Received: May 3, 2023. Revised: May 24, 2023. 

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

# Optimizing Parallel FIR Filter Architecture for Time-Sensitive Applications: A Design Approach for High-Throughput and Area Efficiency

**Kunjan D. Shinde[1]*** **Vijaya C[1]**

[1]*Research Centre, Department of Electronics & Communication Engineering, Shri Dharmasthala Manjunatheshwara College of Engineering & Technology, Dharwad, Karnataka, India*
* Corresponding author's Email: kunjan18m@gmail.com

**Abstract:** The advancement in very large scale integration (VLSI) technology and field programmable gate array's (FPGA) parallel constructive nature for digital circuits has made the implementation of finite impulse response (FIR) filters increasingly relevant in real-time. In FIR filters the computational complexity increases with the length of the filter; numerous techniques have been developed to design viable architectures for realizing FIR filters. The multiple input multiple output (MIMO) based parallel FIR filter architecture often requires a large area for realization as the level of parallelism and filter order increases; this study presents an architecturally enhanced, novel parallel architecture for FIR filter design which eliminates the resource dependencies on the level of parallelism. The proposed architecture addresses the issue by limiting the number of multipliers required irrespective of the change in the level of parallelism and further modifies the data flow to improve the iteration period in the proposed design. The level of parallelism (L) is fixed to 8 in the presented study as the value is neither too low nor high; it is important to note that the increase in the level of parallelism will increase the number of samples generated at a given time. The presented work is demonstrated using tap 16 FIR filter implemented on FPGA VIRTEX 4 xc4vsx35-10ff668 and VIRTEX 5 XC5VSX95T-1FF1136 platforms, and filter-order 16 FIR filter is implemented on ARTIX 7 xc7a200t-2fbg676. The functionality is verified using Xilinx ISE 14.7 and Matlab 2018 environments. The post-synthesis results and comparative study show the design validation; the proposed architecture outperforms the benchmark and conventional MIMO-based parallel FIR filter architectures with an improvement in total delay by 70% w.r.t. [2] and 93% w.r.t. [3] and 2% w.r.t. [4] as compared in case studies 1 – 3. The reduction in area metric is observed as dipping in the slice requirement by 31%, 47%, and 110% and dipping in the LUT requirement by 58%, 66%, and 65%, further, the delay metric is improved by 49%, 80%, and 67% when compared with MIMO-based parallel architecture of [1] on the same platforms suggested in [2-4] respectively.

**Keywords:** Digital filter, Parallel FIR filter, FIR filter on FPGA, FIR filter architecture, Architectures for digital filter, High-speed filter architecture, Area-throughput optimized parallel FIR filter, MIMO-based parallel FIR Filter.

## 1. Introduction

Signal processing is an important and crucial task that basically deals with extracting artifacts and information from the captured signals. The signals captured in real-time contain unwanted signals along with the required information; various noises are introduced at the time of signal acquisition from sensors and the signals also get corrupted while moving in the noisy medium. The primary phase in signal processing is filtering which is accomplished by the use of filters [1, 14]. Digital filters are used to retain the original information while suppressing unwanted interference and noise [5]. In digital signal processing, digital filters like Infinite Impulse Response (IIR) and FIR are often used to accomplish filtering tasks; FIR filter is of a finite impulse response period, which settles to zero in a finite amount of time, meanwhile, the IIR filters consist of internal feedback due to which the output responds indefinitely even in the absence of input and hence the FIR filters are more preferred [1-13, 14, 19-24]. The response of an Nth-order FIR filter takes an exact N+1 sample before it settles to zero [1,

5, 14]. The FIR filters are frequently used to pre-process signals of various kinds including multimedia (audio, image, video), bio-medical (ECG, EEG, EMG,.), digital communication (SDR, channel separation,.), seismic signals, and more. FIR filters can be specifically developed to process such signals, where the filter order and filter coefficients are designed to meet the requirements of a given end application. The filter architecture describes the connectivity of functional blocks of the filter which are essential for filter realization; the functional blocks are multiplier, delay unit, and adder.

The scope of this study is to estimate the impact of the proposed parallel FIR filter design with various optimized benchmarks FIR filter architectures developed on reconfigurable platforms and to validate the parallel processing with extensive hardware support. The proposed parallel FIR filter architecture eliminates the MIMO structure and serial-to-parallel unit which are essential in a conventional MIMO-based parallel FIR filter [1]. The proposed architecture uses constant multiplication and shift units to resemble the functionality; this modification enhances the speed by reducing the critical path and limits the multiplier requirement in the design. The multiplier dependency on the change in the level of parallelism is eliminated as no duplication of resources is required in the proposed design; further, with the use of symmetric coefficients, the multiplier requirement can be reduced by half the number of the multiplier as required originally. The impact of the proposed parallel FIR filter architecture is measured with conventional MIMO-based parallel architecture [1] and the benchmark existing studies [2-4] on a uniform FPGA platform under the same conditions as identified in the literature [2-4].

The following major concerns are addressed in the presented work:

- Optimization of resource requirements in MIMO-based parallel FIR filter.
- Improvement of throughput/iteration period of parallel FIR filter architecture.
- The change in the level of parallelism has no impact on the multiplier requirements, while the area dependencies observed in MIMO-based structure is simplified.

The rest of the paper is organized as in section 2 various related study on FIR filter architecture is discussed, in section 3 parallel architecture and its design with an example is discussed, in section 4 gives the details of the proposed parallel FIR filter architecture and its design with an example and the results are presented in section 5 with the experimental setup, tools used, comparative study and interpretations made.

## 2. Related work

The following study gives the FIR filter design using various benchmark architectures and the work carried out by various researchers for efficient FIR filter realization. The related studies are grouped based on the similarities in architectural enhancements, functional block optimizations, and related schemes for filter realization, as these aspects have been focused on in the presented study while designing a parallel FIR filter.

Various FIR filter architectures have been developed by researchers to address the issues related to FIR filter design [1-4], the major huddle in the design of FIR filter is the multiplier unit [5], as the FIR filter needs higher order for near ideal response and hence the computational complexity increases, thereby the number of multipliers required also increases [1, 14]. Multiplication is one of the fundamental operations in all DSP applications therefore extra care is to be taken while designing the FIR filter [3, 10]. In order to implement a digital filter for high-speed operation, the architecture and its internal blocks are to be optimized, the performance of the multiplier has to be enhanced or the multiplication operation has to be eliminated and replaced with an alternate methodology to improve the filter performance, such modifications may result in fast computation [1-4]. The lookup-table (LUT)-less modified distributed arithmetic (DA) architecture is used in [2] to create the FIR filter. DA is a popular method for FIR filter design. The multiplexer-adder design in [2] eliminates the necessity for pre-computing weighted sums for the LUTs. It is stated that the design consumes fewer resources and is quicker than other versions of DA. In contrast to traditional DA, the design of [2] can work even when the input range is extended. In [6] DA and off-set binary coding (OBC) are used to overcome the difficulties associated with MAC-based FIR filter design. OBC-DA based FIR filters use less hardware than DA owing to lower LUT requirements, but have a little delay due to the additional address generating overhead [6]. The DA unit in [12] does not use a lookup table (LUT). In comparison to the previous DA-based approach, it uses half the number of registers to hold the total of distinct combinations of input samples. The drawback of DA-based design is limitations imposed on the filter order, linear-phase

characteristics, and accuracy while translating the multiplier operation with DA schemes of [2, 6, 12].

As multipliers are the most expensive resource and power-intensive block, by quantizing the product bits, a multiplier's area-power complexity can be reduced [3, 10]. The least significant bit (LSB) in DSP systems is frequently truncated to achieve uniform word length throughout the system [3]. The Urdhva Tiryagbhyam-based vedic multiplier (VM) architecture is utilized to avoid calculating LSBs at each stage, resulting in a smaller footprint and lower power usage. In [10], the VD-CLA-FIR filter is built using the Vedic design algorithm with the carry look ahead (CLA) technique, the FIR filter's performance is effective in noise-free ECG signal extraction. The VM is constructed utilizing a carry-save adder in [7], which improves the multiplier's performance while also increasing its speed. The study in [8] investigates various multiplication approaches along with various adders for FIR filter realization. The FIR filter in [9] is formed using a VM and the urdhva tiryagbhyam method, here the adder unit is produced using an advanced addition approach and a comparative study gives that SQRT CSLA-based design is efficient in the research work conducted. The major concern of the study [3, 7-10] is block-level optimization i.e. multiplier block; these designs have bottlenecks of underlying filter architecture even though the functional block is improvised. The critical path of the FIR filter is from in several stages of multipliers and adders as expressed in [5].

A Time-division multiplexing (TDM)-based approach can be utilized to build a multi-channel FIR filter architecture; in [4,] a single multiplier and adder is employed regardless of the number of channels and taps. The notion of resource sharing is used to optimize logic resources. For optimization of resources based on look-up-tables, output product coding (OPC) and dual port schematic are employed [4]. The use of shared resources limits the computational power by underlying hardware and memory access based design, this may not be suitable for higher filter orders and high precession based architectures. Due to the splitting of filter in several stages for multichannel processing and the overhead associated in processing the signal may increase the design complexity. The study in [4] is developed for 8-bit precession and the same may not be suitable for higher word length.

The study on the multiplier and adders design is discussed in [15-17, 25], for DSP/ embedded application the improvements on these blocks may further improve to consider signed notations for real-time scenario based FIR filter design and hence the work is referred to understand the working nature of multipliers/adders. The work [15-17] explores Braun multipliers with various methods and parallel prefix adders to improve the systems performance metric. The information and details in [18-20] give the required theory and background knowledge on the various FPGA and its internal architectures, number representations with signed and CSD notations, and a basic understanding of how signals are processed.

The study in [21-24] gives the polyphase decomposition based parallel FIR filter architectures with improvements in functional blocks and architecture for higher value of subfilter [21]. These architectures require extra processing to compute subfilter coefficients while the decomposed values are to be realized with separate multipliers, this increases the complexity in the design and area constraints. Further, the latency and memory requirement is also increased due to the decomposition of filter to subfilter and recreating the output sample. These drawbacks limit the parallelism required for time-sensitive applications.

To summarize, the FIR filter and typical architectures referred to in the study are based on parallel MIMO-based FIR filter [1], DA-based design [2, 6, 11, 12], Vedic-multiplier based design [3, 10, 7-9, 21-24] uses a polyphase based design which limits the parallelism to a smaller value and other schemes like ISC [1], OPC [4]. It is important to note the study relating to architectural enhancements is the major key for filter optimization. This creates room for the following research questions.

- Can existing SISO-based FIR filters meet the demand for time critical/sensitive applications.
- Functional block-based optimization in FIR filters has an unavoidable bottleneck of underlying filter architecture.
- Duplication of resources/hardware is found in the MIMO-based parallel FIR filter. Is it possible to optimize area /resource utilization.

## 3. Parallel architectures and its design for FIR filter

The functionality of the FIR filter can be described in many ways, in the presented work we are using a difference equation to describe the FIR filter, using difference equation we can identify the details of the FIR filter’s arithmetic/functional blocks as required for VLSI platform. Filter architecture describes the connectivity of functional

blocks to realize the DSP algorithm and also captures data flow, critical path, and resource requirements for the same. The functional blocks of the FIR filter are delay, adder, and multiplier units, these are arranged in such a way that the resulting interconnections formulate the working nature of the FIR filter and the way in which they are connected together is called filter architectures. FIR filter described using difference equation is represented in Eqs. (1) and (2) gives the elaborated view of $N^{th}$ order (T=N+1 tap) FIR filter [1, 5].

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) \quad (1)$$

Expanding the above benchmark equation,

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) + \cdots b_{N-1} x(n-N-1) \quad [1,5] \quad (2)$$

Where y(n) is output sample, N is filter order, $b_k$ is filter coefficients and x(n) is input sample.

A Nth order filter has T =N+1 taps or its length is N+1, which results in N+1 multiplication ((N+1)/2 multiplication in case of symmetric coefficients) and N additions ((N+1)/2 addition in case of symmetric coefficients) and N delay units [5]. The use of symmetric coefficients can reduce the FIR filter resource requirements to a greater extent, but the choice of filter coefficients being asymmetric or symmetric is up to the designer. It is important to note that the filter order is an end-application dependent variable (i.e. the filter order required to realize certain processing is dependent on the type of signal being processed). Thereby the increase in filter order will increase the demand for resources required to realize the filtering operation and increases the complexity to meet the design constraints.

### 3.1 Parallel FIR filter architecture

The study in this section addresses the MIMO-based parallel FIR filter architecture [1] and its design. Eq. 1 gives the serial input serial output (SISO) based architecture for the FIR filter; this must be transformed into MIMO-based architecture which is also known as parallel processing. In such parallel FIR filter architecture, multiple samples are processed in a single clock cycle by the use of duplicated hardware, this type of processing is also known as block processing where the level of parallel processing is the block size (L) which indicates the number of hardware copies created [1]. The parallel architecture is often disregarded due to the area complexity. The constructive nature of the parallel architecture provides a solution to the power and speed requirements of the FIR filter for devices operating on battery and demanding higher throughput as a performance metric. Fig. 1 gives the block diagram of a parallel FIR filter, the complete filtering operation with parallel processing in the FIR filter is designed by the use of serial to parallel and parallel to serial units to feed parallel data in and out of the MIMO unit. The critical path through a conventional parallel architecture can be computed using Eq. (3) [1].

$$T_{parallel} = \frac{1}{L} T_{clk} \geq \frac{1}{L} T_{Critical\_Path} \quad (3)$$

where $T_{parallel}$ is the computation time interval for parallel architecture/ iteration period,

$T_{clk}$ is clock period,

$L$ is level of parallelism/ block size

$T_{Critical\_Path}$ is maximum possible critical path in parallel FIR Filter

In such systems it is important to note that the $T_{parallel} \neq T_{clk}$ as multiple outputs are computed in a single clock cycle.

Parallel processing architectures compute multiple outputs at a single clock cycle and hence the iteration period of the system is $\mathbf{T_{parallel}}$ which is not the same as the systems clock period [1], thereby the effective processing speed is increased by the level of parallelism (L) [1, 5]. The architectures for FIR filters are SISO design based whereas, in parallel FIR filter architectures are MIMO design based. The related work and conventional systems are often SISO, in some cases parallel architecture-based design uses a polyphase methodology [21-24], the level of parallelism may not be a flexible parameter along with the filter coefficients being decentralized with the certain process and this

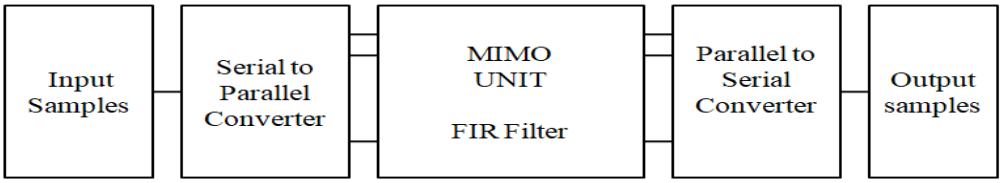



Figure. 1 Block diagram of the conventional parallel architecture for FIR filter [1]

increases the functional block requirements. The use of the MIMO unit may create an opportunity to explore the possible solutions which improve the performance of parallel architecture while the level of parallelism can be easily customized.

Applications with time-bound demand are critical systems that can make good use of parallel architecture to meet the high computational demand without upgrading/changing the underlying hardware. It is worth noting that the parallel filter architecture design becomes faster by L times as the L number of samples are processed in a single clock cycle while the conventional architecture can produce only one sample in a similar time period.

### 3.2 Case study – tap 3 parallel FIR filter designed using a block size(L) of 3

This section provides a detailed perspective on the realization of parallel FIR filter and to understand the origin of the proposed work the data flow within such system is to be noted. Consider a tap 3 FIR filter with a block size of three, Eq. (4) represents the SISO system for the tap 3 FIR filter which can also be treated as a parallel filter with a block size of one and the implementation structure looks the same as tap 3 FIR filter with direct form I architecture. It is observed that this architecture uses 3 multipliers and 2 adders to obtain the output sample as filter results. The critical path is equal to $\mathbf{T_M+2T_A}$ [5], where $\mathbf{T_M}$ is the delay through the multiplier unit and $\mathbf{T_A}$ is the delay through the adder unit.

$$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) \quad (4)$$

Expanding the Eq. (4) of tap 3 FIR filter with a block size of three, the Eqs. (5)-(7) represents the parallel architecture for tap 3 FIR filter with a block size of three.

$$y(3k) = b_0x(3k) + b_1x(3k-1) + b_2x(3k-2) \quad (5)$$

$$y(3k+1) = b_0x(3k+1) + b_1x(3k) + b_2x(3k-1) \quad (6)$$

$$y(3k+2) = b_0x(3k+2) + b_1x(3k+1) + b\_2\,x(3k) \quad (7)$$

The samples represented are discrete time samples and the notations 3k, 3k+1 and 3k+2 represents the input samples put through the serial to a parallel unit, where *x(3k+2)* is the latest input sample, *x(3k+1)* is the one unit delayed version of input sample and *x(3k)* is the two-unit delayed version of input sample, the '*k'* denotes the sample cycle and at k[th] clock cycle the 3 inputs are processed and 3 outputs are generated as indicated in Table 1. The input samples *x(3k), x(3k+1),* and *x(3k+2)* are fed into the MIMO unit where the computations are carried out and respective output samples *y(3k), y(3k+1),* and *y(3k+2)* are obtained at k[th] cycle. The internal details of the MIMO unit are shown in Fig. 2 b. It may be noted that the sampling interval is $\mathbf{T_{clk}/L}$.

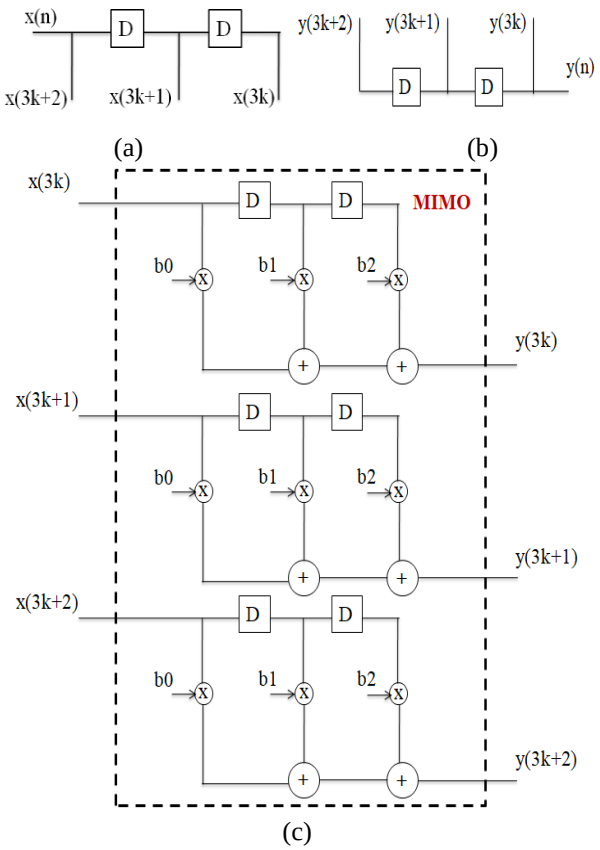



Figure. 2 Parallel FIR filter: (a) Serial to a parallel unit, (b) Parallel to serial unit, and (c) MIMO unit of tap 3 parallel FIR filter L= 3 [1]

Now, let's look in detail and understand the data flow within a parallel architecture while the focus on input sample, filter coefficient, output sample and timeline of the samples produced is to be noted. The tap 3 parallel FIR filter architecture is developed using Eqs. (5)-(7), Table 2 illustrates the data flow of samples in a parallel architecture.

It is important to note the data flow of the samples with different timelines; The input sample is multiplied with filter coefficients which are indicated column-wise and the summation of these inner products is carried out row-wise to generate the parallel output. The timeline (T0, T1, …) is processing and generating three samples at a time as the level of parallelism(L) is 3. Let's focus on the

Table 1. Samples processed in Parallel FIR filter architecture

| $k^{th}$ Sample cycle | Input sample being processed are x(3k), x(3k+1), x(3k+2) | Output sample being produced are y(3k), y(3k+1), y(3k+2) |
|---|---|---|
| **0** | x(0), x(1), x(2) | y(0), y(1), y(2) |
| **1** | x(3), x(4), x(5) | y(3), y(4), y(5) |
| **2** | x(6), x(7), x(8) | y(6), y(7), y(8) |
| **…** | … | … |

Table 2. A generalized view of data flow within tap 3 parallel FIR filter architecture with (block size) L=3

| Time line | Input sample | Filter Coefficients being multiplied column-wise with input sample and summed row-wise for output | | | Output samples Computed |
|---|---|---|---|---|---|
| | | $b_0$ | $b_1$ | $b_2$ | |
| **T0** | x(3k) | x(3k) | x(3k-1) | x(3k-2) | y(3k) |
| | x(3k+1) | x(3k+1) | x(3k) | x(3k-1) | y(3k+1) |
| | x(3k+2) | x(3k+2) | x(3k+1) | x(3k) | y(3k+2) |
| **T1** | x(3k+3) | x(3k+3) | x(3k+2) | x(3k+1) | y(3k+3) |
| | x(3k+4) | x(3k+4) | x(3k+3) | x(3k+2) | y(3k+4) |
| | x(3k+5) | x(3k+5) | x(3k+4) | x(3k+3) | y(3k+5) |

input samples with the common notations and it is observed that the input sample of some instances is being multiplied by the different filter coefficients in the MIMO block (as highlighted in Table 2 with the same color). For example, the input sample x(3k) is getting multiplied with all the filter coefficients (b0,b1,b2) at different MIMO blocks (i.e. x(3k) multiplied by b0 in 1st MIMO unit, x(3k) multiplied by b1 in 2nd MIMO unit, x(3k) multiplied by b2 in 3rd MIMO unit,) at a different instant. This identified task is effectively utilized in the proposed parallel FIR filter by clubbing this individual multiplication of filter coefficients in different MIMO units into a single structure. Hence the resultant organization eliminates the MIMO structure as presented in the proposed work.

## 4. Proposed parallel FIR filter architecture and its design

With reference to the data flow in conventional MIMO-based parallel FIR filter [1], it is observed that the MIMO units are obtained by replicating the original block by '**L**' a number of times and it is important to note that the same set of operations like input samples being multiplied by filter coefficients are also repeated by '**L**' times. Further, to eliminate the repeated task and save the resource requirements of parallel FIR filter the proposed work is enforced.

In line with the data flow of input samples observed in Table 2, this creates a scope for optimization as the given input sample at an instance is being multiplied with different filter coefficients in relatively different MIMO units. This constructive flow can be brought together as a constant multiplication and saving the product terms for later use has effectively eliminated the conventional MIMO structure observed in parallel FIR filters. The new arrangements of the input samples and filter coefficients have reduced the multiplier requirements to a greater extent and improvement in the critical path can also be observed when compared with conventional parallel FIR filter with MIMO unit. In the presented work, the architectural improvements and reduction of resources are focused on.

### 4.1 Proposed architecture

The design and development of the proposed parallel FIR filter architecture is modeled to operate at higher throughput and with fewer area requirements when compared with the conventional parallel architecture. The block diagram of the proposed parallel FIR filter architecture is given in Fig. 3, the highlighted block in the diagram gives the novelty of the proposed work.

The proposed architecture for parallel FIR filter consists of three major units as indicated below

i. Constant multiplication with filter coefficients
ii. Shift units and
iii. Adder tree for parallel structure.

The proposed parallel FIR filter architecture uses constant multiplication and shift units to functionally replicate the MIMO and serial-to-

Received: May 3, 2023. Revised: May 24, 2023. 

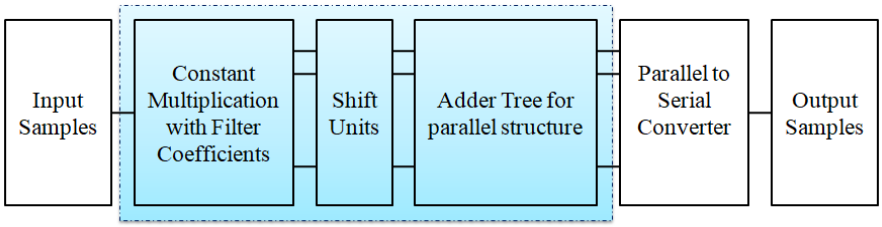



Figure. 3 Block diagram of proposed parallel FIR filter architecture

Table. 3 Resource requirements of FIR filter using various architectures

| Parameters/ Filter Architectures | | | Asymmetric coefficients | | Symmetric coefficients | |
|---|---|---|---|---|---|---|
| | | | No. of Multipliers required | No. of Adders required | No. of Multipliers required | No. of Adders required |
| **Case Study: tap 3 FIR filter resource requirements** | | | | | | |
| **Direct Form** | | | 3 | 2 | 2 | 2 |
| **Transposed Form** | | | 3 | 2 | 2 | 2 |
| **Pipelined** | | | 3 | 2 | 2 | 2 |
| **Conventional MIMO-based Parallel Architecture [1]** | **Block Size (L)** | 1 | 3 | 2 | 2 | 2 |
| | | 2 | 6 | 4 | 4 | 4 |
| | | 3 | 9 | 6 | 6 | 6 |
| **Proposed parallel** | | **1** | **3** | **2** | **2** | **2** |
| | | **2** | **3** | **4** | **2** | **4** |
| | | **3** | **3** | **6** | **2** | **6** |
| **$N^{th}$ Order FIR filter resource requirements for L level of parallelism** | | | | | | |
| **Conventional MIMO-based Parallel Architecture [1]** | | | **(N+1)× L** | **N×L** | **((N+1)/2) ×L** | **((N+1)/2) ×L** |
| **Proposed parallel Architecture** | | | **N+1** | **N×L** | **(N+1)/ 2** | **((N+1)/ 2) ×L** |

parallel unit observed in the parallel FIR filter. The proposed architecture and highlighted blocks together form a high-speed and area-efficient design to realize a parallel FIR filter. It can be noted that only one set of multipliers is required in the design of the proposed parallel architecture which is estimated to be equal to the multiplier requirement in direct form FIR filter architecture irrespective of the change in the level of parallelism in the proposed work. As the multiplier is one of the critical blocks in the DSP system and hence its design and optimization are to be taken on priority bases. The reduction of multiplier requirements in parallel architecture is a major advantage for FIR filter design.

The critical path of the proposed architecture is given by Eq. (8) and thereby the overall delay/ iteration time (speed metrics) is given by $\mathbf{T_{parallell\_proposed}}$ in Eq. (9) which is computed in line with Eq. (3). The shift unit's number of stages is calculated using Eq. (10).

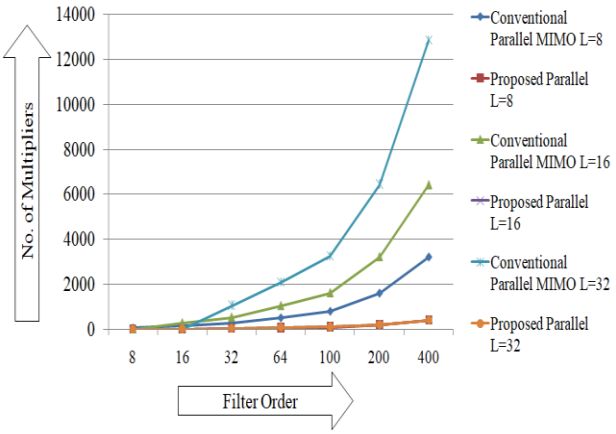
$$T_{Critical\_path\_proposed} = T_{Constant\_mutliplication\_unit} + T_{shift\_unit} + T_{adder\_tree\_unit} \quad (8)$$

$$T_{parallel\_proposed} = \frac{1}{L} T_{clk} \geq \frac{1}{L} T_{Critical\_path\_proposed} \quad (9)$$

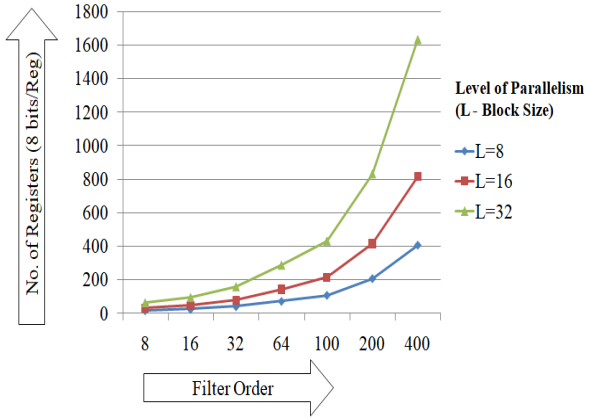
$$L_{shift\_unit} = T+L-1 \quad or \quad N+L \quad (10)$$

To estimate the resource requirements of the proposed architecture Table 3 is used. In Table 3, the FIR filter's resource requirements are segregated in line with the filter coefficients being asymmetric or symmetric. The use of symmetric filter coefficients reduces the multiplier requirements by half the original need for any given filter architecture.

From Table 3, we can observe an interesting impact on the design which is on the requirement of multiplier blocks, for **$N^{th}$** order/ **T** tap parallel FIR filter with **L** level of parallelism, the multiplier required are **((N+1)×L) or (T×L)** and for the proposed architecture it needs only **(N+1) or T** number of multipliers; as the filter order and level of parallelism have a direct impact on the multiplier requirements. In the proposed architecture, the input sample is getting multiplied with all the filter coefficients, this kind of multiplication resembles constant multiplication. Efficient realization of such block may further improve the performance of the said system.




(a)




(b)

Figure. 4 (a) Multiplier resource requirements in conventional and proposed parallel FIR filter architecture (b) Register requirements for shift unit in proposed parallel FIR filter architecture

To estimate the multiplier resource required for parallel FIR filters with various filter orders and different level of parallelism, the calculations of Table 3 is considered and Fig. 4 (a) is developed. It is can be noted from the Fig. 4 (a) the conventional parallel FIR filter requires a large number of multipliers and as the filter order increases the growth is exponential; whereas, the proposed parallel FIR filter has a constant requirement on the multipliers while the level of parallelism has no impact at all. With the proposed approach for the design of parallel FIR filters, the designers can efficiently develop filters with the least number of multipliers while having a higher level of parallelism for higher throughput.

Fig. 4 (b) gives the requirement of register in the shift unit of the proposed architecture. It can be noted that the precession of the signal under study have to be efficiently modeled as the proposed architecture requires a large number of register to store the products from the constant multiplication stage. This unit has a direct dependency on filter

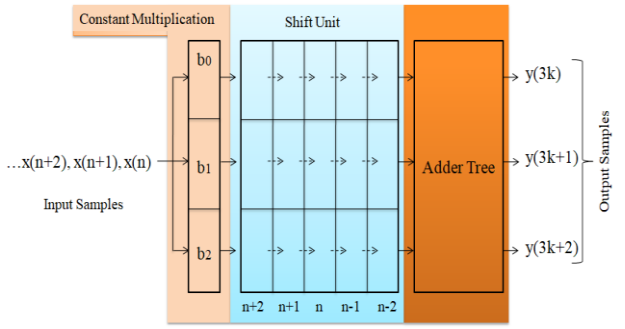



Figure. 5 Data flow of the proposed Tap 3 parallel FIR filter architecture

order as well as on the level of parallelism as estimated in Eq. (10). The Fig. 4 (b) indicates the register requirement in terms of registers length being 8-bit and with change in the level of parallelism and filter order we can see an increase in registers requirements.

### 4.2 Case study: Proposed architecture - tap-3 parallel FIR filter using a block size of 3

The proposed architecture optimizes the uses of repeated multiplications observed in a conventional parallel architecture. This is accomplished by the use constant multiplication unit where all the filter coefficients are multiplied with the arriving input sample. The product terms from this stage are stored in the shift unit and with the occurrence of $\mathbf{T_{clk}}$ the content within the shift unit is shifted. These shifted product terms are added according to the adder tree for generating the final output of the parallel FIR filter as illustrated in Fig. 5.

Fig. 5 gives the data flow diagram and internal block details of the proposed parallel FIR filter architecture with tap 3 FIR filter and level of parallelism set to 3, from the above structure we can note that the constant multiplication uses only 3 multipliers (2 in case of symmetric coefficients) and the length shift unit registers is 5 (where N=2, L=3, and T=3) stages which can be computed using Eq. (10). The adder tree uses the shifted versions of product terms to generate the parallel outputs in the given iteration.

## 5. Results & discussions

This section provides details on the experimental setup adopted, tools used to validate the presented study, and comparative analysis with conventional MIMO-based parallel FIR filter and other benchmarks FIR filters architectures. The functional verification of the parallel FIR filter design is carried out with a common set of numerical samples applied to the filters and it is noted that the common

set of the same output samples was produced.

To simplify the comparative study with other FIR filter architectures, the proposed parallel FIR filter is designed with a level of parallelism (L) set to 8, as in section 4.1 it is clearly mentioned that the increase in the level of parallelism will increase the throughput of parallel architecture and in current section the focus on the impact of various filter architectures is estimated with proposed work. The impact of variation in filter order with a level of parallelism is depicted in fig.4.1 and 4.2 and hence for the analysis purpose L=8 is considered for parallel architectures presented in this study.

### 5.1 Experimentation setup

In the presented work, the FIR filter is designed with the specification mentioned in literature [2-4] which are benchmark designs of FIR filters with different architectures performing filtering action for various end-applications. The experimental setup of [2-4] is referred and the case studies are presented below. The parallel FIR filter design of [1] uses the MIMO unit for parallel structure realization, while the choice on the use of level of parallelism is up to the designer hence as an ideal case we have chosen L=8, as this value is neither low nor high for all the three case studies considered. Further, the impact of change in level of parallelism and filter order in general is depicted in Fig. 4 (a) and (b). The structure and design of [1] are considered for comparing conventional parallel architecture with the proposed work.

#### 5.1.1. Case study 1: Related study [1, 2, and 6] is considered:

The work of [2, 6] is developed on Virtex-4 sx FPGA which is an optimized platform for high-performance DSP, the device selects the most favorable mix of resources for the respective application. Virtex 4 devices are formed on 90nm process technology delivered on 300mm wafers.

In line with the research work of [2, 6], the DA and modified DA-based FIR filter architecture was used as a benchmark study case, as the study reflects its area efficiency but the delay involved in processing the samples is very large.

| ***FIR Filter Specifications [2]:*** | |
|---|---|
| Filter type | : Low pass filter |
| Filter tap | : 16 |
| Filter coefficients | : Windowing – Hamming |
| Cut off frequency | : 10 Hz |
| Sampling frequency | : 100 Hz |
| FPGA platform | : VIRTEX 4 xc4vsx35-10ff668 |

The precession of samples is defined in [2] is considered in this case study, as the input signal x[n] is represented in 16-bit, filter coefficients h[k] is represented in 24 bit and the filter output y[n] is represented in 24-bit [2]. The presented work uses 16-bit signed notation is used for both input x[n] and filter coefficient h[k] representation and a full precision of 32 bits output samples is used in the presented work.

#### 5.1.2. Case study 2: Related study [1, 3, and 7-10] is considered:

The work of [3, 7-10] is developed to Artix 7 FPGA which gives a cost-optimized performance in logic, signal processing, and more. The Artix 7 FPGA is ideal for cost susceptible applications that need high closing stage features; it is formed on 28nm process technology.

In line with the research work [1, 3, 7-10] carried out the word length of the input is set to 16 bits [3]. The study in [3] focuses on an area and power optimization using Vedic mathematics, in [7] the FIR is designed with multiplier build using Vedic sutra, and its resulting inner products are summed using carry save adder, whereas in [8] the impact of various multipliers designed using parallel adders is measured and tabulated, the best result is picked out their work for the reference. It is to be noted that the area constraints in [3] have a good improvement to the study of [7, 8], and also the delay is reduced to a certain extent. In [9, 10] the work focuses on the improvement in the FIR filter by upgrading the internal blocks of the filter like MAC/ multiplier/ adder, some improvements in the area are observed but no major changes in the speed metric are noted. The work in [3, 7-10] focuses on improvements in FIR filter design with functional block optimization.

| ***FIR Filter Specifications [3]:*** | |
|---|---|
| Filter type | : Low pass filter |
| Filter tap | : 17 (Filter order 16) |
| Filter coefficients | : Windowing – Blackman |
| Cut off frequency | : 50 Hz |
| Sampling frequency | : 400 Hz |
| FPGA platform | : ARTIX 7 xc7a200t-2fbg676 |

The precession of samples is defined in [3] is considered in this case study, as the input signal x[n] is 16 bit, the study uses filter coefficients which are

quantized to a 16-bit binary representation. The presented work uses 16-bit signed notation is used for both input x[n] and filter coefficient h[k] representation and a full precision of 32 bits output samples is used in the presented work.

#### 5.1.3. Case study 3: Related study [1, 4, and 11-13] is considered:

The work of [4, 11-13] is developed on Virtex-5 SXT FPGA which is often used for DSP applications, the architecture uses new six-input LUTs, and it is produced on 65 nm process technology.

In line with the research work of [1, 4, 11-13], the study in [4] focuses on the multichannel FIR filter architecture while the core is single channel FIR filters; the design uses a conventional FIR filter with OPC and dual port memory based LUT. OPC-based design performs well as observed in [4]. In [11, 12] FPGA based analysis is carried out where the role of FIR filter is of major concern. The work in [13] uses a logic core IP of the FIR filter referred to in [4] for the comparative study and the same is carry-forwarded in the presented work. The work in [4] is performed with a word length of 8 bits; to measure the impact of the proposed architecture the requirements stated in [4] are adopted.

***FIR Filter Specifications [4]:***

| | |
|---|---|
| Filter type | : Low pass filter |
| Filter tap | : 16 |
| Filter coefficients | : Windowing – Hamming |
| Cut off frequency | : 10 Hz |
| Sampling frequency | : 100 Hz |
| FPGA platform : | VIRTEX 5 XC5VSX95T-1FF1136 |

The precession of samples is defined in [4] is considered in this case study, tap 16 single channel FIR filter is considered. The presented work uses 8-bit signed notation is used for both input x[n] and filter coefficient h[k] representation and a full precision of 16 bits output samples is used in the presented work.

### 5.2 Tools used

The presented research work is carried out on Xilinx 14.7 version for synthesis and simulations with the Xilinx ISE environment. The tap 16 filters of [2, 4] & filter order 16 of [3] is considered in the presented study, the FIR filter is implemented on the FPGA platform as VIRTEX 4 xc4vsx35-10ff668 in [2], ARTIX 7 xc7a200t-2fbg676 in [3] and VIRTEX 5 XC5VSX95T-1FF1136 in [4]. The filter specifications mentioned in [2-4] are used as case studies mentioned in section 5.1. To generate the filter coefficients in Matlab 2018 is used with filterDesigner command. The rest of the synthesis and implementation details are kept the same as given in the related work [2-4]. The results obtained on individual FPGA platforms are tabulated in Table 4-6 respectively.

### 5.3 Comparative analysis

This section presents the comparative analysis of various benchmark FIR filter architectures, the study is developed on the FPGA platform to moderate the parallel architecture on real-time bases. The performance metric like the area is estimated based on the utilization of No. of Slices, LUT, BRAM, and more; delay is estimated with the post-synthesis implementation of the design on the targeted FPGA, and the total delay is subdivided into logic and route delays. The resource utilization of conventional parallel MIMO based [1], proposed parallel and related studies [2-4] are aligned to the same platform and performance metric.

**Case study 1:** Table 4 gives the comparative details on the DA based [2, 6], conventional parallel MIMO-based (L=8) [1], and proposed parallel (L=8) FIR filter architectures. From fig.6 and 7, it is observed that the proposed parallel FIR filter architecture realized on the same platform consumes 31% lesser No. of slices and 58% lesser No. of LUTs when compared with [1]. The delay metric is improved by 70% when compared with [2] and 49% improvement when compared with [1].

It is observed that the CSD-based variant is consuming 33% lesser No. of slices and 52% lesser No. of LUTs when compared with [1] and it is observed that, this was consuming 2% lesser No. of slices and consumes 13% more No. of LUTs when compared with the original proposed architecture. The delay metric is further improved 51% using CSD-based design when compared with the original proposed architecture.

The requirement on multipliers is to be observed, as in the case of conventional parallel architecture the design requires 64 multipliers, whereas the proposed architecture is realized with only 8 multipliers irrespective of the change in the level of parallelism is the same for both parallel architectures (i.e. L=8) and in CSD based design 0 multipliers are used.

**Case study 2:** Table 5 gives the comparative details on the Vedic multiplier based [3, 7-10], conventional parallel MIMO-based (L=8) [1], and proposed parallel (L=8) FIR filter architectures. From Fig. 8 and 9 it is observed that the proposed parallel FIR

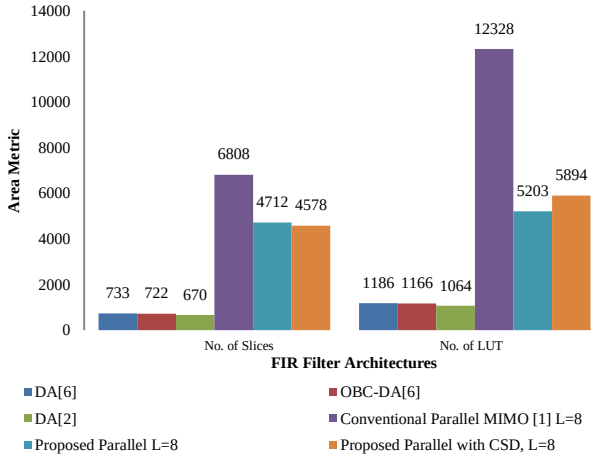



Figure. 6 Graphical representation of area unitization in various FIR filter architectures on FPGA [1, 2, 6]

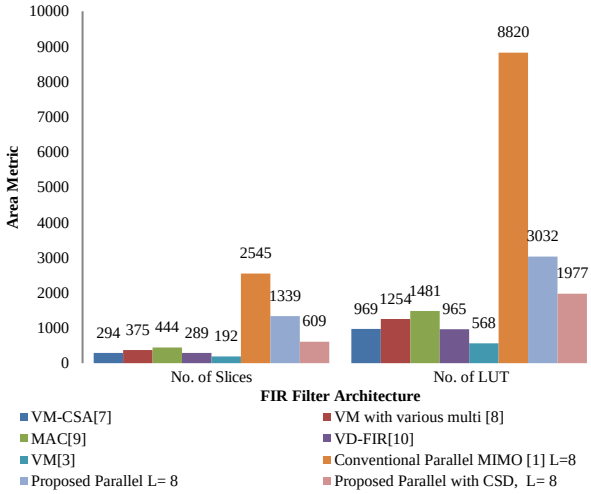



Figure. 8 Graphical representation of area unitization in various FIR filter architectures on FPGA [1, 3, 7-10]

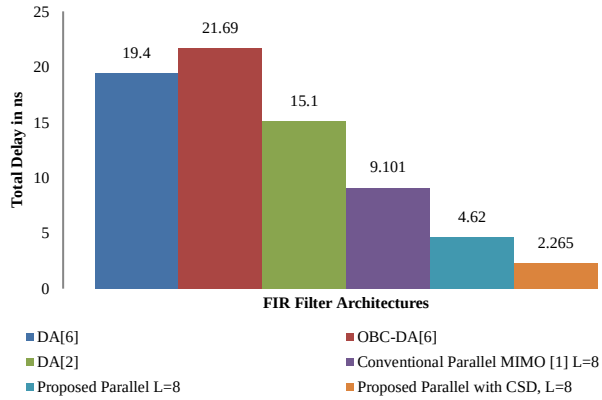



Figure. 7 Graphical representation of delay metric in various FIR filter architectures on FPGA [1, 2, 6]

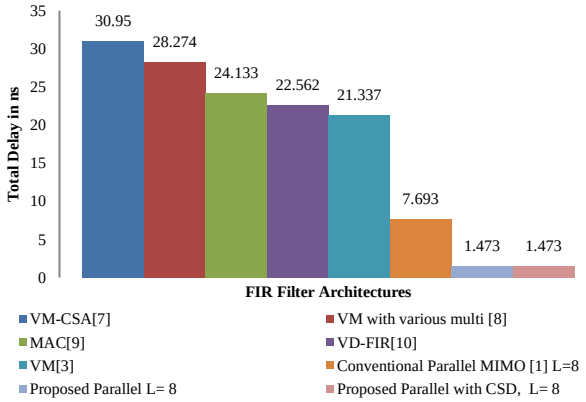



Figure. 9 Graphical representation of delay metric in various FIR filter architectures on FPGA [1, 3, 7-10]

filter architecture realized on the same platform consumes 47% lesser No. of slices and 66% lesser No. of LUT's when compared with [1]. The delay metric is improved by 93% when compared with [3] and 80% improvement when compared with [1].

It was observed that the new variant was consuming 76% lesser No. of slices and 77% lesser No. of LUTs when compared with [1] and was consuming 54% lesser No. of slices and 34% lesser No. of LUTs, and no improvement in the delay is observed when compared with the original proposed architecture.

The requirement on multipliers is to be observed, as in the case of conventional parallel architecture the design requires 72 multipliers, whereas the proposed architecture is realized with only 9 multipliers while the level of parallelism is the same for both parallel architectures (i.e. L=8) and in CSD based design 0 multipliers are used.

**Case study 3:** Table 6 gives a detailed analysis of the DA based [4,11-13], conventional parallel MIMO-based (L=8) [1], and proposed parallel (L=8) FIR filter architectures. It is observed that the proposed parallel FIR filter architecture realized on the same platform consumes 110% lesser No. of slices (NOS), 28% lesser SREG, and 65% lesser No. of SLUT's when compared with [1]. The delay metric is improved by 2% when compared with [4] and 67% improvement when compared with [1].

It was observed that the new variant was consuming 63% lesser No. of slices, 73% lesser SREG, and 80% lesser No. of SLUT's when compared with [1] and was consuming 22% lesser No. of slices, 62% lesser SREG, and 44% lesser No. of SLUT's when compared with the original proposed architecture.

The requirement on multipliers is to be observed, as in the case of conventional parallel architecture the design requires 64 multipliers, whereas the proposed architecture is realized with only 8 multipliers while the level of parallelism is the same for both parallel architectures (i.e. L=8) and in CSD based design 0 multipliers are used.

As observed in case study 1 and 3; which uses the DA-based FIR filter design, the performance of the designed system is limited to either functional

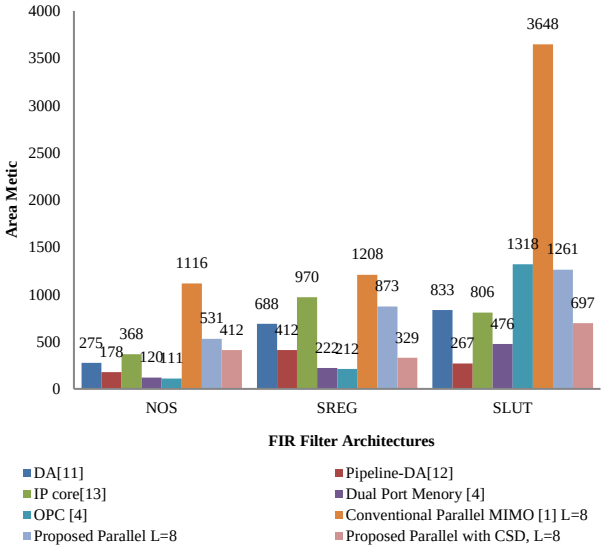



Figure. 10 Graphical representation of area unitization in various FIR filter architectures on FPGA [1, 4, 11-13]

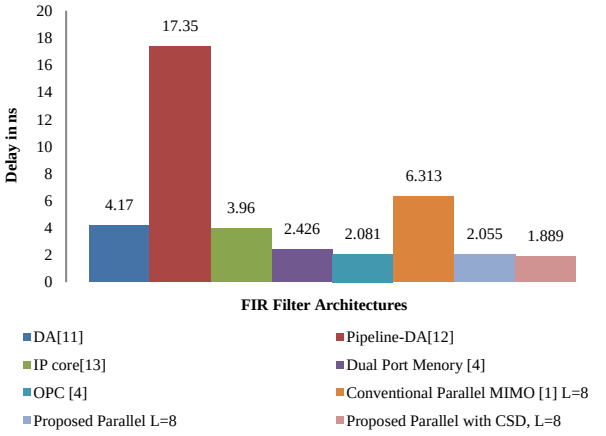



Figure. 11 Graphical representation of delay metric in various FIR filter architectures on FPGA [1, 4, 11-13]

block simplification or adopting to hybrid scheme on DA, and a related scheme to meet the computational demand. In case study 2, the optimization on the multiplier block with various considerations on a multiplier simplification using Vedic algorithm, MAC and more are provided while no improvement in the underlying architecture. Such a design approach would yield to the inefficient realization of FIR filters with high throughput functional blocks for time-sensitive end-applications. The study on MIMO-based parallel FIR filter architectures was also undertaken for the structure-based impact analysis and the requirement on multipliers is observed. The presented comparative study has given insights into the improvements needed at the architectural level while the related work might not address the demand of time-critical applications. The proposed parallel architecture for the FIR filter shows improvement in performance metrics which are achieved without functional block optimization at any level and the enhancement is exclusively due to the proposed architecture. A tentative variant of the proposed architecture with CSD for constant multiplication uses zero multipliers as the multiplication operations are replicated with a series of shifts and add operations with CSD representation. Note that the conventional parallel and proposed architecture uses symmetric filter coefficients.

### 5.4 Comparative tables

The following Tables 4-6 gives the detailed tabulation of case studies 1-3 respectively; the comparison is performed under the same conditions as mentioned case studies 1-3 described in section 5.1 with filter design details as considered in [2-4] respectively.

## Conclusion

The presented study is structured on FIR filter architectures and their impact on the filter's performance. The proposed parallel FIR filter architecture addresses the issue of area requirements by eliminating the MIMO unit and replacing it with a functionally equivalent constant multiplier unit with a shift unit as indicated. It is observed that a new structure for parallel architecture uses a fixed number of multipliers irrespective of change in the level of parallelism, this enhances the architecture and optimizes the area- throughput and resource requirement for the parallel FIR filters. The study is developed on FPGA platforms to efficiently validate the parallelism found in such DSP algorithms. The comparative tables demonstrate the effectiveness of the proposed architecture with case studies [2-4] and MIMO-based parallel architecture [1]. It is observed that the presented study gives the improvement in total delay by 70% w.r.t. [2] and 93% w.r.t. [3] and 2% w.r.t. [4] as compared in case studies 1 – 3. The reduction in area metric is observed as dipping in the slice requirement by 31%, 47%, and 110% and dipping in the LUT requirement by 58%, 66%, and 65%, further, the delay metric is improved by 49%, 80%, and 67% when compared with MIMO-based parallel architecture of [1] on the same platforms suggested in [2-4] respectively. Further, a tentative variant of the proposed architecture is designed by modifying a constant multiplication unit crafted with CSD-based multiplications, this realization leads to an additional 10-20% of optimization in the area-delay metric of the proposed work. The level of parallelism for parallel FIR filters can be scaled up or down as per the design requirements, in the

Table. 4 Post synthesis implementation details of tap 16 FIR filter architectures on FPGA Virtex 4 xc4vsx35-10ff668

| Relevant Work | | Resource Utilization | | | |
|---|---|---|---|---|---|
| | | No. of Slices | No. of LUT | Delay in ns (Freq in MHz) | No. Of Multipliers Used |
| DA [6] (2018) | | 733 | 1,186 | 19.4 | NA |
| OBC-DA [6] (2018) | | 722 | 1,166 | 21.69 | NA |
| DA [2] (2021) | | 670 | 1,064 | 15.1 | NA |
| Conventional Parallel MIMO L=8 [1] | | 6,808 | 12,328 | 9.101 (109.88) | 64 (16×16) |
| Proposed parallel L=8 | Conventional Multiplier | **4,712** | **5,205** | **4.620 (216.47)** | **8 (16×16)** |
| | With CSD | **4,578** | **5,894** | **2.265 (441.46)** | **0** |

Table. 5 Post synthesis implementation details of filter order 16 FIR filter architectures on FPGA Artix 7 xc7a200t-2fbg676

| Relevant Work | | Resource Utilization | | | |
|---|---|---|---|---|---|
| | | No. of Slices | No. of LUT | Delay in ns (Freq in MHz) | No. Of Multipliers Used |
| VM-CSA [7] (2015) | | 294 | 969 | 30.952 | NA |
| Various Mult[8] (2017) | | 375 | 1254 | 28.274 | NA |
| MAC [9] (2018) | | 444 | 1481 | 24.133 | NA |
| VD-FIR [10] (2019) | | 289 | 965 | 22.562 | NA |
| VM [3] (2020) | | 192 | 568 | 21.337 | NA |
| Conventional Parallel MIMO L=8 [1] | | 2545 | 8820 | 7.693 (129.99) | 72 (16×16) |
| Proposed parallel L= 8 | Conventional Multiplier | **1339** | **3032** | **1.473 (678.89)** | **9 (16×16)** |
| | With CSD | **609** | **1,977** | **1.473 (678.89)** | **0** |

Table. 6 Post synthesis implementation details of tap 16 FIR filter architectures on FPGA VIRTEX 5 XC5VSX95T-1FF1136 with a word length of 8-bits

| Relevant work | | Resource Utilization | | | | |
|---|---|---|---|---|---|---|
| | | NOS | SREG | SLUT | Delay in ns (Freq in MHz) | No. Of Multipliers Used |
| DA [11] (2008) | | 275 | 688 | 833 | 4.17 | NA |
| P-DA [12] (2011) | | 178 | 412 | 267 | 17.35 | NA |
| Logic IP core [13] | | 368 | 970 | 806 | 3.96 | NA |
| DPM [4] (2017) | | 120 | 222 | 476 | 2.426 | NA |
| OPC [4] (2017) | | 111 | 212 | 1318 | 2.081 | NA |
| Conventional Parallel MIMO L=8 [1] | | 1,116 | 1,208 | 3,648 | 6.313 (158.4) | 64 (16×16) |
| Proposed parallel L= 8 | Conventional Multiplier | **531** | **873** | **1,261** | **2.055 (486.6)** | **8 (16×16)** |
| | With CSD | **412** | **329** | **697** | **1.889 (529.4)** | **0** |

Note. NOS – Number of Slices, SREG – Slice Registers, SLUT – Slice LUTs

current study it is set to L=8 as a moderate level which is neither too low nor high.

## Conflicts of interest

The authors have no conflict of interest.

## Author contributions

Conceptualization, Kunjan D. Shinde; methodology, Kunjan D. Shinde; software, Kunjan D. Shinde; validation, Kunjan D. Shinde and Vijaya

C; formal analysis, Kunjan D. Shinde; investigation, Kunjan D. Shinde; resources, Kunjan D. Shinde; data curation, Kunjan D. Shinde; writing—original draft preparation, Kunjan D. Shinde; writing—review and editing, Vijaya C; visualization, Kunjan D. Shinde; supervision, Vijaya C.

## Acknowledgment

I would like to express my gratitude to my research supervisor, Vijaya C, who guided me throughout this study. I wish to acknowledge the suggestions provided by the doctoral committee. I extend my gratitude to the principal and management of SDM College of Engineering and Technology, Dharwad for the academic environment provided.

## References

[1] K. K. Parhi, *VLSI Digital Signal Processing System- Design and Implementation*, Wiley students edition, pp 63-83, 2013.

[2] S. Narendiran and E. P. Jayakumar, "An Efficient Modified Distributed Arithmetic Architecture Suitable for FIR Filter", In: *Proc. of International Conf. on Wireless Communications, Signal Processing and Networking (WiSPNET)*, Chennai, India, pp. 89-93, 2021.

[3] S. Janwadkar and R. Dhavse, "Strategic Reduction of Area and Power in FIR Filter Architecture for ECG Signal Acquisition", In: *Proc. of International Conf. on India Council International Conference (INDICON)*, New Delhi, India, pp. 1-7, 2020.

[4] P. J. Britto and D. Vaithiyanathan. "An Efficient Multichannel FIR Filter Architecture for FPGA and ASIC Realizations", *International Journal of Applied Engineering Research*, Vol. 12, No. 10, pp. 2209-2220, 2017.

[5] K. D. Shinde and C. Vijaya. "Bottlenecks in Finite Impulse Response Filter Architectures on a Reconfigurable Platform", In: *Proc. of International Conf. on Recent Advances in Artificial Intelligence and Data Engineering(AIDE)*, Karnataka, India, pp. 309-325, 2022.

[6] S. Akhter, S. Kumar, and D. Bareja, "Design And Analysis Of Distributed Arithmetic Based FIR Filter", In: *Proc. of International Conf. on Advances in Computing, Communication Control and Networking (ICACCCN)*, India, pp. 721-726 2018.

[7] A Mittal and A. Nandi, "Design of 16-bit FIR Filter using Vedic Multiplier with Carry Save Adder", In: *Proc. of International Conf. on IRF*, India, pp. 57- 60, 2015.

[8] A. Mittal, A. Nandi, and D. Yadav, "Comparative Study of 16-order FIR Filter Design using Different Multiplication Techniques", *IET Circuits, Devices & Systems*, Vol. 11, No. 3 pp. 196-200, 2017.

[9] N. S. Rai, B. S. P. Shree, Y. P. Meghana, A. P. Chavan, and R. Aradhya, "Design and implementation of 16 tap FIR filter for DSP Applications", In: *Proc. of International Conf. on Advances in Electronics, Computers and Communications (ICAECC)*, Bangalore, India, pp. 1-5, 2018.

[10] M. Sumalatha, P. V. Naganjaneyulu, and K. S. Prasad, "Low power and low area VLSI implementation of vedic design FIR filter for ECG signal de-noising", *Microprocessors and Microsystems Journal*, Elsevier, Vol. 71, pp. 1-13, 2019.

[11] P. K. Meher, S. Chandrasekaran, and A. Amira, "FPGA Realization of FIR Filters by Efficient and Flexible Systolization Using Distributed Arithmetic", *IEEE Transactions on Signal Processing*, Vol. 56, No. 7, pp. 3009-3017, 2008.

[12] P. K. Meher and S. Y. Park, "High-Throughput Pipelined Realization Of Adaptive FIR Filter Based On Distributed Arithmetic", In: *Proc. of International Conf. on VLSI and System-on-Chip*, Hong Kong, China, pp. 428-433, 2011.

[13] Xilinx, Inc., *LogiCORE IP FIR Compiler v5.0*, San Jose, CA, USA, 2010.

[14] C. Vijaya and U. Kumar, *Digital Signal Processing*, Elite publishers, p. 387, 2004.

[15] K. D. Shinde, K. A. Kumar, and C. N. Shilpa, "Impact of VLSI design techniques on implementation of parallel prefix adders", In: *Proc. of International Conf. on Soft Computing Systems (ICSCS)*, Kollam, India, pp. 473-482, 2018.

[16] K. D. Shinde, K. AmitKumar, D. S. Rashmi, S. Rukhsar, H. R. Shilpa, and C. R. Vidyashree, "A Novel Approach to Design Braun Array Multiplier Using Parallel Prefix Adders for Parallel Processing Architectures: -A VLSI Based Approach", In: *Proc. of International Conf. on Soft Computing Systems (ICSCS)*, Kollam, India, pp. 602-614, 2018.

[17] K. A. Asha and K. D. Shinde, "Performance analysis and implementation of array multiplier using various full adder designs for DSP applications: A VLSI based approach", In: *Proc. of International Conf. On Intelligent Systems*

*Technologies and Applications*, India, pp. 731-742, 2016.

[18] Xilinx FPGA details and datasheet accessed on 12-April-2023. VIRTEX 5 family https://docs.xilinx.com/v/u/en-US/ds100 VIRTEX 4 family https://www.fpgakey.com/xilinx-parts/xc4vsx35-10ff668i ARTIX 7 family https://www.xilinx.com/products/silicon-devices/fpga/artix-7.html.

[19] K K. Parhi, *VLSI Digital Signal Processing System- Design and Implementation*, Wiley students edition, pp. 31-40, pp. 559-583, 2013.

[20] A. P. Vinod, E. Lai, D. L. Maskell, and P. K. Meher, "An Improved Common Subexpression Elimination Method For Reducing Logic Operators In FIR Filter Implementations Without Increasing Logic Depth", *Integration*, Vol. 43, No. 1, pp. 124-135, 2010.

[21] K. A. Rao, A. Kumar, D. Kaplun, S. K. Patel, and N. Purohit, "Design of low complexity parallel polyphase finite impulse response filter using coefficient symmetry", *IET Circuits, Devices & Systems*, Vol. 17, No. 1, pp. 29-37, 2023.

[22] K. A. Rao, M. Pandit, and N. Purohit, "Efficient 3-parallel polyphase odd length FIR filter using Brent Kung adder and Booth multiplier for VLSI applications", In: *Proc. of International Conf. On Electrical, Electronics and Computer Engineering (UPCON)*, Prayagraj, India, pp.1-5, 2022.

[23] M. Prabhavathy and S. Sakthivel, "VLSI Implementation of Fully Parallel and CSD FIR Filter Architecture", In: *Proc. of International Conf. on Smart Electronics and Communication (ICOSEC)*, Trichy, India, pp. 321-327, 2022.

[24] J. Ye, M. Yanagisawa and Y. Shi., "Scalable Hardware Efficient Architecture for Parallel FIR Filters with Symmetric Coefficients", *Electronics*, Vol. 11, No. 20, p. 3272, 2022.

[25] M. Madni, and C. Vijaya, "Hand Gesture Recognition Using Semi Vectorial Multilevel Segmentation Method with Improved ReliefF Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 447- 457, 2021, doi: 10.22266/ijies2021.0630.37.