International Journal of Emerging Engineering Research and Technology
Volume 3, Issue 8, August 2015, PP 110-116
ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)

Design and Implementation of Wallace Tree Multiplier Using Kogge Stone Adder and Brent Kung Adder

G. Shireesha[1], Dr. G. Kanaka Durga[2]

[1]*M.E, Department of Electronics and Communication Engineering, M.V.S.R Engineering College, Hyderabad*
[2] *Professor & HOD,Department of InformationTechnologyEngineering,M.V.S.REngineeringCollege,Hyderabad*

ABSTRACT

A fixed point Wallace tree multiplier architecture is used to perform multiple multiplications on different data paths. Wallace tree multiplier using Kogge Stone and Brent Kung adders perform more number of multiplications in parallel with fewer extra carry save adder stages than existing multiplier. The modified n-bit Wallace tree multiplier structure is used to perform four (n/2)×(n/2)-bit multiplications, two n×(n/2)-bit multiplications and one n×n-bit multiplication in parallel. In the existing Wallace tree multiplier design Carry lookahed adder (CLA) is used. To further improve the speed and to reduce the area parallel prefix adders are used in the modified Wallace tree multiplier. The Kogge Stone adder (KSA) is used for high-speed and Brent Kung adder (BKA) is used to reduce the area. Wallace tree multiplier using KSA and BKA are implemented using Xilinx 13.2

Keywords: Kogge stone adder, Brent Kung adder, Wallace tree multiplier.

INTRODUCTION

The performance of the embedded system, microprocessor and many modern DSP applications are mainly dependent on the performance of the multipliers as it is the key element. An efficient system can be build by making suitable multiplier for the system. A multiplier has two stages, partial product generation and partial product addition. An n bit multiplier produce n number of partial products and to reduce this number of partial products modified Booth algorithm [1] was used. But booth encoder in this multiplier increases the circuit depth. The n bit Baugh Wooly array multiplier [2] has the circuit depth O(n). In modern technology, vector processors [3] are playing a major role to achieve data level parallelism (DLP). Here multiple operands are following the multiple data paths in the same hardware. So only one instruction is used to perform the operations on the vector of data. The twin precision based array multiplier is explained in [4] is used for data level parallelism. Where the full precision multiplier is used to implement two half precision multiplications with circuit depth of O(n). That is the 8-bit multiplier is used to perform two 4-bit multiplications or one 8 bit multiplication at a time. The depth of Carry Save Adder (CSA) is O(1). The depth of the carry save addition tree is O(log2 n) for Wallace tree [5] based multiplier and O(n) for Braun based[5] multiplier, where n is the number of bits. The quarter precision Wallace tree multiplier [6] is used to produce the multiple multiplications to improve the speed with increased circuit depth as trade off. To further improve the speed and reduce the area Wallace tree multiplier using Kogge Stone adder and Brent Kung adder [7]-[8] are used.

32-BIT WALLACE TREE MULTIPLIER

The modern digital signal processor requires large multiplier to compute complex signal processing operations. The DSP processor shows the need for 64-bit Multiplier, where four 32×32-bit multiplications or sixteen 16×16-bitmultiplications or four16×32-bit multiplications or four8×8-bit multiplications are performed using one 64-bit Multiplier in parallel. In the same way, the Wallace tree multiplier architecture is allowed to perform more than one multiplication in parallel to achieve data level parallelism in vector processors. The 32-bit Wallace tree multiplier is having 8 carry save stages and 54-bit final adder to get the product. Where the modified 32-bit Wallace structure is having

Address for correspondence:

sirihancy@gmail.com

Block I, Block II, Block III and 25-bit recursive doubling based CLAs are involved. Fig. 3, 4 and 5 are showing the architecture for Block I, Block II and Block III respectively. Each of the Block I will act as16-bit Wallace tree multiplier's carry reduction tree and they are getting the partial products from each of the quarters The Block I is having 6carry save stages. The final carry and sum from CSA14 of Block I is sent to 25-bit CLA to get four16-bit multiplication results at a time.

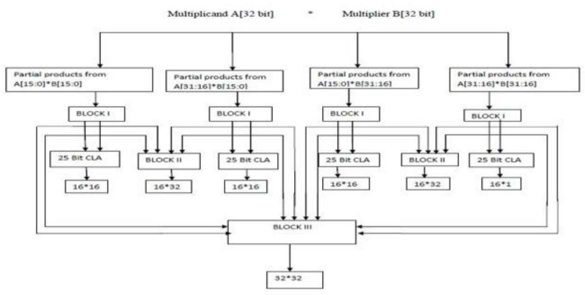


Fig1. *Wallace tree multiplier*

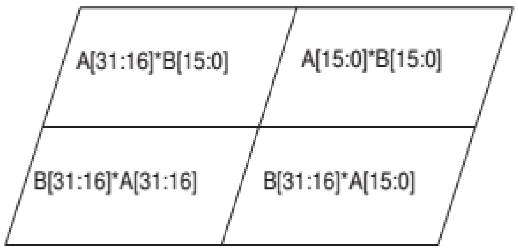


Fig2. *Partial product arrangement*

Block III respectively. Each of the Blocks I will act as16-bit Wallace tree multipliers carry reduction tree and they are getting the partial products from each of the quarters. The Block I is having six carry save stages. The final carry and sum from CSA14 of Block I is sent to 25-bit CLA to get four16-bit multiplication results at a time.

Blocks of 32-bit Wallace Tree Multiplier

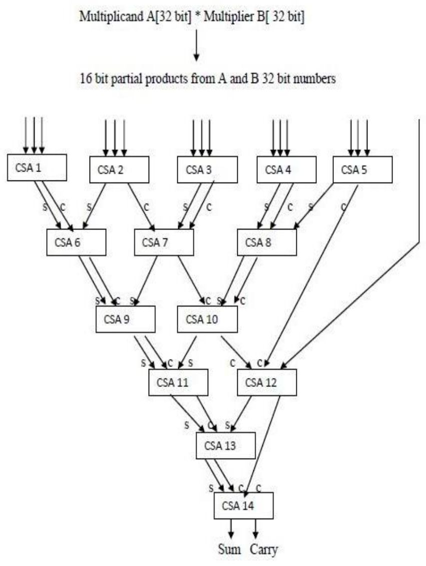

Multiplicand A[32 bit] * Multiplier B[32 bit]
16 bit partial products from A and B 32 bit numbers
CSA 1
CSA 2
CSA 3
CSA 4
CSA 5
CSA 6
CSA 7
CSA 8
CSA 9
CSA 10
CSA 11
CSA 12
CSA 13
CSA 14
Sum
Carry

Fig3. *Block I of Wallace tree multiplier*

Block II

The inputs to Block II are from the output of CSA14 of two of the Block I, which tends to produce two16×32-bit multiplication results in parallel. The Block II is having two carry save stages and one 40-bit recursive doubling based CLA. Similarly the inputs to block III are from the output of CSA14 of the entire Block I, which tends to produce one 32×32-bit multiplication result. The Block III is having four carry save stages and one54-bit recursive doubling based CLA. And all the Block II, Block III and 25-bit CLAs are in parallel. Therefore the critical path of the modified structure includes Block I and Block III. So the total critical depth of the 32-bit Wallace tree multiplier is equal to the addition of number of carry stages of Block I, number of carry save stage of Block III and depth of 54-bit recursive doubling based CLA. So the critical path of Wallace structure includes 10 carry save stages and one 54-bit recursive doubling based CLA. So the Wallace structure requires two extra carry save stages than existing 32-bit Wallace structure and these causes slightly increase in the worst path delay of modified system than the existing multiplier. The 32-bit Wallace structure has 30 carry save adders and one 54-bit recursive doubling based CLA. The Block I has14 Carry Save adders (CSA), Block II has 2 Carry Save adders and cell III has six Carry Save adders. So the 32-bit Wallace structure has(4∗14)+(2∗2)+(1∗6) = 66 Carry Save adders, four 25-bit recursive doubling based CLAs, two 40-bit recursive doubling based CLAs and one54-bit recursive doubling based CLA. And hence, this huge difference causes increase in total cell area, total number of cells and net power than conventional structure.

Block III

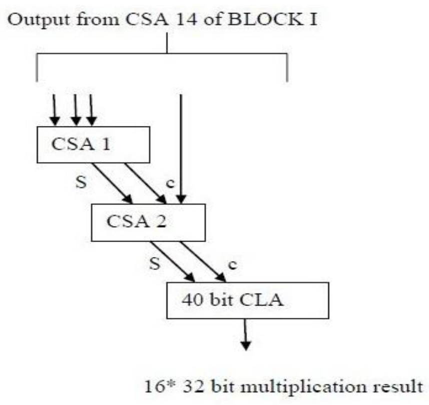

Output from CSA 14 of BLOCK I
CSA 1
S
c
CSA 2
S
c
40 bit CLA
16* 32 bit multiplication result

Fig4. *Block II of Wallace tree multiplier*

Block III

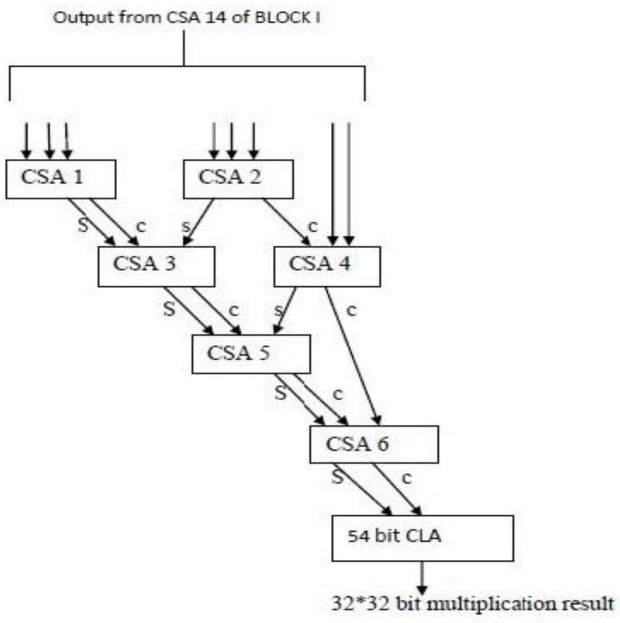

Output from CSA 14 of BLOCK I
CSA 1
CSA 2
S
c
S
c
CSA 3
CSA 4
S
c
S
c
CSA 5
S
c
CSA 6
S
c
54 bit CLA
32*32 bit multiplication result

Fig5. *Block III of Wallace tree multiplier*

MODIFIED WALLACE TREE MULTIPLIER

In Wallace tree multiplier [6] CLA is replaced with Kogge Stone adder (KSA) and Brent Kung adder(BKA) to further improve the speed and reduce the area. Kogge Stone adder and Brent Kung adders are the parallel prefix adders. Parallel-prefix structures are found to be common in high performance adders because the delay is logarithmically proportional to the adder width .The parallel prefix adders are more flexible and are used to speed up the binary additions. Parallel prefix adders are obtained from Carry Look Ahead (CLA) structure. The construction of parallel prefix adder involves three stages

1. Pre- processing stage
2. Carry generation network
3. Post processing

Pre-possessing stage

In this stage we compute, generate and propagate signals to each pair of inputs A and B. These signals are given by the logic equations 1&2

Pi=Ai xor Bi (1)

Gi=Ai and Bi (2)

Carry generation network

In this stage we compute carries corresponding to each bit. Execution of these operations is carried out in parallel. After the computation of carries in parallel they are segmented into smaller pieces. It uses carry propagate and generate as intermediate signals which are given by the logic equations 3 and 4

$CP_{i:j}=P_{i:k+1}$ and $P_{k:j}$ (3)

$CG_{i:j}=G_{i:k+1}$ or ($P_{i:k+1}$ and $G_{k:j}$) (4)

Post processing

This is the final step to compute the summation of input bits. It is common for all adders and the sum bits are computed by logic equation 5 and 6:

Ci-1= (Pi and Cin) or Gi (5)

Si=Pi xor Ci-1 (6)

These parallel prefix adders contains gray cells and block cells. The black cell (BC) generates the ordered pair, the gray cell (GC) generates only left signal [7].

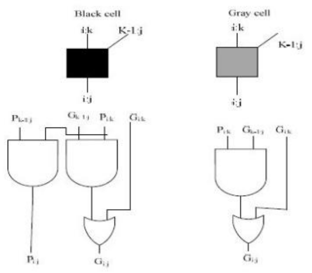


Fig6. *Black cell and gray cell*

32 BIT WALLACE TREE MULTIPLIER USING KSA

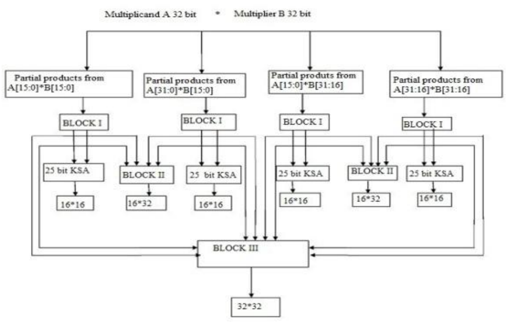

Multiplicand A 32 bit * Multiplier B 32 bit
Partial products from A[15:0]*B[15:0]
Partial products from A[31:0]*B[15:0]
Partial produts from A[15:0]*B[31:16]
Partial products from A[31:16]*B[31:16]
BLOCK I
BLOCK I
BLOCK I
BLOCK I
25 bit KSA
BLOCK II
25 bit KSA
25 bit KSA
BLOCK II
25 bit KSA
16*16
16*32
16*16
16*16
16*32
16*16
BLOCK III
32*32

Fig7. *Modified Wallace tree multiplier using KSA*

Kogge-Stone adder is a parallel prefix form of carry look-ahead adder. A parallel prefix adder can be represented as a parallel prefix graph consisting of carry operator nodes. The time required to generate carry signals in this prefix adder is O(log n). It is a fastest adder design and common design for high performance adders in industry .In pre computation stage propagation and generation operations are performed. Stage1 used 14 black cells and one gray cell. Stage2 used 12 black cells and two gray cells. Stage3 used 8 black cells and 4 gray cells. Stage4 used 8 gray cells. In final computation stage sum and carry generated.

16-bit Kogge Stone adder

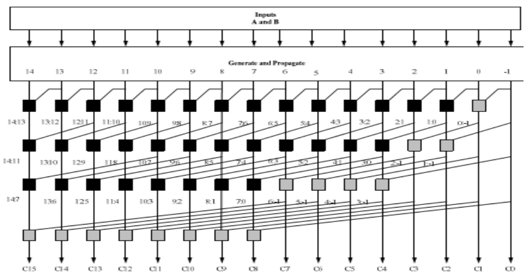


Fig8. *16-bit Kogge Stone adder*

Block II

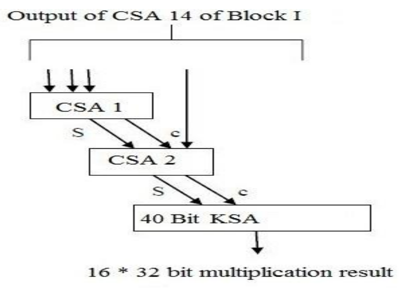

Output of CSA 14 of Block I
CSA 1
S
C
CSA 2
S
C
40 Bit KSA
16 * 32 bit multiplication result

Fig9. *Block II of modified Wallace tree multiplier using KSA*

Block III

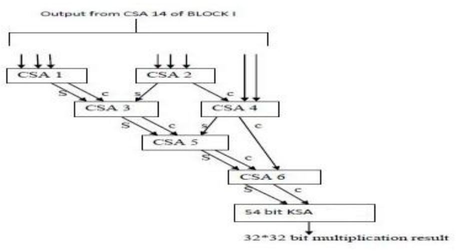

Output from CSA 14 of BLOCK I
CSA 1
CSA 2
CSA 3
CSA 4
CSA 5
CSA 6
54 bit KSA
32*32 bit multiplication result

Fig10. *Block III of modified Wallace tree multiplier using KSA*

32-BIT WALLACE TREE MULTIPLIER USING BKA

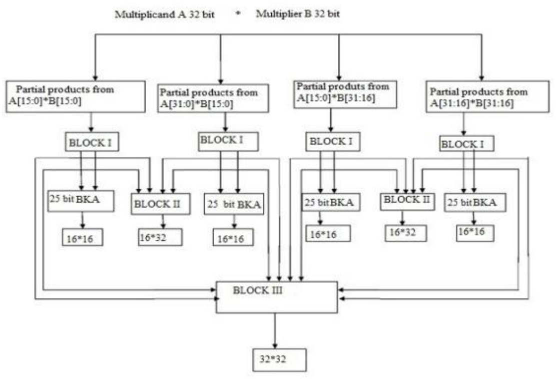

Multiplicand A 32 bit
*
Multiplier B 32 bit
Partial products from A[15:0]*B[15:0]
Partial products from A[31:0]*B[15:0]
Partial produts from A[15:0]*B[31:16]
Partial products from A[31:16]*B[31:16]
BLOCK I
25 bit BKA
BLOCK II
16*16
16*32
BLOCK III
32*32

Fig11. *Modified Wallace tree multiplier using BKA*

The cost and wiring complexity is greatly reduced using Brent Kung adders .In pre computation stage we are doing propagation and generation operations. After that stage 1 used seven black cells and one gray cell. Stage 2 used three black cells and one gray cell. Stage 3 used one black cells and one gray cell. Stage 4 used one gray cell.stage5 used one gray cell. Stage 6 used three gray cells. Stage 7 used seven gray cells. In final computation stage sum and carry are generated.

16-Bit Brent Kung Adder

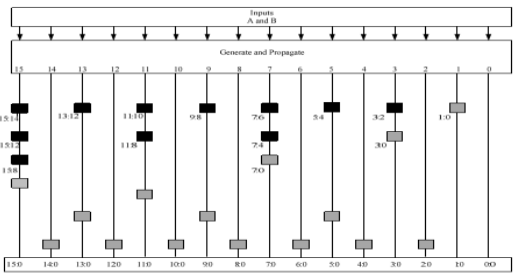


Fig12. *16-bit Brent Kung adder*

Block II

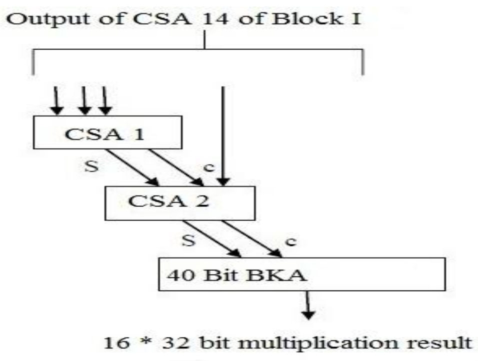

Output of CSA 14 of Block I
CSA 1
S
C
CSA 2
S
C
40 Bit BKA
16 * 32 bit multiplication result

Fig13. *Block II of modified Wallace tree multiplier using BKA*

Block III

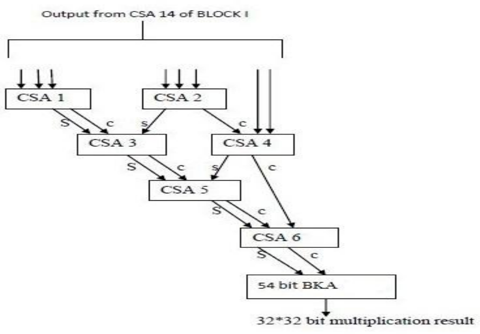
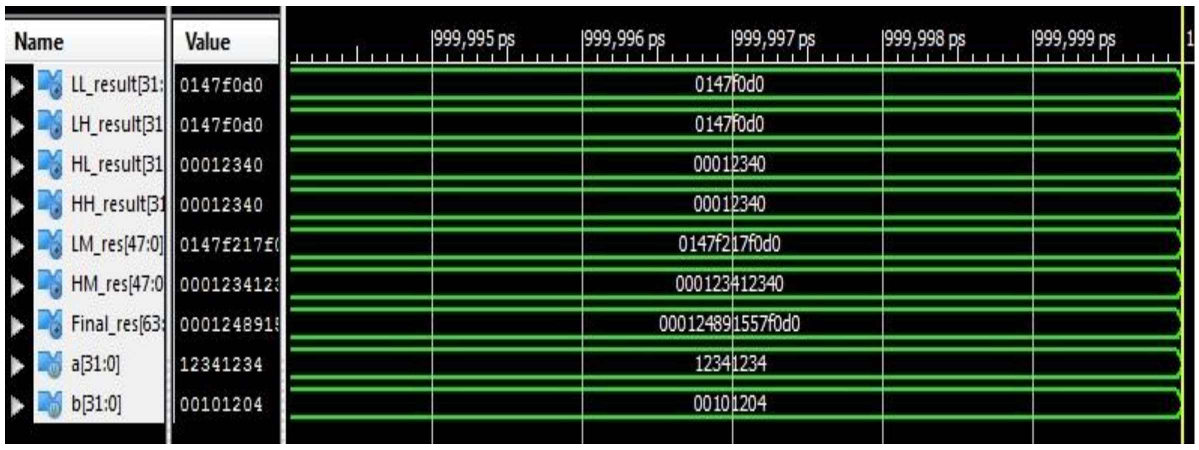

Output from CSA 14 of BLOCK I
CSA 1
CSA 2
CSA 3
CSA 4
CSA 5
CSA 6
54 bit BKA
32*32 bit multiplication result

Fig14. *Block III of modified Wallace tree multiplier using BKA*

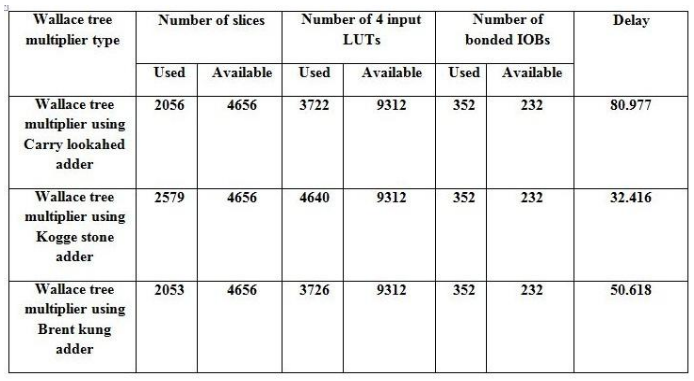
SIMULATION WAVEFORMS



Name	Value	999,995 ps – 999,999 ps
LL_result[31:	0147f0d0	0147f0d0
LH_result[31	0147f0d0	0147f0d0
HL_result[31	00012340	00012340
HH_result[31	00012340	00012340
LM_res[47:0]	0147f217f(	0147f217f0d0
HM_res[47:0	00012341 2:	000123412340
Final_res[63:	00012489 1!	000124891557f0d0
a[31:0]	12341234	12341234
b[31:0]	00101204	00101204

The simulation is done using Xilinx ISE 13.2 tool. In the above waveform a and b are 32-bit numbers. The final output is 64-bit multiplication result. In parallel four 16*16 bit and two 16*32 bit results also produced.

COMPARISON OF AREA AND DELAY OF WALLACE TREE MULTIPLIER USING ADDERS



Wallace tree multiplier type	Number of slices		Number of 4 input LUTs		Number of bonded IOBs		Delay
	Used	Available	Used	Available	Used	Available	
Wallace tree multiplier using Carry lookahed adder	2056	4656	3722	9312	352	232	80.977
Wallace tree multiplier using Kogge stone adder	2579	4656	4640	9312	352	232	32.416
Wallace tree multiplier using Brent kung adder	2053	4656	3726	9312	352	232	50.618

The delay of Wallace tree multiplier using Kogge Stone adder is improved from 80.977ns to 32.416ns. Wallace tree multiplier using Brent Kung adder delay is improved from 80.977ns to 50.916ns and also the area is also reduced by three slices.

CONCULSION

A 32-bit Wallace tree multiplier is modified and redesigned. The modified Wallace tree multiplier achieves data level parallelism in vector processors. In addition part CSA, CLA adders used to enhance to prefix adders KSA, BKA. KSA adder having high-speed and BKA adder take less area so three adders using 32 bit Wallace tree multiplier is implemented by Xilinx 13.2 and hardware kit FPGA spartan3e.

REFERENCES

[1] P.E. Madrid, B. Millar, and E.E. Swartzlander, "Modified booth algorithm for high radix multiplication", IEEE International Conference on Computer Design: VLSI in Computers and Processors, pp. 118-121, Oct.1992.

[2] C.E. Kozyrakis and D.A. Patterson, "Scalable, vector processors for embedded systems", Micro, IEEE Journals and magazines, vol. 23, no.6, pp. 36-45, 2003.

[3] Sjalander M and Larsson-Edefors P, "High-Speed and Low-Power Multipliers Using the Baugh-Wooley Algorithm and HPM Reduction Tree", IEEE International Conference on Electronics, Circuits and Systems, page(s) 33-36, Sep. 2008.

[4] M. Sjalander and P. Larsson-Edefors, "Multiplication acceleration through twin precision", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, no. 9, pp. 1233-1246, Sept. 2009.

[5] C. S. Wallace, 'A suggestion for a fast multiplier', IEEE Transactions on Electronic Computers, vol. EC-13, no. 1, pp. 1417, Feb. 1964

[6] Mohamed Asan Basiri, M. Samaresh Chandra Nayak and Noor Mahammad Sk, 'Multiplication Acceleration Through Quarter Precision Wallace Tree Multiplier,' IEEE 2014.

[7] Sudheer kumar Yezerla, B Rajendra Naik ,'Design and estimation of delay, power, and area for parallel prefix adders', IEEE 2014.

[8] Adilakshmi Siliveru, M.Bharathi 'Design of Kogge stone and Brent kung adders using degenerate pass transistor logic', International journal of emerging science and engineering 2013.