International Journal of Science Engineering and Advance Technology

Design and Implementation of Unsigned Multiplier Using COSA

S.Varaprasad Babu, PG Scholar, Dept of ECE, Nova College of Engineering, JNTUK, A.P, India
T.Babu Rao, Asst.Prof, Dept.of ECE, Nova College of Engineering, JNTUK, A.P, India

Abstract: —In this paper, design of COSA multiplier is presented, by introducing Carry Save Adder (CSA) in partial product lines. The multiplier given in this paper is modeled using VHDL (Very High Speed Integration Hardware Description Language) for 32-bit unsigned data. The four binary adder architectures belong to a different adder class based 32-bit unsigned integer multiplier. The four binary adder architectures are Ripple carry adder (RCA), Carry Look-ahead adder (CLAA), Carry Select adder (CSLA) and Conditional Sum adder (COSA). All these multipliers multiply two 32-bit unsigned integer values and give a product term of 64-bit values. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the systemThis work evaluates the performance of the proposed designs in terms of delay, speed(frequency)and memory.

Keywords: RCA, CLAA, CSLA, COSA, Delay, Area, Ripple carry adder (RCA), Timing.

I. INTRODUCTION The role of arithmetic in VLSI design has been increasing in importance over the last several years due to demand for processors that are smaller faster and dissipate less power an aspect of logic design with the objective of developing appropriate algorithms in order to achieve an efficient utilization of the available hardware. The basic operations are addition, subtraction, multiplication and division. In this, we are going to deal with the operation of additions implemented to the operation of multiplication. The repeated form of the addition operations and shifting results in the multiplication operations. In VLSI designs, speed, power and chip area are the most often used measures for determining the performance of the VLSI architecture. Multipliers are vital components of any processor or computing machine. More often than not, performance of microcontrollers and Digital signal processors are evaluated on the basis of number of multiplications performed in unit time. Hence better multiplier architectures are bound to increase the efficiency of the system. However, multiplication is not as a simple operation as addition or subtraction, because it takes more time to perform two subtasks, addition and shifting. Typically, a multiplication operation takes between 2 and 8 cycles [1]. Therefore, using high-speed multipliers is a critical requirement for high performance processors. Multipliers use the addition operation for all the partial products. The adder can be a ripple-carry adder, a carry look-ahead adder, or any other adder [2,3]. However, using a fast adder to complete the multiplication operation improves the overall performance of the computer system.

Our study is focused on multipliers using unsigned data. VHDL, a Very high speed integrated circuit Hardware Description Language, was used to model and simulate the multiplier design. Several researchers had addressed the adder performance issues and others did the same with regard to the multiplier performance. Sertbas and Özbey worked on the performance analysis of classified binary adder architectures. They compared the ripple adder, carry-look-ahead adder, carry select adder and the conditional sum adder. They used VHDL implementation for their designs and comparison studies. Their work included the unit-gate models for area and delay [4]. Asadi and Navi developed a new 54×54 bit multiplier using a high-speed carry look-ahead adder. Their proposed multiplier reduced the number of transistors, delay and power consumption [1]. To design a better architecture the basic adder blocks must have reduced delay time consumption and area efficient architectures.

In this study, first a qualitative evaluation of the classified 8-bit binary adder architectures is given. Among the huge member of the adders we wrote VHDL (Hardware Description Language) code for ripple-carry, carry-select, carry look-ahead and conditional sum adder to emphasize the common performance properties belong to their classes

II. BINARY ADDER ARCHITECTURES

Addition is the most common and often used arithmetic operation on microprocessor, digital signal processor, especially digital computers. Therefore, regarding the efficient implementation of an arithmetic unit, the binary adder structures become a very critical hardware unit. The

efficient implementation of adder circuits does not only rely on optimal composition of speed-up schemes but also includes potential circuit simplifications and optimizations. This leads us to the various adder architectures described in this section. With respect to asymptotic delay time and area complexity, the binary adder architectures can be categorized into four primary classes as given in Table 1. The first class consists of the very slow ripple carry adder with the smallest area. In the second class, the carry-skip, carry-select and carry- increment adders with multiple levels have small area requirements and shortened computation times. From the third class, the carry look-ahead adder and from the fourth class, the parallel prefix and conditional sum adders represent the fastest addition schemes with the largest area complexities.

TABLE I: Categorization Of Adder Architectures

Complex (A)	*Delay (T)*	*Product (A X T)*	*Adder Class Schemes*	*l*
O(n)	O(n)	$O(n^2)$	ripple carry	1
O(n)	$O\left(n^{1/1+1)}\right)$	$O\left(n^{1+2/+1)}\right)$	Carry selected Carry Skip Carry-inc	2
O(n)	O(logn)	O(nlogn)	look-ahead	3
O(nlogn)	O(logn)	$O(n\log^2 n)$	Cond.sum Parallel prefix	4

*1 denotes the level number

The circuit structure of all architectures will be given by the set of logic equations for the composing bit slices. Finally, exact time and area complexities are given for all adder architectures based on the unit-gate model.

A. Ripple Carry Adder (RCA)

The well known adder architecture, ripple carry adder is composed of cascaded full adders for n-bit adder, as shown in fig.1. It is constructed by cascading full adder blocks in series. The carry out of one stage is fed directly to the carry-in of the next stage. For an n-bit parallel adder it requires n full adders [6].

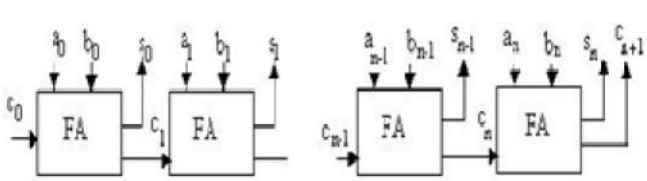


Fig.1. n-bit Ripple Carry Adder.

Delay from Carry-in to Carry-out is more important than from input to carry-out or carry-in to sum, because the carry-propagation chain will determine the latency of the whole circuit of Ripple Carry adder.

Logic equations:

$g=a_ib_i$, $p_i=a_ix\ or\ b_i$----------1

$C_{i+1}=g_i+p_ic_i\ S=p_ixor\ C_{i-}$ ----------2

Complexity and Delay for n-bit RCA structure:
ARCA = O (n) = 7n
TRCA = O (n) = 2n

B. Carry Select Adder (CSLA)

In Carry select adder scheme, blocks of bits are added in two ways: one assuming a carry-in of 0 and the other with a carry-in of 1. This results in two pre computed sum and carry-out signal pairs (s0i-1:k , c0i ; s1i-1:k , c1i), later as the block's true carry-in (ck) becomes known , the correct signal pairs are selected. Generally multiplexers are used to propagate carries. Fig.2 depicts the carry-select adder structure for n-bit adder binary inputs [6].

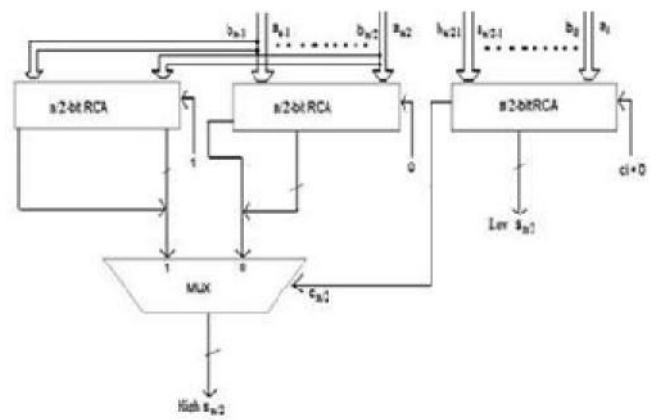


Fig.2 Carry Select Adder with level1 using n/2- bit RCA.

Because of multiplexers larger area is required. Have a lesser delay than Ripple Carry Adders. Hence we always go for Carry Select Adder while working with smaller no of bits.

Logic equations:

S_i-1:k=c_k s_0i-1:k+c_ks_1i-1:k ----------3

$C_i=c_kc_0i+c_kc_1i$ ----------4

Complexity and Delay for n-bit CSLA structure:
ACSLA = O (n) = 14n
TCSLA = O (n1/*l+1) = 2.8n1/2.

C. Carry Look-ahead Adder (CLAA)

A carry-lookahead (CLA) adder with a binary-tree structure is composed by only hierarchical logic levels. Carry Look-ahead Adder can produce carries faster due to carry bits generated in parallel by an additional circuitry whenever inputs change. This technique uses carry bypass logic to speed up the carry propagation. Let ai and bi be the augends and addend inputs, ci the carry input, si and ci+1, the sum and carry-out to the ith bit position. If the auxiliary functions, pi and gi called propagate and generate signals, the sum output respectively is defined as follows.

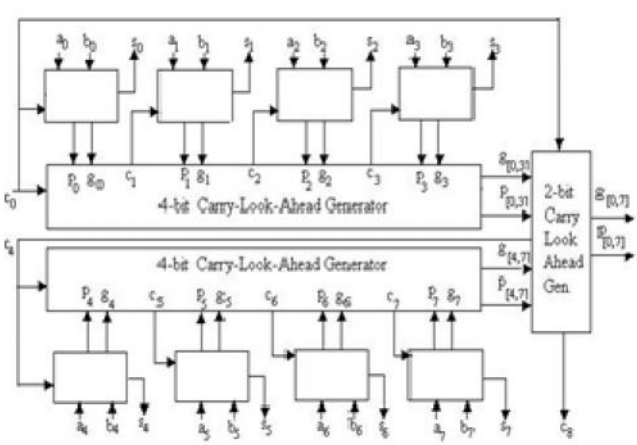
$p_i=a_i+b_i=a_ib_i$ ----------5

$s_i=a_i \text{ xor } b_i \text{ xor } c_i,\ c_i+1 = g_i+p_ic_i$ ----------6

As we increase the no of bits in the Carry Look-ahead adders, the complexity increases because the no. of gates in the expression Ci+1 increases. Instead we will use here Carry Look Ahead adder (less bits) in levels to create a larger CLA. So we can define carry look ahead over a group of 4 bits [6]. The well known expression of the carry recurrence can be yielded as follows:

$C_{i+1}=g_i+g_{i-1}p_i+g_{i-2}p_{i-1}p_i+..+c_0p_0p_1p_2$ -------7

Fig.3 depicts the block diagram of 8-bit Carry look-ahead adder.



Complexity and Delay for n-bit CLA structure:
ACLA = O (n) = 14n
TCLA = O (log n) = 4 log2n

D. Conditional Sum Adder (COSA)

The basic idea in the conditional sum adder is to generate two sets of outputs for a given group of operand bits, say, k bits. Each set includes k sum bits and an outgoing carry.. Once the incoming carry is known, then one can select the correct set of outputs (out of the two sets) without waiting for the carry to further propagate through the k positions [5]. In this method, the n-bits operands are divided into smaller groups and in this way, the serial carry propagation inside the separate groups can be done in parallel, increasing the speed of the adder. In fig.4, the application of conditional sum method is shown for the addition of two 8-bit binary numbers.

This is relatively an expensive design, as for N=2n bit addition, it requires 2N-1 full adders and 2n+1-n-2 multiplexers, which are not present in the traditional ripple carry adder. The two sets of outputs (DHA) shown in fig.4 are given in the logic equations.
Logic Equations:

$$C_{i+1}^0 = a_ib_i, S_i^0 = a_ib_i \quad \text{---------- 8}$$

$$C_{i+1}^0 = a_i + b_i, S_i^1 = S_1^0 \quad \text{----------9}$$

Complexity and delay functions for n-bit COSA:
ACOSA = O (logn) = 3nlog2n
TCOSA = O (logn) = 2 log2n

III. ANALYSIS OF ADDERS

A. Delay of Adders

TABLE II: Delay of Adders in ns

DELAY	2-bit	4-bit	8-bit
RCA	0.59	0.74	1.34
CSLA	0.30	0.59	0.98
CLAA	0.54	0.645	1.35
COSA	0.33	0.57	0.95

B. Area of Adders

TABLE III: Area of Adders in um^2

AREA	2-bit	4-bit	8-bit
RCA	342.6	917.4	3126.61
CSLA	732.2	1993.7	6006.7
CLAA	337.0	1269.8	4225.7
COSA	307.2	1536.1	6548.4

C. Power of Adders

TABLE IV: Power of Adders in uw

POWER	2-bit	4-bit	8-bit
RCA	19.561	32.219	50.613
CSLA	24.287	47.617	85.111
CLAA	12.387	37.949	61.977
COSA	11.731	33.186	65.898

We can see from the above tables that COSA has the slowest delay time and CLAA, RCA has largest delay times. Considering area RCA occupies less area when compared to all other adders. Considering power COSA is best for lower bits and RCA for more number of bits. An adder must have a performance that all the constraint such as area, power and delay must be moderate such that they can be used in most of the digital computer arithmetic applications. By area-delay analysis we can conclude that even though RCA has lowest area-delay product for higher order bits it is more. By power-delay analysis COSA is best when compared to all other adders. So for high performance COSA is best suitable adder as shown in Figs.5 and 6.

IV. MULTIPLIER FOR UNSIGNED DATA

Multiplication is a complex operation, whether performed in hardware or software. Multiplication involves the

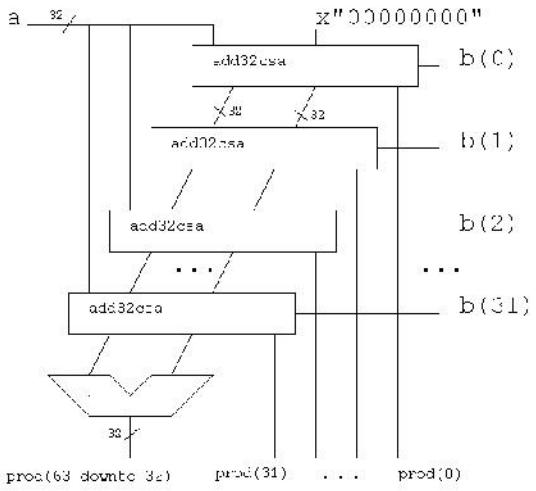


Fig.7. A partial schematic of multiplier

generation of partial products, one for each digit in the multiplier, as in Fig.7. These partial products are then summed to produce the final product. The partial products are easily defined. When the multiplier bit is 0, the partial product is 0.When the multiplier is 1, the partial product is the multiplicand. For this operation, each successive partial product is shifted one position to the left relative to the preceding partial product. The multiplication of two n-bit binary integers results in a product of up to 2n bits in length [7].

V. ALGORITHM FOR MULTIPLIER

Let the product register size be 64 bits. Let the multiplicand registers size be 32 bits. Store the multiplier in the least significant half of the product register. Clear the most significant half of the product register [7]. Repeat the following steps for 32 times:

- If the least significant bit of the product register is "1" then add the multiplicand to the most significant half of the product register.
- Shift the content of the product register one bit to the right (ignore the shifted-out bit.)
- Shift-in the carry bit into the most significant bit of the product register.

Fig.8.Shows a block diagram for such a multiplier

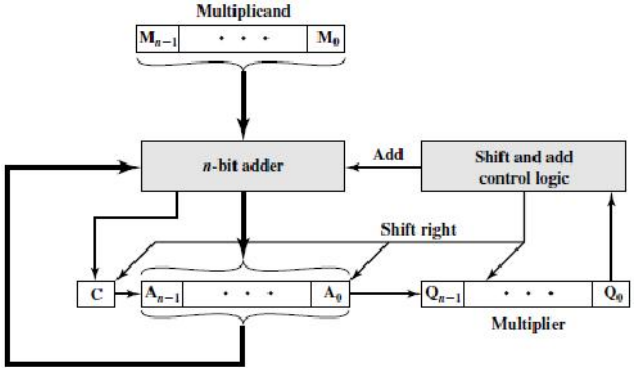


Fig.8.Multiplier of two n-bit values.

VI. VHDL SIMULATIONS

VHDL simulations of all the four multipliers are presented in this section. In this, waveforms, timing diagrams and the device utilization for all the RCA, CLA, CSLA and COSA based multipliers are shown in the figs.9 to 12. The VHDL code for all multipliers, using RCA, CLA, CSLA and COSA, are generated. VHDL model has been developed using XILINX ISE 13.1 and timing diagrams are viewed through ISE Simulator. The multipliers use two 32- bit values.

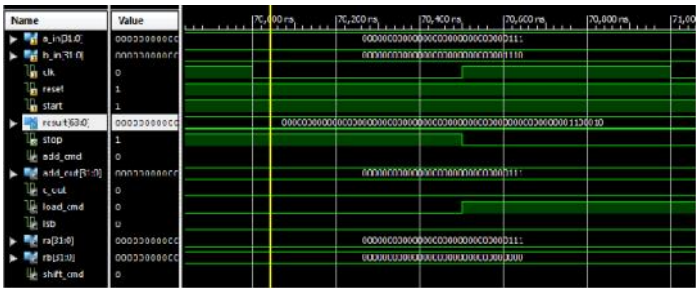


Fig.9. Timing waveform of Multiplier using RCA.

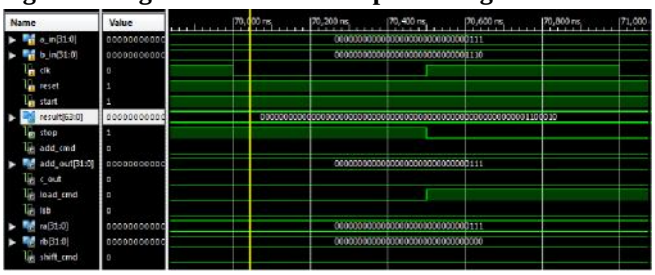


Fig.10. Timing waveform of Multiplier using CLAA.

Fig.11. Timing waveform of Multiplier using CSLA.

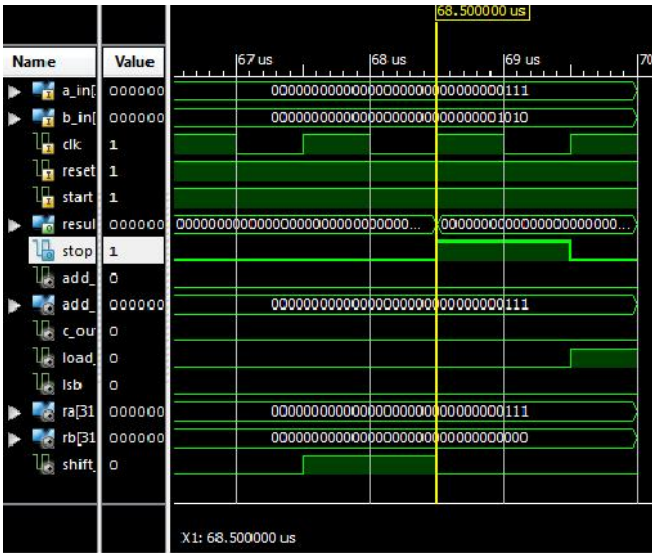


Fig.12. Timing waveform of Multiplier using COSA

VII. DEVICE SUMMARY OF MULTIPLIERS

Design of Multipliers using different adders is implemented in Virtex XC6VLX75T Device a Xilinx family with the package of FF484 and speed grade of -3 by using planahead software in Xilinx ISE 13.1 Design tool. Device utilization summary of all multipliers using different adders as components is given in Table 5.

TABLE V: Device Utilization Summary of Multipliers For Different Adders

Resource	Utilization				Available
	RCA	CLAA	CSLA	COSA	
FlipFlops	75	75	75	96	93120
LUT	178	184	182	180	46560
Slices	55	62	57	60	11640
IO	131	131	131	131	240
ILogicE1	32	32	32	32	360
Global clock buffer	2	2	2	2	32
Average Fanout	3.81	3.93	3.72	3.80	

The multipliers Minimum period report is given in Table 6.

TABLE VI:
Minimum clock period Report in ns

Minimum Period	RCA	CLAA	CSLA	COSA
TS_Clk	3.197	3.375	3.469	3.289

Worst case time delay for multiplier using different adders is given in table 8.

TABLE VII: Maximum Time Delay after Routing in ns

Timing	RCA	CLAA	CSLA	COSA
Delay	1.442	1.449	1.45	1.411

VIII. PERFORMANCE COMPARISONS

Performance of all these multipliers is compared for area, delay and area-delay product.

A. Area Analysis

When compared for area the performance of all these multipliers is shown in the fig.13

AREA

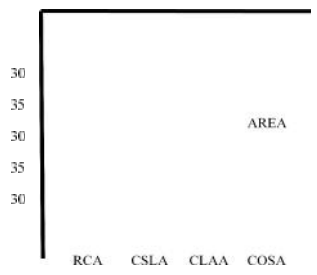


Fig.13. Area analysis in logic cells.

B. Delay Analysis

When compared for delay the performance of all these multipliers is given in fig.14.

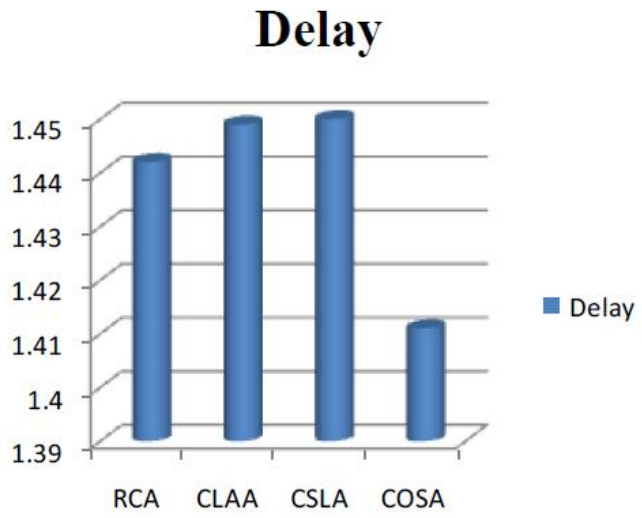


Fig.14. Delay Analysis in ns.

C. Area-Delay Analysis

The area-delay analysis of these multipliers is given in the fig.15.

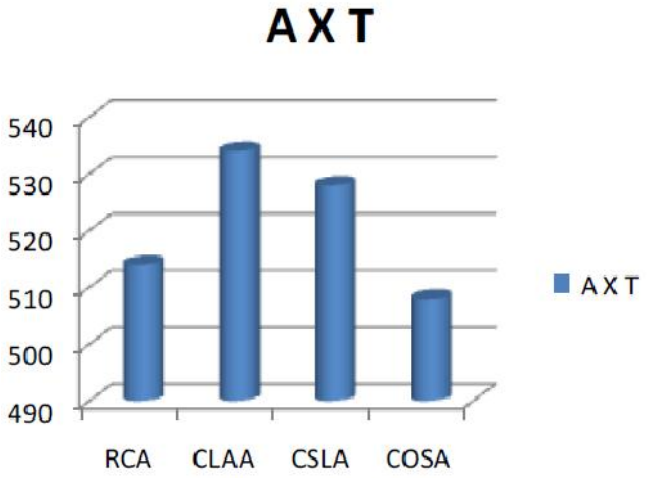


Fig.15. Area- delay analysis.

From the above figures we can conclude that COSA Multiplier has high performance when compared to other multipliers.

IX. CONCLUSIONS

A design and implementation of a VHDL-based 32 bit unsigned multiplier with RCA, CLA, CSLA and COSA is presented. The adders structures are designed using Microwind software and area-delay, power-delay products are analyzed. VHDL, a Very High Speed Integrated Circuit Hardware Description Language, is used to model and simulate all multipliers. Design of Multipliers using different adders is implemented in Virtex XC6VLX75T

Device a Xilinx family with the package of FF484 and speed grade of -3 by using PlanAhead software in Xilinx ISE 13.1 Design tool. CSLA multiplier uses fewer resources when compared to all other adders but it has more delay time. When compared to Timing COSA has the minimum delay and all other adders have the maximum delay. From Area-delay analysis we can conclude that COSA multiplier has high performance when compared to other multipliers.

X. REFERENCES

[1] P. Asadi and K. Navi, "A novel highs-speed 54-54 bit multiplier", Am. J Applied Sci., vol. 4 (9), pp. 666-672. 2007.

[2] W. Stallings, Computer Organization and Architecture Designing for Performance, 71h ed., Prentice Hall, Pearson Education International, USA, 2006, ISBN: 0-13-185644-8.

[3] J. F. Wakerly, Digital Design-Principles and Practices, 4th ed., Pearson Prentice Hall, USA, 2006. ISBN: 0131733494.

[4] A. Sertbas and R.S. Ozbey, "A performance analysis of classified binary adder architectures and the VHDL simulations", J Elect. Electron. Eng., Istanbul, Turkey, vol. 4, pp. 1025-1030, 2004.

[5] Prof. Christos Vezyrtzis, a handout on “Conditional Sum Adders: Detailed Implementation” October 7, 2013.

[6] P. S. Mohanty, "Design and Implementation of Faster and Low Power Multipliers", Bachelor Thesis. National Institute of Technology, Rourkela, 2009.

[7] Hasan Krad and Aws Yousi ("Design and Implementation of a Fast Unsigned 32-bit Multiplier Using VHDL", 2010.