High Speed Complex Vedic Multiplier using Hybrid Square Brent Kung Adder Technique

Shivdyal Ram
M. Tech. Scholar
Department of EC
Rabrindra Nath Tagore University
Bhopal, India

Ravitesh Mishra
Associate Professor
Department of EC
Rabrindra Nath Tagore University
Bhopal, India

Dr. Sanjeev Gupta
Dean Academics
Department of EC
Rabrindra Nath Tagore University
Bhopal, India

Abstract- The main objective of this research paper is to design architecture for complex Vedic multiplier by rectifying the problems in the existing method and to improve the speed by using the Brent Kung adder with the help of hybrid square technique. The Vedic multiplier algorithm is normally used for higher bit length applications and ordinary multiplier is good for lower order bits. In this paper design complex multiplier with the help of Vedic multiplier and Brent Kung adder is present. The proposed algorithm is implementation Xilinx software with Vertex-7 device family.

Keywords – Vedic Multiplier, Complex Multiplier, Hybrid Square Kogge Stone Adder, Brent Kung Adder, Xilinx Software

I. INTRODUCTION

Arithmetic is the oldest and most elementary branch of Mathematics. The name Arithmetic is derived from the Greek word “arithmos”. Arithmetic is used by almost everyone, for tasks ranging from simple day to day work. As a result, the need for a faster and efficient Arithmetic unit in computers is the interesting topic over decades. The four basic operations in elementary arithmetic are addition, subtraction, multiplication and division. Multiplication is the basic mathematical operation of scaling one number by another. Multiplication is used in today‘s engineering field covering many Digital Signal Processing (DSP) applications such as convolution, Fast Fourier Transform (FFT), filtering and in Arithmetic Logic units of a microprocessor, microcontroller and most of the Embedded controllers. The demand for high speed processing has been increasing as a result of expanding computer and signal processing applications. Higher throughput arithmetic operations are essential 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 faster multiplier circuit has been a subject of interest over decades. Reducing the time delay and power consumption are very essential requirements for many applications.

Parallel multiplication is used to meet out the current requirement. Two types of parallel multiplications are array multiplication and tree multiplication. The basic multiplier is a simple array multiplier and it is designed based on shift- and – add operation. One of the examples for array multiplication is the Braun multiplier and is designed for unsigned binary numbers. For tree structure Wallace multiplier is designed and it is also for an unsigned binary numbers. In the array multiplication, for signed numbers Baugh – Wooley, Booth Multiplier and Modified Booth Algorithm (MBA) are used. Dadda is another type of multiplier based on tree structure and is used for the multiplication of the signed numbers. These conventional binary multipliers for unsigned numbers are considered for comparison. Vedic mathematics is the system of mathematics followed in ancient India and mainly deals with Vedic mathematical formulae and their applications to various branches of mathematics. The word 'Vedic' is derived from the word 'Veda' which means the storehouse of all knowledge.

Vedic mathematics was reconstructed from the ancient Indian scriptures (Vedas) by Sri Bharati Krishna Tirthaji (1884-1960), after his eight years of research on Vedas. According to his research, Vedic mathematics is mainly based on sixteen principles or word-formulae and thirteen sub-corollaries which are termed as Sutras. This is a very interesting field and presents some effective algorithms which can be applied to various branches of Engineering such as Computing and Digital Signal Processing. Vedic mathematics reduces the complexity in calculations that exist in conventional mathematics. Generally there are sixteen sutras available in Vedic mathematics.

Among them only two sutras are applicable for multiplication operation. They are Urdhava Triyakbhyam sutra (literally means vertically and crosswise) and Nikhilam Sutra (literally means All from 9 and last from 10). Urdhava-Triyakbhyam is a generic method for multiplication. The logic behind Urdhava Triyakbhyam sutra is very much similar to the ordinary array multiplier. Here the binary implementation of this algorithm is derived based on the same logic used for decimal numbers. The binary implementation of Nikhilam Sutra is not yet successful.

This is a special case in multiplication. Another algorithm used to simplify the multiplication process is Karatsuba algorithm. This Karatsuba algorithm uses a divide-and-conquer approach where it breaks down the inputs into Most Significant half and Least Significant half and this process continue until the operands are of 8-bits wide. Karatsuba algorithm is best suited for operands of higher bit length. But at lower bit lengths, it is not as efficient as it is at higher bit

lengths. This method was discovered by Anatoli Karatsuba in 1962. It reduces the number of multipliers required, by replacing the multiplication operations by addition operations. Addition operations are faster than multiplications and hence the speed of the multiplier is increased. As the amount of bits increase, the efficiency of the multiplier will also increase.
In this research paper, a novel architecture of complex multiplier is designed using hybrid square kogge stone adder.

II. TYPES OF MULTIPLIER

Multipliers play an important role in today's digital signal processing and various other applications. Essential design targets of multiplier include high speed, low power consumption, regularity of layout and hence less area or even combination of them in one multiplier are required thereby making them suitable for various VLSI implementations.
Most multiplication techniques can be classified as Array multipliers and Tree multipliers. A detailed discussion on the different types of multipliers is done in the following sections.

(i) Array Multipliers

Array multipliers can be implemented by directly mapping the manual multiplication into hardware. The partial products are accumulated by an array of adder circuits. An n x n array multiplier requires n (n-1) adders and n^2 AND gates.

(ii) Carry Save Array Multiplier

The carry-save array multiplier uses an array of carry-save adders for the accumulation of partial product. It uses a carry-propagate adder for the generation of the final product. This reduces the critical path delay of the multiplier since the carry-save adders pass the carry to the next level of adders rather than the adjacent ones.

(iii) Wallace Tree Multiplier

C. S. Wallace (1964) propounded a fast technique to perform multiplication. A Wallace tree multiplier offers faster performance for large operands. Unlike an array multiplier the partial product matrix for a treemultiplier is rearranged in a tree-like format, reducing both the critical path and the number of adder cells needed.

(iv) Dadda Multiplier

Dadda multiplier is a hardware multiplier designed similar to Wallace multiplier. Unlike Wallace multipliers that perform reductions as much as possible on each layer, Dadda multipliers do as few reductions as possible. Due to this, Dadda multipliers have less expensive reduction phase, but the numbers may be a few bit longer, thus requiring slightly bigger adders. This implies that fewer columns are compressed in the initial stages of the column compression tree, and more columns in the later levels of the multiplier.

(v) Booth Wallace Tree Multipliers

The use of Booth's algorithm, in multiplication presents an efficient solution that suits the demands of high-speed multipliers, which also need to be efficient in terms of hardware design/area complexity.
In this multiplier, the partial products are generated using the Booth method, while their summation is done using Wallace tree structure. This compresses the addition process. The adder is used in adder accumulator configuration, where the multiplication result is added every time to the partial product register.

Table 1: Comparison of Multiplier

Types of Multiplier	Delay/ Increase in bit size	Structure/ Complexity	Area	Speed
Array	Linear	Regular/ Low	High	Low
Wallace	Logarithmic	Irregular/ High	High	High
Dadda Tree	Logarithmic	Irregular/ High	High	High
Booth	Non-linear	Regular/ Medium	Low	Medium

III. VEDIC MULTIPLIER

Vedic multiplier and hybrid square Kogge Stone adder can compare with conventional method which is computed by Vedic multiplier, full adder and half adder. Proposed technique provides less path delay and less area. Input sequence of Conventional method is much more than to proposed method, however proposed method has less propagation delay. Area and propagation delay can be reduced by the aid of hybrid square Kogge Stone adder. This adder will be designed like as ripple carry adder.
Logic Diagram of Vedic Multiplier using Kogge Stone Adder is shown in figure 1. Eventually, all the designing levels of digital system or IC's Packages depend on number of gates in a single chip that is also called bottom up approach. Modified KS adder can be reduced regarding the area or number of gates. If we remove the first XOR gate from modified KS adder nothing will be changed for result but area and propagation delay will be reduced.

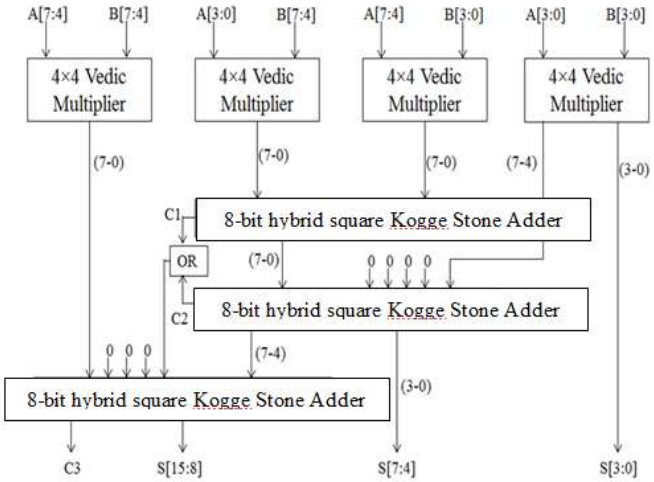

Figure 1: 8-bit Vedic Multiplier using Hybrid Square Kogge Stone Adder

(i) Hybrid Square Kogge Stone Adder

The complete functioning of KSA can be easily comprehended by analyzing it in terms of three distinct parts:

(a) Preprocessing: - This step involves computation of generate and propagate signals corresponding too each pair of bits in A and B. These signals are given by the logic equations below:

$$P_i = A_i \, xor \, B_i \tag{1}$$

$$G_i = A_i \, and \, B_i \tag{2}$$

(b) Carry look ahead network: - This block differentiates KSA from other adders and is the main force behind its high performance. This step involves computation of carries corresponding to each bit.

$$C_i = G_i \, or \, (P_i \, and \, C_{i-1}) \tag{3}$$

(c) Post processing: - This is the final step and is common to all adders of this family (carry look ahead). It involves computation of sum bits. Sum bits are computed by the logic given below:

$$S_i = P_i \, xor \, C_{i-1} \tag{4}$$

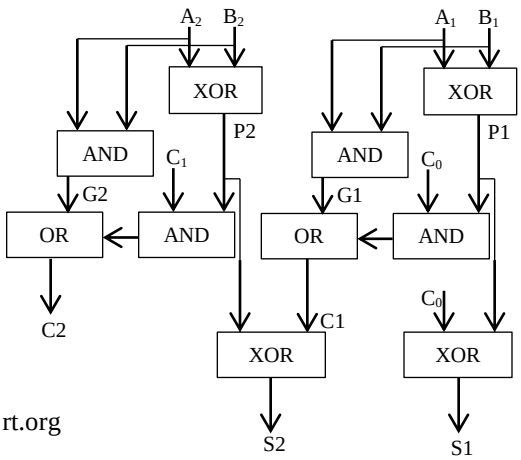

Figure 2: 2-bit Hybrid Square Kogge Stone Adder

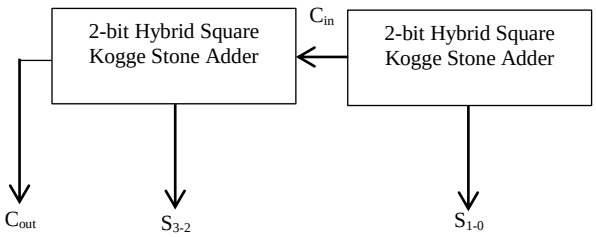

Figure 3: 4-bit Hybrid Square Kogge Stone Adder

(ii) Brent Kung Adder

The Brent–Kung adder is a parallel prefix adder (PPA) type of carry look adder (CLA). Proposed by Richard Peirce Brent and Hsiang Te Kung in 1982 it acquainted higher consistency with the snake structure and has less wiring clog prompting better execution and less important chip zone to actualize contrasted with the Kogge– Stone adder (KSA). It is likewise substantially speedier than ripple carry adders (RCA).

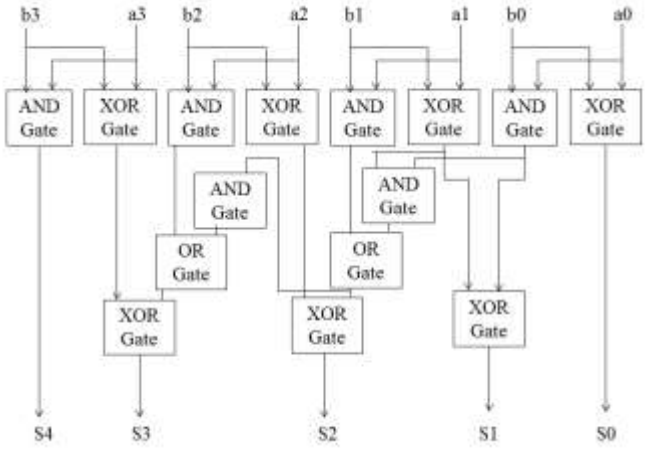

Figure 4: Block Diagram of Brent Kung Adder

IV. COMPLEX MULTIPLIER

Suppose two numbers are complex then

$$A = A_r + jA_i$$

$$B = B_r + jB_i$$

The product of A and B then

$$P = A \times B$$

$$P = A_r \times B_r - A_i \times B_i + j(A_r \times B_i + A_i \times B_r)$$

$$P_r = A_r \times B_r - A_i \times B_i$$

$$P_i = A_r \times B_i + A_i \times B_r$$

Where P_r and P_i is represents the real and imaginary part of the output of the complex multiplier. A_r and A_i is represents the real and imaginary part of the first input of the complex multiplier. B_r and B_i is represents the real and imaginary part of the second input of the complex multiplier.
Complex multiplier for four Vedic multipliers is shown in figure 5. In this block diagram reduce four Vedic multipliers to three Vedic multipliers is shown in below:

$$P_r = A_r \times B_r - A_i \times B_i = A_r(B_r + B_i) - B_i(A_r + A_i)$$

$$P_i = A_r \times B_i + A_i \times B_r = A_r(B_r + B_i) + B_r(A_i - A_r)$$

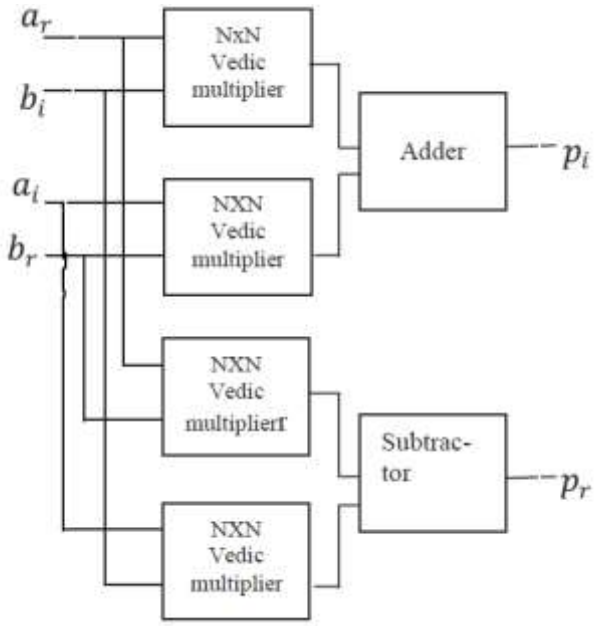

Figure 5: Block Diagram of Complex Multiplier for four Vedic Multiplier

V. SIMULATION ANALYSIS

Simulation of these experiments can be done by using Xilinx 14.2 i VHDL tool. In this paper we are focusing on propagation delay. Propagation delay must be less for better performance of digital circuit.
As shown in table I the number of slice Look Up Tables (LUT), Input Output Buffers (IOB), XOR gate and delay are obtained for the Vedic multiplier using hybrid square kogge stone adder, ripple carry adder and Brent Kung adder. From the analysis of the results, it is found that the Vedic multiplier using hybrid square kogge stone adder gives a superior performance in terms of delay and Vedic multiplier using Brent Kung adder gives a good performance in terms of slice LUT and XOR gate as compared with ripple carry adder.

Table I: Comparison Result for 8-bit Vedic Multiplier

Design	Slice LUTs	IOB	XOR Gates	Delay
Vedic Multiplier using Ripple Carry Adder				
Vedic Multiplier using Hybrid Square Kogge Stone Adder	132	32	152	7.508 ns
Vedic Multiplier using Brent Kung Adder	119	32	152	7.786 ns

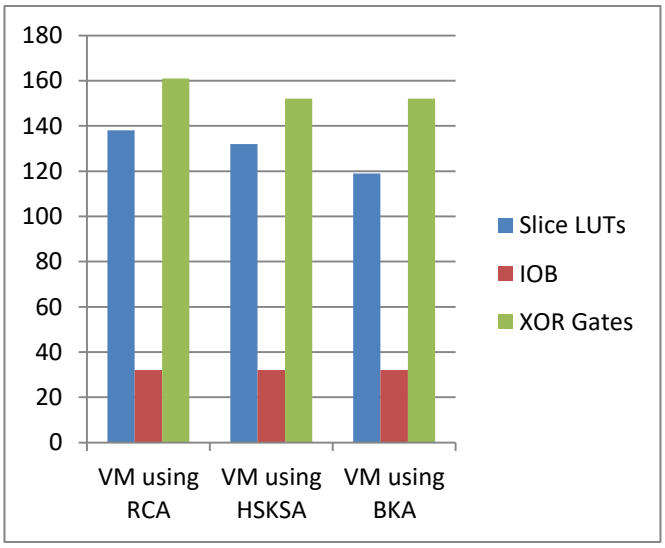

Figure 6: Bar graph of the 8-bit Vedic multiplier

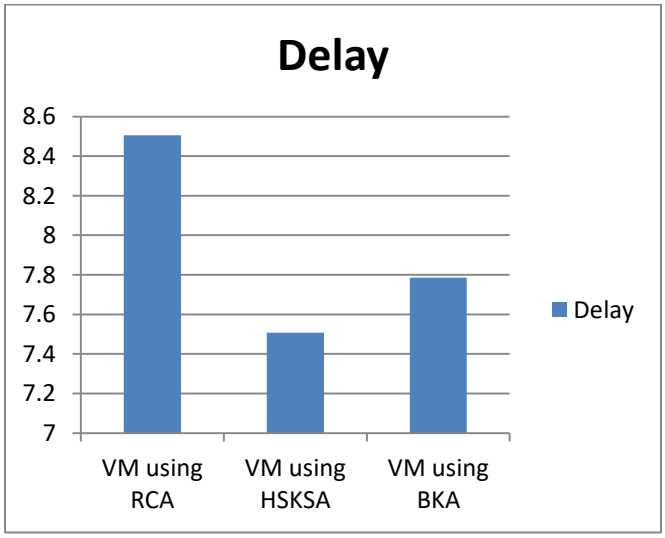

Figure 7: Bar graph of the 8-bit Vedic multiplier for Delay

Figure 6 and figure 7 shows the graphical illustration of the performance of VM using different adder discussed in this research work in term of number of LUTs, XOR gates, IOB and delay. From the above graphical representation it can be

inferred that the VM using BKA algorithm gives the best performance as compared with previous algorithm.

As shown in table II the number of LUTs, delays are obtained for the complex Vedic multiplier using Brent Kung adder and previous algorithm. From the analysis of the results, it is found that the complex Vedic multiplier using Brent Kung adder gives a superior performance as compared with previous algorithm for Vertex-7 device family. The output waveform of the 32-bit complex multiplier using Brent Kung adder is shown in figure 8 and figure 9 respectively.

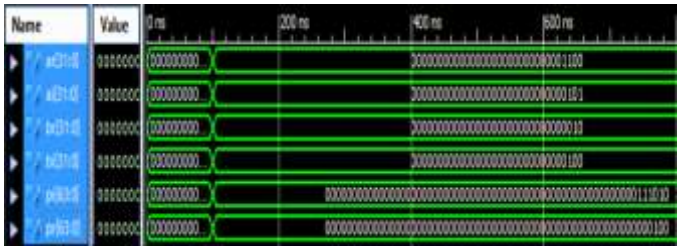

Figure 8: Output Binary Waveform of 32-bit Complex Multiplier using HS-KSA

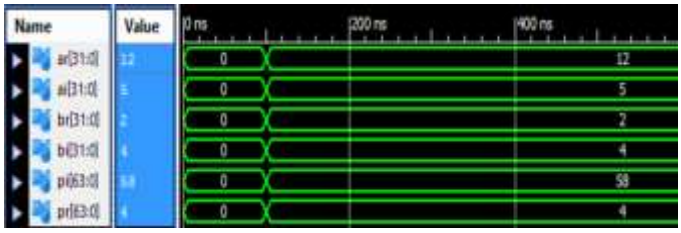

Figure 9: Output Decimal Waveform of 32-bit Complex Multiplier using HS-KSA

Table II: Comparison Result for 32-bit Complex Vedic Multiplier for four Vedic Multiplier

Design	Number of LUTs	Number of IOBs	XOR Gates	Delay
Complex Vedic Multiplier [2]	10416	256	-	25.98 ns
Complex Vedic Multiplier using Ripple Carry Adder	10642	256	12929	26.93 ns
Complex Vedic Multiplier using Hybrid Square Kogge Stone Adder	10222	256	12260	25.20 ns
Complex Vedic Multiplier using Brent Kung Adder	9300	256	12260	25.52 ns

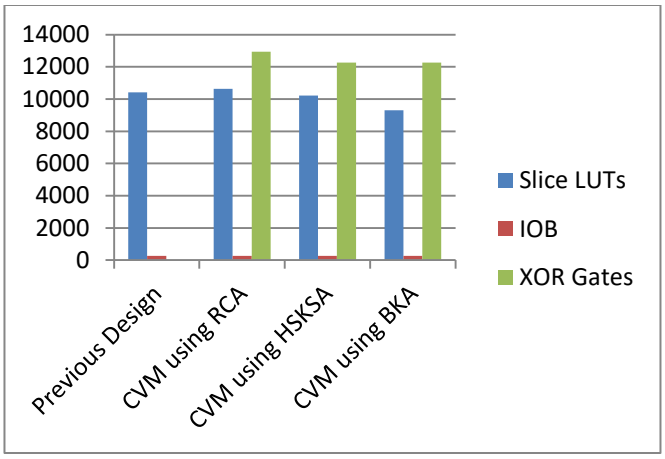

Figure 10: Bar graph of the 32-bit Complex Vedic multiplier

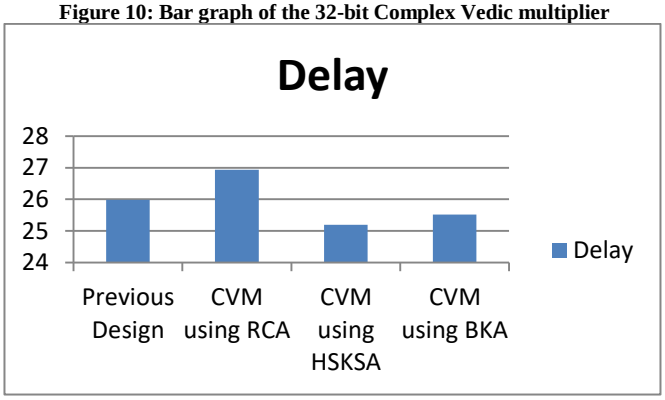

Figure 11: Bar graph of the 32-bit Complex Vedic multiplier for Delay

Figure 10 and figure 11 shows the graphical illustration of the performance of CVM using BKA algorithm discussed in this research work in term of number of slice, number of LUTs and delay. From the above graphical representation it can be inferred that the CVM using BKA algorithm gives the best performance as compared with previous algorithm.

VI. CONCLUSION

In this paper design of hybrid square Kogge Stone adder, Brent Kung adder Vedic multiplier and complex Vedic multiplier is presented. From implementation results it is observed that the hybrid based Vedic Multiplier and complex Vedic multiplier consumes less delay compare to previous design.

REFRENCES

[1] Vijaya Lakshmi Bandi, “Performance Analysis for Vedic Multiplier using Modified Full Adders”, International Conference on Innovations in Power and Advanced Computing Technologies, i-PACT2017.

[2] K. Deergha Rao, Ch. Gangadhar and Praveen K Korrai, “FPGA Implementation of Complex Multiplier Using Minimum Delay Vedic Real Multiplier Architecture”, 2016 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), IEEE 2016.

[3] Soma Bhanu Tej, “Vedic Algorithms to develop green chips for future”, International Journal of Systems, Algorithms &

Applications, Volume 2, Issue ICAEM12, February 2012, ISSN Online: 2277-2677.

[4] Youngjoon Kim and Lee-Sup Kim, “A low power carry select adder with reduced area”, IEEE International Symposium on Circuits and Systems, vol.4, pp.218-221, May 2001.

[5] Y. Choi, “Parallel Prefix Adder Design,” Proc. 17th IEEE Symposium on Computer Arithmetic, pp 90-98, 27th June 2005.

[6] Kogge P and Stone H (1973), “A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Relations”, IEEE Transactions on Computers, Vol. C-22, No. 8, pp. 786-793.

[7] Madhu Thakur and Javed Ashraf (2012), “Design of Braun Multiplier with Kogge-Stone Adder & It’s Implementation on FPGA”, International Journal of Scientific & Engineering Research, Vol. 3, No. 10, pp. 03-06, ISSN 2229-5518.

[8] Pakkiraiah Chakali and Madhu Kumar Patnala (2013), “Design of High Speed Kogge-Stone Based Carry Select Adder”, International Journal of Emerging Science and Engineering (IJESE), Vol. 1, No. 4, ISSN: 2319-6378.

[9] Somayeh Babazadeh and Majid Haghparast, "Design of a Nanometric Fault Tolerant Reversible Multiplier Circuit" Journal of Basic and Applied Scientific Research, 2012.