Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 11, November-2021

Review on Recent Advances in VLSI Multiplier

Sulbha Bhongale*[1], Dr. Yogesh Kandhare[2], Santosh Bobade[3]
Department of electronics and communication
[1],[2]Infinity Management & Engineering College, Sagar
[3]Department of Physics, Jaypee University of Engineering and Technology, Raghogarh

***Abstract*:- Low power very large-scale integration (VLSI) circuit is vital criteria for designing an energy efficient design for prime performance and the compact device design. Multiplier plays an important role for planning the energy economical processor that determine the potency of the processor. Multiplier has important role in DSP, DIP, microprocessor, and microcomputer application. Among all the arithmetic operation that exist, a processor consumes most of the time and hardware resources for carrying out multiplication when compared to other operations like addition and subtraction. In this review, Array multiplier, Booth multiplier, Modified booth multiplier, Wallace tree multiplier, and Modified booth allace tree multiplier have been reviewed considering several performance parameters, such as speed, area, power consume, circuit complexity. The fast multiplier is desired in any application. While comparing various multipliers, it has been observed that the selecting an appropriate multiplier for any application is always a tradeoff between speed, power and the area. Array multiplier has the largest delay and low power consumption and have large area, while booth encoded allace tree multiplier has least delay and it also have large area. In above multipliers, one of the limitations is no of partial products. The booth algorithm for multiplication overcomes the limitation and hence, the performance is improved. In this review, the booth algorithm and its variant which outperform the several multipliers will be discussed. The detail comparison of multipliers and their performance matrix in term of delay, area and power consumption will be provided, so that the most appropriate multiplier can be chosen for application.**

Keywords: Array multiplier, Wallace tree multiplier, Booth multiplier, Modified booth multiplier, low power VLSI

INTRODUCTION:

The multiplication is an important central function in arithmetic logic operation in several application such as digital filtering, digital communication. The faster device with low power consumption is the demand of every consumer. High speed components of the device and reduce the power consumption leads to enhanced performance. These days, the demand of portable electronics modules is on rise in which various digital signal processing (DSP) are used, Therefore, the compact and the faster multiplier plays a vital role in designing such modules.[1] The computational performance of a DSP system is limited by its multiplier performance and multiplicand dominates the execution time of most of the DSP algorithms [], and therefore high-speed multiplier is desired with an ever -increasing quest for greater computing power on battery operated mobile devices [2]. The design emphasis has shifted from optimizing conventional delay time, area, size to minimize power dissipation. Therefore, for maintaining the high performance, the speed and area of multiplier are a major design issues and they need to be optimized for enhancing performance.[3]

The architecture of multiplier can be split into three stages namely, partial product generation stage, partial product reduction stage and the final addition of the reduced partial product stage. Traditionally, the shift and add algorithm have been used to perform the multiplication.[7] However, it is not suitable for faster multiplication in VLSI because it requires a greater number of adder units which leads to higher delay in performing multiplication operation. Some of traditional approaches for speeding up multiplication operation is reducing the number of partial products by using multibit compressor.[8]

Multiplier can be classified into two categories namely, serial and parallel multipliers. In a serial multiplier, each bit of multiplier is used for evaluating the partial product whereas in parallel multiplier, partial products from each bit of multiplier are computed in parallel. The main parameter that determines the performance of the parallel multiplier is the number of partial products, that is to be added. In a parallel multiplier, the speed is compromised to achieve better performance in terms of area and consumption. [8] Depending upon the type of application, the parallel or the serial multiplier can be used. Some of the known parallel multipliers are array multiplier, Wallace tree multiplier, Booth multiplier and modified Booth multiplier. These multipliers are discussed in following text keeping focus on Booth multipliers.

Booth multiplication algorithm treat both signed and unsigned numbers. The main use of booth algorithm is to reduced the number of partial products by reducing the number of multiplier bits. It is used in different recoding scheme as radix 2, radix 4, radix 8. In wallace tree algorithm the number of seqential adding stage is reduced, which is used for improving speed, modified booth algorithm also reduces the partial product.[7]

ARRAY MULTIPLIER:

Array multiplier is a regular structure multiplier; Array multiplier is digital combinational circuit used for multiplying two binary number by employing an array of full adder and half adder. This multiplier perform multiplication by the standard add and shift algorithm.[2] The partial product is generated by multiplying the multiplicand with each bit of the multiplier. The partial product obtained are then shifted depending on their bit order and are finally added at the last stage. The number of partial products that are generated is equal to the number of multiplier bit. Array multiplier has very regular and systematic structure, its delay becomes very large for a large word length Min C. Park et al (1993) [4] presented a new architecture using dual array tree structure. Which include dual partial product array divided from one partial product plane of conventional array multiplier. Array multiplier consumes more power and optimum number of components required. It requires larger number of gates, and therefore the area also increases. So, the array multiplier performs poor in terms of area [3]. It is a fast multiplier but hardware complexity increases due to the large number of gates. The schematic representation of conventional array multiplier with CSA is shown in fig.1.

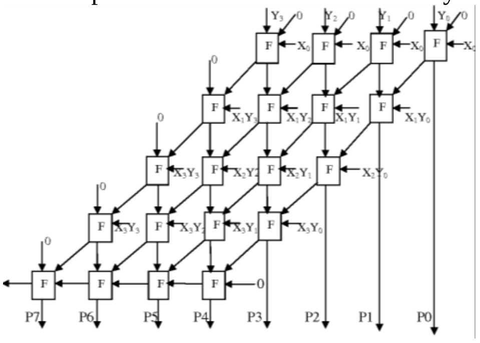

Fig:1 – Conventional array multiplier with CSA.

WALLACE TREE MULTIPLIER:

A Wallace tree multiplier is a hardware implementation of digital circuit that perform multiplication of two binary number. Wallace tree multiplier are devised by the Australian computer scientist Chris wallace in 1964[1]. The wallace tree multiplier lessens the number of partial products and use carry select adder for the addition of partial products. Wallace tree is an irregular in structure in which informal implementation does not supply the systematic method for the compressor interconnection, still it is an efficient method of adding partial product in parallel [4]. In this method, three step process is used to multiply two integers. There are two different designs of Wallace tree multiplier. The first one is considered using only half adder and full adder while the second one uses a more sophisticated carry skip adder (CSA). At the initial step, partial products are formed by multiplying each bit of multiplicand by each bit of multiplier. So, the number of partial products is determined the length of multiplicand [2]

				A_3	A_2	A_1	A_0
				B_3	B_2	B_1	B_0
				P_{03}	P_{02}	P_{01}	P_{00}
			P_{13}	P_{12}	P_{11}	P_{10}	
		P_{23}	P_{22}	P_{21}	P_{20}		
	P_{33}	P_{32}	P_{31}	P_{30}			
	P_{33}	P_{23}	S_{04}	S_{03}	S_{02}	S_{01}	P_{00}
		C_{04}	C_{03}	C_{02}	C_{01}		
	P_{33}	P_{32}	P_{31}	P_{30}			
	P_{33}	S_{14}	S_{13}	S_{12}	S_{11}	S_{01}	P_{00}
	C_{14}	C_{13}	C_{12}	C_{11}			
C_{out}	S_{24}	S_{23}	S_{22}	S_{21}	S_{11}	S_{01}	P_{00}

Fig. 2 4x4 Schematic design of Wallace Tree Multiplier

Half adder is used in each column where there are two bits and full adder is used in each column where there are three bits in column, any single bits in column is passed to next stage in the same column without any operation. This reduction procedure is repeated in each successive stage until only two remain.[3] In the final stage, the remaining two rows added. After completing all the three stages we get 8 bits output. The schematic example is shown in Fig. 2.

Speed, Area, and the power consumption of the multiplier will directly be proportional to the efficiency of compressor. The compressor (3:2) is used to make overall multiplier faster. A wallace tree using 3:2 compressor, three partial products are passed to one-bit full adder which is called three input wallace tree circuit and output of signal is applied to next stage full adder of the same number of bit located is one bit higher position [1]. Wallace tree is tree having carry save adder are arranged as shown in Fig. 3. A carry save adder (CSA) consists of full adder, but 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 combine with sum later. The schematic representation of Wallace tree multiplier is shown in Fig. 3.

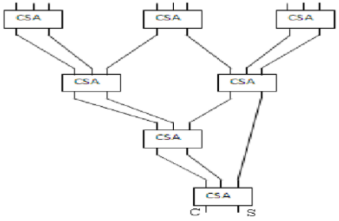

Figure 3. Wallace Tree Multiplier using 3:2 Compressor

The purpose of CSA is to improve worst case path delay [5]. A 4 bit is used for implementing CSA. The carry output from first addition is the carry input in the second addition. The advantage of CSA is to increase the maximum frequency [9]. The wallace tree multiplier using CSA occupies smallest area.

In the wallace tree method, due to irregular structure, lay outing the circuit becomes more complex although the speed of the operation is high, since the circuit has an irregular structure.

BOOTH MULTIPLIER:

This algorithm was devised by Andrew Donald Booth in 1950. While doing study on crystallography Booth used reception desk calculators that shift faster than adding and formed the algorithm to increase the speed. It treats both signed and unsigned numbers. The flowchart for Booth algorithm is provided in Fig. 4. It is extremely prevailing algorithm for signed number multiplication, which considers both positive and negative number uniformly [10]. An algorithm uses 2's complement presentation of signed binary number for multiplication [8]. Each multiplier bit generates one multiple of the multiplicand that should be added to the partial product. In this case, the delay of the multiplier is mainly governed by number of additions to be performed. If there is way to reduce the number of additions, the performance will be better. It is the standard technique used in chip design and provides significant improvement over the long multiplication [7]. The schematic flow chart of Booth algorithm is shown in Fig. 4 and application of Booth algorithm shown in table 1,2 step by step.

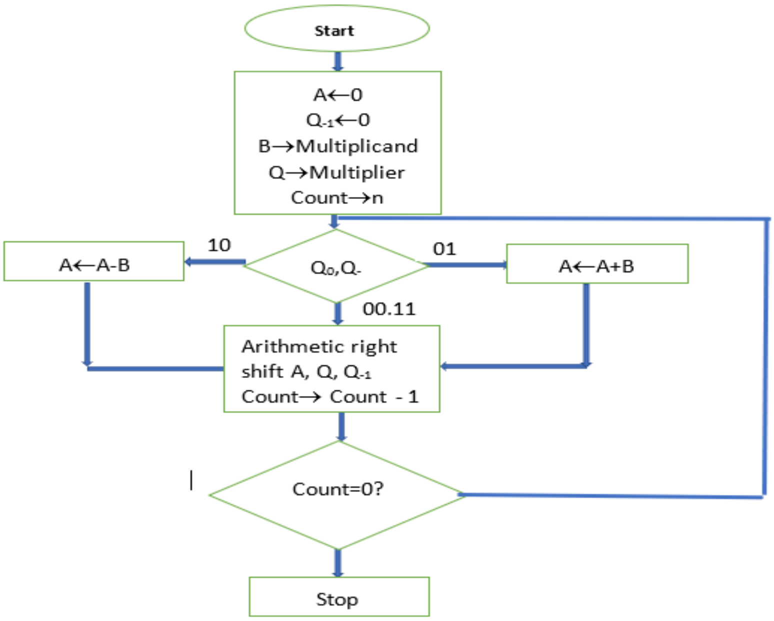

Figure 4. Flow Chart of Booth Multiplier

Procedure;
M is the multiplicand
Q is the multiplier
Consider 1 bit register Q-1 and initialize to 0
Consider register A and initialize to 0
Condition;
1] If Q0, and Q-1 are same 11 or 00 then perform arithmetic right shift by 1 bit
2] If Q_0 and Q_{-1} = 10 then perform A=A+M and then perform arithmetic right shift
3] IF Q_0 and Q_{-1} = 10 then perform A = A-M then perform arithmetic right shift
Example 5 x -4 = -20
Multiplicand = 5
Multiplier = -4

Table 1 Multiplication of two number (5 x -4) in step by step using Booth Algorithm

Count	A	Q	Q_{n-1}	Operations
	0000	1100	0	Initialization
1	0000	0110	0	Right shift
2	0000	0011	0	Right shift
3	1011	0011	0	A = A-M = A + (M` +1)
	1101	1001	1	Right shift
4	1110	1100	1	Right shift
	00010011			
	Ans = 00010100			2`s complement of 20

Table 2 Truth table 1 for booth multiplier

Q	Q_{n+1}	Recorded bits	Operation
0	0	0	Shift
0	1	+1	Add
1	0	-1	Subtract
1	1	0	Shift

MODIFIED BOOTH MULTIPLIER:

Another improvement in the multiplier is by decreasing the number of individual partial products. To achieve high speed multiplication, algorithm using parallel counter like modified booth algorithm has been demonstrated and practiced. It reduces number of multiplicand and number of partial products.[8] It groups three consecutive bits at a time, in that three-bit 2 bits on left hand side are present bit and third bit is the carry bit from previous pair of bits. The multiplier operates much faster than array multiplier for longer operands because it's time to complete the operation is proportional to the log of the word length of operand. The main benefit of this multiplier is that if the successive bits in multiplicand are same, then addition can be skipped [6]. The number of partial products is downed by a factor of half by using the technique of booth recoding [7]. The grouping of bits and booth recording determine the number of partial products.

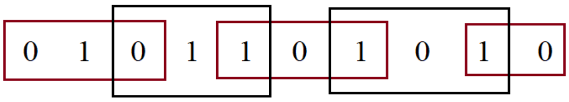

Figure 4: Grouping of bits from the multiplier term

In a given multiplier, the group of three bits are considered, it initiates from the LSB bit and first group is formed by two bits, onward consider a group of three bits in which one bit will be overlapped on the previous group as shown in figure 4. Thus, groups multiplier will result in the production of bits between these five bits are as follows -2, -1, 0, +1, and +2 which are tabulated in table 3.

Table 3 Recoding table of radix-4 Booth multiplier

Block	Recoded digits	Operation
000	0	0
001	+1	+1
010	+1	+1
011	+2	+2
100	-2	-2
101	-1	-1
110	-1	-1
111	0	0

Consider the scheme of radix-4 Booth multiplier, a demonstrative example of multiplication of two number is described below. Assuming the integers 10 and 5, the multiplication results into 50. The multiplicand along with its binary equivalent and multiplier along its binary number are,

Multiplicand 10 = 001010
Multiplier 5 = 000101
0 0 0 1 0 1 0 = 0 1 1 (grouping of bits)

```
        0 0 1 0 1 0
              0 1 1
---------------------
        0 0 1 0 1 0
    0 0 1 0 1 0
0 0 0 0 0 0
---------------------
0 0 0 0 1 1 0 0 1 0
```

Algorithm:
1] Check whether the multiplier and multiplicand are having negative sign if so convert 2's complement
2] Add implied zero to the multiplier and then convert the multiplier into other form using Booth recoding table shown in above table
3] Now multiply the multiplier with multiplicand
a] If the multiplier was zero then the multiplicand multiplied with zero will result in the output is zero.

b] If the multiplier was +1 then, the result of the multiplicand with the multiplier was same as the multiplicand
c] If the multiplier was -1 then, the result was calculated by 2's complement of the multiplicand
4] Finally add all generated partial products adders

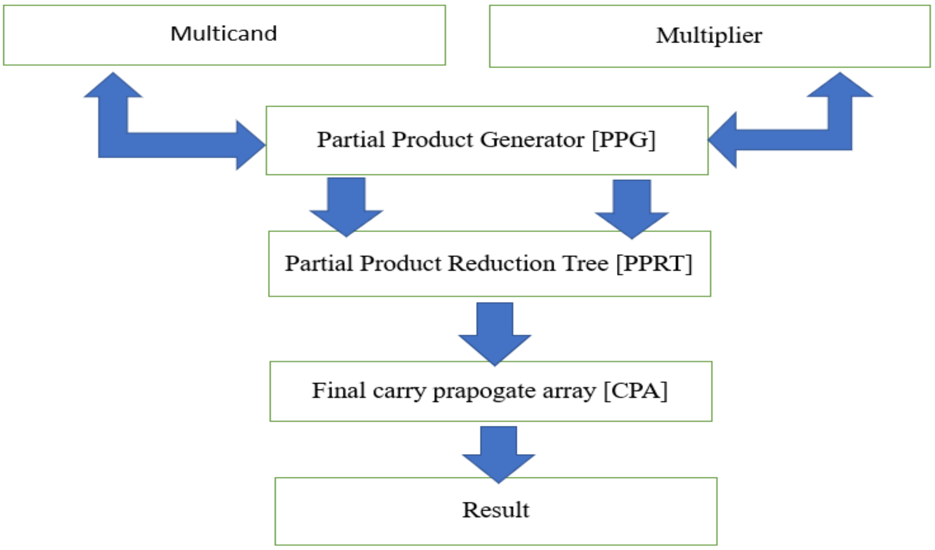
Multicand
Multiplier
Partial Product Generator [PPG]
Partial Product Reduction Tree [PPRT]
Final carry prapogate array [CPA]
Result

Figure 6 Block diagram of Modified Booth Algorithm

The figure 6 shows the block diagram of Modified Booth algorithm. The partial product generator [PPG] generates the partial product from the multiplicand and encoded multiplier.[6] The partial product obtained at the output of PPG are then added by PPRT without carry propagation. Finally, the summed result obtained from PPRT stage are added using carry propagation array [CPA]. [7]

Table 3 Comparison of Multiplier

Multiplier types	Speed (NS)	Power consumption (W)	Circuit complexity	Area	Delay (NS)
Array multiplier	Low	Most	Simple	Small	72.986
Wallace tree multiplier	Higher	More	Medium	Large	53.198
Booth multiplier	High	Less	Complex	Medium	55.70
Modified Booth multiplier	Highest	Less	More complex	Largest	54.50

CONCLUSION:

In this article, various types of multipliers have been reviewed considering the performance parameters and complexity of design. The results from different sources are tabulated in Table 3. It can be observed that the trade of between the various performance parameter should be made while selecting the multiplier for some application. In term of complexity, array multiplier out performs all the multipliers while the speed is relatively the lowest among all. If we consider the power consumption of multiplier, which is the most important parameter, in the case of a portable devices, modified Booth algorithm excels among all the discussed multipliers.

REFERENCES:

[1] K. N. Singh and H. Tarunkumar, "A review on various multipliers designs in VLSI," Annual IEEE India Conference, pp 1-4, 2015
[2] B. Lamba and A. Sharma, "A review paper on different multipliers based on their different performance parameter", 2nd International Conference on Inventive Systems and Control, pp 324-327, 2018
[3] Bhawna Singroul, Pallavee Jaiswal, "A review on performance Evaluation different digital multiplier in VLSI using VHDL", International journal of Engineering research & technology, vol-7, issue-5 ,2018
[4] Savita Nair, "A review paper on comparison of multiplier based on performance parameter", International journal of computer application, vol-2, pp 6-9, 2014
[5] Sumit Vaidya, Deepak Dandekar, "A review on delay performance comparison of multiplier in VLSI circuit design", International journal of computer network & communication, Vol 2, issue 4, pp 47-55, 2010
[6] Shweta S. Khobragade, Swapnil P. Kormore, "A review on low power VLSI design of Modified Booth multiplier", International journal of Engineering and Advanced Technology, vol-2, issue-5, pp 463-466, 2015
[7] Bhavya Lahari Gundapaneni, JRK kumar Dabbakutti, " A review on Booth Algorithm for the design of multiplier", International journal of innovative technology and Exploring Engineering , vol-8, issue-7, pp 1506-1509 , 2019
[8] Kiran Kumar, S. Anusha, G. Y. Rekha, "A design of low power modified Booth multiplier", International journal of current engineering and scientific research, vol-5, issue-4, pp 287-292, 2018
[9] Soniya, Suresh kumar, "A review of different types multiplier and multiplier accumulator unit", International journal Emerging Trends and technology in computer science, vol-2, issue-4, pp 364-368, 2013
[10] A. D. BOOTH, "A signed Binary multiplication technique", in journal of Mech. APPL. Math, Oxford University press, vol-4, pp 236-240,1951