Design & Implementation of Multiplier less FIR FILTER

B.Madhu
Pg Scholar(Vlsi&Es)
Dept Of Ece
Vidya Bharathi Institute Of Technology
Janagaon, Warangal, Telangana

Shanigarapu Naresh Kumar
Associate Professor
Dept Of Ece
Vidya Bharathi Institute Of Technology
Janagaon, Warangal, Telangana

B Kranthi Kumar
Associate Professor
Dept Of Ece
Vidya Bharathi Institute Of Technology
Janagaon, Warangal, Telangana

Abstract: In this paper, FPGA realization of MUX based multiplier and odd multiple scheme architectures are proposed for FIR filter and discussed in terms of complexity. In digital filter implementation, the multiplier usage is avoided by using MUX based multiplier and Look Up Table (LUT) based multiplier. These multipliers are used for constructing direct form FIR filters with signed number representation. The two architectures have been implemented using Verilog and synthesized using SPARTRAN XC3S500E. The performance is analyzed for mux based and LUT based Multiplier.

Keywords:-FIR Filter, Look up Table,Reconfigurable Architecture, Distributed Arithmetic.

I.INTRODUCTION

Finite Impulse Response (FIR) digital filter is widely used as a basic block in signal and image processing applications. The number of multiply-accumulate (MAC) operations required per filter output increases linearly with the order of filter, but implementation of higher order filters in real-time is another challenging task. Recently with the advent of software defined radio (SDR), the research has been concentrated on realization of FIR filters[I][2][3][4] mainly due to its high flexibility and low complexity[5][6][7]. The digit-based reconfigurable architecture presented in [3] provides a flexible and low power solution with a wide range of precision and tap length of FIR filters. Conventionally, the FIR filters are designed based on programmable multiply-accumulate [MAC] architecture [6] and systolic architecture [7]. The performances of the designs are analyzed in terms of hardware complexity, power consumption and throughput. In [6], the programmable MAC architectures consume low power with reduced supply voltage and it requires large area. In [7], even though systolic based architecture reduces the complexity, it increases the latency when the order of the filter increases. Several attempts have been made and it continued to develop low-complexity dedicated VLSI systems for these filters. There are several issues in the hardware implementation of Digital filters. The direct implementation of N-tap FIR filters which requires N MAC operations are too expensive to implement in hardware due to its logic complexity and area requirement. Distributed Arithmetic (DA) and LUT constitute memory-based techniques. Among them, DA is applied for inner product computation [8]-[17], LUT provides computation of multiplication [18]-[24]. In the LUT based approach, multiplications of input values with a fixed coefficient and consisting of all possible pre-computed product values correspondding to all possible values of input multiplicand, while in the DA-based approach, an LUT is used to store all possible values of inner-products of a fixed-point vector. If the inner-products are implemented directly, the memory-size of LUT-multiplier based implementation increases exponentially with the input word length, whereas the memory size of the DA-based approach increases exponentially with the inner-product-Iength.

It is observed that the memory-size reduction is achieved by such decompositions is accompanied by raise in latency as well as the number of adders and latches. In this paper FPGA realization of the two FIR filter architectures such as Odd multiple storage scheme [4] and MUX based multiplier are discussed.

Both Odd multiple storage scheme and MUX based multiplier have multiplier less architecture to reduce complexity in the design, by manipulating the odd multiples of the fixed coefficient in LUT design, and general multipliers replaced by shift and add operations respectively.

The performances of the proposed architectures are analyzed in terms of area and time complexities by varying the number of taps.

The rest of the paper is organized as follows. Section II describes the details of Multiplier less architecture design of FIR Filter and their implementation. The performance of the design are analyzed and discussed in Section III. Finally, Section IV concludes the paper.

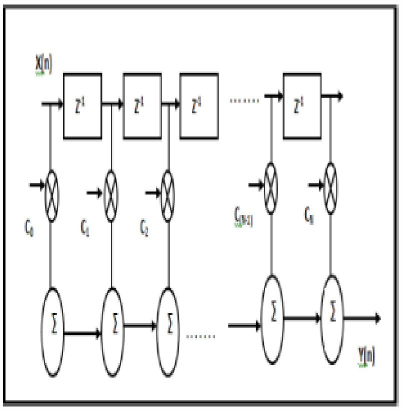


Figure-1:General structure of an FIR filter

II. FIR FILTERS

In signal processing, the filter is used to remove some unwanted component or feature from a signal thereby improving the quality of signal. It alters the amplitude and/ or phase characteristics of a signal in a desired manner with respect to frequency. The primary function of filter are – to confine a signal into a prescribed frequency band, to decompose a signal into two or more sub-bands, to modify the frequency spectrum of a signal and to model the input output relationship of a system. Filters are extensively used in signal processing and communication system in applications like noise reduction, echo cancellation, image enhancement, speech and waveform synthesis etc. There are two main kind of filter: analog and digital filter. Analog filter has analog signal at both its input & output and are made up from components such as resistors, capacitors and op amps to produce the required filtering effect. Such filters are fast and simple to realize but are little stable, sensitive to temperature variations and expensive to realize in large amounts.

Digital filter on the other hand uses digital processor to perform numerical calculations on sampled values of the signal and eliminate the problems associated with their classical analog counterparts, thus are preferably used in place of analog filter [1]. Broadly, digital filters are classified as: Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filter. FIR filters have linear phase, stability, fewer finite precision errors, and efficient implementation hence preferred over IIR filter.

FINITE IMPULSE RESPONSE FILTER

Finite impulse response (FIR) filters are a class of digital filters that have a finite impulse response and are among one of the primary types of filter used in DSP and communication system [3]. They do not have any feedback and therefore if excited by impulse response, the ouput will invariably become zero. The input- output relationship of FIR filter is given by (1) y $n = (k)$ $(n - k$ $N-1$ $k=0$ $)$ (1) where, p(k), k = 0,1,2,3……N-1 are the impulse response coefficients of the filter. N is the filter length that is number of coefficients.

TRANSPOSED DIRECT FORM OF AN FIR FILTER:

As shown in Fig. 1[1], FIR filtering operation performs the weighted summations of input sequences, called as convolution sum, which are frequently used to implement the frequency selective low-pass, high-pass, or band-pass filters .Generally, since the amount of computation and the corresponding power consum ption of FIR filter are directly proportional to the filter order, if we can dynamically change the filter order by turning off some of multipliers, significant power savings can be achieved. However, performance degradation should be carefully considered when we change the filter order.

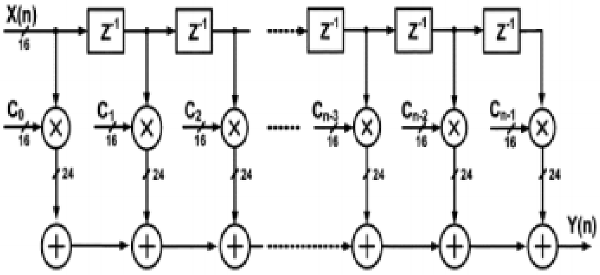


Figure-2: Transposed direct form of an FIR filter

III. EXISTING SYSTEMS

MULTIPLlER LESS STRUCTURES BASED FOR FIR FILTER

Figure 1 shows an implementation of N tap FIR filter with the usage of three elements namely adders, multipliers and delay elements. Let X(n) and Y(n) be the input and output sequences of the FIR filter respectively. Consider an N-tap FIR filter that can be formulated as

$$Y(n)=\sum_{k=0}^{N-1} h_k X(n-k) \qquad (1)$$

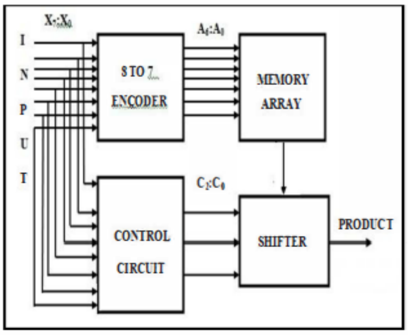


Figure-3- Proposed architecture for odd multiple storage scheme

TABLE I ODD MULTIPLE STORAGE SCHEME

Symbol	Address $A_6A_5A_4A_3A_2A_1A_0$	Stored value	Input $X_7X_6X_5X_4X_3X_2X_1X_0$	Product value	# of shifts	Control $C_2C_1C_0$
P_0	0	A	1	A	0	0
			10	2A	1	1
			100	4A	2	10
			1000	8A	3	11
			10000	16A	4	100
			100000	32A	5	101
			1000000	64A	6	110
			10000000	128A	7	111
P_1	1	3A	11	3A	0	0
			110	6A	1	1
			1100	12A	2	10
			11000	24A	3	11
			110000	48A	4	100
			1100000	96A	5	101
			11000000	192A	6	110
⋮	⋮	⋮	⋮	⋮	⋮	⋮
P_{121}	1111001	243A	11110011	243A	0	0
P_{122}	1111010	245A	11110101	245A	0	0
P_{123}	1111011	247A	11110111	247A	0	0
P_{124}	1111100	249A	11111001	249A	0	0
P_{125}	1111101	251A	11111011	251A	0	0
P_{126}	1111110	253A	11111101	253A	0	0
P_{127}	1111111	255A	11111111	255A	0	0

where hk is the kth coefficient of the filter impulse response. In the above equation, the filter coefficients do not vary with regard to the input signal or the noise. This implies that the coefficients are fixed. Such filters whose coefficients are fixed are called as fixed filters or filters with fixed characteristics . In the filter the MAC structure and delay blocks are the main building blocks. The performance of the filter can be enhanced by the introduction of two architecture schemes which reduces the complexity and critical path. The two schemes discussed here are ODD and MUX based multiplier. A. Odd Multiple Storage Scheme This method proposed by [4] stores the product of odd multiple of co-efficient and the input. The advantage of storing odd multiple is that even multiples can be obtained by a simple left shift operation. For any input vector of N bits the number of locations in the LUT is identified by 2N. However, the odd multiple storage scheme reduces the number of locations by a factor of 2. Considering an input vector of 8 bits, uses only 128(27) unlike 256 locations of conventional

LUT. Table I gives the complete insight about the look up table of odd multiple storage scheme. The 7-bit address (A6-AO) given in Table I identifies the stored values of LUT and it also gives a complete detail about the number of shifts to be performed for the corresponding input bits Hence for the input bits'1O' the LUT stored value A is shifted one time to get 2A.The control bits c2 cl cO decides the number of shifts to be performed. The Figure 2 gives the proposed structureof the odd multiple storage schemes. The architecture has an 8 to 7 bit encoder which maps the 8-bit input word(x7:xO) to a 7 bit address(A6:AO) according to the logical relations.

$$A_6 = X_7.X_0 \quad (2\text{-a})$$

$$\begin{aligned}A_5 &= (X_6.\overline{X_5}.\overline{X_4}.\overline{X_3}.X_2.X_1.X_0) + (X_7.X_6.X_5.\overline{X_4}.\overline{X_3}.X_2.X_1.X_0) + (X_6.X_4.\overline{X_3}.X_2.X_1.X_0)\\ &+(X_5.X_1.X_0.\overline{X_2}) + (\overline{X_6}.X_1.X_0) + (\overline{X_6}.X_5.X_1) + (X_6.X_3.X_2.X_1.X_0)\end{aligned} \quad (2\text{-b})$$

$$A_4 = (X_7.X_2.\overline{X_1}.\overline{X_0}) + (X_6.X_1.\overline{X_0}) + (X_5.X_0) \quad (2\text{-c})$$

$$A_3 = (X_7.X_3.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.X_2.\overline{X_1}.\overline{X_0}) + (X_5.X_1.\overline{X_0}) + (X_4.X_0) \quad (2\text{-d})$$

$$\begin{aligned}A_2 &= (X_7.\overline{X_6}.X_5.\overline{X_4}.X_3.X_2.X_1.\overline{X_0}) + (X_7.\overline{X_6}.X_5.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_7.X_6.X_5.X_4.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.X_3.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_5.X_2.\overline{X_1}.\overline{X_0}) + (X_4.X_1.\overline{X_0} + (X_3.X_0)\end{aligned} \quad (2\text{-e})$$

$$\begin{aligned}A_1 &= (\overline{X_7}.\overline{X_6}.\overline{X_5}.X_4.X_3.\overline{X_2}.\overline{X_1}.X_0) + (X_7.\overline{X_6}.\overline{X_5}.X_4.X_3.\overline{X_2}.X_1.\overline{X_0})\\ &+(X_7.\overline{X_5}.X_4.X_3.\overline{X_2}.X_1.\overline{X_0}) + (X_7.X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(\overline{X_4}.X_3.\overline{X_2}.X_1.\overline{X_0}) + (X_6.X_4.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_3.X_5.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_5.X_4.X_3.\overline{X_2}.X_1.\overline{X_0})(X_4.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_3.X_2.X_1.\overline{X_0}) + (X_2.X_0)\end{aligned} \quad (2\text{-f})$$

$$\begin{aligned}A_0 &= (\overline{X_7}.X_6.\overline{X_5}.\overline{X_4}.X_2.X_1.\overline{X_0}) + (\overline{X_7}.X_5.\overline{X_4}.X_2.X_1.\overline{X_0})\\ &+(\overline{X_7}.X_4.X_2.X_1.\overline{X_0}) + (\overline{X_6}.\overline{X_5}.\overline{X_4}.X_2.X_1.\overline{X_0})\\ &+(X_7.X_6.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_5.X_4.X_3.X_2.X_1.X_0) + (X_4.X_3.X_2.X_1.X_0)\\ &+(X_3.X_2.\overline{X_1}.\overline{X_0}) + (X_1.X_0)\end{aligned} \quad (2\text{-g})$$

It chooses one output address value depending upon the output obtained from the encoder. The value obtained from the memory array is an odd multiple value of the co-efficient. In order to obtain the even multiple values, the shift operation is to be performed. The control unit generates the control bits according to the relations 3a to 3c.

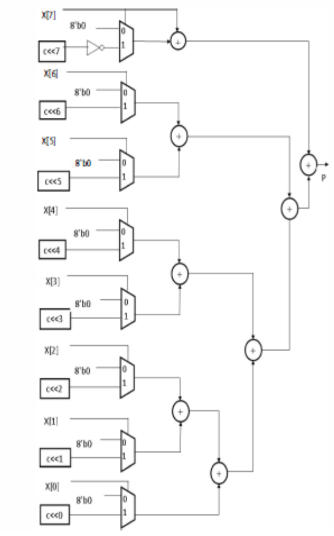


Figure-4-Proposed MUX based Multiplier architecture

$$\begin{aligned}C_2 &= (X_7.\overline{X_6}.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_4.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\end{aligned} \quad (3\text{-a})$$

$$\begin{aligned}C_1 &= (X_7.\overline{X_6}.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_4.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0})\\ &+(X_3.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_2.\overline{X_1}.\overline{X_0})\end{aligned} \quad (3\text{-b})$$

$$\begin{aligned}C_0 &= (\overline{X_7}.\overline{X_6}.X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.X_1.\overline{X_0}) + (\overline{X_7}.\overline{X_6}.X_4.\overline{X_3}.\overline{X_2}.X_1.\overline{X_0})\\ &+(\overline{X_7}.\overline{X_6}.X_3.\overline{X_2}.X_1.\overline{X_0}) + (\overline{X_7}.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.X_1.\overline{X_0})\\ &+(X_7.\overline{X_6}.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.X_1.\overline{X_0}) + (X_7.\overline{X_6}.X_5.\overline{X_4}.\overline{X_3}.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_7.\overline{X_6}.X_4.\overline{X_3}.X_2.\overline{X_1}.\overline{X_0}) + (X_7.\overline{X_6}.X_3.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_6.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_5.\overline{X_5}.\overline{X_4}.\overline{X_3}.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_5.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.X_5.\overline{X_4}.\overline{X_3}.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_6.X_4.\overline{X_3}.X_2.\overline{X_1}.\overline{X_0}) + (X_3.\overline{X_2}.\overline{X_1}.\overline{X_0}) + (X_6.X_3.X_2.\overline{X_1}.\overline{X_0})\\ &+(X_2.\overline{X_1}.\overline{X_0})\end{aligned} \quad (3\text{-c})$$

Input, X	P = X.C	X in powers of 2
0001	1C	2^0
0010	2C	2^1
0011	3C	2^1+2^0
0100	4C	2^2
0101	5C	2^2+2^0
0110	6C	2^2+2^1
0111	7C	$2^2+2^1+2^0$
1000	-8C	-2^3
1001	-7C	-2^3+2^0
1010	-6C	-2^3+2^1
1011	-5C	$-2^3+2^1+2^0$
1100	-4C	-2^3+2^2
1101	-3C	$-2^3+2^2+2^0$
1110	-2C	$-2^3+2^2+2^1$
1111	-1C	$-2^3+2^2+2^1+2^0$

Table- Mux Based Multiplier

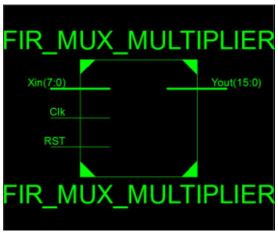
When the input word is 00000000 then the Reset (Rst) operation is performed by setting the reset bit which is given by the logical relation

$$Rst = \overline{X_7}.\overline{X_6}.\overline{X_5}.\overline{X_4}.\overline{X_3}.\overline{X_2}.\overline{X_1}.\overline{X_0}$$

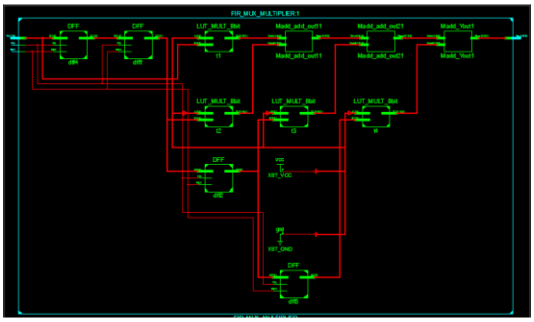
B. Proposed MUX based multiplier Figure 3 shows a shift and add multiplier implemented
with the usage of namely adders, and MUX. Since the general multipliers are replaced by MUX, shifters and adders, referred to as multiplierless implementation. Consider an eight bit multiplication of signed numbers. Here the multiplication is carried out only with MUX and shifters. For example consider two signed 8-bit numbers x and c bits, represented in two's complement form.

IV. PROPOSED SYSTEMS

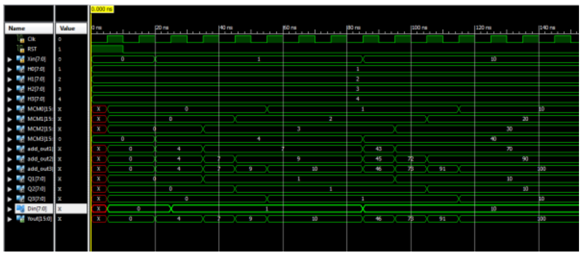
The proposed system was LUT based multiplierless FIR Filter which allows to reduce the maximum range of area so that he complexity will get reduced and increments the performance and computing will be very high. The following below are the results corresponding to proposed method.



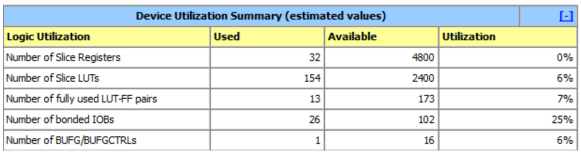
a)



b)

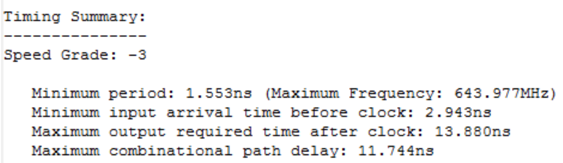


c)



Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	32	4800	0%
Number of Slice LUTs	154	2400	6%
Number of fully used LUT-FF pairs	13	173	7%
Number of bonded IOBs	26	102	25%
Number of BUFG/BUFGCTRLs	1	16	6%

d)



```
Timing Summary:
---------------
Speed Grade: -3

   Minimum period: 1.553ns (Maximum Frequency: 643.977MHz)
   Minimum input arrival time before clock: 2.943ns
   Maximum output required time after clock: 13.880ns
   Maximum combinational path delay: 11.744ns
```

e)

Figure:5-a)RTL Block, b)RTL Logic Diagram, c)Waveform, d) Area Report,
e)Delay Report

CONCLUSION

In this paper an efficient Low complexity based FIR filter architecture using MUX based multiplier and LUT based multiple scheme have been discussed and implemented effectively. The results of FIR filter architectures are analyzed and compared with respect to area and power. It is found that LUT multiple scheme occupies less area compared to MUX based FIR Filter architecture. Thus the proposed FIR filter architecture achieves low area and more flexibility and hence it is well suitable for VLSI implementation.

REFERENCES

[1) Digital Signal Processing solution: "Designing for Optimal Results High-Performance DSP using Virtex-FPGAs," Xilinx corporation, pp. 99-103, 2005

[2) David V. Anderson, and ErhanOzalevli, "A Reconfigurable MixedSignal VLSI implementation of Distributed Arithmetic Used for Finite Impulse Response Filtering," IEEE Transactions on Circuits and Systems-I: Regular Papers, Vol. 55, No. 2, March 2008.

[3) Kuan-hung, and Tzi-Dar, "A low power digit based Reconfigurable FIR Filter," IEEE Transactions on circuits and systems, Vol. 53, Aug 2006.

[4) Pramod Kumar Meher, "New Approach to Look Up Table Design and Memory-Based Real ization of FIR Digital Filter," IEEE Transactions on circuits and systems irregular papers, Vol. 57, No. 3, March 2010.

[5] R.Mahesh, and AP Vinod, "New Reconfigurable Architectures for implementing FIR Filter with low complexity," IEEE Transactions on computer aided design of integrated circuits and systems, Vol. 29, Feb 2010.

[6] T. Solla, and O. Vainio, "Comparison of program

mable FIR filter architectures for low power," in Proc. of 28th European Solid State Circuits Conference, pp. 759-762, September 24 - 26, 2002

[7] Pramod Kumar Meher, "FPGA Realization of FIR Filters by Efficient and Flexible Systolization Using Distributed Arithmetic", IEEE Transactions on signal processing, Vol. 56, No. 7, July 2008.

[8] A Croisier, D. J. Esteban, M. E. Levilion, and V. Rizo,"Digital filter for PCM encoded signals," U.S. Patent 3 777 130, Dec. 4, 1973.

[9] H. Yoo and D. V. Anderson, "Hardware-efficient distributed arithmetic architecture for High-order digital filters," in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP), Mar. 2005, vol. 5, pp. v/125-v/128.

[10] Xilinx Incorporation, "The Role of Distributed Arithmetic in FPGAbased signal Processing," Xilinx application notes, San Jose, CA

[I I] Young-Ho Seo and Dong-Wook Kim, "A New VLSI Architecture of parallel multiplier -Accumlator Based on Radix-2 Modified Booth Algorithm," IEEE Transactions on VLSI Systems, Vo1.I8, No.2, Feb 2010.

[12] Chip-Hong Chang, "Radix-8 Booth Encoded Modulo Multipliers with Adaptive Delay for High Dynamic Range Residue Number System, "IEEE Transactions on Circuits and Systems-I: Regular Papers, 2010

[13] Liu Ming, Yan Chao," The Multiplexed Structure of Multi-channel FIR Filter and its Resources Evaluation," International Conference on Computer Distributed Control and Intelligent Enviromental Monitoring, IEEE 2012.

[14] Asgar Abbaszadeh and khostov D.sadeghipour"A New Hardware Efficient Reconfigurable FIR Filter architecture suitable for FPGA appl ications"proc IEEE DS P 2011

[15] 1. Park, et aI., "Computation Sharing Programmable FIR Filter for LowPower and High-Performance Applications", IEEE J. Solid stateCir. Sys., vo1.39, no.2, pp.348-357, Feb. 2004

[16] Y. H. Chan and W. C. Siu,"On the realization of discrete cosine transform using the distributed arithmetic," IEEE Trans. Circuits Syst. I,Fundam. Theory Appl., vol. 39, no.9, pp. 705-712, Sep. 1992.

[17] H.-c. Chen, 1.-1. Guo, T.-S. Chang, and C.-W. Jen, "A memory-efficient realization of cyclic convolution and its application to discrete cosine transform," IEEE Trans. Circuits Syst. Video Technol., vol. 15,no. 3, pp. 445-453, Mar. 2005.