A Review Paper on Comparison of Multipliers based on Performance Parameters

Savita Nair
PG Student
PIIT, Panvel
India

Ajit Saraf
Assistant Professor
PIIT, Panvel
India

ABSTRACT

Among all the arithmetic operations that exist, a processor consumes most of its time and hardware resources in carrying out multiplication when compared to other operations like addition and subtraction. In this paper comparative study is done of four multipliers namely, Array multiplier, Modified booth multiplier, Wallace tree multiplier and modified Booth-Wallace tree multiplier based of various performance parameters like speed, area, power consumed and circuit complexity. It is always necessary to design a fast multiplier in VLSI so as to enhance system performance.

General Terms

Partial Product Generation, Partial Product Reduction

Keywords

Array Multiplier, Modified Booth Multiplier, Modified Booth-Wallace tree Multiplier, Wallace tree Multiplier

1. INTRODUCTION

A multiplier is one of the main blocks in most of the digital signal processing (DSP) systems. The various DSP applications where a multiplier plays a vital role include digital filtering, digital communications and many more. Computational performance of a DSP system is limited by its multiplication performance [9] and since, multiplication dominates the execution time of most DSP algorithms [3], and therefore high-speed multiplier is much desired [4]. With an ever-increasing quest for greater computing power on battery-operated mobile devices, design emphasis has shifted from optimizing conventional delay time, area size to minimizing power dissipation while still maintaining the high performance [5]. Hence designing of multipliers that offer any of the following design targets – high speed, low power consumption, less area or even a combination of them is of great interest [1]

The architecture of multiplier can be divided into three stages namely, partial product generation stage, partial product reduction stage and the final addition of the reduced partial products stage. Traditionally shift and add algorithm had been used but however this is not suitable for fast multiplication in VLSI from delay point of view as more adders will be needed. The speed of multiplication can be increased by reducing the number of partial products or by increasing the speed of accumulation of partial products. Multibit compressors can be used for reducing the number addition stages of partial products.

Multipliers can be classified into two, serial and parallel multipliers. In serial multiplier, each bit of multiplier is used for evaluating the partial products whereas in parallel multipliers, partial product from each bit of multiplier is computed in parallel. The main parameter that determines the performance of parallel multipliers is the number of partial products that is to be added. In serial-parallel multipliers speed is compromised to achieve better performance in terms of area and power consumption. Depending on the type of application parallel or serial multiplier can be used. Modified Booth algorithm is one of the most popular algorithms used for reducing the number of partial products. In Wallace Tree algorithm the number of sequential adding stages is reduced which thereby helps for achieving improved speeds. The combination of Modified Booth algorithm and Wallace Tree technique can provide an advantage of both the algorithms in one multiplier.

In the following sections, the four multipliers namely, Array multiplier, Modified Booth multiplier, Wallace tree multiplier and modified Booth-Wallace tree multiplier are compared based on different performance parameters.

2. ARRAY MULTIPLIERS

Array multiplier is a regular structured multiplier. This multiplier performs multiplication by the standard add and shift operation using “add and shift” algorithm. The partial products are generated by multiplying the multiplicand with each bit of the multiplier. The partial products obtained are then shifted depending on their bit orders and are finally added at the last stage. The numbers of partial products that are generated is equals the number of multiplier bits.

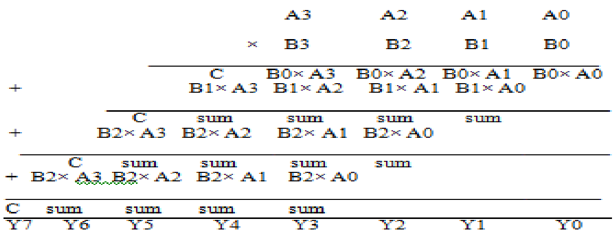
A3 A2 A1 A0
× B3 B2 B1 B0
C B0× A3 B0× A2 B0× A1 B0× A0
+ B1× A3 B1× A2 B1× A1 B1× A0
C sum sum sum sum
+ B2× A3 B2× A2 B2× A1 B2× A0
C sum sum sum sum
+ B2× A3 B2× A2 B2× A1 B2× A0
C sum sum sum sum
Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0

Fig 1: Array Multiplication structure

The block diagram of a 32-bit array multiplier is drawn below in fig 2

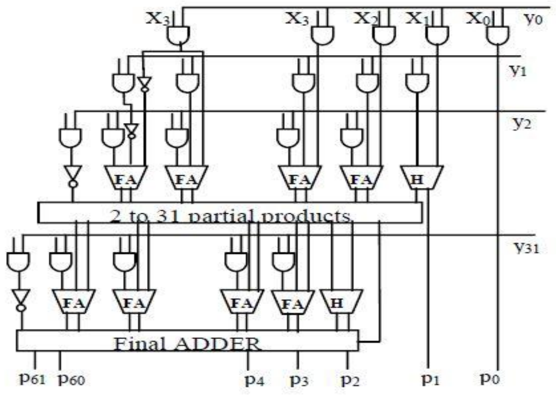

Fig 2: 32-bit Array Multiplier[14]

Array Multiplier gives more power consumption as well as optimum number of components required, but delay for this multiplier is larger. It also requires larger number of gates because of which area is also increased; due to this array multiplier is less economical [2] [10].Thus, it is a fast multiplier but hardware complexity is high [11].

3. MODIFIED BOOTH MULTIPLIER

Another improvement in the performance of a multiplier can be obtained by reducing the number of partial products that are generated .The Modified Booth recording multiplier is one such multiplier, it scans the three bits at a time to reduce the number of partial products [13].

The Modified Booth multiplier design uses a modified Booth encoder and selector technique to reduce and rearrange the partial products. This reduces the number of gate count and improves the performance of the multiplier. The Booth encoder encodes the multiplier bits using Radix-4 or Radix-8 algorithm. A k-bit binary number can be interpreted as a k/2-digit Radix-4 number, a k/3-digit Radix-8 number and so on, that deals with more than one multiplier bits in each cycle for using high radix multiplication [6].

The modified Radix-4 Booth algorithm is widely used when the operands are equal to or greater than 16 bits. In this algorithm, the two's complement of a multiplier is encoded in order to reduce the number of partial products that needs to be added. The multiplier, Y can be written in two's complement form as: [8]

$$Y= -Y_{n-1}2^{n-1}+ \sum_i Y_i 2^i \ ; 0 \le i \le n-2$$

It can be also written as [8]

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

The encoding of the signed multiplier Y, using the Radix-4 Booth algorithm is shown below in Table 1. The Radix-4 Booth recoding method encodes the bits of the multiplier into [-2, 2]. The multiplier bits are divided in blocks of three, such that each block overlaps the previous block by one bit. Examining the multiplier from the least significant bit makes it advantageous. The overlapping of the block is done so that we come to know what has happened in the previous block, as the most significant bit of the block acts like a sign bit.

Table 1: Radix-4 recoding [8]

Multiplier Bits			**Recorded Operation on Multiplicand, X**
Y_{2i+1}	**Y_{2i}**	**Y_{2i-1}**	
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 recoding applies the same algorithm technique as that of Radix-4, here instead of triplets, that is, blocks of three, multiplier bits are divided into blocks of four. Each quartet is encoded as a signed digit using Table 2.

Table 2: Radix-8 recoding [8]

Multiplier Bits				**Recorded Operation on Multiplicand, X**
Y_{i+2}	**Y_{i+1}**	**Y_i**	**Y_{i-1}**	
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	-X
1	1	1	0	-X
1	1	1	1	0X

The partial product generator (PPG) generates the partial product from the multiplicand and the encoded multiplier. The partial products obtained at the output of PPG are then added by partial product reduction tree (PPRT) without carry propagation. Finally, the summed results obtained from PPRT stage are added using Carry Propagate Adder (CPA).The block diagram of Modified Booth multiplier is shown below in fig 3

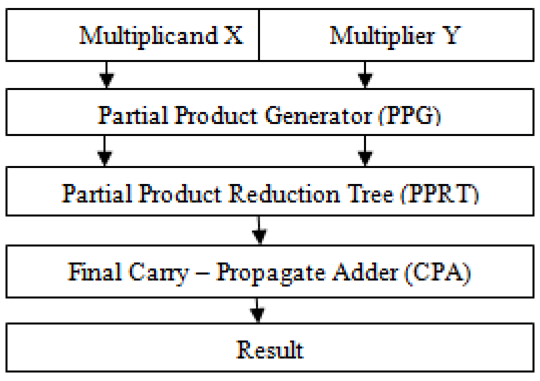
Multiplicand X
Multiplier Y
Partial Product Generator (PPG)
Partial Product Reduction Tree (PPRT)
Final Carry – Propagate Adder (CPA)
Result

Fig 3: Block Diagram of Modified Booth Multiplier

To speed up the multiplication Booth encoding performs various steps of multiplication at once. From the basics of Booth Multiplication it can be proved that the addition/subtraction operation can be skipped if the successive bits in the multiplicand and multiplier are same. Thus in most of the cases the delay of Booth Multiplication is smaller than that of Array Multiplier. However the performance of Booth Multiplier for delay is dependent on input data.

The method of Booth recording reduces the numbers of adders and hence the delay required to produce the partial sums by examining three bits at a time in Radix-4 or four bits at a time in Radix-8. The high performance of booth multiplier comes with the drawback of power consumption. The reason is large number of adder cells required that consumes large power [13].

4. WALLACE TREE MULTIPLIER

A fast process for multiplication of two numbers was developed by Wallace [12]. Here the Wallace tree has taken the role of accelerating the accumulation of the partial products. Using this method, a three step process is used to multiply two numbers, the partial products are formed, the partial product matrix is reduced to a two row matrix where sum of the row equals the sum of partial products, and the two resulting rows are summed with a fast adder to produce a final product. 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 proportional to the efficiency of the compressors, that is, a 4:2 compressor will be fast than a 3: 2 compressor as it takes four partial products at a time. In a Wallace tree using 3:2 compressor, three partial products are passed to a one bit full adder which is called a three input Wallace tree circuit, and the output signal (sum signal) 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 bits located at a one bit higher position.

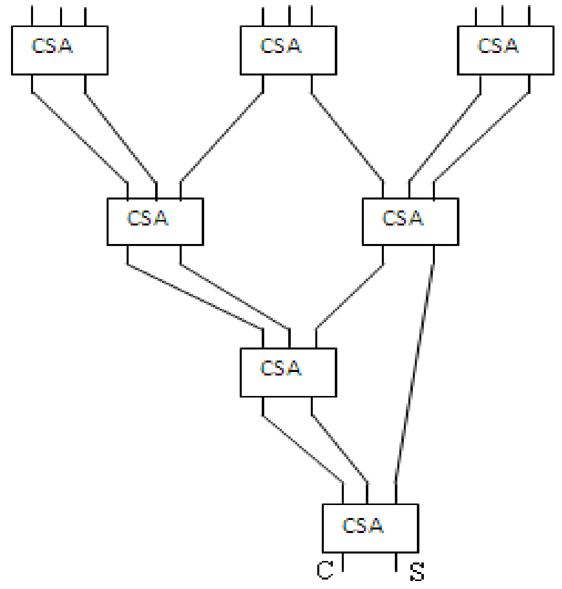

Fig 4: Wallace tree Multiplier with 3:2 Compressor [14]

Wallace tree is a tree having carry-save adders arranged as shown in figure 4. A carry save adder consists of full adders, but the carry output from each bit is brought out to form second result vector rather being than wired to the next most significant bit. The carry vector is 'saved' so that it can be combined with the sum later. In the Wallace tree method, the circuit layout is not easy although the speed of the operation is high since the circuit has an irregular structure [2].

5. MODIFIED BOOTH-WALLACE MULTIPLIER

The Modified Booth-Wallace tree multiplier consists of four major modules namely, Booth encoder, partial product generator, Wallace tree and carry look-ahead adder [7].The Booth encoder performs Radix-4 or Radix-8 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 to make it a fast multiplier [8]. The partial products are further supplied to Wallace Tree and added appropriately. The results are finally added using a fast adder, that is, Carry Look-ahead Adder (CLA) to get the final product as shown in fig 5:

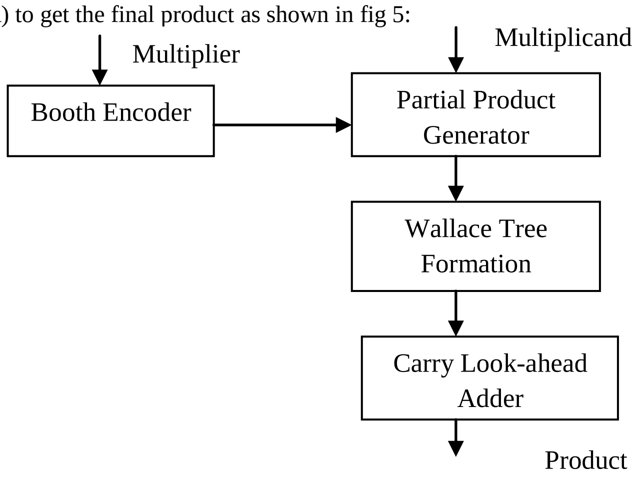

Fig 5.Block Diagram of Modified Booth-Wallace Tree Multiplier [8]

The Modified Booth algorithm reduces the number of partial products to half of the number of multiplier bits if Radix-4 booth algorithm is used and one-third of the number of multiplier bits if Radix-8 booth algorithm is used.Modified Booth algorithm is used in order to save chip area rather than reducing delay time and by using Wallace tree operation time is reduced because it makes use of Carry Save Adders (CSA) for fast accumulation of the partial products. By using Wallace tree multiplier,power consumtion too is reduced when compared with other multipliers. To further increase the speed, carry-look-ahead (CLA) adder is used as the final adder.

6. CONCLUSION

In this section, four multipliers discussed above are compared based on speed, area, circuit complexity and power consumed. The basic Array multiplier is the simplest of all multipliers in its circuit complexity and therefore occupies small area but these multipliers multiplies at a low speed using the basic add and shift algorithm and consumes maximum power. Based on the above study, it can be concluded that of all the multipliers, Modified Booth Wallace tree multiplier is the fastest as it takes the advantage of both multipliers, that is, Modified Booth multiplier wherein the number of partial products are reduced by half or one third of the multiplier bits based on the algorithm it uses ,that is, Radix-4 or Radix-8 and Wallace tree multiplier increases speed of accumulation because of the use of carry-save adders but it has the drawback of complex circuitry thus occupying the largest area. Thus a multiplier should be selected depending of the performance measures and the nature of the applications. Based on the discussion in the above sections, comparison of multipliers based on various parameters is shown in table 3:

Table 3: Comparison of multipliers

Multiplier Types	**Speed**	**Power Consumption**	**Circuit Complexity**	**Area**
Array	Low	Most	Simple	Small
Modified Booth	High	Less	Complex	Medium
Wallace tree	Higher	More	Medium	Large
Modified Booth-Wallace tree	Highest	More	More Complex	Largest

The speed of multiplier could be increased even more by booth encoding of large blocks of multiplier bits which would still reduce the number of partial products and by using compressors larger than 4:3 which could consider more number of partial products at a time and further accelerate the speed of multiplication. If floating point numbers are used in designing multiplier, the system could be made more accurate.

7. ACKNOWLEDGMENTS

I am thankful to my guide Assistant Professor Ajit Saraf (Pillai's Institute of Information Technology, India) for the helpful discussions and valuable guidance and Saraswati College of Engineering for giving the opportunity for giving a chance to participate in ICAST 2014.

8. REFERENCES

[1] 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

[2] "A Novel Parallel Multiply and Accumulate (V-MAC) Architecture Based On Ancient Indian Vedic Mathematics" Himanshu Thapliyal and Hamid Rarbania.

[3] "Low power and high speed 8x8 bit Multiplier Using Non- clocked Pass Transistor Logic" C.Senthilpari, Ajay Kumar Singh and K. Diwadkar, 1-4244-1355-9/07, 2007, IEEE.

[4] Kiat-seng Yeo and Kaushik Roy "Low-voltage, low power VLSI sub system" Mc Graw-Hill Publication.

[5] Jong Duk Lee, Yong Jin Yoony, Kyong Hwa Leez and Byung-Gook Park "Application of Dynamic Pass Transistor Logic to 8-Bit Multiplier" Journal of the Korean Physical Society, Vol.38, No. 3, pp.220-223, March 2001

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

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

[8] Aparna P R, Nisha Thomas, "Design and Implementation of High Performance Multiplier using HDL", 2011.

[9] "ASIC Implementation of 4 Bit Multipliers" Pravinkumar Parate, IEEE Computer society. ICETET, 2008.25.

[10] Morris Mano, "Computer System Architecture", PP. 346-347, 3rd edition, PHI. 1993.

[11] Jorn Stohmann Erich Barke, "A Universal Pezaris Array Multiplier Generator for SRAM-Based FPGAs" IMS-Institute of Microelectronics System, University of Hanover Callinstr, 34, D- 30167 Hanover, Germany.

[12] Moises E. Robinson and Ear Swartzlander, Jr. "A Reduction Scheme to Optimize the Wallace Multiplier" Department of Electrical and Computer Engineering, University of Texas at Austin, USA.

[13] Tam Anh Chu, "Booth Multiplier with Low Power High Performance Input Circuitary", US Patent, 6.393.454 B1, May 21, 2002.

[14] Sumit R. Vaidya and D. R. Dandekar, "Performance Comparison of Multipliers for Power-Speed Trade-off in VLSI Design", Recent advances in networking, VLSI and Signal Processing, ISSN: 1790-5117.