© 2018 IJSRST | Volume 4 | Issue 9 | Print ISSN : 2395-6011 | Online ISSN : 2395-602X
Themed Section: Science and Technology

Configurable Fir Filter Using Different Multiplier Technique

Anubhav Shankhwar[1], Shweta Agarwal[2]
[1]Research scholar, Department of Electronics and Comm., RGPV Bhopal (M.P.), SRCEM Banmore, Morena, India
[2]Assistantce professor, Department of Electronics and Comm., RGPV Bhopal (M.P.), SRCEM Banmore, Morena, India

ABSTRACT

Finite-Impulse Response (FIR) filler is playing the role of the backbone of the modern communication systems. The filler is mainly used to separate the required signals from the unwanted signals. During the separation the number of algorithms are employed which are responsible for the filtration speed. Today's era of technology demands the maximum operating frequency. In order the boost the filtration the components of filler must be efficient. The speed of FIR filter is mainly depends on multiplier used in it. For this reason the FIR filler which is depicted here have the highly efficient multiplier. This presented filler can also configure for the different filtering co-efficient. Transpose form finite-impulse response (FIR) filters are naturally pipelined and support various consistent multiplications (MCM) technique that results in significant saving of computation. However, transpose form structure does not directly support the block processing unlike direct-form configuration. In this paper, we explore the possibility of realization of block FIR filter in transpose form configuration for area-delay efficient realization of large order FIR filters for both fixed and reconfigurable applications. Based on a detailed computational analysis of transpose form configuration of FIR filter, we have derived a flow graph for transpose form block FIR filter with optimized register complexity. A generalized block formulation is presented for transpose form FIR filter. We have derived a general multiplier-based architecture for the proposed transpose form block filter for reconfigurable applications. A low-complication design is using by the MCM scheme is also presented for the block implementation of fixed FIR filters. The proposed structure involves significantly less area-delay product (ADP) and less energy per sample (EPS) than the existing block implementation of direct-form structure for medium or large filter lengths, while for the short-length filters, the block implementation of direct-form FIR structure has less ADP and less EPS than the proposed structure.

Keywords: FIR filter, Multiplier, Digital filter, DSPs, FPGA, MCSA, CSA, VLSI.

I. INTRODUCTION

Finite-Impulse Response (FIR) digital filter that is widely used in several digital signal processors applications, such as speech processing, loud speaker equalization, echo cancellation, adaptive noise cancellation, and various communication applications, including software-defined radio (SDR) and so on [1]. Many of these applications require FIR filters of large order to meet the stringent frequency specifications [2]–[4]. These filters need to support high sampling rate for high-speed digital communication [5]. The number of multiplications and additions required for each filter output, however, increases linearly with the filter order. As the word indicates, a filter separates a desired signal from unwanted disturbances. For example, when we want to remove a disturbance such as noise from an audio signal, we design an appropriate filter that passes only the desires signal. But only in a few cases can we remove the disturbance completely and recover the desired signal; most of the time we have to settle for a compromise, most of the

disturbance is rejected, most of the signal is recovered. The first candidate in filter is a linear filter. The main reason for this choice is that we have a good understanding of how a linear system operates. It is only when a linear design fails or it yields unsatisfactory results that we look for other solutions, such as nonlinear or, adaptive techniques,

FIR filter plays an important role in digital signal processing, comparing with analogy filter, it has many merits: stability, strong reliability, without voltage floating and noise problem. Besides, its non-recursive construction contributes to its fast operation rate and low error. What'more, its linear phase property makes it used widely in many fields such as differential meter, integrator, image processing, data transmission and so on.

FIR filter architecture has multiplier, adder and delay unit. So FIR filter performance is mainly based on multiplier. In this paper presents FIR filter implantation of Booth multiplier using Modified Carry Save Adder (MCSA) and Carry Save Adder (CSA). These techniques are used to improve the performance of delay and Area. The code is written in Verilog HDL and it is simulated in ModelSim 10.4c and synthesis is done in Xilinx Vivado. Finally the design is implemented in Kintex 7 KC705 FPGA. Different types of multipliers are used for comparing the results such as Booth multiplier, vedic multiplier etc.

II. FUNDAMENTALS OF BOOTH ALGORITHM

C	A	Q	M	Comment
0	0000	1010	1100	Initial value
0	0000	0101	1100	Since Q0=0 Shift C,A,Q by 1bit
0	1100	0101	1100	Since Q0=1 C,A A+M
0	0110	0010	1100	Shift C,A,Q by 1 bit
0	0011	0001	1100	Since Q0=0 Shift C,A,Q by 1 bit
0	1111	0001	1100	Since Q0=1 C,A A+M
0	0111	1000	1100	Shift C,A,Q by 1 bit

Booth multiplication algorithm or Booth algorithm was found after the inventor Andrew Donald Booth. It can be defined as an algorithm or method of multiplying binary numbers in two's complement notation. It is a simple method to multiply binary numbers in which multiplication is performed with repeated addition operations by following the booth algorithm [6]. After using this booth algorithm for multiplication some operation is further modified and since, named as modified booth algorithm.

Booth multiplication algorithm be expressed by three major steps as displayed in the design of booth algorithm figure that includes generation of partial product called as recoding, reducing the partial product in two rows, and addition that gives final product [7]. For a better kind of modified booth algorithm & for multiplication, we must know about each block of booth algorithm for multiplication process.

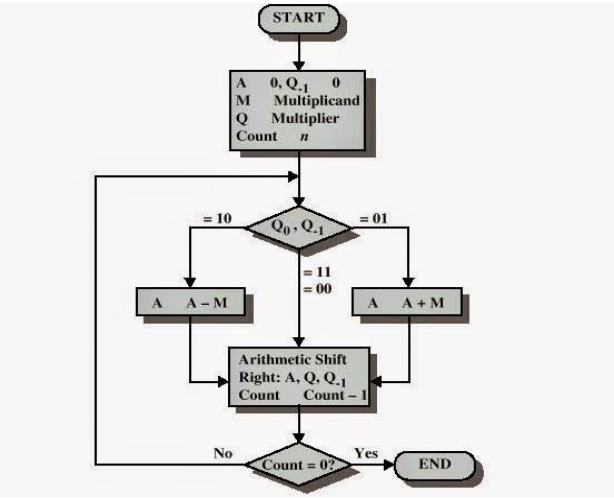

Figure 1. Booth Algorithm for Two's Complement Multiplication

Example of signed binary multiplication

1010*1100

Answer (AQ)=01111000

III. PROPOSED CONFIGURABLE FIR FILTER USING BOOTH ALGORITHM

The proposed co-efficient representation technique uses pipelining technique in order to increase the performance. We explore the possibility of realization of block FIR filter in transpose form configuration for area-delay efficient realization of large order FIR filters for both fixed and reconfigurable applications. Based on a detailed computational analysis of transpose form configuration of FIR filter, we have derived a flow graph for transpose form block FIR filter with optimized register complexity.

The output of an FIR filter of length N can be computed using the relation

$$y(n) = \sum_{i=0}^{N-1} h(i) \cdot x(n-i). \quad (1)$$

The computation of (1) can be expressed by the recurrence relation

$$Y(z) = [z^{-1}(\cdots(z^{-1}(z^{-1}h(N-1) + h(N-2)) + h(N-3)) \cdots + h(1)) + h(0)]X(z). \quad (2)$$

A generalized block formulation is presented for transpose form FIR filter. We have derived a general multiplier-based architecture for the proposed transpose form block filter for reconfigurable applications. A low-complexity design using the MCM scheme is also presented for the block implementation of fixed FIR filters. The proposed structure involves significantly less area delay product (ADP) and less energy per sample (EPS) than the existing block implementation of direct-form structure for medium or large filter lengths, while for the short-length filters, the block implementation of direct-form FIR structure has less ADP and less EPS than the proposed structure.

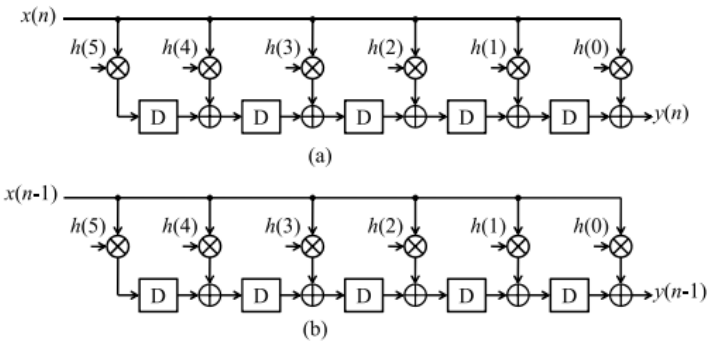

Figure 2. DFG of transpose form structure for N = 6. (a) DFG-1 for output y(n). (b) DFG-2 for output y(n - 1).

The data-flow graphs (DFG-1 and DFG-2) of transpose form FIR filter for filter length $N = 6$, as shown in Figure 2, for a block of two successive outputs *y(n), y(n*1*)* that are derived from (2). The product values and their accumulation paths in DFG-1 and DFG-2 of Figure 2 are shown in data- flow tables (DFT-1 and DFT-2) of Figure 2. The arrows in DFT-1 and DFT-2 of Figure 2 represent the accumulation path of the products. We find that five values of each column of DFT-1 are same as those of DFT-2. These redundant computations of DFG-1 and DFG-2 can be avoided using non-overlapping sequence of input blocks, as shown in Figure 3. DFT-3 and DFT-4 of DFG-1 and DFG-2 for no overlapping input blocks are, respectively, shown in Figure 3(a) and (b). As shown

in Figure 3(a) and (b), DFT-3 and DFT-4 do not involve redundant computation. It is easy to find that the entries in gray cells in DFT-3 and DFT-4 of Figure 3(a) and (b) correspond to the output *y(n)*, whereas the other entries of DFT-3 and DFT-4 correspond to *y(n* 1*)*. The DFG of Figure 1 needs to be transformed appropriately to obtain the computations according to DFT-3 and DFT-4.

ccs	M_1	M_2	M_3	M_4	M_5	M_6
1	$x(n\text{-}5)h(5)$	$x(n\text{-}5)h(4)$	$x(n\text{-}5)h(3)$	$x(n\text{-}5)h(2)$	$x(n\text{-}5)h(1)$	$x(n\text{-}5)h(0)$
2	$x(n\text{-}4)h(5)$	$x(n\text{-}4)h(4)$	$x(n\text{-}4)h(3)$	$x(n\text{-}4)h(2)$	$x(n\text{-}4)h(1)$	$x(n\text{-}4)h(0)$
3	$x(n\text{-}3)h(5)$	$x(n\text{-}3)h(4)$	$x(n\text{-}3)h(3)$	$x(n\text{-}3)h(2)$	$x(n\text{-}3)h(1)$	$x(n\text{-}3)h(0)$
4	$x(n\text{-}2)h(5)$	$x(n\text{-}2)h(4)$	$x(n\text{-}2)h(3)$	$x(n\text{-}2)h(2)$	$x(n\text{-}2)h(1)$	$x(n\text{-}2)h(0)$
5	$x(n\text{-}1)h(5)$	$x(n\text{-}1)h(4)$	$x(n\text{-}1)h(3)$	$x(n\text{-}1)h(2)$	$x(n\text{-}1)h(1)$	$x(n\text{-}1)h(0)$
6	$x(n)h(5)$	$x(n)h(4)$	$x(n)h(3)$	$x(n)h(2)$	$x(n)h(1)$	$x(n)h(0)$

(a)

ccs	M_1	M_2	M_3	M_4	M_5	M_6
1	$x(n\text{-}6)h(5)$	$x(n\text{-}6)h(4)$	$x(n\text{-}6)h(3)$	$x(n\text{-}6)h(2)$	$x(n\text{-}6)h(1)$	$x(n\text{-}6)h(0)$
2	$x(n\text{-}5)h(5)$	$x(n\text{-}5)h(4)$	$x(n\text{-}5)h(3)$	$x(n\text{-}5)h(2)$	$x(n\text{-}5)h(1)$	$x(n\text{-}5)h(0)$
3	$x(n\text{-}4)h(5)$	$x(n\text{-}4)h(4)$	$x(n\text{-}4)h(3)$	$x(n\text{-}4)h(2)$	$x(n\text{-}4)h(1)$	$x(n\text{-}4)h(0)$
4	$x(n\text{-}3)h(5)$	$x(n\text{-}3)h(4)$	$x(n\text{-}3)h(3)$	$x(n\text{-}3)h(2)$	$x(n\text{-}3)h(1)$	$x(n\text{-}3)h(0)$
5	$x(n\text{-}2)h(5)$	$x(n\text{-}2)h(4)$	$x(n\text{-}2)h(3)$	$x(n\text{-}2)h(2)$	$x(n\text{-}2)h(1)$	$x(n\text{-}2)h(0)$
6	$x(n\text{-}1)h(5)$	$x(n\text{-}1)h(4)$	$x(n\text{-}1)h(3)$	$x(n\text{-}1)h(2)$	$x(n\text{-}1)h(1)$	$x(n\text{-}1)h(0)$

(b)

Figure 3. (a) DFT of multipliers of DFG shown in Figure 1(a) corresponding to Output y(n). (b) DFT of multipliers of DFG shown in Figure 1(b) corresponding to output y(n–1). Arrow: accumulation path of the products.

After using the booth multiplier in the FIR fillter we found that the number of LUTs are reduced. Not only the system become compact but also able to work at lower power requirment. There is masive change in other parameter also which are shown in the result section of the paper.

IV. VEDIC MULTIPLIER

Vedic multiplier is using the techniques of vedic mathematics it is based on the 16 sutras which perform operation on mathematics like algebra, geometry, airhtmetic.The proposed vedic sutra is Uradhava Triyagbhyam. It is the most generalised sutra for implementation of vedic multiplication. It uses vertically and crosswise multiplication technique the example for vedic multiplier is shown in below figure.The multiplier is based on this sutra has the advantage that as the number of bit increases, gate delay and area increases very slowyly as comapred to other conventional mulitpliers.

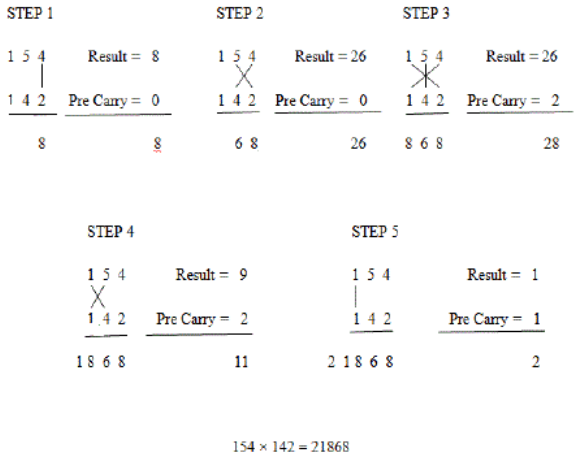

Figure 4. Multiply ow two decimal numbers by using Urdhava Triyagbhyam technique

V. RESULTS

In this paper, a digit-reconfigurable FIR filter architecture is proposed. The design concepts of the architecture and the circuit are presented. Testing results show that the fabricated chip draws only 16.5 mW from a 2.5-V power supply while running at 86 MHz.

Table 2

Multiplier	Delay (ns)	Number of flip Flop	LUT	Power (mw)
Conventiona l Multiplier	5.399	307	6646	0.494
Vedic Multiplier	5.337	284	4315	0.383
Booth Multiplier	5.131	279	3577	0.327

In the present work Implementation of digital filters by FPGA device shows superior performance than DSP processors with respect to frequency, time and filtering of the signal. FIR filter shows better results than IIR filters to get good operating frequency and better phase. FPGAs are more sophisticated, perform well and integrating more DSP blocks in FPGAs is becoming the standard and more efficient. FPGA devices consume less power than General Purpose Processors. The configuration files can be stored or generated as and when required and downloaded into FPGA to reconfigure them. Therefore it is more advantage for remote user. As FPGAs increase in density and performance capability, more signal processing functions can be incorporated to the radar system for filtering and signal correlation. Integrating DSP functions into a single FPGA, the system-level will be reduced in size, weight and power. Thus FGPGA based signal processing is best suitable for radar applications to capture the features and eliminate the noise in the signal to get accurate wind velocity, particle density, etc in the space.

The output waveform is given in Figure 5&6. It show that when we put certain values how it provides the output. The wave is taken from the Xilinx Vivado 2017. It is screen shot of the wave to test the simulate the code written in the verilog HDL.

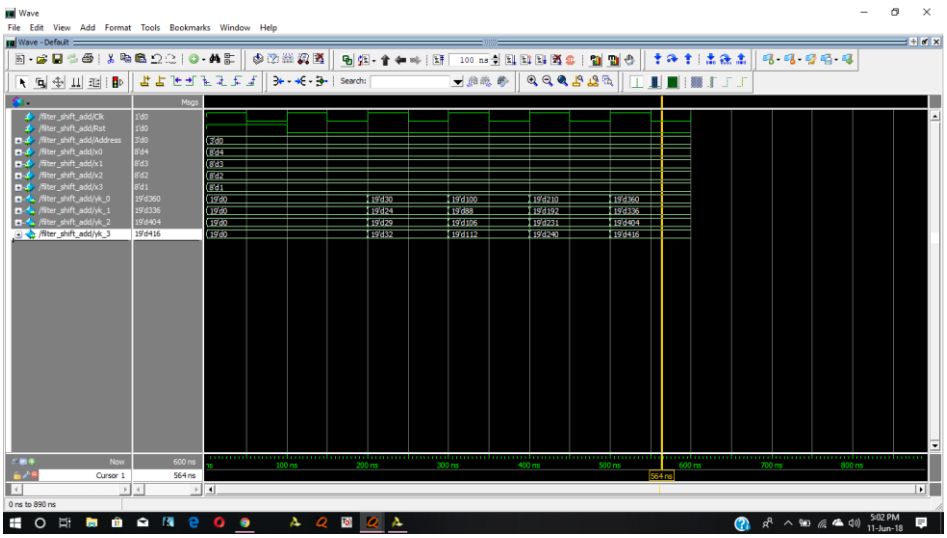

Figure 5. Output waveform of Booth's multiplier

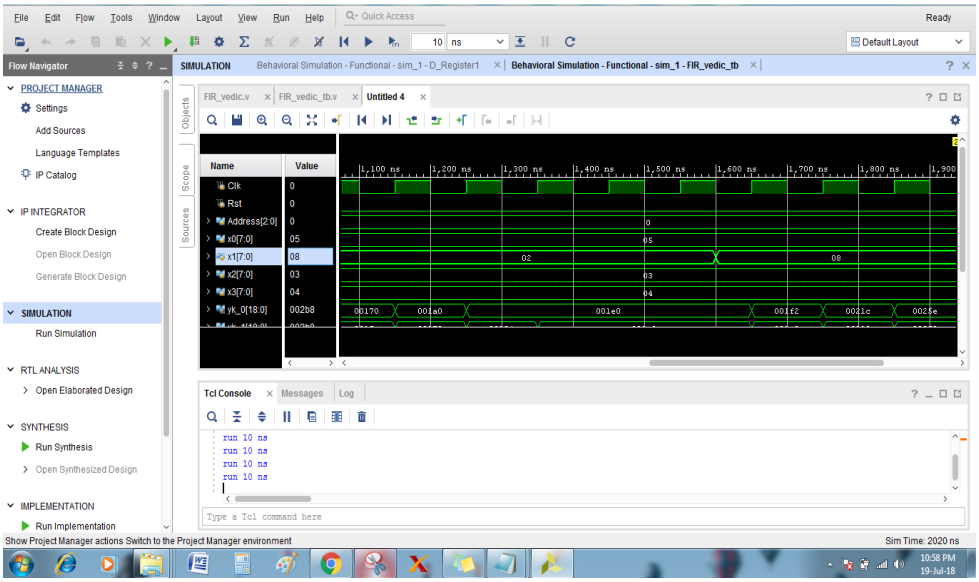

Figure 6. Wave form of Vedic Multiplier

VI. SCOPE FOR THE FUTURE WORK

The future research is going on FPGA to develop automatic control switches for MST radar centre. The system is subjected to continuous development at present. There is much left to be done with the field programmable systems, with a lot of promise yet to be fulfilled. The FPGAs may become the cornerstone of many future computation systems. FPGAs are becoming an integral part of system design. Increasingly, more functionality is being ported to the FPGA. However, FPGAs, traditional DSPs and GPPs are going to coexist, and a flexible platform will include a mix of all. FPGA based system design should be treated like a hardware design exercise and not just a software design project planning stage. With an FPGA implementation,

VII. REFERENCES

[1]. Prof. Pushpalata Verma, K. K. Mehta" Implementation of an Efficient Multiplier based on Vedic Mathematics Using EDA Tool" International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-5, June 2012.

[2]. Asmita Haveliya, "A Novel Design for High Speed Multiplier for Digital Signal Processing Applications (Ancient Indian Vedic mathematics approach)", Int. Journal of Technology and Engineering System (IJTES), V 2, n1,pp. 27-31, Jan March, 2011.

[3]. Ware, F.A., McAllister, W.H., Carlson, J.R., Sun, D.K., and Vlach, R.J., 64 Bit Monolithic Floating Point Processors,

[4]. IEEE Journal of Solid-State Circuits, vol. 17, no.

[5]. October 1982, pp. 898-90. 5. P. M. Kogge and H. S. Stone, A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations

[6]. Tanner EDA Inc 1988, User's Mannual,2005

[7]. Parth Mehta and Dhanashri Gawali, "Conventional versus Vedic mathematics method for Hardware implementation of a multiplier", International conference on Advances in Computing, Control, and Telecommunication Technologies, pp. 640- 642, 2009.