

# Implementation of High Performance FIR filter using High Speed & Low Area Multiplier

Bhawana Datwani, Himanshu Joshi

**Abstract**— In current scenario, low power consumption and high speed are some of the most important criteria for the fabrication of DSP systems and any high performance systems. Optimizing the speed and power of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas. In our project we are trying to determine the best solution to this problem by comparing a few multipliers and choosing perfect multiplier for implementation of FIR filter. So in this paper designing a FIR filter, which is efficient not only in terms of delay and speed but also in terms of power. The simulations have been carried out using the Xilinx ISE tool.

**Index Terms**— DSP, FIR filter, Multiplier, Xilinx ISE

## I. INTRODUCTION

The multiplier [1]-[3], [5] is one of the key hardware blocks in most of high performance systems such as digital signal processors and microprocessors [2]. With the fast advances in technology, many researchers are working on the most efficient multipliers [5]. They key requirement is not only higher speed and lower power consumption but also occupying reduced silicon area. This makes them well-suited for various complex and portable VLSI circuit [6] implementations. However, the reality is that the area and speed are two conflicting performance factors. Thus, increased speed always results in larger area. In this paper, we found a better trade-off between the two, by realizing a marginally decreased delay which increases the speed performance [3] through a small rise in area such that increase in the number of transistors [6]. The new design lowers the delay of the widely approved Wallace tree multiplier [7]. On the conventional multiplier, the structural optimization is performed, in such a way that the latency of the total circuit reduces significantly. The Wallace tree basically multiplies two unsigned integers [7]. In this project we compare the working & the characteristics of different multiplier [8] individually and then choosing the perfect multiplier by implementing each of them separately in FIR filter.

The parallel multipliers like radix 2 and radix 4, Wallace multiplier [8] perform the computations using less number adders and thus have lesser iterative steps which results in requiring lesser space as compared to the serial multiplier. Here now we are comparing Booth and Wallace multiplier [13] [15] to find the efficient one. Area is a very important factor because in the fabrication of chips [1] and high

performance system, requires components which are as small as possible and covering low space.

## II. FIR FILTER THEORY

Finite Impulse Response (FIR) filter are type of digital filter and consist of impulse response among non-recursive digital filters which is finite in length [3]. FIR filters are non-recursive digital filters such that the current output is calculated solely from the current and previous input values. FIR filter has been selected for this thesis due to their good characteristics. [3][9] FIR filter has no feedback and its input-output relation is given by

$$y(n) = \sum_{k=0}^{n-1} b_k x(n-k) \quad (1)$$

Here,  $x[n]$  and  $y[n]$  are the filter input and filter output respectively,  $a[k]$  is the filter coefficients,  $N$  is the filter coefficient number. The  $\Sigma$  denotes summation from  $k=0$  to  $k=n$  where  $n$  is the number of feed forward taps in the FIR filter. Transfer function of FIR filter can be represented as [3] [9]:

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{n=0}^{N-1} h[n] \cdot z^{-n} \quad (2)$$

The frequency response realized in the time domain is of more concern for FIR filter realization (both hardware and software). The transfer function can be calculated via the z-transform of a FIR filter frequency response [9].

## III. DIGITAL ADDERS

In digital electronics, adder is a type of digital circuit that performs addition of two numbers. As described in [10], many computers and other kinds of processors, adders are common not only in the ALU(s), but also in other parts of the processor, where they calculate addresses, table indices, and many more.

### A. Ripple Carry Adder

A ripple carry adder is a digital circuit that produces the arithmetic sum of two binary numbers. Full adders [12] are cascaded to construct ripple carry adder, with the carry output from each full adder linked to the carry input of the next full adder in the chain. As shown in Figure 1 the interconnection of four full adder (FA) circuits to provide a 4-bit ripple carry adder [9] [12]. It can be seen from Figure the input is coming from the right side because the first cell traditionally represents the least significant bit (LSB). Bits  $a_0$  and  $b_0$  in the figure represent the least significant bits of the numbers to be added. The  $s_0$ - $s_3$  expressing the sum output.

**Manuscript received .**

**Bhawana Datwani**, M.Tech. Scholar, Department of ECE, Jagannath University, Jaipur, India, (e-mail: [bhawana2190@gmail.com](mailto:bhawana2190@gmail.com)).

**Himanshu Joshi**, Assistant Professor, Department of ECE, Jagannath University, Jaipur, (Himanshu.joshi@jagannathuniversity.org)

## Implementation of High Performance FIR filter using High Speed & Low Area Multiplier

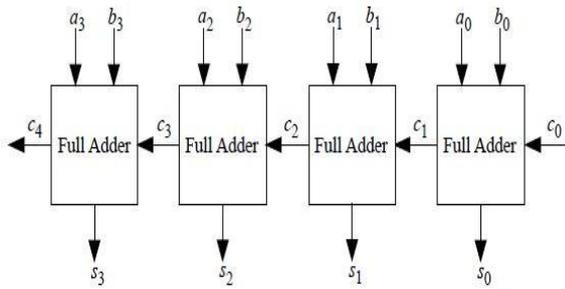


Figure 1 Ripple Carry Adder

### B. Carry Look Ahead Adder

Carry- Look Ahead Adder (CLA) is designed to eliminate the latency introduced by the repelling effect of the carry bits in RCA. The CLA improves speed by reducing the amount of time required to determine carry bits [12]. The concepts of generating (G) and propagating (P) carries is used in CLA. Two signals called *P* and *G* are used in this adder for each bit position [9] [12]. The *P* and *G* are shown below.

$$C_{i+1} = G_i + P_i \cdot C_i \quad (3)$$

Here,  $G_i = A_i \cdot B_i$  and  $P_i = (A_i \text{ xor } B_i)$

$S_i = A_i \text{ xor } B_i \text{ xor } C_i = P_i \text{ xor } C_i$ .

The  $S_i$  and  $C_{i+1}$  represent the sum and carry for  $i^{\text{th}}$  full adder respectively.

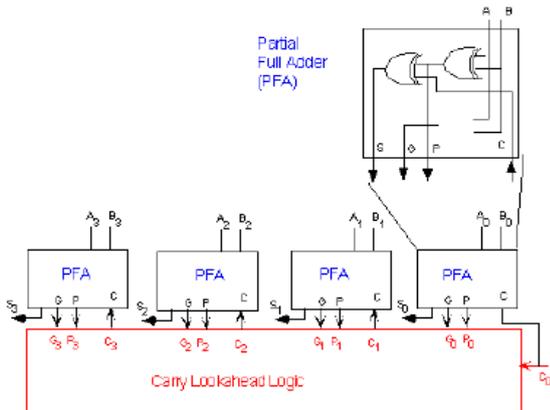


Figure 2 Carry Look ahead Adder

The carry-look ahead adder can be split in two modules:

- (1) The Partial Full Adder , PFA, which generates  $S_i$ ,  $P_i$  and  $G_i$ .
- (2) The Carry Look-Ahead Logic, which generates the carry-out bits.

### C. Carry Select Adder

A Carry Select Adder employ a logic element that evaluate the  $(n+1)$  bit addition of two  $n$ -bit numbers. The carry select adder [10] [12] usually includes two ripple carry adder and a multiplexer. With a carry select adder sum of two  $n$ -bit numbers is done by using two adders (therefore two ripple carry adders) in order to perform the adding up twice, one tie with the appropriation of the carry existence zero and the other assuming one.

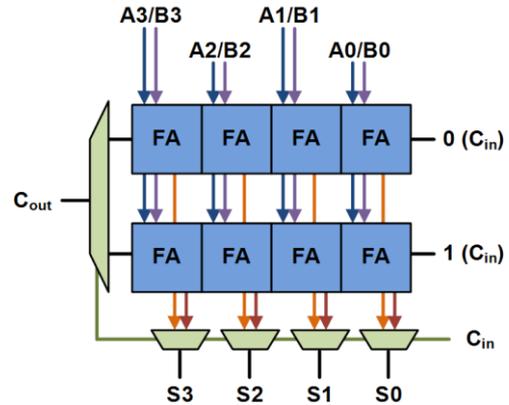


Figure 3 Carry Select Adder

### D. Carry Save Adder

A Carry Save Adder (CSA) is type of digital adder[9][10] which generates low carry signal propagation delay, but in place of adding two input numbers to a single sum output, it adds three input numbers to an output pair of numbers. When its two outputs are then summed by using a carry-look a head or ripple-carry adder[12], we obtain the sum of all three inputs.

## IV. DIGITAL MULTIPLIERS

A Binary multiplier is an electronic digital hardware device used in digital electronics or a computer or other electronic device to perform rapid multiplication of two numbers in binary representation [5] [11]. It is built using binary adders [10]. Multiplier plays an important role in today's digital signal processing and various other applications [2]. In high performance systems such as microprocessor, DSP etc. addition and multiplication of two binary numbers is essential and most often used in arithmetic operations. Statics shows that addition and multiplication is performed in almost 70% instructions in microprocessor and most of DSP algorithms perform [2].

### A. Array Multiplier

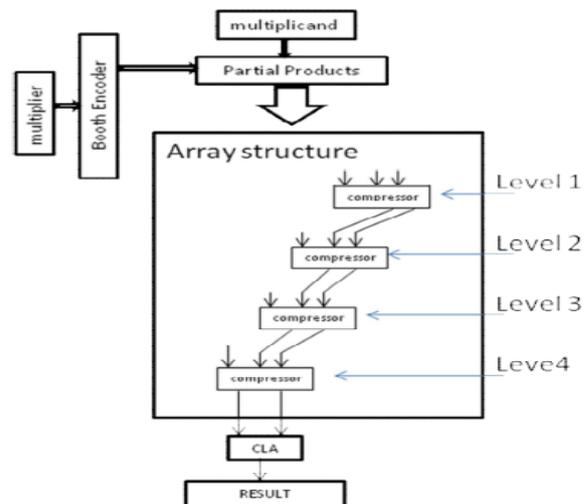


Figure 4 Array Multiplier

Array multiplier is well recognized because of its regular structure. Multiplication process is based on repeated addition and shifting procedure. Each partial product is generated by the multiplication of the multiplicand with one multiplier digit then these partial product are shifted according to their bit sequences and then added [9] [13]. The summation can be performed with normal carry propagation adder. Total N-1 adders are required where N is the no. of multiplier bits. The n-operand array consists of n-2 compressor.

**B. Radix-2 Booth Multiplier**

This is technique that allows for smaller, faster multiplication circuits by recoding the numbers that are multiplied. Partial products is reduced by factor 2 which implies that it allows only half of product which is needed during computation. The Booth's algorithm is for multiplying binary signed number in 2's complement [9] [13]-[14]. Let R and M are the multiplicand and multiplier respectively; and let n and q represent the number of bits in R and M. Take the 2's complement of R which is given as -R. For calculation, make the table of U, V, X and X-1 variable, respectively.

Step1:

- Fill M value in the table.
- Fill 0 for M-1 value it should be the previous first least significant bit of M.
- Fill 0 in U and V rows which show the product of M and X at the end of multiplication operation.
- Take n rows for every cycle; this is because we are multiplying n bits numbers.

Table 1 Making of Booth table

U	V	X	X-1
0000	0000	1100	0

Load the value  
 1<sup>st</sup> cycle  
 2<sup>nd</sup> cycle  
 3<sup>rd</sup> Cycle  
 4<sup>th</sup> Cycle

Step2: Booth algorithm requires evaluation of the multiplier bits, and shifting of the partial product. Use the first least significant bits of the multiplier "M", and the previous least significant bits of the multiplier "M - 1" to determine the arithmetic action.

Table 2 Shift in Booth table

U	V	X	X-1
0000	0000	1100	0
0000	0000	0110	0



- Determine the two least significant (right most) bits of M.
  - If they are 00, no change.
  - If they are 11, no change.
  - If they are 01, add X+A.
  - If they are 10, add (-X)+A.

Table 3 Partial Products generation

U	V	X	X-1
0000	0000	1100	0
0000	0000	0110	0
0000	0000	0011	0

← Shift only

Table 4 Final Shift

U	V	X	X-1
0000	0000	1100	0
0000	0000	0110	0
0000	0000	0011	0
1110	0000	0011	0
1111	0000	1001	1
1111	1000	1100	1

← Shift only

Arithmetically shift the value calculate in step1-4 by signal place of right.

- Take U & V together and arithmetically right shift which store the sign bit of 2's complement number. Hence a positive number and a negative number remains unchanged.
- Right shift circulate M due to this not use of two for the M value.
- Repeat the same steps until the n cycles are completed. So the answer is shown, in the last rows of U and V.

**C. Radix-4 Booth Multiplier**

The shortcomings of Radix-2 can get overcome by Radix-4 [13] [14] [15] in which it handle more than one bit of multiplier in each cycle. The modified Booth's algorithm starts by appending a zero to right of LSB of multiplier. This recoding scheme applied to a parallel multiplier halves the no. of partial products so the multiplication time & hardware requirement can get reduce [8] [14].

Radix-4 Booth algorithm examines strings of three bits according to the following algorithm given below[14]:

- Increase the sign bit 1 position if required to verify that n is even.
- The right side of the LSB of the multiplier adds with 0.
- As per the value of all vectors, all Partial Product will be 0, +y, -y, +2y or -2y.

The values of y are comes negative due to taking the 2's complement. The multiplication of y performs by left shifting y by one bit. As a result implementing of n-bit parallel multipliers, only n/2 partial products are created[9][14].

**D. Wallace Tree Multiplier**

A Wallace tree multiplier[7][15] is an proficient hardware implementation of a digital circuit that multiplies two integers. Number of partial products gets reduced and for the addition of partial products uses carry select adder.

Wallace tree is known for their good computation time, when adding multiple operands to two outputs using 3:2 or 4:2 compressors or both. Wallace tree ensures the lowest whole delay [15]

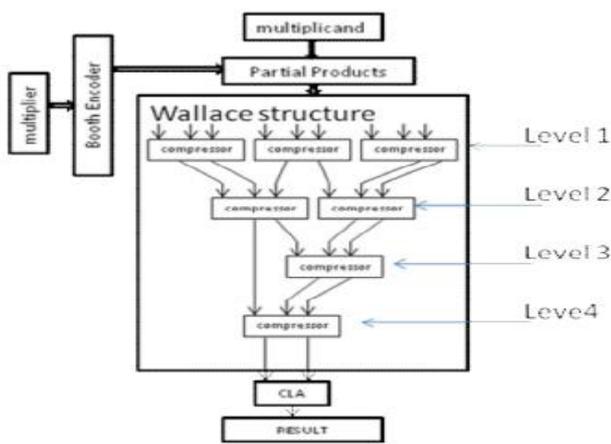


Figure 5 Structure of Wallace Multiplier

V. RESULTS

After analyzing both the multipliers, and compare their characteristics in terms of multiplication speed, no of computations required, no of hardware, we come on finding that Wallace multipliers is much better than Booth multipliers. By implementing both Radix-2 & Radix -4 and Wallace multiplier and we analysis that their computation speed of Wallace multiplier is faster.

In this project these multipliers are implemented with FIR filters to compare some characteristics like the speed, power consumption, computations, hardware requirement of the system. Coding of all the multipliers have done separately in VHDL & simulate it to get the accurate waveforms as output. Then we implement these multipliers separately with FIR filters using computation techniques like FFT, DFT. These coding also written in VHDL language & simulate it to get the RTL circuit of each system. Also get the lookup table , where we get the exact no of i/p, o/p/ no of slices requirement etc for the system. Xilinx Estimator analysis these simulated results & determine the power consumption of each system. These results are given below.

Table 5 Area & Delay of Radix -4 Booth multiplier

No. of slices	204
No. of 4-input LUTs	391
No.of bonded I/Os	65
Delay(ns)	6.177

Table 6 Area & Delay of Wallace Multiplier

No. of slices	9
No. of 4-input LUTs	16
No.of bonded I/Os	16
Delay(ns)	5.895

Table 7 FIR using Different multipliers

Type of multiplier	No.of Slices	No. of 4-input LUTs	No. of bonded I/Os	No. of slice FFs	Delay
Radix-4 Booth Multiplier	157	297	26	45	15.4 4ns

Wallace Multiplier	27	47	21	23	8.51 Ins
--------------------	----	----	----	----	----------

VI. CONCLUSION

This paper is the clear model of different multiplier and their implementation in tap delay FIR filter. We found that the Wallace multipliers are much option than the serial multiplier. We concluded this from the result of delay and the total area. In case of Wallace multipliers, the total area is much less than that of booth multipliers. Hence the power consumption is also less. This is clearly depicted in our results. This speeds up the calculation and makes the system faster. While comparing the radix 2 and the radix 4 booth multipliers we found that radix 4 consumes lesser power than that of radix 2. we found that Wallace multiplication method is better than other multipliers in terms of speed, area and power. So by using Wallace multiplier we can achieve the fast and efficient multiplication.

VII. FUTURE WORK

One possible direction is to increase the number of bit of multiplier. We have only considered 8 bit for encoding as it is a simple and popular choice. Higher number of bits recoding further reduces the number of LUT's and thus has the potential of reducing the area.

REFERENCES

- [1] C. H. Chang, J. Gu, M. Zhang, "Ultra low-voltage low power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits" IEEE Transactions on Circuits and Systems .
- [2] Tan, Li "Digital Signal Processing: Fundamentals and Applications" Edition 2007.
- [3] M.Gnanasekaran, M. Manikandan " Low delay-High compact FIR filter using Reduced Wallace Multiplier" 2nd International Conference on Current Trends in Engineering and Technology, ICCTET'14© IEEE 2014.
- [4] Proakis, J.G., Manolakis, D.G., "Digital Signal Processing" 3rd Edition, PHI publication, 2004.
- [5] Aparna, P.R., Thomas, N. "Design and implementation of a high performance multiplier using HDL" International Conference on Computing, Communication and Applications (ICCCA), 2012 (978-1-4673-0270-8).
- [6] H. B. Bakoglu and J. D. Meindl, "Optimal Interconnection Circuits for VLSI," IEEE Transactions on Electron Devices, Vol. ED-32, No. 5, pp. 903-909, May 1985.
- [7] Vinoth, C.; Bhaaskaran, V.S.K.; Brindha, B.; Sakthikumar, S.; Kavinilavu, 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
- [8] Chen Ping-hua and Zhao Juan; "High-speed Parallel 32x32-b Multiplier Using a Radix-16 Booth Encoder," Third International Symposium on Intelligent Information Technology Application
- [9] D.Jaya Kumar, Dr.E.Logashanmuga "Performance Analysis of FIR filter using Booth Multiplier" 2nd International Conference on Current Trends in Engineering and Technology, ICCTET'14© IEEE 2014.
- [10] H.Chang, J.Gu, M.Zhang (2004) ,"Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits", Circuits and Systems Regular Papers, IEEE Transactions page(s): 1985- 1997, Volume: 51, Issue: 10, Oct. 2004. Rashmi Ranjan et al./ International Journal of Computer Science & Engineering Technology (IJCSET)
- [11] Aparna, P.R., Thomas, N. "Design and implementation of a high performance multiplier using HDL" International Conference on Computing, Communication and Applications (ICCCA), 2012 (978-1-4673-0270-8)

- [12] Rashmi Solanki, Prashant Gurjar, Pooja Kansliwal, Mahendra Vucha "VLSI Implementation of Adders for High Speed ALU" International Journal of Computer Application, September 2011.
- [13] 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, 04 Feb. 2010.
- [14] M Bansal, "High performance pipelined signed 64\*64-bit multiplier using radix-32 modified Booth algorithm and Wallace structure", International Conference on Computational Intelligence and Communication Systems, 2011.
- [15] Abdullah A. AlJuffri, Aiman S. Badawi, Mohammed S. Ben Saleh, Abdul Fattah M. Obeid, Syed Manzoor Qasim "FPGA Implementation of Scalable Microprogrammed FIR Filter Architectures using Wallace Tree and Vedic Multipliers" IEEE, 2015 (978-1-4799-5680-7/15)

**Bhawana Datwani** M.tech. scholar, Jagannath University. I have completed my B.Tech. in Electronics & Communication from RTU in 2012. I have experience of 3 years of teaching. I am doing my research work in VLSI field.

**Himanshu Joshi** Assistant Professor Department of ECE in Jagannath University, Jaipur, India. He has completed his M.Tech (VLSI and Embedded system) in 2011 from Gyan Vihar University, Jaipur, and B.E degree in 2007 from Rajasthan University. He is currently working in the VLSI and Communication Field.