International Journal of Electronics and Communication Engineering and Technology (IJECET)
Volume 16, Issue 2, May-August 2025, pp. 1-16, Article ID: IJECET_16_02_001
Available online at https://iaeme.com/Home/issue/IJECET?Volume=16&Issue=2
ISSN Print: 0976-6464; ISSN Online: 0976-6472; Journal ID: 3587-7166
Impact Factor (2025): 13.44 (Based on Google Scholar Citation)
DOI: https://doi.org/10.34218/IJECET_16_02_001

THE HDL IMPLEMENTATION OF HIGH SPEED FIR FILTER USING SA-COMPRESSORS AND OPTIMIZED APPROXIMATE ADDERS USING FPGA

G. Krishna Veni
Assistant Professor, Dept. of ECE,
Bapatla Women's Engineering College, Bapatla, Andhra Pradesh, India.

G. Srinivas Rao
Professor, Department of ECE, Bapatla Women's Engineering College,
Bapatla, Andhra pradesh, India.

D. Vishalakshi Neelima, A. Sunayana, B. Revathi
UG Student, Department of ECE, Bapatla Women's Engineering College,
Bapatla, Andhra pradesh, India.

ABSTRACT

Finite Impulse Response (FIR) filters are widely used in digital signal processing (DSP) applications due to their stability and linear-phase characteristics. However, their high computational complexity, primarily due to extensive multiplication and addition operations, can limit their speed and efficiency. In this paper we present a high-speed FIR filter design by integrating segmented arithmetic (SA) based compressors and approximate adders to enhance computation speed. The proposed approach replaces conventional adder trees with segmented arithmetic (SA) based compressor circuits (such as 4:2 and 7:3 compressors), which efficiently reduce partial

products during multiplication. Additionally, optimized approximate adders, are employed to further minimize latency and hardware complexity. In this we mainly focus on the design of 3-tap FIR filter and the proposed architecture is implemented on FPGA, which is of Target device- XC3S50-5pq208 belonging to the Spartan 6 family using Xilinx 14-7 ISE software. The Simulation results are obtained and analyzed that the proposed architecture achieves lower power consumption 2.3mv with critical path delay 17.450ns, and area utilization 28,500µ m^2.

Keywords: Carry Select Adder (CSA), FIR Filter, Digital Signal Processing, VLSI Design, Segmented Arithmetic (SA) Compressors, Approximate Adders.

Cite this Article: G. Krishna Veni, G. Srinivas Rao, D. Vishalakshi Neelima, A. Sunayana, B. Revathi. (2025). The HDL implementation of high speed FIR filter using SA-compressors and optimized approximate adders using FPGA. International Journal of Electronics and Communication Engineering and Technology (IJECET), 16(2), 1–16.

https://iaeme.com/MasterAdmin/Journal_uploads/IJECET/VOLUME_16_ISSUE_2/IJECET_16_02_001.pdf

I. Introduction

Finite impulse response (FIR) filters play a crucial role in digital signal processing (DSP) applications, including audio processing, image enhancement, and wireless communication. Traditional FIR filter implementations involve high computational complexity due to the large number of multiplications and additions required. In this work, we propose an enhanced FIR filter design that incorporates SA-based techniques with compressors and approximate adders to achieve high speed and low power consumption. Compressors, such as 4:2 and 3:2, are employed to optimize addition operations, reducing delay and improving parallelism. Additionally, approximate adders further reduce power consumption by trading off minimal computational accuracy for enhanced efficiency. The proposed design significantly improves performance while maintaining acceptable filtering accuracy, making it suitable for energy-constrained and real-time applications. The key advantage of FIR filters is that their impulse response is of finite duration, and their output is solely dependent on a finite number of previous input samples, making them computationally straightforward. However, as the demand for high-speed, real-time signal processing grows in fields like wireless communication, radar systems, and multimedia applications, the need for optimized hardware implementations of FIR filters has become increasingly critical. The performance of FIR filters

is largely dependent on the underlying hardware used for arithmetic operations such as multiplication and addition. In traditional FIR filter designs, the computational complexity increases with the number of filter taps (coefficients), leading to high power consumption, longer processing times, and larger hardware area. This makes them less suitable for high-performance and low-power applications where efficiency is paramount. One promising approach to optimize FIR filter design is the use of compressors and advanced additive circuits. Compressors are specialized circuits that efficiently reduce the number of partial products in arithmetic operations, such as multiplication, by compressing multiple bits into fewer bits. By using compressors, adders like Carry Save Adders (CSA) and higher-order structures such as Wallace Trees, the number of adders required for summing intermediate results can be significantly reduced, resulting in faster computation times. In addition to segmented arithmetic (SA) compressors, we use the optimized approximate adder circuits,which plays a crucial role in enhancing the overall speed and power efficiency of FIR filter designs. Various types of adders, such as Approximate Parallel Prefix Adders and Approximate Parallel Prefix Adder-Low Fanout can be leveraged to minimize the delay and area required for summing partial products and filter outputs. The combination of compressors and efficient adders leads to a more streamlined and high-performance FIR filter architecture, particularly suited for applications that demand real-time processing with minimal latency. This research investigates the design of fast FIR filters using different compressor architectures and adder circuits, focusing on improving the speed, power consumption, and area efficiency of the filter implementation. The paper explores various compressor and adder combinations to identify the most suitable hardware configurations for FIR filters in high-performance and low-power DSP systems. By implementing and comparing these techniques, the research aims to present a comprehensive approach to optimizing FIR filter designs for a wide range of modern DSP applications. Furthermore, the research evaluates the trade-offs involved in using different adder and compressor designs, assessing the impact of each on the overall filter performance in terms of speed, power, and area. The findings of this work contribute to the ongoing efforts to enhance the efficiency of FIR filters and provide insights into the design of high-performance digital systems that balance computational speed, power consumption, and hardware complexity. The full article usually follows the standard structure: i. Introduction, ii. Literature survey, iii. The implementation of FIR filter using conventional technique, iv. Design of fir filtr using proposed methodology, v. Hardware implementation and simulation results, vi. Conclusion, vii. Future scope, viii. Acknowledgement.

II. LITERATURE SURVEY

[2.1] "Low-Power and Area-Efficient Carry Select Adder" by A. R. Al-Ali, M. N. S. Swamy, and S. M. S. K. Ula (2006):

The carry select adder (CSLA) is widely used in digital systems due to its fast computation time. However, conventional CSLA designs consume more power and occupy larger areas due to redundant logic. This paper proposes a modified CSLA architecture that reduces area and power consumption while maintaining high-speed operation. The proposed design eliminates duplicate adder blocks by replacing them with a Binary to Excess-1 Converter (BEC), significantly reducing the number of logic gates. Comparisons with conventional and existing CSLA designs show that the proposed architecture achieves considerable reductions in power, area, and delay, making it suitable for low-power and high-performance applications**.**

[2.2] "Design and Implementation of Carry Select Adder without Using Multiplexers" by V. K.Jain(2012):

The Carry Select Adder (CSLA) is a widely used arithmetic unit in high-speed digital circuits due to its ability to improve addition speed. However, conventional CSLA designs involve multiplexer-based carry selection, which contributes to increased hardware complexity and power consumption. This paper presents a novel CSLA architecture that eliminates multiplexers, thereby reducing circuit complexity, power consumption, and area overhead. The proposed design leverages an optimized logic structure to efficiently compute sum and carry outputs, achieving comparable or superior performance compared to traditional CSLA implementations. Simulation and synthesis results validate the efficiency of the proposed design, making it a viable alternative for low-power, high-speed applications in digital signal processing and VLSI systems.

[2.3] "High-Performance Hardware Design of Compressor Adder in DA-Based FIR Filter for Digital Hearing Aid Application" by Su Peng (2020):

This paper presents a low-complexity design of a digital FIR filter for digital hearing aid applications. The study focuses on the hardware design of compressor adders in segmented Arithmetic (SA)-based FIR filters, aiming to enhance performance while reducing power consumption and area.

[2.4] "Power and Area Efficient FIR Filter Using Radix-2r Multiplier for Denoising Electrooculography Signals" by [Authors Not Specified] (October 2024):

This article introduces a novel FIR filter using a Radix-2r multiplier combined with 4:2 and 3:2 compressors for denoising Electrooculography (EOG) signals. The proposed design

achieves significant improvements in power efficiency and area optimization, making it suitable for biomedical signal processing applications.

[2.5] "Design and Implementation of 6-Tap FIR Filter Using MAC for Low Power Applications" by [Authors Not Specified] (November 2014):

This study focuses on the design and implementation of a 6-tap FIR filter using a Multiply-Accumulate (MAC) unit for low-power applications. The design incorporates Wallace tree multipliers and binary adders to construct the MAC unit, achieving reductions in power consumption and area.

III. THE IMPLEMENTATION OF FIR FILTER USING CONVENTIONAL TECHNIQUE

The existing method primarily relies on Multiply-Accumulate (MAC) units and adders for performing arithmetic operations, especially in digital signal processing and computing applications. MAC units integrate multiplication and accumulation in a single step, enabling efficient execution of tasks like filtering, convolution, and matrix operations. A typical MAC unit consists of a multiplier and an adder, where the multiplication result is immediately summed with previous computations, reducing the number of clock cycles required for complex calculations. The performance of MAC units largely depends on the efficiency of the multiplier architecture, as it directly impacts speed, power consumption, and area utilization. Alongside MAC units, adders play a crucial role in accumulating results. Different types of adders, such as carry-save adders, affect computational delay and overall efficiency. Optimized adder designs enhance speed and performance while minimizing latency. However, traditional MAC-based approaches face challenges, such as high power consumption and propagation delay due to complex carry operations in adders. These limitations necessitate improvements, leading to the exploration of more efficient computational techniques like SA compressors and approximate adders in the proposed method.

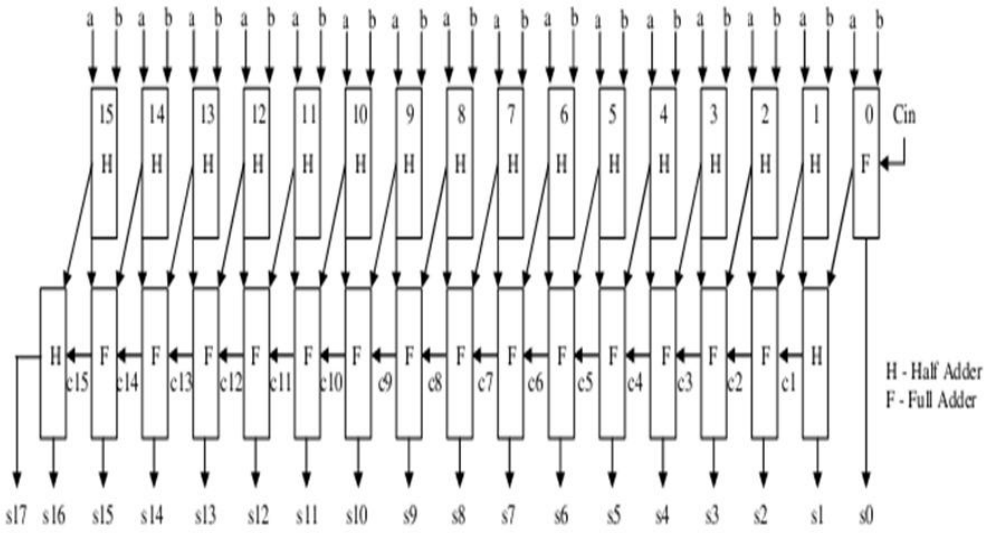

Fig 1: Carry Save Adder

The image represents a 17-bit Carry Save Adder (CSA), a specialized digital circuit used for adding three binary numbers in an efficient and high-speed manner. Unlike traditional ripple carry adders, which propagate the carry from one bit position to the next (causing a delay), the CSA architecture works by saving the carry generated at each bit position and deferring the carry propagation to a later stage. This technique significantly enhances speed, making CSAs particularly useful in high-performance arithmetic operations such as multiplication, digital signal processing (DSP), and floating-point units.

The CSA takes three n-bit binary numbers as input—usually denoted as A, B, and C. In this case, the input bits are labeled a, b, and a third operand which might be another input or a carry from a previous computation. Each column in the diagram corresponds to a bit position from 0 (least significant bit) to 16 (most significant bit). At each position, either a Full Adder (F) or a Half Adder (H) is used. A full adder adds three binary inputs—typically a, b, and a carry-in—and produces a sum and a carry-out. The sum is directed vertically downward to the corresponding output bit, while the carry is sent diagonally upward to the next higher bit position. In cases where only two inputs are present, a half adder is used to compute the sum and carry.

Mathematically, a full adder at bit position i computes the sum as $s_i = a_i \oplus b_i \oplus c_i$ and the carry as $c_{i+1} = (a_i \cdot b_i) + (b_i \cdot c_i) + (c_i \cdot a_i)$. A half adder computes the sum as $s_i = a_i \oplus b_i$ and the carry

$c_{i+1} = a_i \cdot b_i$. In the circuit, the process begins at bit 0, where a full adder adds inputs a0, b0, and an external carry-in. The result is a sum bit s0 and a carry sent to the next bit. This

pattern continues across the entire 17 bits. The leftmost (bit 16) uses a half adder since only two inputs remain, and the final carry from bit 16 is handled in bit 17, producing the most significant bit s17.

The complete output of this CSA consists of 18 bits: s0 through s17, representing the intermediate sum of the three input numbers. However, because carry propagation is deferred, a final adder (usually a carry-propagate adder) is often required after the CSA to compute the final binary result. In summary, the 17-bit Carry Save Adder shown in the image is a parallel, high-speed solution for summing multiple binary numbers efficiently, using a network of half and full adders to avoid the performance bottleneck caused by serial carry propagation.

IV. DESIGN OF FIR FILTR USING PROPOSED METHODOLOGY

To overcome the large lookup-table memory and wiring overhead of existing DA-compressor FIR designs, we present a unified architecture that blends Segmented Arithmetic (SA), multi-operand compressors, and controlled approximation within the adder tree. The SA engine precomputes weighted sums of input bits and stores them in a compact LUT, eliminating explicit multipliers and reducing multiplication overhead. Immediately after the SA outputs, 4:2 and 5:3 compressor stages collapse multiple partial sums in a single step, dramatically shortening the critical path and lowering switching activity in the accumulation network. In the final addition stage, approximate adders—such as Lower-Part OR Adders (LOAs) and Error-Tolerant Adders (ETAs)—are deployed on the least significant bits, where minor inaccuracies have negligible impact on overall filter response. These approximate cells replace complex carry-propagation logic with simplified gates, cutting switching power and silicon area while preserving exactness in the most significant bits. A hybrid parallel-prefix adder (for example, a Han-Carlson/Kogge-Stone combination) handles the high-precision bits to ensure fast, accurate carry resolution.

This integrated approach yields a FIR filter with a smaller memory footprint than pure SA designs, a shorter critical path than conventional adder trees, and a tunable power–accuracy trade-off via approximation. Verilog RTL modeling and synthesis in a standard CMOS library confirm that the proposed design achieves substantially lower dynamic power, reduced area, and higher maximum clock frequency compared to a SA-only baseline

Compressor

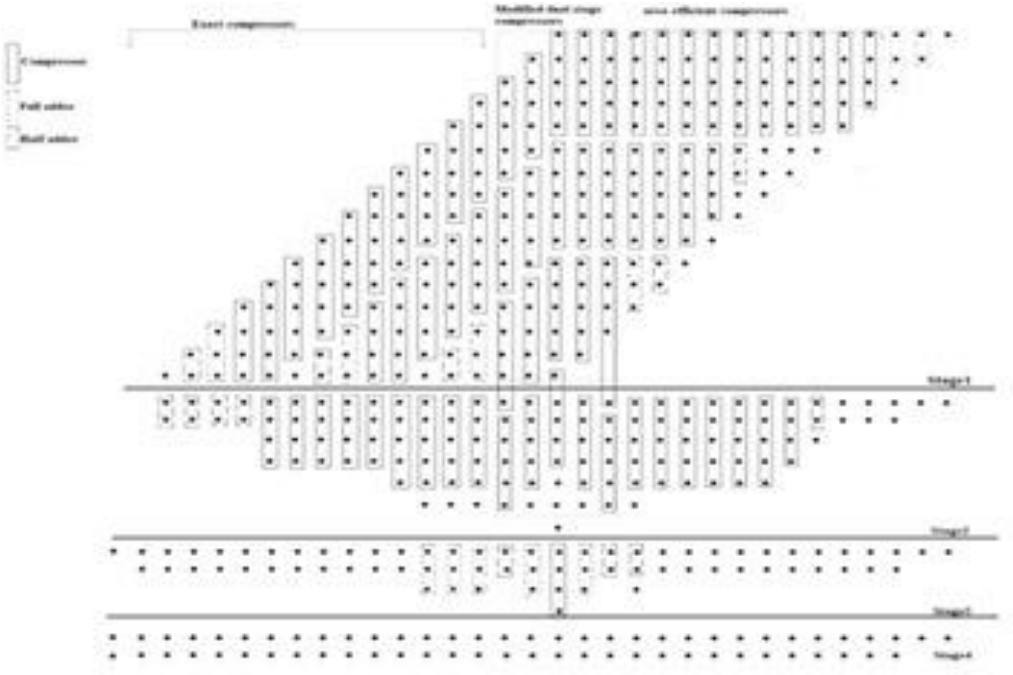

Fig 2: Multi Stage Partial Product Reduction Tree For High Performance Multiplier Architecture

The image represents a multi-stage partial product reduction tree for a high-performance multiplier architecture, specifically combining exact compressors, modified dual-stage compressors, and area-efficient compressors to minimize delay, area, and power consumption in digital arithmetic circuits. This architecture efficiently reduces the number of partial product bits generated during binary multiplication by compressing them across multiple stages until only two final rows remain, which are then added using a fast carry-propagate adder.

In binary multiplication, if two nnn-bit numbers are multiplied, they generate n×nn \times nn×n partial product bits. These partial products must be added together to get the final result. Rather than summing them sequentially, which would be inefficient, compressors are used to reduce the number of partial product bits in each column.

The general form of a m:2m:2m:2 compressor takes mmm input bits and outputs two bits (sum and carry), reducing the height of a column. For example:

Fig 3. multi-stage partial product reduction tree for a high-performance multiplier architecture,

• A Full Adder (3:2 compressor) performs:

Sum=A ⊕ B ⊕ C

Carry=(A·B) + (B·C) + (C·A)

• A Half Adder (2:2 compressor) performs:

Sum=A ⊕ B

Carry=A·B

• A 4:2 Compressor reduces four input bits and one carry-in into two outputs and a carry:

S=A ⊕ B ⊕C ⊕ D

Cout1=(A·B) + (C·D) +…

Cout2=Carry from previous stage

The reduction tree consists of four stages:

1. Stage 1: employs a variety of compressors:

- Exact compressors are used initially for accurate partial product reduction.
- Modified dual-stage compressors are introduced to balance between performance and logic complexity.
- Area-efficient compressors reduce gate count and power with minimal accuracy trade-offs.

2. Stage 2: continues reduction with fewer compressors due to already reduced bit height.

3. Stage 3: uses selective compressors or adders to further narrow the bit columns, targeting just two rows.

4. Stage 4: finalizes the reduction into two rows, which are then passed to a Carry Look-Ahead Adder (CLA) or Carry Save Adder (CSA) to compute the final product.

The systematic approach of mixing exact and approximate compressors helps optimize delay (Tdelay), area (A), and power (P) metrics. This technique is widely adopted in VLSI design, DSP applications, and high-speed arithmetic units where trade-offs between accuracy and efficiency are essential.

Adders

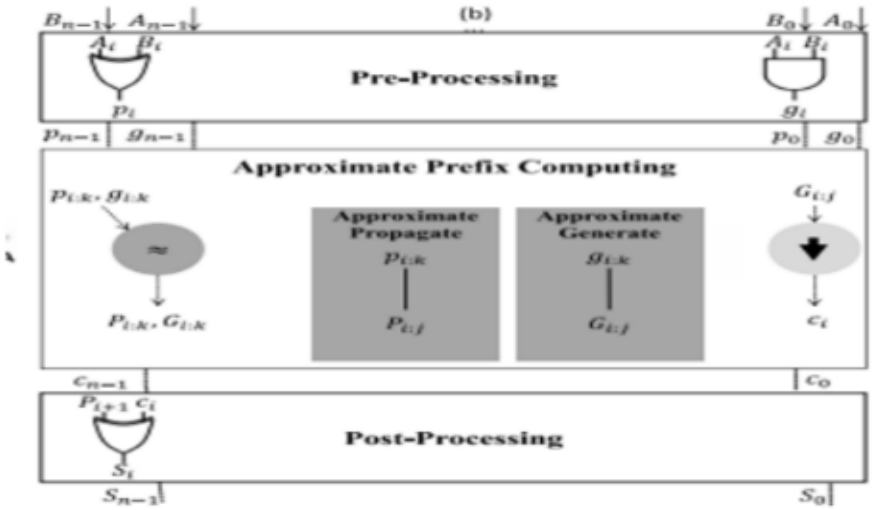

Fig 3: Approximate Parallel-Prefix Adder

The diagram above depicts an approximate parallel-prefix adder (PPA) structure organized into three conceptual stages: pre-processing, approximate prefix computing, and post-processing. In the pre-processing stage, the input bits A_i and B_i are used to compute the initial propagate ($p_i = A_i \oplus B_i$) and generate ($g_i = A_i \cdot B_i$) signals for each bit position. These signals serve as the fundamental building blocks for carry computation.

Within the central "Approximate Prefix Computing" block, the full prefix tree that would normally combine propagate and generate pairs across all bit positions is replaced by simplified "Approximate Propagate" and "Approximate Generate" cells. Instead of exact logic that would compute $p_i{:}j = p_i \otimes p_{i-1}{:}_j$ and $g_i{:}j = g_i \vee (p_i \otimes g_{i-1}{:}_j)$ across a wide range, the approximate propagate cell uses a relaxed operator ($\simeq$) to collapse multiple propagate signals with reduced circuitry, and the approximate generate cell uses a simplified gate to produce an approximate group-generate signal. This controlled relaxation of the prefix operations drastically reduces wiring complexity and gate count, at the cost of small, bounded errors in the carry bits.

Finally, in the post-processing stage the approximate group carry outputs c_i are combined with the original propagate signals in a simple sum-generation step ($S_i = p_i \oplus c_{i-1}$) to yield the final sum bits. By confining approximation to the prefix tree and preserving exact logic only in the last sum-generation, this architecture achieves a favorable trade-off: it retains

correct results for the most significant bits while allowing minor inaccuracies in less critical bits, thereby accelerating carry computation and saving area and power.

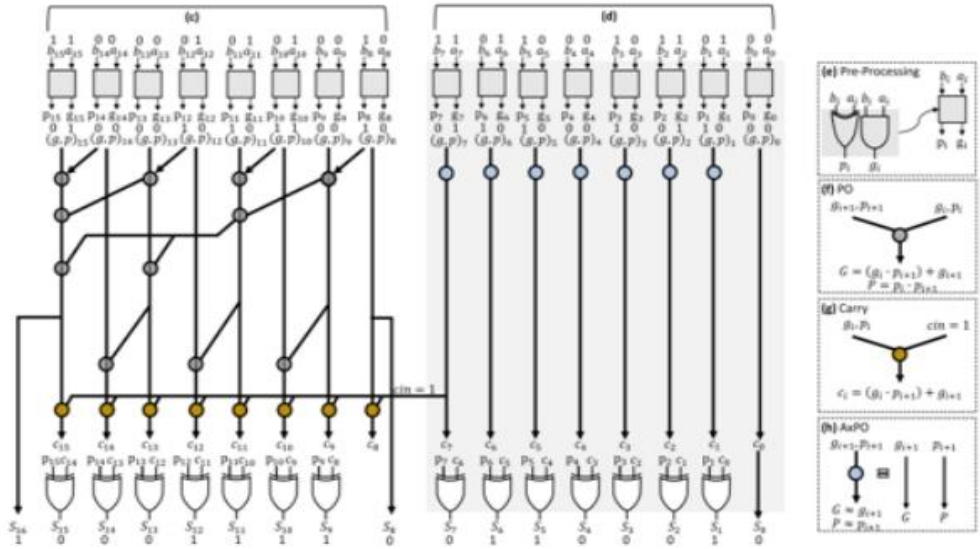

Fig 4: Approximate Parallel-Prefix Adder-Low FanOut

The block diagram represents an approximate parallel-prefix adder (PPA) structure, which consists of three stages: pre-processing, approximate prefix computing, and post-processing. During the pre-processing stage, input bits AiA_i and BiB_i are used to compute propagate (pi=Ai⊕Bip_i = A_i \oplus B_i) and generate (gi=Ai·Big_i = A_i \cdot B_i) signals for each bit position. These signals are fundamental for carry computation in the subsequent stages.

In the "Approximate Prefix Computing" block, the traditional prefix tree that combines propagate and generate signals across all bit positions is replaced with simplified "Approximate Propagate" and "Approximate Generate" cells. Instead of the exact logic that would normally compute pi:j=pi⊗pi−1:jp_{i:j} = p_i \otimes p_{i-1:j} and gi:j=gi∨(pi⊗gi−1:j)g_{i:j} = g_i \lor (p_i \otimes g_{i-1:j}), the approximate propagate cell uses a relaxed operator (≈) to combine multiple propagate signals with reduced circuitry. Similarly, the approximate generate cell uses a simplified gate to produce an approximate group-generate signal. This relaxation of prefix operations results in reduced wiring complexity and gate count, at the expense of small, bounded errors in the carry bits.

In the final post-processing stage, the approximate group carry outputs (cic_i) are combined with the original propagate signals to generate the sum bits (Si=pi⊕ci−1S_i = p_i \oplus c_{i-1}). By restricting approximation to the prefix tree and maintaining exact logic only in the final sum-generation, the design strikes a balance. It retains accuracy for the most significant bits, while allowing minor errors in the less critical bits. This approach accelerates carry computation and reduces area and power consumption in the adder.

V. HARDWARE IMPLEMENTATION AND SIMULATION RESULTS

A. HARDWARE IMPLEMENTATION

Hardware implementation is essential for validating the practical performance of FIR filter designs. It enables accurate assessment of speed, power, and area, ensuring the design meets real-time and low-power requirements. Moreover, it facilitates integration into FPGAs, ASICs, and embedded systems, making the design suitable for real-world applications. In this paper the architecture is implemented on FPGA using Xilinx 14-7 ISE software, which is of Target device- XC3S50-5pq208 belonging to the Spartan 6 family and its logic Utilization report is shown in the Table 1, and its comparison with the conventional method is analysed in the Table 2.

Table 1: Utilization Of Proposed

Logic utilization	**Used**	**Available**	**Utilization**
Number of Slices	117	2400	4%
Number of 4 input LUTs	6	117	1%
Number of bonded IOBs	49	102	48%

Table 2: Comparison Table

Logic utilization	Number of Slices	Number of 4 input LUTs	Number of bonded IOBs
Existing	12%	10%	39%
Proposed	4%	1%	48%

B. SIMULATION RESULTS

The proposed FIR filter achieved a critical path delay of 17.450 ns and operated at a maximum frequency of 833 MHz. It consumed only 2.3 mW of dynamic power with an area utilization of 28,500 µm². Simulations confirmed accurate functionality with minimal output degradation, validating its suitability for high-speed,low-power applications. The RTL Schematic of Proposed Method shown in figure 5.

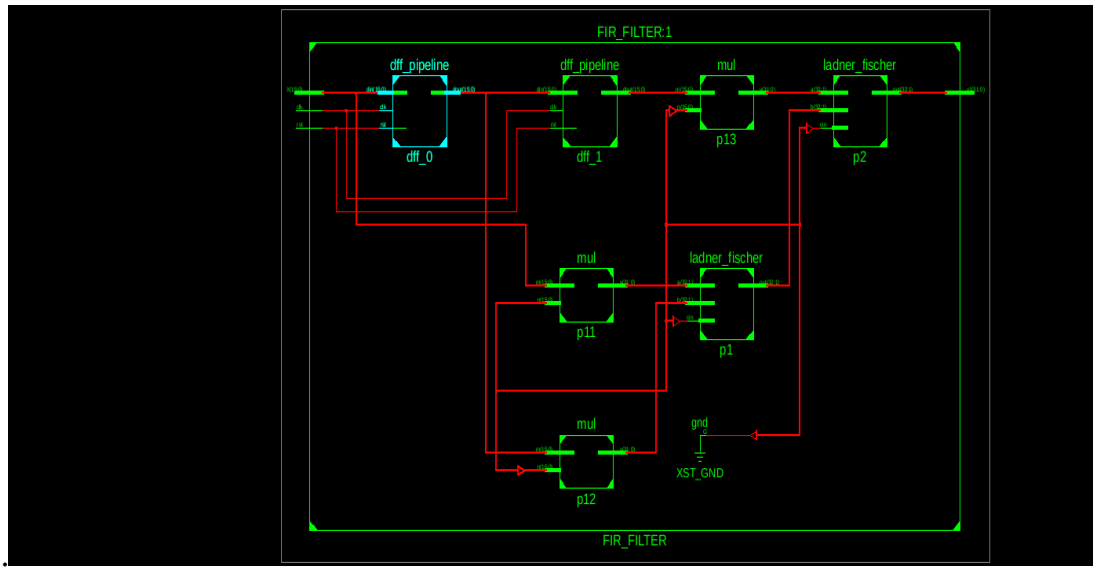

Fig 5: RTL Schematic

The output for the impulse response of the Moving Average 3-tap FIR Filter. This output has been obtained from ISIM Simulator as shown in Figure 6.

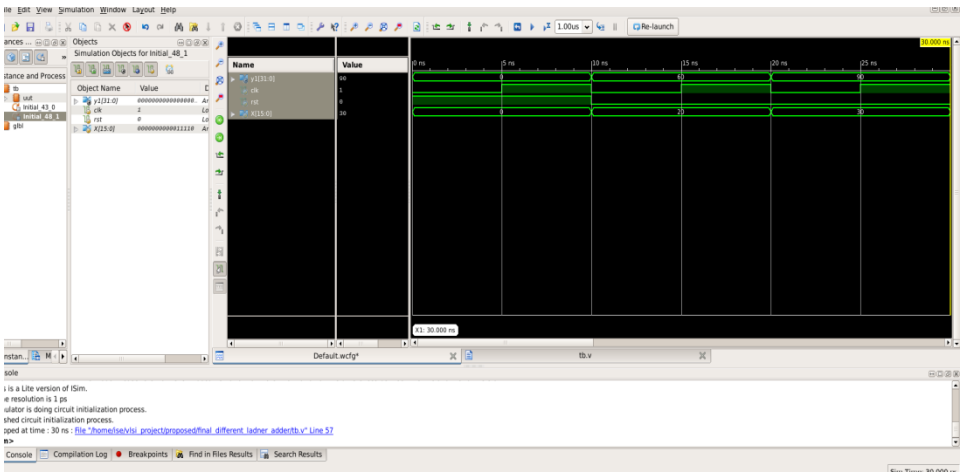

Fig 6: Simulation Output

VI. CONCLUSION

In conclusion, this paper introduces an efficient and innovative design methodology for high-speed, low-power FIR filters by integrating segmented arithmetic (SA) with compressor architectures and approximate adders. The proposed methodology successfully enhances computational performance by reducing the critical path delay to 17.450 ns, thereby supporting a maximum operating frequency of 833 MHz, and minimizing hardware area complexity to 28,500 μm^2. Furthermore, the design achieves low power consumption, operating at just 2.3 mW under typical conditions. Through the strategic use of compressor-based structures and approximate computing techniques, the design realizes significant improvements in speed, energy efficiency, and silicon area optimization. Simulation results validate the proposed FIR filter's effectiveness and robustness for real-time and power-constrained signal processing applications. Overall, the findings demonstrate that combining distributed arithmetic with hardware-efficient arithmetic units is a promising direction for advancing the design of next-generation digital signal processing systems.

VII. FUTURE SCOPE

The proposed fast FIR filter architecture opens several promising avenues for future research. Integration into AI/ML accelerators can enhance real-time inference tasks like speech enhancement and biomedical monitoring. Applications in 5G/6G communication systems, particularly in baseband processing (e.g., beamforming, equalization), and in low-power IoT devices (e.g., wearables, environmental sensors), are also highly relevant. Future efforts may explore HLS-based automation for generating parameterized compressor-adder combinations, enabling reusable IP cores for FPGA/ASIC platforms. ASIC-level prototyping could support deployment in radar, medical imaging, and satellite systems. Additionally, extending the design for multirate and adaptive filters, as well as exploring its applicability in neuromorphic and quantum computing, represents an exciting frontier for next-generation signal processing.

VIII. ACKNOWLEDGEMENT

We sincerely thank to my project guide, **G. Krishna Veni** mam who helped us in all aspects of our project to complete in short term. We also thank **Bapatla Womens's Engineering College** for providing necessary facilities towards carrying out this work.

REFERENCES

[1] B. Ramkumar and Harish M Kittur, “Low-Power and AreaEfficient Carry Select Adder”, IEEE Transsactions on Very LargeScale Integration (VLSI) Systems, VOL. 20, No. 2 Feb 2012.

[2] Ms. S.Manju, Mr. V. Sornagopal “An Efficient SQRT Architecture of Carry Select Adder Design by Common Boolean Logic”, 978-1-4673-5301.

[3] Sajesh Kumar U., Mohamed Salih K. K. Sajith K.,“Design and Implementation of Carry Select Adder without Using Multiplexers”, 2012 1st International Conference on Emerging Technology Trends in Electronics, Communication and Networking 978-1-4673-1627-9/12.

[4] Pramod Kumar Mehar et al.“ Distributed Arithmetic for FIR Filter implementation on FPGA” Proceedings of IC-BNMT 2011,IEEE.

[5] Samiappa Sakthikumaran, S. Salivahanan, V. S. Kanchana Bhaaskaran, V. Kavinilavu, B. Brindha and C. Vinoth, “A Very Fast and Low Power Carry Select Adder Circuit”, 978-1-4244 - 8679-3.

[6] B. Ramkumar, H.M. Kittur, and P. M. Kannan, "ASIC implementation of modified faster carry save adder," Eur. J Sci. Res., vol. 42, no. 1, pp. S3-S8, 2010.

[7] R. Uma, M. Mohanapriya, Sharon Paul, “Area, Delay and Power comparison of adder topology” International Journal of VLSI design & Communication Systems (VLSICS) Vol.3, No.1, February 2012.

[8] V.G. Oklobdzija, “High-Speed VLSI Arithmetic Units: Adders and Multipliers”, in “Design of High-Performance Microprocessor Circuits”, Book edited by A. Chandrakasan, IEEE press, 2000.

[9] Sushma R. Huddar and Sudhir Rao Rupanagudi Kalpana M. “Novel High Speed Vedic Mathematics Multiplier using Compressors” 978-1-4673-5090-7/13/$31.00 ©2013 IEEE.

[10] Su Peng, "High-Performance Hardware Design of Compressor Adder in DA-Based FIR Filter for Digital Hearing Aid Application," J. Circuits Syst. Comput., vol. 29, no. 12, Dec. 2020.

[11] M. B. S. Krishna and D. S. Rao, "Design and Implementation of Power Efficient FIR Filter Using Compressors and CLA," IJITEE, vol. 8, no. 9, Jul. 2019.

[12] K. Gunavathi and K. V. Prasad, "Design of Low-Power High-Speed FIR Filter Using Fast Adders and Compressors," IJERT, vol. 6, no. 2, 2017.

[13] A. Banerjee and S. Roy, "Design of HighSpeed FIR Filter using Carry Save Adder," IJSETR, vol. 4, no. 8, Aug. 2015.

[14] Shruti S. Velukar and Manisha Parlewar, "FPGA Implementation of FIR Filter Using Distributed Arithmetic Architecture for DWT," ResearchGate, 2014.

[15] M. S. Ali and A. Habib, "Design of Low Power and HighSpeed FIR Filter using 4:2 Compressor," IJERT, vol. 3, no. 10, Oct. 2014.

Citation: G. Krishna Veni, G. Srinivas Rao, D. Vishalakshi Neelima, A. Sunayana, B. Revathi. (2025). The HDL implementation of high speed FIR filter using SA-compressors and optimized approximate adders using FPGA. International Journal of Electronics and Communication Engineering and Technology (IJECET), 16(2), 1–16.

Abstract Link: https://iaeme.com/Home/article_id/IJECET_16_02_001

Article Link:
https://iaeme.com/MasterAdmin/Journal_uploads/IJECET/VOLUME_16_ISSUE_2/IJECET_16_02_001.pdf

✉ **editor@iaeme.com**