

A LOW POWER FUSED ADD-MULTIPLY DESIGN BASED ON BOOTH RECORDER

Vidya Baby

PG scholar
Department of Electronics and
Communication Engineering
Sree Sakthi Engineering College
Coimbatore
mail:vidyanbaby90@gmail.com

S.Prabhavathy

Assistant Professor
Department of Electronics and
Communication Engineering
Sree Sakthi Engineering College
Coimbatore
mail:mevlissec@gmail.com

ABSTRACT

Digital Signal Processing (DSP) applications involves numerous advanced operations. This paper proposes a new advancement in the design of Fused Add-Multiply operator, which finds its extensive use in many of the such DSP applications especially in arithmetic circuits. This paper includes the techniques to implement the direct recoding of sum of two externally given input signals to their corresponding Modified Booth (MB) form. This particular representation introduces a structured and efficient recoding technique by suggests that of three totally different schemes by incorporating them in FAM design. As compared to all the existing methods, this new technique yields wide reductions in terms of the critical path delay, hardware complexity and power consumption of the FAM unit.

General Terms

Digital Signal Processing, Fused Add-Multiply operator, recoding,

Keywords

Fused Add-Multiply operator, Modified Booth encoding, arithmetic circuits.

1. INTRODUCTION

Digital Signal Processing (DSP) applications realize in depth use in modern consumer market providing custom accelerators for the domains of the multimedia systems, communications etc.. Typical Digital Signal Processing (DSP) applications involves an outsized range of arithmetic operations as their implementation is predicted on computationally intensive areas such as Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), Finite Impulse Response (FIR) filters and the signals convolution. The performance of such Digital Signal Processing (DSP) systems is inherently laid low with the choices of their style relating to the design and allocation of the arithmetic units. Recent studies shows that the design of arithmetic components combining operations which share data, can lead to the better performance improvement. Based on the results that an addition can often be subsequent to a multiplication, which is

used in the Multiply-Accumulator (MAC) and Multiply-Add (MAD) units were leading to the more efficient implementations of DSP algorithms compared to the conventional one. Now most of the DSP applications are based on Add-Multiply (AM) operations. The simplest design of the AM unit, by first assigning an adder and then giving its output to the input of a multiplier, increases significantly both area and critical path delay of the circuit. Focusing an optimized design of AM operators, fusion techniques are employed based on the direct recoding of the sum of two numbers in its Modified Booth (MB) form. Thus, the carry-propagate (or carry-look-ahead) adder of the conventional AM design is avoided resulting in considerable gains of performance. There is signed-bit MB recoder which transforms redundant binary inputs to their MB recoding form. In the recent days, we have the technique for the design of high performance flexible coprocessor architectures targeting the computationally intensive DSP applications. These optimized designs will results in improvements in both area and critical path. An another method is the recoding of a redundant input from its carry-save form to the corresponding borrow-save form keeping the critical path of multiplication operation fixed. This paper focuses on the efficient design of FAM operators, focusing the optimization of the recoding scheme for direct shaping of the MB form of the sum of two numbers (Sum to MB - S-MB). We evaluated the performance of the projected S-MB technique by scrutiny its three completely different schemes with the existing schemes.

2. RELATED WORKS

This section includes the works already done by various researchers on this particular topic using different methods and algorithms. Following is the brief of some of them. In paper[3], author introduces the optimization of the bit-width and hardware resources without sacrificing the frequency response and output signal precision. In [7], this paper proposes the implementation of the Modified Booth Algorithm (Radix-4) and its comparison with the Booth Algorithm (Radix-2). From the comparison, they came in to the conclusion that the Modified Booth Algorithm which employs both the addition

and subtraction and also treats both the positive and negative operands uniformly, apart from the Radix-2 Booth Algorithm for the signed numbers. Paper [4], proposes the implementation of the Fast Fourier Transform (FFT) implementation with fused implementation with fused floating point operations for the efficient implementation of the Fast Fourier Transform (FFT) processor. In [9], this paper proposes the floating point divider for the interval arithmetic. This provides an interval arithmetic provides an efficient method for monitoring and controlling errors in numerical calculations and can be used to solve problems that cannot be efficiently solved with the floating point arithmetic by means of the hardware design. Paper [5], proposes a new VLSI architecture of parallel-multiplier-accumulator based on radix-2 modified booth algorithm. It is based on the algorithm that multiplies two signed-binary numbers in two's complement notation.

3. PROPOSED METHOD

3.1. Overview

This paper focus on the efficient design of AM units which implement the operation $Z=X(A+B)$. An optimized design of the AM operator is based on the fusion of the adder and the MB encoding unit within a single data path block by direct recoding of the sum $Y=A+B$ to its MB format. This fused Add -Multiply (FAM) circuitry contains only one adder at the end. A better area reduction, the least path critical delay etc. are ensured in the proposed scheme. For the direct recoding of two numbers in the MB representation of their corresponding sum, three new different schemes are proposed. H we aree adding a correction term CT to the CSA adder tree to correct the errors present in the intermediate stages, but it is optional, since the error rate will be less.

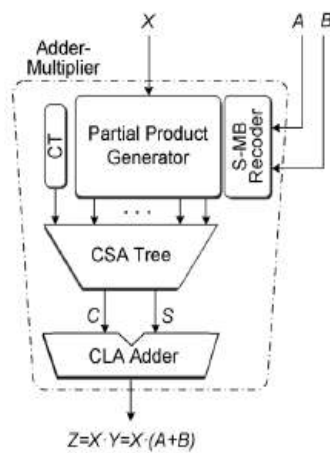


Fig 1: AM operator based on the fused design with direct recoding of the sum of A and B in its MB representation

3.2 Modified Booth Form

Modified Booth (MB) is assigned-digit 2's complement based radix-4 encoding technique. Its main advantage is that it reduces by half the number of partial products in multiplication comparing to all other radix-2 representation. Digits y_j^{MB} belongs to $\{-2, -1, 0, +1, +2\}$. 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 a 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 and the corresponding table is given below.

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

The Modified Booth Encoding table shows all possible combinations inputs so as to get the corresponding Modified Booth(MB) form having specially designed variables say s, one and two as shown in Table 1, these signals are used for the generation of the partial products by making use of the basic logic gates. Whenever we are giving the binary signals as the input, it will produce the MB signals as our output. Based on this table we can directly convert the binary input signal to their corresponding MB digit format. Internally it is done by a encoder-decoder circuitry, which in turn can be obtained from this Modified Booth Encoding table itself.

Table 1. Modified Booth Encoding Table

| BINARY | | | y_j^M | MB ENCODING | | | INPUT CARRY $c_{in,j}$ |
|------------|----------|------------|---------|-------------|--------------------|--------------------|------------------------|
| y_{2j+1} | y_{2j} | y_{2j-1} | | sign $=s_j$ | $\times 1 = one_j$ | $\times 2 = two_j$ | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | +1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | +1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | +2 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | -2 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | -2 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | -1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

The signals are represented by the relation,

$$one_j = y_{2j-1} \text{ XOR } y_{2j} \quad (2)$$

$$two_j = (y_{2j+1} \text{ XOR } y_{2j}) \cdot (\text{NOT } one_j) \quad (3)$$

$$\text{and } s_j = y_{2j+1} \quad (4)$$

The gate level schematic implementation of the signal is represented by the help of the basic logic gates say, the XOR gate as well as the bubbled AND gate, and the

structure is shown below.

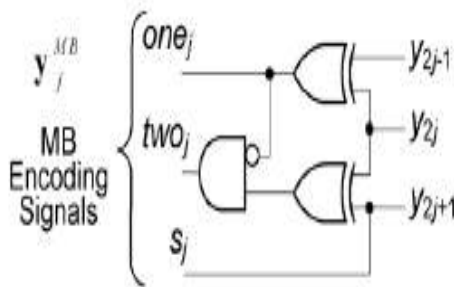


Fig 2 : Gate level schematic for the implementation of the MB encoded signals.

3.3. Partial Product Generation

For the implementation of the Fused Add Multiply (FAM) design the multiplier is a parallel one based on the MB algorithm. To do the FAM implementation say ,X.Y. The partial product generated at the i-th bit P_{ij} of the partial product PP_{ij} is based on the expression given by,

$$P_{ij} = ((x_i \text{ XOR } s_j) \text{ AND } one_j) \text{ OR } ((x_{i-1} \text{ XOR } s_j) \text{ AND } two_j) \quad (5)$$

After the partial products are generated, they are added, prop-erly weighted, through a Wallace Carry-Save Adder (CSA) tree. The partial products can be generated by the following way as shown in the given circuit, by making use of basic logic gates.

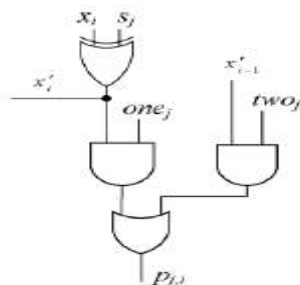


Fig 3: Generation of partial products for the MB multiplier.

3.4. Sum To Modified Booth Encoding

It uses specially designed half-adders and full-adders for the implementation of the proposed scheme as described below.

3.4.1. Specially Designed Signed-Bit Full Adders and Half Adders for the Structured Signed Bit Arithmetic

It is the major part to design a set of bit-level signed Half Adders and Full Adders apart from

the conventional HA and FA, to recode the sum of two consecutive bits of the input A and B for the even as well as odd bits respectively. The separate truth tables and the corresponding design of the newly designed Half Adders and Full Adders are given below: The Boolean expression of the HA* is given by,

$$c = p \text{ OR } q \quad (6)$$

$$s = p \text{ XOR } q \quad (7)$$

the truth tables as well as the schematic representations of all the other specially designed components are given below. Here the selection of each component will depend upon the sign of the input as well as the output signals.

Table 2
Truth Table Of HA*

| inputs | | Output value | outputs | |
|--------|-------|--------------|---------|-------|
| P (+) | q (+) | | c (+) | s (-) |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | +1 | 1 | 1 |
| 1 | 0 | +1 | 1 | 1 |
| 1 | 1 | +2 | 1 | 1 |

$$\text{Output value} = 2.c - s = p + q$$

Considering that p, q are the binary inputs and c, s are the outputs of a HA* which implements the relation $2.c - s = p + q$ where s is considered negatively signed, the output takes on of the values {0, +1, +2} we can apply this for its dual also , as shown below:

Table 3
Truth Table Of Dual Of HA*

| inputs | | Output value | outputs | |
|--------|-------|--------------|---------|-------|
| P (-) | q (-) | | c (-) | s (+) |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | -1 | 1 | 1 |
| 1 | 0 | -1 | 1 | 1 |
| 1 | 1 | -2 | 1 | 0 |

$$\text{Output value} = 2.c - s = p + q$$

Table 4
Truth Table Of HA**

| inputs | | Output value | outputs | |
|--------|-------|--------------|---------|-------|
| P (-) | q (+) | | c (+) | s (-) |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | +1 | 1 | 1 |
| 1 | 0 | -1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 |

$$\text{Output value} = 2.c - s = -p + q$$

Above figure shows the operation of HA** which implements the relation $2.c - s = -p + q$, by considering p and q as the binary inputs and c as well as s as the sum as well as carry signal respectively. The operation is given by the expression $2.c - s = -p + q$ and it will manipulates a negative and a positive input resulting in the output values $\{-1, 0, +1\}$. The schematic for the above structure is given below.

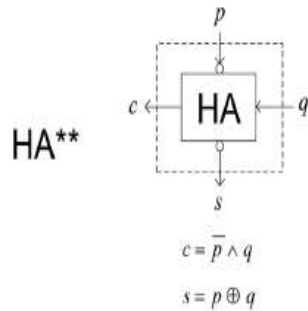


Fig 4: Boolean equations and schematics of HA**

In the same way the schematics and the Boolean equations of the FA* and FA** are given below by means of the basic logic gates.

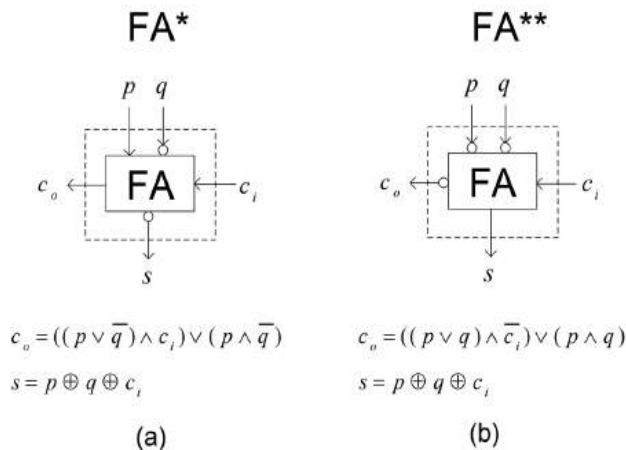


Fig 5: Boolean equations and schematics for signed (a) FA* and (b) FA**

Table 5
Truth Table Of FA*

| inputs | | | Output Value | Outputs | |
|--------|-------|--------|--------------|---------|-------|
| p (+) | q (-) | ci (+) | | co (+) | 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 = $2.c - s = p - q + c_i$

Table 6
Truth Table Of FA**

| Inputs | | | Output Value | Outputs | |
|--------|-------|--------|--------------|---------|-------|
| p (-) | q (-) | ci (+) | | co (+) | 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 = $-2.c - s = p - q + c_i$

3.4.2. Proposed S-MB Recoding Techniques

By exploring three different alternative schemes of the S-MB recoding technique, each of which can be applied in either signed or unsigned numbers which consist of even or odd bits. The three different recoding schemes are as follows. All these schemes are making use of HA, HA*, FA as well as FA** for its operation to perform for the even and odd bits. All this S-MB1, S-MB2 and S-MB3 are designated depending on the sign of the input as well as output signals. In all the three schemes, the input signals will be the same eight bit inputs say X,A as well as B. The difference comes only for the sign of the inputs as well as the outputs respectively. Here it involves the area as well as the power considerations too.

3.4.2.1. S-MB 1 Recoding Scheme

S-MB1 scheme makes use of FA, FA* as well as FA** for its operation for even and odd bits respectively.

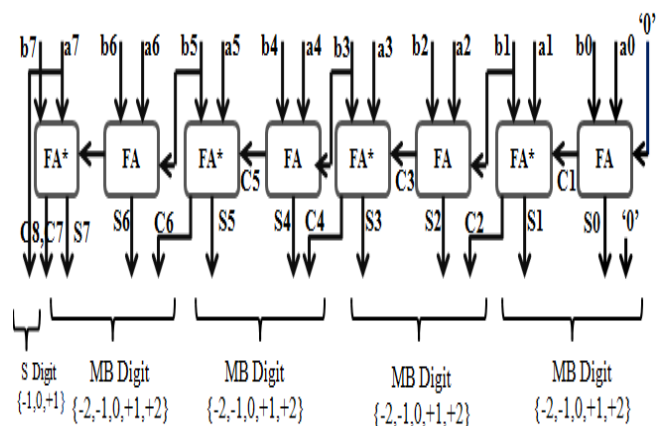


Fig 6: S-MB 1 recoding scheme for even number of bits.

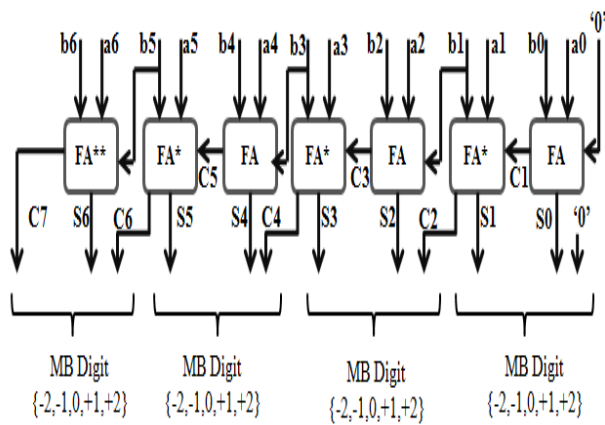


Fig 7: S-MB1 recoding scheme for odd number of bits.

3.4.2.2. S-MB2 Recoding Scheme

S-MB 2 scheme makes use of FA and FA* and FA** for its operation for the even and odd bits respectively.

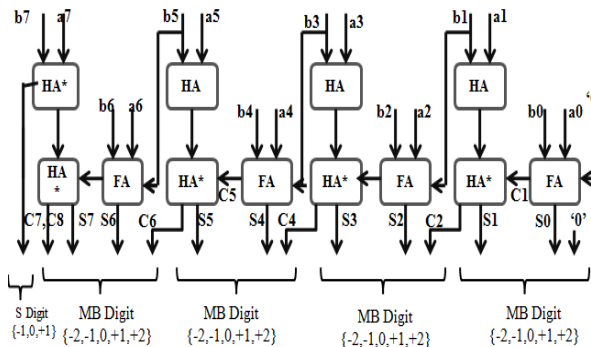


Fig 8: S-MB2 recoding scheme for even number of bits

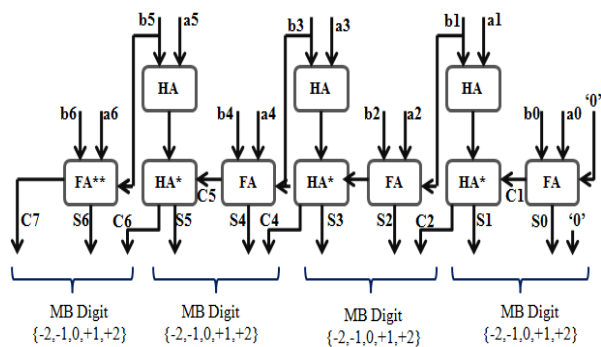


Fig 9: S-MB2 recoding scheme for odd number of bits.

3.4.2.3. S-MB3 Recoding Scheme

S-MB 3 scheme make use of HA*, HA** as well as FA for its operation for the odd and evenbits.

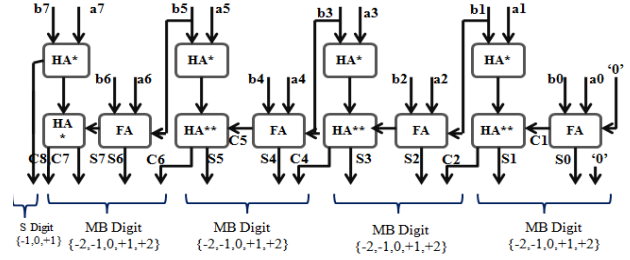


Fig 10: S-MB3 recoding scheme for even number of bits.

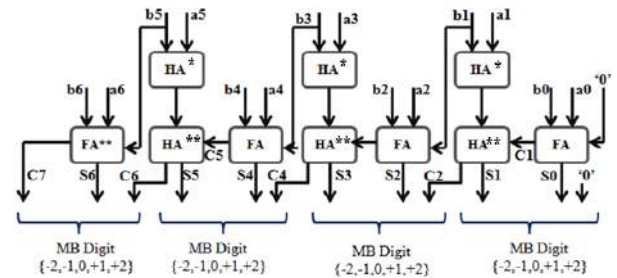


Fig 11: S-MB3 recoding scheme for odd number of bits.

4. RESULTS AND DISCUSSION

4.1 Simulation Environment

By making use of the waveforms, it is very easier to evaluate the effectiveness of the proposed architecture through architectural simulation. Model sim is used for simulation and Xilinx is used for evaluating the time, area and power for the existing and proposed architecture.

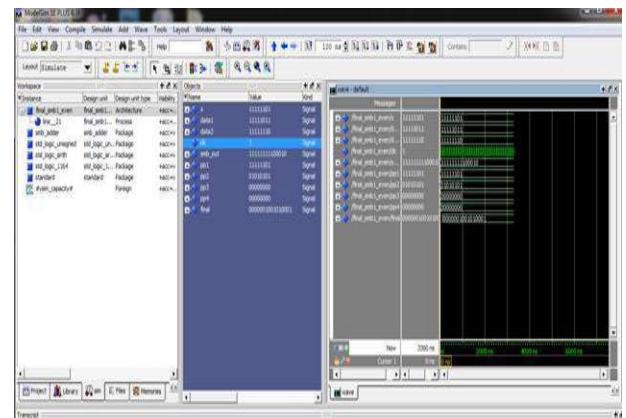


Fig 12: Simulation result of S-MB based scheme

4.2. PERFORMANCE EVALUATION

A comparison study among the three different schemes in terms of the area complexity and critical delay among the three different schemes are given below.

Table 7
Performance Comparison Of The Proposed
Recoding Schemes With Existing Scheme

| | | Power (mW) | Time Delay(nS) | Area (total gate count) |
|--------------------|-------|---------------|-------------------|----------------------------------|
| Existing method | | 172 | 11.024 | 1468 |
| Proposed method | S-MB1 | 139 | 11.014 | 1378 |
| | S-MB2 | 152 | 10.854 | 1354 |
| | S-MB3 | 146 | 10.854 | 1354 |

5. CONCLUSION

For optimizing the design of the fused-add multiply three new different schemes are proposed, for the direct recoding of the sum of two numbers in its MB form. When compared to the existing recoding schemes, the proposed schemes deliver considerable improvements in both area, delay and power consumption. Regarding the over all performance S-MB2 and S-MB3 least delay and it uses the least amount of the gate count or area used. The S-MB1 based scheme will suits for the low power applications where the power consumption is very less.

ACKNOWLEDGEMENT

I would like to thank my guide and all the reviewers for their constructive comments and active support to my work.

REFERENCES

[1] Kostas Tsoumanis, Sotiris Xydis, Constantinos Efstathiou, Nikos Moschopoulos, and Kiamal Pekmestzi, "An Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Operator" iee transactions on circuits and systems—i: regular papers, vol. 61, no. 4, april 2014.

[2] Basant Kumar Mohanty and Sujit Kumar Patel, "Area-Delay-Power efficient Carry-Select adder", IEEE Trans. On circuits and Systems, vol. 61, no. 6, jan. 2014, pp. 418-422.

[3] Shen-Fu Hsiao, Jun-Hong Zhang Jian, and Ming-Chih Chen, "Low-cost FIR filter designs based on faithfully rounded truncated multiple constant multiplication/Accumulation" IEEE transactions on circuits and systems—ii: express briefs, vol. 60, no. 5, may 2013.

[4] 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.

[5] Y.H.Seo and D.W.Kim, " A New VLSI architecture of parallel Multiplier based on Radix-2 modified booth algorithm," IEEE Trans. Very Large Scale Inter. (VLSI) Syst., vol.18, no.2, pp.201-208, Feb.2010.

[6] S.Nikolaïdis, E.Karaolis, and E.D.Kyriakis-Bitzaros, "Estimation of signal transition activity in FIR filters implemented by a MAC architecture," IEEE Trans. Comput.- Aided Des. Integer. Circuits Syst., vol.19, no.1, pp.164-169, jan.2000.

[7] Sukhmeet Kaur¹, Suman² and Manpreet Signh Manna³, "Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix-2)", Advance in Electronic and Electric Engineering. ISSN 2231-1297, Volume 3, Number 6 (2013), pp. 683-690.

[8] 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.

[9] 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.

[10] 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.