The Novel High Speed Data Vedic Mathematics Multiplier by using Compressors

[1]MACHAVARAPU MOUNIKA [2]M SIVA DURGA

[1]M.Tech Student, [2]Assistant Professor

[1,2] Department Of Electronics & Communication, Sri Sunflower College of Engineering And Technology, Lankapalli(Challapalli), Krishna Dt, Andhra Pradesh

ABSTRACT:

High Performance of trending technology in the VLSI Field supports ongoing expectation for high speed processing and lower area consumption. Since multiplier unit forms basic part of a processor, high speed multiplication becomes a requirement. Here, a novel architecture capable of performing high speed multiplication with the help of vedic mathematics is discussed. Along with whcih 4:2 compressors and 7:2 compressors are used for the purpose of addition. When compared with existing methods, this type of multiplier using compressors is faster in performance. The designs are being experimented usign Xilinx spartan 3 e series using which simulated results of time and area of the novel architecture are being calculated. This project proposes the hardware implementation of vlsi architecture for high speed vlsi design of complex multiplier using vedic mathematics that have been modified to improve performance. This project was implemented in Verilog the coding is done in Verilog HDL and the FPGA synthesis is done using Xilinx Spartan library. The main advantage of implementation using hardware based high speed VLSI design of complex multiplier using vedic mathematics its inherent speed over software based methods.

KEYWORDS: Multiply and Accumulate, Urdhva-tiryakbyham, Nikhilam Sutra

1] INTRODUCTION:

Multiplication is an important fundamental function in arithmetic operations. Multiplication-based operations such as Multiply and Accumulate(MAC) and inner product are among some of the frequently used Computation- Intensive Arithmetic Functions(CIAF) currently implemented in many Digital Signal Processing (DSP) applications such as convolution, Fast Fourier Transform(FFT), filtering and in microprocessors in its arithmetic and logic unit. Since multiplication dominates the execution time of most DSP algorithms, so there is a need of high speed multiplier. Currently, multiplication time is still the dominant factor in determining the instruction cycle time gof a DSP chip.

The demand for high speed processing has been increasing as a result of expanding computer and signal processing applications. Higher throughput arithmetic operations are important to achieve the desired performance in many real-time signal and image processing applications. One of the key arithmetic operations in such applications is multiplication and the development of fast multiplier circuit has been a subject of interest over decades. Reducing the time delay and power consumption are very essential requirements for many applications. This work presents different multiplier architectures. Multiplier based on Vedic Mathematics is one of the fast and low power multiplier.

Minimizing power consumption for digital systems involves optimization at all levels of the design. This optimization includes the technology used to implement the digital circuits, the circuit style and topology, the architecture for implementing the circuits and at the highest level the algorithms that are being implemented. Digital multipliers are the most commonly used components in any digital circuit design. They are fast, reliable and efficient components that are utilized to implement any operation. Depending upon the arrangement of the components, there are different types of multipliers available. Particular multiplier architecture is chosen based on the application.

2] LITERATURE REVIEW From the rigorous review of related work and published literature it is observed that many researchers have designed different techniques for high speed communication in different techonologies. 1. S. He, and M. Torkelson: The design of efficient structure for pipelined bit-serial multiplication of complex numbers using distributed arithmetic under the condition of equal word length for both operands is presented. Equal word length operands can give maximum computation precision when the constraint for communication capability has been set, as in many realistic signal processors. With a different treatment of the residue error of the offset binary representation, the proposed structure has better AT/sup 2/ performance compared with existing approach. It also enables "safe" rounding and presents an exact two word length latency. Finally, multiple level logic optimization is applied in each of the building blocks, showing that there still can be significant improvement even on the so called "well-structured" designs.

2.R. Krishnan, G. A. Jullien Recently, the Quadratic Residue Number System (QRNS) has been introduced, which allows the multiplication of complex integers with two real multiplications. Restrictions on the form of the moduli can be removed if an increase in real multiplications from two to three can be tolerated; the resulting number system has been termed the Modified Quadratic Residue Number System (MQRNS). In this paper the MQRNS is defined, and residue to binary conversion techniques in both the QRNS and MQRNS are presented. Hardware implementations of non-recursive and recursive digital filters are also presented where the QRNS and MQRNS structures are realized using a bit-slice architectures.

3] PROPOSED TECHNIQUE:

Urdhva-tiryakbyham is applicable to all cases of multiplication, so we have chosen this sutra for implementation of vedic multiplier.

4] URDHVA-TIRYAKBYHAM SUTRA

The multiplier is based on an algorithm Urdhva Tiryakbhyam (Vertical & Crosswise) of ancient Indian Vedic Mathematics. Urdhva Tiryakbhyam Sutra is a general multiplication formula applicable to all cases of multiplication. It literally means "Vertically and crosswise". It is based on a novel concept through which the generation of all partial products can be done with the concurrent addition of these partial products. The parallelism in generation of partial products and their summation is obtained using Urdhava Triyakbhyam explained in fig 2.1. The algorithm can be generalized for n x n bit number. Since the pat1ial products and their sums are calculated in parallel, the multiplier is independent of the clock frequency of the processor. Thus the multiplier will require the same amount of time to calculate the product and hence is independent of the clock frequency. The net advantage is that it reduces the need of microprocessors to operate at increasingly high clock frequencies. While a higher clock frequency generally results in increased processing power, its disadvantage is that it also increases power dissipation which results in higher device operating temperatures. By adopting the Vedic multiplier, microprocessors designers can easily circumvent these problems to avoid catastrophic device failures.

5] Multiplication of two decimal numbers: 525 x 123

To illustrate this multiplication scheme, let us consider the multiplication of two decimal numbers (325 * 738). Line diagram for the multiplication is shown in Fig.2.2. The digits on the both sides of the line are multiplied and added with the carry from the previous step. This generates one of the bits of the result and a carry. This carry is added in the next step and hence the process goes on. If more than one line are there in one step, all the results are added to the previous carry. In each step, least significant bit acts as the result bit and all other bits act as carry for the next step. Initially the carry is taken to be zero. To make the methodology more clear, an alternate illustration is given with the help of line diagrams in figure 2.2 where the dots represent bit „0‟ or „1‟.

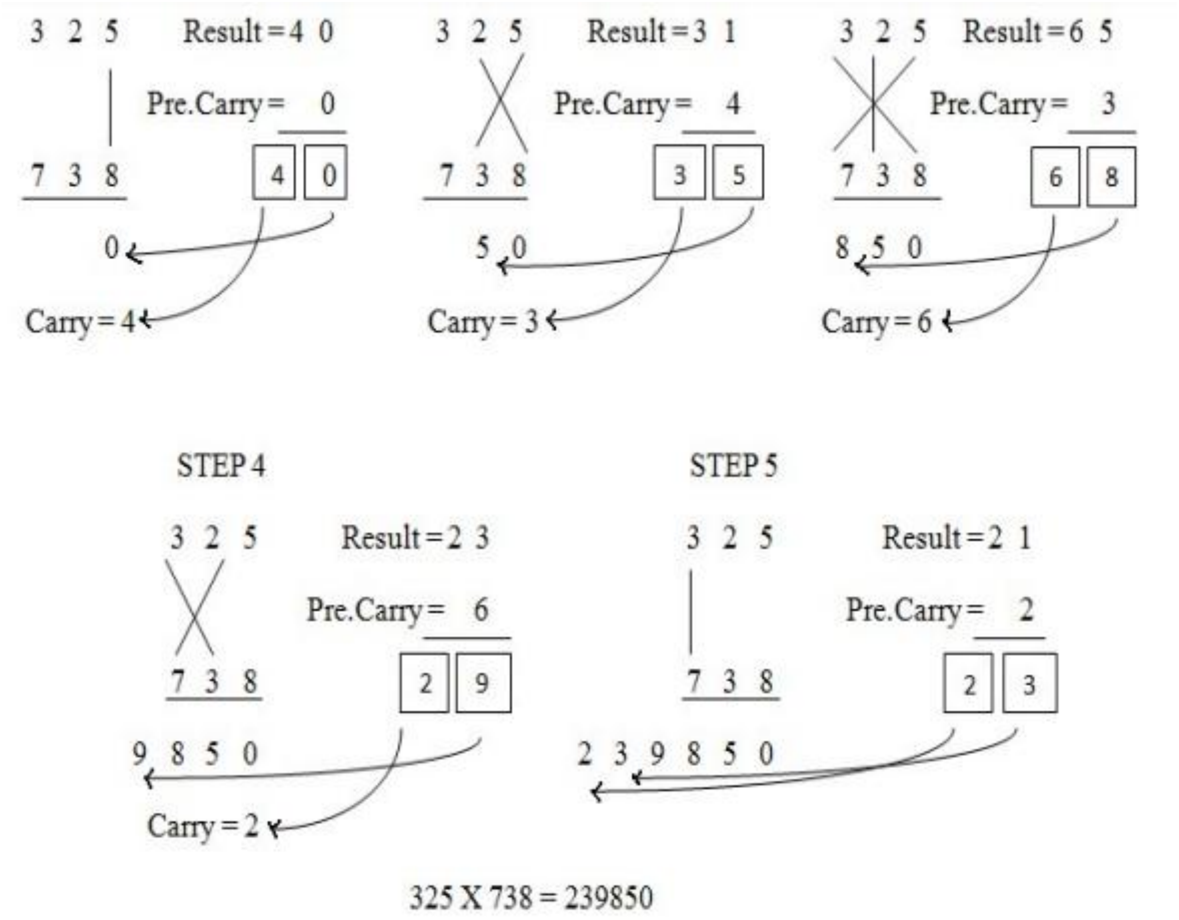


Fig 1. Example for UT method

Algorithm for 4 x 4 bit Vedic multiplier Using Urdhva Tiryakbhyam (Vertically and crosswise) for two Binary numbers

6] OFFSET COMPENSATED MILLER T/H:

CP = Cross Product (Vertically and Crosswise)
X3 X2 X1 X0 Multiplicand Y3 Y2 Y1 Y0 Multiplier

H G F E D C B A

P7 P6 P5 P4 P3 P2 P1 P0 Product

PARALLEL COMPUTATION METHODOLOGY

1. CP X0 = X0 * Y0 = A

Y0

2. CP X1 X0 = X1 * Y0+X0 * Y1 = B

Y1 Y0

3. CP X2 X1 X0 = X2 * Y0 +X0 * Y2 +X1 * Y1 = C

Y2 Y1 Y0

4. CP X3 X2 X1 X0 = X3 * Y0 +X0 * Y3+X2 * Y1 +X1 * Y2 = D

Y3 Y2 Y1 Y0

5. CP X3 X2 X1 = X3 * Y1+X1 * Y3+X2 * Y2 = E

Y3 Y2 Y1

6. CP X3 X2 = X3 * Y2+X2 * Y3 = F

Y3 Y2 7 CP X3 = X3 * Y3 = G

Y3

7] NIKHILAM SUTRA

Nikhilam Sutra literally means “all from 9 and last from 10”. Although it is applicable to all cases of multiplication, it is more efficient when the numbers involved are large. Since it finds out the compliment of the large number from its nearest base to perform the multiplication operation on it, larger is the original number, lesser the complexity of the multiplication. We first illustrate this Sutra by considering the multiplication of two decimal numbers (96 * 93) where the chosen base is 100 which is nearest to and greater than both these two numbers.

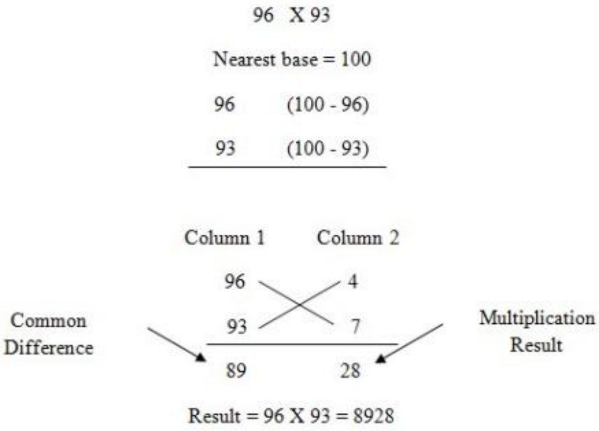


Figure 2 Multiplication Using Nikhilam Sutra

8] SQUARE ALGORITHM

In order to calculate the square of a number, we have utilized “Duplex” D property of Urdhva Triyakbhyam. In the Duplex, we take twice the product of the outermost pair and then add twice the product of the next outermost pair and so on till no pairs are left. When there are odd numbers of bits in the original sequence, there is one bit left by itself in the middle and this enters as its square. Thus for 987654321,

D= 2 * (9 * 1) + 2 * (8 * 2) + 2 * (7 * 3) + 2 * (6 * 4) + 5 * 5 = 165.

Further, the Duplex can be explained as follows

For a 1 bit number D is its square.

For a 2 bit number D is twice their product

For a 3 bit number D is twice the product of the outer pair + square of the middle bit.

For a 4 bit number D is twice the product of the outer pair + twice the product of the inner pair.

The algorithm is explained for 4 x 4 bit number. The Vedic square has all the advantages as it is quite faster and smaller than the array, Booth and Vedic multiplier [10, 13].

9] CUBE ALGORITHM:

The cube of the number is based on the Anurupye Sutra of Vedic Mathematics which states “If you start with the cube of the first digit and take the next three numbers (in the top row) in a Geometrical Proportion (in the ratio of the original digits themselves) then you will find that the 4th figure (on the right end) is just the cube of the second digit”.

If a and b are two digits then according to Anurupye Sutra,

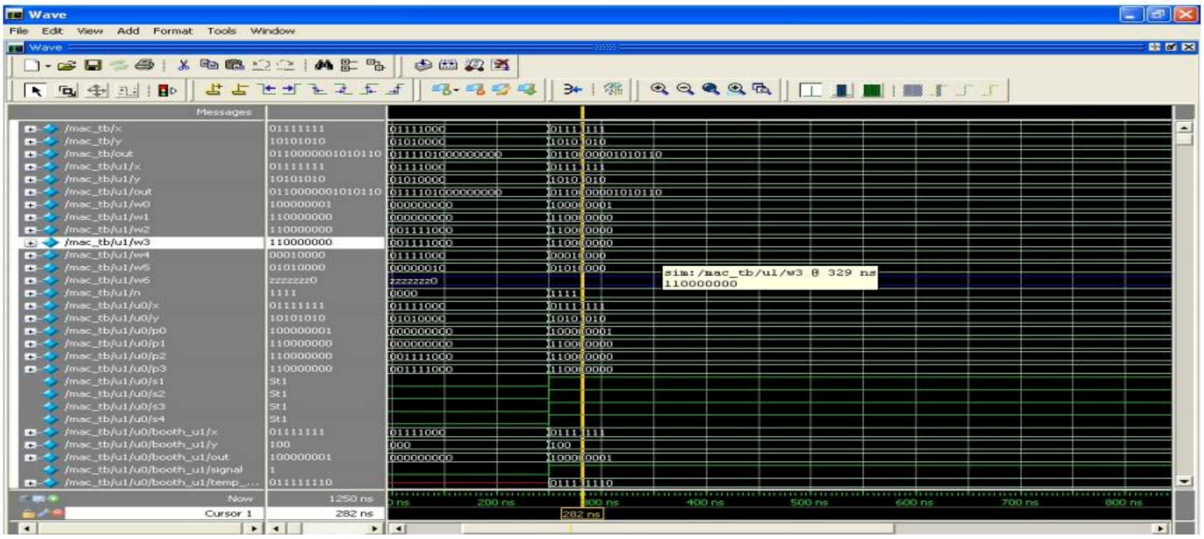
$$\begin{array}{ccccc} a^3 & a^2b & ab^2 & b^3 & \\ & 2a^2b & 2ab^2 & & \\ \hline a^3 + & 3a^2b + & 3ab^2 + & b^3 & =(a+b)^3 \\ \hline \end{array}$$

This sutra has been utilized in this work to find the cube of a number. The number M of N bits having its cube to be calculated is divided in two partitions of N/2 bits, say a and b, and then the Anurupye Sutra is applied to find the cube of the number. In the above algebraic explanation of the Anurupye Sutra, we have seen that a3 and b3 are to be calculated in the final computation of (a+b) 3. The intermediate a3 and b3 can be calculated by recursively applying Anurupye sutra. A few illustrations of working of Anurupye sutra are given below [10].

$$\begin{array}{lcccc} (15)^3 = & 1 & 5 & 25 & 125 \\ & & 10 & 50 & \\ \hline & 3 & 3 & 7 & 5 \\ \hline \end{array}$$

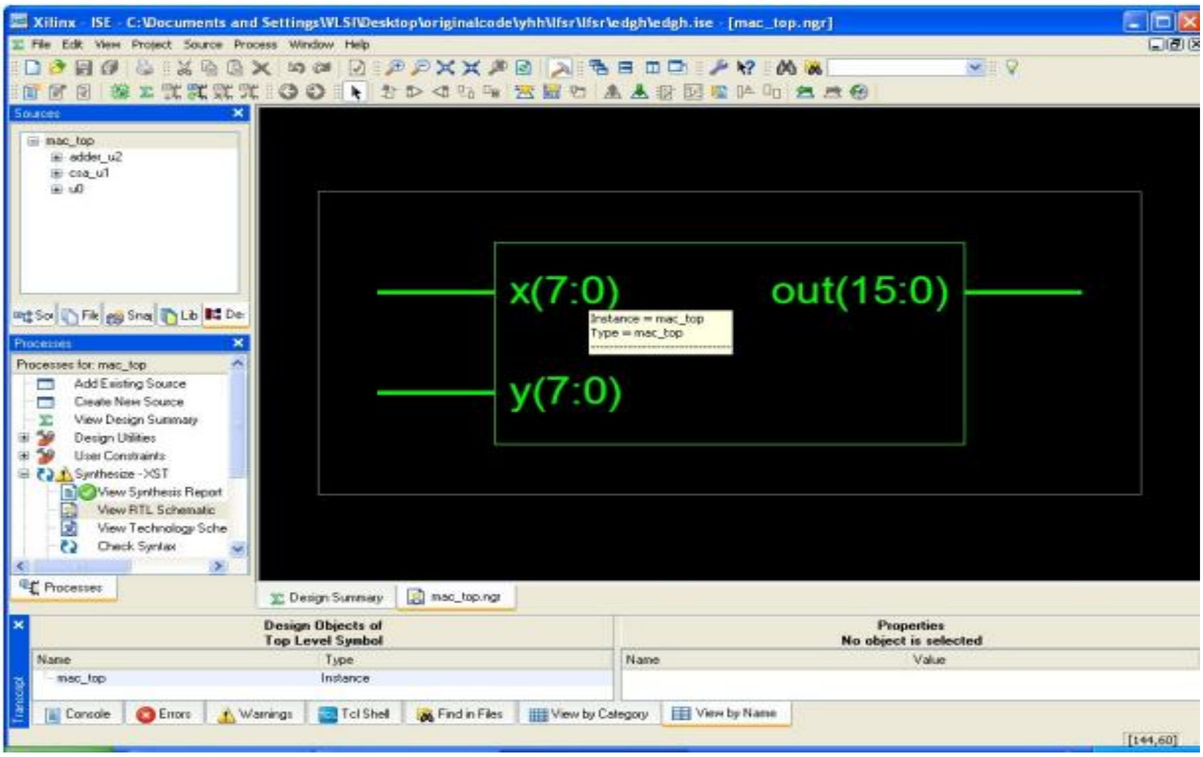
Thus the result shows that the Vedic square multiplier is smallest and the fastest of the reviewed architectures. The Vedic square and cube architecture proved to exhibit improved efficiency in terms of speed and area compared to Booth and Array Multiplier. Due to its parallel and regular structure, this architecture can be easily realized on silicon and can work at high speed without increasing the clock frequency. It has the advantage that as the number of bits increases, the gate delay and area increase very slowly as compared to the square and cube architectures of other multiplier architecture. Speed improvements are gained by parallelizing the generation of partial products with their concurrent summations. It is demonstrated that this design is quite efficient in terms of silicon area/speed. Such a design should enable substantial savings of resources in the FPGA when used for image/video processing application

10] RESULT:



Messages	
/mac_tb/x	01111111
/mac_tb/y	10101010
/mac_tb/out	0110000000101010110
/mac_tb/u1/x	01111111
/mac_tb/u1/y	10101010
/mac_tb/u1/out	0110000000101010110
/mac_tb/u1/w0	100000001
/mac_tb/u1/w1	110000000
/mac_tb/u1/w2	110000000
/mac_tb/u1/w3	110000000
/mac_tb/u1/w4	00010000
/mac_tb/u1/w5	01010000
/mac_tb/u1/w6	zzzzzzz0
/mac_tb/u1/n	1111
/mac_tb/u1/u0/x	01111111
/mac_tb/u1/u0/y	10101010
/mac_tb/u1/u0/p0	100000001
/mac_tb/u1/u0/p1	110000000
/mac_tb/u1/u0/p2	110000000
/mac_tb/u1/u0/p3	110000000
/mac_tb/u1/u0/s1	St1
/mac_tb/u1/u0/s2	St1
/mac_tb/u1/u0/s3	St1
/mac_tb/u1/u0/s4	St1
/mac_tb/u1/u0/booth_u1/x	01111111
/mac_tb/u1/u0/booth_u1/y	100
/mac_tb/u1/u0/booth_u1/out	100000001
/mac_tb/u1/u0/booth_u1/signal	1
/mac_tb/u1/u0/booth_u1/temp_...	011111110
Now	1250 ns
Cursor 1	282 ns

Simulation Results



Xilinx - ISE - C:\Documents and Settings\VLSI\Desktop\originalcode\yhh\lfsr\lfsr\edgh\edgh.ise - [mac_top.ngr]

Instance = mac_top
Type = mac_top

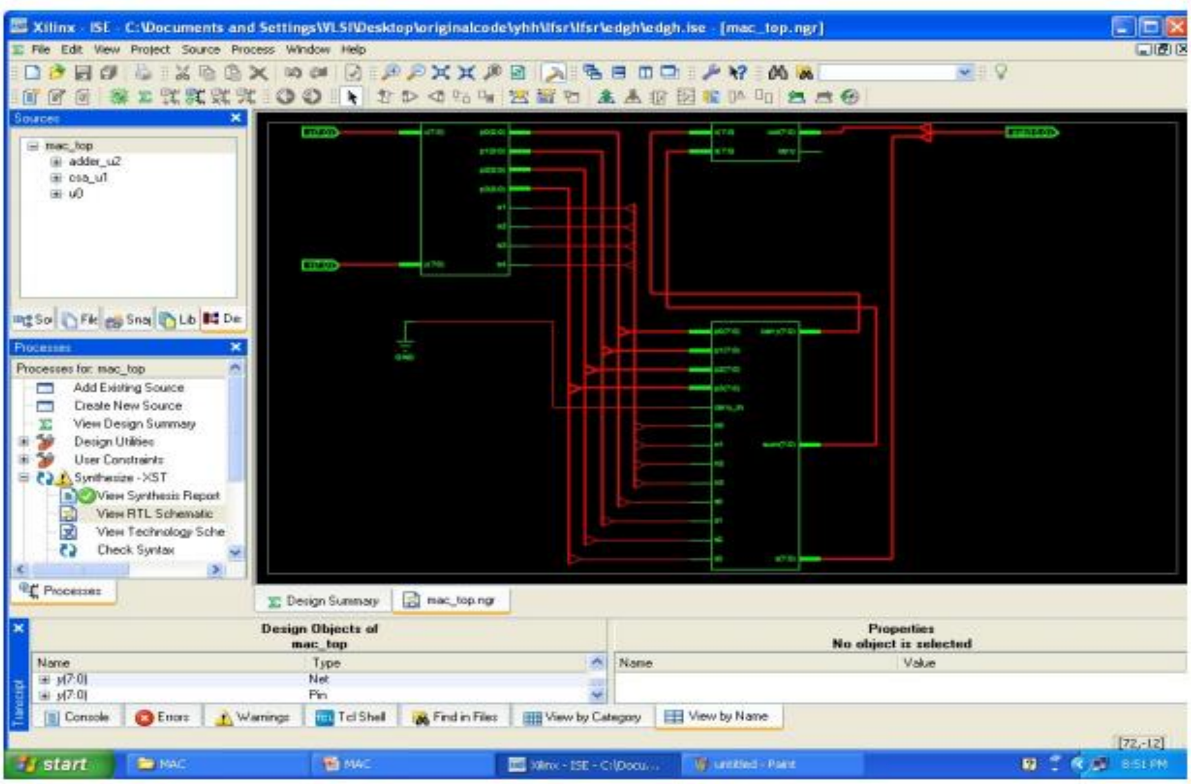
Design Objects of Top Level Symbol

Name	Type
mac_top	Instance

Properties
No object is selected

Name	Value

SYNTHESIS RESULTS



Xilinx - ISE - C:\Documents and Settings\VLSI\Desktop\originalcode\yhh\lfsr\lfsr\edgh\edgh.ise - [mac_top.ngr]

Design Objects of mac_top

Name	Type
y(7:0)	Net
y(7:0)	Pin

Properties
No object is selected

Name	Value

Design Summary

11] CONCLUSION

An 8x8 multiplier-accumulator (MAC) is presented in this work. A Radix 4 Modified Booth multiplier circuit is used for MAC architecture. Compared to other circuits, the Booth multiplier has the highest operational speed and less hardware count. The basic building blocks for the MAC unit are identified and each of the blocks is analyzed for its performance. Power and delay is calculated for the blocks. 1-bit MAC unit is designed with enable to reduce the total power consumption based on block enable technique. The MAC unit designed in this work can be used in filter realizations for High speed DSP applications.

12] REFERENCES

[I] P. K. Saha, A. Banerjee, and A. Dandapat, "High Speed Low Power Complex Multiplier Design Using Parallel Adders and Subtractors," International Journal on Electronic and Electrical Engineering, (/JEEE), vol 07, no. II, pp 38-46, Dec. 2009.

[2] R. E. Blahut, Fast Algorithms for Digital Signal Processing, Reading, MA: Addison-Wesley, 1987.

[3] S. He, and M. Torkelson, "A pipelined bit-serial complex multiplier using distributed arithmetic," in proceedings IEEE International Symposium on Circuits and Systems, Seattle, W A, April 30-May-03, 1995,pp. 2313-2316.

[4] J. E. VoIder, "The CORDIC trigonometric computing technique," IRE Trans. Electron. Comput., vol. EC-8, pp. 330-334, Sept. 1959.

[5] R. Krishnan, G. A. Jullien, and W. C. Miller, "Complex digital signal processing using quadratic residue number systems," IEEE Trans. Acoust., Speech, Signal Processing, vol. 34, Feb. 1985.

[6] T. Aoki, K. Hoshi, and T. Higuchi, "Redundant complex arithmetic and its application to complex multiplier design," in Proceedings 29th IEEE International symposium on MultipleValued-Logic, Freiburg, May 20- 22,1999, pp. 200-207.

[7] Z. Huang, and M. D. Ercegovac, "High-Performance Low-Power Leftto- Right Array Multiplier Design," IEEE Transactions on Computers, vol 54, no. 3, pp 272-283, March 2005.

[8] S. Y. Kung, H. J. Whitehouse, and T. Kailath, VLSI and Modern Signal Processing, Englewood Cliffs, NJ: Prentice-Hall, 1985.

[9] P. Mehta, and D. Gawali, "Conventional versus Vedic mathematical method for Hardware implementation of a multiplier," in Proceedings IEEE International Cotiference on Advances in Computing, Control, and Telecommunication Technologies, Trivandrum, Kerala, Dec. 28-29, 2009, pp. 640-642.