ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

Volume 7, Issue 7, July 2020

Design and Implementation Radix-8 High Performance Multiplier Using High Speed Compressors

M.Satheesh

Dept. of Electronics and Communication Engineering, Siddartha Educational Academy Group of Institutions, Tirupathi, India

ABSTRACT: This paper presents an area efficient implementation of a high performance parallel multiplier. Radix-4 Booth multiplier with 3:2 compressors and Radix-8 Booth multiplier with 4:2 compressors are presented here. The design is structured for m × n multiplication where m and n can reach up to 126 bits. Carry Lookahead Adder is used as the final adder to enhance the speed of operation. Finally the performance improvement of the proposed multipliers is validated by implementing a higher order FIR filter. The design entry is done in Verilog HDL and simulated using ModelSim SE 6.4 design suite from Mentor Graphics. It is then synthesized and implemented using Xilinx ISE 14.3 targeted towards Spartan 3 FPGA.

I. INTRODUCTION

With the rapid advances in multimedia and communication systems, real-time signal processing and large capacity data processing are increasingly being demanded. The multiplier is an essential lament of the digital signal processing such as filtering and convolution. Most digital signal rocessing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete avelet transform (DWT). As they are basically accomplished by repetitive application of multiplication and addition, their speed becomes a major factor which determines the performance of the entire calculation. Since the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined more by the multiplier. Furthermore, multiplier consumes much area and dissipates more power. Hence designing multipliers which offer either of the following design targets – high speed, low power consumption, less area or even a combination of them is of substantial research interest. Multiplication operation involves generation of partial products and their accumulation. The speed of multiplication can be increased by reducing the number of partial products and/or accelerating the accumulation of partial products. Among the many methods of implementing high speed parallel multipliers, there are two basic approaches namely Booth algorithm and Wallace Tree compressors. This paper describes an efficient implementation of a high speed parallel multiplier using both these approaches. Here two multipliers are proposed. The first multiplier makes use of the Radix-4 Booth Algorithm with 3:2 compressors while the second multiplier uses the Radix-8 Booth algorithm with 4:2 compressors. The design is structured for m x n multiplication where m and n can reach up to 126 bits. The number of partial products is n/2 in Radix-4 Booth algorithm while it gets reduced to n/3 in Radix-8 Booth algorithm. The Wallace tree uses Carry Save Adders (CSA) to accumulate the partial products. This reduces the time as well as the chip area. To further enhance the speed of operation, carry-look-ahead (CLA) adder is used as the final adder.

II. ARCHITECTURE

The architecture of the proposed multiplier is shown in Fig. It consists of four major modules: Booth encoder, partial product generator, Wallace tree and carry look-ahead adder[4]. The Booth encoder performs Radix-2 or Radix-4 encoding of the multiplier bits. Based on the multiplicand and the encoded multiplier, partial products are generated by the generator. For large multipliers of 32 bits, the performance of the modified Booth algorithm is limited. So Booth recoding together with Wallace tree structures have been used in the proposed fast multiplier. The partial products are supplied to

ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

Volume 7, Issue 7, July 2020

Wallace Tree and added appropriately. The results are finally added using a Carry Look-ahead Adder (CLA) to get the final product.

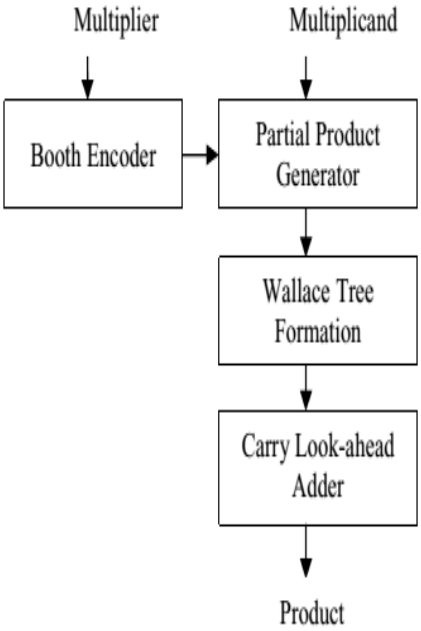


Figure Block diagram of Wallace Booth Multiplier

Radix – 4 Booth Algorithm

Booth algorithm is a powerful algorithm [5] for signed number multiplication, which treats both positive and negative numbers uniformly. Since a k-bit binary number can be interpreted as k/2-digit Radix-4 number, a k/3-digit Radix-8 number and so on, it can deal with more than one bit of the multiplier in each cycle by using high radix multiplication[6]. The major disadvantage of the Radix-2 algorithm was that the process required n shifts and an average of n/2 additions for an n bit multiplier. This variable number of shift and add operations is inconvenient for designing parallel multipliers. Also the Radix-2 algorithm becomes inefficient when there are isolated 1's. The Radix-4 modified Booth algorithm overcomes all these limitations of Radix-2 algorithm. For operands equal to or greater than 16 bits, the modified Radix-4 Booth algorithm has been widely used. It is based on encoding the two's complement multiplier in order to reduce the number of partial products to be added to

n/2. The multiplier, Y in two's complement form can be written as in

$$Y = -Y_{n-1}\,2^{n-1} + \Sigma_i\, Y_i 2^i \quad ; \quad 0 \le i \le n-2 \qquad (1)$$

It can be written as

$$Y = \Sigma_i (-2\,Y_{2i+1} + Y_{2i} + Y_{2i-1})\,2^{2i} \; ; 0 \le i \le n-2 \qquad (2)$$

Table I shows the encoding of the signed multiplier Y, using the Radix-4 Booth algorithm. Radix-4 Booth recoding encodes multiplier bits into [-2, 2]. Here we consider the multiplier bits in blocks of three, such that each block overlaps the previous block by one bit. It is advantageous to begin the examination of the multiplier with the least significant bit. The overlap is necessary so that we know what happened in the last block, as the most significant bit of the block acts like a sign bit.

ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

Volume 7, Issue 7, July 2020

TABLE I. RADIX - 4 BOOTH RECODING

Multiplier Bits Y_{2i-1}	Y_{2i}	Y_{2i+1}	Recoded Operation on multiplicand, X
0	0	0	0X
0	0	1	+X
0	1	0	+X
0	1	1	+2X
1	0	0	-2X
1	0	1	-X
1	1	0	-X
1	1	1	0X

Radix 8 Booth Algorithm

Radix-8 Booth recoding applies the same algorithm as that of Radix-4, but now we take quartets of bits instead of triplets. Each quartet is codified as a signed digit using Table II. Radix-8 algorithm reduces the number of partial products to n/3, where n is the number of multiplier bits. Thus it allows a time gain in the partial products summation.

TABLE II. RADIX -8 BOOTH RECODING

Multiplier Bits Y_{i+2}	Y_{i+1}	Y_i	Y_{i-1}	Recoded Operation on multiplicand, X
0	0	0	0	0X
0	0	0	1	+X
0	0	1	0	+X
0	0	1	1	+2X
0	1	0	0	+2X
0	1	0	1	+3X
0	1	1	0	+3X
0	1	1	1	+4X
1	0	0	0	-4X
1	0	0	1	-3X
1	0	1	0	-3X
1	0	1	1	-2X
1	1	0	0	-2X
1	1	0	1	-1X
1	1	1	0	-1X
1	1	1	1	0X

Rdix-8 Booth encoder Table

Wallace Tree

In a Wallace tree multiplier the addition of bits within one column is rearranged still further. Let's first re-visit the columns involved in the computation of a product. In a Wallace tree multiplier the addition of bits within one column is rearranged still further. Let's first re-visit the columns involved in the computation of a product. Each column is characterised by the inputs to that column, and the outputs from that column. The inputs to a column are the bits of the

ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

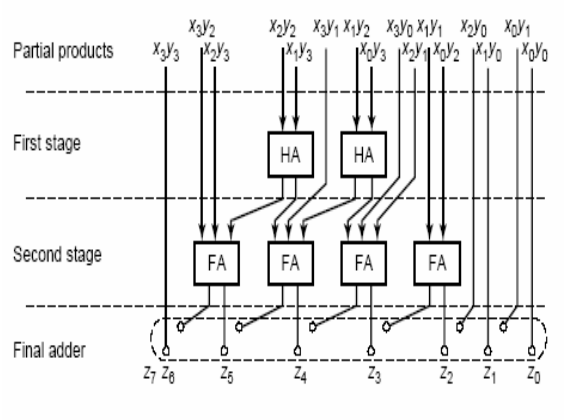
(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

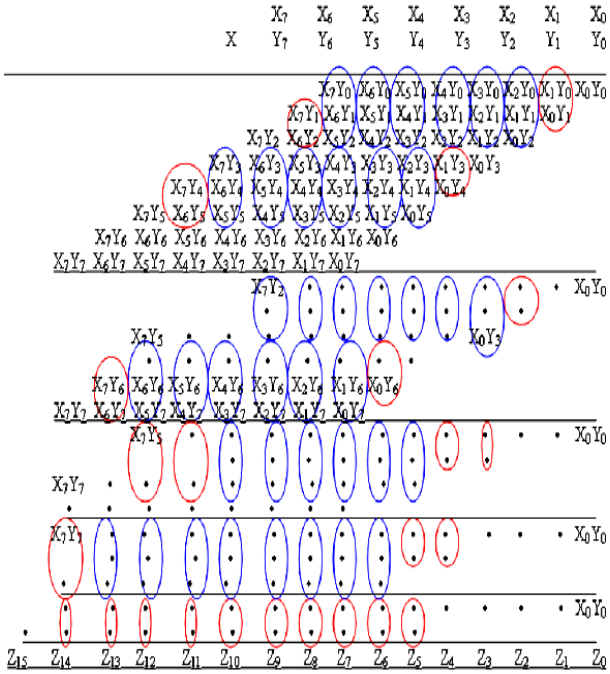
Volume 7, Issue 7, July 2020

partial product (Booth or non-Booth encoded) plus. The carry bits from one column to the right plus. The sum bits that are generated within the column. The outputs from a column are the carry bits to the column one to the left plus. The last two sum bits in that column that are passed to the CLA. All bits from partial products are available at the same time. So in the Wallace tree multiplier we use a tree of adder cells. The carry in comes CSA fashion from the previous column. The carry out goes CSA fashion to the next column. In comparison to the basic array multiplier, the delay from partial products to final sum bits in a column is O(ln(n)) rather than O(n).

A fast process for multiplication of two numbers was developed by Wallace. By using the Wallace method, a three step process is used for the multiplication of two numbers; the bit products are formed. The bit product matrix are reduced to a two row matrix, where sum of the row equal to the sum of bit products, and two resulting rows of the bit product are summed with a fast adder(compressor) to produce a final product.The basic Wallace tree multiplier structure. In the Wallace Tree method, three single bit signals are passed to a one bit full adder which is called a three input Wallace Tree circuit, and the output signal (sum) is supplied to the next stage full adder of the same bit, and the carry output signal thereof is passed to the next stage full adder of the same no of bit, and the carry output signal thereof is supplied to the next stage of the full adder located at a one bit higher position.



Wallace tree multiplier



Conventional 8x8 bit Wallace Tree Multiplier

ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

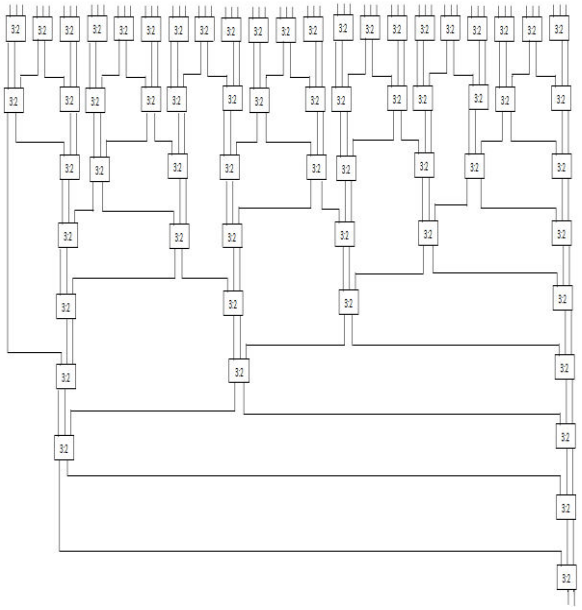
(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

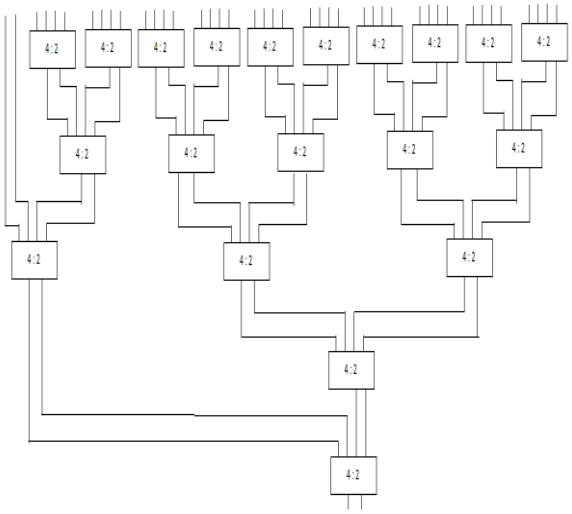
Volume 7, Issue 7, July 2020

The 14 T full adders are used for the existing design of the Wallace tree multiplier. By using this full adder circuit, the power consumption is high and the delay is increased during the partial product addition stage.Compressor is mainly used to reduce the power consumption during the partial product addition stage and the critical path delay is also highly reduced.

The Wallace tree method is used in high speed designs in order to produce two rows of partial products that can be added in the last stage. Also critical path and the number of adders get reduced when compared to the conventional parallel adders. Here the Wallace tree has taken the role of accelerating the accumulation of the partial products. Its advantage becomes more pronounced for multipliers of greater than 16 bits .The speed, area and power consumption of the multipliers will be in direct proportion to the efficiency of the compressors. The Wallace tree structure with 3:2 compressors and 4:2 compressors is shown in Fig and Fig respectively.
In this regard, we can expect a significant reduction in computing multiplications.



Wallace Tree using 3:2 compressors



Wallace Tree using 4:2 compressors

ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

Volume 7, Issue 7, July 2020

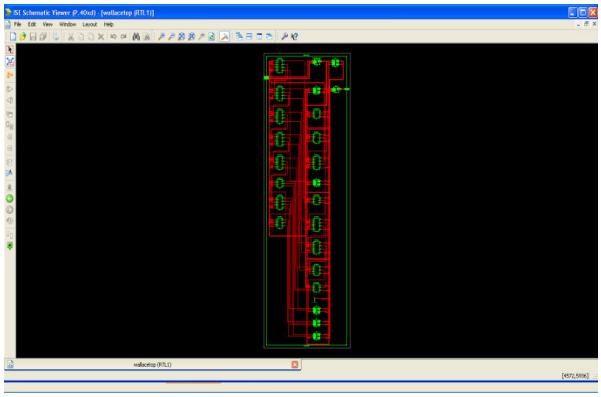
The 3:2 compressors make use of a carry save adder .The carry save adder outputs two numbers of the same dimensions as the inputs, one is a sequence of partial sum bits and other is a sequence of carry bits. In carry save adder, the carry digit is taken from the right and passed to the left, just as in conventional addition; but the carry digit passed to the left is the result of the previous calculation and not the current one. So in each clock cycle, carries only have to move one step along and the clock can tick much faster. Also the carry-save adder produces all of its output values in parallel, and thus has the same delay as a single full-adder. The 4:2 compressors have been widely employed in the high speed multipliers to lower the latency of the partial product accumulation stage. A 4:2compressor can be built using two 3:2 compressors. Owing to its regular interconnection, the 4:2 compressors is ideal for the construction of regularly structured Wallace Tree with low complexity.

III. DESIGN SUMMARY

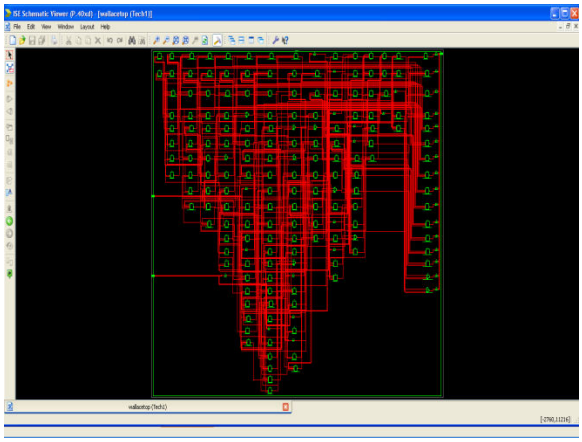
Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	89	4656	1%
Number of 4 input LUTs	158	9312	1%
Number of bonded IOBs	32	232	13%

IV. SIMULATION RESULTS

RTL schematic



Technology Schematic



ISSN: 2395-7639

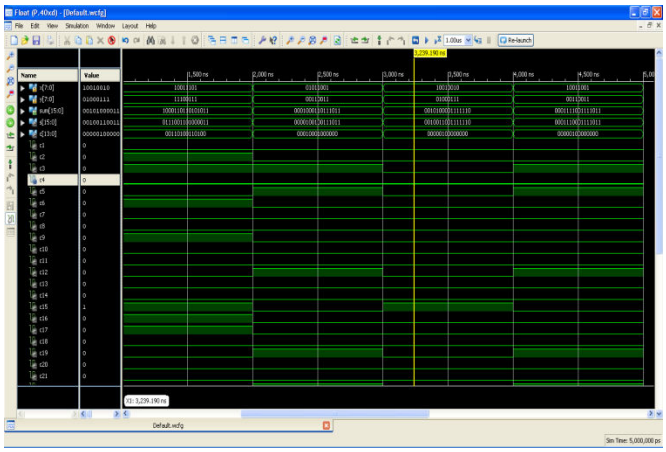
International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

(A Monthly, Peer Reviewed Online Journal)

Visit: www.ijmrsetm.com

Volume 7, Issue 7, July 2020

SIMULATION RESULTS



REFERENCES

[1] Dong-Wook Kim, Young-Ho Seo, "A New VLSI Architecture of Parallel Multiplier-Accumulator based on Radix-2 Modified Booth Algorithm", Very Large Scale Integration (VLSI) Systems, IEEE Transactions, vol.18, pp.: 201-208, 04 Feb. 2010

[2] Prasanna Raj P, Rao, Ravi, "VLSI Design and Analysis of Multipliers for Low Power", Intelligent Information Hiding and Multimedia Signal Processing, Fifth International Conference, pp.: 1354-1357, Sept. 2009

[3] Lakshmanan, Masuri Othman and Mohamad Alauddin Mohd.Ali, "High Performance Parallel Multiplier using Wallace-Booth Algorithm",
Semiconductor Electronics, IEEE International Conference , pp.: 433- 436, Dec. 2002.

[4] Jan M Rabaey, "Digital Integrated Circuits, A Design Perspective", Prentice Hall, Dec.1995

[5] Louis P. Rubinfield, "A Proof of the Modified Booth's Algorithm for Multiplication", Computers, IEEE Transactions,vol.24, pp.: 1014-1015, Oct. 1975

[6] Rajendra Katti, "A Modified Booth Algorithm for High Radix Fixedpoint Multiplication", Very Large Scale Integration (VLSI) Systems, IEEE Transactions, vol 2, pp.: 522-524, Dec. 1994.

[7] C. S. Wallace, "A Suggestion for a Fast Multiplier", Electronic Computers, IEEE Transactions, vol.13, Page(s): 14-17, Feb. 1964

[8] Hussin R et al , "An Efficient Modified Booth Multiplier Architecture", IEEE International Conference, pp.:1-4, 2008.