Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 5, Issue 01, January -2018

# Design of High speed fixed and reconfigurable FIR filter for speech signal processing

D. Sony,

*Assistant Professor, Department of ECE,*
*Geethanjali College of Engineering and Technology, Hyderabad, Telangana, India.*

**Abstract**—*Transpose form finite-impulse response (FIR) filters are inherently pipelined and support multiple constant multiplications (MCM) technique that results in significant saving of computation. However, transpose form configuration 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-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 than the existing block implementation of direct-form structure for medium or large filter lengths, while for the short-length filters For the same filter length and the same block size, the proposed structure involves less area and delay that of the existing direct-form block FIR structure. All the synthesis and simulation results of the Proposed High performance FIR Filters are performed on Xilinx ISE 14.7 using Verilog HDL.*

***Index Terms***— *Block processing, finite-impulse response (FIR) filter, reconfigurable architecture, VLSI.*

## I. INTRODUCTION

FINITE-IMPULSE response (FIR) digital filter is widely used in several digital signal processing 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]. Very often 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. Since there is no redundant computation available in the FIR filter algorithm, real-time implementation of a large order FIR filter in a resource constrained environment is a challenging task. Filter coefficients very often remain constant and known a priori in signal processing applications. This feature has been utilized to reduce the complexity of realization of multiplications. Several designs have been suggested by various researchers for efficient realization of FIR filters (having fixed coeffi- cients) using distributed arithmetic (DA) [18] and multiple constant multiplication (MCM) methods [7], [11]–[13]. DA-based designs use lookup tables (LUTs) to store pre-computed results to reduce the computational complexity. The MCM method on the other hand reduces the number of additions required for the realization of multiplications by common sub-expression sharing, when a given input is multiplied with a set of constants. The MCM scheme is more effective, when a common operand is multiplied with more number of constants. Therefore, the MCM scheme is suitable for the implementation of large order FIR filters with fixed coefficients. But, MCM blocks can be formed only in the transpose form configuration of FIR filters.

Block-processing method is popularly used to derive high-throughput hardware structures. It not only provides throughput-scalable design but also improves the area-delay efficiency. The derivation of block-based FIR structure is straightforward when direct-form configuration is used [16], whereas the transpose form configuration does not directly support block processing. But, to take the computational advantage of the MCM, FIR filter is required to be realized by transpose form configuration. Apart from that, transpose form structures are inherently pipelined and supposed to offer higher operating frequency to support higher sampling rate.

There are some applications, such as SDR channelizer, where FIR filters need to be implemented in a reconfigurable hardware to support multi-standard wireless communication [6]. Several designs have been suggested during the last decade for efficient realization of reconfigurable FIR (RFIR) using general multipliers and constant multiplication schemes [7]–[10]. A RFIR filter architecture using computation sharing vector-scaling technique has been proposed in [7]. Chen and Chiueh [8] have proposed a canonic sign digit (CSD)-based RFIR filter, where the nonzero CSD values are modified to reduce the precision of filter coefficients without significant impact on filter behavior. But, the reconfiguration overhead is significantly

large and does not provide an area-delay efficient structure. The architectures in [7] and [8] are more appropriate for lower order filters and not suitable for channel filters due to their large area complexity. Constant shift method (CSM) and programmable shift method have been proposed in [9] for RFIR filters, specifically for SDR channelizer. Recently, Park and Meher [10] have proposed an interesting DA-based architecture for RFIR filter. The existing multiplier-based structures use either directform configuration or transpose form configuration. But, the multiplier-less structures of [9] use transpose form configuration, whereas the DA-based structure of [10] uses direct-form configuration. But, we do not find any specific block-based design for RFIR filter in the literature. A block-based RFIR structure can easily be derived using the scheme proposed in [15] and [16]. But, we find that the block structure obtained from [15] and [16] is not efficient for large filter lengths and variable filter coefficients, such as SDR channelizer. Therefore, the design methods proposed in [15] and [16] are more suitable for 2-D FIR filters and block least mean square adaptive filters.

I explore the possibility of realization of block FIR filter in transpose form configuration in order to take advantage of the MCM schemes and the inherent pipelining for area-delay efficient realization of large order FIR filters for both fixed and reconfigurable applications.

I have accompanied underneath technique to achieve my challenge.

1) Statistics flow Graphs derivation for transpose shape block FIR filter with decreased register complexity.
2) System of transpose shape block FIR filter.
3) Layout of transpose shape block filter out for reconfigurable applications of FIR filters.
4) By means of the usage of the MCM idea for the effectuation of block FIR filter with constant degree for less-complexity design.

The remainder of this paper is organized as follows. In Section II, computational analysis and mathematical formulation of block transpose form FIR filter are presented. The proposed architectures for fixed and reconfigurable applications are presented in Section III. Hardware and time complexities along with performance comparison are presented in Section IV. Finally, the conclusion is drawn in Section V.

## II. COMPUTATIONAL ANALYSIS AND MATHEMATICAL FORMULATION OF BLOCK TRANSPOSE FORM FIR FILTER

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

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

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

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

*A. Computational Analysis*

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 derived from (2). The product values and their accumulation paths in DFG-1 and DFG-2 of Fig. 1 are shown in data- flow tables (DFT-1 and DFT-2) of Fig. 2. The arrows in DFT-1 and DFT-2 of Fig.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 (shown in gray color in Fig. 2).

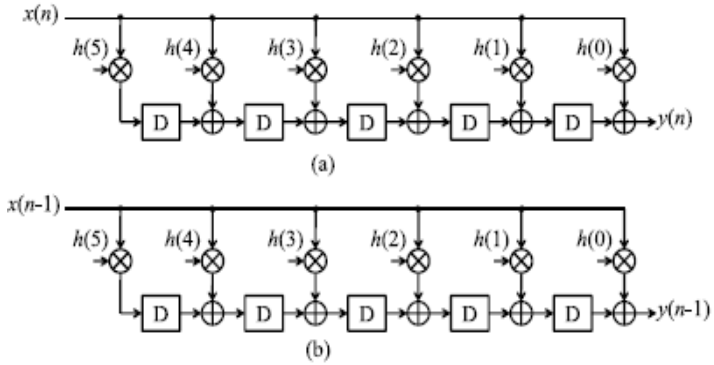



Fig.1. data waft Graphs of transpose form structure for filter period N=6. a) DFG-1 for output y (n). (b) DFG-2 for output y (n−1).

These redundant computations of DFG-1 and DFG-2 can be avoided using non-overlapped sequence of input blocks, as shown in Fig. 3. DFT-3 and DFT-4 of DFG-1 and DFG-2 for non-overlapping input blocks are, respectively, shown in Fig. 3(a) and (b). As shown in Fig. 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 Fig. 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 Fig. 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 | $x(n-5)h(5)$ | $x(n-5)h(4)$ | $x(n-5)h(3)$ | $x(n-5)h(2)$ | $x(n-5)h(1)$ | $x(n-5)h(0)$ |
| 2 | $x(n-4)h(5)$ | $x(n-4)h(4)$ | $x(n-4)h(3)$ | $x(n-4)h(2)$ | $x(n-4)h(1)$ | $x(n-4)h(0)$ |
| 3 | $x(n-3)h(5)$ | $x(n-3)h(4)$ | $x(n-3)h(3)$ | $x(n-3)h(2)$ | $x(n-3)h(1)$ | $x(n-3)h(0)$ |
| 4 | $x(n-2)h(5)$ | $x(n-2)h(4)$ | $x(n-2)h(3)$ | $x(n-2)h(2)$ | $x(n-2)h(1)$ | $x(n-2)h(0)$ |
| 5 | $x(n-1)h(5)$ | $x(n-1)h(4)$ | $x(n-1)h(3)$ | $x(n-1)h(2)$ | $x(n-1)h(1)$ | $x(n-1)h(0)$ |
| 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 | $x(n-6)h(5)$ | $x(n-6)h(4)$ | $x(n-6)h(3)$ | $x(n-6)h(2)$ | $x(n-6)h(1)$ | $x(n-6)h(0)$ |
| 2 | $x(n-5)h(5)$ | $x(n-5)h(4)$ | $x(n-5)h(3)$ | $x(n-5)h(2)$ | $x(n-5)h(1)$ | $x(n-5)h(0)$ |
| 3 | $x(n-4)h(5)$ | $x(n-4)h(4)$ | $x(n-4)h(3)$ | $x(n-4)h(2)$ | $x(n-4)h(1)$ | $x(n-4)h(0)$ |
| 4 | $x(n-3)h(5)$ | $x(n-3)h(4)$ | $x(n-3)h(3)$ | $x(n-3)h(2)$ | $x(n-3)h(1)$ | $x(n-3)h(0)$ |
| 5 | $x(n-2)h(5)$ | $x(n-2)h(4)$ | $x(n-2)h(3)$ | $x(n-2)h(2)$ | $x(n-2)h(1)$ | $x(n-2)h(0)$ |
| 6 | $x(n-1)h(5)$ | $x(n-1)h(4)$ | $x(n-1)h(3)$ | $x(n-1)h(2)$ | $x(n-1)h(1)$ | $x(n-1)h(0)$ |

(b)

Fig.2. (a) Multipliers of DFG has shown similar to output y (n).
(b) Multipliers of DFG has proven corresponding to output y(n-1).

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

(a)

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

(b)

Fig.3. DFT of DFG-1 and DFG-2 for three non-overlapped 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).

B. DFG Transformation

The computation of DFT-3 and DFT-4 can be realized by DFG-3 of non-overlapping blocks, as shown in Fig.4. We refer it to block transpose form type-I configuration of block FIR filter. The DFG-3 can be retimed to obtain the DFG-4 of Fig. 5, which is referred to block transpose form type-II configuration. Note that both type-I and type-II configurations involve the same number of multipliers and adders, but type-II configuration involves nearly L times less delay elements than those of

type-I configuration. We have, therefore, used block transpose form type-II configuration to derive the proposed structure. In Section II-C, we present mathematical formulation of block transpose form type-II FIR filter for a generalized formulation of the concept of block-based computation of transpose form FIR filers.

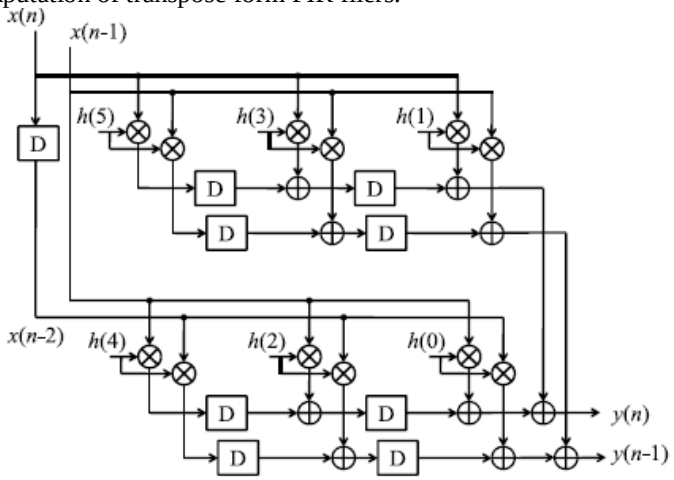



Fig.4. Merged DFG (DFG-3: transpose form type-I configuration for block FIR structure).

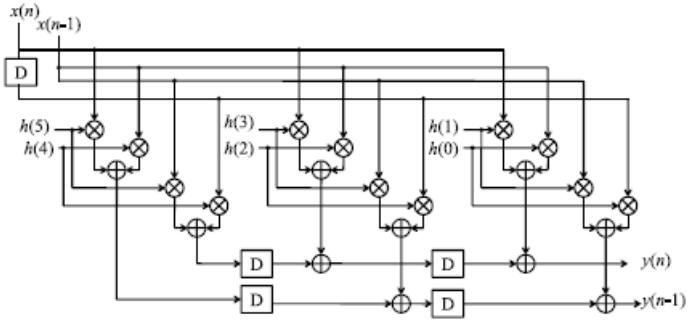



Fig.5. DFG-4 (retimed DFG-3) transpose form type-II configuration for block FIR structure.

C. Mathematical Formulation of the Transpose Form Block FIR

Filter Suppose in every cycle, the block FIR filter takes a block of L new input samples, and processes those to produce a block of L output samples. The k-th block of filter output y (k) is computed using the relation

$$y(k) = X_k.h \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

$$X_k = [X_k{}^0 + X_k{}^1 + \ldots X_k{}^4 \ldots\ldots + X_k{}^{N-1}]$$

Where the weight vector h is defined as

$$h = [h(0), h(1), h(2), \ldots\ldots, h(N-1)]^T$$

The input matrix $X_k$ is define …… (4)

Here $X_k{}^i$ is the (i+1)th column of $X_k$ are defined as $X_k{}^i$

$$= [x(kL-i), x(kL-i-1)\ldots\ldots x(kL-i-L+1)]^T \quad \ldots.. (5)$$

Substituting (4) in (3), the matrix-vector product is expressed in the form of scalar–vector product as

$$y(k) = \sum_{i=0}^{N-1} h(i).\ X_k{}^i \quad \ldots\ldots\ldots\ldots (6)$$

Suppose N is a composite number and decomposed as N = M L, then index i is expressed as i = l + mL, for $0 \le l \le L-1$, and $0 \le m \le M-1$. Substituting i = l + mL in (5), we have

$$X_k^{l+ml} = X_{k-m}^{l} \quad \text{..... (7)}$$

Substituting (7) in (4), we have

$$\mathbf{X}_k = [\mathbf{x}_k^0 \ \mathbf{x}_k^1 \cdots \mathbf{x}_k^{L-1} \ \mathbf{x}_{k-1}^0 \ \mathbf{x}_{k-1}^1 \cdots \mathbf{x}_{k-1}^{L-1} \cdots \mathbf{x}_{k-M+1}^0 \mathbf{x}_{k-M+1}^1 \cdots \mathbf{x}_{k-M+1}^{L-1}] \quad (8)$$

Substituting (8) in (3), we have

$$Y(k) = \sum_{l=0}^{L-1} \sum_{m=0}^{M-1} h(l+ml).\ X_{k-m}^{l} \quad \text{.....(9)}$$

Therefore, the input matrix $X_k$ from equation (8), it has a concerning feature. The current data block is $X_k^0$. The blocks are delayed by 1, 2… (M-1) cycles, while

$$X_k = [X_{k-1}^0, X_{k-2}^0, \ldots\ldots, X_{k-M+1}^0] \ .$$

The coincided blocks are shown by various clock cycles periods that is 1 clock cycle, 2 clock cycle… (M-1) cycles delayed pattern of coincided block $X_k^1$. If I want to take the feature of this advantage, the input matrix $X_k$ is broken down into M small matrices $S_k^0$ and it contains L-input blocks $\{X_k^0, X_k^1, \ldots\ldots\ldots, X_k^{L-1}\}$, and $S_k^1$ input blocks $\{X_{k-1}^0, X_{k-1}^1, \ldots\ldots\ldots, X_{k-1}^{L-1}\}$. Likewise, the input block $\{X_{k-M+1}^0, X_{k-M+1}^1, \ldots\ldots, X_{k-M+1}^{L-1}\}$ constituted the matrix $X_k^{M-1}$. The vector coefficient h is also broken down into small weight vectors

$C_m = \{h(mL), h(mL+1), \ldots\ldots, h(mL+L-1)\}$. Interestingly, $S_k^m$ is symmetric and gives the following equation.

$$S_k^m = S_{k-m}^0 \quad \text{..... (10)}$$

According to the equation (10), $S_k^m$ lies between $(1 \le m \le M-1)$. Where small m is the cycle delayed with respect to $S_k^0$. After computation of equation (9) can be shown in matrix vector product using $S_{k-m}^0$ and $C_m$ . It has shown in the below as

$$Y_k = \sum_{m=1}^{M-1} r_k^m \quad \text{..... (11a)}$$

$$r_k^m = S_{k-m}^0 . C_m \quad \text{..... (11b)}$$

The computations of (11) may be expressed in a recurrence form

$$\mathbf{Y}(z) = \mathbf{S}^0(z)[(z^{-1}(\cdots(z^{-1}(z^{-1}\mathbf{c}_{M-1} + \mathbf{c}_{M-2}) + \mathbf{c}_{M-3}) + \cdots) + \mathbf{c}_1) + \mathbf{c}_0] \quad (12)$$

Here $S^0(Z)$ and Y (z) are the z-domain representation of $S_k^0$ and $y_k$ , respectively. The Data Flow Graph-4 of transpose form block model-II configuration (demonstrated Fig. 5 for N=6 and L=2). It can be deduced using the recurrence relation of (12). The delay operator $\{z^{-1}\}$ of (12) constitutes a delay for a block of data in the transpose form eccentric-II structure that stores the product of $S_k^0$ and $C_m$ . The proposed structure (transpose form type-II) is presented in Section III.

## III. PROPOSED STRUCTURES

There are several applications where the coefficients of FIR filters remain fixed, while in some other applications, like SDR channelizer that requires separate FIR filters of different specifications to extract one of the desired narrowband channels from the wideband RF front end. These FIR filters need to be implemented in a RFIR structure to support multi-standard wireless communication [6]. In this section, we present a structure of block FIR filter for such reconfigurable applications. In this section, we discuss the implementation of block FIR filter for fixed filters as well using MCM scheme.

### *A. Proposed Structure for Transpose Form Block FIR Filter for Reconfigurable Applications*

The proposed structure for block FIR filter is [based on the recurrence relation of (12)] shown in Fig. 6 for the block size L=4. It consists of one coefficient selection unit (CSU), one register unit (RU), M number of inner-product units (IPUs), and one pipeline adder unit (PAU). The CSU stores coefficients of all the filters to be used for the reconfigurable application.

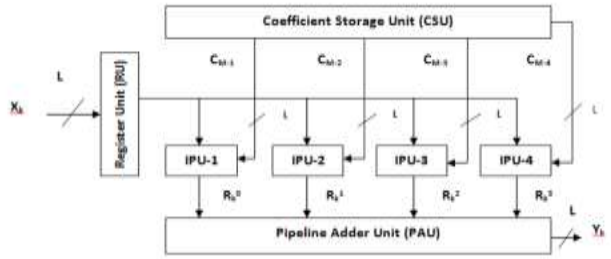
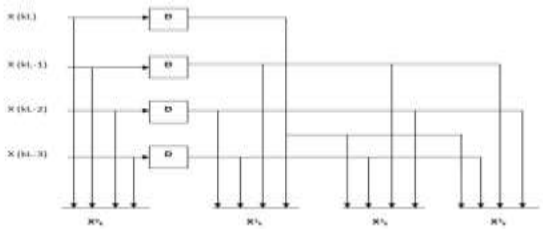



Fig.1. Block diagram of MCM with fixed coefficient architecture

It is implemented using N ROM LUTs, such that filter coefficients of any particular channel filter are obtained in one clock cycle, where N is the filter length. The RU [shown in Fig.7 (a)] receives xk during the kth cycle and produces L rows of $s_k^0$ in parallel. L rows of $s_k^0$ are transmitted to M IPUs of the proposed structure. The M IPUs also receive M short-weight vectors from the CSU such that during the kth cycle, the (m+1)th IPU receives the weight vector cM−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 $r_k^m$ . Therefore, each IPU performs L inner-product computations of L rows of $s_k^0$ with a common weight vector cm. The structure of the (m +1)th IPU is shown in Fig.7(b). It consists of L number of L-point inner-product cells (IPCs). The (l +1)th IPC receives the (l +1)th row of $s_k^0$ and the coefficient vector cm, and computes a partial result of inner



product r(kL−l), for $0 \le l \le L-1$.

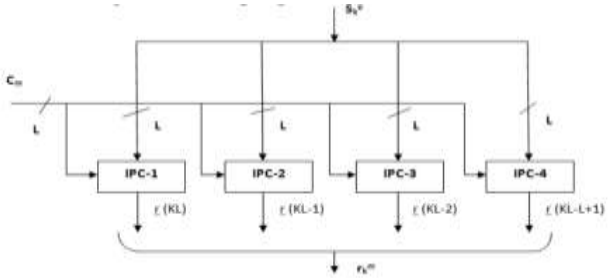



Fig.7. (a) Internal structure of RU for block size L=4. (b) Structure of (m + 1)th IPU.

Internal structure of (l + 1) th IPC for L=4 is shown in Fig.8 (a). All the M IPUs work in parallel and produce M blocks of result $r_k^m$ . These partial inner products are added in the PAU [shown in Fig. 8(b) ] to obtain a block of L filter outputs. In each

cycle, the proposed structure receives a block of L inputs and produces a block of L filter outputs, where the duration of each cycle is $T = [T_M + T_A + T_{FA}(\log_2 L)]$. TM is one multiplier delay, TA is one adder delay, and TFA is one full-adder delay.

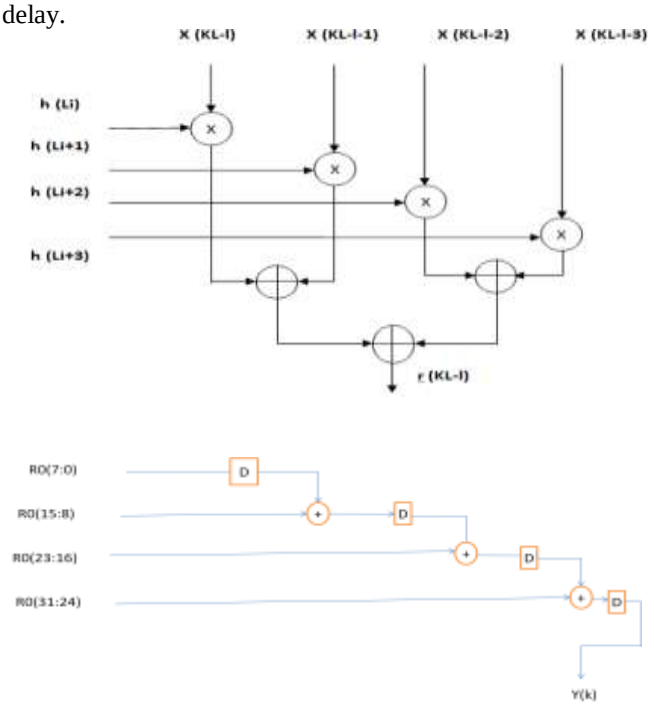



Fig.8. (a) Internal structure of (l + 1)-th IPC for L=4. (b) Structure of PAU for block size L = 4.

B. MCM-Based Implementation of Fixed-Coefficient FIR Filter

We discuss the derivation of MCM units for transpose form block FIR filter, and the design of proposed structure for fixed filters. For fixed-coefficient implementation, the CSU of Fig. 6 is no longer required, since the structure is to be tailored for only one given filter. Similarly, IPUs are not required. The multiplications are required to be mapped to the MCM units for a low-complexity realization. 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 $s_k{}^0$ to perform horizontal and vertical common sub-expression elimination [17] and to minimize the number of shift-add operations in the MCM blocks.

The recurrence relation of (12) can alternatively be expressed as

$$\mathbf{Y}(z) = z^{-1} \cdots z^{-1}(z^{-1}\mathbf{r}_{M-1} + \mathbf{r}_{M-2} + \mathbf{r}_{M-3} + \cdots + \mathbf{r}_1 + \mathbf{r}_0 \quad (13)$$

The M intermediate data vectors rm, for $0 \le m \le M-1$ can be computed using the relation

$$\mathbf{R} = \mathbf{S}_k^0 \cdot \mathbf{C} \quad (14)$$

Where R and C are defined as

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_0^T & \mathbf{r}_1^T & \cdots & \mathbf{r}_{M-1}^T \end{bmatrix} \quad (15a)$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_0^T & \mathbf{c}_1^T & \cdots & \mathbf{c}_{M-1}^T \end{bmatrix} \quad (15b)$$

To illustrate the computation of (14) for L=4 and N=16, we write it as a matrix product given by (16). From (16), 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, as shown in Table I

TABLE I: MCM IN TRANSPOSE FORM BLOCK FIR FILTER OF LENGTH 16 AND BLOCK SIZE 4

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

Input matrix:

$$\mathbf{R} = \begin{bmatrix} x(4k) & x(4k-1) & x(4k-2) & x(4k-3) \\ x(4k-1) & x(4k-2) & x(4k-3) & x(4k-4) \\ x(4k-2) & x(4k-3) & x(4k-4) & x(4k-5) \\ x(4k-3) & x(4k-4) & x(4k-5) & x(4k-6) \end{bmatrix} \times \begin{bmatrix} h(0) & h(4) & h(8) & h(12) \\ h(1) & h(5) & h(9) & h(13) \\ h(2) & h(6) & h(10) & h(14) \\ h(3) & h(7) & h(11) & h(15) \end{bmatrix} \quad (16)$$

Whereas x (4k) appears in only one row or one column. Therefore, 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.

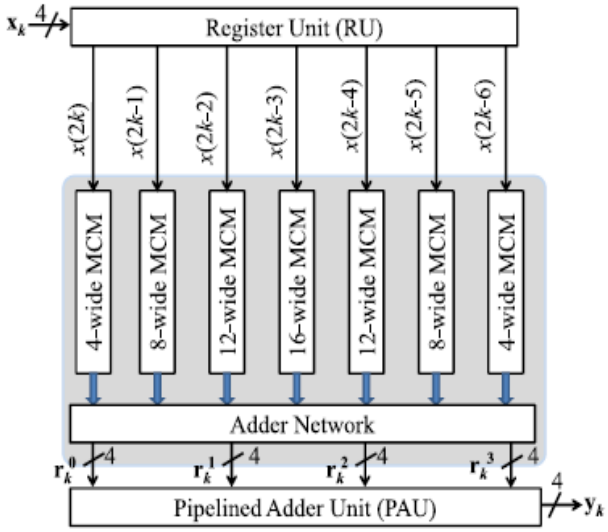



Fig.9. Proposed MCM-based structure for fixed FIR filter of block size L=4 and filter length N=16.

The proposed MCM-based structure for FIR filters for block size L=4 is shown in Fig.9 for the purpose of illustration. The MCM-based structure (shown in Fig.9) involves six MCM blocks corresponding to six input samples. Each MCM block produces the necessary product terms as listed in Table I. The sub-expressions of the MCM blocks are shift added in the

adder network to produce the inner-product values (rl,m), for $0 \le l \le L-1$ and $0 \le m \le (N/L)-1$ corresponding to the matrix product of (14). The inner-product values are finally added in the PAU of Fig.8 (b) to obtain a block of filter output.

## IV. COMPLEXITIES AND PERFORMANCE CONSIDERATIONS

### A. Hardware and Time Complexities

The proposed structure for reconfigurable application consists of one CSU, one RU, M IPUs, and one PAU. The CSU consists of N ROM units of P words each, where P is the number of FIR filters to be implemented by the proposed reconfigurable structure. We have excluded complexity of CSU in the performance comparison, since it is common in all the RFIR structures. Each IPU is comprised of L IP cells, where each IP cell involves L multipliers and (L−1) adders. The RU involves (L−1) registers of B-bit width. The PAU involves L (M−1) adders and the same number of registers, where each register has a width of (B+B ), B, and B respectively, being the bit width of input sample and filter coefficients. Therefore, the proposed structure involves L N multipliers, L(N−1) adders, and [B(N−1) + B (N−L)] (flip flops) FFs; and processes L samples in every cycle where the duration of cycle period $T = [T_M + T_A + T_{FA}(\log_2 L)]$. We do not find a multiplier-based direct-form block FIR structure on RFIR in the literature. However, direct-form multiplier-based block FIR structure can be derived from the block formulation of [15]. We have derived the direct-form block FIR structure using [15, eq. (4)], and estimated its hardware and time complexities for comparison purpose.

### B. Performance Comparison

The hardware and time complexities of the proposed structure and the extracted direct-from structure of [15] along with those of the existing RFIR filter structures in [9] and [10] are listed in Table II for comparison. We have assumed fixed word length (B+B) for the adder tree in case of direct-form structure, as well as the pipeline adder in case of transpose form structure. As shown in Table II, the direct-form structure of [15] and the proposed structures involve the same number of multipliers and adders, but the proposed one involves {(log2 M−1)TFA} less cycle period, where M = N/L, at a marginal cost of B (N−L) FFs. The register complexity of the proposed structure is independent of block size as in the case of direct-form structure. However, the cycle period of the proposed structure depends on the input-block size, whereas in case of the existing direct-form block FIR structure of [15], it depends on the filter length. Since filter length is usually higher than the block length, the cycle period of the existing direct-form structure increase for large order filters. To compare with the DA-based structure of [10] and the proposed structure, we find that the proposed structure involves (LN) multipliers in place of (3NBB/2) MUXes (bit level), nearly (2L/B) times more adders and B N more FFs, but offers nearly L times higher throughput. Similarly, compared with the CSM-based structure of [9], the proposed structure involve (LN) multipliers in place of ≈(7NBB/3) MUXes (bit level), ≈(3L/B) times more adders, B (L−1) less FFs, and offers L times higher throughput. In spite of more FFs, the proposed structure may have less area-delay product (ADP) than the existing direct-form structure due to its small cycle period.

TABLE II GENERAL COMPARISON OF HARDWARE AND TIME COMPLEXITIES

We have estimated hardware and time complexities of the proposed structure for block sizes L=4, 8, and filter lengths N=32 and 64. Also, we have estimated the hardware and the time complexities of the direct-form structure extracted from [15] for the same block size and for the filter lengths, and those in [9] and [10] for the same filter lengths. We have considered B = 8 (word length of input sample), B = 16 (word length of filter coefficient), and 24-bit word length for the intermediate and output signals for all the designs. The estimated values are listed in Table III for comparison. We can find from Table III that the multiplier and adder complexities of the proposed structure increases proportionately with block size and filter length as in the case of direct-form structure. The cycle period of direct-form structure increases proportionately with the filter length such that it increases by an amount TFA when filter-length doubles. But, cycle period of the proposed structure is independent of filter length and increases by an amount TFA when the block size doubles. The area-delay performance of the proposed structure is found better than that of direct-form structure of [15] for higher filter lengths due to smaller cycle period. Besides, the proposed structure supports MCM scheme when fixed-coefficient filters are implemented, whereas direct-form structure does not support MCM scheme. We have shown that the proposed structure offers both horizontal and vertical MCM, which can be exploited in the proposed structure to reduce the area complexity substantially further compared with the direct-form structure for the implementation of fixed filters.

Compared with the direct-form structure, the proposed structure for block size 4 involves 192, 448, 960, and 1984 more FFs and its cycle period less by TFA, 2TFA, 3TFA, and 4TFA for filter lengths 16, 32, 64, and 128, respectively. The proposed structure involves more number of FFs than the direct-form structure for higher filter lengths, but the excess area due to those FFs is very small compared with the total area of the direct-form structure.

TABLE III AREA DELAY COMPARISON OF PROPOSED AND EXISTING STRUCTURES FOR B=8 AND B=16

| METHOD NAME | AREA IN NUMBER OF LUT | | | MEMORY IN Kilobytes | DELAY(ns) |
|---|---|---|---|---|---|
| FILTER NAME | LUT | NO.OF SLICES | NO.OF FF | SIZE | ns |
| PROPOSED FIR FILTER Virtex-6 xc6vcx75t-2-ff784 | 154 | 110 | 54 | 283876 | 10.175 |
| PROPOSED ONE ON EXISTING FPGA KIT Virtex-5 xc5vsx95t-2-fff1136 | 552 | 102 | 28 | 301072 | 16.923 |

From above tabular form has shown the differences between the proposed and existing on proposed one. If we are observed above values, the number of LUTs required in proposed structure is less than the existing one. Despite more FFs, the proposed structure will have less area delay than the existing direct form structure. Because of the higher filter lengths, the proposed one involves more number of Flip-flips than the existing one. The area has been taken by Flip-flips are very less compared with the area of existing one. Finally, I have reduced the number of LUTs, less memory to store values and less time period compared with direct form structure.

## V. SIMULATION RESULTS

All the synthesis and simulation results of the Proposed High performance FIR Filters are performed using Verilog HDL. The synthesis and simulation are performed on Xilinx ISE 14.4. The corresponding simulation results of the Proposed High performance FIR Filters are shown below.

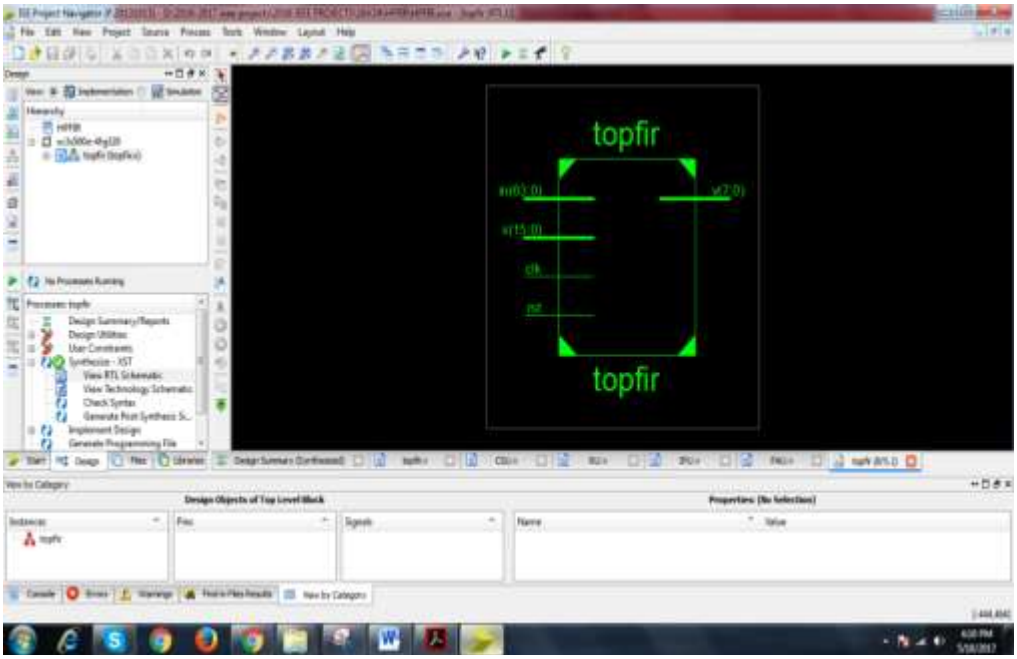



Fig.10 RTL schematic of Top-level of Proposed High performance FIR Filter

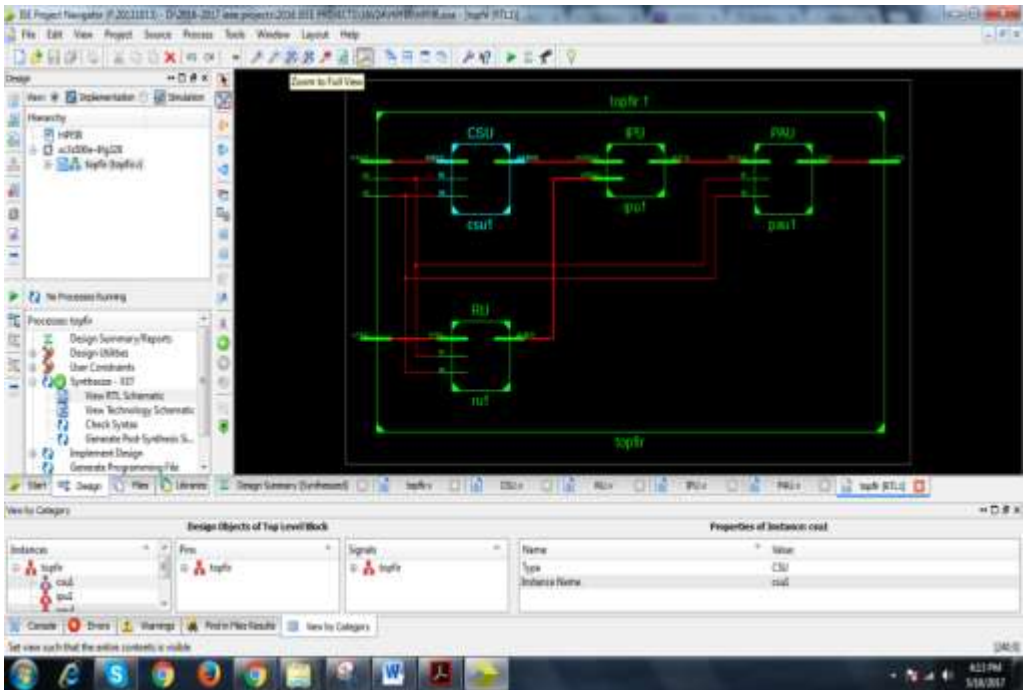



Fig.11 RTL schematic of Internal block of Proposed High performance FIR Filter

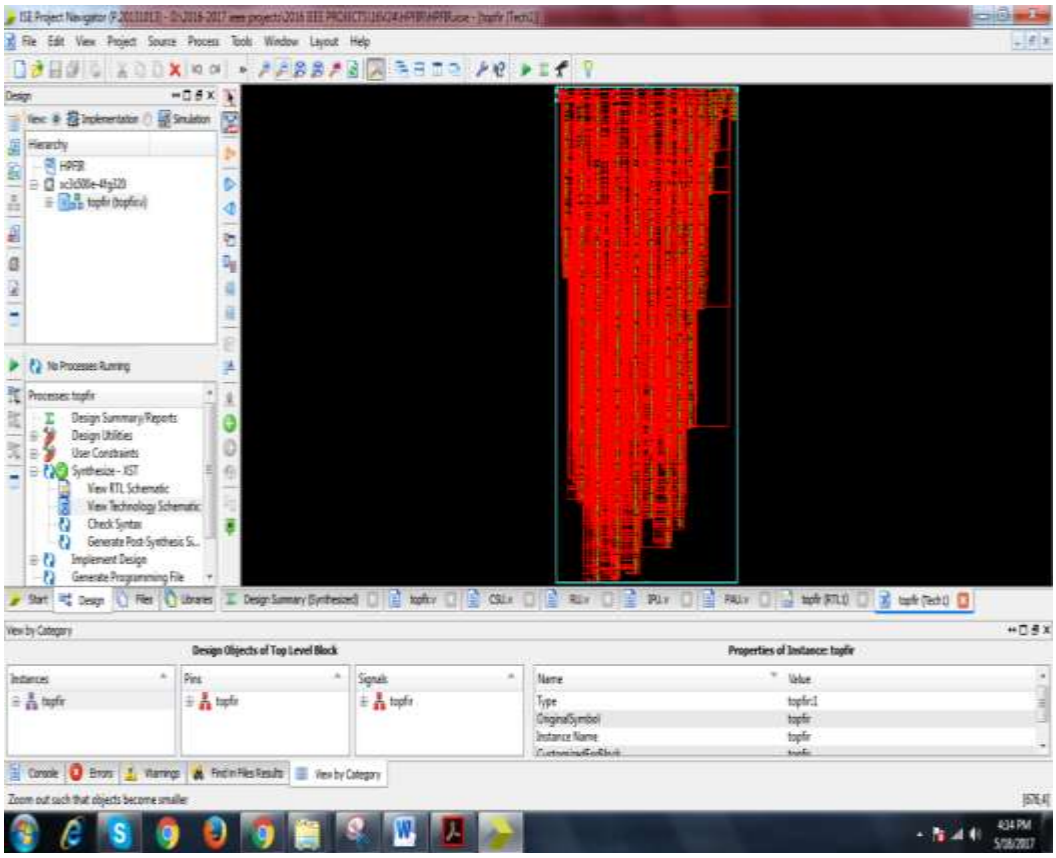


Fig.12 Technology schematic of Internal block of Proposed High performance FIR Filter

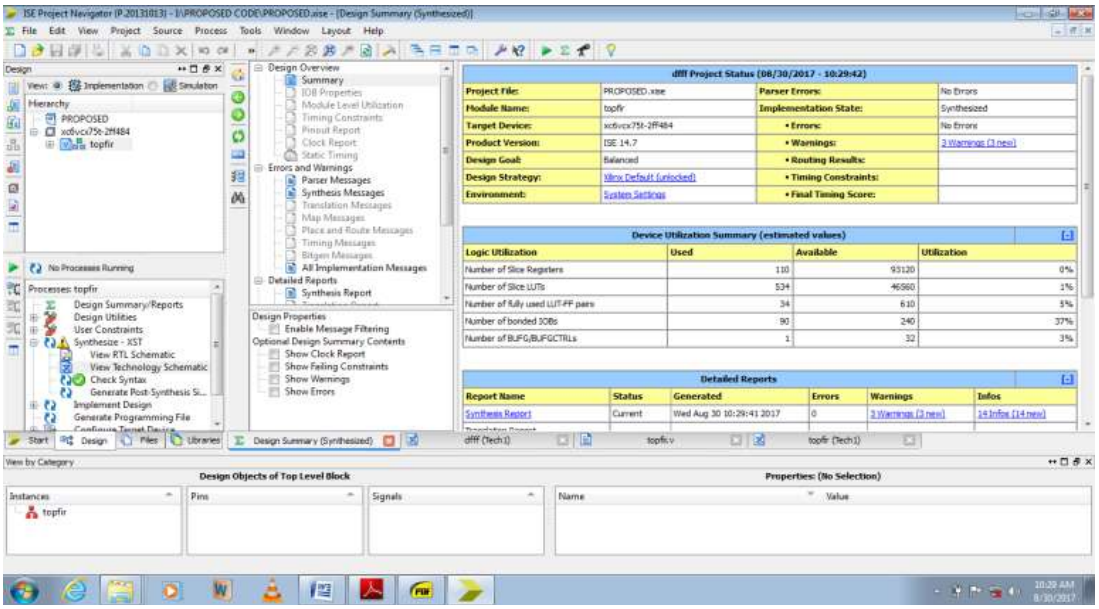


Fig.13 Synthesis report of Proposed High performance FIR Filter

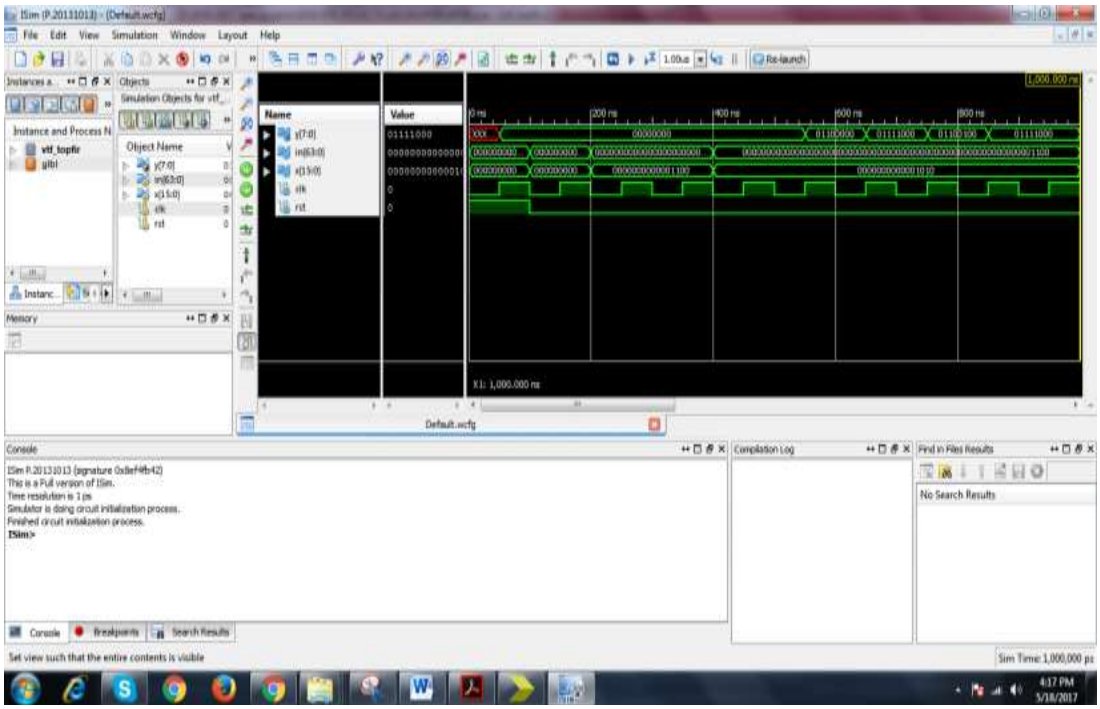


Fig.14 Simulated outputs for Proposed High performance FIR Filter

## VI. CONCLUSION

All the synthesis and simulation results of the Proposed High performance FIR Filters are performed on Xilinx ISE 14.7 using Verilog HDL. In this paper, we have explored the possibility of realization of block FIR filters in transpose form configuration for areadelay efficient realization of both fixed and reconfigurable applications. A generalized block formulation is presented for transpose form block FIR filter, and based on that we have derived transpose form block filter for reconfigurable applications. We have presented a scheme to identify the MCM blocks for horizontal and vertical subexpression elimination in the proposed block FIR filter for fixed coefficients to reduce the computational complexity.

**REFERENCES**

[1] J. G. Proakis and D. G. Manolakis, Digital Signal Processing: Principles, Algorithms and Applications. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.
[2] T. Hentschel and G. Fettweis, "Software radio receivers," in CDMA Techniques for Third Generation Mobile Systems. Dordrecht, The Netherlands: Kluwer, 1999, pp. 257–283.
[3] E. Mirchandani, R. L. Zinser, Jr., and J. B. Evans, "A new adaptive noise cancellation scheme in the presence of crosstalk [speech signals]," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 39, no. 10, pp. 681–694, Oct. 1995.
[4] D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in Proc. IEEE Southeastcon, Apr. 1993, p. 1–6.
[5] J. Mitola, Software Radio Architecture: Object-Oriented Approaches to Wireless Systems Engineering. New York, NY, USA: Wiley, 2000.
[6] 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.
[7] J. Park, W. Jeong, H. Mahmoodi-Meimand, Y. Wang, H. Choo, and K. 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.
[8] K.-H. Chen and T.-D.Chiueh, "A low-power digit-based reconfigurable FIR filter," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 8, pp. 617–621, Aug. 2006.
[9] R. Mahesh and A. P. Vinod, "New reconfigurable architectures for implementing FIR filters with low complexity," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 29, no. 2, pp. 275–288, Feb. 2010.
[10] S. Y. Park and P. K. Meher, "Efficient FPGA and ASIC realizations of a DA-based reconfigurable FIR digital filter," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 61, no. 7, pp. 511–515, Jul. 2014.
[11] 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.
[12] 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.
[13] P. K. Meher, "New approach to look-up-table design and memorybased realization of FIR digital filter," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 3, pp. 592–603, Mar. 2010.
[14] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. New York, NY, USA: Wiley, 1999.
[15] B. K. Mohanty and P. K. Meher, "A high-performance energy-efficient architecture for FIR adaptive filter based on new distributed arithmetic formulation of block LMS algorithm," IEEE Trans. Signal Process., vol. 61, no. 4, pp. 921–932, Feb. 2013.