Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 11, November 2014, pg.437 – 446

RESEARCH ARTICLE

OPTIMIZING THE POWER USING FUSED ADD MULTIPLIER

[1]Dr. C.Venkatesh
Associate Professor, Department of Electronics and Communication Engineering,
Sri Eshwar College of Engineering, Coimbatore-641202, Tamil Nadu, India
E Mail id: cvenkateshmail@yahoo.com, +91 7667421400.

[2]S.Poonkuzhali
PG Student, Department of Electronics and Communication Engineering,
Sri Eshwar College of Engineering, Coimbatore-641202, Tamil Nadu, India
E Mail id: poonkuzhali159@gmail.com, +91 8870606672.

ABSTRACT – Many Digital Signal Processing (DSP) applications carry out a large number of complex arithmetic operations. Multiplier take important role in high performance of the system, reduce in power and area. This paper is focus on optimizing the design of Fused Add Multiply (FAM) operator. This implements a new technique by direct recoding of sum two numbers in Modified Booth (MB) form. It is used for both signed and unsigned Radix-4 which is a parallel multiplier. An efficient multiplier with reduce partial product by N/2 where N is the number of multiplicand. The proposed FAM unit is coded in VHDL, simulated and synthesized using Xilinx ISE tool. The performance of FAM unit is compared with other existing technique in terms of power consumption and critical path. The proposed FAM unit yields considerable reduction in terms of critical delay and power consumption.

KEYWORDS – Digital Signal Processing (DSP), Multiply-Accumulator (MAC), Add-Multiply (AM) operation, Modified Booth (MB) algorithm, Carry Look Ahead Adder (CLA).

1. INTRODUCTION

Multiplier plays an important part in digital signal processing (DSP) system. It is used in implementation of recursive, transverse filters and Discrete Fourier Transforms. The performance of DSP systems is inherently affected by decisions on their design regarding the allocation and the architecture of arithmetic units. Using of the large large computation leads to increases in power and area of the system.

In [1], a two-stage recorder which converts a number in carry-save form to its MB representation. The first stage transforms the carry-save form of the input number into signed-digit form which is then recoded in the second stage and it matches the form that the MB digits request. Recently, this technique had been used for the design of high performance flexible coprocessor architectures targeting the computationally intensive DSP applications.

In [2] the recoding of a redundant input from its carry-save form to the corresponding borrow-save form keeping the critical path of the multiplication operation fixed. The multiplication operation product is obtained by adding partial products presented in [3], thus the final speed of the multiplier circuit depends on the

speed of the adder circuit and the number of partial products generated. Radix-8 booth encoded technique used then there are only 3 partial products and only one CSA and CLA is required to produce the final product. Through this algorithm it increases in power of the system.

Based on the observation that an addition can often be subsequent to a multiplication (e.g., in symmetric FIR filters), the Multiply-Accumulator (MAC) and Multiply-Add (MAD) units were introduced [4] leading to more efficient implementations of DSP algorithms compared to the conventional ones, which use only primitive resources [5]. Several architectures have been proposed to optimize the performance of the MAC operation in terms of area occupation, critical path delay or power consumption [6], [7].

In [8], MAC components increase the flexibility of DSP data path synthesis as a large set of arithmetic operations can be efficiently mapped on the system. The fused operations are a two-term dot product and add-subtract unit. The FFT processors [9] use "butterfly" operations that consist of multiplications, additions, and subtractions of complex valued data. Both radix-2 and radix-4 butterflies are implemented efficiently with the two fused floating-point operations.

Although the direct recoding of the sum of two numbers in its MB form leads to a more efficient implementation of the Fused Add-Multiply (FAM) unit compared to the conventional one. The existing recoding schemes are based on complex manipulations in bit-level, which are implemented by dedicated circuits in gate-level. This paper focuses on the efficient design of FAM operators and targeting the optimization of the recoding scheme for direct shaping of the MB form of the sum of two numbers. The adoption of the proposed recoding technique delivers optimized solutions for the FAM design enabling the targeted operator to be timing functional (no timing violations) for a larger range of frequencies. Also, under the same timing constraints, the proposed designs deliver improvements in both area and power consumption, thus outperforming the existing Modified Booth Scheme recoding solutions.

2. PROPOSED METHOD

2.1. FAM ARCHITECTURE

The direct recoding of the sum of two numbers in its MB form produce more efficient implementation of the FAM unit compared to the conventional one. The existing recoding schemes are based on complex manipulations in bit-level, which are implemented by circuits in gate-level. This paper focuses on the efficient design of FAM operators, targeting the optimization of the recoding scheme for direct shaping of the MB form of the sum of two numbers. More specifically, this paper proposes a new recoding technique which decreases the critical path delay, reduces area and power consumption. The proposed Modified Booth Recoding Scheme algorithm is structured, simple and can be easily modified in order to be applied either in signed or unsigned numbers, which comprise of odd or even number of bits. In this technique, explore an alternative schemes of the proposed Modified Booth Recording Scheme approach using conventional and signed-bit Full Adders (FAs).

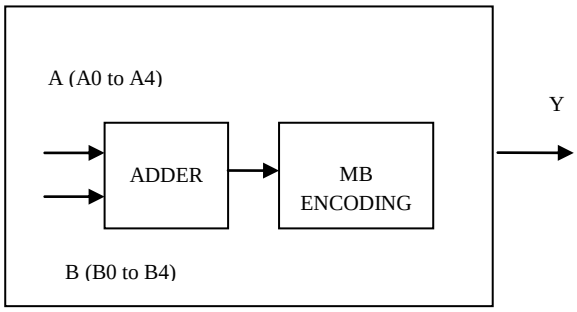


Fig.1. FAM UNIT DESIGN

In the FAM design presented in Fig. 1. It shows the multiplier is a parallel one based on the MB algorithm. Consider the product X·Y, where X is the multiplier and Y is the output of adder A and B. The term Y is expressed in terms of (y_{n-1} y_{n-2} ……. y_1 y_0) is encoded based on the MB algorithm and X in terms of (x_{n-1} x_{n-2} ….. x_1 x_0). Both X and Y consist of n=2k bits where the value of k is 4 and are in 2's complement form.

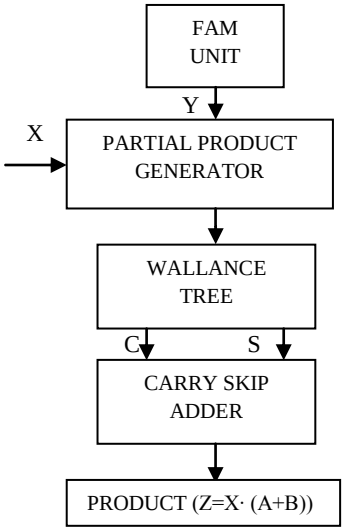


Fig.2. FAM ARCHITECTURE

The basic FAM architecture is shown in Fig.2. The generation of k partial products can be written as,

$$PP = X \cdot Y_j^{MB} = \bar{p}\, 2^n + \sum_{i=0}^{n-1} p_{j,i} \cdot 2^i \quad (1)$$

Where PP is the generation of partial products and digit Y_j^{MB} is correspond to the three consecutive bits, Y_{2j+1}, Y_{2j}andY_{2j-1}. Table I shows how they are formed by summarizing the MB encoding technique. Each digit is represented by three bits named S, one and two. The sign bit S shows if the digit is negative (S=1) or positive (S=0). Signal one shows if the absolute value of a digit is equal to 1 (one=1) or not (one=0). Signal two shows if the absolute value of digit is equal to 2 (two=1) or not (two=0). Using these three bits we calculate the MB digits Y_j^{MB} by the following relation:

$$Y_j^{MB} = (-1)^{sj} \cdot [one_j + two_j] \quad (2)$$

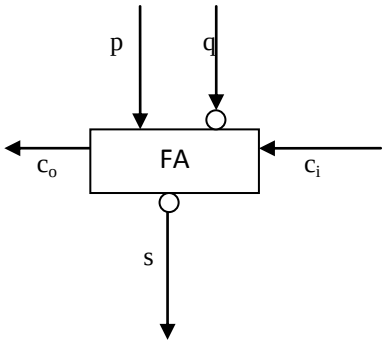
TABLE – 1
Modified Booth Encoding Table.

Binary			MMB Encoding			carry
Y_{2j+1}	Y_{2j}	Y_{2j-1}	Sign= sj	onej	twoj	
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	0	1	0	0
0	1	1	0	0	1	0
1	0	0	1	0	1	1
1	0	1	1	1	0	1
1	1	0	1	1	0	1
1	1	1	1	0	0	0

After the partial products are generated, they are added through a Wallance Carry Save Adder (CSA) tree along with the correction term. Finally the carry save output of the Wallance CSA tree is leaded to a fast Carry skip adder to form the final result Z = X·Y.

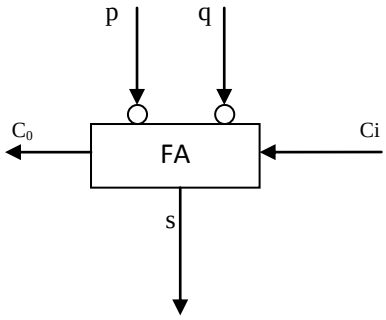
2.2 MODIFIED BOOTH RECODING SCHEME (MBRS)

In MBRS recoding technique, the sum of two consecutive bits of the input A and B are converted into MB digit Y_j^{MB}. The most significant digit is negatively weighted while the two least significant digits is positive weight The two pairs of bits in MB form are used signed-bit arithmetic. For this purpose, a set of bit-level signed Full Adders (FA) are consider here to calculate the carry and sum. The schematics and Boolean equation for even and odd bit half adders is shown in Fig.3 and Fig.4.



Full Adder for Odd Bit,
Carry = $((p \vee \bar{q}) \wedge c_i) \vee (p \wedge \bar{q})$,
Sum = p XOR q XOR c_i

Fig. 3 Boolean equation and schematics for half adder odd bit



Full Adder for Even Bit,
Carry = $((p \vee q) \wedge \bar{C}\imath) \vee (p \wedge q)$,
Sum = p XOR q XOR c_i

Fig. 4 Boolean equation and schematics for half adder even bit

The basic operation of full adder for odd bit and even bit is shown in the Table 4 and 5 respectively. Considering that p, q and c_i are the binary inputs and c_0, s are the outputs (carry and sum respectively) of a full adder for even bits. It implements the relation $2 \cdot c - s = p - q + c_i$ where the bits s and q are considered negatively signed (Table 4). Table 5 show the operation of full adder for odd bits which implements the relation $2 \cdot c + s = -p - q + c_i$ and manipulates a negative (q) and a positive (p) inputs.

Table –2
Truth Table For Full Adder Odd Parity Dual Operation

Inputs			Output Value	Output	
p(+)	q(-)	c_i		Carry, C	Sum, S
0	0	0	0	0	0
0	0	1	+1	1	1
0	1	0	-1	0	1
0	1	1	0	0	0
1	0	0	+1	1	1
1	0	1	+2	1	0
1	1	0	0	0	0
1	1	1	+1	1	1

Output value = $2.c_0 - s = p - q + c_i$

Table –3
Truth Table For Full Adder Even Parity Dual Operation

Inputs			Output Value	Output	
p(-)	q(-)	c_i		Carry, C	Sum, S
0	0	0	0	0	0
0	0	1	+1	0	1
0	1	0	-1	1	1
0	1	1	0	0	0
1	0	0	-1	1	1
1	0	1	0	0	0
1	1	0	-2	1	0
1	1	1	-1	1	1

Output value = $-2.c_0 + s = -p-q+c_i$

In the proposed recoding technique is referred as MBRS. For the given circuits Fig 5 Full adder is modified to obtain low power and high speed. The sum of A and B is given by the next relation

$$Y = A + B = y_k \cdot 2^{2k} + \sum_{j=0}^{k-1} y_j^{MB} \cdot 2^{2j} \quad (5)$$

Where $y_j^{MB} = -2s_{2j+1} + s_{2j} + c_{2j}$ (6)

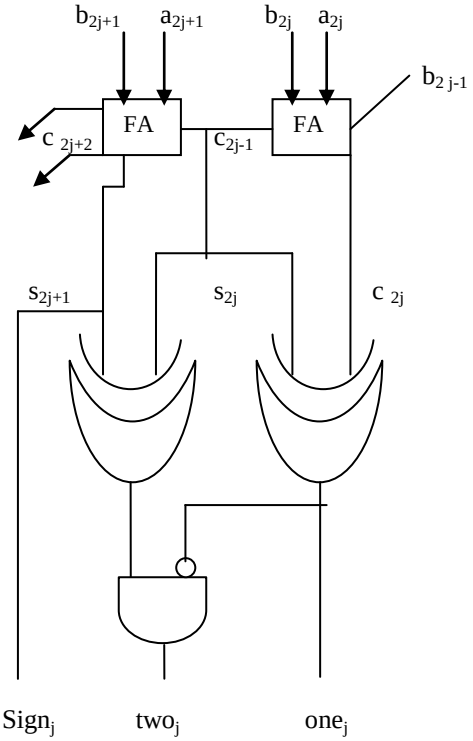


Fig. 5 Circuit diagram for proposed MBRS

The critical path delay of MBRS recoding scheme can be calculated as:

$$T_{MBRS-2} = T_{FA\ Carry} + T_{FA\ Sum} \quad (7)$$

Where $T_{FA,Carry}$ are the delays of shaping the output carry of a conventional even bit Full Adder respectively and $T_{HA,Sum}$ is the delay of forming the sum of a signed odd bit full adder.

3. ADDER ARCHITECTURE

In digital system there are many ways to perform binary addition, each of with its own area\delay trade-off. A great many tricks are used to speed up addition, encoding, replication of factors, and precharging are just some of them. In this paper, Wallance tree and Carry skip adder are used.

3.1 WALLANCE TREE

In order to speed up multiplication is to use more adders to speed the accumulation of partial products. The best-know method for speeding up the accumulation is the Wallace tree, which is an adder tree built from Carry Save Adder, which is simply an array of full adders. A carry save adder is given three n-bit numbers a,b,c computes two new numbers that is carry and sum. The structure of the Wallance tree is shown in the Fig.6.

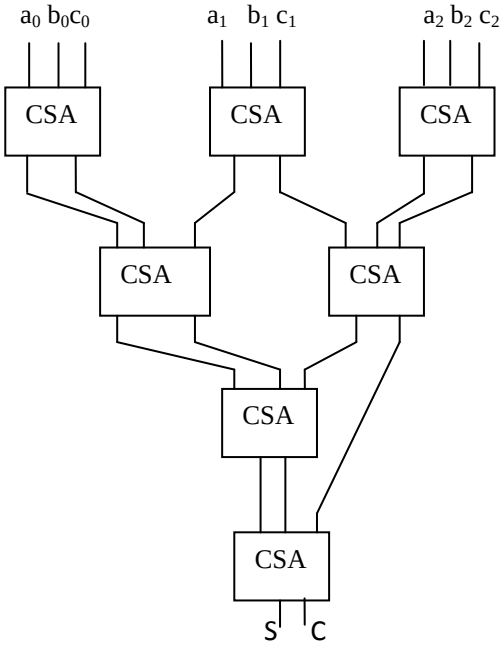


Fig. 6 structure of wallance tree

A wallance tree multiplier is fater than a simple array multipliter because its height is logarithmic in the word size, not liner. A carry save adder, given three n-bit numbers a,b,c, computes two new products of sum and carry. The partial products are introduced at the top of the tree. Each of the outputs is shifted left by one bit scince it represent the carry out. Then the produced carry and sum products are given to the input of the carry skip adder.

3.2 CARRY SKIP ADDER

The carry skip is a set of bits is the same as the carry in to those bits. This adder make use of the carry propogate relationship. The structure of carry skip adder is shown in fig. 7. Thus the carry chain for a carryskip adder divided into the group.

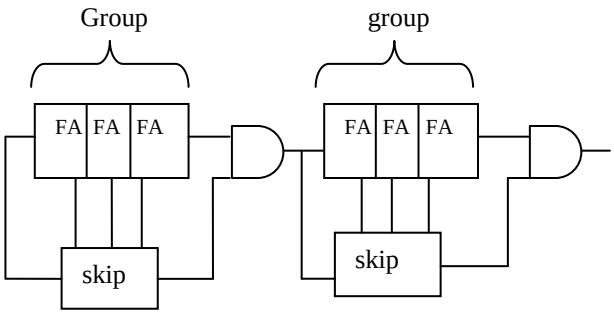


Fig 7. Structure of carry skip adder

Each time carry is propagated for every bit in the stage, then a bypass gate sends the stage carry input directly to the carry output. It reduces the delay in the circuit. The critical path of a carry-skip-adder begins at the first full-adder, passes through all adders and ends at the sum-bit. Carry-skip-adders are chained to reduce the overall critical path, since a single n-bit carry-skip-adder has no real speed benefit compared to a n-bit carry ripple adder. Thus the final result of Z=(X·Y) is obtained from carry skip adder.

4. POWER ANALYSIS

The power of FAM unit using Carry look ahead adder and carry save adder are compared and tabulated in table 4. It also shows that the proposed fused add multiple unit algorithm consumes less power than the existing algorithm like modified booth recoder, basic recoding algorithm etc. The Fig. 8 shows a bar chart with power in y-axis and various techniques in x- axis. From the fig.8 it is found that the FAM with carry skip adder posses the least power of 0.034W, while the basic recoding algorithm posses the maximum of 0.122W.

Table 4- Power analysis

Various technique	Power (W)
FAM+CSA adder	0.034
FAM+CLA adder	0.038
MB ALGORITHM	0.109
BASIC RECORDING ALGORITHM	0.122

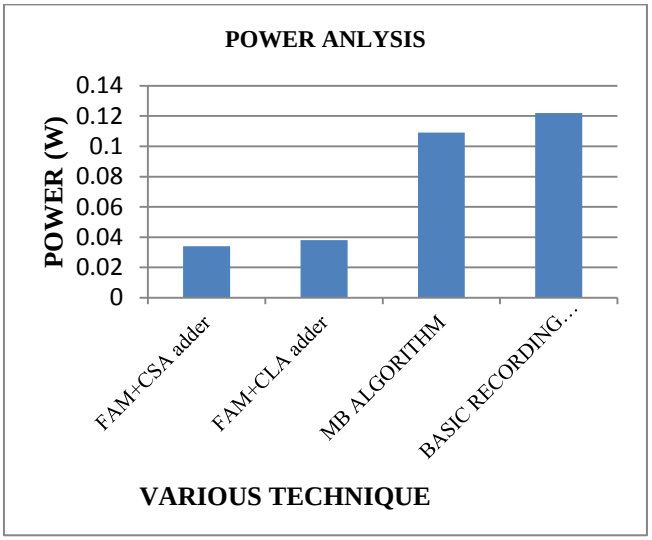


Fig 8. Power Vs Various technique

The Fused add multiply unit with carry skip adder posseses lower power compared to the fused add multiply uint with carry look ahead adder. The carry skip adder uses the advantage of reducing the critical path in the circuit and operates faster with lower delay overheads.

5. SIMULATION RESULTS

The proposed Fused add unit is implemented in the VHDL, simulated with Xilinx and modelsim. Testing is carried out using various inputs. This is an used to reduced the computation in the circuits. The simulation results of fused add multiply unit with carry look ahead adder is shown in Fig.9 FAM Unit with CLA adder for even bit and Fig.10 FAM unit with CLA adder for odd bit. An two 8-bit input A and B is given to the FAM unit. Then produced output 8-bit from FAM unit is multiplied with the 8-Bit multiplicand X and partial products are generated. Then the generated 16-bit Z output is taken from CLA adder .

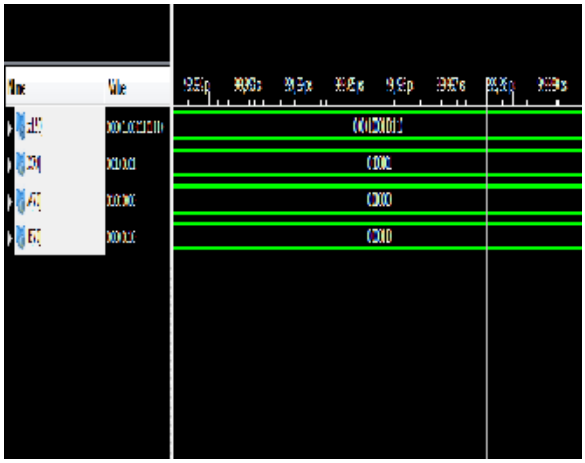


Fig. 9 FAM unit with CLA adder for even bit

Fig. 10 FAM unit with CLA adder for odd bit

The simulation result of FAM with CSA adder is shown in Fig. 11 FAM unit with CSA adder for even bit and Fig 12 FAM unit with CSA adder for odd bit. Similar to the first architecture two 8-bits inputs of A and B is given to the FAM unit. Then produced output 8-bit from FAM unit is multiplied with the 8-Bit multiplicand X and partial products are generated. Then the generated 16-bit Z output is taken from CSA adder.

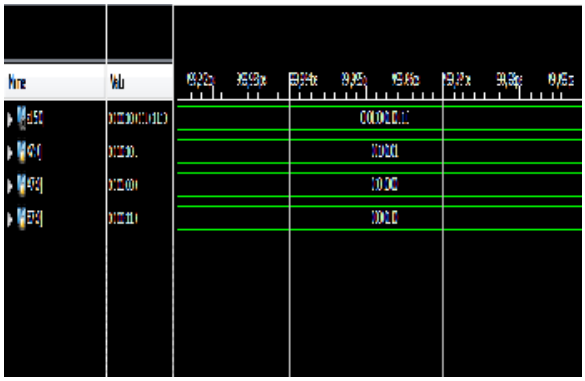


Fig. 11 FAM unit with CSA adder for even bit

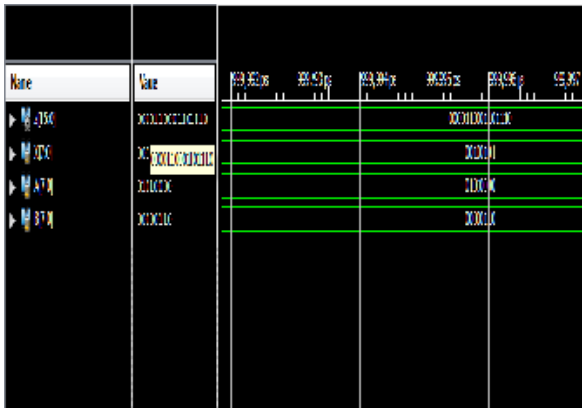


Fig.12 FAM unit with CSA adder for bit bit

The power consumption of the two algorithm is analysed in Xilinx using XP power analyser. The values of power in case of FAM with CLA is same for both even and odd bit shown in Fig. 13 and 14 of FAM unit with CLA adder.

On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	8	1920	0
Signals	0.000	52	...	...
MULTs	0.000	1	4	25
IOs	0.004	40	66	61
Leakage	0.034			
Total	0.038			

Fig. 13 Power of FAM with CLA adder

On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	8	1920	0
Signals	0.000	52	---	---
MULTs	0.000	1	4	25
IOs	0.000	40	66	61
Leakage	0.034			
Total	0.034			

Fig. 14 power of FAM with CSA adder

From the fig.13 provides the power value in case of FAM unit with carry skip adder. From the simulation results and power analysis it is found that the FAM with Carry skip adder consumes 0.034 W which is 0.004 W less than that of the FAM with CLA adder.

6. CONCLUSION

The proposed fused add multiply algorithm is an efficient arithmetic algorithm in which the delay and complexity of the system is reduced. The proposed algorithm uses fused add multiply unit which possess binary addition and multiplication is simpler by generating the partial product. The modified booth algorithm use large area for arithmetic process and computation time is longer whereas the fused add multiply algorithm generate the partial products with less computation time. A wallance tree is used in fused add multiply algorithm that acts as the carry save adder to increase the speed of multiplication process and reduce in area.

The fused add multiply unit consumes less power than the other existing codes. The major component of FAM is multi-bit adder whose design will have a significant impact on the overall performance of the FAM system. The FAM with CLA adder uses 0.038W POWER while the FAM with CSA adder uses only 0.034W of power. From the proposed models, it is found that the FAM with CSA adder consumes power which is 0.004W less than the one with CLA adder. Further the power can be reduced by replacing the CLA adder or CSA adder with more advanced adder.

REFERENCES

[1] Kostas Tsoumanis, Sotiris Xydis and Kaimal Pekmestzi "An Optimized Modified Booth Recoder for efficient design of the Add- Multiply Operator, " IEEE Trans. Vol. 61,No. 4, April 2014.
[2] Paladugu Srinivas Teja " Design of radix-8 Booth Multiplier Using Koggestone Adder For High Speed Airihmetic Applications " EEIEJ, Vol.1, No. 1, February 2014.
[3] Minu Thomas "Design and Simulation of Radix-8 Booth Encoder Multiplier for Signed and Unsigned Numbers, " IJSRD, vol.1, Issue 4, 2013.
[4] A.Amaricai, M. Vladutiu, and O. Boncalo, "Design issues and implementations for floating-point divide-add fused," IEEE Trans. Circuits Syst. II–Exp. Briefs, vol. 57, no. 4, pp. 295–299, Apr. 2010.
[5] E. E. Swartzlander and H. H. M. Saleh, "FFT implementation with fused floating-point operations," IEEE Trans. Comput., vol. 61, no. 2, pp. 284–288, Feb. 2012.
[6] Y.-H. Seo and D.-W. Kim, "A new VLSI architecture of parallel multiplier–accumulator based on Radix-2 modified Booth algorithm," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 2, pp. 201–208, Feb. 2010.
[7] N. H. E. Weste and D. M. Harris, "Datapath subsystems," in CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Readington: Addison-Wesley, 2010, ch. 11.
[8] Z. Huang andM. D. Ercegovac, "High-performance low-power left-toright array multiplier design," IEEE Trans. Comput., vol. 54, no. 3, pp.272–283, Mar. 2005.
[9] R. Zimmermann and D. Q. Tran, "Optimized synthesis of sum-of-products," in Proc. Asilomar Conf. Signals, Syst. Comput., Pacific Grove, Washington, DC, 2003, pp. 867–872.

