Design and Simulation of Multiplier for High-speed Application

Amit Kumar Yadav[1], Satyendra Sharma[2]
[1,2] *Department of Electronics & Communication Engineering*
[1,2] *Noida Institute of Engg & Tech. Greater Noida*
[1]shivaamit88@gmail.com
[2]satyendracomn@gmail.com

***Abstract*—This paper describes an efficient implementation of high speed multiplier at the algorithm and architecture level, addresses Low-Power, High Speed and Less Area multiplier design systematically from two aspects: internal efforts considering multiplier architectures and external efforts considering input data characteristics. We can achieved High-throughput rate by a new architecture implementing our earlier multiplication technique in conventional register pipelining at the bit level. The multiplier is designed by using Xilinx 14.7 for its synthesis result and Modelsim simulator is used for simulation. In this paper we present a study of Booth Multiplier for Area, Power and Speed in VLSI design of 8 bit Multipliers.**

***Index Terms*- Adder, Array Multiplier, Booth Encoder, FIR Filter.**

I. INTRODUCTION

An Arithmetic and logical unit play an important role in digital systems. Particularly Multiplication is very relevant instead of other arithmetic operators, such as division or exponentiation, which one is possible by multiplier as building blocks. Multipliers are the slowest device; to achieve faster computation Booth's algorithms used in which quick and accurate performance of multiplier operation has been done. These algorithms provide high performance rather than other multiplication algorithms. Multipliers are key components of many high performance systems such as FIR filter, microprocessors, digital signal processors, etc. [18] A system's

Performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system [1]. Furthermore, it is generally the most area consuming [2]. Hence, optimizing the speed and area of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas. As a result, a whole spectrum of Multipliers with different area and speed constraints has been designed. Parallel Multipliers at one end of the spectrum and fully serial multipliers at the other end, in between are digital serial multipliers where single digits consisting of several bits are operated on. These multipliers have moderate performance in both speed and area. However, existing digital serial multipliers have been plagued by complicated switching systems and irregularities in design. Radix- 2n [5] multipliers which operate on digits in a parallel fashion instead of bits bring the pipelining to the digit level and avoid most of the above problems. They were introduced by M. K. Ibrahim in 1993[22].

II. BOOTH'S RECODING ALGORITHM

Parallel Multiplication using basic Booth's Recoding algorithm technique based on the fact that partial product can be generated for group of consecutive 0's and 1's which is called as Booth's recoding. These Booth's Recoding algorithm is used to generate efficient partial product. These Partial Products always have large number of bits than the input number of bits.

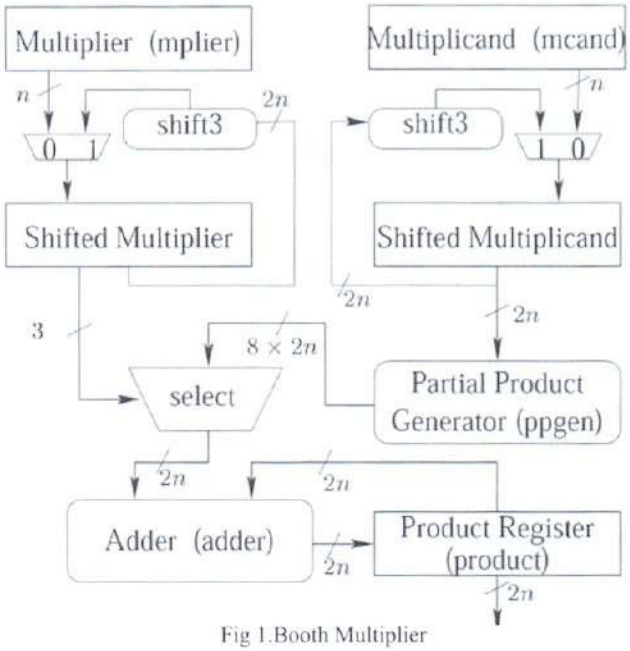


Fig 1.Booth Multiplier

This width of partial product usually depends upon the radix scheme used for recoding. So, these scheme uses less partial products which comprises low power an There are two types of algorithm Radix-2 and Radix-4 to generate efficient partial products for multiplication using Booth algorithm.

III. ALGORITHM FOR RADIX-2

Booth has proposed Radix algorithm for high speed multiplication which reduces partial products for multiplication. The Booth's algorithm for multiplication is based on this observation. To do a multiplication A*B, where

A= an, an-1.....a0 is a multiplier

B= bn, bn-1.....b0 is a multiplicand

then, we check every two consecutive bits in A at a time.

Radix-2:

Suppose A is Multiplier having value -5 and B is Multiplicand having value +2 then,

B=> 0010 (+2)

A=> 1011 (-5)

After looking into above table for multiplicand, first we see two LSB values and then adjacent values in A. We get partial product as:-

i) For 10 we have to perform -1.B, i.e., 2's complement of B, 1110.

ii) For 11 we have to put all 0's i.e., 0000.

iii) For 01 we have to perform 1.B, i.e., value of B,0010

iv) For 10 again -1.B, i.e. 1110.

Here, some bits are encapsulated called as correction bits to match the width of partial products.

Radix-4 Recoding:

For radix-4 recoding, the popular algorithm is parallel recoding or Modified Booth recoding [21]. In parallel radix-4 recoding, Y becomes:

$$Y = \sum_{i=0}^{n/2-1} v_i 4^i = \sum_{i=0}^{n/2-1} (-2y_{2i+1} + y_{2i} + y_{2i-1})4^i \quad (1)$$

Where y-1=0 and $v_i \in \{2,1,0,1,2\}$. As all bits are recorded in parallel without data dependency, the recorded digits are available simultaneously

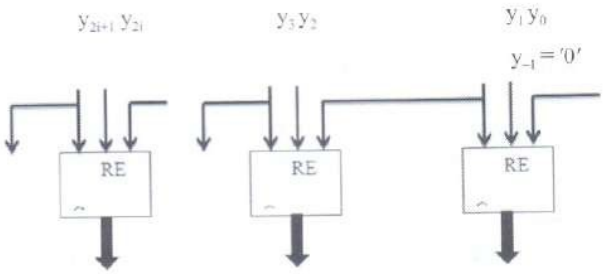


Fig 2. Parallel Radix-4 Recoding

IV. BOOTH MULTIPLIER

Radix 4 Booth Multiplier consists of the following components

1.Adders

2.Array Multiplier

3.Booth Multiplier

4.Booth Encoder

5.FIR Filter (Finite Impulse Response Filter)

Adders

An adder is a digital circuit that performs addition of numbers. In modern computers adders reside in the arithmetic logic unit (ALU) where other operations are performed. Although adders can be constructed for many numerical representations, such as Binary-coded decimal or excess-3, the most common adders operate on binary numbers. In case where two's complement is being used to represent negative numbers it is trivial to modify an adder into an adder-sub tractor.

Type of Adders

For addition of two single bit numbers adder's half adder can be used. A half adder has two inputs, generally labeled A and B, and two outputs, the sum S and carry C. S is the two-bit XOR of A and B, and C is the AND of A and B. Essentially the output of a half adder is the sum of two one-bit numbers, with C being the most significant of these two outputs.

The second type of single bit adder is the full adder. The full adder takes into account a carry input such that multiple adders can be used to add larger numbers. To remove ambiguity between the input and output carry lines, the carry in is labelled Ci or Cin while the carry out is labelled Co or Cout.

Half adder:

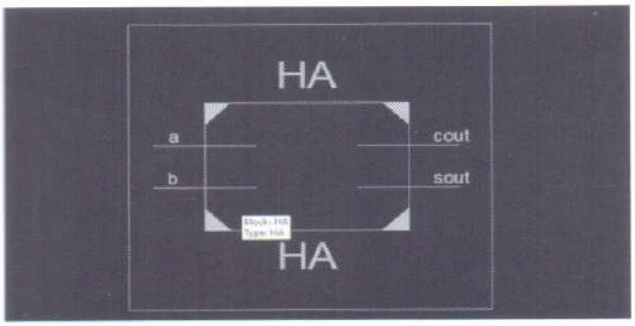


Fig 3. Half Adder (RTL) Circuit Diagram

A half adder is a logical circuit that performs an addition operation on two binary digits.

The half adder produces a sum and a carry value which are both binary digits.

$S = A \oplus B$

$C = A \,.\, B$

V. BASICS OF MULTIPLICATION

Binary Multiplier:

A Binary multiplier is an electronic hardware device used in digital electronics or a computer or other electronic device to perform rapid multiplication of two numbers in binary representation. It is built using binary adders.

The rules for binary multiplication can be stated as follows:

1. If the multiplier digit is a 1, the multiplicand is simply copied down and represents the product.
2. If the multiplier digit is a 0 the product is also 0.

For designing a multiplier circuit we should have circuitry to provide or do the following three things:

1. It should be capable of identifying whether a 0 or 1.
2. It should be capable of shifting left partial products.
3. It should be able to add all the partial products to give the products as sum of partial products.
4. It should examine the sign bits. If they are alike, the sign of the product will be a positive, if the sign bits are opposite product will be negative.

It is concluded that it is not necessary to wait until all the partial products have been formed before summing them. In fact the addition of partial product can be carried out as soon as the partial product is formed.

Notations:

a – multiplicand
b – multiplier
p – Product
Binary multiplication (e.g. n=4)

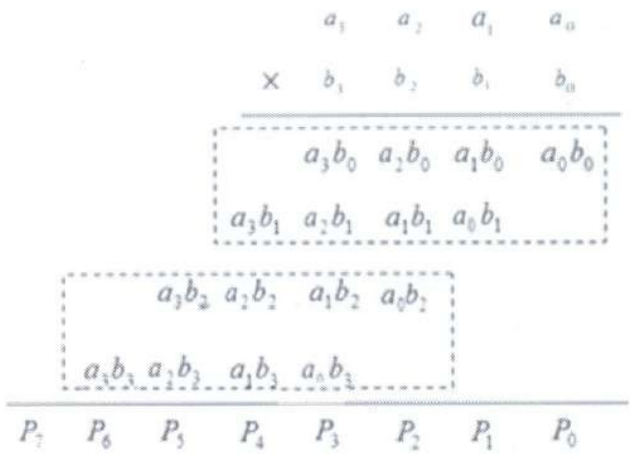


Fig. 4: Basic Binary Multiplication

The left example shows the multiplication procedure of two unsigned binary digits while the one on the right is for signed multiplication. The first digit is called Multiplicand and the second Multiplier. The only difference between signed and unsigned multiplication is that we have to extend the sign bit in the case of signed one, as depicted in the given right example in PP row 3. Based upon the above procedure, we can deduce an algorithm for any kind of multiplication which is shown in Figure 2.5. Here, we assume that the MSB represents the sign of digit.

Partial Product Generator:

In Unsigned Multipliers, normally we are generating partial products and adding them to generate result of multiplier. Let 'A' and 'B' are two n-bit unsigned numbers which is generating product 'Z' which is of 2n-bit. First we are generating Partial products by using 'AND' operation. For n bit multiplication n*n number of partial product are

generated. Let us take two 8-bit numbers A7-A0 called Multiplicand and B7-B0 called Multiplier as inputs of multiplier, partial products are generated by AND operation of each bit of 'A' with each bit of 'B', so 8*8=64 number of partial products are generated. Each bit of multiplicand is generated by performing AND operation with every bit of multiplicand a0 and again AND operation with b0-b7 producing m00-m07 8 partial product for first row. Similarly, for other rows we are using AND operation of a1-a7 with b0-b7 for producing other remaining partial products i.e. from m01-m7.

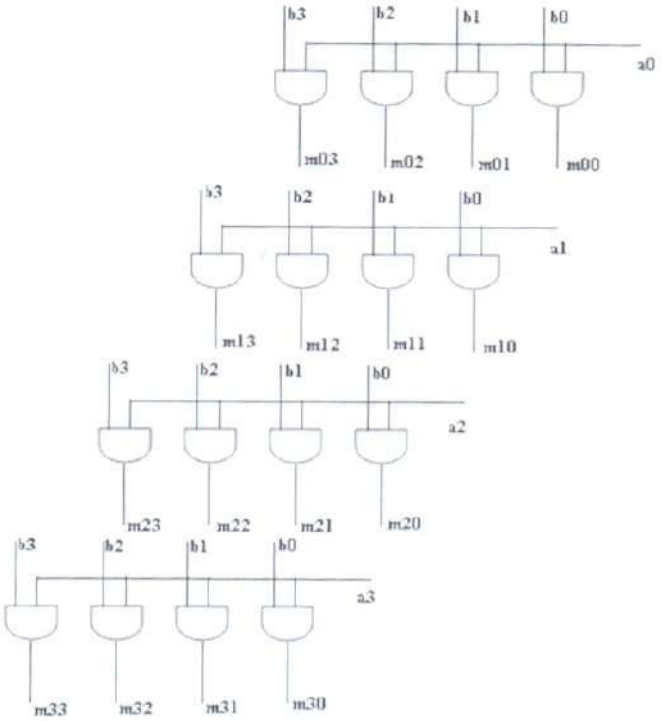


Fig. 5: Partial Product Generator (4 Bit)

Modified Booth Encoder (MBE):

Modified Booth encoding is most often used to avoid variable size partial product arrays. Before designing a MBE, the multiplier B has to be converted into a Radix-4 number by dividing them into three digits respectively according to Booth Encoder Table given afterwards. Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) of the multiplier. The figure above shows that the multiplier has been divided into four partitions and hence that mean four partial products will be generated using booth multiplier approach Instead of eight partial products being generated using conventional multiplier.

$$Z_n = -2^* B_{n+1} + B_n + B_{n-1}$$

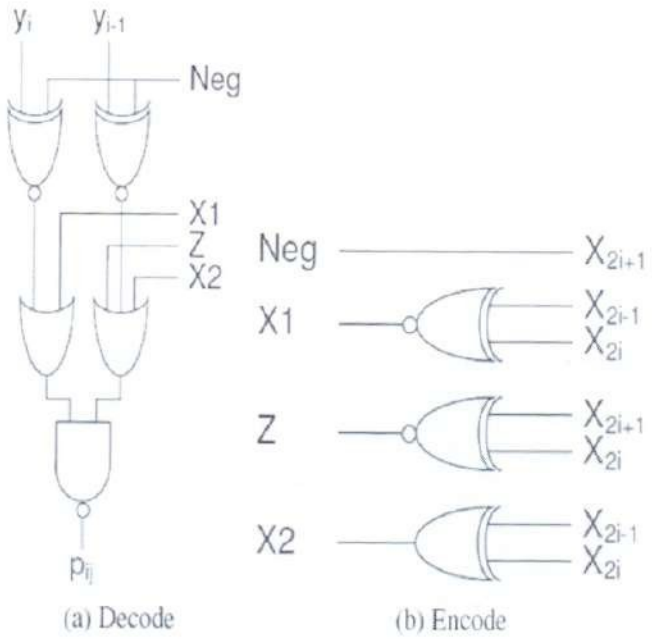


Fig. 6: Encode and Decode Circuit for Modified Booth Encoder

$$Z_n = -2^* B_{n+1} + B_n + B_{n-1}$$

Array Multiplier:

New multiplication algorithm based on a different mechanism has been proposed by Dr. Pekmestzi (National Technical University of Athens)•Algorithm is symmetric because at each step one bit of the multiplier and one bit of the multiplicand are processed. Mathematically, this method can be described: Consider two positive integer numbers, X and Y:

$$X = X_{n-1} X_{n-2} \ldots X_0 = \sum_{j=0}^{n-1} X_j 2^j \qquad 2.1$$

$$Y = y_{n-1} y_{n-2} \cdots y_0 = \sum_{j=0}^{n-1} y_j 2^j \qquad 2.2$$

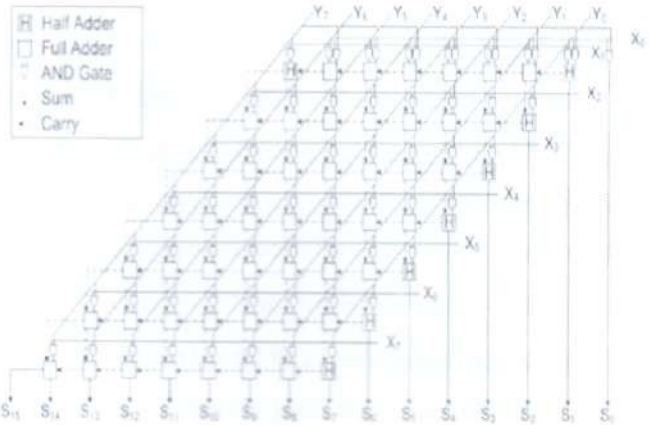


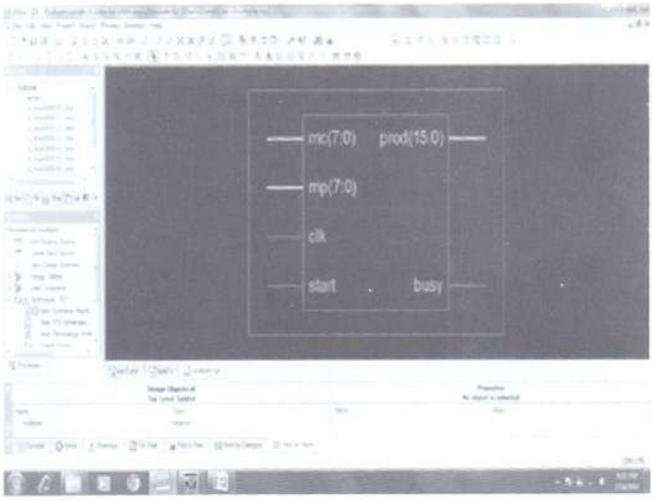
Fig. 7: Block Diagram of an Unsigned 8-Bit Array Multiplier

In Array Multiplier, consider two binary numbers A and B, of m and n bits. There are mn summands that are produced in parallel by a set of mn AND gates. n x n multiplier requires n (n-2) full adders, n half-adders and n2 AND gates. Also, in Array Multiplier worst case delay would be (2n+1) td.

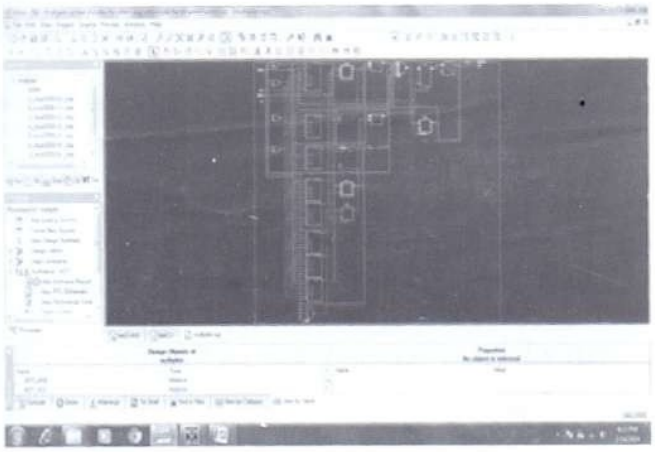
Two conventional array multipliers can be used to generate partial products. According to the way of carry propagation, it can be classified into two structures: ripple-carry array (RCA) and carry-save array (CSA). In RCA multiplier all adder cells are composed of RCA adders. For example, it needs 3N adders to accomplish multiplication in an N × N multiplier. However, delay time needed in the worst case is (2N + 1) full adder delay.

VI. SYNTHESIS & SIMULATION RESULTS

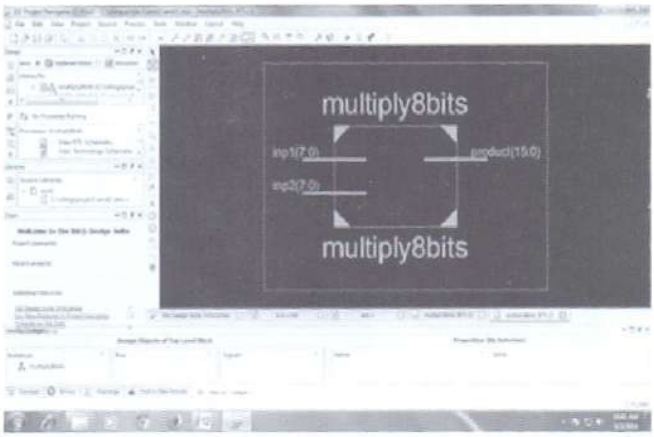
RTL View of Booth Multiplies



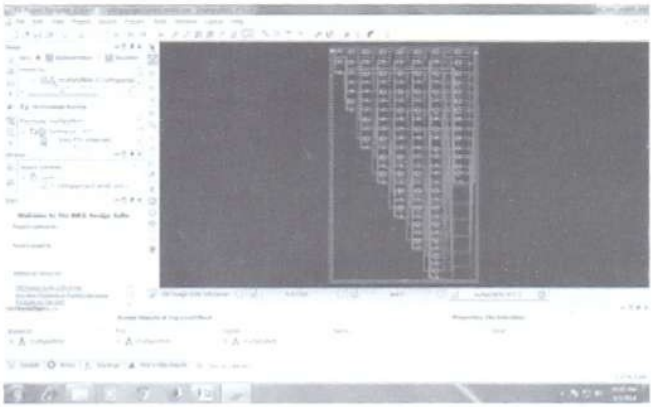
Internal RTL View of 8x8 Booth Multiplier:



RTL View of 8x8bit Array Multiplier:



Internal RTL View of 8x8bit Array Multiplier:



Simulation result:

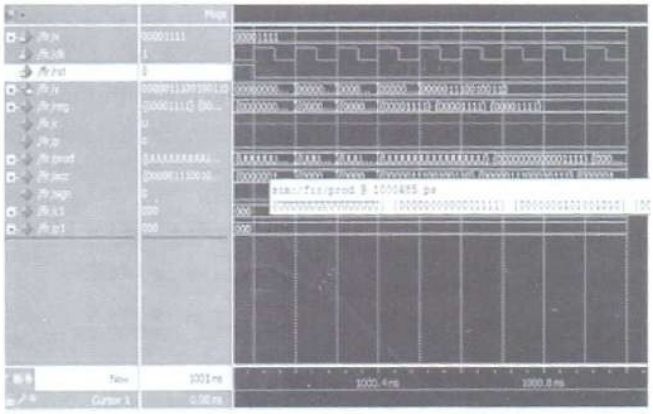


Fig. 8: Simulation Result of 8X8 Booth Multiplier (A)

This result shows simulation result for an unsigned number when we are giving the input as an 8 bit number 00001111. The clock applied is one. The output is a 16 bit data.

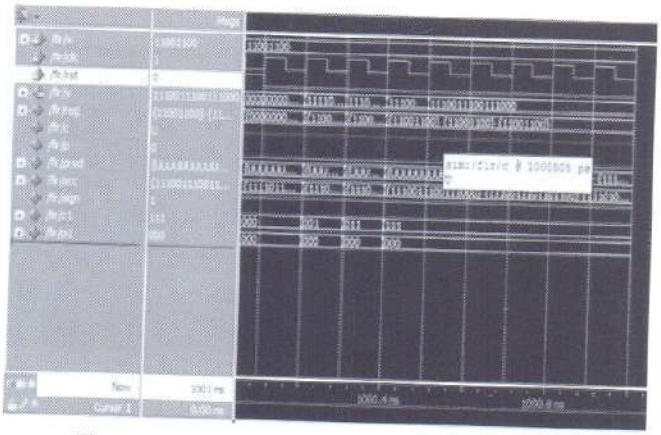


Fig. 9: Simulation Result of 8X8 Booth Multiplier (B)

The above simulation result shows the 8x8 multiplication of a signed number. Here the input is an 8 bit signed number clock one is applied. Here the output is a 16 bit number as shown in the figure.

Comparison Summary for multiplier:

PARAMETER	Array Multiplier	Booth Multiplier
Number of 4 input LUTs	129	40
Number of occupied Slices	68	21
Total Power (μW)	63.331	35.521
Cell Area	2499	2234

VII. CONCLUSION

Comparison summary for the Array Multiplier and Booth Multiplier from table it can be observed that Booth Multiplier have small number of LUTs, Slices and less Cell Area, and low power consumption. Array Multiplier, but due to its higher complexity, its layout is complex.

Hence for small area requirement and for less delay requirement Booth's Multiplier is suggested.

REFERENCES

[1] A. Dandapat, S. Ghosal, P. Sarkar, D. Mukhopadhyay (2009), "A 1.2-Ns16×16-Bit Binary Multiplier Using. High Speed Compressors", International Journal Of Electrical, Computer, And Systems Engineering, 2009, 234-239.

[2] Gu, C.H.Chang (2003), "Ultra Low Voltage Low Power 4-2 Compressor For High Speed Multiplications", Circuits And Systems, 2003.Iscas '03. Proceedings Of The International Symposium, Vol. 5, May 2003, 321-324.

[3] M.D. Ercegovac And T. Lang, Digital Arithmetic, Morgan Kaufmann Publishers, Elsevier Science Ltd., 2003.

[4] H.-C. Chow And I-C.Wey, "A 3.3v 1ghz High Speed Pipelined Booth Multiplier," In Proc. 2002 Int. Symp. Circuits And Systems, Vol.1, pp.457-460, May 2002.

[5] Israel Koren, Computer Arithmatics Algorithms A.K.Peters Ltd. ISBN 1568811608., Dec 2001

[6] P.-M. Seidel, L.D. Mcfearin, And D.W. Matula, "Binary Multiplication Radix-32 And Radix-256," In Proc. 15th IEEE Symp. Computer Arithmetic, pp.23-32, June 2001.

[7] K. Choi And M. Song, "Design Of A High Performance 32*32-Bit Multiplier With A Novel Sign Select Booth Encoder," In Proc. 2001 IEEE Int. Symp. Circuits And Systems, Vol.2, pp.701-704, May 2001.

[8] L.-H. Chen, W.-L. Liu, And O.T.-C. Chen, "Determination Of Radix Numbers Of The Booth Algorithm For The Optimized Programmable Fir Architecture," In Proc. 2000 IEEE Int. Symp. Circuits And Systems, Vol.2, pp.345-348, May 2000.

[9] W.-C. Yeh And C.-W. Jen, "High-Speed Booth Encoded Parallel Multiplier Design," IEEE Trans. Comput., Vol.49, No.7, pp.692-701, July 2000.

[10] C.H.Chang, J.Gu, M.Zhang (2004) ,"Ultra Low-Voltage Low-Power Cmos 4-2 And 5-2 Compressors For Fast Arithmetic Circuits", Circuits And Systems Regular Papers, IEEE Transactions Page(S): 1985- 1997, Volume: 51, Issue: 10, Oct.

[11] T.K. Callaway And E.E. Swartzlander, Jr., "Power-Delay Characteristics of Cmos Multipliers," In Proc. 13th IEEE Int. Symp. Computer Arithmetic, pp.26-32, 1997.

[12] G. Goto, Et. Al., "A 4.1-Ns Compact 54*54-B Multiplier Utilizing Sign-Select Booth Encoders," IEEE J. Solid-State Circuits, Vol.32, pp.1676-1682, Nov.1997.

[13] R. Fried, "Minimizing Energy Dissipation In High-Speed Multipliers," In Proc. 1997 Int. Symp. Low Power Electronics And Design, pp.214-219, Aug 1997.

[14] E. De Angel And E.E. Swartzlander, Jr., "Low Power Parallel Multipliers," In VLSI Signal Processing, Ix, pp.199-208, Oct. 1996.

[15] N. Ohkubo, Et. Al., "A 4.4 Ns Cmos 54*54-Bmultiplier Using Pass-Transistor Multiplexer," IEEE J. Solid-State Circuits, Vol.30, pp.251-257, March 1995.

[16] N.H.E. Weste, K. Eshraghian, Principles Of Cmos VLSI Design. Addison-Wesley Publishing Company, 1993.

[17] John G.Proakis, Dimitris G. Monolakis, "Digital Signal Processing, Principles, Algorithms, And Applications", Fourth Edition., 1992

[18] P.J. Song And G. De Micheli, "Circuit And Architecture Trade-Offs For Highspeed Multiplication," IEEE J. Solid-State Circuits, Vol.26, pp.1184-1198,Sept. 1991.

[19] C.S. Wallace, "A Suggestion For A Fast Multiplier," IEEE Trans. Electronic Computers, Vol.Ec-13, pp.14-17, Feb. 1964.

[20] O.L. Macsorley, "High-Speed Arithmetic In Binary Computers," IRE Proceedings, Vol.49, pp.67-91, 1961.

[21] A.D.Booth, A Signed Binary Multiplication Technique, Quarterly Journal Of Mechanics And Applied Mathematics, Vol-IV,Pt-2-1951.

Amit Kumar Yadav did his B.Tech. from IILM, Gr. Noida and at present he is pursuing his M.Tech. degree in VLSI from NIET Greater Noida.

Satyendra Sharma has received B.E and M.Tech degree in Electronics and Communication Engineering and VLSI Design from Institution of Engineers (India), Calcutta and Uttar Pradesh Technical University, Lucknow. He is pursuing Ph.D from NIT Kurukshetra. He has served 20 years in Indian Air Force for installation, operation and maintenance for various Radar and the Communication equipments. Presently, he is working as Associate Professor in Noida Institute of Engineering & Technology and heading the department of ECE.

Mr. Sharma has published 23 papers in scientific journals and conferences in the field of wireless communication and VLSI design. He has guided many projects for UG/PG programs, but the exceptions are 'Moon Buggy' for NASA(USA), Ultrasonic Radar for terror detection' and Digital Stick for physically challenged personnel etc. He is a life member of ISTE and AMIE.