Design of Efficient FIR filter with EDBNS multiplier using Transpose method for various Applications

[1] L. Durga Prasad, [2] J.E.N.Abhilash, [3] B.Subrahmaneswara Rao
[1] Student, M.Tech, [2] Associate Professor, [3] Professor
[1] VLSI System Design, Dept. of ECE,
[1] Swarnandhra College of Engineering &Technology, Narsapur, A.P, INDIA.

Abstract: **In FIR Filter Realization, Transpose form FIR filters are naturally pipelined and support multiple constant multiplication (MCM) technique that results in major saving of computation. By including block formulation method in transpose form data samples in fixed size blocks are processed one after the other which results in processing more number of samples in parallel. This project is mainly replacing the general multiplier present in existing transpose form fixed and reconfigurable filter architectures with EDBNS multiplier for better performance. Due to its scarcity and instinctive abstraction of the sum of binary shifted partial products, the sharing of adders in the time-multiplexed multiple constant multiplication blocks of the programmable FIR filters can be maximized by a direct mapping from the quasi-minimum EDBNS. The multiplexing cost can be further reduced by inclusion double base terms. The shifting operation delay has been reduced by using barrel shifters. The architectures have been simulated and synthesized on vertexE using VHDL which results in 4% reduction in PDP and 56.7% reduction in ADP.**

***IndexTerms*—Transpose form, Block formulation, Reconfigurable, Multiple Constant multiplication, extended double base number system, power delay product (PDP), area delay product (ADP).**

I. INTRODUCTION

The interest for low power digital signal processing (DSP) systems has expanded because of unstable development in versatile processing and convenient mixed media applications. A standout amongst the most broadly utilized operations performed in DSP is finite impulse response (FIR) filtering. Finite impulse response (FIR) digital filter have extensive variety of digital signal processing applications, for example, speech processing, loud speaker equalization, echo cancellation and communication applications including Software Defined Radio (SDR).

These applications require FIR filters of huge request to meet frequency specifications and need to help high rate of sampling for high speed digital communication. As FIR filter order increases the number of multiplication and addition also increases linearly for obtaining filter output. Since there is no repetitive computation accessible in FIR filter algorithm. Effective acknowledgment of FIR filter can be performed utilizing distributed arithmetic (DA) and multiple constant multiplication (MCM) techniques. DA based designs are lookup tables (LUTs) to store precomputed results to lessen the computational intricacy. The MCM strategy lessen the number of additions required for the acknowledgment of multiplication by regular sub-expression sharing when a given input is multiplied with an arrangement of filter coefficients.

The MCM method is more viable, when a typical operand is multiplied with more number of constants and it is appropriate for execution of huge order FIR filter with fixed coefficients. In any case, MCM blocks can be framed just in FIR filter in transpose form design. Throughput of hardware structure can be improved by block processing method. There for it helps for development in area delay parameters. In direct form FIR filter design block processing is simple and straight forward, while the transpose form design does not directly support block formulation. FIR filter in transpose form design can be acknowledged with the assistance of MCM technique to support block processing method.

The models are more appropriate for lower order filters and not proper for channel filters because of their huge area intricacy. Constant shift method (CSM) and programmable shift technique are utilized for RFIR filters, particularly for SDR channelizer. The FIR filter structure with multiplier utilizes either direct form design or transposes form design. Be that as it may, the multiplier less structures utilize transpose form design, though the DA based structures utilizes direct form design. It comprises of two strategies constant shift method (CSM) and programmable shift method (PSM). CSM helps for rapid operation and PSM helps for low area and low power reconfigurable FIR filter. Efficient digital reconfigurable finite impulse response filter architecture is realized by Md. Zameeruddin and has proposed productive technique for lookup table outline in memory based FIR filter. In lookup table based approach the memory elements store all the feasible estimations of product of filter coefficients. It comprises of four-three bit address encoder, three-eight line address decoder, memory array of eight words, NOR cell, control circuit and barrel shifter. LUT based multiplication used to decrease memory size. It has less area, low dormancy and high throughput as well.

There are a few applications, for example, SDR channelizer. A few designs have been proposed amid the most recent decade for efficient implementation of reconfigurable FIR (RFIR) utilizing general multipliers and constant multiplication schemes .However, the reconfiguration overhead is significantly expansive and does not give an area- delay efficient structure. The structures are more suitable for lower order filters and not appropriate for channel filters because of their expansive area complexity Constant shift method (CSM) and programmable shift method have been proposed for RFIR filters, specifically for

SDR channelizer. As of late, Park and Meher [10] have proposed a fascinating DA-based design for RFIR filters. The current multiplier-based structures utilize either direct form configuration or transpose form configuration. Be that as it may, we don't find any specific block-based design for RFIR filter in the writing. A block-based RFIR structure can without much of a stretch be determined utilizing the plan proposed below. Be that as it may, we find that the block structure acquired isn't efficient for substantial filter lengths and variable filter coefficients, for example, SDR channelizer. As a result, the design techniques proposed are more appropriate for 2-D FIR filters and block least mean square versatile filters. In this paper, we investigate the likelihood of implementation of block FIR filter in transpose form configuration keeping in mind the end goal to exploit the MCM plans and the natural pipelining for area-delay efficient realization of extensive order FIR filters for both fixed and reconfigurable applications.

II. REALIZATION OF TRANSPOSE FORM FIR FILTER

The data-flow graphs (DFG) of transpose form FIR filter for filter length N= 6 as shows below.

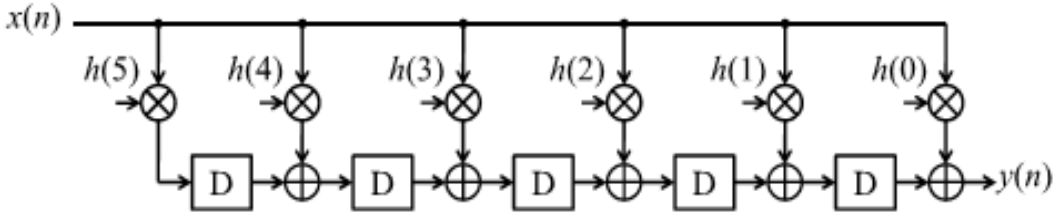


Fig.1. DFG 1 of transpose form structure for N =6 for output y (n).

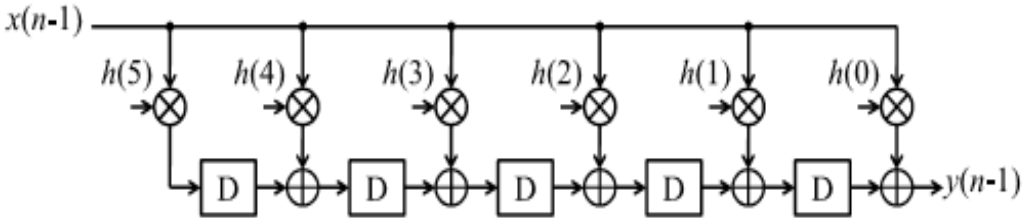


Fig.2. DFG 2 of transpose form structure for N =6 for output y (n-1).

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

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

-

The data flow graphs (DFG-1 and DFG-2) of transpose form FIR filter for filter length N = 6, as shown in Fig.1, for a block of two successive outputs {y (n), y (n−1)} that are gotten from the above DFG's. The product values and their accumulation paths in DFG-1 and DFG-2 are shown in data- flow tables (DFT-1 and DFT-2) of Fig. 3 and 4. The arrows in DFT-1and DFT-2 of Fig. 3 and 4, represent the accumulation path of the products. We see that five values of each column of DFT-3 are same as those of DFT-4 (shown in Gray color in Fig.3 and 4). These unnecessary computations of DFG-1 and DFG-2 can be abstained from utilizing nonoverlapped sequence of input blocks, as shown in figure 5.

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	$x(n)h(5)$	$x(n)h(4)$	$x(n)h(3)$	$x(n)h(2)$	$x(n)h(1)$	$x(n)h(0)$

Fig.3.DFT-1 Corresponding to output y (n)

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	$x(n\text{-}1)h(5)$	$x(n\text{-}1)h(4)$	$x(n\text{-}1)h(3)$	$x(n\text{-}1)h(2)$	$x(n\text{-}1)h(1)$	$x(n\text{-}1)h(0)$

Fig. 4. DFT-2 Corresponding to output y (n − 1).

The arrows in DFT-1 and DFT-2 of Fig. 2 speak to the aggregation path of the products. Block-processing method is profoundly used to determine high-throughput in hardware structures yet in addition enhances the area-delay efficiency. The formation of block-based FIR structure is clear-cut in the direct-form FIR, though the transpose form configuration does not straightforwardly support block processing. However, to take the computational benefit of the MCM, FIR filter is required to be realized by transpose form structure. The nonoverlapping input blocks are gotten from the block processing. The block does not include repetitive calculation. It is simple to find that the sections in grey cells in underneath (Fig.5) table relate to the output y (n) and correspond to y(n−1).

ccs	M_1	M_2	M_3	M_4	M_5	M_6
1	$x(n\text{-}10)h(5)$	$x(n\text{-}10)h(4)$	$x(n\text{-}10)h(3)$	$x(n\text{-}10)h(2)$	$x(n\text{-}10)h(1)$	$x(n\text{-}10)h(0)$
2	$x(n\text{-}8)h(5)$	$x(n\text{-}8)h(4)$	$x(n\text{-}8)h(3)$	$x(n\text{-}8)h(2)$	$x(n\text{-}8)h(1)$	$x(n\text{-}8)h(0)$
3	$x(n\text{-}6)h(5)$	$x(n\text{-}6)h(4)$	$x(n\text{-}6)h(3)$	$x(n\text{-}6)h(2)$	$x(n\text{-}6)h(1)$	$x(n\text{-}6)h(0)$
4	$x(n\text{-}4)h(5)$	$x(n\text{-}4)h(4)$	$x(n\text{-}4)h(3)$	$x(n\text{-}4)h(2)$	$x(n\text{-}4)h(1)$	$x(n\text{-}4)h(0)$
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	$x(n)h(5)$	$x(n)h(4)$	$x(n)h(3)$	$x(n)h(2)$	$x(n)h(1)$	$x(n)h(0)$

ccs	M_1	M_2	M_3	M_4	M_5	M_6
1	$x(n\text{-}11)h(5)$	$x(n\text{-}11)h(4)$	$x(n\text{-}11)h(3)$	$x(n\text{-}11)h(2)$	$x(n\text{-}11)h(1)$	$x(n\text{-}11)h(0)$
2	$x(n\text{-}9)h(5)$	$x(n\text{-}9)h(4)$	$x(n\text{-}9)h(3)$	$x(n\text{-}9)h(2)$	$x(n\text{-}9)h(1)$	$x(n\text{-}9)h(0)$
3	$x(n\text{-}7)h(5)$	$x(n\text{-}7)h(4)$	$x(n\text{-}7)h(3)$	$x(n\text{-}7)h(2)$	$x(n\text{-}7)h(1)$	$x(n\text{-}7)h(0)$
4	$x(n\text{-}5)h(5)$	$x(n\text{-}5)h(4)$	$x(n\text{-}5)h(3)$	$x(n\text{-}5)h(2)$	$x(n\text{-}5)h(1)$	$x(n\text{-}5)h(0)$
5	$x(n\text{-}3)h(5)$	$x(n\text{-}3)h(4)$	$x(n\text{-}3)h(3)$	$x(n\text{-}3)h(2)$	$x(n\text{-}3)h(1)$	$x(n\text{-}3)h(0)$
6	$x(n\text{-}1)h(5)$	$x(n\text{-}1)h(4)$	$x(n\text{-}1)h(3)$	$x(n\text{-}1)h(2)$	$x(n\text{-}1)h(1)$	$x(n\text{-}1)h(0)$

Fig.5. DFT of DFG-1 and DFG-2 for three nonoverlapped input blocks [x(n), x(n−1)], [x(n−2), x(n−3)],and [x(n−4),x(n−5)].(a)DFT-3 for computation of output y(n). (b) DFT-4 for computation of output y (n − 1).

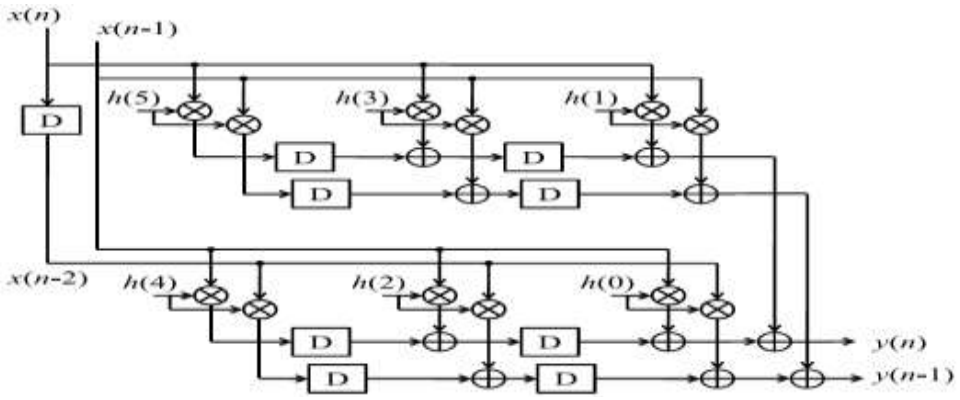


Fig.6.Merged DFG (DFG-3: transpose form type-I configuration for block FIR structure).

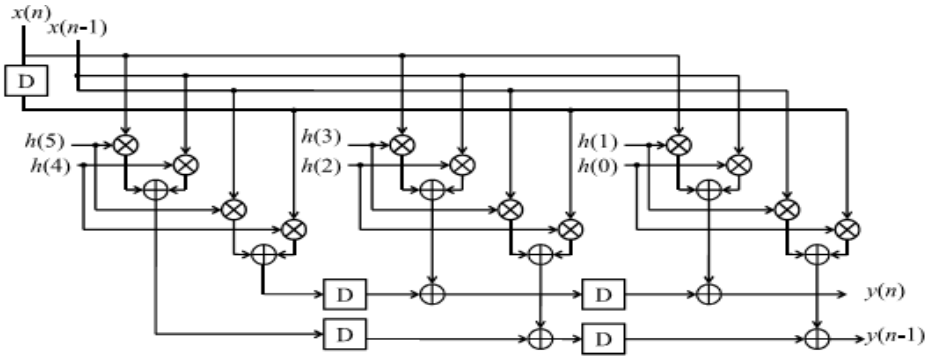


Fig.7. DFG-4 (retimed DFG-3) transpose form type-II configuration for block FIR structure.

III. COMPUTATIONAL ANALYSIS

The data-flow graphs (DFG-1 and DFG-2) of transpose form FIR filter for filter length N = 6, as appeared in Fig.1and fig.2, for a block of two progressive outputs {y (n), y (n − 1)}.The product esteems and their amassing paths in DFG-1 and DFG-2 are appeared in data- flow tables (DFT-1 and DFT-2). The arrows in DFT-1 and DFT-2 signify the accumulation path of the products.

We find that five estimations of every segment of DFT-1 are same as those of DFT-2 (appeared in gray color in Fig. 3 and 4).These excess computation of DFG-1 and DFG-2 can be abstained from utilizing nonoverlapped sequence of input blocks, as appeared in Fig. 7. DFT-3 and DFT-4 of DFG-1 and DFG-2 for non overlapping input blocks are correspondingly appeared in Fig. 5(a) and (b).

As shown in Fig. 5(a) and (b), DFT-3 and DFT-4 don't include repetitive calculation. It is simple to find that the entries in gray cells in DFT-3 and DFT-4 of Fig. 3(a) and (b) relate to the output y(n), though alternate entries of DFT-3 and DFT-4 relate to y(n−1).The DFG of Fig.1 should be changed properly to acquire the computations as indicated by DFT-3 and DFT-4. The calculation of DFT-3 and DFT-4 can be realized by DFG-3 of non overlapping blocks, as appeared in Fig.6. We allude it to block transpose form type-I configuration of block FIR filter. The DFG-3 can be retimed to acquire the DFG-4 of Fig. 7, which is alluded to block transpose form type-II configuration.

Note that both type-I and type-II configurations include a similar number of multipliers and adders, however type-II configuration includes almost L times fewer delay components than those of type-I configuration. We have, in this way, utilized block transpose form type-II configuration to determine the proposed structure. In Section II-C, we show mathematical formulation of block transpose form type-II FIR filter for generalized formulation of the model of block-based computation of transpose form FIR filters.

IV. PROPOSED STRUCTURE FOR TRANSPOSE FORM BLOCK FIR FILTER FOR RECONFIGURABLE APPLICATIONS

The proposed structure for block FIR filter is appeared in Fig. 8 for the block size L = 4. It comprises of one coefficient selection unit (CSU), one register unit (RU), M number of internal product units (IPUs), and one pipeline adder unit (PAU).

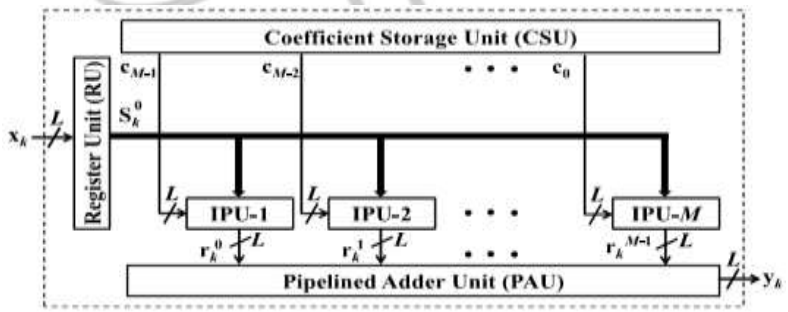


Fig. 8. Proposed structure for block FIR filter.

The CSU stores coefficients of all the filters to be utilized for the reconfigurable application. It is actualized utilizing N ROM LUTs, with the end goal that filter coefficients of a specific channel filter are acquired in one clock cycle, where N is the filter length. The RU [shown in Fig. 9(a)] gets x_k during the kth cycle and creates L columns of S_k^0 in parallel. L rows of S_k^0 are transmitted to M IPUs of the proposed structure. The M IPUs additionally get M short-weight vectors from the CSU to such an extent that amid the kth cycle, the (m + 1) th IPU gets the weight vector c_{M-m-1} from the CSU and L rows of S_k^0 form the RU. Each IPU performs matrix vector product of $S_k^0$with the short-weight vector cm, and computes a block of L partial filter outputs (r_k^m).

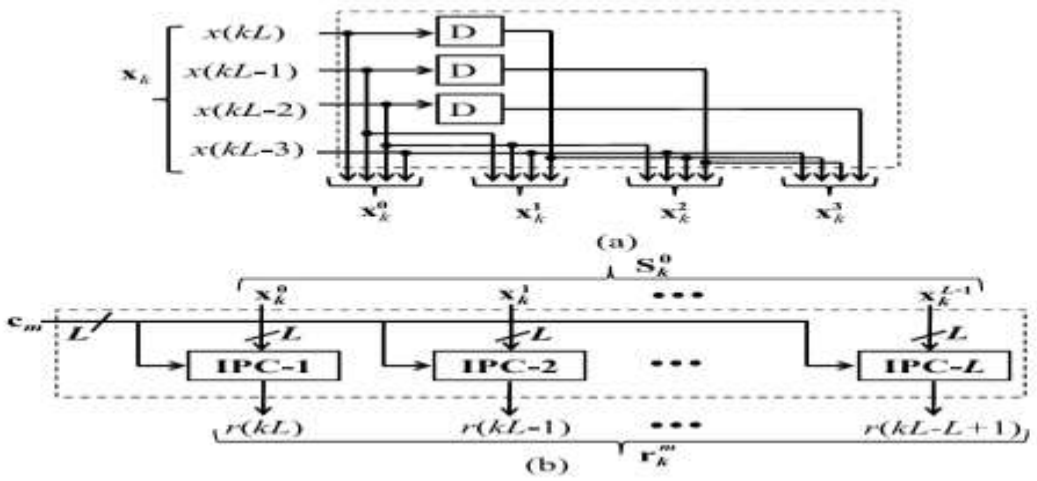


Fig.9. (a) Internal structure of RU for block size L = 4. (b)Structure of (m + 1) th IPU.

In this way, each IPU performs L inner-product calculations of L rows of S_k^0 with a typical weight vector c_m. The structure of the (m+1)th IPU is appeared in Fig. 7(b). It comprises of L number of L-point inner-product cells (IPCs). The (l+1)th IPC gets the (l+1)th row of $S_k^0$and the coefficient vector c_m, and computes a partial outcome of inner product r (kL −l), for $0 \le l \le L - 1$.

Internal structure of (l + 1)th IPC for L = 4 is appeared in Fig. 10(a). All the M IPUs work in parallel and deliver M blocks of result (r_k^m).These partial inner products are included the PAU [shown in Fig. 10(b)] to acquire a block of L filter outputs. In each cycle, the proposed structure gets a block of L inputs and produces a block of L filter outputs, where the length of each cycle is $T = T_M + T_A + T_{FA} \log_2 L$, T_M is one multiplier delay, T_A is one adder delay, and T_{FA} is one full-adder delay

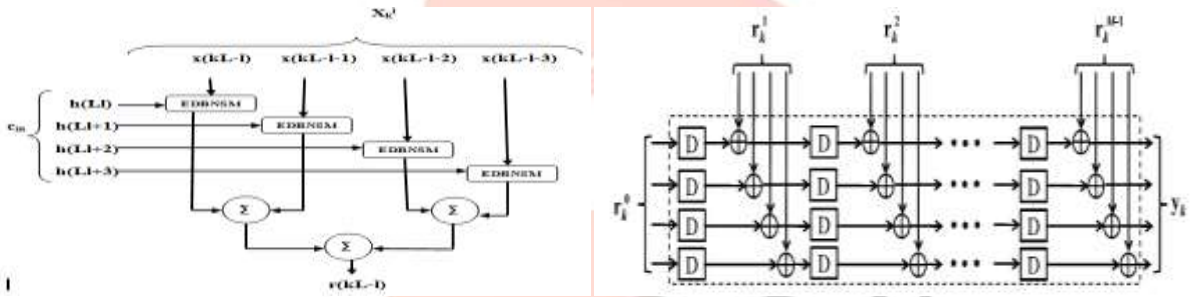


Fig.10. (a) Internal structure of (l + 1) th IPC for L = 4. (b) Structure of PAU for block size L = 4.

V. EXTENDED DOUBLE BASE NUMBER SYSTEM

As multipliers are much slower and expend generously more area than adders, programmable filter raises the cost and inertness of the system, as exemplified by the dominating complexity of coefficient computation in channel equalization. To avoid the time consuming and area intensive multipliers, they are supplanted by simpler arithmetic operators in the form of shift-and-add network based on the transposed direct form implementation. The shifted partial products of the filter coefficients have to be recalculated before they are delayed and accumulated by the structural adders. Subsequently, the shifters are not fixed but rather differ with the changing partial products. The partial products additionally should be either computed on request or precomputed and prestored for selection, which brings about a time multiplexed multiple constant multiplication (TM-MCM) square.

In this paper, we have extended the double base number system (DBNS) to summarize the binary shifts in tandem with several most repeatedly encountered common sub expressions. A new formulation of the common sub expression seek issue as a quasi-minimized extended DBNS (EDBNS) generation problem is proposed, which has prompted significant diminishment in the number of distinct partial products for a given word length of programmable coefficients. With this number system, an efficient design for the implementation of TM-MCM is derived as shown in Fig. 1. It comprises of a power-of- b generator (POBG), blocks of power-of- b selector (POBS) and blocks of double base coefficient generator (DBCG), where b is the second base number in EDBNS and N is the number of taps. In our design, those distinctive partial product terms can be generated incrementally with one adder each. The sizes of the multiplexers in the programmable units and the lookup tables for the selector logic are further minimized by exploiting the unique properties of EDBNS in the exponential Diophantine equation.

The output sequence of an n -tap finite impulse response (FIR) digital filter can be computed by the following discrete time convolution.

$$Y[n] = h[n]*x[n] = \sum_{i=0}^{N-1} h(i).x(n-i)$$

Where x[n] and y[n] are the n-th time domain input and output data samples. The samples h[i], i=0, 1, 2,N-1, also known as the filter coefficients, are the impulse response of the filter transfer function H(z), i.e., $H(z) = \sum_{i=0}^{N-1} h[i]\, z^{-1}$

For fixed point implementation, each coefficient is quantized into a finite precision integer. The quantized coefficient, denoted by c, can be written as a sum of power-of-two (POT) terms as follows:

$$C = \sum_{i=0}^{w-1} b_i \, 2^i$$

Where $b_i \in \{0,1\}$ for binary representation and $b_i \in \{0,1,-1\}$ for signed digit representation. is the word length required by the representation to express all the integer coefficients of the filter.

The discrete convolution can be implemented using a network of hardwired shifters and adders. The number of adders required is determined by the number of nonzero POT terms in the binary or signed digit representation of all coefficients of the filter. It can be incredibly diminished by sharing common POT terms (also known as common sub expressions) [20]. The search space for the design is achievable if the filter coefficients are fixed. If the coefficient values are allowed to change, such as those in an adaptive filter, then all the 2^w w -bit integers need to be represented. The sums of two adjoining nonzero POT terms in binary (i.e., bit pattern "11") and CSD (i.e., bit patterns " "), and their negation corresponding to the decimal numbers 3 and -3, appear to be the most frequently occurred common sub expressions in digital filter design. By recursively decaying c in (2) into the sum of $(2^{+\alpha+1} + 2^{\alpha}) = 3 \times 2^{\alpha}$ and a smaller integer of c-3 x 2^{α}, and factoring out 3 in each iteration, a sparse double base number system (DBNS) can be obtained. This DBNS is defined in [35] for any integer as:

$$C = \sum_{\alpha,\beta} d_{\alpha,\beta} 2^{\alpha} 3^{\beta}$$

Where, $d_{\alpha,\beta} \in \{0, 1\}$, α and β and are the exponents of 2 and 3 of the double base (or 2-integer) term, respectively.

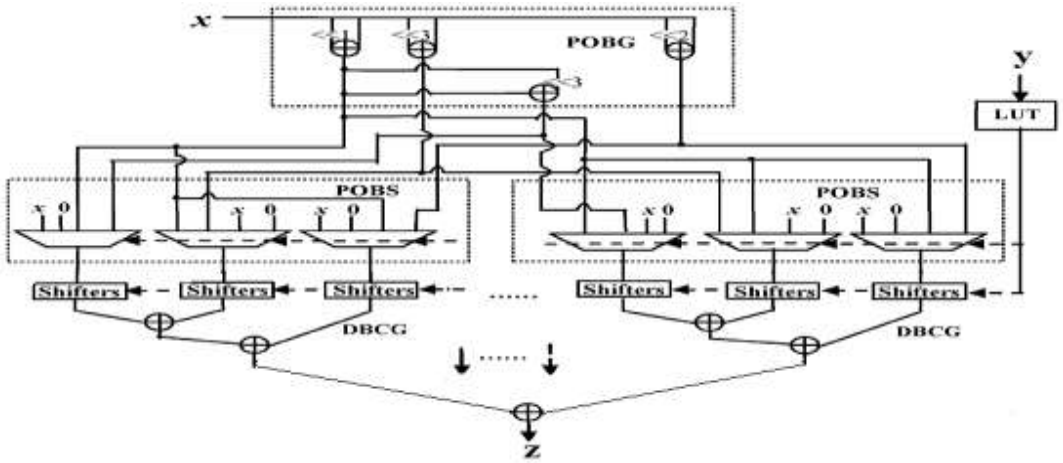


Fig.11 EDBNS multiplier

VI. MCM-BASED IMPLEMENTATION OF FIXED-COEFFICIENT FIR FILTER

The coefficients of FIR filter are fixed in some application which is known as fixed coefficients. The transpose form FIR filter is naturally a pipelined structure which supports the multiple constant multiplications (MCM) technique however direct form FIR filter structure does not support MCM technique. The MCM is more efficient in Transpose form when the common operand is multiple with the set of constant coefficients that diminish the computational delay. The implementation of MCM technique is simpler in fixed coefficient Transpose form FIR filter but complex in reconfigurable coefficients. In fixed coefficients transpose FIR filter, area and delay are diminished by utilizing MCM technique. For fixed-coefficient implementation, the CSU is no longer required, since the structure is to be customized for just one given filter. Similarly, IPUs are not required. The multiplications are required to be mapped to the MCM units for a low-complexity realization.

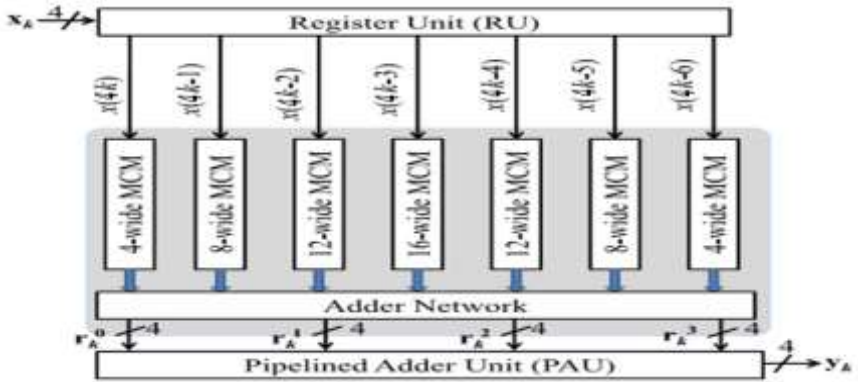


Fig.12 Proposed MCM-based structure for fixed FIR filter of block size L = 4 and filter length N = 16.

In the following, we show that the proposed formulation for MCM-based implementation of block FIR filter makes use of the symmetry in input matrix $\mathbf{S}_k^0$ to execute horizontal and vertical common sub expression deduction and to reduce the number of shift-add operations in the MCM blocks.

Table .1 MCM in Transpose Form Block Fir Filter of Length 16 and Block Size 4

Input sample	Coefficient Group
$x(4k)$	$\{h(0), h(4), h(8), h(12)\}$
$x(4k-1)$	$\{h(0), h(4), h(8), h(12)\}$ $\{h(1), h(5), h(9), h(13)\}$
$x(4k-2)$	$\{h(0), h(4), h(8), h(12)\}$ $\{h(1), h(5), h(9), h(13)\}$ $\{h(2), h(6), h(10), h(14)\}$
$x(4k-3)$	$\{h(0), h(4), h(8), h(12)\}$ $\{h(1), h(5), h(9), h(13)\}$ $\{h(2), h(6), h(10), h(14)\}$ $\{h(3), h(7), h(11), h(15)\}$
$x(4k-4)$	$\{h(1), h(5), h(9), h(13)\}$ $\{h(2), h(6), h(10), h(14)\}$ $\{h(3), h(7), h(11), h(15)\}$
$x(4k-5)$	$\{h(2), h(6), h(10), h(14)\}$ $\{h(3), h(7), h(11), h(15)\}$
$x(4k-6)$	$\{h(3), h(7), h(11), h(15)\}$

To illustrate the computation of 4 tap Fixed FIR and 16 coefficients, we write it as a matrix product by utilizing the above relation. We can observe that the input matrix contains six-input samples $\{x(4k), x(4k-1), x(4k-2), x(4k-3), x(4k-4), x(4k-5), x(4k-6)\}$, and multiplied with several constant coefficients. MCM can be applied in both horizontal and vertical direction of the coefficient matrix. The sample $x(4k-3)$ appears in four rows or four columns of the above input matrix.

Input matrix:

$$\mathbf{R} = \begin{bmatrix} x(4k) & x(4k-1) & x(4k-2) & x(4k-3) \\ x(4k-1) & x(4k-2) & x(4k-3) & x(4k-4) \\ x(4k-2) & x(4k-3) & x(4k-4) & x(4k-5) \\ x(4k-3) & x(4k-4) & x(4k-5) & x(4k-6) \end{bmatrix} \times \begin{bmatrix} h(0) & h(4) & h(8) & h(12) \\ h(1) & h(5) & h(9) & h(13) \\ h(2) & h(6) & h(10) & h(14) \\ h(3) & h(7) & h(11) & h(15) \end{bmatrix}$$

Whereas $x(4k)$ appears in only one row or one column. Subsequently, all the four rows of coefficient matrix are involved in the MCM for the $x(4k-3)$, whereas only the first row of coefficients are involved in the MCM for $x(4k)$. For larger values of N or the smaller block sizes, the row size of the coefficient matrix is larger that results in larger MCM size across all the samples, which results into larger saving in computational complexity.

VII. RESULTS AND COMPARISIONS

Reconfigurable FIR filter structure in transpose form configuration is implemented using VHDL language for filter length 16 and block size L=4. Xilinx ISE web pack 9.1 software is used for design, synthesis and implementation. VHDL is used to describe the behavior and structure of system and circuit design. The proposed structure is simulated on the Xilinx system generator. The simulated output waveforms are obtained with the help of Model Sim 6.3csoftware. Area, delay and power can be measured and comparisons are performed.

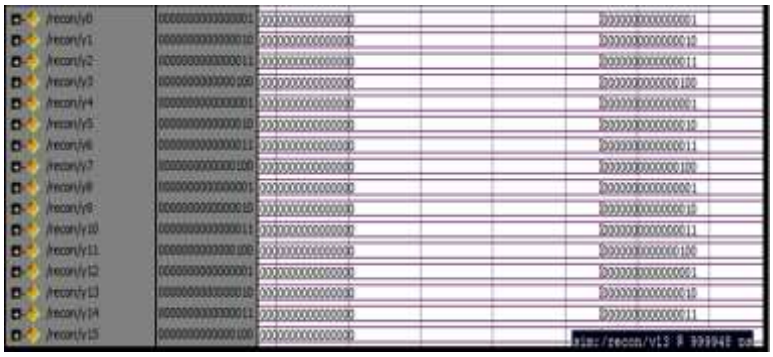
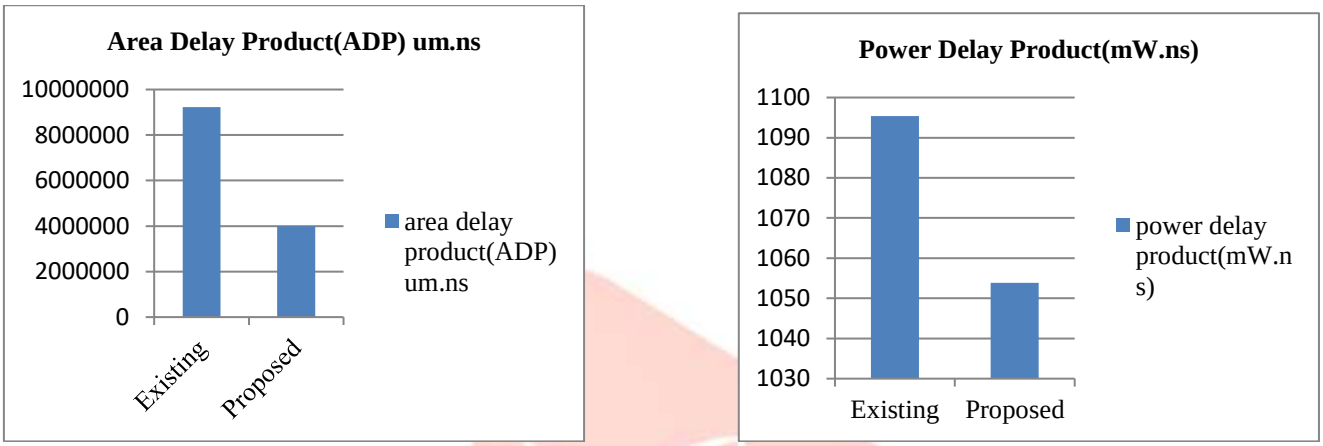
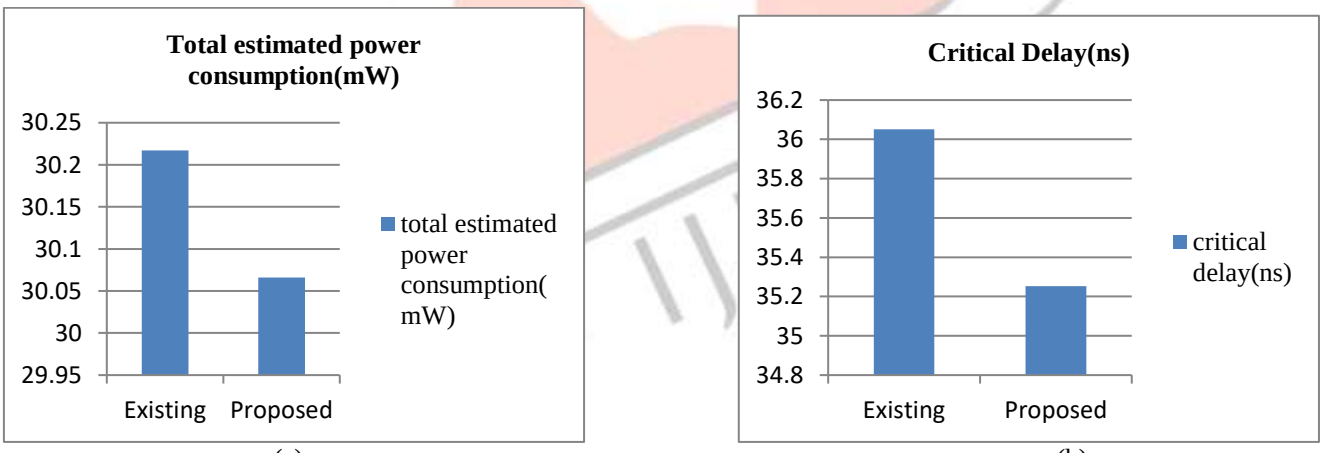


Fig.13. simulated wave form



(a) (b)

Fig.14 ADP and PDP of the proposed structure compared with existing structure. (a) ADP . (b) PDP.



(a) (b)

Fig.15 Power and Delay comparisions. (a) power. (b) delay

VIII. CONCLUSION

Thus the possibility of block realization in FIR filters for transpose form configuration with area delay efficient realization of both fixed and reconfigurable applications have been exploited by using EDBNS algorithm for multiplication between filter coefficients and samples in inner product unit. The memory based approach of storing and retrieving the product value is much faster than performing a computation at runtime The new architecture ensures low-complexity as well as reconfigurability for FIR filters in the channelizers of SDRs. The architecture has been simulated and synthesized by Xilinx 9.li ISE & X-Power analysis is employed for the calculation of power dissipation. Performance comparison shows that the proposed structure with EDBNS involves significantly less ADP and less PDP than the existing architecture.

REFERENCES

[1] Basant Kumar Mohanty and Pramod Kumar Meher," A High-Performance FIR Filter Architecture for Fixed and Reconfigurable Applications" IEEE transactions on very large scale integration (VLSI) systems, vol. 24, no. 2, Feb. 2016.
[2] Jiajia Chen, Member, IEEE, Chip-Hong Chang, Senior Member, IEEE, Feng Feng, Weiao Ding, and Jiatao Ding "Novel Design Algorithm for Low Complexity Programmable FIR Filters Based on Extended Double Base Number System" IEEE transactions on circuits and systems—I, vol. 62, no. 1, Jan. 2015
[3] S. Y. Park and P. K. Meher, "Efficient FPGA and ASIC realizations of a DA-based reconfigurable FIR digital filter," IEEE Trans. CircuitsSyst. II, Exp. Briefs, vol. 61, no. 7, pp. 511–515, Jul. 2014
[4] M. Cieplucha, "High performance FPGA-based implementation of a parallel multiplier-accumulator," in Proc. 20th Int. Conf.Mixed Design Integr. Circuits Syst., Gdynla, Poland, Jun. 2013, pp. 485–489.
[5] P. K. Meher, "New approach to look-up-table design and memory-based realization of FIR digital filter," IEEE Trans. CircuitsSyst. I, Reg. Papers, vol. 57, no. 3, pp. 592–603, Mar. 2010.
[6] R. Mahesh and A. P. Vinod, "New reconfigurable architectures for implementing FIR filters with low complexity," IEEE Trans. Comput.-AidedDesign Integr. Circuits Syst., vol. 29, no. 2, pp. 275–288, Feb. 2010.
[7] P. K. Meher, S. Chandrasekaran, and A. Amira, "FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic," IEEE Trans. Signal Process., vol. 56, no. 7, pp. 3009–3017,
Jul. 2008.
[8] P. K. Meher, "Hardware-efficient systolization of DA-based calculation of finite digital convolution," IEEE Trans. Circuits Syst. II, Exp. Briefs,vol. 53, no. 8, pp. 707–711, Aug. 2006.
[9] K.-H. Chen and T.-D. Chiueh, "A low-power digit-based reconfigurableFIR filter," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 8,pp. 617–621, Aug. 2006.
[10] A. P. Vinod and E. M. Lai, "Low power and high-speed implementation of FIR filters for software defined radio receivers," IEEE Trans. Wireless Commun., vol. 7, no. 5, pp. 1669–1675, Jul. 2006.
[11] J. Park, W. Jeong, H. Mahmoodi-Meimand, Y. Wang, H. Choo, andK. Roy, "Computation sharing programmable FIR filter for low-power and high-performance applications," IEEE J. Solid State Circuits, vol. 39, no. 2, pp. 348–357, Feb. 2004.
[12] J. Mitola, Software Radio Architecture: Object-Oriented Approaches to Wireless Systems Engineering. New York, NY, USA: Wiley, 2000.
[13] T. Hentschel and G. Fettweis, "Software radio receivers," inCDMA Techniques for Third Generation Mobile Systems. Dordrecht, the Netherlands: Kluwer, 1999, pp. 257–283.
[14] J. G. Proakis and D. G. Manolakis, Digital Signal Processing: Principles, Algorithms and Applications. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.
[15] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process. vol. 42, no. 9, pp. 569–577, Sep. 1995..
[16] E. Mirchandani, R. L. Zinser, Jr., and J. B. Evans, "A new adaptive noise cancellation scheme in the presence of crosstalk [speech signals]," IEEETrans. Circuits Syst. II, Analog Digit. Signal Process., vol. 39, no. 10,
pp. 681–694, Oct. 1995.
[17] D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in Proc. IEEESoutheastcon, Apr. 1993, p. 1–6.

.