

COMPARISON OF DIFFERENT TYPES OF MULTIPLIERS WITH RESPECT TO SPEED, AREA AND POWER

¹MADHU GUDHIMETLA, ²CM ANANDA

¹Ph.D Scholar, AcSIR, ²Senior Principal Scientist, CSIR-NAL.

Abstract- Nowadays, digital circuits play a vital role while designing of DS (Digital Signal) Processor, graphical systems, FPGA/ASIC, multimedia systems. An increasing number of high speed DSP applications, need of high precision (16-bit or more) fixed or floating point multiplier suitable for VLSI implementation. In this paper I concentrated mainly on design of digital circuits which occupies less area, high speed performance which consumes less power. For example, one of the versatile digital circuit is "Multiplier". I emphasizing the multiplier circuit because of system performance, generally determined by the performance of the multiplier. Multiplier is a fundamental unit of digital system and signal processing. Multiplier and adder are basic arithmetic unit of digital systems and signal processing. Major problem in multiplier shows in terms of speed, power, area and complexity. High speed DSP computation applications such as Fast Fourier Transform (FFT) requires addition, multiplication because as large number of filtration, convolutions and related operations are needed during the Digital Signal Processing. Nowadays. Modern computer system requires the high speed performance multiplication for signed, unsigned numbers. Multiplier is also essential for all types of wireless communications applications. In this novel I compared all different types of multiplier their performance in terms of speed, area and power. For this I used Xilinx ISE 14.5 version.

Index Terms- Carry Save Adder.

I. INTRODUCTION

The performance of multiplication is crucial for multi-media applications such as 3D-graphics and signal processing systems, which depends on the execution of large number of multiplications. High speed multipliers are the core components of Digital Signal Processors (DSP). As large number of filtration, convolutions and related operations are needed during the digital signal processing. As the size, power consumption and silicon area greatly dependent upon the multiplication's word length. It is very important to have an efficient design in terms of performance, area, speed of multiplier.

The maximum operating performance of system depends upon that of the multiplier which contains the three major steps for calculating the product generation

- ❖ Recoding and generating partial product
- ❖ Reducing the partial products by partial product reduction scheme (Wallace Tree)
- ❖ Adding the remaining two rows of partial products by using carry-propagate adder (Ex. Carry Look ahead adder) to obtain, final product.

Multiplier architecture is reconfigured so as to enhance their performance and thereby improving the efficiency of applications. We compare general multipliers from an architecture point of view, maximum clock frequency, latency, throughput, resource usage, as well as dynamic power consumption.

II. DIFERENT TYPES OF MULTIPLIERS

- ❖ Array Multipliers
- ❖ Braun Multipliers

- ❖ Wallace Tree Multipliers
- ❖ Dadda Multipliers
- ❖ Baugh-Wooley Multiplier
- ❖ Booth Multiplier(Radix-2)
- ❖ Modified Booth Multiplier(Radix-4,8,32,64)

A. Array Multiplier

In digital multiplication is most extensively used operation (especially signal processing), people who design digital signal processing sacrifice a lot of chip area, in order to multiply as fast as possible.

AND gates	HA	FA	Total Adders
$M*N$	N	$(M-2)*N$	$(M-1)*N$

TABLE I: MxN array multiplier

Hence it requires more area, which consumes the more power. It is one of the major drawback of array multiplier. It is suitable for only un-signed numbers. Every multiplier has own disadvantages and also advantages are there. It is suitable for only signed numbers. Array multiplier has minimum complexity for less number of bits, because number of bits are increases partial product increases, it implies that complexity is more, delay increases, power dissipation also increases. This is main drawback of array multiplier.

B. Braun Array Multiplier

It is simple parallel multiplier that is commonly known as carry array multiplier. This also suitable for only un signed numbers. It consists of array of and gates and adders in iterative structure and it does not require a logic registers. This also known as non-additive multiplier, since it does not add an additional operand to the result of multiplication

AND GATES	ADDERS
$M \cdot N$	$M \cdot (N-1)$

TABLE II: $M \times N$ Braun multiplier

If the number of bits increases then number of adders and AND gates components also increases, which may result increases the power consumption, occupies more area, take more time. Hence suitable for only less than 16-bit. One of the major drawback is delay of Braun multiplier 'I' is dependent on the delay of the full adder cell and also a final adder in the last row. (the speed and power of the full adder is very important for large arrays).

C. Wallace Tree Multiplier

Wallace tree multiplier designs capitalize on the concept of carry save adder. It is like multiplier designs in the sense that they generate and sum partial products in parallel, but improve critical path

through the multiplier. 4x4 bit Wallace tree multiplier figure shown in below. In first stage, multiply together (AND gates), each X_i with each Y_i to produce of total of $n \cdot n$ intermediate wires. Each wire has said to be weight, for example $X_0 \cdot Y_0$ has weight $:1(2^0 \cdot 2^0 = 1)$. Similarly, $X_3 \cdot Y_3$ has weight: 64. Reduce the number of intermediate wires using additional stages composed of full adders and half adders. Combine any three wires with same weight using a full adder, result in next stage is one wire of the same weight (i.e., sum) and one wire a higher weight (i.e., carry). Combine any two wires with same weight using a half adder, result in next stage is one wire of the same weight (i.e., sum) and one wire with given weight, just it passes through the next stage. In final stage, all weights have just one or two wires in them. Combine the wires into two 2n-bit values and them with standard values.

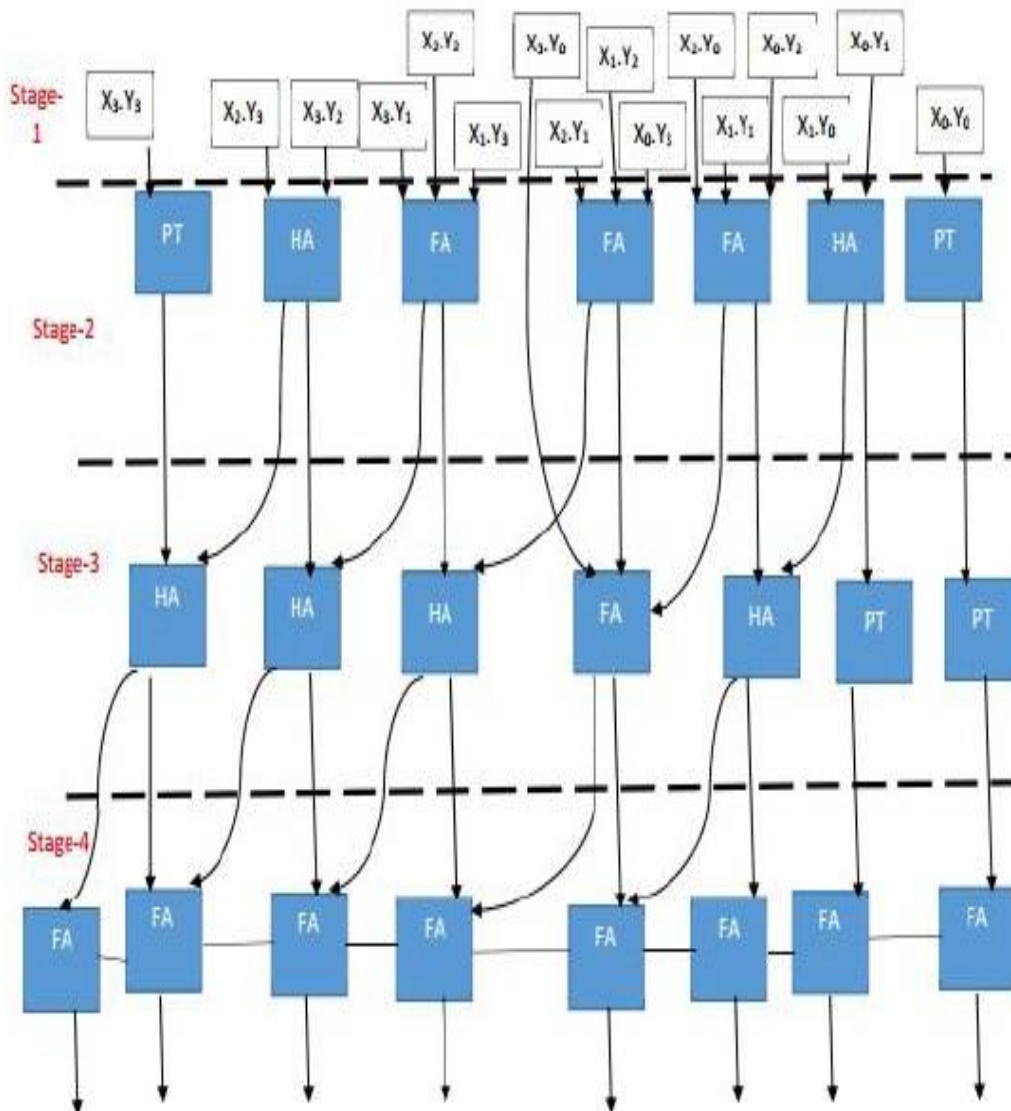


Fig.1. 4x4-bit Wallace Tree Structure^[1]

	STAGE-1 OUTPUT	STAGE-2 ACTION	OUTPUT	STAGE-3 ACTION	OUTPUT
Weight-1	1	PT	1	PT	1
Weight-2	2	HA	1	PT	1
Weight-4	3	FA	2	HA	1
Weight-8	4	FA	3	FA	2
Weight-16	3	FA	2	HA	2
Weight-32	2	HA	2	HA	2
Weight-64	1	PT	2	HA	2
Weight-128	0				1

TABLE III: Actions in constructing Wallace multiplier for 4-bit inputs

D. Dadda Multiplier

Dadda multiplier are a refinement of the parallel multipliers first presented by Wallace. Both Wallace and Dadda multiplier consists of 3 stages. The partial product matrix is formed in first stage by N*N AND gates(Where N is number of bits). In the dot diagram notation developed by Dadda, each partial product is represented by dot. Full adder is an implementation of a (3,2) counter, which takes 3 inputs and produces 2 outputs. Similarly, half adder is implementation of (2,2) counter which takes 2 inputs and 2 outputs^[4].

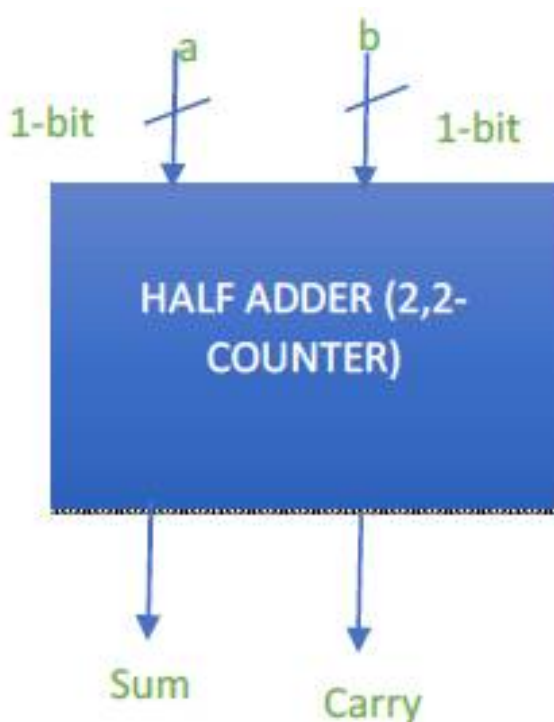


Fig.2: 2,2-Counter

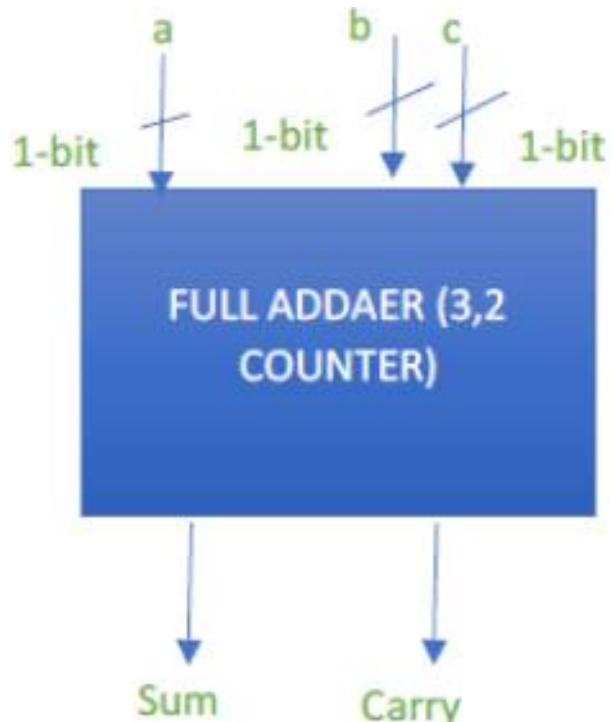


Fig.3: 3,2-Counter

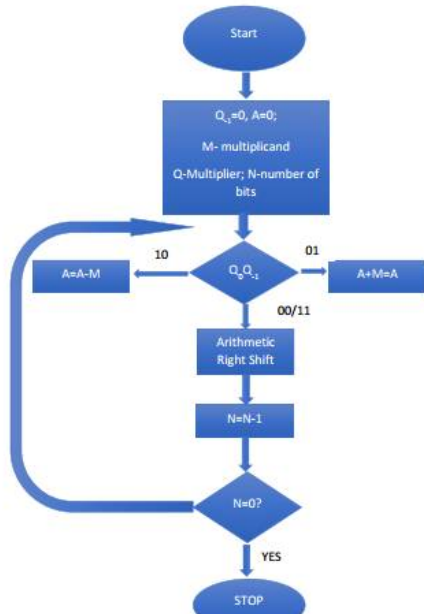


Fig. 4. Booth Multiplication Algorithm [2]

F. Modified Booth Encoding Algorithm

An increasing number of high speed DSP applications have need of high precision (16-bit or more) fixed or floating point multiplier suitable for VLSI implementation [8]. Multipliers are a critical aspect of any DSP Chip development, there are several references in the literature to their VLSI implementation. Modified booth partial products are conventionally added one at a time in an adder array whose results is formed in a final carry propagate added stage [7]. Modified booth algorithm reduces the number of partial products to be generated and is known as fastest multiplication algorithm. Modified booth multiplier that uses increases parallels to reduce the number of gate delays through it.

Our goal is to reduce computation time by using booth’s algorithm for multiplication and to reduce chip area by using carry save adders arranged in Wallace tree structure. Modified booth multiplier is used to reduce the number of partial products compare to basic booth multiplier. Modified booth multiplier performs both signed and unsigned multiplications

i+1	i	i-1	Operation
0	0	0	0*M
0	0	1	1*M
0	1	0	1*M
0	1	1	2*M
1	0	0	-2*M
1	0	1	-1*M
1	1	0	-1*M
1	1	1	0*M

Table VI: Modified Booth Algorithm [8]

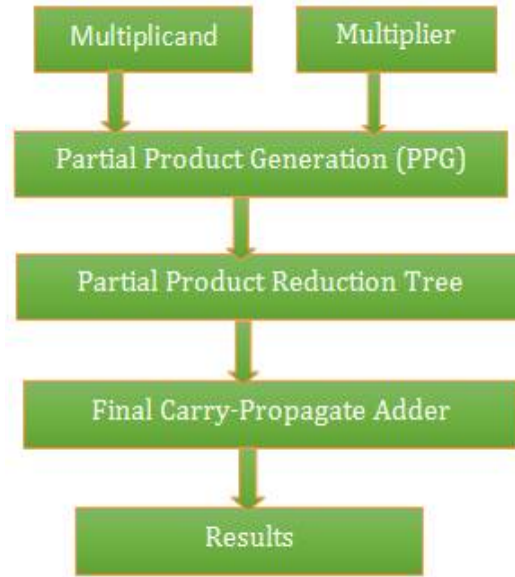


Fig.5: Modified Booth Encoding Scheme

III. RESULTS

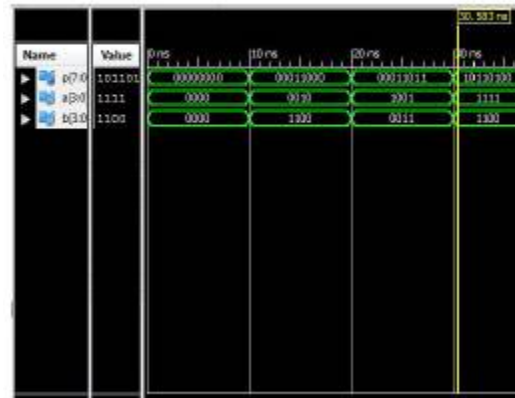


Fig.6 output results of 4x4-multiplier

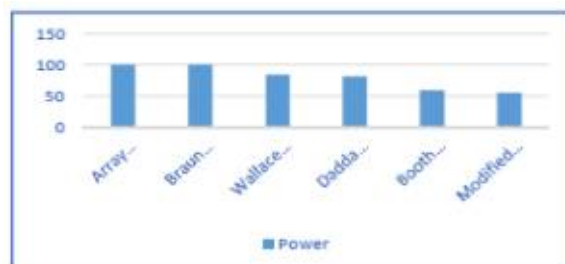


Fig.7 Comparison of different types of Multipliers with respect to power

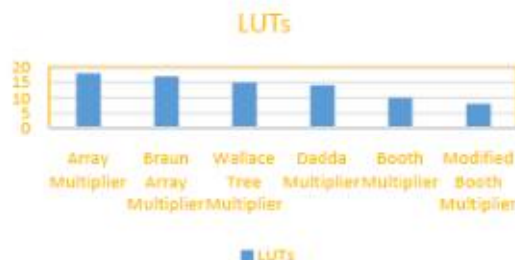


Fig.8 Comparison of different types of Multipliers with respect to area

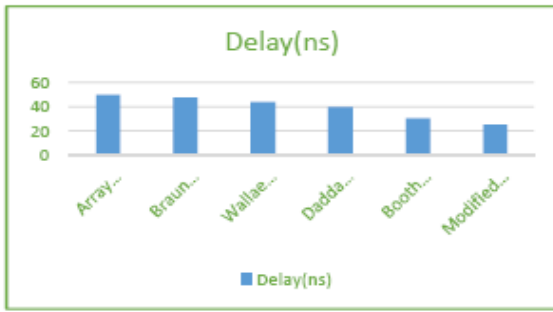


Fig.9 Comparison of different types of Multipliers with respect to power

REFERENCES

[1] Texts in computer science: A Practical Introduction to Computer Architecture.
 [2] W.C. Yeh and C.W. Jen, "High-Speed Booth Encoded Parallel Multiplier Design", IEEE Trans. Computers, vol.49, p.p. 692-701,2000.

[3] Vinoth, C.; Bhaaskaran, V.S.K.; Brindha, B.; Sakthikumar, S.; Kavnilavu, V.; Bhaskar, B.; Kanagasabapathy, M.; and sharath, B.; "A novel low power and high speed wallace tree multiplier for RISC processor," 3rd international conference on electronics computer technology(ICECT),2011, vol.1, pp.330-334,8-10 April 2011.
 [4] Srihari veeramachaneni; kirthi M Krishna; Lingamneni Avinash; Sreekanth reedy puppala and M.B Srinivas; "Novel Architectures for High-speed and low-power 3-2,4-2 and 5-2 compressors," 20th International conference on VLSI Design, 2007, pp.324-329, Jan.2007.
 [5] E.E. Swartz lander Jr., "Merged Arithmetic," vol.29, pp.946-950,1980.
 [6] Dadda, "On Parallel digital multipliers," Alta Frequenza, vol.45, pp. 547-580, 1976.
 [7] K.A.C. Bickerstaff, E.E. Swartz lander Jr., and M.J. Schulte, "analysis of column compression multipliers," 15th IEEE Symp. on computer arithmetic, pp. 33-39, 2001.
 [8] Ravindra P Rajput, M. N Shanmukha Swamy, "High Speed modified booth encoder multiplier for signed and unsigned numbers", IEEE International Conference on Modelling and Simulation, pp. 649-654, 2012.

★ ★ ★