

An Intelligent Retiming-Based Optimization Framework for High-Performance FIR Digital Filter Architectures Using Improved Hippopotamus Optimization Algorithm

Shafiqullah Sarwary¹, Mandeep Kaur^{2*}, Charanjit Singh³

¹ Research Scholar, Department of ECE, Punjabi University, Patiala, India, sarwary205@gmail.com

² Assistant Professor, Department of ECE, Punjabi University, Patiala, India, ermandeep0@gmail.com

³ Professor, Department of ECE, Punjabi University, Patiala, India, channisingh@yahoo.com

* Corresponding author: E-mail address: ermandeep0@gmail.com

Abstract

Finite Impulse Response (FIR) digital filters are essential components in digital signal processing systems; however, high-order FIR filter implementation faces challenges related to hardware complexity, register utilization, and critical path delay. Retiming techniques provide an effective solution for improving the speed and resource efficiency of VLSI-based FIR architectures by optimizing the placement of memory elements. This research presents an intelligent retiming-based optimization framework using an Improved Hippopotamus Optimization (IHO) algorithm for enhancing FIR filter performance. The proposed approach combines retiming transformation with metaheuristic optimization to achieve efficient register distribution and improved timing characteristics. The performance of the optimized FIR architectures is evaluated using multiple parameters, including register count, clock period, operating frequency, and execution time. Comparative analysis with Hippopotamus Optimization (HO), Schooling Aquatic Optimization (SchoA), and Moth Flame Optimization (MFO) demonstrates the effectiveness of the proposed method. The optimized FIR-TDF10 architecture achieves an improvement in operating frequency from 108.24 MHz to 175.91 MHz, along with reduced register utilization and enhanced computational efficiency. The results confirm that the proposed IHO-based retiming framework provides an effective approach for designing high-speed and area-efficient FIR digital filters for advanced VLSI applications.

Keywords: FIR Filter; Retiming; Improved Hippopotamus Optimization; Metaheuristic Optimization; VLSI Design; Register Reduction; Clock Period Optimization.

1. Introduction

Finite Impulse Response (FIR) filters are key components of many digital signal processing (DSP) systems. They are stable and provide a linear phase response, which is very important for many applications. Due to the finite impulse response, they are able to provide very precise frequency-selective filtering. Because of these characteristics, FIR filters are used in a lot of applications such as wireless communication systems, biomedical signal processing, image processing, and others. They are also used in radar systems, audio processing, and in embedded computing systems. Recent developments in high-speed and low-power DSP systems impose significant design challenges for the realization of FIR filters, especially high-order FIR filters. The additional complexity of the computations, the increased hardware utilization, and the additional delay impose significant constraints on the design of such filters. The performance of an FIR filter can be influenced greatly by the architectural parameters such as number of arithmetic units, amount of memory, register distribution, and critical path delay. Most existing FIR filter structures require a large number of registers to realize adequate filter performance. However, the number of registers greatly affects the area, power, and even hardware complexity of the overall architecture. Moreover, the long chains of combinational logic between different sequential elements in the filter architecture severely restrict the maximum operating frequency of the filter. Hence, the optimization of an FIR filter architecture has become a major area of research to achieve higher speed, lower area, and VLSI-efficient designs.

Retiming is an optimization technique that allows the timing of a digital circuit to be improved without changing its functionality. It works by relocating registers in a design to distribute delay through a circuit of combinational logic in the best possible way. Delay elements within the design can also be rearranged to get the best results. In digital circuit design, retiming can be used to reduce the critical path of a design, increase its frequency of operation, and improve overall timing performance. For FIR filter architectures, retiming has the potential to decrease the clock period of a design and improve its resource usage. However, creating an optimal retiming for a complex filter is a difficult problem because the search space for possible designs is very large, and there are many competing objectives to optimization.

Metaheuristic optimization algorithms are recently popular to solve complex engineering optimization problems. Most of these algorithms are inspired from nature, and therefore, are called nature-inspired optimization algorithms. These algorithms are applied for hardware optimization and digital circuit design using various single- and multi-objective optimization problems. These problems usually involve several competing objectives such as area, delay, power and performance. While there are many conventional metaheuristic optimization techniques, they have some limitations such as slow convergence, premature convergence and low search accuracy for large-scale design spaces.

Improved Hippopotamus Optimization (IHO) is an improved searching capability to find the optimal solutions by making a good trade-off between exploration and exploitation. IHO is further integrated with retiming-based FIR filter optimization to address the limitations of conventional filter structures. By combining intelligent optimization with retiming, optimal register allocation, minimum clock period, and highest operating frequency can be achieved. Design of high-performance FIR filter architectures requires a number of competing design objectives to be traded off rather than design optimized for a single metric. Reducing the number of registers required improves hardware efficiency whilst reducing the clock period improves processing speed. Thus a multi-objective optimisation approach is required to trade off hardware complexity against timing. Architectures can be evaluated using a number of different metrics including number of registers, clock period, maximum frequency and execution time allowing evaluation of effectiveness of optimisation.

The paper proposes an intelligent retiming-based optimization framework for FIR digital filter architectures using the Improved Hippopotamus Optimization algorithm (IHOA). The proposed approach investigates the effect of optimized retiming vectors and compares the results with other state-of-the-art optimization algorithms, namely Hippopotamus Optimization (HOA), Schooling Aquatic Optimization (SAOA), and Moth Flame Optimization (MFOA). In order to evaluate the performance of IHOA in different scenarios, several FIR filter structures have been investigated. The results obtained from the different architectures in terms of number of registers, timing consumption and computational effort have been presented and discussed. The proposed framework successfully exploits the potential of retiming in optimizing the implementation of FIR filters in terms of register usage and timing consumption. The main objective of this work is to design and implement high speed and efficient utilization of resources of implementation of FIR digital filters which can be implemented as VLSI systems for current and future applications in the field of DSP. Efficient design of implementation of FIR filters using a novel approach of optimization is presented, and its application for design of efficient digital systems with improved performance and reduced number of resources which increase capabilities of computation are discussed.

2. Literature Review

The design of efficient architectures for real-time implementation of Finite Impulse Response (FIR) digital filters has continued to be a relevant area of research in the field of digital signal processing. As advances in digital system design continue to allow for the implementation of higher speed, lower power, and smaller area systems, there is a growing interest in the design of architectures that can efficiently implement FIR filters. The work of Lee et al. [1] introduced a number of techniques for the design of hardware-efficient architectures for the implementation of programmable FIR filters. These techniques were used to design an architecture that makes efficient use of available hardware and is particularly suited to the implementation of a number of different filtering applications. A computer-aided design system, FIRGEN, was introduced by Jain et al. [2] for the design of high-performance integrated circuits for FIR filters. The design system takes an architectural approach to the problem and addresses a number of issues involved in the implementation of such filters.

With the advent of digital signal processing applications, the need for high-speed arithmetic units and efficient architecture realization has increased. A digital audio signal processor architecture that performs high-speed signal processing was developed by Matsushita et al. [3]. In this architecture, arithmetic units and data processing units are designed to be efficient. High-speed CMOS adder and multiplier modules for digital signal processing have been investigated by Kernhof et al. [4]. The performance of a DSP architecture greatly depends on the arithmetic components. Designing flexible adaptive FIR filter VLSI architectures that can be used for communication applications with adaptive processing capabilities and flexible hardware structure was the focus of work by Borth et al. [5].

Research regarding improving performance of FIR filters through the means of VLSI-optimization using distributed arithmetic, parallelization, multipliers-optimization and low power design methods has been lately ongoing. A VLSI architecture for the implementation of an adaptive Finite Impulse Response filter using Distributed Arithmetic for low power DSP applications has been proposed by Rao et al. in [6]. Implementation of FIR filter for low power applications using hybrid multiplier architecture with optimized operation scheduling has been presented by S. et al. in [7]. The research also focused on improving computational efficiency and on implementation of the designed filter using FPGA. VLSI-optimized implementation of two-dimensional Finite Impulse Response filters using approach based on

rotational symmetry and parallel Distributed Arithmetic has been proposed by Odugu et al. in [8] for the purposes of efficient execution of image and signal processing tasks.

In FIR filters, the multiplications are by far the largest contributors to both area and delay. Thus, the optimization of multipliers has recently received a lot of attention. In [9], dynamic architectures for multiplications that can be used for implementing FIR filters have been studied in detail for improving their performance. For the physical design of 2D-FIR filters, binary particle swarm optimization (BPSO) was applied for VLSI floor planning in [10] in order to improve the major design metrics. The implementations of FIR filters for 2D-convolutions on FPGAs have been reviewed in [11] for high-throughput architectures. Parallel architectures were the key to achieve high-throughput processing.

The FIR filters can be implemented by using the multiplier-free architectures, coefficient optimized processing, and high performance arithmetic. In this direction, V. et al. [12] have proposed the enhanced multiplier-free VLSI architectures for signal processing. The design and implementation of high resolution digital decimation filter for SADA/DAC has been presented by Zhou et al. [13]. This paper highlights the importance of efficient filtering in high resolution signal processing. The coefficient-based filters and their high-throughput multiplier implementations for high-performance digital signal processing have been investigated by Vats et al. [14]. The paper deals with the optimization of Dadda multiplier structures for signal processing and its application to enhance the performance of DSP systems has been studied by H. L. and Kavida [15].

FIR architectures have recently been designed for high-speed and energy efficient DSP applications using techniques such as parallel processing, approximate computing and high-performance multipliers. Recently, Bharatha et al. [16] proposed an implementation of an enhanced fast DSP capable FIR filter architecture to improve computational efficiency. Further, Potharaju and Saminadan [17] presented implementation of the efficient FIR filter architecture using approximate multipliers and adders to trade-off between hardware and performance. In addition, Karthik et al. [18] proposed implementation of a high-speed two-dimensional FIR filter using high-performance Vedic multipliers. The low-power parallel transposed FIR filter architectures were designed using modified gate diffusion input by Murugesan and Prathika [19]. An implementation of a twin-mode adaptive FIR filter using distributed arithmetic was proposed by A. et al. [20] to realize adaptive as well as computationally efficient FIR filter.

Optimization of FIR filters has been a major research area and significant progress has been achieved in improving different aspects of FIR filters, including reduction of multipliers, power optimization and even architectural improvement. However, optimal design of high-order FIR filters that considers timing, number of registers to be used and hardware complexity simultaneously is a challenging task. Using retiming as an optimization technique is effective since it tries to meet timing by moving registers while keeping the circuit functional. The search space for optimal retiming of complex FIR filters is very large. Using intelligent optimization techniques in conjunction with retiming is a promising approach for designing high-speed FIR filters and also for reducing resources needed in implementation.

3. Proposed Methodology

This work presents a smart retiming-based design optimization methodology in order to increase the performance of implementations of Finite Impulse Response (FIR) digital filters by combining retiming design transformations with the Improved Hippopotamus Optimization (IHO) algorithm. The main objective of this proposed methodology is to reduce the hardware size and to decrease the number of timing constraints of the Filter in order to increase its operating frequency while keeping its filtering characteristics. This methodology includes a combination of architectural-level design optimizations and a nature-inspired computational approach for optimal placement and buffering of registers in the Filter architecture.

This framework goes through three major stages of FIR filter architecture modeling, retiming-based design transformation and intelligent design optimization by using IHO algorithm. In the architecture modeling stage, FIR filter is analyzed to obtain critical path, arithmetic components and required registers. In the design transformation stage, the retiming model is applied to re-distribute delay components to obtain an optimized retiming solution by taking into account register count, clock period and maximum operating frequency using the IHO algorithm. An overview of the framework of the intelligent approach for retiming-based optimization of FIR filters is shown in Figure 1. At the beginning, a model of the FIR filter is created in the form of the architectural description. In this description, the main parameters of the filter, such as the number of registers, clock period and critical path delay, are determined. The following stage is retiming. It transforms the architecture of the filter by moving registers without changing its functionality. The created solutions for retiming of the filter are optimized by the Improved Hippopotamus Optimization algorithm.

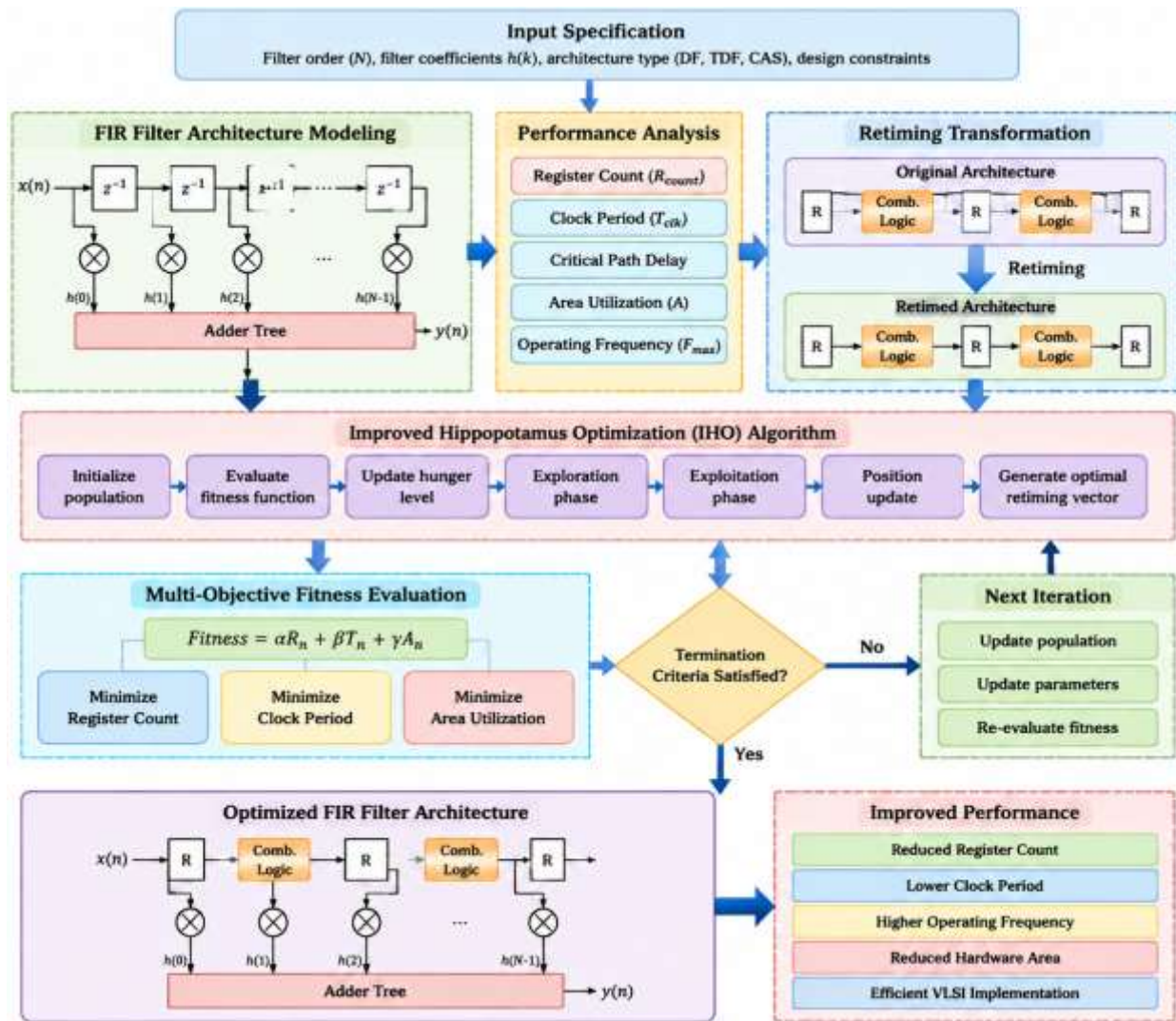


Figure.1: Overall Framework of the Proposed Intelligent Retiming-Based FIR Filter Optimization Method
The main characteristic of this algorithm is the combination of the stages of search and exploitation by using the corresponding solutions as input for the current iteration. The quality of each solution is estimated by a multi-objective fitness function that takes into account the complexity of the optimized hardware and its timing characteristics. Optimization continues until certain termination criteria are fulfilled. The output of the optimization process is an optimal solution for retiming of the filter that enables its implementation with maximum frequency, minimum number of registers, and best possible VLSI design quality. In this work, a multi-objective design approach is used to optimize between hardware efficiency and timing performance. Each candidate solution, generated by the optimizer and corresponding to a different retiming solution, is evaluated using a fitness function. The design that results in the minimum hardware overhead while improving the timing figures of merit is chosen as the optimal solution for implementing high-speed FIR filters for VLSI-based digital signal processing applications.

3.1 FIR Filter Architecture

The FIR filter is a non-recursive digital filter structure in which the output response depends only on the present and previous input samples. The general mathematical representation of an N-tap FIR filter is expressed as:

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

where $y(n)$ represents the filter output, $x(n)$ denotes the input signal sequence, $h(k)$ represents the filter coefficient, and N indicates the filter order.

The transfer function of an FIR filter can be represented as:

$$H(z) = \sum_{k=0}^{N-1} h(k)z^{-k} \quad (2)$$

The FIR architecture consists of three fundamental components: delay elements, multipliers, and adders. The delay elements store previous input samples, the multipliers perform coefficient multiplication, and the adders accumulate the multiplication results to generate the final output response. The input sample delay operation is expressed as:

$$x_k(n) = x(n - k) \quad (3)$$

The multiplication operation between input samples and filter coefficients is represented as:

$$m_k(n) = h(k)x(n - k) \quad (4)$$

The output accumulation process is defined as:

$$y(n) = m_0(n) + m_1(n) + \dots + m_{N-1}(n) \quad (5)$$

FIR architectures can be implemented using different architectural forms, like direct form, transposed direct form and cascaded forms. The transposed direct form structure has better pipelining due to placement of registers near the multiplier output. However, as the order of filter increases, number of operations and memory required increases which affects hardware efficiency negatively.

The computational complexity of FIR filtering increases with the number of taps and can be represented as:

$$C = N(M + A) \quad (6)$$

Where M represents multiplication operations and A represents addition operations.

The area requirement of an FIR architecture depends on the number of hardware resources utilized:

$$A_{total} = A_M + A_A + A_R \quad (7)$$

Where A_M , A_A and A_R denote multiplier delay, adder delay, and register delay.

Optimization of these architectural parameters is essential for achieving high-speed and resource-efficient FIR implementations.

3.2 Retiming Model

Retiming is a hardware optimization technique that modifies the position of registers within a digital circuit while preserving the input-output functionality. The main objective of retiming is to balance the pipeline stages, reduce critical path delay, and improve clock frequency.

A digital circuit can be represented as a directed graph:

$$G = (V, E) \quad (8)$$

Where V represents the set of computational nodes and E represents the connections between nodes.

Each edge contains a number of delay elements represented as:

$$W: E \rightarrow Z^+ \quad (9)$$

The retiming process assigns a retiming value to each node:

$$r: V \rightarrow Z \quad (10)$$

The number of registers after retiming between two connected nodes is calculated as:

$$W_r(u, v) = W(u, v) + r(v) - r(u) \quad (11)$$

Where $W_r(u, v)$ represents the updated register count after retiming.

The total number of registers in the architecture can be represented as:

$$R_{total} = \sum_{(u,v) \in E} W_r(u, v) \quad (12)$$

The retiming transformation maintains functional equivalence by ensuring that the register count remains non-negative:

$$W_r(u, v) \geq 0 \quad (13)$$

The clock period of the retimed architecture is determined by the maximum combinational delay between consecutive registers:

$$T_{clk} = \max(D_i) \quad (14)$$

Where D_i represents the delay of the i th pipeline stage.

The operating frequency after retiming is calculated as:

$$F_{max} = \frac{1}{T_{clk}} \quad (15)$$

The optimization objective of retiming is formulated as:

$$\text{Minimize}(T_{clk}, R_{total}) \quad (16)$$

The retiming vector is represented as:

$$R = [r_1, r_2, r_3, \dots, r_n] \quad (17)$$

For a given target performance, the optimal retiming solution is a vector that minimizes hardware complexity. By integrating retiming with intelligent optimization techniques, large design spaces containing a multitude of different register configurations can be searched automatically for the optimal retiming vector using the Improved Hippopotamus Optimization (IHO) algorithm. This algorithm evaluates various architectural alternatives against user-specified performance targets.

3.3 Proposed IHO Optimization Framework

In this work, an Improved Hippopotamus Optimization (IHO) algorithm is integrated into a retiming framework to find the optimal number of registers for given FIR digital filter structures. The IHO algorithm, inspired by hunting and searching behaviors in nature, uses a population of organisms, where each organism represents a possible retiming solution. In the optimization process, IHO tries to minimize hardware size and delay while increasing the clock frequency of the filter. In the framework, the quality of a candidate solution is evaluated using three different design metrics, namely the number of registers, clock period, and area.

The search agents represent possible retiming solutions, and their positions in space correspond to different register configurations within the proposed FIR architecture. Within given search boundaries, an initial set of potential solutions is generated. Their performance is evaluated by means of a multi-objective fitness function. In each iteration of the optimization process the positions of the search agents are updated based on the current Hippopotamus value, the best solution found so far, and on the agents' exploration- vs. -exploitation behavior, which is adapted online.

The mathematical formulation of the proposed IHO optimization process is described as follows.

The population of search agents is represented as:

$$X = \{X_1, X_2, X_3, \dots, X_N\} \quad (18)$$

Where N represents the number of search agents.

Each candidate solution is represented by a retiming vector:

$$X_i = [x_{i1}, x_{i2}, x_{i3}, \dots, x_{id}] \quad (19)$$

Where d represents the dimension of the optimization problem.

The initial position of the i th search agent is generated as:

$$X_i = X_{min} + rand(0,1)(X_{max} - X_{min}) \quad (20)$$

Where X_{min} and X_{max} represent the lower and upper boundaries of the search space. The boundary constraints of the retiming vector are defined as:

$$X_{min} \leq X_i \leq X_{max} \quad (21)$$

The fitness value of each candidate solution is calculated based on the design objectives:

$$Fitness = f(R_{count}, T_{clk}, A) \quad (22)$$

Where R_{count} represents register count, T_{clk} represents clock period, and A represents area utilization. The weighted multi-objective fitness function is expressed as:

$$Fitness = w_1 R_{count} + w_2 T_{clk} + w_3 A \quad (23)$$

Where w_1, w_2, w_3 represent weighting factors. The normalized register utilization is calculated as:

$$R_n = \frac{R_{count} - R_{min}}{R_{max} - R_{min}} \quad (24)$$

The normalized clock period is represented as:

$$T_n = \frac{T_{clk} - T_{min}}{T_{max} - T_{min}} \quad (25)$$

The normalized area utilization is obtained as:

$$A_n = \frac{A - A_{min}}{A_{max} - A_{min}} \quad (26)$$

The combined optimization objective is formulated as:

$$Fitness = \alpha R_n + \beta T_n + \gamma A_n \quad (27)$$

Where α, β and γ represent objective importance coefficients.

The best solution obtained during each iteration is represented as:

$$X_{best} = \min(Fitness(X_i)) \quad (28)$$

The Hippopotamus level of each search agent is calculated as:

$$H_i = \frac{Fitness_i - Fitness_{best}}{Fitness_{worst} - Fitness_{best}} \quad (29)$$

The adaptive Hippopotamus coefficient is expressed as:

$$H_c = H_{max} - (H_{max} - H_{min}) \frac{t}{T} \quad (30)$$

Where t represents the current iteration and T represents the maximum number of iterations. The exploration coefficient is defined as:

$$E = 2 \left(1 - \frac{t}{T} \right) \quad (31)$$

The exploitation coefficient is represented as:

$$P = \frac{t}{T} \quad (32)$$

The distance between the current solution and best solution is calculated as:

$$D_i = |X_{best} - X_i| \quad (33)$$

The movement direction toward the optimal solution is expressed as:

$$S_i = X_{best} - X_i \quad (34)$$

The exploitation-based position update is represented as:

$$X_i^{t+1} = X_{best} + P \times H_i \times D_i \quad (35)$$

The adaptive movement strategy is expressed as:

$$X_i^{t+1} = X_i^t + H_i(E \times S_i + P \times D_i) \quad (36)$$

The random search behavior is defined as:

$$X_i^{t+1} = X_i^t + rand(-1,1)(X_{max} - X_{min}) \quad (37)$$

The updated solution after boundary control is represented as:

$$X_i^{t+1} = \min(\max(X_i^{t+1}, X_{min}), X_{max}) \quad (39)$$

The improvement rate of each search agent is calculated as:

$$IR_i = \frac{Fitness_i^t - Fitness_i^{t+1}}{Fitness_i^t} \quad (40)$$

The convergence factor is expressed as:

$$C_f = 1 - \frac{t}{T} \quad (41)$$

The adaptive search parameter is calculated as:

$$A_s = C_f \times rand \quad (42)$$

The exploration probability is represented as:

$$P_e = rand(0,1) < C_f \quad (43)$$

The exploitation probability is defined as:

$$P_x = 1 - P_e \quad (44)$$

The new candidate solution generation is represented as:

$$X_{new} = X_i + A_s(X_{best} - X_i) \quad (45)$$

The solution acceptance condition is expressed as:

$$Fitness(X_{new}) < Fitness(X_i) \quad (46)$$

The accepted solution update is represented as:

$$X_i = X_{new} \quad (47)$$

The population best solution update is defined as:

$$X_{best}^{t+1} = \min(Fitness(X_i^{t+1})) \quad (48)$$

The mean population position is calculated as:

$$X_{mean} = \frac{1}{N} \sum_{i=1}^N X_i \quad (49)$$

The diversity measurement of the population is represented as:

$$Diversity = \frac{1}{N} \sum_{i=1}^N |X_i - X_{mean}| \quad (50)$$

The adaptive diversity control factor is expressed as:

$$D_c = \frac{Diversity}{Diversity_{max}} \quad (51)$$

The modified exploration parameter is calculated as:

$$E_m = E \times D_c \quad (52)$$

The modified exploitation parameter is represented as:

$$P_m = P(1 - D_c) \quad (53)$$

The optimization termination condition is defined as:

$$t = T \quad (54)$$

The final optimal retiming vector is obtained as:

$$R_{opt} = X_{best} \quad (55)$$

The optimized register count is calculated as:

$$R_{optimum} = \sum W(R_{opt}) \quad (56)$$

The optimized clock period is obtained as:

$$T_{opt} = \min(T_{clk}) \quad (57)$$

The maximum operating frequency after optimization is expressed as:

$$F_{max,opt} = \frac{1}{T_{opt}} \quad (58)$$

The percentage improvement in operating frequency is calculated as:

$$Improvement(\%) = \frac{F_{new} - F_{old}}{F_{old}} \times 100 \quad (59)$$

The reduction in register count is calculated as:

$$Reduction(\%) = \frac{R_{old} - R_{new}}{R_{old}} \times 100 \quad (60)$$

The delay improvement percentage is represented as:

$$Delay_{gain}(\%) = \frac{T_{old} - T_{new}}{T_{old}} \times 100 \quad (61)$$

The execution time of the optimization algorithm is expressed as:

$$T_{exe} = T_{iteration} \times T_{evaluation} \quad (62)$$

The convergence rate is calculated as:

$$CR = \frac{Fitness_{initial} - Fitness_{final}}{Fitness_{initial}} \quad (63)$$

The final optimization problem can be represented as:

$$Minimize\{R_{count}, T_{clk}, A\} \quad (64)$$

subject to:

$$W_r(u, v) \geq 0 \quad (65)$$

The IHO framework for retiming of digital FIR filters is used to iteratively improve candidate solutions for architectures until an optimal solution is found. The framework integrates adaptive models for Hippopotamus, exploration and exploitation as well as multi-objective optimization. Using the optimized retiming the number of registers is decreased, the clock period is minimized and the operating frequency is increased, which is important for high-performance VLSI-design.

4. Objective Function

The objective function is a key component in the intelligent retiming-based optimization framework for designing FIR filters. It optimizes between three different objectives, namely the number of registers, clock period, and area utilization. In order to find the optimal solution for the design of a FIR filter, the Improved Hippopotamus Optimization (IHO) algorithm searches through the solution space of all possible retiming solutions. Each candidate solution is evaluated by a multi-objective fitness function that is defined by the objective function. The objective function assesses the quality of a candidate solution and selects the best solution from all solutions in the solution space.

Reducing the register count between stages leads to lower circuit area and better resource utilization. The clock period is the maximum delay between stages, and reducing the clock period helps to increase the operating frequency of the architecture. The area utilization parameter of the implemented architecture is also considered.

The combined objective function is formulated as:

$$Fitness = \alpha R_n + \beta T_n + \gamma A_n \quad (66)$$

Where R_n, T_n, A_n represent the normalized register count, normalized clock period, and normalized area utilization, respectively. The coefficients α, β and γ represent the weighting factors assigned to each optimization parameter.

The normalized register count is expressed as:

$$R_n = \frac{R - R_{min}}{R_{max} - R_{min}} \quad (67)$$

The normalized clock period is calculated as:

$$T_n = \frac{T - T_{min}}{T_{max} - T_{min}} \quad (68)$$

The normalized area utilization is represented as:

$$A_n = \frac{A - A_{min}}{A_{max} - A_{min}} \quad (69)$$

The optimization problem can be mathematically represented as:

$$Minimize\ Fitness(R, T, A) \quad (70)$$

subject to:

$$R \geq 0, \quad T > 0, \quad A > 0 \quad (71)$$

The optimal retiming solution is given by the minimum fitness value found during the search using the objective function as the measure. In turn, the function is minimized to achieve an optimal register balancing, minimum timing slack, and high performance of the implemented digital FIR filters. The objective function allows for a joint optimization of all relevant design parameters while trying to achieve the best possible trade-off between them. This function is particularly suitable for a high-performance VLSI implementation of FIR filters.

5. Experimental Setup

In this work, we evaluate the performance of our proposed framework for intelligent retiming-based FIR filter design using several different FIR architectures as well as various metaheuristics. Here, we assess the

performance of the Improved Hippopotamus Optimization (IHO) algorithm to find near-optimal retiming solutions that result in good trade-off between hardware complexity and timing performance. We have used several FIR filter structures such as Direct Form (FIR-DF), Transposed Direct Form (FIR-TDF) and Cascade Form (FIR-CAS) of various orders.

This novel approach was also compared with several approaches for the VLSI design optimization, i.e., Hippopotamus Optimization (HO), Schooling Aquatic Optimization (SchoA), and Moth Flame Optimization (MFO). To assess the effect of the retiming optimization on the hardware efficiency and the processing speed, relevant design parameters of the VLSI circuit such as the number of registers, clock period, maximum frequency, and execution time were considered for the comparative analysis.

The FIR filter architectures were studied before and after retiming. The retiming solutions were then optimized by applying the IHO algorithm. An multi-objective fitness function, which includes the number of registers, the timing delay and area-related objectives, was used for the optimization. The optimization was performed in an iterative manner until the termination criteria were met, and the optimal retiming solution as represented by the retiming vector was found. The framework has been evaluated and validated for the optimized FIR filters in order to assess its effectiveness. Experimental results reveal the ability of proposed method to reduce hardware cost of implementation, to minimize clock period and to increase frequency of operation. The experimental setup represents a complete and realistic environment for assessment and verification of proposed optimization method for high-performance VLSI-based FIR digital filters.

6. Results and Discussion

In this paper, different FIR digital filter architectures as well as optimization methods have been used to evaluate the performance of the proposed retiming-based optimization approach. The Improved Hippopotamus Optimization (IHO) algorithm has been evaluated by measuring several VLSI design metrics, i.e., number of registers, critical path/instruction cycle, operating frequency, execution time, and convergence and overall performance improvement. Results show that IHO can effectively find the optimal retiming configuration, thus significantly improving the hardware efficiency of given FIR filter architectures.

Figure 2 demonstrates a comparison of different optimization methods, including IHO, HO, SchoA and MFO. The register count and clock period of the FIR architecture that was optimized using the different methods are shown. As it can be seen from the results of the comparison provided in Figure 2, IHO requires the fewest number of registers and achieves the lowest clock period compared with the other optimization methods. By sharing registers and selecting the optimal retiming vectors, IHO achieves reduced hardware complexity and high processing speed.

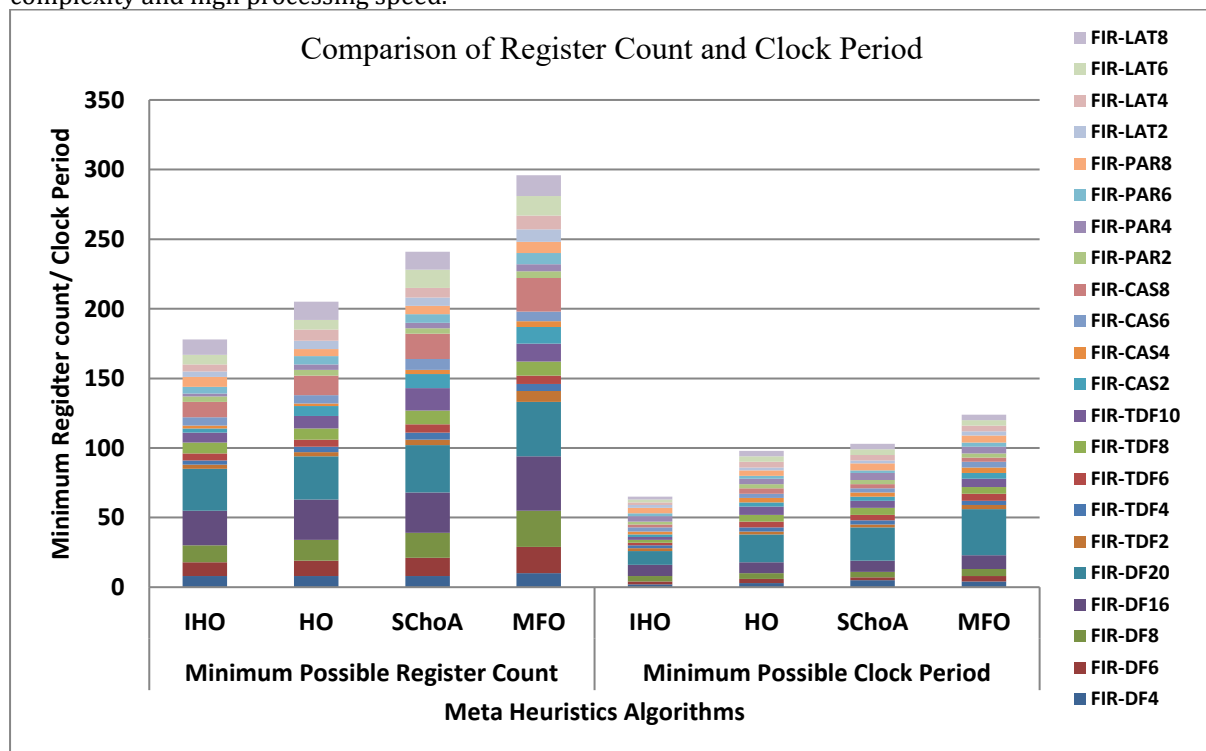


Figure.2: Comparison of IHO, HO, SchoA, MFO based digital filters in term of Register count and Clock Period

The performance figures for different FIR structures before and after retiming are shown in Figure. 3. Several different FIR structures have been considered within the design space and their operating frequency, as well as their overall timing figures, have been compared before and after retiming using the new timing driven framework for retiming. In addition to providing significant improvements over existing designs, the retiming within the proposed framework has enabled the creation of high speed designs such as the optimized FIR-TDF10 implementation whose operating frequency has increased from 108.24 MHz to 175.91 MHz. Figure 4 demonstrates execution time for different optimization approaches for the FIR filter architecture optimization. The proposed IHO algorithm leads to a reduction in execution time in comparison with the HO, SchoA and MFO methods. Due to its fast convergence IHO is able to find an optimal retiming solution very fast, keeping optimization quality on a high level. As a result, IHO is suitable for the complex FIR filter design space with a vast number of possible solutions.

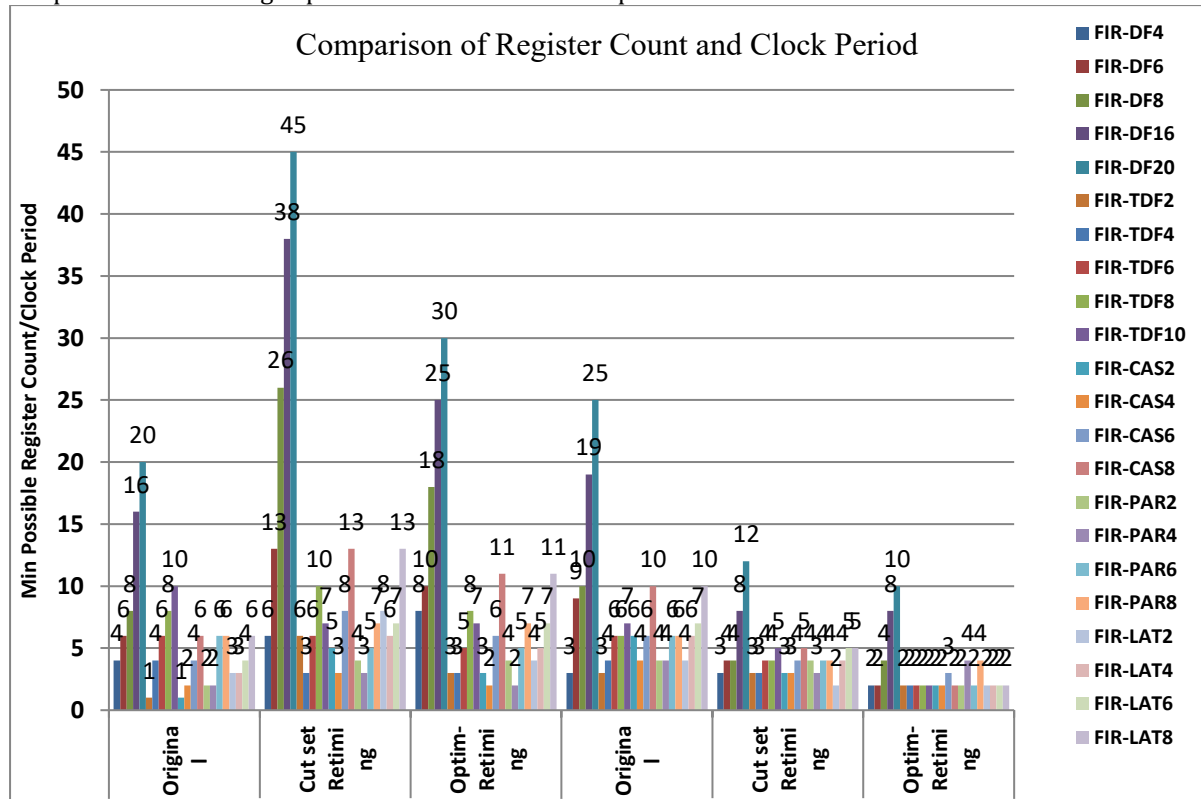


Figure.3: Comparison of register count and clock period for the different Models

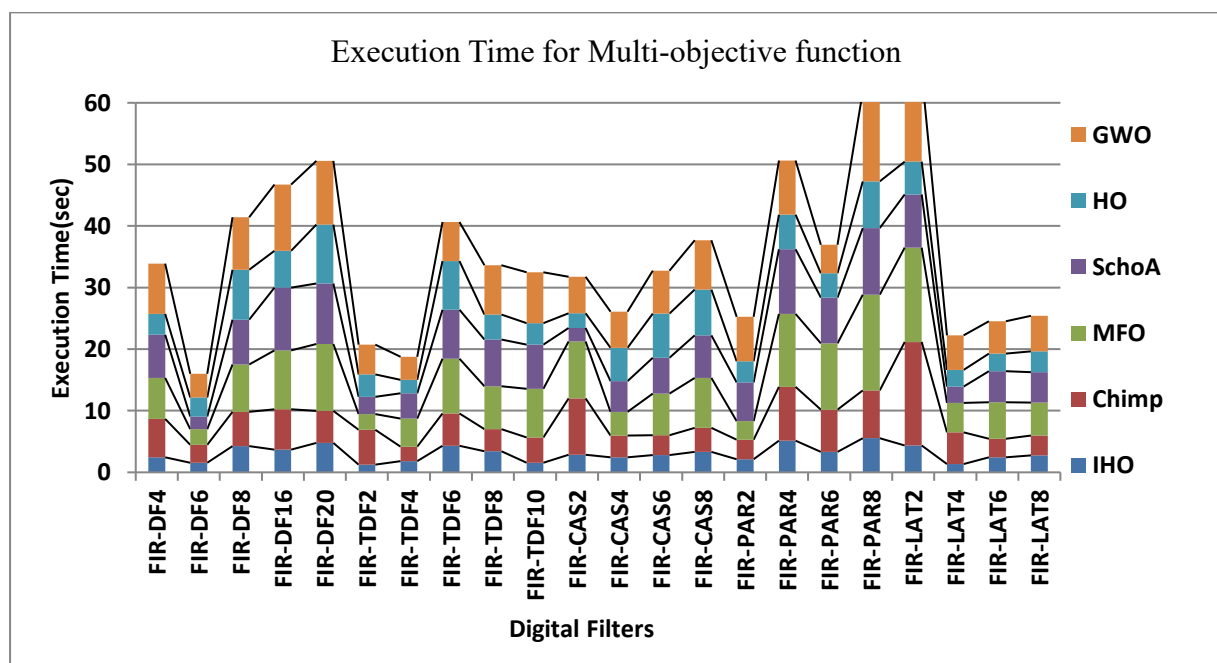


Figure.4: Comparison of Execution time of Multi Objective Function with different algorithms

The convergence of the optimization methods to find the optimal retiming solution is depicted in Figure 5. This figure plots the objective function values obtained at various iterations during the search for the optimum. The figure clearly shows that IHO has faster convergence and attains lower fitness value than other methods. The fast convergence of IHO to find optimal retiming solution is attributed to the adaptive exploration and exploitation of search space during the optimization of design parameters such as number of registers, clock period and area. The lower value of the objective function indicates that IHO has effectively performed the multi-objective optimization of design parameters. Figure 6 illustrates the performance improvement for the various FIR architectures considered in this study after applying the retiming and IHO optimization to the original, retimed architectures. A marked increase in the performance index is observed upon applying the IHO optimization. The reasons for this increased performance are 1) efficient register relocation, 2) reduced critical path delay, and 3) efficient hardware utilization. It can also be observed that higher-order FIR filter architectures receive the greatest performance improvement. This is because the IHO-based design methodology is highly suitable for VLSI realization of complex digital signal processing systems.

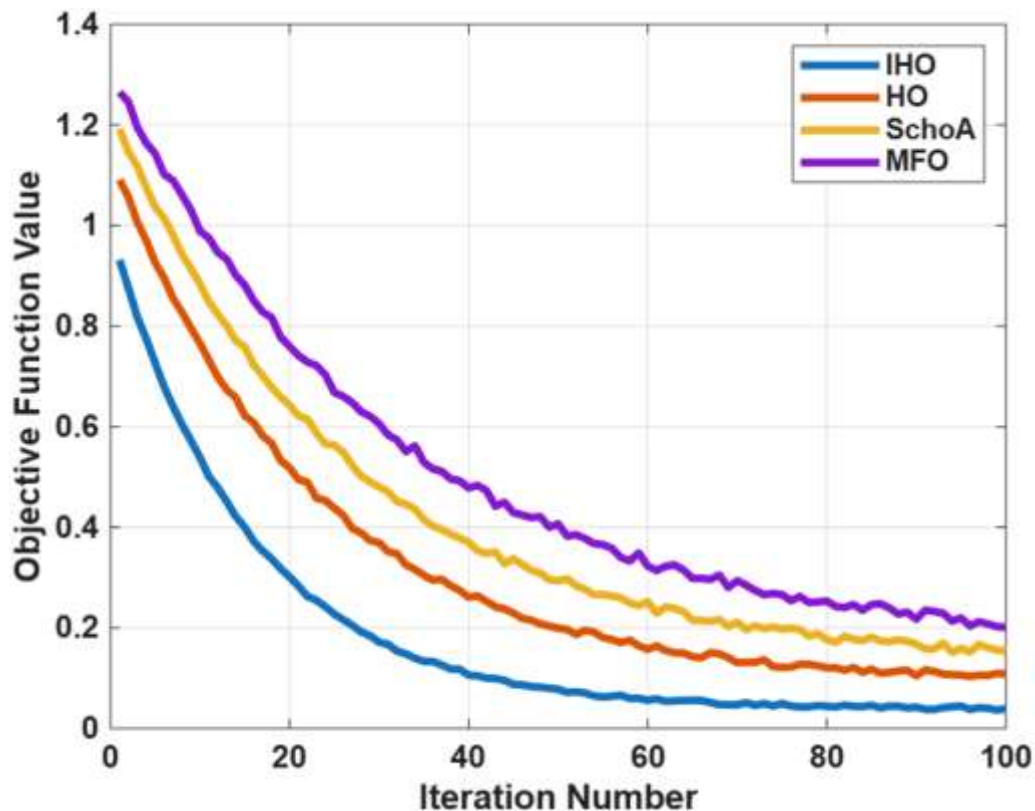


Figure 5. Convergence Characteristics of IHO Optimization Algorithm for FIR Filter Retiming

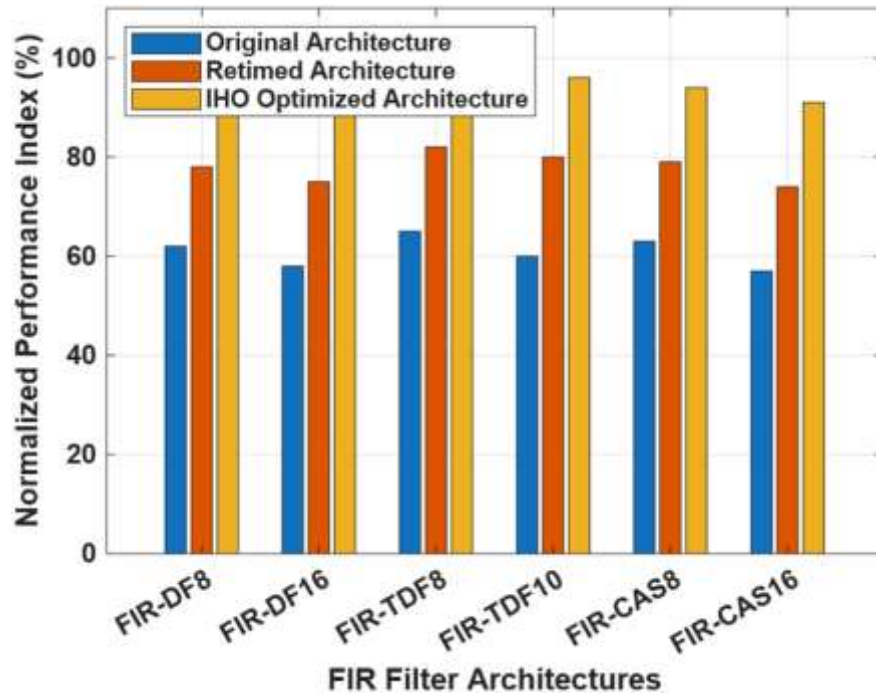


Figure 6. Performance Improvement Analysis of FIR Architectures Before and After IHO-Based Retiming

Overall, the results confirm that the proposed IHO-based retiming framework provides an effective optimization solution for high-performance FIR digital filter architectures. In addition to improving key performance measures (i.e., register utilization, timing, convergence, and frequency), IHO also makes it possible to design very cost-efficient VLSI architectures for high-speed digital signal processing systems.

7. Conclusion

This research presents an intelligent retiming-based optimization framework for improving the performance of FIR digital filter architectures using the Improved Hippopotamus Optimization (IHO) algorithm. The proposed approach effectively combines retiming transformation and metaheuristic optimization to achieve better trade-offs between hardware complexity and timing performance. The obtained results demonstrate that optimized retiming configurations significantly improve the efficiency of FIR filter implementations by reducing unnecessary register utilization and minimizing critical path delay. The performance evaluation confirms that the proposed framework enhances the operating frequency of the FIR-TDF10 architecture from 108.24 MHz to 175.91 MHz, achieving approximately 62.5% improvement compared with the conventional architecture. The optimization process also reduces register requirements and provides improved convergence behaviour, where the IHO algorithm achieves a lower objective function value compared with HO, SchoA, and MFO optimization techniques. The reduced execution time of the proposed algorithm further indicates its effectiveness for solving complex FIR architecture optimization problems. The proposed methodology provides a suitable solution for high-speed and resource-efficient VLSI-based FIR filter design. Future research can focus on extending the framework toward low-power optimization, adaptive FIR filter architectures, ASIC implementation, real-time signal processing applications, and integration with advanced machine learning-based hardware optimization techniques for automated VLSI design.

References

1. Hwan-Rei Lee, Chein-Wei Jen and Chi-Min Liu, "A new hardware-efficient architecture for programmable FIR filters," in *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 9, pp. 637-644
2. R. Jain, P. T. Yang and T. Yoshino, "FIRGEN: a computer-aided design system for high performance FIR filter integrated circuits," in *IEEE Transactions on Signal Processing*, vol. 39, no. 7, pp. 1655-1668, July 1991
3. Y. Matsushita, T. Jibiki, H. Takahashi and T. Takamizawa, "A 32/24 bit digital audio signal processor," in *IEEE Transactions on Consumer Electronics*, vol. 35, no. 4, pp. 785-792, Nov. 1989
4. J. Kernhof, M. A. Beunder, B. Hoefflinger and W. Haas, "High-speed CMOS adder and multiplier modules for digital signal processing in a semicustom environment," in *IEEE Journal of Solid-State Circuits*, vol. 24, no. 3, pp. 570-575, June 1989

5. D. E. Borth, I. A. Gerson, J. R. Haug and C. D. Thompson, "A flexible adaptive FIR filter VLSI IC," in IEEE Journal on Selected Areas in Communications, vol. 6, no. 3, pp. 494-503, April 1988
6. D. R. Rao, S. Paradesi, K. S. Pavani and M. M. H. N, "VLSI Architecture of Adaptive FIR Filter Using Distributed Arithmetic for Low-Power DSP Applications," 2026 Fifth International Conference on Power, Control and Computing Technologies (ICPC2T), Raipur, India, 2026, pp. 985-990
7. S. S, I. S. N, K. K, A. K and K. S, "Design and FPGA Development of a Hybrid Multiplier Architecture with Standard Operation Scheduling for a Low-Power FIR Filter," 2026 Fourth International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 2026, pp. 1-6
8. V. K. Odugu, H. B. Gade, S. Odugu and B. Bapuji, "VLSI-Optimized 2D FIR Filters: Exploiting Rotational Symmetry with Parallel Distributed Arithmetic," 2026 IEEE International Conference on Intelligent Systems, Smart and Green Technologies (ICISSGT), Visakhapatnam, India, 2026, pp. 1-6
9. S. P. Mary, V. A. G. Moorthi, N. Matheshkumar and S. Arunkumar, "Design and Analysis of Dynamic Multiplier Architecture with Application to FIR Filter," 2026 2nd International Conference on Cognitive Computing in Engineering, Communications, Sciences and Biomedical Health Informatics (IC3ECSBHI), Greater Noida, (Delhi NCR), India, 2026, pp. 141-145
10. K. M. Ashraf and G. Naga Swetha, "VLSI Floor Planning Of 2-D FIR With Binary Particle Swarm Optimization," 2025 3rd International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 2025, pp. 273-281
11. N. Kumar, A. V, D. Vaithiyanathan, M. Verma, B. Kaur and P. Verma, "FPGA-Based FIR Filter for 2D Convolution in Image Processing Applications: A Review," 2025 IEEE 4th International Conference on Technology, Engineering, Management for Societal impact using Marketing, Entrepreneurship and Talent (TEMSMET), New Delhi, India, 2025, pp. 1-6
12. V. B. E, R. G, M. Kulkarni, S. N. Parveen, T. R M and V. M, "Design and Implementation of Enhanced Multiplier-Free Architecture for VLSI Signal Processing," 2025 2nd Asia Pacific Conference on Innovation in Technology (APCIT), MYSORE, India, 2025, pp. 1-5
13. H. Zhou, M. Wen, U. -F. Chio and T. Liu, "A Digital 1024 × Decimation Filter for Sigma-Delta Analog to Digital Converter," 2025 International Conference on Frontiers Technology in Circuits and Systems (FTCS), Chongqing, China, 2025, pp. 397-401
14. S. Vats, A. V, D. Vaithiyanathan and M. Verma, "Tape-out Ready Analysis of a Coefficient-based Filter Using High Throughput Multiplier Architecture," 2025 Second International Conference on Pioneering Developments in Computer Science & Digital Technologies (IC2SDT), Delhi, India, 2025, pp. 226-231
15. H. M. L and C. Kavida, "Optimization of Signal Processing - Based Dadda Multiplier for Efficient Multiplication in Digital Signal Processing Applications," 2025 2nd International Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering (ICECONF), Chennai, India, 2025, pp. 1-9
16. B. Bharatha, S. Hemanth, M. U. Kudalagi, V. Udupi and N. Shylashree, "Enhanced FIR Filter Architecture for Fast and Efficient DSP Implementation," 2025 6th International Conference on Communication, Computing & Industry 6.0 (C2I6), Bengaluru, India, 2025, pp. 1-5
17. V. S. Potharaju and V. Saminadan, "Efficient FIR Architecture with Approximate Multipliers and Adders for DSP," 2025 6th International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2025, pp. 392-395
18. A. Karthik, K. Mohanakrishnan, C. Gokul and A. Mohamed Sohaib, "High Speed 2d FIR Filter with Improved Vedic Multiplier Approaches," 2025 6th International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2025, pp. 206-212
19. K. S. Murugesan and M. E. Prathika, "Design and Implementation of Low Power Parallel Transposed FIR Filter Using Modified Gate Diffusion Input Technique," 2025 9th International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 2025, pp. 1594-1599
20. A. D, D. K, H. M, S. S. Gurusami and S. Padmapriya, "Design of a Twin-Mode Adaptive Fir Filter Using Distributed Arithmetic Technique," 2025 International Conference on Next Generation Computing Systems (ICNGCS), Coimbatore, India, 2025, pp. 1-6