International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 9, September - 2013

Comparative Analysis Of Multipliers

(serial and parallel with radix based on booth algoritham)

Deepak Bordiya
Department of Electronics & Communication
Acropolis Institute of Technology
Indore, India

Lalit Bandil
Department of Electronics & Communication
Acropolis Institute of Technology
Indore, India

Abstract

These papers present an efficient implementation of high speed multiplier using the shift and add method, Radix_2, Radix_4 Booth multiplier algorithm. In this paper we compare the working of the multiplier by implementing each of them separately in FIR filter. For this purpose we will first design three different type of multipliers using shift method, radix two and radix four booth multiplier with booth algorithm. Then we will compare the working of different multipliers by comparing the some parameters which are responsible for the multiplier working like (speed, area, power and number of PPs) etc. The multiplier circuit is designed using VHDL and simulated using Xilinx ISE Simulator or Modelsim design tool.

Keywords— *(multipliers , shift and add , radix , booth, and Xilinx.)*

I. INTRODUCTION

The multipliers are the better option for high-speed data processing. Various algorithms proposed for multiplication are, Booth Algorithm, Braun and Baugh - Wooley. Multipliers have large area, long latency and consume considerable power. Therefore, low-power multiplier design has been an important part in low-power VLSI system design. A system's performance is generally determined by the performance of the multiplier because the multiplier is the slowest element in the system. area and speed are usually conflicting constraints so that for improving the speed of the system results in larger areas. The need for low-power VLSI system arises from two main forces. First, with the steady growth of operating frequency and processing capacity per chip, large amount of current has been delivered and the heat generate due to large power consumption must be removed by proper cooling techniques. Second, battery life in portable or mobile electronic devices is limited Continuous advances of microelectronic technologies make better use of energy, encode data more effectively, transmit information more reliable, etc. Particularly, many of these technologies address low-power consumption to meet the requirements of various portable applications. In these application systems, a multiplier is a fundamental arithmetic unit and widely used in circuits.

A) Add and Shift Multiplier

The requirement is to design an multiplier based on the shift and add method. The overall architecture is shown in Figure. The multiplier shall accept as inputs an multiplier and multiplicand as well as a Start signal. The multiplier shall then calculate the result using the shift and add method and provide the result along with a Stop signal. The design shall be coded in VHDL and simulated for proper functionality and timing.

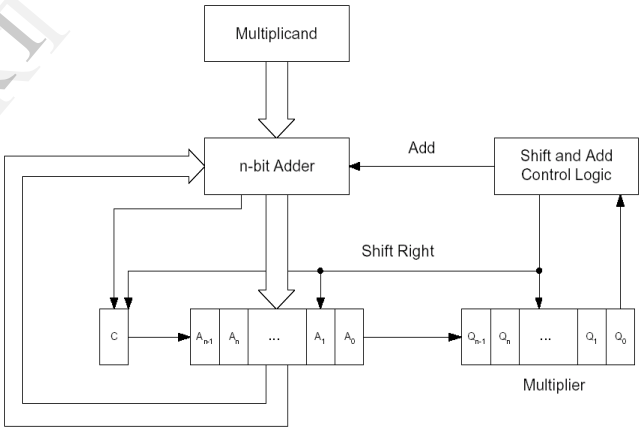

FIG : - ADD AND SHIFT MULTIPLIER

Among other multipliers, shift-and-add multipliers have been used in many other applications for their simplicity and relatively small area requirement. Higher radix multipliers are faster but consume more power since they employ wider registers, and require more silicon area due to their more complex logic.

The design was implemented as a finite state machine with states and transition logic as shown in Figure. The Start signal transitions the state machine out of the idle state and into the initialize state whereby it commands the multiplicand and multiplier to be loaded into registers. Once loaded, the state machine goes through a series of test

and shift, or test, add and shift operations depending on the status of the LSB bit.

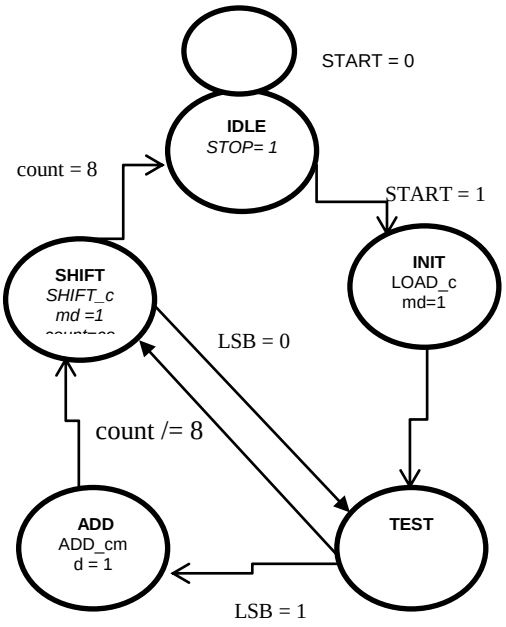

FIG: - CONTROLLER FSM

Upon reaching the maximum count for the multiplication cycle, the state machine goes back to the idle state and outputs a Stop signal.

B). Booth algorithm & multiplier

In 1951, based on the idea that computers are faster at shifting bits than adding them, Andrew Donald Booth developed an algorithm known as Booth's algorithm. There were many such discoveries through the years to improve the efficiency and performance of the multiplication algorithms.

An ALU addition operation can be very time consuming when done repeatedly. Recognizing the fact that computers can shift bits faster than adding bits, Booth developed an algorithm, which reduces the number of additions that take place when multiplying two numbers. Booth's algorithm multiples two numbers in 2's complement form. It creates an initial guess for the product, which is zeros followed by the multiplier on the right half of the product. Instead of using one bit of the multiplier to determine whether we need to add and shift or merely shift the intermediate step of multiplication, Booth's algorithm uses two right most bits of the product to determine the next step.

Among the many methods of implementing high speed parallel multipliers, there is one basic approach namely Booth algorithm. DSP applications require high computational speed and, at the same time, suffer from stringent power dissipation constraints. Multiplier modules are common to many DSP applications. However, they suffer from a bad regularity. Hence, when regularity, high-performance and low power are primary concerns, Booth multipliers tend to be the primary choice.

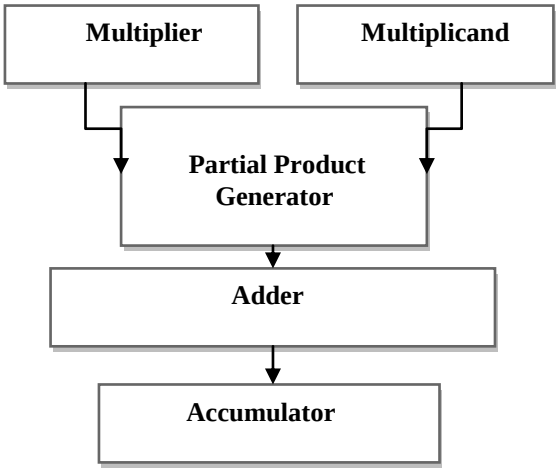

FIG: -BOOTH MULTIPLIER ARCHITECTURE

Booth multipliers allow the operation on signed operands in 2's - complement. They derive from array multipliers where, for each bit in a partial product line, an encoding scheme is used to determine if this bit is positive, negative or zero. The Modified Booth algorithm achieves a major performance improvement through radix-4 encoding.

The Booth's algorithm is also performed with partial products, but it uses several partial product generators together with several adders that operate in parallel. Each partial product obtained is shifted left or right depending on whether the starting bit was the less or the most significant and added up. The number of partial products generated is bound above by the size of the multiplier operand. So, once the sum of the partial products is obtained, the rest of this sum is finally the result of the multiplication.

In an n-bit modified Booth multiplier, the number of Booth encoders is n/2 and the number of partial product generator (PPG) circuits is approximately n2 , hence power consumption and die area in the Booth section is dominated by PPG. So, integration of PPG (Booth Decoder) section is more important than Booth encoder (BE) block. The conventionally used modified Booth selector computes the partial product of jth bit and ith row by using the equation1.

PP=(Xj . X1 _ 2 + Xj _ 1 .X1 _ 1) XOR Neg ----------(1).

Where Xj and Xj-1 are the multiplicand inputs of weight 2j and 2j-1 respectively, X1_2 and X1_1 determine whether the multiplicand should be doubled or not and Neg is a digit which determines if the multiplicand should be inverted or not.

1). Partial Product Generation Circuit

The partial product generation circuit takes the booth encoder output along with the input multiplicand X. If the input

multiplicand and multiplier are n-bits means, then (0……..n/2-1) n/2 number of one dimensional partial product bits are generated.

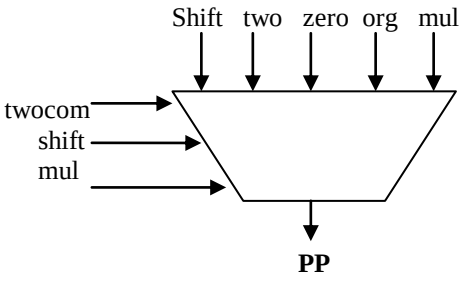

Fig:- PARTIAL PRODUCT GENRATION CIRCUIT.

The 2's complement representation of the multiplicand X is

tmp = ~X + 1'b1……. (1)

The shift, two, zero, org and mul signals can be explained as

shift = {~tmp [n-1], tmp [n-2:0], 1'b0} ……...(2)

two = {~tmp [n-1], tmp}…… (3)

zero = {1'b1, `n'b0} ……(4)

org = {~X [n-1], X}…… (5)

mul = {~X [n-1], X [n-2:0], 1'b0} ……...(6)

Where n is the number of bits in the multiplicand X. The inputs shift, mul, org, two and zero will be based on the multiplicandX. Depending on the selection line bits namely mul, shift and twocom which are the outputs of the booth encoder circuit, one of the inputs, namely shift, org, mul, two and zero which are a one dimensional array, will be selected by the 5X1 multiplexer as a partial product.

More regular and suitable multiplier designs based on the Booth recoding technique have been proposed. The main purpose of these designs is to increase the performance of the circuit by the reduction of the number of partial products. Although the Booth algorithm provides simplicity, it is sometimes difficult to design for higher radices due to the complexity to pre-compute an increasing number of multiples of the multiplicand within the multiplier unit. In the Modified Booth algorithm approximately half of the partial products that need to be added is used.

C). Booth with Radix 2 and Radix 4 Multipliers:-

Booth algorithm itself can be of two types

1. Radix-2 algorithm.

2. Radix-4 algorithm.

1). Radix- 2 Multiplier

This is technique that allows for smaller, faster multiplication circuit by recoding the numbers that are multiplied. It allows only half of product which is needed during computation that is no. Of partial products is reduced by factor 2.

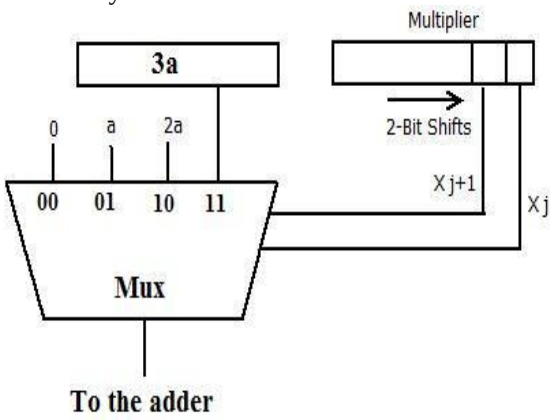

Fig:- BLOCK DIAGRAM OF RADIX-2 MULTIPLIER

It has two major drawbacks

1) As no of_Add and subtraction operations become variable which is inconvenient for parallel multiplier.

2) It is inefficient when there are isolated 1's.

2). Radix 4 Multiplier

These multiplication schemes handle more than one bit of the multiplier in each cycle.

A higher representation radix leads to fewer digits. Thus, a digit-at-a time multiplication algorithm requires fewer cycles as we move to higher radices, which means fewer partial products. The reduction in the number of cycles, along with the use of recoding and carry-save adders, leads to significant gains in speed over basic multipliers.

Mac Sorely proposed a modification of Booth's algorithm a decade after. The modified Booth's algorithm (radix-4 recoding) starts by appending a zero to the right of x0 (multiplier LSB). Carry-save adders (CSA) can be used to reduce the number of addition cycles as well as to make each cycle faster. A row of binary FA is used as a mechanism to reduce three numbers to two numbers, rather than finding a single "sum" A carry save adder is very fast because it simply outputs the carry bits instead of propagating them to the left. As will be presented in the next section, we apply carry save adders in the partial product lines of an array multiplier circuit in order to speed-up the carry propagation along the array.

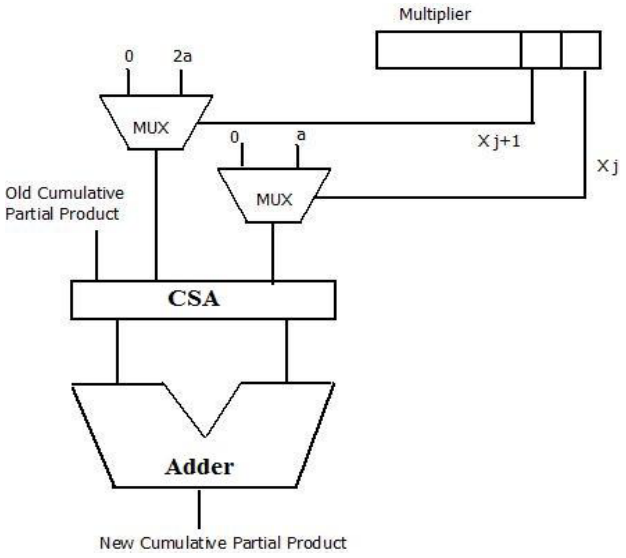

Fig:- BLOCK DIAGRAM OF RADIX-4 MULTIPLIER

a). CARRY SAVE ADDER

As adders are one of the most widely used components in integrated circuits, designing efficient adders has been the goal of much research in VLSI design. First we compute the sum ignoring any carries and separately we can compute carry on a column by column basis. Now sum can be computed by addition's' and 'c' in final stage of addition.

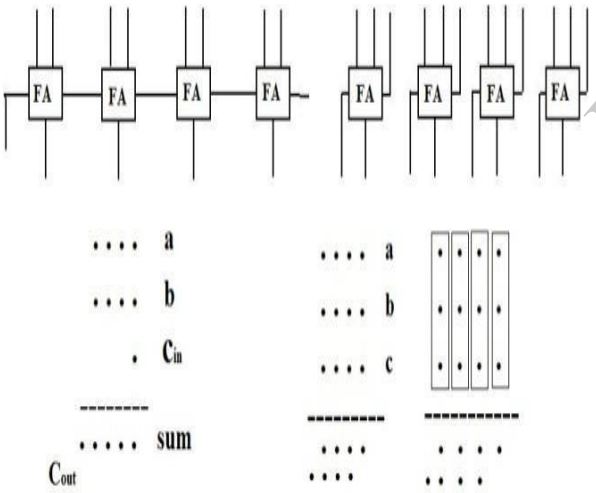

Fig :- BLOCK DIAGRAM OF CSA

b). CARRY SAVE ADDER WITH UNIT ADDERS

If we used unit adders instead of full adders the partial products are further reduced. The unit adders or carry save adders reduce the number of partial products and sum rows. The carry save adder increases the speed of Booth Multiplier structure. As in this the partial products are added sequentially. In this A+B+C+D= (A+B) + (C+D).

That is A and B, C and D are added in parallel. And then they are added together. They require only two full adder delays whereas A+B+C+D require three full adder delays. This is shown in Figure.

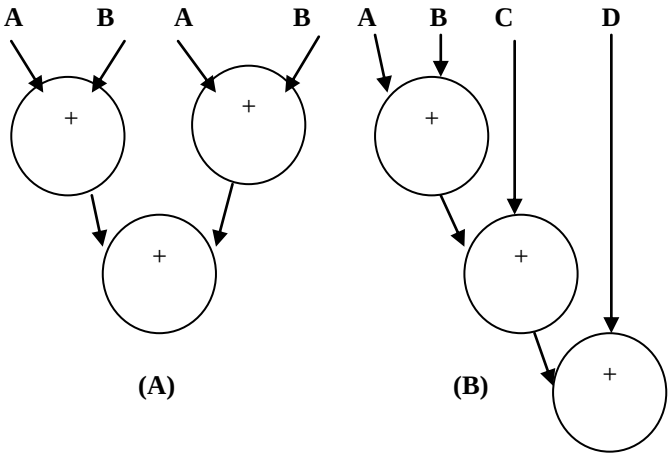

Fig : (A)TWO ADDER DELAY LEVEL
(B) THREE ADDER DELAY LEVEL

A unit adder is having four data inputs and one carry input. It generates sum bits and carryout as the outputs. This is shown in below fig.

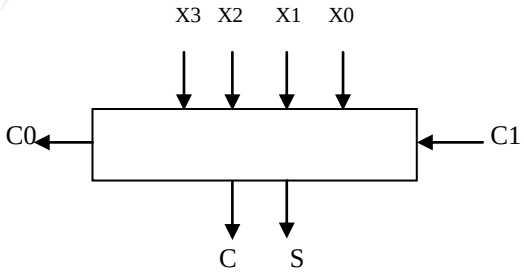

Fig: - A CSA WITH UNIT ADDERS

D) Conclusion

The above comparisons prove that the designed architecture is more efficient than the conventional one in terms of area and delay. Therefore, designed Booth architecture with radix 2 and radix 4 is low area, high speed, simple and efficient for VLSI hardware implementation. Due to efficient performance of designed Booth with radix 4, a tradeoff occurred between delay, area and power. There is a minute increase in power consumption of designed Booth with radix 4 as compared to radix 2 and add and shift multipliers. In case of parallel (Radix

2 and Radix 4) multipliers, the total area is much less than that of serial (add and shift) multipliers. Hence the power consumption is also less. This is clearly depicted in our paper. This speeds up the calculation and makes the system faster. if we want to reduce the delay time we use the radix 8 with FCSA (A type of adder used in digital logic. It can be contrasted with the simpler, but usually slower, ripple carry adder).

E) Future work

One possible direction is radix higher-than-4. We have only considered radix-4 as it is a simple and popular choice. Higher-radix further reduces the number of PPs and thus has the potential of power saving. If we add Wallace tree for design of booth multiplier with Carry Select Adder then we found that the half number of PPs reduction is done. The multiplier is known as Modified Booth Multiplier. MBM (Modified Booth Multiplier) architecture is low area, high speed, simple and efficient for VLSI hardware implementation. Due to efficient performance of MBM with CSA, a tradeoff occurred between delay, area and power. There is a minute increase in power consumption of designed MBM with CSA as compared to high speed Radix multipliers and MBM (Modified Booth Multiplier).the below graph shows the comparison of multipliers with PPs.

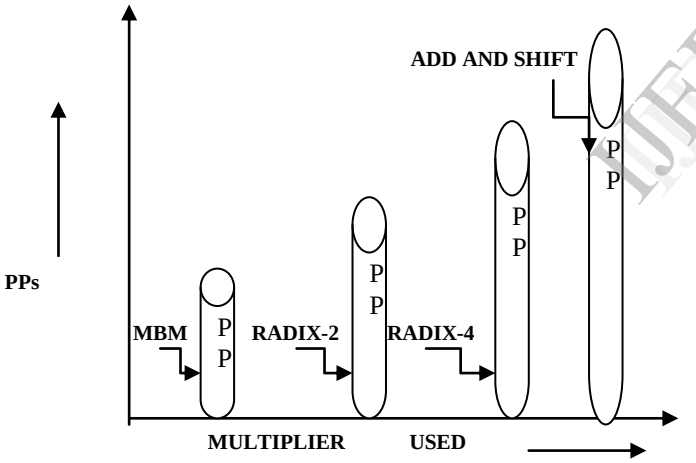

FIG :- Comparison of Multipliers with PPs

F) References

[1].L. Aksoy, E. Costa, P. Flores, J.Monteiro, "Exact and approximant Algorithm for the Optimization of area and Delay in Multiple Constant Multiplications", IEEE Tans. Computer -Aided Design, vol. 27, no. 6, 2008.

[2]. Gary W. Bewick. Fast Multiplication: Algorithms and Implementation. *Technical report CSL-TR*-94-617, Stanford University, April 1994.

[3].Parhami, Behrooz. *Computer Arithmetic: Algorithms and Hardware Designs.* New York: Oxford, 2000.

[4]. Tam, Nguyen, Long Pham, Jon Benson. BOOTH'S Multiplier & 32 Bit ALU for ARM7 microprocessor ECE 345 Initial project proposal Date: FEB /8/ 2000.

[5]. Patterson, David and Hennessy, John. *Computer Organization and Design - The Hardware / Software Interface*. San Francisco: Morgan Kaufmann Publishers, 1998.

[6]. Martinez – Peiro and Lars Wanhammar. Department of Electronic Eng. University of Valencia. High Speed, Low Complexity FIR Filter Using Multiplier Block Reduction and Polyphase Decomposition.

[7]. Ruchi Sharma .Analysis of Different Multiplier with Digital Filters Using VHDL Language. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-2, Issue-1, October 2012.

[8]. Sarita Chouhan1, Rajasthan Technical University, Kota, Rajasthan India Yogesh Kumar2, Rajasthan Technical University, Kota, Rajasthan India. Low Power Designing Of FIR Filters. International Journal of Advanced Technology & Engineering Research (IJATER).

[9]. Kulvir Singh Research Scholar,ACSD C-DAC, Mohali, Punjab , Dilip Kumar Sr.Design Engineer, ACSD C-DAC, Mohali, Punjab. Modified Booth Multiplier with Carry Select Adder using 3-stage Pipelining Technique

"