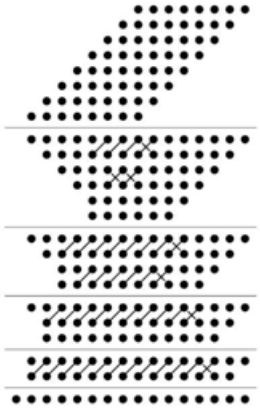

Figure 5[2]: Dadda Multiplier for N=8

V. BRAUN'S MULTIPLIER:

Braun's multiplier is an n × m bit parallel multiplier and generally known as carry save multiplier and is constructed with m × (n-1) address and m × n AND gates. The Braun's multiplier has a glitching problem which is due to the ripple carry adder in the last stage of the multiplier.

A. Mathematical Basics:

Consider a generic m by n multiplication of two unsigned n-bit numbers Y = Ym-1 …. Y0 and X = Xn-1…X0

$$Y = \sum_{i=0}^{m-1} Y_i 2^i \qquad (2)$$

$$X = \sum_{i=0}^{n-1} X_i 2^i \qquad (3)$$

The product P = P2n-1 …P1P0, which results from multiplying the multiplicand Y by the multiplier X, can be written as follows:

$$P = XY = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (X_i.Y_j) 2^{i+j} \qquad (4)$$

An n*n bit Braun multiplier is constructed with n(n-1) adders and n2 AND gates as shown in the fig.6, where,
X: 4 bit Multiplicand
Y: 4 bit Multiplier
P: 8 bit Product of X & Y
Pn : Xi Yi is a Product bit
The internal structure of the full adder can be realized using FPGA. Each products can be generated in parallel with the AND gates. Each partial product can be added with the sum of partial product which has previously produced by using the row of adders. The carry out will be shifted one bit to the left or right and then it will be added to the sum which is generated by the first adder and the newly generated partial product.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

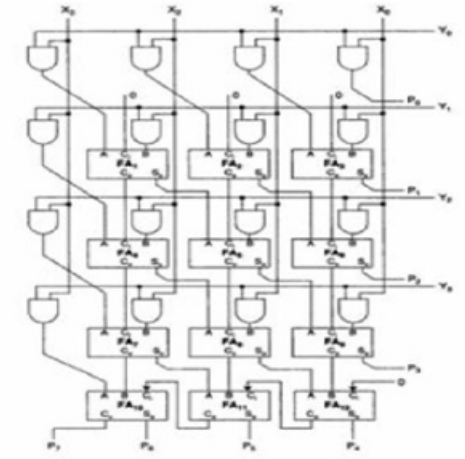

Figure 6[3]: Braun multiplier for n=4

Since the 64 bit braun multiplier is difficult to represent, a typical 4-bit architecture is shown in the above figure 6.

VI. CARRY SAVE ADDER:

The carry-save unit consists of n full adders, each of which computes a single sum and carry bit based solely on the corresponding bits of the three input numbers. Given the three n - bit numbers a, b, and c, it produces a partial sum PS and a shift-carry
SC.
PSi = ai^bi^ci (5)
SCi = (ai^bi)|(ai^ci)|(bi^ci)
(6)
Since the representation of 128 bit carry save adder is infeasible , hence a typical 8 bit carry save adder is shown in the figure 7[1].Here we are computing the sum of two 128 bit binary numbers, then 128 half adders at the first stage instead of 128 full adder. Therefore , carry save unit comprises of 128 half adders, each of which computes single sum and carry bit based only on the corresponding bits of the two input numbers[1].

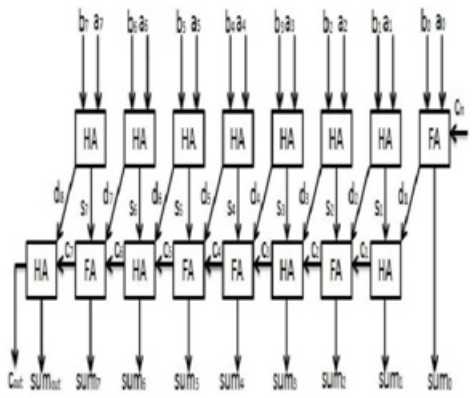

Figure 7[12]: Carry save adder for n=8

VII. RESULTS:

The design is done using Verilog-HDL by using tool Xilinx ISE 10.1i and target family Spartan 3E,Device-XC3S500,speed-5,package: FG320.As a previous work,64 bit MAC is constructed using Wallace multiplier but here different MAC units are constructed and compared the performance with the earlier is done here the multipliers designed are below (i) Vedic multiplier (ii) Dada multiplier (iv) Braun multiplier (v) Wallace multiplier. The delay table and the corresponding graph for different multipliers is shown below.Hence,form the the below table it is clear that the delay for the vedic,dadda.,braun multipliers is less than the Wallace multipliers if we choose the best multiplier to implement it in MAC unit it may results in better performance and better results.

MULTIPLIER	Maximum Combinational path Delay(ns)			
	8 BIT	16 BIT	32 BIT	64 BIT
VEDIC	18.353	24.541	31.835	33.882
DADDA	20.871	28.037	35.372	37.57
BRAUN	21.049	28.155	35.444	37.642
WALLACE	26.248	33.354	40.643	42.842

TABLE I.:DELAY OF DIFFERENT MULTIPLIERS:

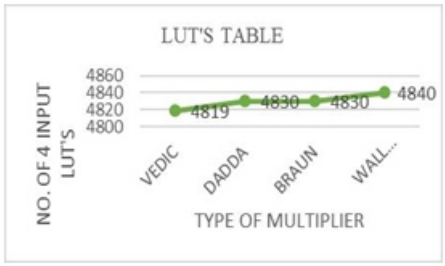

Figure 9 : No.of 4 input LUT comparison of different MAC units.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

TABLE II .DELAY OF MAC UNITS.

SI. NO	MAC NAME	LOGIC DELAY (ns)	ROUTE DELAY (ns)	TOTAL DELAY (ns)
1	VEDIC MULTIPLIER CARRY SAVE ADDER	245.03	70.73	325.76
2	DADDA MULTIPLIER CARRY SAVE ADDER	251.73	79.02	330.76
3	BRAUN MULTIPLIER CARRY SAVE ADDER	260.12	83.03	343.15
4	WALLACE MULTIPLIER CARRY SAVE ADDER	263.57	86.43	350.01

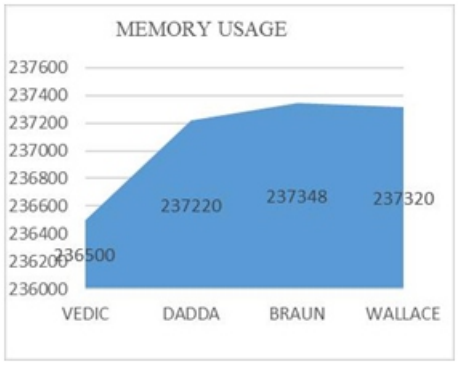

Figure 10: Slice usage of different MAC unit. Figure 11 : Memory usage of different MAC units.

The figure 8 and 9 shows the delay and no. of 4 input LUT for different types of MAC units. The figure 10 shows the no. of slices for the above four MAC units and figure 11 represents the amount of memory taken by different MAC units. From the above all the results it is clear that the vedic,dadda,braun MAC units has least delay and less memory required than Wallace MAC unit.

The simulation results of 64 bit different types of MAC units is shown below. Most of the simulation result it is observed that there is a delay by one clock cycle, this is because it takes some time to compute since the multiplier used here is 64 bit. The overall simulated output is shown in below figures.

The below figures from figure 12 to figure 14 shows the simulation results of the different types of 64 bit MAC units.

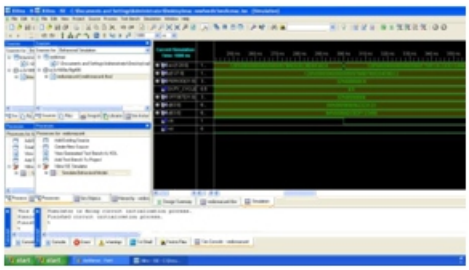

Figure 12: Simulation result of Vedic MAC unit.

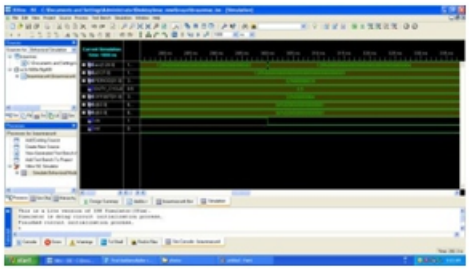

Figure 13: Simulation result of Braun MAC unit.

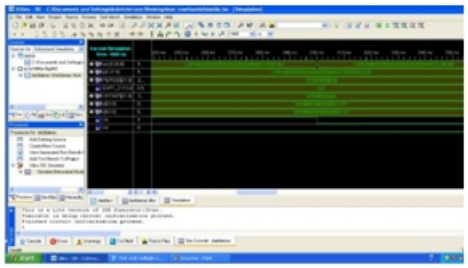

Figure 14: Simulation result of dadda MAC unit.

VIII. CONCLUSION:

I am Anusha Kola and would like to thank the publishers, researchers for making their resources material available. I am greatly thankful to Assistant Prof: Miss.Ch.Nirmala for their guidance. We also thank the college authorities, PG coordinator and Principal for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and family members.

IX. ACKNOWLEDGMENTS:

I am T.BHAVANI and would like to thank the publishers, researchers for making their resources material available. I am greatly thankful to Assistant Prof: Miss .K.Anuradha for their guidance. We also thank the college authorities, PG coordinator and Principal for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and family members

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

REFERENCES:

[1]P.jagadeesh"Design ofHigh Performance 64 bit MAC Unit" 2013 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2013].

[2]W.J. Townsend, E.E. Swartzlander Jr., and J.A. Abraham, "A Comparison of Dadda and Wallace Multiplier Delays, "Proc.SPIE, Advanced Signal Processing Algorithms, Architectures, and Implementations XIII, pp. 552-560, 2003.

[3]M.H. Rais, M.H. Al Mijalli, "Braun's multipliers: Spartan-3AN based design and implementation", J. Comput. Sci., vo. 7, no. (11), pp.1629-1632, 2011.

[4]L. Dadda, "Some Schemes for Parallel Multipliers," Alta Frequenza,vol. 34, pp. 349-356, 1965.

[5]C.S. Wallace, "A Suggestion for a Fast Multiplier," IEEE Trans.Electronic Computers, vol. 13, no. 1, pp. 14-17, Feb. 1964.

[6]Ron S. Waters and Earl E. Swartzlander, Jr., "A Reduced Complexity Wallace Multiplier Reduction, " IEEE Transactions On Computers, vol. 59, no. 8, Aug 20 10.

[7]Rais, M.H., 2009a. FPGA design and implementation of fixed width standard and truncated 6×6-bitmultipliers: A comparative study. Proceedings of the 4th IEEE International Design and TestWorkshop, Nov. 15-17, IEEE Xplore Press,Riyadh, Saudi Arabia, pp: 1-4.DOI:10.1109/IDT.2009.5404081.

[8]Rais, M.H., 2010a. Hardware implementation of truncated multipliers using Spartan 3AN, Virtex-4and Virtex-5 devices. Am. J. Eng. Applied Sci., 3:201-206. DOI: 10.3844/ajeassp.2010.201.206.

[9]Pushpalata Verma, K. K. Mehta, " Implementation of an Efficient Multiplier based on Vedic Mathematics Using EDA Tool," International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-1, Issue-5, June 2012.

[10]Manoranjan Pradhan, Rutuparna Panda, Sushanta Kumar Sahu, "Speed Comparison of 16x16 Vedic Multipliers," International Journal of Computer Applications (0975 – 8887),Volume 21– No.6, May 2011.

[11]Naveen K Gahlan, Prabhat, Jasbir Kaur " Implementation of WallaceTree Multiplier Using Compressor" International Journal of Computer & Technology.

[12]W.J. Townsend, E.E. Swartzlander Jr., and J.A. Abraham, "A Comparison of Dadda and Wallace Multiplier Delays,"Proc.-SPIE, Advanced Signal Processing Algorithms, Architectures ,and Implementations XIII, pp. 552-560, 2003.

Author's Detail:

MS. T.BHAVANI. MTech student, in M.Tech Student, Dept of ECE in KITS for women's, kodad, T.S, India

MS.K.Anuradha. working as a Assistant at ECE in KITS for women's,kodad, T.S, India JNTUH Hyderabad. She has 5 years of UG/PG Teaching Experience.