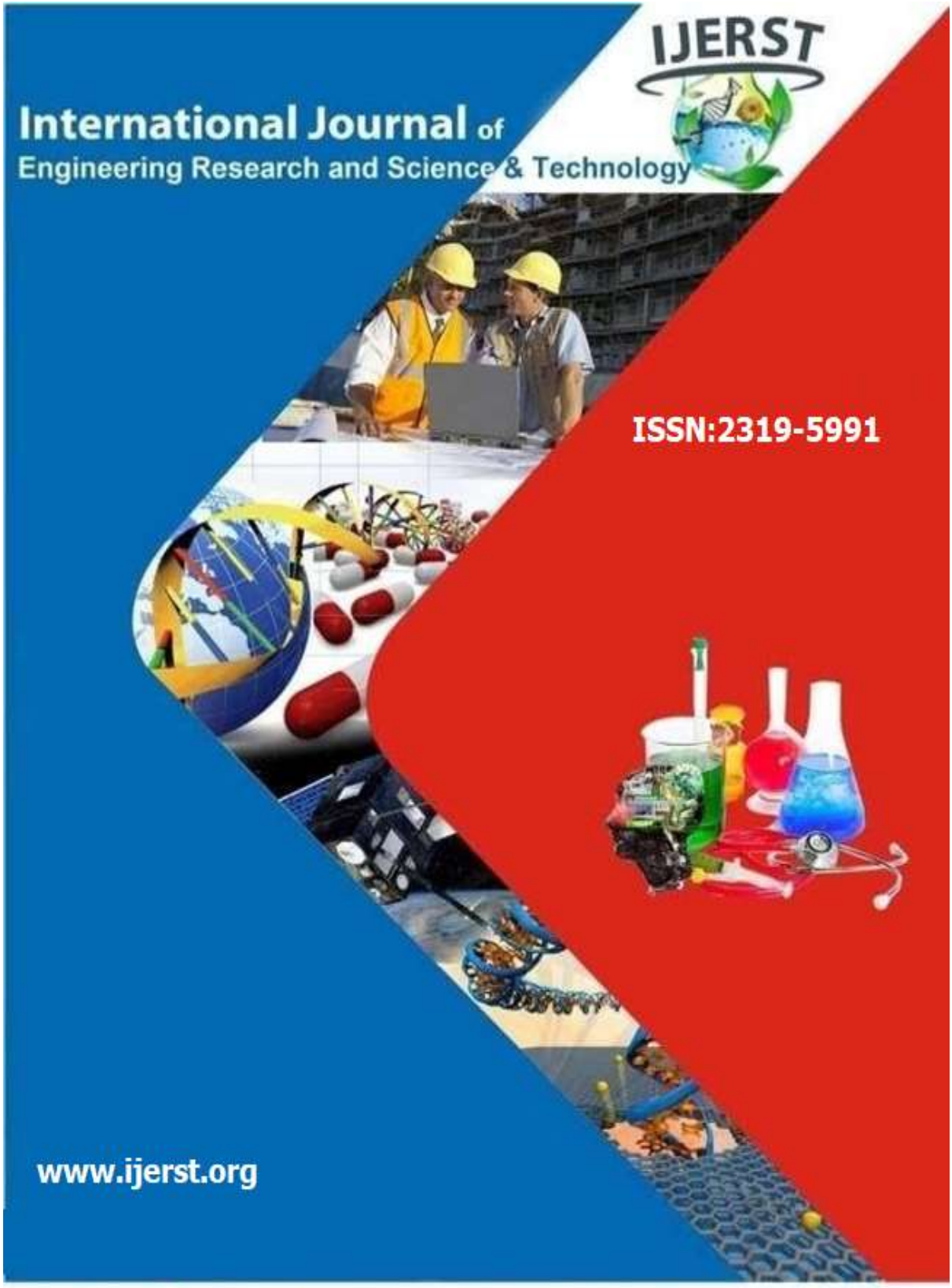
International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025



E-mail: editor@ijerst.org or ijerst.editor@gmail.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

VLSI IMPLEMENTATION OF ADAPTIVE FIR FILTER USING DA WITH LOW POWER AND LOW AREA

[1] Mr.Ikkurthi Rama Koteswara Rao,[2] Gadde Suresh, [3] Tanniru Sivaparvathi
[1]Associate Professor,[2]Professor, [3]Student
Department of Electronics and Communication Engineering
Prakasam Engineering College (Autonomous)

ABSTRACT

The growing demand for efficient digital signal processing (DSP) in wireless communication systems has led to a 30% increase in research on advanced filter designs like Finite Impulse Response (FIR) filters. FIR filters are crucial for reducing noise and enhancing signal quality, with over 50% of modern wireless systems relying on them. However, traditional FIR filters using basic adder-based architectures face limitations in speed and power efficiency, leading to suboptimal performance in high-frequency communication systems. Conventional FIR filters implemented with basic adders suffer from high power consumption and delay, which become significant bottlenecks in real-time communication applications. This paper introduces a novel approach using Distributive Arithmetic (DA)-based Look-Up Table (LUT) architectures with parallel registers for FIR filters, which improves computational efficiency. By leveraging DA-LUT and parallel processing, the proposed FIR filter design reduces latency, power consumption, and area while maintaining high performance, making it ideal for high-speed wireless communication systems.

I. INTODUCTION

1.1 Overview

FIR filters have found numerous applications in radar, sonar, seismology, wireless communications, radio astronomy, acoustics and biomedicine. FIR filter is a finite-length unit impulse response filter, also known as a non-recursive filter, which is the most basic element in a digital signal processing system. It can guarantee arbitrary amplitude-frequency characteristics while having strict linear phase-frequency characteristics, and its unit sampling response is finite, so the filter is a stable system. A FIR filter is a filter structure that can be used to implement almost any sort of frequency response digitally. An FIR filter is usually implemented by using a series of delays, multipliers, and adders to create the filter's output. In signal processing, a FIR filter is a filter whose impulse response is of finite duration, because it settles to zero in finite time. This is in contrast to infinite impulse response (IIR) filters, which may have internal feedback and may continue to respond indefinitely. Finite impulse response models are based on FIR filters, which are a type of a signal processing filter whose impulse response is of finite duration because it settles to zero in finite time.

Properties of FIR Filter:

An FIR filter has a number of useful properties which sometimes make it preferable to an
IIR filter. FIR filters:

- **Require no feedback**: This means that any rounding errors are not compounded by summed iterations. The same relative error occurs in each calculation. This also makes implementation simpler.
- **Inherent stability**: This is due to the fact that, because there is no required feedback, all the poles are located at

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

the origin and thus are located within the unit circle.

- **Phase Issue**: This can easily be designed to be linear phase by making the coefficient

Sequence symmetric; linear phase, or phase change proportional to frequency,

Corresponds to equal delay at all frequencies. This property is sometimes desired for Phase-sensitive applications.

1.2 RESEARCH MOTIVATION

The motivation for researching VLSI-based FIR filters in the context of MIMO-OFDM transmission through a noisy channel stem from the critical importance of achieving robust and fault-tolerant communication systems. In scenarios where audio signals undergo transmission from a source through a channel susceptible to noise, the integrity of the transmitted signal becomes compromised. The integration of FIR filters at the receiver end, implemented as VLSI-based circuits, becomes crucial for mitigating the impact of noise on the audio signal.

The primary research motivation lies in addressing the potential hardware faults that may affect the functionality of the FIR filter within the receiver. As FIR filters are complex circuits that involve various hardware resources, ensuring their reliability and fault tolerance is a significant challenge. The consequences of a faulty FIR filter can be severe, leading to system failure and degradation of audio quality. Investigating and implementing fault-tolerant mechanisms within the VLSI-based FIR filter design becomes paramount to enhance the overall robustness of the communication system.

On the other hand, when the FIR filter operates perfectly, it serves as a powerful tool for noise reduction, resulting in a clean and high-quality audio output. This research motivation, therefore, involves not only designing efficient and high-performance VLSI-based FIR filters but also ensuring their fault tolerance and reliability in real-world applications. The overarching goal is to contribute to the development of communication systems that can reliably deliver clear and noise-free audio signals even in the presence of hardware faults, thereby enhancing the overall resilience and dependability of audio transmission systems.

1.3 Existing System

Implementing Finite Impulse Response (FIR) filters using basic adders and multipliers has its advantages, but it also comes with some notable drawbacks. One significant limitation is the potential for high hardware complexity. FIR filters often require a large number of taps to achieve the desired frequency response, and each tap involves a multiplier and an adder As the number of taps increases, so does the hardware requirement for adders and multipliers, leading to a more complex and resource-intensive implementation. This increased complexity can result in higher power consumption, larger chip area, and higher costs, making it less practical for certain applications where resource constraints are critical.

Furthermore, the speed of operation is a concern when implementing FIR filters with basic adders and multipliers. The extensive number of multiplications and additions in the filter structure can result in a high computational load. This can lead to longer processing times, limiting the filter's ability to meet real-time requirements in certain applications. Optimizing the speed-performance trade-off often involves intricate design considerations, and achieving high-speed operation without compromising on hardware complexity or precision can be a challenging task. As a result, FIR filter implementations using basic adders and multipliers may struggle to meet the

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

demanding performance requirements of certain signal processing applications.

1.4 Problem Statement

Imagine a scenario where you're tasked with designing Finite Impulse Response (FIR) filters for a critical signal processing application. The project demands filters of higher complexity to meet stringent performance requirements. The filters must process incoming signals with precision and efficiency, adding layers of intricacy to the design process.

Moreover, the increased complexity introduces longer signal paths and net delays within the system. These delays can hinder real-time signal processing capabilities, leading to latency issues that may not be acceptable for the application's requirements. The added path delays also raise concerns regarding the system's ability to maintain synchronization and timing integrity, which are crucial in high-speed data processing environments.

In essence, the endeavour to design FIR filters with higher complexity entails navigating a myriad of challenges ranging from resource constraints and timing considerations to power management and coefficient accuracy. Addressing these challenges requires a meticulous and comprehensive approach to ensure the filters meet the stringent requirements of the application while maintaining optimal performance and efficiency.

1.5 Research Objective

The transposed form rearranges the direct form structure to reduce the number of delays and simplify the design. The adders are placed before the multipliers, and the delay elements are shared among different taps, reducing the number of required delay elements. Pipelining involves breaking down the computation into stages, and each stage includes a multiplier and an adder. The pipeline structure helps improve the throughput of the FIR filter at the cost of increased latency. Multiplier less FIR filters aim to minimize or eliminate the need for multipliers by using alternative methods such as shift-and-add or constant coefficient multipliers. These designs often rely on additions, shifts, and simple arithmetic operations to achieve the filtering operation without dedicated multipliers.

1.6 Performance Metrics

1.61 Area metrics

LUT: A LUT in general terms is basically a table that determines what the output is for any given input(s). In the context of combinational logic, it is the truth table. This truth table effectively defines how your combinatorial logic behaves. The LUTs mainly define the behaviour of the combinational logic designed with a VHDL or Verilog code. It simply generates output based on the input combination. An LUT carries a customized truth table for every possible input.

FF: Flip-Flop (FF) and Latch are digital electronic circuits that are used to store information in bits as they have two stable states. One FF or latch can store 1 bit of information. FF is a circuit that can be made to change its state by applying signals to one or more control inputs and will have one or two outputs. Flip-flops are used for synchronizers for asynchronous signals and delay circuits for digital signals as well as counters, frequency dividers, etc.

Buffers: In VLSI interconnect buffers are used to restore the signal level affected by the parasitic. However buffers have a certain switching time that contributes to overall signal delay. A digital buffer (or a voltage buffer) is an electronic circuit element that is used to isolate the input from the output, providing either no voltage or a voltage that is same as the input voltage. It draws very little current and will not disturb the original circuit. It is also called as unity gain buffer or a because it provides a gain of 1, which means it

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

provides at most the same voltage as the input voltage, serving no amplification function.

1.62 Power Metrics

Static Power: Static power, also known as leakage power, is a component of power consumption in VLSI (Very Large-Scale Integration) circuits that arises when transistors are in a steady-state condition, meaning they are not actively switching. Static power is the power consumed when there is no circuit activity or you can say, when the circuit is in quiescent mode. In the presence of a supply voltage, even if we withdraw the clocks and don't change the inputs to the circuit, the circuit will still consume some power, called the static power consumption.

Dynamic Power: Dynamic power is a significant component of power consumption in VLSI (Very Large-Scale Integration) circuits and refers to the power dissipated during the dynamic operation of the circuit, specifically during logic state transitions. It results from the charging and discharging of capacitances in the circuit as signals transition from one logic state to another. Dynamic power consumption is a key consideration in designing energy-efficient integrated circuits. Dynamic Power is the major component of the power dissipated in circuits and also contributes to the peak power. It is a function of the supply voltage, the switching frequency and the output load. Dynamic Power has two major components: the short circuit power and the switching power.

Logic power: Logic power in VLSI (Very Large-Scale Integration) refers to the power consumed by the logic gates and circuits within an integrated circuit during its operation. Logic power consists of both dynamic and static power components. Understanding and managing logic power are crucial aspects of VLSI design to achieve energy-efficient and reliable circuits.

1.63 Delay metrics

Setup Delay: Setup delay refers to the minimum time interval required for the input signal to remain stable before the active edge of the clock signal arrives at the flip-flop or latch. It ensures that the input signal is properly sampled and latched by the sequential element. Setup delay violations occur when the input signal changes too close to the active edge of the clock, potentially leading to incorrect data capture. Managing setup delay is crucial for ensuring reliable operation and timing correctness in VLSI circuits. Techniques like timing analysis, clock tree synthesis, and proper timing constraints are used to address setup delay requirements in VLSI designs.

Hold Delay: Hold delay refers to the minimum time that data must be held stable at the input of a flip-flop or latch after the clock edge to ensure proper capturing of the data. It's essential to prevent data corruption due to setup and hold time violations. Hold time violations occur when data changes too quickly after the clock edge, leading to incorrect data capture. Techniques like proper timing constraints, buffer insertion, and clock skew optimization are used to address hold time violations in VLSI designs.

Logic Delay: Logic delay refers to the time it takes for a signal to propagate through a logic gate or a combination of logic gates. It represents the inherent delay associated with performing a logical operation on input signals and producing the corresponding output signal. Logic delay depends on factors such as gate capacitance, transistor sizes, and technology parameters. Minimizing logic delay is important for achieving high-speed operation and optimizing the performance of VLSI circuits. Techniques such as gate sizing, logic restructuring, and technology selection are used to manage and reduce logic delay in VLSI designs.

1.7 Applications

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

FIR filters are widely used for tasks such as noise reduction, signal equalization, and system identification. When implemented using inner products and parallel accumulations, they offer efficiency and speed advantages.

Digital Signal Processing (DSP): FIR filters are fundamental components in DSP applications. Inner products and parallel accumulations allow for efficient and fasts implementation of FIR filters in real-time signal processing systems.

Equalization: FIR filters can be used to equalize audio signals by compensating for frequency response variations. Implementing FIR filters through inner products and parallel accumulations enables real-time equalization in audio systems.

Noise Cancellation: FIR filters can be employed for noise cancellation in audio signals. The parallel processing capability facilitates the removal of unwanted noise components effectively.

Channel Equalization: In communication systems, FIR filters can be used for channel equalization to compensate for signal distortion introduced by the communication channel. Inner products and parallel accumulations enhance the speed and efficiency of these equalization processes.

Interference Rejection: FIR filters are employed to reject unwanted interference in communication signals, and parallel processing helps achieve this in a timely manner.

1.8 Advantages

Implementing FIR (Finite Impulse Response) filters through inner products and parallel accumulations offers several advantages, particularly in terms of computational efficiency and real-time processing capabilities

Inner Products: Computing the inner product involves multiplying corresponding elements of two vectors and summing the results. This operation is highly parallelizable and can be efficiently implemented using SIMD (Single Instruction, Multiple Data) instructions in modern processors, leading to faster computations.

Parallel Accumulations: The parallel processing of multiple data points simultaneously enhances the overall computational efficiency of FIR filters. This is particularly beneficial for applications requiring real-time processing and low-latency responses.

Low Latency: Inner products and parallel accumulations enable the efficient processing of multiple data points concurrently, reducing latency in real-timesignal processing applications. This is crucial in systems such as audio processing, communication systems, and control systems where timely responses are essential

Parallel Processing: FIR filters implemented through parallel accumulations can take advantage of parallel processing architectures, such as multi-core processors or GPUs. This parallelization allows for simultaneous computation of filter outputs for different input samples, leading to improved throughput.

II. LITERATURE SURVEY

2.1 INTRODUCTION

Implementing FIR (Finite Impulse Response) filters through inner product units and parallel accumulations represents a powerful technique for efficient and high-performance signal processing. In this approach, the input signal is convolved with the filter coefficients using inner product units, which compute the dot product between the input samples and corresponding filter coefficients. These inner product units are typically implemented using dedicated hardware components such as multipliers and adders, allowing for fast and concurrent computation of filter outputs. By leveraging parallelism, multiple inner product units can operate simultaneously on different

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

segments of the input signal, significantly reducing processing time and improving throughput.

The use of inner product units and parallel accumulations in FIR filter implementation offers several advantages in terms of performance, resource utilization, and flexibility. By distributing the computational workload across multiple processing units, inner product units enable efficient utilization of hardware resources while maintaining high throughput. Furthermore, the parallel nature of accumulation enables seamless integration with pipelined architectures and parallel processing pipelines, enabling scalable and customizable FIR filter designs. However, the design and optimization of inner product units and parallel accumulation architectures require careful consideration of factors such as hardware complexity, timing constraints, and resource constraints. Addressing these challenges involves exploring novel design methodologies, algorithmic optimizations, and hardware-accelerated techniques to realize FIR filters with optimal performance and efficiency. In essence, FIR filter implementation through inner product units and parallel accumulations represents a sophisticated yet versatile approach to signal processing, offering a pathway towards high-speed, real-time, and resource-efficient filtering solutions in diverse application domains.

2.2 Related Works

Are primarily based on MIMO-OFDM, the utilization of FIR digital filters is an extremely important element. These filters are vital for a wide variety of features which are finished in the digital sign processor (DSP), along with signal filtering, equalization [1, and noise correction]. Nevertheless, achieving excessive-speed implementation of FIR filters even as simultaneously limiting electricity consumption has become a severe hassle. This is especially genuine whilst contemplating the accomplishments that have been made inside the field of VLSI generation [2] and the sizable software of virtual sign processors.

There had been some of distinctive attempts made to broaden specialized and bendy designs for the implementation of FIR filters on ASIC [3] and FPGA platforms [4]. The MIMO-OFDM-based communication systems where those designs are applicable [5] are particularly relevant. An extensive sort of problems served because the impetus for the development of those architectural designs. Systolic designs, which can be characterized through simplicity, regularity, and modularity, become an attractive architectural paradigm for effectively enforcing computation-extensive DSP programs [6]. These designs provide the ability for high throughput thru the usage of strategies such as pipelining and parallel processing. Although there had been improvements, traditional architectures, that are closely depending on multipliers, have boundaries in terms of the amount of chip area to be had and the range of processing elements (PE) that can be covered [7]. As a end result of its excessive-throughput processing abilties, regularity, and price-powerful computing designs, the multiplier-less DA-based approach has received recognition as a means of addressing those constraints. When it involves FPGA implementation, this method is particularly promising as it uses the LUT structures that are inherent in FPGA logic [8].

When it involves MIMO-OFDM communique systems, FIR filters that have a limited impulse reaction are vital for efficient sign processing. Even though they're correct, the same old methods frequently face problems in terms of the quantity of area hired, the quantity of electricity consumed, and the processing speed. The findings of this look at present a novel approach that makes use of DA-LUTs in an effort to stay away from these limitations.

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

[9] The proposed technique is constructed on the muse of DA, that's a mathematical framework that makes it easier to carry out calculations speedy with the aid of utilising chance distributions. Through the usage of DA-LUTs for arithmetic operations and the illustration of clear out coefficients as possibility distributions, the layout system for FIR filters may be simplified, which in the end results in improved performance in phrases of area performance, electricity consumption, and processing velocity [10]. A nuanced adjustment of precision tiers in DA-LUTs is made feasible with the aid of the adaptability of this method, which strikes a stability among performance and resource utilization.

2.3 Research Gaps

In the realm of FIR (Finite Impulse Response) filters, numerous research gaps persist, underscoring the evolving landscape of digital signal processing. One prominent area of inquiry lies in the implementation of FIR filters across diverse hardware platforms. While FIR filters have been extensively studied and deployed, there remains a need for novel implementation techniques that optimize resource utilization, mitigate power consumption, and address timing constraints. As hardware architectures continue to evolve, researchers face the challenge of developing FIR filter implementations that seamlessly integrate with emerging technologies such as Internet of Things (IOT), edge computing, and embedded systems. Bridging this gap entails exploring innovative hardware design methodologies, leveraging advancements in Field-Programmable Gate Arrays (FPGAs), Application-Specific Integrated Circuits (ASICs), and System-on-Chip (SOC) platforms to deliver efficient and scalable FIR filter solutions.

Another pressing research gap in FIR filter implementation revolves around the quest for adaptive and dynamic filtering mechanisms. Traditional FIR filters operate with fixed coefficients, limiting their adaptability to changing signal conditions or environmental factors. In dynamic environments characterized by varying noise levels, signal characteristics, and system requirements, there arises a need for FIR filters capable of autonomously adjusting their coefficients in real-time. However, designing adaptive FIR filters poses significant challenges, including algorithmic complexity, convergence issues, and computational overhead. Researchers are thus tasked with devising robust adaptive filtering algorithms that strike a balance between adaptability, computational efficiency, and stability, paving the way for FIR filters that can dynamically respond to evolving signal processing demands.

III. EXISTING SYSTEM

3.1 INTRODUCTION

Finite Impulse Response (FIR) filters are vital components in digital signal processing, serving various applications like noise reduction, equalization, and signal conditioning. Unlike Infinite Impulse Response (IIR) filters, FIR filters have a finite duration of impulse response, making them inherently stable and linear phase. This overview explores the architecture, design considerations, implementation techniques, and applications of FIR filters utilizing adders and multipliers, which are fundamental building blocks in their hardware realization. FIR filters process digital signals by convolving the input sequence with a finite-duration impulse response. The output at each time instant is the weighted sum of past input samples. FIR filters are widely used due to their stability, linear phase, and straightforward implementation.

The hardware architecture of FIR filters involves a cascade of adders and multipliers. The input signal passes through a delay line, with each tap corresponding to a multiplier.

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

The multiplier scales the delayed input sample by the respective filter coefficient, and the products are summed up using adders to produce the filter output. Designing FIR filters requires specifying parameters such as passband ripple, stopband attenuation, and transition bandwidth. The filter order, determined by the number of taps, affects the filter's frequency response. Various design methods like windowing, frequency sampling, and optimization techniques are employed to meet the desired specifications. FIR filters can be implemented on different hardware platforms such as FPGAs, ASICs, and DSPs. Performance analysis of FIR filters involves evaluating parameters such as frequency response, phase response, group delay,stability, and computational complexity. FIR filters find applications in various signal processing tasks such as low-pass filtering, high-pass filtering, band-pass filtering, and notch filtering. They are also used for tasks like echo cancellation, adaptive filtering, and system identification in communication systems, audio processing devices, biomedical signal analysis, and image processing. FIR filters with adders and multipliers offer a versatile and efficient solution for digital signal processing tasks. Their stability, linear phase, and ease of implementation make them suitable for a wide range of applications across different domains. Understanding the principles and techniques involved in FIR filter design and implementation is essential for developing effective signal processing solutions tailored to specific requirements. FIR filters with adders and multipliers serve as indispensable tools in the digital signal processing toolkit, enabling the manipulation and analysis of signals in diverse applications with precision and efficiency.

3.2 EXISTING FIR FILTER

Finite Impulse Response (FIR) filters are widely used in digital signal processing to shape and enhance signals. These filters have a finite duration of response to an impulse input and are implemented using various techniques. One such technique involves utilizing array multipliers and ripple carry adders. Array multipliers are hardware units that perform high-speed multiplication, while ripple carry adders are sequential circuits that excel at addition operations. By combining these components, FIR filters can achieve parallel processing and efficient addition, leading to improved speed and accuracy.

Array multipliers are crucial in implementing FIR filters as they perform the multiplication operations required in the filtering process. These multipliers consist of an array of digital circuits, each capable of multiplying two numbers. By employing multiple circuits in parallel, array multipliers enhance the speed and efficiency of multiplication. In the context of FIR filters, array multipliers are used to multiply the input samples with the filter coefficients. The inputs are simultaneously fed into the array multiplier, and the outputs of each multiplication operation are combined to produce the filtered output signal. This parallel processing capability allows for faster and more efficient filtering of signals.

3.2.1. RIPPLE CARRY ADDERS

Ripple carry adders play a vital role in FIR filters by performing addition operations. These adders are sequential circuits that add two binary numbers by carrying over the carry bit from one stage to the next. In the context of FIR filters, ripple carry adders accumulate the multiplied outputs from the array multipliers. The filtered output signal is obtained by summing up these accumulated values. Ripple carry adders, with their simple design and efficient carry propagation mechanism, ensure accurate and reliable addition operations in the filtering process. Their ability to propagate the carry bit efficiently allows for precise

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

summation of the multiplied outputs, further enhancing the filtering process.

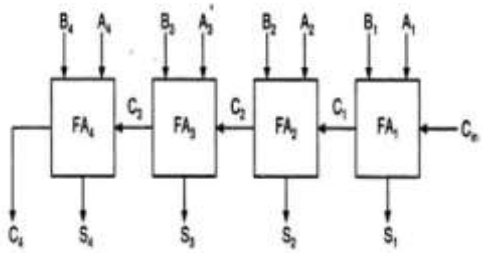


Figure 3.2.1. Ripple Carry Adders

3.2.2. ARRAY MULTIPLIERS

The combination of array multipliers and ripple carry adders in FIR filters showcases the power of parallel processing and efficient addition operations. Array multipliers enable parallel processing of multiplication operations, significantly reducing computational time. The outputs of the array multipliers are then fed into the ripple carry adders, which accumulate these values to produce the final filtered output signal. This integration of parallel processing and efficient addition operations enhances the overall performance and efficiency of FIR filters. Firstly, the parallel processing capability of array multipliers allows for faster execution of multiplication operations, reducing the overall filtering time. This is especially beneficial in real-time signal processing applications where low latency is crucial. Secondly, ripple carry adders provide efficient and accurate addition operations, ensuring the reliability of the filtered output. The combination of these components enables FIR filters to achieve high-speed multiplication and precise summation, resulting in enhanced signal quality.

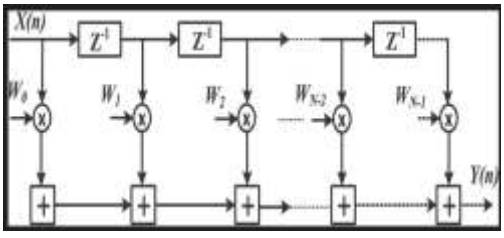


Figure 3.2.2.Array Multipliers

3.3 DRAWBACKS

FIR filters implemented using array multipliers and ripple carry adders offer numerous advantages in digital signal processing, but they also come with notable drawbacks that warrant attention and consideration. Let's delve into these drawbacks in detail:

Hardware Complexity: One significant drawback of FIR filters employing array multipliers and ripple carry adders is the inherent hardware complexity associated with their implementation. As the number of filter taps increases, the hardware requirements for array multipliers and ripple carry adders grow exponentially. This results in larger silicon area usage and increased power consumption, making these filters less feasible for applications with stringent resource constraints, such as embedded systems or portable devices.

Finite Precision Effects: Another significant concern is the susceptibility to finite precision effects and quantization errors. In digital signal processing systems, computations are performed using finite precision arithmetic due to hardware limitations. As a result, multiplication and addition operations may introduce rounding and truncation errors, leading to inaccuracies in the filter output. These errors can accumulate and degrade the overall performance of the FIR filter, particularly in applications requiring high precision filtering, such as audio processing or medical imaging.

Propagation Delay and Latency: The sequential nature of ripple carry adders introduces propagation delays that can significantly impact the latency of FIR filters. As the number of filter taps increases, the propagation delay incurred by ripple carry adders also increases, potentially leading to higher filter latency. In real-time signal processing applications where low latency is

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

crucial, such as telecommunications or control systems, excessive filter latency can be detrimental to system performance.

Limited Throughput: The throughput of FIR filters implemented using array multipliers and ripple carry adders is constrained by the sequential nature of the addition process. While array multipliers enable parallel computation of inner products, the subsequent addition of these products using ripple carry adders occurs sequentially. As a result, the maximum achievable throughput of the filter is limited, especially in applications requiring high-speed signal processing or processing of large datasets.

IV. PROPOSED SYSTEM

4.1 Introduction

Implementing FIR filters through inner product units and parallel accumulations represents a powerful technique for efficient and high-performance signal processing. In this approach, the input signal is convolved with the filter coefficients using inner product units, which compute the dot product between the input samples and corresponding filter coefficients. These inner product units are typically implemented using dedicated hardware components such as multipliers and adders, allowing for fast and concurrent computation of filter outputs. By leveraging parallelism, multiple inner product units can operate simultaneously on different segments of the input signal, significantly reducing processing time and improving throughput. Parallel accumulation is another key aspect of FIR filter implementation using inner product units. After computing the dot products between input samples and filter coefficients, the resulting products are accumulated in parallel to generate the filtered output samples. This parallel accumulation process involves summing the products from different inner product units in a coordinated manner, often utilizing dedicated adder trees or pipelined structures to achieve high-speed accumulation. Parallel accumulation not only enhances computational efficiency but also facilitates real-time processing of high-frequency signals by minimizing latency and maximizing throughput. Moreover, the scalability of parallel accumulation allows for the implementation of FIR filters with varying tap lengths and processing requirements, making it suitable for a wide range of signal processing applications.

The use of inner product units and parallel accumulations in FIR filter implementation offers several advantages in terms of performance, resource utilization, and flexibility. By distributing the computational workload across multiple processing units, inner product units enable efficient utilization of hardware resources while maintaining high throughput. Furthermore, the parallel nature of accumulation enables seamless integration with pipelined architectures and parallel processing pipelines, enabling scalable and customizable FIR filter designs. However, the design and optimization of inner product units and parallel accumulation architectures require careful consideration of factors such as hardware complexity, timing constraints, and resource constraints. Addressing these challenges involves exploring novel design methodologies, algorithmic optimizations, and hardware-accelerated techniques to realize FIR filters with optimal performance and efficiency. In essence, FIR filter implementation through inner product units and parallel accumulations represents a sophisticated yet versatile approach to signal processing, offering a pathway towards high-speed, real-time, and resource-efficient filtering solutions in diverse application domains.

4.2 Proposed Methodology

The proposed DA-LUT based FIR filter out, that is introduced for MIMO-OFDM

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

applications, is subjected to a comprehensive evaluation on this section. The system architecture of the proposed method is depicted in Figure 1. This methodology makes use of parallel registers, FIR filter coefficients, and the DA-LUT so that it will attain high-pace and green filtering of input facts. Through the usage of distribution arithmetic for arithmetic operations, the DA-LUT is a critical component in the accomplishment of the intention of improving computational efficiency. The recursive nature of the filtering operation is ensured by using the feedback loop that extends from the filtered output to the accumulator. This feedback loop is an enormous contributor to the general effectiveness of the MIMO-OFDM-FIR clear out.

In addition, the coefficients of the FIR filter, which are represented by means of the symbol h(n), are initialized. The subsequent filtering operations depend on these coefficients, which play a sizeable element in defining the traits of the FIR filter out and are important for the filtering operations. Next, gift the DA-LUT, that's a critical factor of the technique that has been proposed. Through the storage of pre-calculated values associated with opportunity distributions linked to the FIR filter out coefficients, the DA-LUT makes it feasible to perform arithmetic operations in an effective manner. Through the usage of distribution mathematics, the computation system is simplified, which ends up in advantages in terms of both pace and the efficiency with which sources are applied.

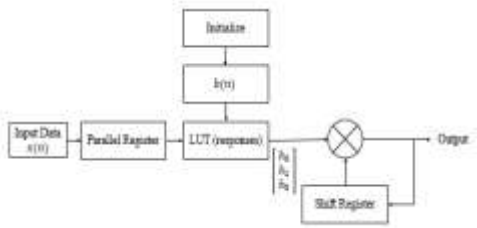


Figure 4.1. Proposed DA-LUT based FIR filter block diagram.

4.3 Parallel Registers

The storing and retrieval of binary statistics in a parallel fashion is accomplished thru the utilization of digital circuits that are known as parallel registers. A wide variety of packages regularly employ them. Some examples of these packages include statistics buffering, information delivery, and facts synchronization, to call just a few of the programs that make use of them. As part of this comprehensive look at, we will check out the fundamentals of the way parallel registers perform, as well as the several bureaucracy, programs, benefits, and disadvantages of these registers.

A collection of turn-flops, commonly of the D-type, which are connected to one another to keep binary facts in a parallel fashion is referred to as a parallel sign up. The potential of every turn-flop is good enough for the storage of a parallel little bit of information. Parallel connections are made among the data inputs and facts outputs of the turn-flops. This allows the facts inputs to be received, and in a similar way, the records outputs of the turn-flops are related in parallel so that the records outputs can be delivered. Both eventualities involve the facts inputs and data outputs of the flip-flops being linked in parallel. This lets in for the records inputs and the information outputs to be received concurrently. To save statistics in a parallel check in, the data this is being input must first be applied simultaneously to the enter pins of each flip-flop this is contained within the register. Following the activation of a control sign, including a clock pulse, the information is secured onto the turn-flops so that you can get prepared for in addition processing. As a result of this, it is viable for the flip-flops to concurrently save all the bits that represent the records this is being acquired. Similarly, to retrieve the facts that has been saved within the parallel sign up, it's miles essential to

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

retrieve the facts that has been saved in each turn-flop concurrently via connecting the output pins of the flip-flops collectively.

4.4 DA-LUT

For all intents and purposes, a DA-LUT is nothing more than memory that stores values that have been precalculated in advance for the numerous possible combinations of input data. Because of this, the table can process the data in a timely and accurate manner. This table presents the DA-LUT operation table that has been proposed. Using this method, the DA-LUT is used to perform a preliminary calculation on all of the inputs and then store the results. The input of the filter is directed toward the DA-LUT, which is the one that is being addressed. When referring to a filter that has four inputs, the term "four tap" signifies not only the number of coefficients that the filter possesses, but also the number of inputs that the filter possesses in addition to the address bit for the DA-LUT. In other words, the term encompasses all of these aspects of the filter. The reason for this is that the phrase "four tap" simultaneously represents all of these things. As a result of the inputs that it receives, each location generates a distinct output in response to those conditions. Valid inputs for this filter include values ranging from 0 (0000) to 15 (1111), which are listed in the range. When utilizing this method, it is not difficult to compute the result for each element of the input that is provided. For example, if the value that is being input is 1011, the output value will be 1.h0 plus 0; h1 plus 1; h2 plus 1; and h3 will equal h0 plus h2 plus 1.h3. If the value that is being input is 1111, the value that will be output will be h0 plus h1 plus h2 plus h3. When the value 0101 is used as the input, the value that is produced will be h1 plus h3. It is possible that the output value will be h0 plus h2 if the value that input is is 1010. A high-level input coefficient is incorporated into the system, as indicated by this term. It is not necessary for us to perform any kind of mathematical calculation to determine that there are sixteen outputs for every matching input.

Table. 1. Third order DA-LUT performance table.

Address	Data
0000	0
0001	h_3
0010	h2
0011	h2 + h3
0100	h1
0101	h1 + h3
0111	h1 + h2 + h3
1000	h0
1001	h0 + h3
1010	h0 + h2
1011	h0 + h2 + h3
1100	h0 + h1
1101	h0 + h1 + h3
1110	h0 + h1+ h2
1111	h0 + h1 + h2 + h3

4.5 Distributive Arithmetic

A bit serial procedure known as DA is used in the computation of FIR filter operations such as correlation, convolution, and inner products. These are all examples of DA. The relevance of DA lies in the fact that it can be mechanized with a high degree of efficiency. When the filter order in a simple DA implementation is raised, the LUT size also rises, which influences both the amount of space and the performance of the implementation. Using the LUT idea, both the performance and the size of the DA was improved. In this case, a signed data value of $(1, 1)$ is used rather of the binary data value of $(1, 0)$. The following expression describes the output of a FIR filter with a 'N' order: y(n). Let us have a look at the inner

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

product of the two vectors d and x, which is given as

$$y = \sum_{k=1}^{K} d_k x_k \quad (1)$$

Here, d_k is a constant coefficient, x_k is the input signal, and K is the total number of words in the input. It is mathematically impossible for the value of the scaled binary integer x_k, which has digits that are the two's complement of one another and N bits, to be more than one. This number has digits that are the two's complement of one another. It is also possible to write x_k as:

$$x_k = -b_{k0} \sum_{n=1}^{N-1} b_{kn} 2^{-n} \quad (2)$$

$$x_k = \{b_{k0}, \dots\dots\dots\dots b_{k(N-1)}\} \quad (3)$$

Here, b_{kn} is the Nth bit of the x_k value that is being represented. The extended form of y was found by substituting equation (3) into equation (2), which results in the following:

$$y = \sum_{k=1}^{K} d_k \left[-b_{k0} + \sum_{k=1}^{K} d_k (-b_{k0})\right] \quad (4)$$

In mathematical notation, the expression for the inner product is often represented by the equation (3). If DA rearrange the way in which the totals are tallied, DA were able to arrive at the equation that is shown below at the end.

$$y = \sum_{k=1}^{K} \left[\sum_{k=1}^{K} d_k b_{kn}\right] 2^{-n} + \sum_{k=1}^{K} d_k (-b_{k0}) \quad (5)$$

In the form of x_k, let us write it out.

$$x_k = \frac{1}{2}[x_k - (-x_k)] \quad (6)$$

It is possible to describe the form of $-x_k$ that corresponds to the two's complement using the symbol.

$$-x_k = -\bar{b}_{k0} + \sum_{n=1}^{N-1} \bar{b}_{kn} 2^{-n} + 2^{-(N-1)} \quad (7)$$

By solving equation (6) using x_k and $-x_k$ as inputs:

$$x_k = \frac{1}{2}\left[-\left(b_{k0} - \bar{b}_{k0}\right) + \sum_{n=1}^{N-1}\left(b_{kn} - \bar{b}_{kn}\right) 2^{-n} - 2^{-(N-1)}\right] \quad (8)$$

For the sake of notational simplicity, let us assume.

$$S_{kn} \begin{Bmatrix} b_{kn} - \bar{b}_{kn}, n \neq 0 \\ \left(b_{kn} - \bar{b}_{kn}\right), n \neq 0 \end{Bmatrix} \quad (9)$$

Here,

$$P(b_n) = \sum_{k=1}^{K} \frac{d_k}{2} S_{kn} \quad (10)$$

$$P(0) = -\sum_{k=1}^{K} \frac{d_k}{2} \quad (11)$$

Here, $P(0)$ is the only word-initial condition register, while $P(b_n)$ may take on a total of $2^{(k-1)}$ different values depending on the context. A word-initial condition register is denoted by the symbol "$P(0)$.". Because of this, the total amount of LUT that was used in DA has been reduced to merely 2^{k-1} rather of the entire 2^k.

V. RESULTS AND DISCUSSIONS

This part of the article provides a comprehensive simulation analysis of the DA-LUT based FIR filter that has been proposed. Simulations are carried out in this location with the assistance of the Xilinx-Vivado software tool. The result of the simulation is depicted in Figure 5.1, which includes the sign, clk, and rst as primary inputs, x as data input, and h0, h1, h2, and h3 as impulse coefficient inputs of the DA-LUT. Furthermore, the data out output is the final output of the FIR filter that is based on DA-LUT.

Figure 5.1. Simulation output

The resource utilization on the FPGA for the proposed DA-LUT-FIR filter is visually represented in Figure 5.2, which provides a comprehensive snapshot of the situation. Programable logic blocks, also known as

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

LUTs, are used to implement combinational logic functions. Out of the 78,600 LUTs that are available, 61 are currently being utilized. The number of Flip-Flops (FFs), which are used to store binary information, is calculated to be twelve out of the total of 157,200 that are available. Out of the total of 250 available blocks, 25 are used by input/output blocks (IoBs), which are responsible for interacting with external devices. In addition, the figure illustrates the utilization of Phase-Locked Loop (BUFG) components, which includes the utilization of one out of every 32 components. These values, when taken together, provide a quantitative measure of the proposed DA-LUT-FIR filter's effective utilization of FPGA resources.

Resource	Utilization	Available	Utilization...
LUT	61	78600	0.08
FF	12	157200	0.01
IO	25	250	10.00
BUFG	1	32	3.13

Figure 5.2. Design summary.

Figure 5.3 provides a precis of the setup time, which sheds light at the timing traits of the machine this is being proposed. There is a total put off 6.139 nanoseconds, which includes the time this is required for logic operations (3.192 nanoseconds) and the internet put off (2.947 nanoseconds). The time it takes for indicators to become stable earlier than they are sampled is known as the setup time, and those figures offer crucial insights into the temporal overall performance of the DA-LUT-FIR filter.

Figure 5.3. Setup Time summary.

The proposed DA-LUT-FIR filter out is displayed in Figure 5.5, which affords a comprehensive evaluation of the electricity consumption of the filter out. Dynamic and static components are the two classes which can be used to categorise the power values. In order to don't forget the variations in sign interest, dynamic strength is in addition broken down into three classes: sign power (0.430uw), good judgment power (0.408uw), and IoB strength (4.268uw). Additionally, the static energy component is furnished in its personal proper (0.115uw). It has been decided that the total dynamic energy is five.106 uw, and while that is added to the static power, the total strength consumption for the device is five.22 uw. These metrics offer valuable insights into the strength efficiency and energy traits of the DA-LUT-FIR clear out that has been proposed, which facilitates in determining whether or not or no longer it is appropriate for practical programs.

Figure 5.4. Hold Time summary.

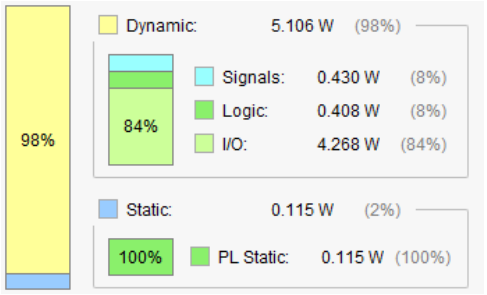


Figure 5.5. Power summary

The area utilization was compared between the existing filter and the proposed filter in Table 1, which can be found below. The number of LUTs has been decreased by approximately 29.07%, which indicates that the implementation of combinational logic functions has become more efficient. Furthermore, there is a significant reduction in FFs, which is approximately 65.71%, which suggests that the storage of binary information has been simplified overall. IoBs also

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

experience a significant decrease of approximately 55.36%, which indicates that their interfacing with external devices has improved. Furthermore, the size of the BUFG components has been decreased by approximately 75%.

Table. 1. Area performance comparison.

Resource	Existing Filter	Proposed Filter
LUTs	86	61
FFs	35	12
IoBs	56	25
BUFG	4	1

The dealy measurement is compared in Table 2, which shows that the proposed filter is superior to the existing filter in a number of different aspects. The logic setup delay has experienced a significant reduction of approximately 41.89%, which indicates that the logic operation has become more efficient. A greater modest reduction of about eight.09% inside the net setup postpone is validated, which suggests that the efficiency of the internet connections has been improved. The general setup put off, which is the overall quantity of time required for signal stabilization, is decreased by using approximately 29.32%, demonstrating the effectiveness of the proposed clear out in accomplishing stability in a greater expedient way. There is a enormous reduction in preserve instances, which might be critical for keeping sign stability. The logic preserve put off has been decreased via about 53.Eighty four%, the net keep put off has been reduced by way of about 71.Sixty five%, and the whole preserve put off has been decreased by way of an amazing 83.64%. This suggests that the proposed clear out has led to an overall improvement in temporal overall performance, as the overall put off has been reduced via about 25.Fifty one percent.

Table. 2. Delay performance comparison.

Dealy (ns)	Existing Filter	Proposed Filter
Logic Setup Delay	5.492	3.192
Net Setup Delay	3.207	2.947
Total Setup Delay	8.7	6.139
Logic Hold Delay	0.672	0.313
Net Hold Delay	0.451	0.128
Total Hold Delay	1.10	0.185
Total Delay	9.8	6.32

The power performance comparison reveals that the proposed filter has made significant improvements in terms of the amount of power it consumes. The power of the signal is reduced by approximately 44.86%, which indicates that the signal transitions are being handled more effectively. There is a decrease in the amount of power that is consumed by logic operations, which is reflected in the Logic Power experiencing a reduction of approximately 25.32%. IoB power consumption has been significantly cut by approximately 35.29 percent, demonstrating an increase in the effectiveness of the interface with external devices. The total dynamic power, which is the power that is consumed as a result of dynamic signal activity, is reduced by approximately 35.11%, which highlights the energy efficiency of the proposed filter. Static Power, which measures the amount of power that is consumed by the circuit when it is not in use, is significantly reduced by approximately 66.96%. The proposed filter is effective in reducing the amount of power that is required, as evidenced by the fact that the Total Power Consumption is reduced by approximately 38.21% overall.

Table. 3. Power performance comparison.

Power (uw)	Existing Filter	Proposed Filter

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

Signal Power	0.781	0.430
Logic Power	0.562	0.408
IoB Power	6.81	4.268
Total Dynamic Power	8.1	5.106
Static Power	0.342	0.115
Total Power Consumption	8.442	5.22

VI. CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

The motive of this study is to investigate the optimization of FIR filters in MIMO-OFDM communique structures. Our technique, which makes use of DA-LUTs with parallel registers, is targeted on improving the efficiency and throughput of FIR filters, which are vital components in decreasing interference problems inside these structures. As a end result of the incorporation of parallel registers, concurrent computation and accumulation of filter taps are made less difficult. This results in a discount in the important route delay and makes it possible to operate FIR filters at better frequencies. The technique that has been proposed makes use of DA, which is a way that allows the effective representation and manipulation of numbers that have distributions that are not uniform. The inherent statistical houses of sign data can be exploited to achieve more suitable precision and reduce quantization mistakes while in comparison to traditional LUTs. This function is specifically useful in sign facts as it lets in for the utilization of these homes. The results of our research show that the DA-LUT design is advanced to the conventional FIR filters. This is tested via a complete performance contrast. A wide variety of things, along with quantization errors, filter out response accuracy, and hardware complexity, are covered in the evaluation criteria. The findings highlight the benefits of our approach, which makes a full-size contribution to the optimization of MIMO-OFDM structures by means of enhancing the effectiveness and precision of FIR filters thru the software of present-day digital sign processing techniques.

6.2 FUTURE SCOPE

The evolution of hardware architectures forms the bedrock upon which future FIR filters will stand. With the proliferation of Field-Programmable Gate Arrays (FPGAs), Application-Specific Integrated Circuits (ASICs), and System-on-Chip (SoC) platforms, the future beckons towards bespoke hardware implementations tailored to FIR filter operations.

Future scopes encompass: Parallel Processing and Pipelining: Exploiting inherent parallelism, future architectures may incorporate advanced parallel processing techniques and pipeline stages, further accelerating filtering operations while optimizing resource utilization.

Hardware Accelerators and Co-Processors: Integration of dedicated accelerators and co-processors can offload computational burdens, enhancing performance and efficiency across diverse application scenarios.

Customizable Architectures: Embracing reconfigurable architectures allows for dynamic adaptability to varied signal processing requirements, catering to a spectrum of applications with flexibility and versatility.

Algorithmic Optimizations and Techniques: Algorithmic innovations serve as catalysts for unlocking the full potential of FIR filters, ushering in new paradigms of efficiency and performance.

Optimized Convolution Algorithms: Fast convolution algorithms and frequency-domain filtering techniques promise to reduce computational complexity, enhancing efficiency in real-time signal processing applications.

Sparse FIR Filtering: Leveraging sparsity in filter coefficients can significantly reduce

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

computational overhead and memory requirements, paving the way for more resource-efficient FIR filtering implementations.

Machine Learning Integration: The fusion of machine learning models with FIR filtering enables adaptive filtering, coefficient optimization, and noise cancellation, empowering FIR filters to dynamically adapt to changing signal environments.

REFERENCES

[1]. Di Meo, Gennaro, et al. "A Novel Low-Power High-Precision Implementation for Sign–Magnitude DLMS Adaptive Filters." Electronics 11.7 (2022): 1005.

[2]. Shrivastava, Prabhat Chandra, et al. "An Efficient Block-Based Architecture for Reconfigurable FIR Filter Using Partial-Product Method." Circuits, Systems, and Signal Processing 41.4 (2022): 2173-2185.

[3]. Thamizharasan, V., and N. Kasthuri. "FPGA implementation of high performance digital FIR filter design using a hybrid adder and multiplier." International Journal of Electronics (2022): 1-21.

[4]. Kumar, Gundugonti Kishore, et al. "Area-, Power-, and Delay-Optimized 2D FIR Filter Architecture for Image Processing Applications." Circuits, Systems, and Signal Processing 42.2 (2023): 780-800.

[5]. Ezilarasan, M. R., J. Britto Pari, and Man-Fai Leung. "Reconfigurable Architecture for Noise Cancellation in Acoustic Environment Using Single Multiply Accumulate Adaline Filter." Electronics 12.4 (2023): 810.

[6]. Sharada, K. A., et al. "High ECG diagnosis rate using novel machine learning techniques with Distributed Arithmetic (DA) based gated recurrent units." Microprocessors and Microsystems 98 (2023): 104796.

[7]. Lakshmi, Vijaya, Vikramkumar Pudi, and John Reuben. "Inner product computation in-Memory using distributed arithmetic." IEEE Transactions on Circuits and Systems I: Regular Papers 69.11 (2022): 4546-4555.

[8]. Diouri, Omar, et al. "Comparison study of hardware architectures performance between FPGA and DSP processors for implementing digital signal processing algorithms: Application of FIR digital filter." Results in Engineering 16 (2022): 100639.

[9]. Vijetha, K., and Rajendra Naik. "Low power low area VLSI implementation of adaptive FIR filter using DA for decision feed back equalizer." Microprocessors and Microsystems 93 (2022): 104575.

[10]. Wang, Xingyang, and Yutong Zhu. "Intelligent Art Design Management Based on Wireless Communication Microprocessor and Mobile Internet." Wireless Communications and Mobile Computing 2022 (2022).

[11]. Ganjikunta, Ganesh Kumar, Mahaboob Basha Mohammed, and Inayatullah Khan Sibghatullah. "Energy Efficient FIR Filter Design Using Distributed Arithmetic." Journal of Shanghai Jiaotong University (Science) (2022): 1-5.

[12]. Magesh, V., and N. Duraipandian. "Implementation of Programmable Finite Impulse Response Filter Using Modified Computation Sharing Multiplier for Hearing Aids." Wireless Personal Communications 129.1 (2023): 255-270.

[13]. Kaur, Mandeep, Ranjit Kaur, and Narinder Singh. "A novel hybrid of

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

chimp with cuckoo search algorithm for the optimal designing of digital infinite impulse response filter using high-level synthesis." Soft Computing (2022): 1-25.

[14]. Karthick, R., et al. "Design and analysis of linear phase finite impulse response filter using water strider optimization algorithm in FPGA." Circuits, Systems, and Signal Processing 41.9 (2022): 5254-5282.

[15]. Odugu, Venkata Krishna, C. Venkata Narasimhulu, and K. Satya Prasad. "A novel filter-bank architecture of 2D-FIR symmetry filters using LUT based multipliers." Integration 84 (2022): 12-25.

[16]. Zhang, Jinzhong, et al. "Adaptive infinite impulse response system identification using an enhanced golden jackal optimization." The Journal of Supercomputing (2023): 1-26.

[17]. Zhang, Ling, Chaolin Rao, and Xin Lou. "Low-power Reconfigurable FIR Filter Design Based on Common Operation Sharing." IEEE Transactions on Circuits and Systems II: Express Briefs (2023).

[18]. Kumar, J. Charles Rajesh, and D. Vinod Kumar. "Energy-efficient, high-performance and memory efficient FIR adaptive filter architecture of wireless sensor networks for IoT applications." Sādhanā 45.4 (2022): 248.

[19]. Reddy, Kasarla Satish, et al. "Efficient FPGA Implementation of an RFIR Filter Using the APC–OMS Technique with WTM for High-Throughput Signal Processing." Electronics 11.19 (2022): 3118.

[20]. Li, Zhen, et al. "Adaptable Approximate Multiplier Design Based on Input Distribution and Polarity." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 30.12 (2022): 1813-1826.