IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Enhanced Pipeline Fir Filter Design Via Accurate Critical Path Optimization

[1]Dr. D. R. Srinivas, [2] N. Lakshmi Anusha, [3]G. Mahitha Reddy, [4]C. Subba Sai Sivani

[1]Associate Professor, [2]Student, [3]Student, [3]Student

[1]Electronics and Communication Engineering,

[1]G Pulla Reddy Engineering College, Kurnool, India

Abstract: This new design method for Finite Impulse Response (FIR) filters focuses on optimizing the "critical path," which is the part of the filter where the signal takes the longest time to travel. By reducing this delay, the filter can process signals faster and more efficiently. This makes the filter work quicker, which is important in systems where speed matters. The optimization also helps reduce the amount of power the filter uses, making it perfect for applications that need to save energy, such as in mobile devices or other low-power systems. The benefits of this method are clear when tested in simulations, showing significant improvements in filter performance for both hardware and software implementations. By cutting down the delay in the critical path, the filter works faster and consumes less power. This approach is especially useful in devices with limited resources, where both speed and energy efficiency are key. Overall, this method offers a smart solution for designing FIR filters that meet the needs of modern systems, making them faster and more power-efficient.

Index Terms **–** Pipelining, Latency, Power-efficiency, FIR Filter, Critical Path, Delay, Low-power systems, Simulation, Filter Performance

I. INTRODUCTION

The design of Finite Impulse Response (FIR) filters plays a critical role in various signal processing applications, where efficiency and speed are crucial. FIR filters are widely used in systems like audio processing, communications, and even in mobile devices. However, these filters can sometimes struggle with delays and high-power consumption, especially in resource-limited environments. To address these challenges, the idea of enhancing FIR filter design through critical path optimization has emerged as a promising solution. By focusing on the critical path, which is the part of the filter where the signal experiences the longest delay, it's possible to reduce both latency and power consumption.

Critical path optimization works by carefully adjusting the filter design to minimize the delay of the longest signal travel route, allowing the system to process information faster. This is essential for improving the overall performance of FIR filters in high-speed applications. Additionally, reducing the critical path directly lowers the energy needed to perform computations, which is especially valuable in portable and low-power devices. With this approach, FIR filters can be made faster and more power-efficient, meeting the growing demands of modern technology for both speed and energy savings.

This enhanced pipeline FIR filter design approach has been tested through simulations, demonstrating significant improvements in filter performance. The optimized design shows that it's possible to create FIR filters that are not only faster but also consume less power, making them suitable for both hardware and software implementations. The results offer a promising solution for designing FIR filters in resource-constrained systems, where both processing speed and energy efficiency are essential. This methodology provides a way to improve FIR filter design without requiring extra hardware, making it a valuable tool in many advanced applications, from mobile technology to embedded systems.

II. LITERATURE REVIEW

1 . An efficient N-bit 8-2 adder compressor with a constant internal carry propagation delay: In 2020, G. Paim, T. V. Fontanari, L. M. G. Rocha, E. A. C. da Costa, and S. Bampi introduced a design for an efficient N-bit 8-2 adder compressor. This design is aimed at improving the performance of digital circuits by optimizing the process of adding binary numbers. Specifically, the 8-2 adder compressor helps in reducing the delay caused by carry propagation during the addition process. The key feature of their design is that it maintains a constant delay for internal carry propagation, making it faster and more reliable, especially for complex digital systems. They presented their work at the International Conference on Electronics Circuits and Systems (ICECS) in 2020, offering a new approach to enhance the efficiency of adders used in various electronics.

2. Area–delay and energy efficient multi-operand binary tree adder: In 2020, S. K. Patel and S. K. Singhal developed a design for an "Area–delay and energy efficient multi-operand binary tree adder." This design focuses on improving the performance of adders used in digital systems, particularly when adding multiple binary numbers. The main goal of their work was to reduce the area (the physical space the adder occupies in a circuit), the delay (the time it takes to complete the addition), and the energy consumption. By using a binary tree structure, their adder efficiently handles multiple operands, making it faster, smaller, and more energy-efficient, which is especially useful for high-performance computing and mobile devices.

3. A high-performance and energy-efficient FIR adaptive filter using approximate distributed arithmetic circuits: In 2019, H. Jiang, L. Liu, P. P. Jonker, D. G. Elliott, F. Lombardi, and J. Han developed a design for "A high-performance and energy-efficient FIR adaptive filter using approximate distributed arithmetic circuits." Their work focuses on improving the performance and energy efficiency of Finite Impulse Response (FIR) adaptive filters, which are widely used in digital signal processing applications like noise reduction, echo cancellation, and communication systems. The key innovation in their design is the use of approximate distributed arithmetic circuits, which reduce the complexity and energy consumption of the filter without significantly affecting its performance. This makes the filter faster and more energy efficient, which is important for applications in resource-constrained environments like mobile devices and embedded systems.

4. Efficient hardware architectures for deep convolutional neural networks (CNNs): In 2018, J. Wang, J. Lin, and Z. Wang developed "Efficient hardware architectures for deep convolutional neural networks (CNNs)." Their work focuses on improving the hardware designs used to accelerate deep learning applications, particularly CNNs, which are widely used for tasks like image and speech recognition. CNNs are computationally intensive, requiring powerful hardware for efficient processing. The authors proposed new hardware architectures that optimize performance, reduce power consumption, and increase the efficiency of CNNs. These architectures aim to make deep learning more feasible and faster, especially in environments with limited resources, such as embedded systems or mobile devices, where efficiency is crucial.

5. Seamless Pipelining of DSP Circuits: P. K. Meher's paper, "Seamless Pipelining of DSP Circuits," published in Circuits, Systems, and Signal Processing in April 2016, discusses efficient techniques for designing high-performance digital signal processing (DSP) circuits. The paper focuses on seamless pipelining, a method that improves processing speed by dividing complex operations into smaller, manageable stages without adding extra delays. This approach helps in reducing power consumption and enhancing computational efficiency, making DSP systems faster and more energy-efficient. The study presents various strategies to optimize hardware implementation, making it useful for applications in communications, multimedia, and real-time signal processing.

6. Low-Power, High-Throughput, and Low-Area Adaptive FIR Filter Based on Distributed Arithmetic: The paper titled "Low-Power, High-Throughput, and Low-Area Adaptive FIR Filter Based on Distributed Arithmetic" by S. Y. Park and P. K. Meher, published in the IEEE Transactions on Circuits and Systems II in June 2013, presents a highly efficient design approach for adaptive finite impulse response (FIR) filters. By leveraging distributed arithmetic (DA), the proposed method eliminates the need for conventional multipliers, which are typically power-hungry and resource-intensive. This shift not only reduces power consumption but also enables higher processing speed and significant savings in hardware area, making the design particularly well-suited for resource-constrained environments. The optimized adaptive FIR filter architecture introduced in this study is tailored for real- time signal processing applications such as wireless communication and audio processing, where a fine balance between performance and efficiency is critical. Through. Innovative use of DA and efficient hardware implementation techniques, the design achieves low power, high throughput, and compact area without compromising adaptive filtering capabilities. The work stands out for its effective trade-off strategy, addressing key constraints in modern digital signal processing systems.

III. EXISTING METHOD

Traditional Pipelining and Critical Path Optimization in VLSI

In VLSI (Very Large-Scale Integration) design, pipelining is a widely used technique to improve the speed and efficiency of circuits. It works by breaking down a large operation into smaller stages, allowing multiple operations to be processed simultaneously. This increases the overall speed, as different stages of a task can be executed at the same time instead of waiting for one task to finish before starting another.

However, traditional pipelining systems face several challenges. One major issue is imbalanced pipeline stages, where some stages take longer to complete than others, leading to inefficiencies. Another issue is pipeline overhead, which includes additional registers and synchronization components that increase the circuit's complexity and power consumption.

To optimize performance, critical path analysis is performed. The critical path is the longest delay between any two points in a circuit, and it determines the maximum speed at which the circuit can operate. If this path is too long, it can slow down the entire system. Techniques such as retiming, which redistributes registers, and clock gating, which reduces unnecessary power usage, are used to improve the system's efficiency.

Despite these optimizations, traditional pipelining still struggles with issues like high power consumption and complex hardware design. To address these problems, researchers have developed advanced techniques such as seamless pipelining, which eliminates unnecessary delays, and distributed arithmetic, which reduces the need for power-hungry multipliers in signal processing applications. These modern approaches help create faster, more efficient, and low-power VLSI circuits for applications in communication, multimedia, and artificial intelligence.

IV. PROJECT METHODOLOGY

4.1 BLOCK DIAGRAM OF PROPOSED SYSTEM

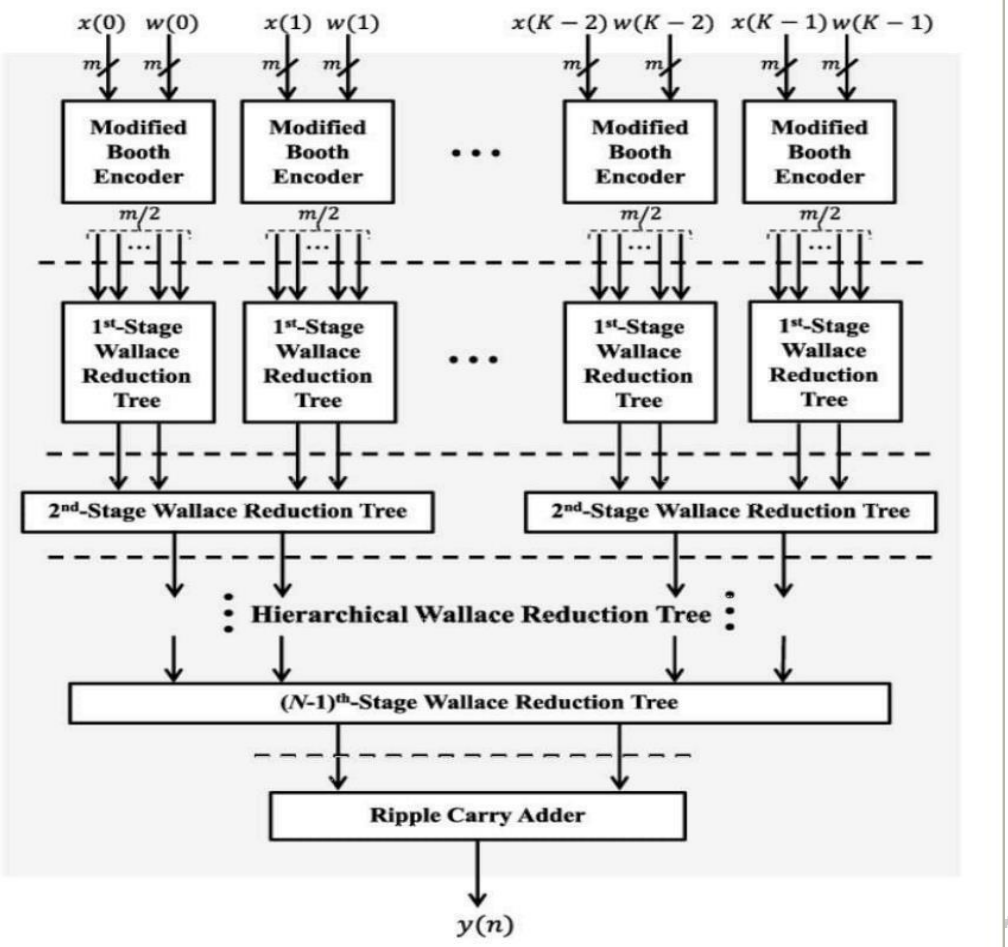

FIG 4.1 BLOCK DIAGRAM OF PROPOSED SYSTEM

BLOCK DIAGRAM DESCRIPTION

The given block diagram represents a high-speed multiplier architecture that utilizes Modified Booth Encoding (MBE), Wallace Tree Reduction, and Ripple Carry Addition to perform efficient multiplication.

Description of the Block Diagram:

1. **Input Stage:**
   - The inputs are represented as x(0),x(1),...x(K−1)x(0), x(1), ... x(K-1)x(0),x(1),...x(K−1) and w(0),w(1),...w(K−1)w(0), w(1), ... w(K-1)w(0),w(1),...w(K−1), which are the multiplicands and multipliers, respectively.
   - Each pair of x(i)x(i)x(i) and w(i)w(i)w(i) is fed into a Modified Booth Encoder (MBE).

2. **Modified Booth Encoder (MBE):**
   - The MBE reduces the number of partial products by encoding groups of bits (typically 3-bit groups) from the multiplier.
   - This step helps in reducing the overall complexity of multiplication**.**

3. **First-Stage Wallace Reduction Tree:**
    - The outputs from the MBE are fed into the 1st-stage Wallace Reduction Tree.
    - This stage reduces the number of partial product rows by efficiently summing them using carry-save adders (CSAs).

4. **Second-Stage Wallace Reduction Tree:**
    - Further reduction of partial product rows is performed in the 2nd-stage Wallace Reduction Tree. o This stage continues to compress the number of bits to make final addition easier.

5. **Hierarchical Wallace Reduction Tree:**
    - Multiple Wallace Reduction Trees are combined in a hierarchical structure to further optimize the summation of partial products.

6. **Final Stage - (N-1) -Stage Wallace Reduction Tree:**
    - At this stage, the final partial products are combined and prepared for final summation.

7. **Ripple Carry Adder (RCA):**
    - The last step in the architecture is the Ripple Carry Adder, which performs the final addition to produce the output y(n)y(n)y(n).
    - Since RCAs propagate carries sequentially, this stage may introduce some delay Compared to previous fast reduction stages.

Key Advantages of This Architecture:

- Speed Optimization: The use of Modified Booth Encoding (MBE) reduces the number of partial products, making multiplication faster.
- Efficient Reduction: The Wallace Tree minimizes delay by parallelizing summation operations using carry-save adders.
- Hierarchical Structure: This design efficiently handles larger multiplications by organizing reduction stages in a structured hierarchy.

4.2 FLOW CHART

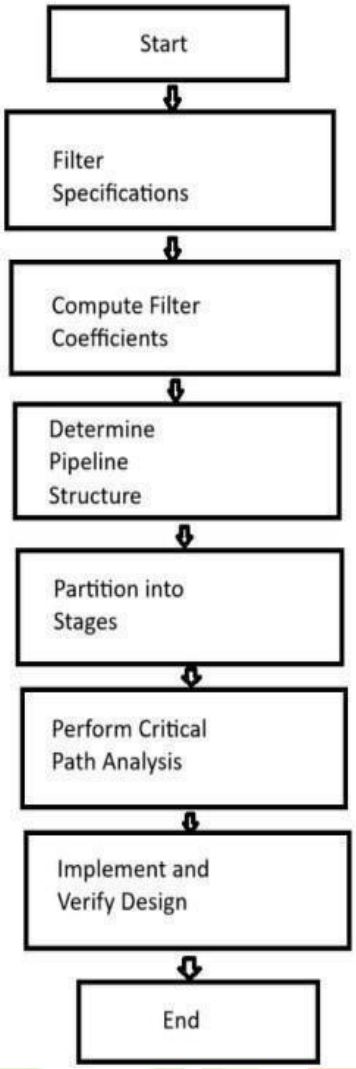

FIG 4.2 FLOW CHART OF THE PROPOSED SYSTEM

FLOW CHART DESCRIPTION

1. **Define Filter Specifications:** Start by setting the filter order, cutoff frequency, and sampling rate based on the target application. These parameters determine the filter's frequency response and influence its complexity and performance. For instance, a higher order provides better selectivity but increases hardware requirements.

2. **Compute Filter Coefficients:** Use established design methods such as the windowing technique or the Parks-McClellan algorithm to calculate the FIR filter coefficients. These coefficients serve as weights applied to the input samples to shape the desired output. The chosen method depends on the required precision and computational efficiency.

3. **Determine Pipeline Structure:** Select an appropriate FIR filter form—commonly direct or transposed—for pipeline implementation. The transposed form is often preferred for hardware due to its lower critical path delay and better parallelism. The structure should balance power, area, and performance.

4. **Partition into Stages:** Divide the filter operations into multiple stages to increase throughput and support parallel processing. Each stage handles a subset of the computations (e.g., multiply-accumulate

operations). Techniques like Distributed Arithmetic (DA) can further optimize by replacing multipliers with look-up tables and shift-add operations.

5. **Perform Critical Path Analysis:** Identify the longest delay path in the pipeline, which dictates the maximum clock speed. Optimize this path by reordering computations, adding pipeline registers, or balancing logic across stages. Reducing the critical path enhances overall speed and efficiency.
6. **Implement and Verify Design:** Deploy the filter on hardware (e.g., FPGA) or software (e.g., DSP processor). Use simulation tools to verify functionality and ensure that performance, latency, and power constraints are met. Post-synthesis analysis and testing are essential for validating real-time operation.

4.3 ALGORITHM USED AND EQUATIONS

Finite Impulse Response (FIR) filters are widely used in digital signal processing (DSP). To enhance their performance, pipelining is used to break long critical paths and optimize the overall speed. Below is an approach to designing an Enhanced Pipeline FIR Filter using Accurate Critical Path Optimization (ACPO).

1. **FIR FILTER EQUATION:** A discrete-time FIR filter of order N is mathematically expressed as:

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

Where:

- y(n) is filter output,
- x(n) is the input sequence,
- h(i) are the filter coefficients,
- N is the filter order.

2. **CRITICAL PATH IN FIR FILTERS**: The critical path in a direct-form FIR filter consists of a multiplication followed by addition (MAC operation). This path affects the filter's speed, and reducing its delay improves performance.

The critical path delay Tcp is given by:

$$\mathbf{Tcp = Tmult + (N-1)Tadd}$$

Where:

- Tmult = time taken for multiplication operation,
- Tadd = time taken for addition operation.

To optimize Tcp, pipeline registers are inserted to reduce dependency between operations.

3. Pipeline Enhancement Strategy

Pipeline optimization aims to break the longest combinational path by inserting register stages after each operation. The key steps include:

- **Decomposing the Filter Structure:**
  - A direct-form FIR filter is divided into multiple pipeline stages.
  - Each stage contains multiplications, additions, and delay elements.
- **Insertion of Pipeline Registers:**
  - Registers are inserted at optimal locations to break the longest computation path.

- The number of registers depends on the filter order and hardware constraints.

➢ **Retiming Optimization:**

- The optimal placement of registers is determined by analyzing register transfer level (RTL) equations.
- Retiming shifts registers across logic gates to minimize delays and balance stages

4. Equations for Critical Path Optimization:

➢ **PIPELINED MAC OPERATION:**

For a single-stage pipeline, the optimized delay is:

$$T_{pipelined} = \max(T_{mult}, T_{add})$$

Which ensures parallel processing of multiplication and addition.

➢ **Multi-Stage Pipeline Optimization***:*

For deeper pipelining, the optimized delay becomes:

$$T_{opt} = T_{CP}/S$$

Where S is the number of pipeline stages. Increasing S reduces Topt, improving speed.

➢ **OVERALL THROUGHPUT IMPROVEMENT:**

The throughput of the pipelined FIR filter is:

$$\text{Throughput} = 1 / T_{opt}$$

A higher number of pipeline stages increases throughput, making the filter faster.

5. Algorithm for Enhanced Pipeline FIR Filter Design:

➢ **INPUT FIR FILTER SPECIFICATIONS:**

- Define order N, coefficients h(i), and sampling frequency.

➢ **DETERMINE INITIAL CRITICAL PATH DELAY Tcp.**

➢ **APPLY PIPELINE REGISTER INSERTION:**

- Insert registers after multiplication and addition operations.
- Use retiming to balance logic levels.

➢ **COMPUTE OPTIMIZED CRITICAL PATH DELAY Topt.**

➢ **VALIDATE THE DESIGN:**

- Simulate the FIR filter in MATLAB/HDL/VERILOG.
- Compare performance with non-pipelined filters.

➢ **IMPLEMENT HARDWARE REALISATION:**

- Delay in FPGA/ASIC with optimized pipeline stages.

V. RESULTS AND DISCUSSION

The enhanced pipelined FIR filter design incorporating Wallace Tree architecture demonstrates significant improvements in performance and efficiency. By accurately analysing and minimizing the critical path, the design achieves faster data throughput and better timing performance. The integration of pipelining with Wallace Tree-based reduction optimizes the summation logic, enabling parallel processing with reduced delay. The RTL structure confirms systematic data flow through multiple stages, ensuring stable and reliable operation. Simulation results validate the functional correctness of the design, showing proper propagation and accumulation of input data. Furthermore, the power analysis indicates efficient resource utilization with low dynamic switching, reflecting the suitability of the design for low-power, high-speed digital signal processing applications.

REFERENCES

1. M. Singh, J. A. Tierno, A. Rylyakov, S. Rylov, and S. M. Nowick, “An adaptively pipelined mixed synchronous-asynchronous digital FIR filterchip operating at 1.3 Gigahertz,” IEEE Trans. VLSI Syst., Jul. 2010.
2. S. S. Nayak and P. K. Meher, “High throughput VLSI implementation of discrete orthogonal transforms using bit-level vector-matrix multiplier,” IEEE Trans. Circuits Syst. II, May 1999.
3. P. K. Meher and S. Y. Park, “Critical-path analysis and low-complexity implementation of the LMS adaptive algorithm,” IEEE Trans. Circuits Syst. I, Oct. 2014.
4. P. K. Meher, “Seamless pipelining of DSP circuits,” Circuits, Systems, and Signal Processing, vol. 35, Apr. 2016.
5. http://www.jetir.org/papers/JETIR2401222.pdf
6. D. Kang, Y. Kang, and Y. Hong, ‘‘VLSI implementation of fractional motion estimation interpolation for high efficiency video coding,’’ Electron. Lett., vol. 51, Jul. 2015.
7. J. Wang, J. Lin, and Z. Wang, ‘‘Efficient hardware architectures for deepconvolutional neural network,’’ IEEE Trans. Circuits Syst. I, Reg. Papers,vol. 65, no. 6, pp,Jun. 2018.
8. L. Dadda and V. Piuri, ‘‘Pipelined adders,’’ IEEE Trans. Comput., vol. 45,no. 3, pp. 348–356, Mar. 1996.
9. J.-Y. Kang and J.-L. Gaudiot, ‘‘A simple high-speed multiplier design,’’IEEE Trans. Comput., vol. 55, no. 10, pp. Oct. 2006.