I J C T A, 9(13) 2016, pp. 5937-5947

Design of Reconfigurable and Energy Efficient FIR Filter Architecture for High Speed DSP Applications

J. Shanmukha Srinivas[1] and U. Hari[2]

ABSTRACT

The two prerequisites of Finite Impulse Response (FIR) filters are re-configurability and low complexity nature. These two are utilized as a part of multi standard wireless communication system. The proposed reconfigurable and low complex FIRs, to be specific constant shifts method and programmable shifts method. This design has the ability to work over variable word length coefficient of the filter .This is not having any hardware overhead. The design actualize reconfigurable filters utilizing common sub-expression elimination program method effectively, it is simple to demonstrate this will offer improvised area, enhanced speed and power diminishment than the conventional filter. The parallel FIR filters are implemented efficiently using these strength reduction technique. Algorithmic strength reduction transformations are for the most part utilized as a part of configuration of different DSP algorithms. This transformation can lead to reduction in silicon area or power consumption in a VLSI implementation and iteration period in a programmable DSP implementation. Another class of algorithms, termed fast FIR algorithms (FFA's), depends upon this approach to produce reduced complexity parallel filtering structure.In this design all the techniques with area, power and delay results are compared using Xilinx 12.1 ISE design suite software tool. Synthesis and simulations are done using SPARTAN 3E –fg320 package board with speed -5.

Keywords: Digital signal processing (DSP), fast finite impulse response filter (FIR) algorithms (FFAs), Very large scale integration (VLSI).

I. INTRODUCTION

The complexity of linear-phase finite-impulse-response (FIR) filters is dominated by the complexity of coefficient multipliers. The number of adders (subtractors) used to implement the multipliers determines the complexity of the FIR filters. It is well known that common sub expression elimination (CSE) methods based on canonical signed digit (CSD) coefficients reduce the number of adders required in the multipliers of FIR filters. A new CSE algorithm using binary

Representation of coefficients for the implementation of higher order FIR filters with a fewer number of adders than CSD-based CSE methods is presented in this paper. We show that the CSE method is more efficient in reducing the number of adders needed to realize the multipliers when the filter coefficients are represented in the binary form. Our observation is that the number of unpaired bits (bits that do not form CSs) is considerably few for binary coefficients compared to CSD coefficients, particularly for higher order FIR filters. As a result, the proposed binary-coefficient-based CSE method offers good reduction in the number of adders in realizing higher order filters. The reduction of adders is achieved without much increase in critical path length of filter coefficient multipliers. Design examples of FIR filters show that our method offers an average adder reduction of 18% over the best known CSE method, without any increase in the logic depth. The common subexpression elimination (CSE) techniques address the issue of minimizing

M.Tech, VLSI Design[1], Department of ECE, SRM University, Chennai, Tamilnadu, India.
Asst. Professor[2], Department of ECE, SRM University, Chennai, Tamilnadu, India.
E-mail: Shan.sj9@gmail.com[1]; hari.u@ktr.srmuniv.ac.in[2]

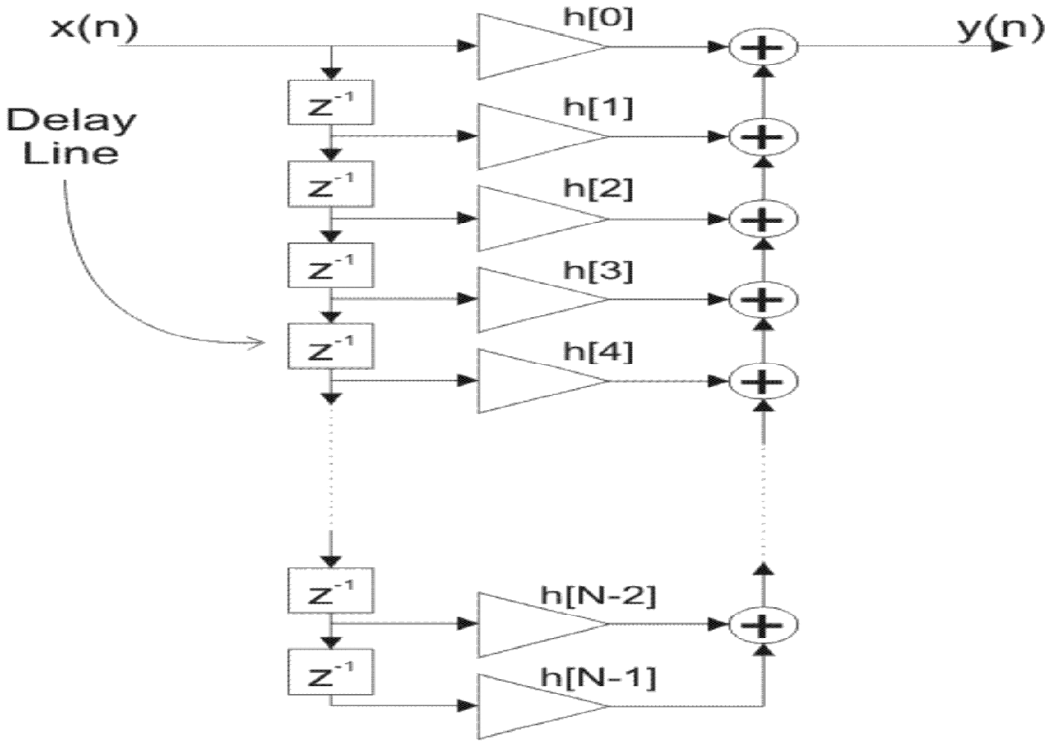

Figure 1: Diagram for Convolutional FIR

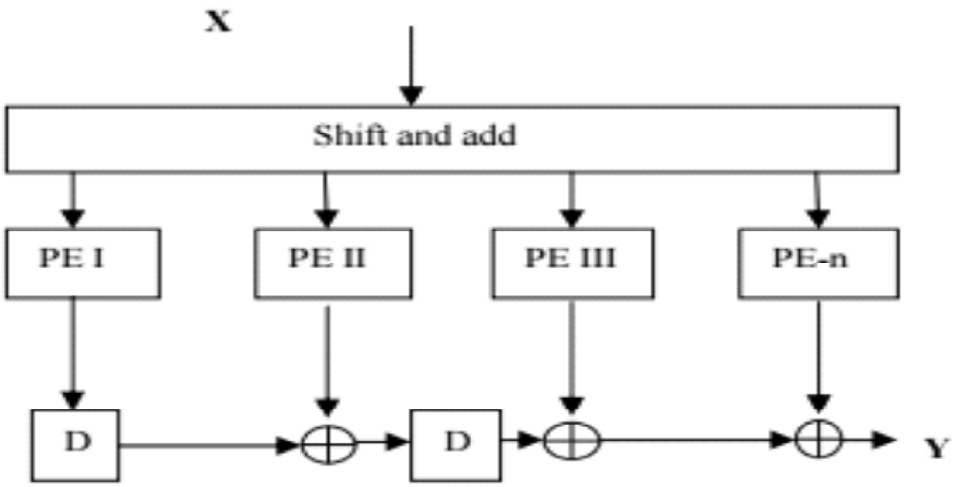

Figure 2: Diagram for proposed FIR using CSM or PSM Method

the number of adders needed to implement the multiple constant multiplication (MCM) blocks. In this paper, we provide a comparison of hardware reductions achieved using the horizontal, vertical, oblique and combining horizontal and vertical CSEs in realizing constant multipliers. The design of multiplier less implementations (which use only adders, subtractors and binary shifts) of fixed-point matrix multipliers is considered and a new common subexpression elimination method is described that recursively extracts signed two-term common subexpressions. Examples are given that show that the resulting adder-cost is significantly lower than for existing algorithms. The complexity of linear phase finite impulse response (FIR) filters used in the channelizer of a software defined radio (SDR) receiver is dominated by the complexity of coefficient multipliers. It is well known that common subexpression elimination (CSE) methods based on canonical signed digit (CSD) coefficients produce low complexity FIR filter coefficient multipliers. A new CSE algorithm based on the binary representation of filter coefficients is presented in the paper. Design

examples of channel filters employed in the digital advanced mobile phone systems (D-AMPS) and personal digital cellular (PDC) receivers show that the proposed method offers an average adder reduction of 23% over the conventional CSD-based CSE method. The complexity of finite impulse response (FIR) filters is dominated by the number of adders (subtractors) used to implement the coefficient multipliers. It is well known that common sub expression elimination (CSE) method based on canonic signed digit (CSD) representation considerably reduces the number of adders in coefficient multipliers. Recently, a binary based CSE (BSE) technique was proposed, which produced better reduction of adders compared to the CSD based CSE. In this paper, we propose a new 4-bit Binary based CSE (BCSE) method which employs 4-bit common subexpressions (CSs). Design examples show an average adder reduction of 31.2 % over the conventional CSD based CSE and 15% reduction over BSE.It is shown that the use of a canonical signed digit (CSD) representation of the filter coefficients can significantly reduce the complexity of the hardware implementation of digital FIR filters. This paper presents an example filter design that shows the error involved in limiting the number of allowable non-zero CSD coefficients for a real FIR band pass filter. If not done carefully, brute force limiting can lead to large errors in the frequency response. The error is evaluated for varying numbers of non-zero CSD coefficients. Lastly, a system level architecture with a multiplier utilizing the properties of the CSD number representation system is proposed.

II. OVERVIEW OF DIFFERENT FIR STRUCTURES

In this section, a brief explanation about the basic traditional and existing FIR filter using different techniques. The implementation of each structure is described.

(A) Convolutional FIR

An n-tap FIR filter can be expressed in the general form as (1),

$$y(n) = \sum_{i=0}^{N-1} h(i)x(n-i), n = 0,1,2,\dots\infty \tag{1}$$

where x(n) is an infinite-length input sequence and h(i) are the coefficients of the length-N FIR filter. N multipliers and N-1 adders are needed to implement an N-tap FIR filter. As like which shows in figure 1.

(B) Constant shift methodfor FIR

In the CSM architecture, the coefficients are stored directly in the LUT. These coefficients are partitioned into groups of 3-bits and are used as the select signal for the multiplexers. The number of multiplexer units required. The CSM can be explained with the help of an 8- bit coefficient h = "0.11111111." This coefficient h is the worst-case 8- bit coefficient since all the bits are non-zero and hence needs a maximum number of additions and shifts. In this case, n = 8 and therefore the number of multiplexers required is 3. The output $y = h * x$ is expressed as $y = 2^{-1}x+2^{-2}x+2^{-3}x+2^{-4}x+2^{-5}x+2^{-6}x+2^{-7}x+2^{-8}x$ (1) By partitioning into groups of three bits from most significant bit (MSB) (1), we obtain

$$h = 2\text{–}1\ (x + 2^{-1}x + 2^{-2}x + 2^{-3}x + 2^{-4}x + 2^{-5}x + 2^{-6}x + 2^{-7}x) \tag{2}$$

$$h = 2^{-1}\ (x + 2^{-1}x + 2^{-2}x + 2^{-3}\ (x + 2^{-1}x + 2^{-2}x) + 2^{-6}\ (x + 2^{-1}x)) \tag{3}$$

Note that the terms $x + 2^{-1}x + 2^{-2}x$ and $x + 2^{-1}x$ can be obtained from the shift and add unit. Then by using the three multiplexers (mux), two 8:1 mux for the first two 3-bit groups and one 4:1 mux for the last two bits of the filter coefficients, the intermediate sums shown inside the equation can be obtained. The final shifter unit will perform the shift operations 2^{-1}, 2^{-3}, and 2^{-6}. Since these shifts are always constant irrespective of the coefficients, programmable shifters are not required and these shifts can be hardwired. The final adder unit will compute the sum of all the intermediate sums to obtain h–x[n].

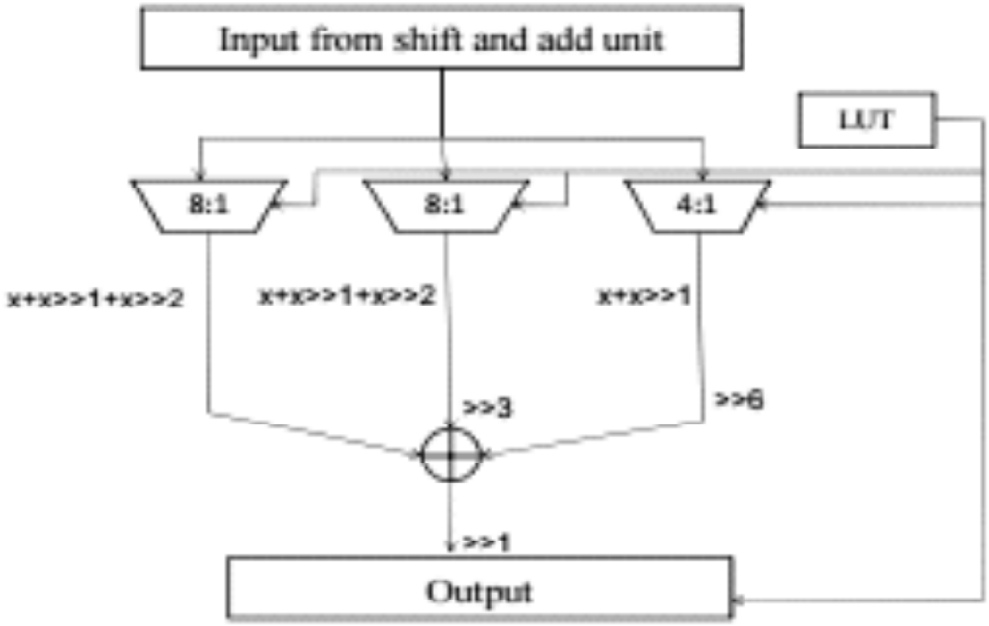

Figure 3: CSM Architecture (x(n)*h(n))

ADVANTAGE OF CSM

The CSM architecture results in high speed filters.

(C) Programmable shift method

The PSM has a pre-analysis part in which the filter coefficients are analyzed using the BCSE algorithm. Thus, the redundant computations (additions) are eliminated using the BCSs and the resulting coefficients in a coded format are stored in the LUT. The shift and add unit is identical for both PSM and CSM. The number of multiplexer units required can be obtained from the filter coefficients after the application of BCSE . The number of multiplexers is selected after considering the number of non-zero operands (BCSs and unpaired bits) in each of the coefficients after the application of the BCSE algorithm. The number of multiplexers will be corresponding to the number of non-zero operands for the worst-case coefficient (worst-case coefficient being defined as coefficient that has the maximum number of non-zero operands).

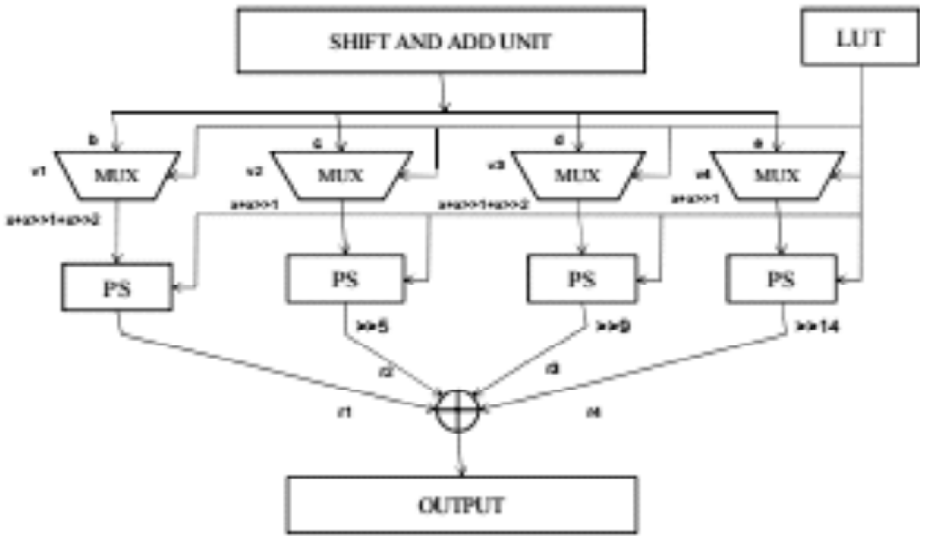

Figure 4: PSM Architecture (x(n)*h(n))

The LUT consists of two rows of 18 bits for each coefficient of the form SDDDDXXDDDDXXMMMML and DDDDXXDDDDXXDDDDXX, where "S" represents the sign bit, "DDDD" represents the shift values from 20 to 2"15 and "XX" represents the input "x" or the BCSs obtained from the shift and add unit. In the coded format, XX = "01" represents "x," "10" represents x+2–1 x, "11" represents x + 2–2 x, and "00" represents x + 2–1 x + 2–2 x, respectively. Thus, the two rows can store up to five operands which is the worst case number of operands for a 16-bit coefficient. In most of the practical coefficients, the number of operands is less than the worst case number of operands, 5. In that case "MMMML" can be used to avoid unnecessary additions.

(D) Fast FIR Algorithm (FFA)

Consider an N-tap FIR filter which can be expressed in the general form as:

$$y(n) = h(n) * \mathrm{x(n)} \sum_{i=0}^{N-1} h(i)x(n-i)\,, n = 0,1,2 \ldots \infty \tag{4}$$

where {x(n)} is an infinite length input sequence and {h(i)} are the length-N FIR filter coefficients. An N-tap FIR filter can be expressed in z-domain as:

$$Y(z) = H(z)X(z) = \sum_{n=0}^{N-1} h(n)z^{-n} \sum_{n=0}^{N-1} x(n)z^{-n} \tag{5}$$

Then, the traditional L-parallel FIR filter can be derived using polyphase decomposition as:

$$\sum_{p=0}^{L-1} Y_p(Z^L)Z^{-p} = \sum_{q=0}^{L-1} X_i\ (Z^q)Z^{-q} \sum_{r=0}^{L-1} H_r(Z^r)Z^{-r} \tag{6}$$

where

$$X_q = \sum_{k=0}^{\infty} z^{-k}x(LK+q), \quad H_r = \sum_{k=0}^{\left(\frac{N}{L}\right)-1} z^{-k}x(LK+r)$$

and

$$Y_{p=} \sum_{k=0}^{\infty} z^{-k}x(LK+p)$$

For *p,q,r = 0,1,2,3……..L-*

1. This *L* - parallel FIR filter can be obtained by decomposing *X(z)*, *H(z)*, and *Y(z)* into *L* subsequences with L^2-subfilter blocks.

(E) FFA

In this approach, we generally design the FIR filter architecture without considering the symmetric coefficients are described briefly in below.

a) FFA 2x2 Filter

According to (3), by placing *L=2* we obtain a two-parallel FIR filter which can be expressed as:

$$\begin{aligned} Y(z) &= Y_o(z^2) + z^{-1}Y_1(Z^2) \\ &= (X_o(z^2) + z^{-1}X_1(Z^2)\ (H_o(z^2) + z^{-1}H_1(z^2)) \end{aligned} \tag{7}$$

i.e.,

$$\begin{aligned} Y_o &= H_oX_o + Z^{-2}H_1X_1 \\ Y_1 &= H_0X_1 + H_1X_1 \end{aligned} \tag{8}$$

As we can observe from this traditional *L*-parallel filter implementation it requires - FIR sub filter blocks each of length N/L and requires a total of $L^2 .N/L$ multiply add operations. We need to derive fast FIR algorithms to reduce the complexity for parallel FIR filters. The example of 2-parallel algorithm is shown in Fig. 1.

The above expressions (7) can be rewritten as follows to reduce the computational complexity in equation (8) shown in Fig. 6:

$$Y_1 = (H_0 + H_1)(X_0 + X_1) - H_0X_0 - H_1X_1$$
$$Y_0 = H_0X_0 + z^{-2} H_1X_1 \quad (9)$$

This low complexity implementation requires three FIR subfilter blocks of length *N/2*, one preprocessing adders, three postprocessing adders, and *3N/2=1.5N* multipliers and *3(N/2-1) +4 = 1.5N+1* adders, which reduces approximately one fourth over traditional two parallel filter hardware cost.

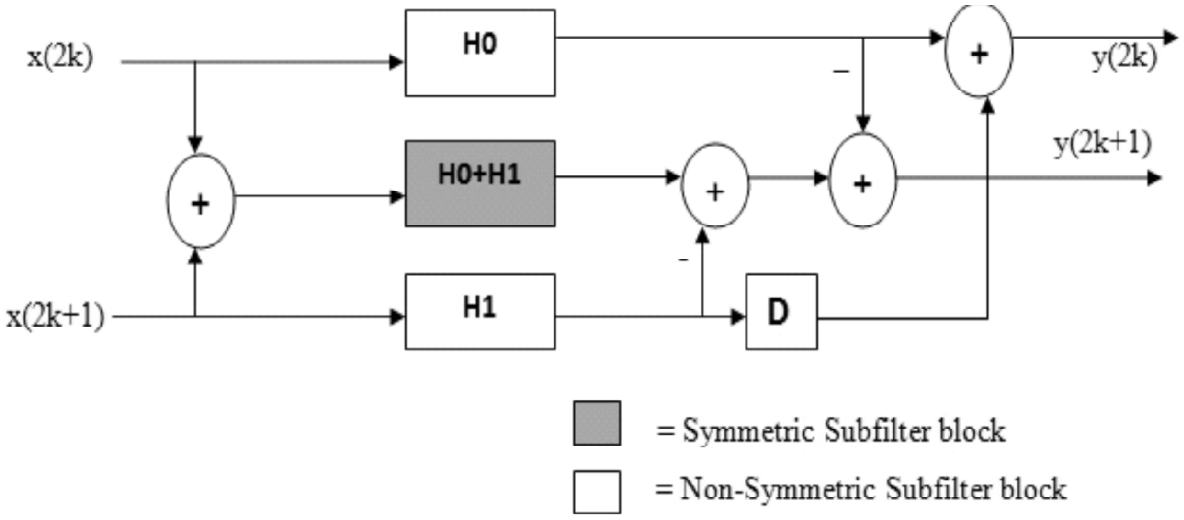

Figure 5: Architecture for FFA 2X2 FIR Filter

III. IMPLEMENTATION

(A) Overall flow of Implemenation of CSM or PSM

It is well known that one of the efficient ways to reduce the complexity of multiplication operation is to realize it using shift and add operations. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Fig. 3, "x>>k" represents the input x shifted right by k units.

All the 3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bit number are generated using only three adders, whereas a conventional shift and add unit would require five adders. Since the shifts to obtain the BCSs are known beforehand, PS are not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit.In both the architectures (CSM and PSM) use the same shift and add unit. Thus, the use of 3-bit BCSs reduces the number of adders needed to implement the shift and add unit compared to conventional shift and add units.

The multiplexer units are used to select the appropriate output from the shift and add unit. All the multiplexers will share the outputs of the shift and add unit. Today every circuit has to face the power consumption issue for both portable device aiming at large battery life and high end circuits avoiding cooling packages and reliability issues that are too complex. It is generally accepted that during logic synthesis power tracks well with area. This means that a larger design will generally consume more power.

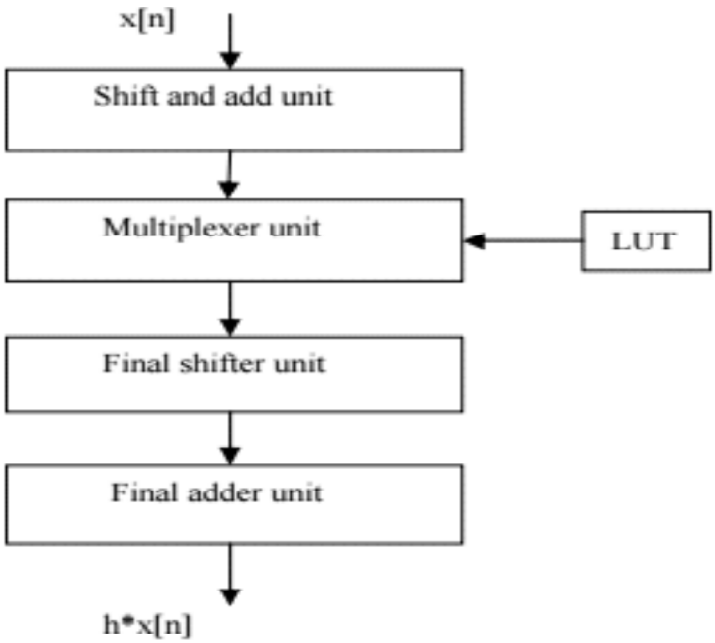

Figure 6: Algorithm for CSM and PSM Implementaion

The multiplier is an important kernel of digital signal processors. Because of the circuit complexity, the power consumption and area are the two important design considerations of the multiplier.For getting the low power low area architecture, the modifications made to the conventional architecture consist of the reduction in switching activities of the major blocks of the multiplier, which includes the reduction in switching activity of the adder and counter. The final shifter unit will perform the shifting operation after all the intermediate additions (i.e., intra-coefficient additions) are done. This unit will compute the sum of all the intermediate additions.

(B) Overall flow of Implemenation of FFA

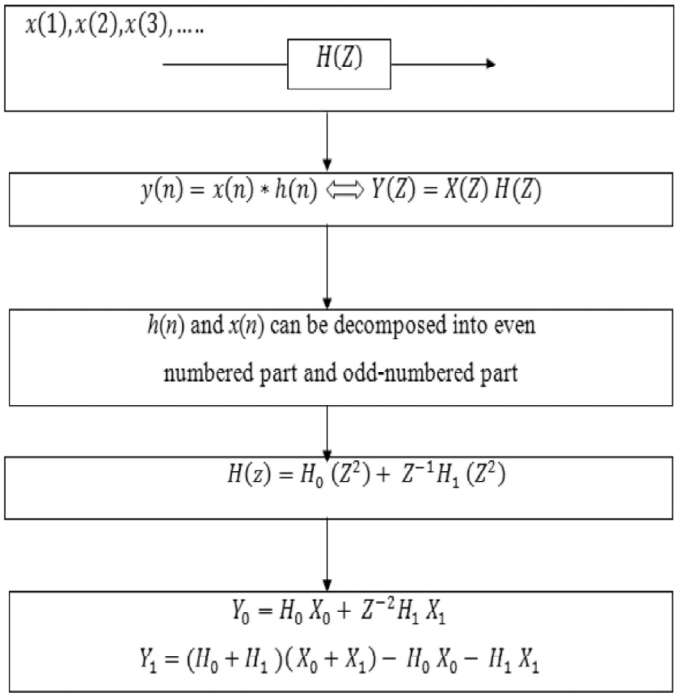

Figure 7: Algorithm for FFA 2x2 FIR Filter

IV. SIMULATION AND SYNTHESIS RESULT

(A) Simulation Results

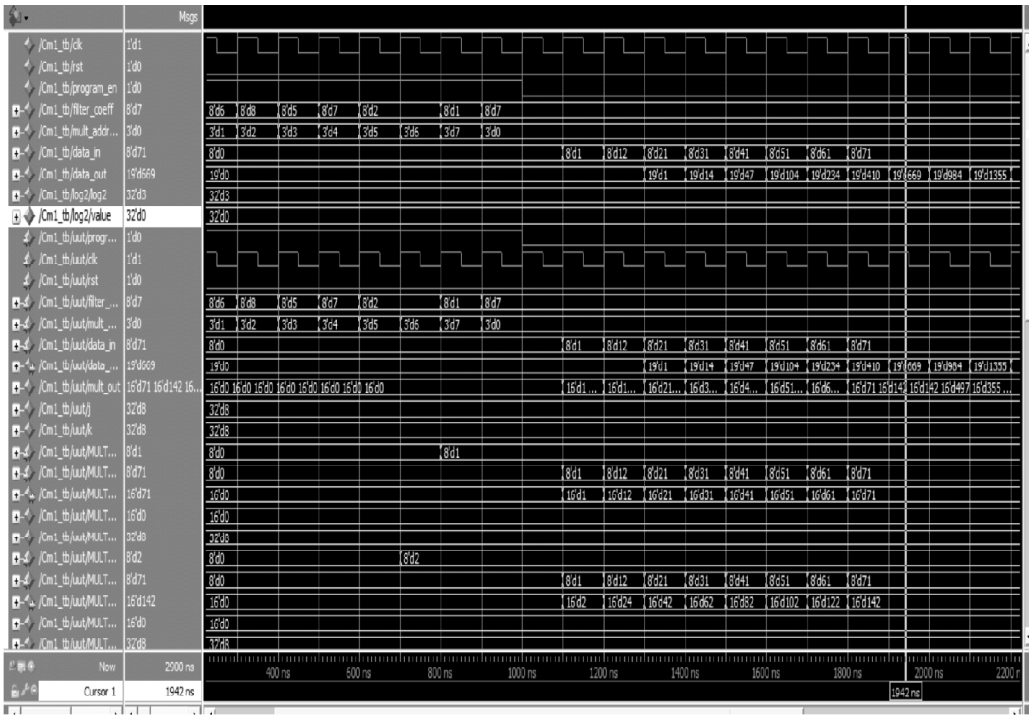

Figure 8: Simulation result of conventional FIR Filter

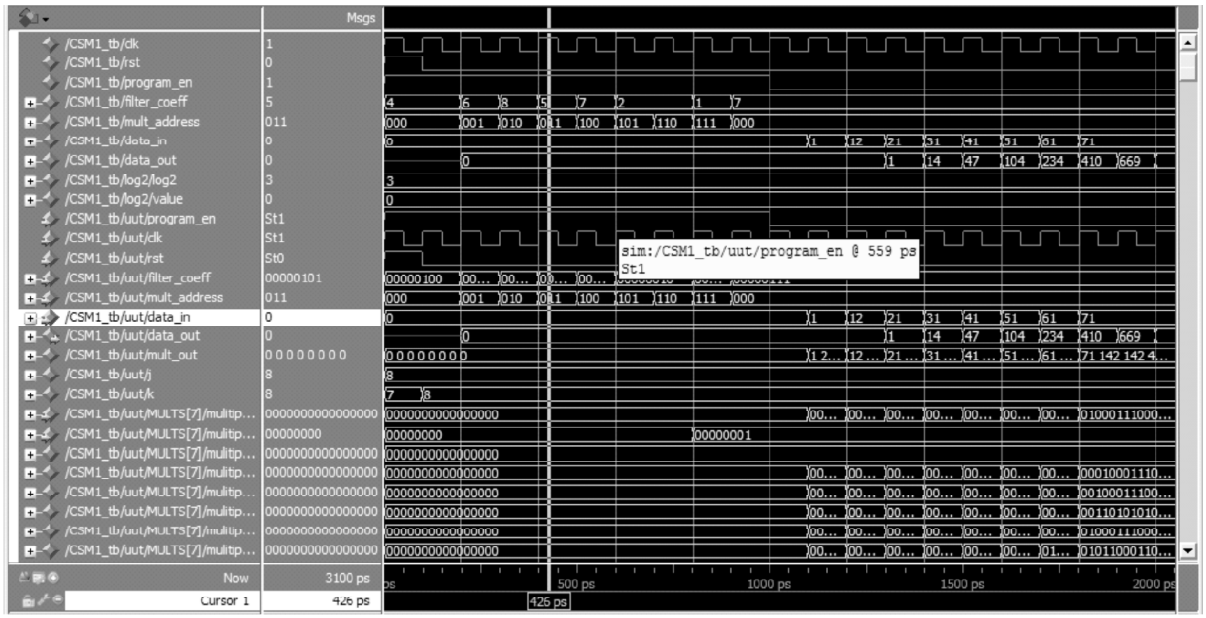

Figure 9: Simulation result of CSM based FIR Filter

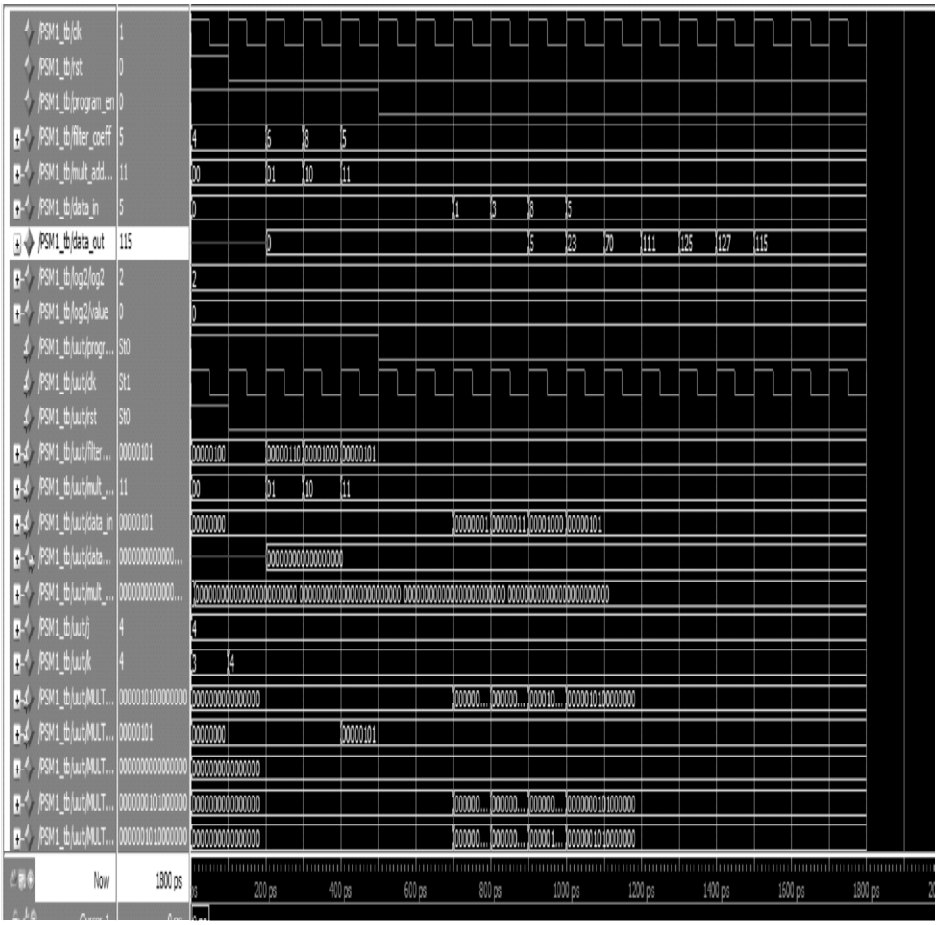

Figure 10: Simualtion result of PSM based FIR Filter

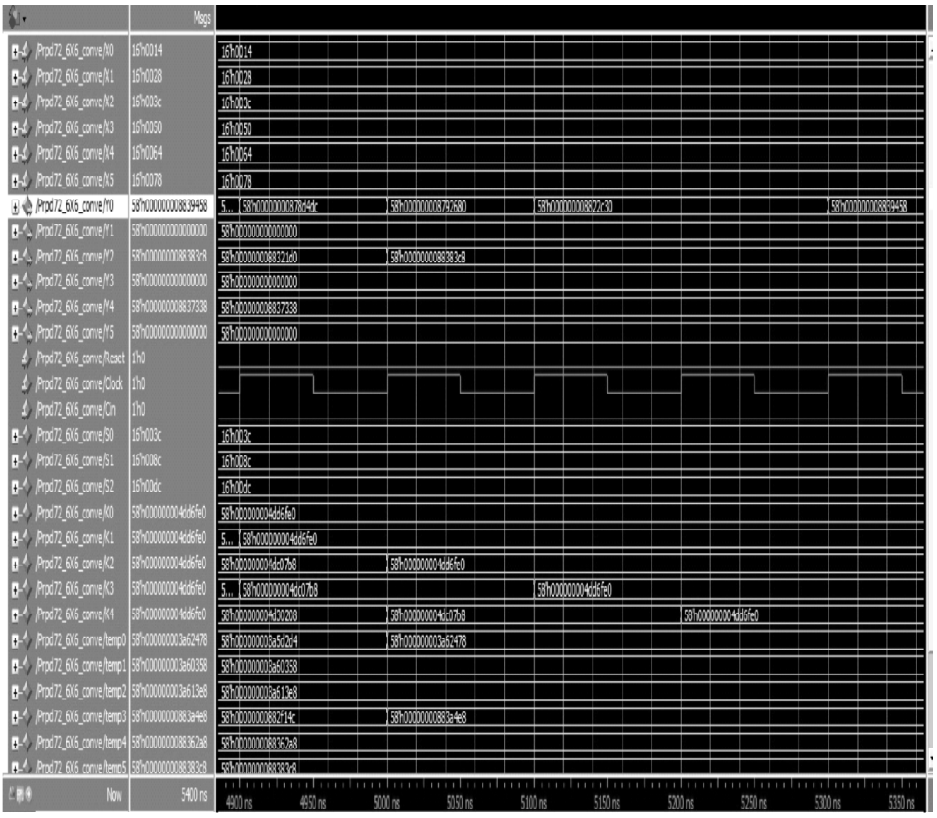

Figure 11: Simualtion result of 2X2 FFA FIR Filter

(B) Synthesis Results

Table 1
Compared Area, Power & delay results

Conventional FIR	*8 TAP*	*16 TAP*	*24 TAP*	*48 TAP*	*72 TAP*
No. of Slices	653	1315	1984	3951	6003
No of slice Flip Flops	236	483	746	1630	2517
No.of 4 input LUTs	1234	2500	3754	7521	11366
No. of Bonded IOBS	41	43	45	47	49
Delay Report	37.767 ns	37.971 ns	38.981 ns	39.861 ns	40.228 ns
Power Report					
Total (mw):	159.40	164.44	164.46	164.51	164.76
Dynamic(mw):	0.95	5.87	5.89	5.95	6.19
static (mw) :	158.45	158.56	158.56	158.57	158.57
CSM FIR	8 TAP	16 TAP	24 TAP	48 TAP	72 TAP
No. of Slices	567	1127	1702	3400	5165
No of slice Flip Flops	237	467	719	1483	2293
No. of 4 input LUTs	1087	2175	3273	6581	9941
No. of Bonded IOBS	41	43	45	47	49
Delay Report	30.097 ns	30.726 ns	31.012 ns	31.362 ns	31.524 ns
Power Report					
Total (mw):	158.94	160.00	160.82	160.93	161.42
Dynamic(mw):	0.50	1.53	2.33	2.45	2.92
static (mw) :	158.44	158.46	158.48	158.49	158.50
PSM FIR	8TAP	16TAP	24TAP	48TAP	72TAP
No. of Slices	744	1499	2269	4552	6902
No. of Slice FlipFlops	281	567	868	1776	2731
No. of 4Input LUT	1418	2868	4326	8715	13157
No. of Bonded IOBs	41	43	45	47	49
Delay Report	29.973 ns	30.09 ns	30.214 ns	30.368 ns	31.199 ns
Power Report					
Total (mw):	158.98	158.99	159.01	159.06	159.34
Dynamic (mw):	0.54	0.55	0.57	0.62	0.89.
static (mw) :	158.44	158.44	158.44	158.44	158.45

V. CONCLUSION

Implementation of different techniques of FIR filter and comparing all the simulation results,synthesis, area, power and delay to each other and hence demonstrated that newly proposed designs are very effective with less hardware circuitry. It also consumes less delay and low power when compared to conventional FIR filter .In this design XILINX ISE 12.1 Design tool and Spartan 3E FPGA package f-320g are used to compare the results tabulated in Table1.

Future scope: In this design, as a future idea, a 2X2 parallel FIR filter for symmetric and non-symmetric coefficients can be built which can help to design higher NXN parallel FIR filter just by cascading 2x2, 3x3, 4x4 and so on.

References

[1] R. Mahesh, Member, IEEE, and A. P. Vinod, Senior Member, “New Reconfigurable Architectures for Implementing FIR Filters with Low Complexity” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, No. 2, February 2010.

[2] J. Mitola, ‘‘Object-oriented approaches to wireless systems engineering,’’in Software Radio Architecture. New York: Wiley, 2000.

[3] R. I. Hartley, ‘‘Subexpression sharing in filters using canonic signed digit multipliers,’’ IEEE Trans. Circuits Syst. II, vol. 43, no. 10, pp. 677-688,Oct. 1996. 5.

[4] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova,A new algorithm for elimination of common subexpressions,’’ IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 18, no. 1, pp. 58-68, Jan. 1999.

[5] M. M. Peiro, E. I. Boemo, and L. Wanhammar, ‘‘Design of highspeedmultiplierless filters using a nonrecursive signed common subexpression algorithm,’’ IEEE Trans. Circuits Syst. II, vol. 49, no. 3, pp. 196-203, Mar. 2002.

[6] R. Mahesh and A. P. Vinod, ‘‘A new common subexpression elimination algorithm for realizing low complexity higher order digital filters,’’ IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 2, pp. 217-219, Feb. 2008.

[7] A. P. Vinod and E. Lai, ‘‘Low power and high-speed implementation of FIR filters for software defined radio receivers,’’ IEEE Trans. Wireless Commun., vol. 5, no. 7, pp. 1669-1675, Jul. 2006.

[8] T. Solla and O. Vainio, ‘‘Comparison of programmable FIR filter Architectures for low power,’’ in Proc. 28th Eur. Solid-State Circuits Conf., Firenze, Italy, Sep. 2002, pp. 759-762.

[9] T. Solla, R. M¨akel¨a, M. Liljeroos, and O. Vainio, ‘‘Applicationspecific Filter processor for flexible receivers,’’ in Proc. 19th NORCHIP Conf., Kista, Sweden, Nov. 2001, pp. 53-58.

[10] D. Hwang, C. Mittelsteadt, and I. Verbauwhede, ‘‘Low power showdown:Comparison of five DSP platforms implementing an LPC speech codec,’’in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Salt Lake City,UT, May 2001, pp. 1125-1128.