Compare Efficiency of Different Multipliers Using Verilog Simulation & Modify an Efficient Multiplier

Article · April 2017

CITATIONS
3

READS
2,045

2 authors:

Abhishek Bhattacharjee
National Institute of Technology, Agartala
5 PUBLICATIONS **13** CITATIONS

SEE PROFILE

Anindya Sen
Heritage Institute of Technology
54 PUBLICATIONS **308** CITATIONS

SEE PROFILE

Compare Efficiency of Different Multipliers Using Verilog Simulation & Modify an Efficient Multiplier

Abhishek Bhattacharjee[1], Anindya Sen[2]

[1,2] *Department of Electronics and Communication Engineering (VLSI), Heritage Institute of Technology, Kolkata, West Bengal*

***Abstract*- Multipliers are the fundamental components in many digital signal processing systems. Many important signal processing systems are designed on VLSI platform as the integration growing rapidly. The signal processing systems & applications requires large computational capability, hence takes considerable amount of energy. In the VLSI system design performance, area and power consumption are three important parameters, of which power consumption is the gets prime importance. In today's world, power consumption is very important factor. The largest contribution to the power consumption in multiplier is due to generation and reduction of partial products. So it is very much important to know the efficiency of different multipliers. This paper represents a detailed comparison between array multiplier, Wallace multiplier, Dadda multiplier, modified array multiplier on the basis of speed, area, power consumption using verilog simulation.**

***Keywords*- Multiplier, Array, Wallace, Dadda, Modified Array, Verilog.**

I. INTRODUCTION

Multiplier is one of the basic functional unit in digital signal processor (DSP). Most of the high performance DSP systems rely on hardware multiplication to achieve high data throughput [5]. and since, multiplication dominates the execution time of most DSP algorithms, therefore high-speed multiplier is much desired[8]. Currently, multiplication time is still the dominant factor in determining the instruction cycle time of a DSP chip.

Multiplication is a fundamental operation in most signal processing algorithms. Multipliers have large area, long latency and consume considerable power. Therefore low-power multiplier design has been an important part in low-power VLSI system design.

Fast multipliers are essential parts of digital signal processing systems. The speed of multiplier operation is of great importance in digital signal processing as well as in the general purpose processors today [6].

In recent years, several power reduction techniques have been proposed for low-power digital design, including the reduction of supply voltage, multi threshold logic and clock speed, the use of signed magnitude arithmetic and differential data encoding, the parallelization or pipelining of operations, and the tuning of input bit-patterns to reduce switching activity.

A basic multiplier can be divided into three parts i) partial product generation ii) partial product addition and iii) final addition [5].

There are number of techniques that to perform binary multiplication. In general, the choice is based upon factors such as latency, throughput, area, and design complexity. More efficient parallel approach uses some sort of array or tree of full adders to sum partial products. Array multiplier, booth multiplier, Wallace Tree multipliers are some of the standard approaches to have hardware implementation of binary multiplier which are suitable for VLSI implementation at CMOS level .This paper represents Array multiplier(using ripple carry adder),Array multiplier(using ripple carry adder & carry save adder),Array multiplier(using carry save adder & carry look ahead adder) ,Baugh-wooley multiplier, Wallace tree, Dadda tree multiplier, Modified array multiplier and compare their efficiency.

II. OBJECTIVE

The objective of good multiplier to provide a physically compact high speed and low power consumption unit. Being a core part of arithmetic processing unit multipliers are in extremely high demand on its speed and low power consumption.

To find the most efficient timing characteristics and area report generated by the Xilinx tool for various multipliers will be compared and analyzed.

III. METHODS AND PERFORMANCES

There are number of techniques that to perform binary multiplication. In general, the choice is based upon factors such as speed, throughput, area, and design complexity power consumption. More efficient parallel approach uses some sort of array or tree of full adders to sum partial products. Array multiplier, Wallace Tree multiplier, Dadda multipliers are most common standard multipliers.

A. Array Multiplier

Array multiplier is an efficient layout of a combinational multiplier. In array multiplier, an array of identical cells generates new partial product & accumulate of it at the same time. One advantage of array multiplier comes from its regular structure .Since it is regular, it is easy to layout and circuit complexity is less. The design time of array multiplier is less than tree multiplier. The second advantage of array multiplier is its ease of design for pipeline architecture.

The main disadvantage of array multiplier is that it requires large number of gates, because of which area is increased. Power consumption is more as more logic gates are needed. Due to this reason array multiplier is less economic. Another disadvantage of array multiplier is the worst-case delay of the multiplier is proportional to the width of the multiplier. The speed will be slow for wide multiplier.

In array multiplier, consider two binary numbers A and B, of m and n bits. There are (m×n) product terms and that are produced in parallel by a set of (m×n) AND gates. A n×n array multiplier requires {n(n-2)} full adders, n half-adders and (n×n) AND gates. Also, in array multiplier worst case delay would be (2n+1)×td where td is the gate delay.[8]

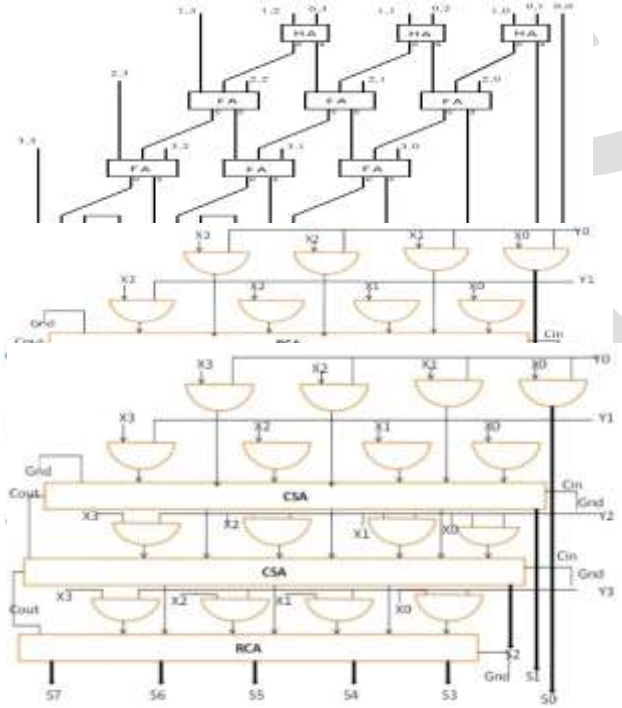


Fig.1 Array multiplier

Array multiplication can be classified into two types. These are unsigned & signed. Unsigned multiplication can be done by three types - array multiplication using ripple carry adder, array multiplication using ripple carry adder & carry save adder, array multiplication using carry save adder, carry look ahead adder. Baugh-wooley multiplication is signed multiplication.

1.1. Array Multiplier (Using RCA)

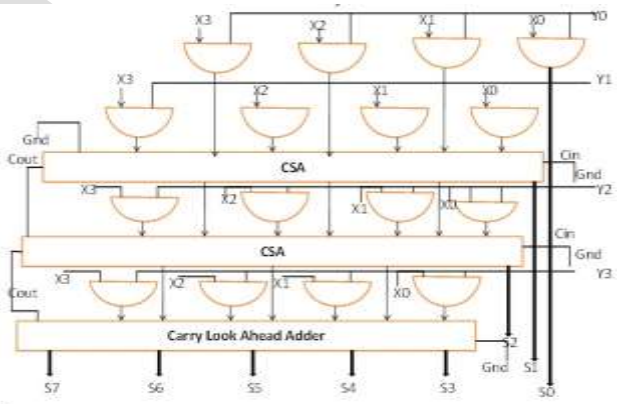
Fig.2 Array multiplier using ripple carry adder

1.2. Array Multiplier (Using RCA&CSA)

Fig.3 Array multiplier using ripple carry adder & carry save adder

1.3. Array Multiplier (Using CSA&CLA)

Fig.4 Array multiplier using carry save adder & carry look ahead adder



Delay in array multiplier is very high using ripple carry adder. Delay can be reduced if carry look ahead adder used instead of ripple carry adder in the final stage. Using these 3 methods only unsigned numbers can multiply.

1.4. Array Multiplier (Signed)/Baugh-Wooley Multiplier

The array multiplier Baugh-Wooley is an efficient way for multiplying both signed and unsigned numbers. Baugh-Wooley algorithm is used in High Performance Multiplier (HPM) tree, which inherits regular and repeating structure of the array multiplier. Baugh-Wooley multiplier exhibits less delay, low power dissipation and the area occupied is also small compared to other array multipliers. The architecture of Baugh-Wooley multiplier is based on carry save algorithm[3].

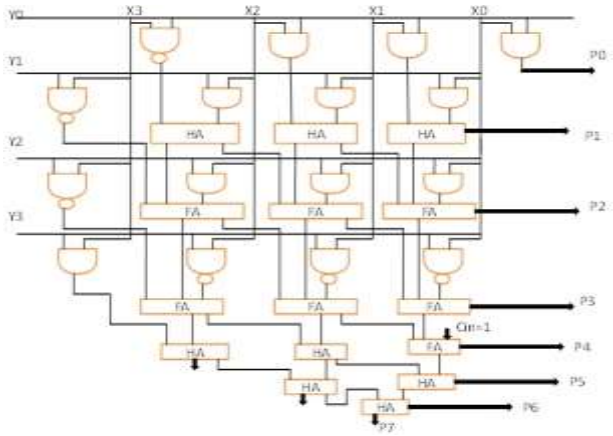


Fig.5 Baugh-wooley multiplier

B. Wallace Tree Multiplier

A fast process for multiplication of two numbers was developed by Wallace. Using this method, a three step process is used to multiply two numbers; the bit products are formed, the bit product matrix is reduced to a two row matrix where sum of the row equals the sum of bit products, and the two resulting rows are summed with a fast adder to produce a final product [2].

In the Wallace multiplier the multiplicand-multipliers are summed up in parallel by means of a tree of carry save adders. A carry save adder sums up three binary numbers and produces two binary numbers.

One advantage of Wallace tree is it has small delay. The number of logic levels required to perform summation can be reduced with Wallace tree, as a result power consumption is less. The main disadvantage of Wallace tree is complex layout and irregular wires.

A dot diagram for an 4 by 4 Wallace multiplier is shown in fig4.This architecture requires 5 fulladder, 3 halfadder to sum the 16 partial products.The Wallace tree multiplier requires a carry look ahead adder which is only 4 bit wide[1].

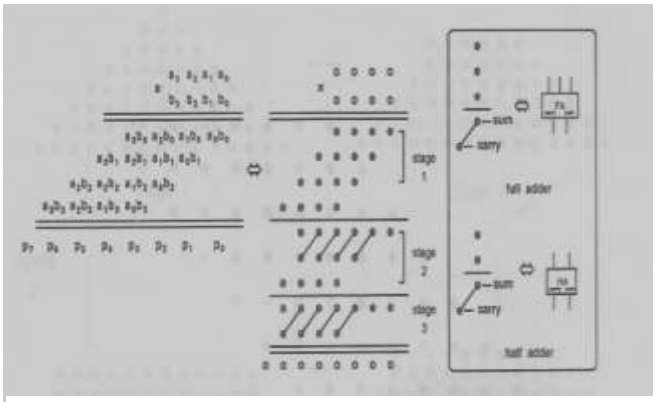


Fig.6 A 4×4 Wallace tree algorithm.

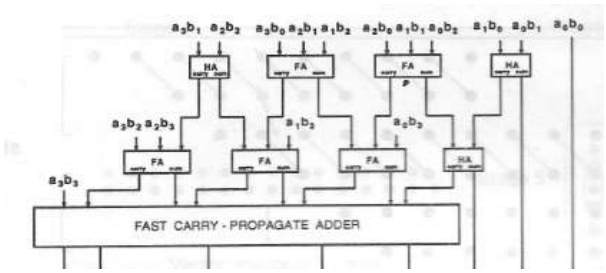


Fig.7 Wallace multiplier

C. Dadda Tree Multiplier

In a parallel multiplier the partial products are generated by using array of AND gates. The main problem is the summation of the partial products, and it is the time taken to perform this summation which determines the maximum speed at which a multiplier may operate.

The advantage of dada multiplier is that in the Wallace method, the partial products are reduced as soon as possible, in contrast, dadda method does the minimum reduction necessary at each level to perform the reduction in the same number of levels as required by the Wallace method resulting in a design with fewer full adders and half adders. So power consumption is less. The disadvantage of dadda method is that it requires a slightly wider, fast CPA and has a less regular structure than Wallace tree.

The number of Halfadder and Fulladder required for Dadda multiplier depends on N, number of bits of the operands.

Number of Fulladder = N^2-4.N+3

Number of Halfadder = N-1

Carry lookahead adder length = 2.N-2 [9].

A dot diagram for an 4 by 4 Dadda multiplier is shown in fig6. This architecture requires 3 Fulladder,3 Halfadder to sum the 16 partial products.The Wallace tree multiplier requires a carry look ahead adder which is 6 bit wide[1].

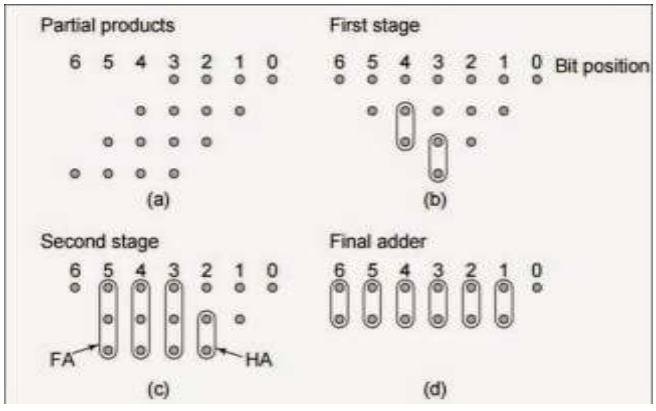


Fig.8 A 4×4 8 Dadda tree algorithm

D. Modified Array Multiplier

Modified array multiplier eliminates the drawback of array multiplier. The combinational path delay in modified array multiplier is less than other unsigned array multipliers. Moreover less number of logic gates are required. So it does not have area problem. And power consumption is least in modified array multiplier. Due to its regular structure it is very much economic [10].

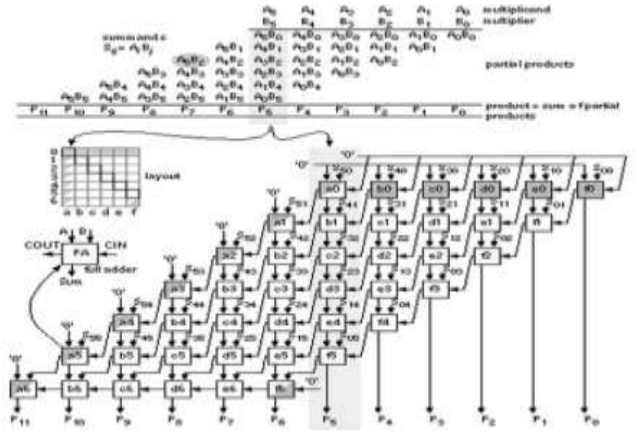


Fig.9 modified array multiplier.

IV. EXPERIMENTAL RESULT

In order to examine efficiency of multipliers we have implement & simulate all of these multipliers (4×4) in XILINX using verilog simulation discussed above.

A. Array Multiplier(Using RCA)

TABLE I

Different inputs to output path delay for the array multiplier using all RCA

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.764
A<0>	S<1>	6.837
A<0>	S<2>	8.921
A<0>	S<3>	11.125
A<0>	S<4>	12.798
A<0>	S<5>	13.304
A<0>	S<6>	14.349
A<0>	S<7>	14.464
A<1>	S<1>	6.669
A<1>	S<2>	8.717
A<1>	S<3>	10.921

Simulation result shows maximum combinational path delay obtained to be 15.607 ns.

B. Array Multiplier (Using CSA & RCA)

TABLE II

Different inputs to output path delay for the array multiplier using CSA and finally RCA

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.555
A<0>	S<1>	6.535
A<0>	S<2>	8.702
A<0>	S<3>	9.501
A<0>	S<4>	10.964
A<0>	S<5>	11.176
A<0>	S<6>	12.717
A<0>	S<7>	11.956
A<1>	S<1>	6.324
A<1>	S<2>	9.112
A<1>	S<3>	9.911
A<1>	S<4>	11.374

Simulation result shows maximum combinational path delay is obtained to be 14.344 ns.

C. Array Multiplier (Using CSA & CLA)

TABLE III

Different inputs to output path delay for the array multiplier using CSA and finally CLA.

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.371
A<0>	S<1>	8.269
A<0>	S<2>	8.682
A<0>	S<3>	9.333
A<0>	S<4>	10.547
A<0>	S<5>	9.955
A<0>	S<6>	11.416
A<0>	S<7>	11.449
A<1>	S<1>	7.868
A<1>	S<2>	8.266
A<1>	S<3>	9.873
A<1>	S<4>	11.087

Simulation result shows maximum combinational path delay is obtained to be 13.231 ns.

D. Baugh-Wooley Multiplier

TABLE IV

Different inputs to output path delay for Baugh-Wooley multiplier.

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.630
A<0>	S<1>	6.652
A<0>	S<2>	7.681
A<0>	S<3>	8.557
A<0>	S<4>	9.205
A<0>	S<5>	9.823
A<0>	S<6>	10.448
A<0>	S<7>	10.386
A<1>	C2	10.345
A<1>	S<1>	6.706
A<1>	S<2>	8.361

Simulation result shows maximum combinational path delay is obtained to be 12.340 ns.

E. Wallace Multiplier

TABLE V
Different inputs to output path delay for the Wallace Tree multiplier using CLA at end

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.644
A<0>	S<1>	7.235
A<0>	S<2>	8.461
A<0>	S<3>	9.300
A<0>	S<4>	10.165
A<0>	S<5>	11.870
A<0>	S<6>	12.172
A<0>	S<7>	12.468
A<1>	S<1>	6.650
A<1>	S<2>	7.937
A<1>	S<3>	9.714
A<1>	S<4>	10.579

Simulation result shows maximum combinational path delay is obtained to be 14.204 ns.

F. Dadda Multiplier

TABLE VI
Different inputs to output path delay for the Dadda multiplier using CLA at end

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.341
A<0>	S<1>	7.369
A<0>	S<2>	8.544
A<0>	S<3>	10.033
A<0>	S<4>	10.799
A<0>	S<5>	11.646
A<0>	S<6>	12.185
A<0>	S<7>	12.620
A<1>	S<1>	6.740
A<1>	S<2>	8.667
A<1>	S<3>	10.020
A<1>	S<4>	10.786

Simulation result shows maximum combinational path delay is obtained to be 14.537 ns.

G. Modified Array Multiplier

TABLE VII
Different inputs to output path delay for the Modified Array multiplier

All values displayed in nanoseconds (ns)

Pad to Pad

Source Pad	Destination Pad	Delay
A<0>	S<0>	6.798
A<0>	S<1>	6.677
A<0>	S<2>	8.464
A<0>	S<3>	9.467
A<0>	S<4>	9.891
A<0>	S<5>	10.608
A<0>	S<6>	11.892
A<0>	S<7>	11.917
A<1>	S<1>	7.651
A<1>	S<2>	8.116
A<1>	S<3>	9.805
A<1>	S<4>	10.894

Simulation result shows maximum combinational path delay is obtained to be 13.305 ns.

V. COMPARISON & DISCUSSION

Different parameters obtained from simulation that is useful for determining the efficiency of the multiplier includes power consumed, number of slices used, Look up tables involved, number of bonded IOBS. Their simulated values for the considered multipliers are tabulated in Table 8 for comparison.

TABLE VIII
comparison between different multipliers on the basis of delay, number of slices, no of 4 input LUTS, no of IOBS, power consumption

Multipliers	Delay(ns)	number of slices	Number of 4-input LUTS	Number of bonded IOBS	Power consumption
Array multiplier(using RCA)	15.607	16	30	16	more
Array multiplier(using RCA & CSA)	14.344	17	30	17	more
Array multiplier(using CSA & CLA)	13.231	18	34	18	most
Baugh-wooley multiplier	12.340	18	32	18	more
Wallace multiplier	14.204	17	29	17	less
Dadda multiplier	14.537	16	28	17	less
Modified array multiplier	13.305	16	27	17	least

FPGA implementation shows that array multiplier using ripple carry adder is efficient layout design, but its combinational path delay is very high. If array multiplier is designed using carry save adder, ripple carry adder combination path delay reduces, circuit becomes fast. But it requires more number of fulladders, So area will increase, so power consumption is high. Reduction of more delay is possible if array multiplier is designed using carry save adder, carry look ahead adder, but area will increase more. But using

these 3 techniques only unsigned numbers can multiply. Using baugh-wooley both signed & unsigned numbers can multiply. Another advantage of baugh-wooley is it exhibits less delay, area occupied is also small compare to other array multipliers. Using Wallace tree it is possible to reduce number of logic levels required to perform summation. Delay will be small in Wallace tree as it uses smaller carry look ahead adder compare to dadda tree, but area is more in Wallace tree compare to dadda tree. In dadda tree, less number of half adders & fulladders required. So area is smaller than Wallace, but delay is slightly higher than Wallace as carry look ahead adder is more bit wide. But dadda multiplier is less regular than Wallace. But dadda tree has less power consumption. Finally this paper represents simulation of modified array multiplier. Out of all multipliers modified array multiplier is efficient because the combinational path delay in modified array multiplier is less than other unsigned array multipliers. Moreover less number of logic gates are required. So it does not have area problem. And power consumption is least in modified array multiplier. Due to its regular structure it is very much economic.

VI. CONCLUSION

It can be concluded that modified array multiplier is superior in all respects like speed, delay, area, complexity. However array multiplier (using RCA) has more power consumption & large number of logic gates required, delay for this multiplier is larger than other array multipliers. But baugh-wooley multiplier is high speed multiplier and both signed , unsigned numbers can multiply, but it also requires large area & power consumption is very much matter of concern in this multiplier. Using Wallace and dadda tree power consumption, area, delay can be minimized little bit but these structures are irregular than array multiplier. So comparing efficiency of all the multipliers discussed above we can say, the modified array multiplier is superior than others in every respect.

ACKNOWLEDGEMENT

The road of knowledge is long and full of obstacles, but going over those obstacles is what ultimately enlightens the individual and strengthens ones spirit. I have been very fortunate to walk through the road of knowledge, not alone, but accompanied by the brightest and warm hearted individual I have ever met.

I, Abhishek Bhattacharjee, as author would take this opportunity to thank my project guide Sri Anindya Sen (Phd) for guiding me in this paper.

He has been a great source of motivation for me, which inspired me a lot to take up the project and deliver it in a nicest possible way.

Finally, I would also like to thank all the faculty

members and staff of Electronics and Communication Engineering Department for their time to time support and co-operation, which has helped me a lot.

REFERENCES

[1]. Keshag K Parhi, VLSI Digital Signal Processing Systems Design and Implementation, Wiley Student Edition, Reprint 2013.

[2]. Chepuri satish, Panem charan Arur, G.Kishore Kumar and G.Mamatha “An Efficient High Speed Wallace Tree Multiplier”.

[3]. Shruti D. Kale, Prof. Gauri N. Zade, IOSR Journal of Engineering (IOSRJEN) www.iosrjen.org ISSN (e): 2250-3021, ISSN (p): 2278-8719, “Design of Baugh-wooley Multiplier using Verilog HDL”.

[4]. Kripa Mathew, S.Asha Latha, T.Ravi, E.Logashanmugam *M.Tech-VLSI design, Sathyabama University, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai 119 “* Design and Analysis of an Array Multiplier Using an Area Efficient Full Adder Cell in 32nm CMOS Technology”,2013.

[5]. N. Ravi1, A.Satish, Dr.T.Jayachandra Prasad and Dr.T.Subba Rao Research Scholar(SVU), Department of Physics, Department of ECE “A New Design for Array Multiplier with Trade off in Power and Area”,3 May 2011.

[6]. Nishat Bano, International Journal of Scientific & Engineering Research, Volume 3, Issue 2, February -2012 1 ISSN 2229-5518 “ VLSI Design of Low Power Booth Multiplier”.

[7]. Morris Mano, “Computer System Architecture”,PP. 346-347, 3rd edition,PHI. 1993.

[8]. Sumit Vaidya and Deepak Dandekar, International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010, “DELAY-POWER PERFORMANCE COMPARISON OF MULTIPLIERS IN VLSI CIRCUIT DESIGN”.

[9]. whitney J.Townsend,Earl E.Swartzlander,Jr.,Jacob A.Abraham “A comparison of Dadda and Wallace multiplier delays”.

[10]. https://www.google.co.in/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwikqfvknOLSAhVDI5QKHZo5CqUQjRwIBw&url=http%3A%2F%2Fwww10.edacafe.com%2Fbook%2FASIC%2FCH02%2FCH02.16.php&psig=AFQjCNHMULOp_S0nB39PSKmNZwePa89RLQ&ust=1490001070395058.