www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001

HIGH-SPEED FIR FILTER DESIGN USING A HYBRID VEDIC-BOOTH MULTIPLIER FOR EFFICIENT DIGITAL SIGNAL PROCESSING

Dr. C. Selvi[1], S Parthibhan[2]

[1]Associate Professor, Department of Electronics and Communication, Muthayammal Engineering college, Namakkal, Tamil Nadu India.

[2]PG Student Department of Electronics and Communication, Muthayammal Engineering college, Namakkal, Tamil Nadu India.

Selvishankar03@gmail.com[1], parthibhan89@gmail.com[2]

DOI: https://www.doi.org/10.58257/IJPREMS41247

ABSTRACT

Finite Impulse Response (FIR) filters are essential components in digital signal processing (DSP) applications, enabling precise frequency domain filtering for audio processing, communication systems, and image enhancement. However, the computational complexity of multiplication operations in FIR filters poses significant challenges in achieving high-speed and power-efficient performance. This work presents an optimized FIR filter architecture that integrates a Hybrid Vedic-Booth Multiplier (HVBM) to enhance computational efficiency, minimize power consumption, and reduce hardware complexity. The HVBM combines Booth Encoding with Vedic Multiplication, efficiently handling signed operations while reducing the number of partial products. Vedic Multiplication partitions computations into smaller 4×4 blocks, enabling parallel processing and minimizing propagation delay, whereas Booth Encoding optimizes sign handling to improve computational speed. Experimental results demonstrate that the proposed FIR filter achieves superior performance, lower latency, and reduced power consumption, making it an ideal solution for high-throughput DSP applications, embedded systems, and VLSI implementations. By leveraging the advantages of Vedic mathematics and Booth Encoding, this work introduces a scalable and power-efficient approach for next-generation real-time signal processing.

Keywords- FIR; DSP; HVBM; DTFT; CLA, ASIC.

1. INTRODUCTION

In signal processing Finite Impulse Response (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. The impulse response of an Nth-order discrete-time FIR filter lasts exactly N + 1 samples before it then settles to zero. FIR filters can be discrete-time or continuous-time,and digital or analog. For a causal discrete-time FIR filter of order N, each value of the output sequence is a weighted sum of the most recent input values 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.Are inherently stable, since the output is a sum of a finite number of finite multiples of the input values, so can be no greater than times the largest value appearing in the input. Can easily be designed to be linear phase by making the coefficient sequence symmetric. This property is sometimes desired for phase-sensitive applications, for example data communications, seismology, crossover filters, and mastering.

An FIR filter is designed by finding the coefficients and filter order that meet certain specifications, which can be in the time-domain (e.g. a matched filter) and/or the frequency domain (most common). Matched filters perform a cross-correlation between the input signal and a known pulse-shape. The FIR convolution is a cross-correlation between the input signal and a time-reversed copy of the impulse-response. Therefore, the matched-filter's impulse response is "designed" by sampling the known pulse-shape and using those samples in reverse order as the coefficients of the filter.

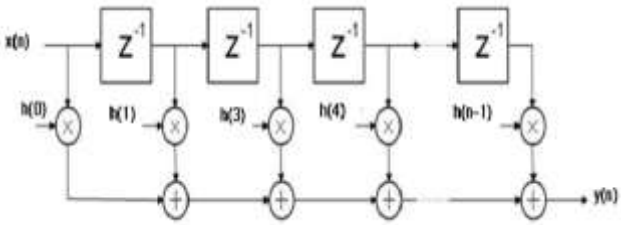


Fig 1. Basic structure of FIR filter

In the window design method, one first designs an ideal IIR filter and then truncates the infinite impulse response by multiplying it with a finite length window function. The result is a finite impulse response filter whose frequency response is modified from that of the IIR filter. Multiplying the infinite impulse by the window function in the time domain results in the frequency response of the IIR being convolved with the Fourier transform (or DTFT) of the window function. If the window's main lobe is narrow, the composite frequency response remains close to that of the ideal IIR filter.

The ideal response is usually rectangular, and the corresponding IIR is a sinc function. The result of the frequency domain convolution is that the edges of the rectangle are tapered, and ripples appear in the passband and stopband. Working backward, one can specify the slope (or width) of the tapered region (transition band) and the height of the ripples, and thereby derive the frequency domain parameters of an appropriate window function. Continuing backward to an impulse response can be done by iterating a filter design program to find the minimum filter order. Another method is to restrict the solution set to the parametric family of Kaiser windows, which provides closed form relationships between the time-domain and frequency domain parameters. In general, that method will not achieve the minimum possible filter order, but it is particularly convenient for automated applications that require dynamic, on-the-fly, filter design

Low-power VLSI design focuses on reducing power consumption in integrated circuits (ICs), which is essential in modern electronics for improving battery life in mobile devices, reducing heat in high-performance systems, and achieving overall energy efficiency. With the increase in IC complexity and the demand for compact, portable, and high-performance electronics, managing power consumption has become a priority in VLSI design. Power reduction techniques in VLSI are classified into multiple categories, including logic-level and circuit-level modifications, each addressing specific aspects of the design to minimize power usage while maintaining performance.

POWER CONSUMPTION IN VLSI AND ITS TYPES

Power consumption in VLSI circuits can be broadly divided into dynamic and static power. Dynamic power is consumed during switching activities, i.e., when transistors transition between logic states, and is a function of factors like clock frequency, switching activity, and capacitance. Static power, or leakage power, occurs even when the circuit is idle due to sub-threshold leakage currents in transistors. As technology scales down, leakage power becomes a significant component of overall power consumption.

LOGIC-LEVEL MODIFICATIONS

Circuit manufacturers must thoroughly test their products before delivering them to customers. The causes of circuit failure can be divided into two main categories: design errors and manufacturing defects. Logic-level modifications focus on optimizing the logic design and data flow within the circuit to reduce unnecessary switching activities, thereby saving dynamic power. These techniques aim to minimize the amount of activity in the logic gates by optimizing the circuit's logic structure without affecting functionality.

- **Gate Sizing**: Adjusting the size of transistors in logic gates to the minimum size required for the performance. Smaller gates consume less power, though they may slow down the circuit in some cases.
- **Clock Gating**: One of the most effective ways to reduce power, clock gating disables the clock signal to certain parts of the circuit when those parts are not in use, effectively reducing switching power. For example, registers and counters may only be active when new data is available, so the clock signal can be gated off to those modules during inactive periods.
- **Operand Isolation**: In certain designs, operand isolation uses control logic to cut off data inputs to certain logic blocks when they are not required, preventing unnecessary switching and reducing power usage.

CIRCUIT-LEVEL MODIFICATIONS

Circuit-level modifications focus on optimizing the transistor circuits and internal elements of the ICs to further minimize power consumption. This level involves changes to the underlying transistor circuits used to implement logic gates and other components.

- **Voltage Scaling**: Reducing the supply voltage (Vdd) is one of the most effective ways to reduce power consumption, as dynamic power is proportional to Vdd2V_{dd}^2Vdd2. However, lower voltages can also reduce speed, so designers must balance power savings with performance needs. For instance, dynamic voltage and frequency scaling (DVFS) adjusts voltage and frequency based on workload requirements, lowering power usage during low-demand periods and increasing performance when needed.
- **Sub-Threshold Logic**: In sub-threshold logic, the supply voltage is reduced below the threshold voltage of transistors, drastically cutting power consumption at the expense of performance. While this approach may lead to

increased delays, it is valuable for ultra-low-power applications such as IoT devices and sensors where speed requirements are minimal.

- **Multi-Threshold CMOS (MTCMOS)**: This technique uses transistors with different threshold voltages within the same design. High-threshold transistors reduce leakage currents for non-critical paths, while low-threshold transistors provide higher speeds for critical paths. MTCMOS is particularly useful in reducing leakage power in idle parts of a circuit.
- **Power Gating**: Power gating disconnects power from parts of the circuit when they are inactive, thereby reducing leakage power. By turning off entire sections of a circuit when they are not required, significant power savings are achieved, though extra circuitry for control is needed. Power gating is ideal for components that remain inactive for extended periods.
- **Body Biasing**: In body biasing, the body (substrate) of a transistor is biased to increase or decrease its threshold voltage. By dynamically adjusting the threshold voltage, body biasing helps to reduce leakage currents in idle states and improve performance during active states. Forward body biasing can reduce the threshold for active regions, while reverse body biasing can increase it for inactive regions.

APPLICATIONS OF MULTIPLIERS IN VLSI

Multipliers are integral to a wide range of applications, particularly in DSP and machine learning accelerators where repeated multiplication is a core operation. In image processing, multipliers enable fast filtering, transformation, and scaling operations. In cryptography, they assist in encryption algorithms that rely on complex arithmetic. Additionally, multipliers are foundational in neural network hardware accelerators, where efficient matrix multiplications are necessary for high-throughput computations.

This in-depth exploration of multipliers in VLSI highlights their diversity, with each type catering to specific requirements in terms of speed, power, area, and accuracy. The optimal choice depends on the application's demands and the design goals for performance and efficiency.

2. APPROXIMATION METHODOLOGIES

Based on the literature survey, it is known that several methods have been adapted in designing Approximate Circuits.Another common approach to circuit-level approximations is the design of arithmetic data paths using approximate adders and/or multipliers. Approximations of the degree of complexity may be rendered in sophisticated architectures, such as Artificial Neural Networks (ANNs), by recognizing essential neurons. Software may also be Approximated using methods such as Computer. The following is a description of the Approximate programming methods through hardware and software stacks.

OUTLINE OF THE APPROXIMATE COMPUTING METHODS

Selective Approximation -Using manually annotated or automatically interpreted code to adaptively bypass error-resilient portions of the application as per user-defined consistency specifications.

Timing Relaxation - Synchronization between concurrent systems or hand-shaking between separate code segments should be relaxed in order to boost energy efficiency and performance.

Domain Specific Approximation - Using domain / application-specific information by classifying data into categories so that sensitive / complex data is processed using correct computation and insensitive data is processed by using approximate modules.

Functional Approximation - Energy-hungry Approximate code fragments are supplemented by approximate alternatives that are either performed using correct or different approximate hardware modules.

Architectural

Selective Approximation – Utilizes specialized approximate hardware components to perform specific instructions / code segments.

Domain Specific Approximation - Integrates Domain / Application relevant expertise to develop proficient approximate architectures.

Functional Approximation - Simplifies the design sophistication of the hardware by utilizing the data type of essential paths or the general implementation of dynamic modules by utilizing their approximate counterpart.

Data Approximation - Relaxed reliability / reliability memory architectures (specifically by easing error correction limits or intelligently handling read / write operations at hardware error-resilient data level) contribute to major developments in resource efficiency and performance.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001

APPROXIMATE ADDERS WITH LOW POWER

Ripple Carry Adder (RCA) is the most common and basic adder design consisting of cascaded full-adder modules. The most influential aspect of RCA is its low power usage relative to all other accurate adder architectures. Applications with tight resource restrictions need much easier and lesser resource consuming modules. Using the usable low-power full-adder modules, low-power RCAs may be built by replacing accurate full-adders with their approximate counterpart. Full-adders close to LSB positions have fewer importance relative to the bits close to MSB, thus, in most instances, LSBs are approximated using various types of Approximate-full-adders.

LOW-LATENCY APPROXIMATE ADDERS

In order to fulfill the high output performance of error-tolerance systems, the truncation of its critical pathways may be abused. In the case of adders, it is also usually accomplished by utilizing several intersecting smaller sub-adder units that run in parallel to produce output with comparatively lower latency

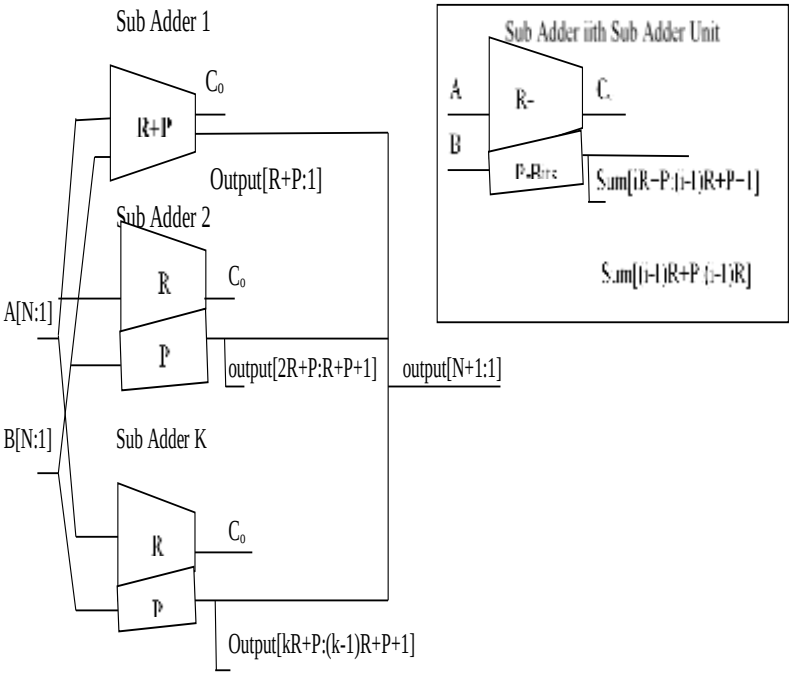


Figure 2. Low – latency approximate adder

Low – latency approximate adder

Occasionally, such adders are often equipped with error identification and correction logic to minimize the error likelihood by dynamically modifying the carry getting disseminated to subsequent levels.

Thorough and time-efficient space exploration of usable low-latency adders require a coherent model that is highly parameterizable and capable of accommodating most (but not all) forms of low-latency adders. To this end, we have designed the GeAr (Shafique et al. 2015) adder structure, that is highly customizable and able to model so many of the state-of – the-art high-efficient adders.

For adding two N-bit operands, the GeAr adder uses k number of L-bit sub-adders that work in parallel to produce an Approximate output with high rate of speed, where L < = N is used. Figure 1. Every sub-adder generates R-bits of output except for the first sub-adder that generates L-bits of output where L = R+P. The number of necessary sub-adders k for the given N, R, and P can be determined by k = (N-P)/R. In fact, for a combined effect of N, R and P is also be true for the GeAr model, when the P-bits of each sub-adder spreads the carry and the carry is produced by the preceding sub-adder.

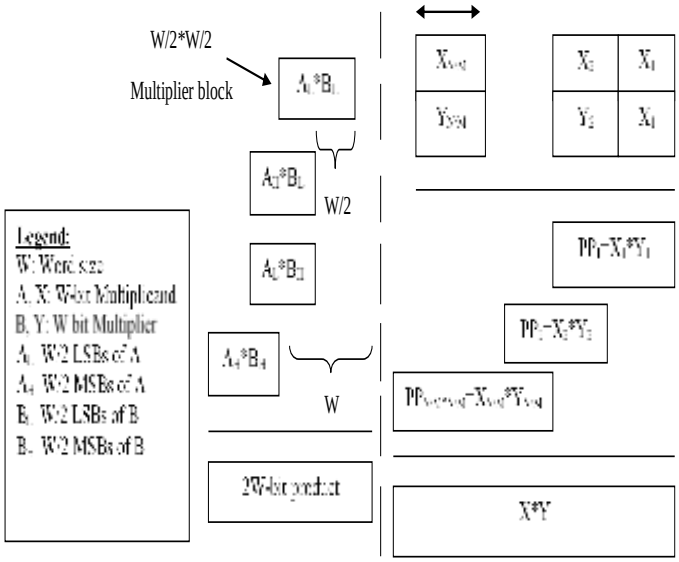
APPROXIMATE MULTIPLIERS

Efficient hardware multiplication is usually done in three stages: (1) generating partial products; (2) intermediate partial product accumulation; and (3) accumulation of the final results by fast adder. Approximations may be used at each point to increase the power, area, efficiency or output performance of the multipliers. Next approach to design intensive multipliers would be to utilize the smaller multipliers as basic components to build larger multipliers. Figure 1.2 shows an illustrative example of such techniques.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001



Throughout the traditional digital architecture, the output metrics are used to define the circuit are: 1) power consumption, 2) latency, 3) critical path delay and 4) silicon area. Likewise, tradeoffs for the attainment of specific operational requirements are often within the same criteria. Nevertheless, in the context of Approximate Computing, there is an additional criterion of efficiency, that is, computational accuracy. Approximate circuits are then expected to be eligible in the aspects of the error statistics.

3. PROPOSED SYSTEM

The proposed system entails the development of a configurable Finite Impulse Response (FIR) filter employing innovative VLSI techniques to optimize performance and versatility. Central to this design is the integration of a Booth Multiplier within the multiplier stage and an Approximate Carry Look-Ahead Adder in the adder stage. This strategic integration enhances both the efficiency and speed of the filter's operations.

The utilization of a Booth Multiplier revolutionizes the multiplication process by significantly reducing the number of partial products required. This reduction not only conserves hardware resources but also minimizes power consumption, making the filter more energy-efficient. Furthermore, the incorporation of an Approximate Carry Look-Ahead Adder accelerates addition operations, thereby augmenting the overall throughput of the filter.

One of the distinguishing features of this FIR filter design is its configurability. Users have the flexibility to adjust critical parameters such as tap coefficients and filter length, tailoring the filter's behavior to meet specific application demands. This configurability empowers the filter to adapt seamlessly to diverse signal processing tasks, enhancing its utility across various domains. To validate the efficacy of the proposed system, extensive simulations and hardware synthesis are conducted. These assessments provide comprehensive insights into the system's functionality, performance metrics, and suitability for real-world applications. The results demonstrate not only the feasibility of the design but also its superiority in terms of speed, area efficiency, and power consumption compared to traditional FIR filter implementations. In essence, the proposed system represents a significant advancement in FIR filter technology, offering a potent solution for real-time signal processing applications. By leveraging cutting-edge VLSI techniques, the system delivers enhanced performance, efficiency, and flexibility, making it an invaluable asset in fields such as telecommunications, image processing, and digital signal processing.

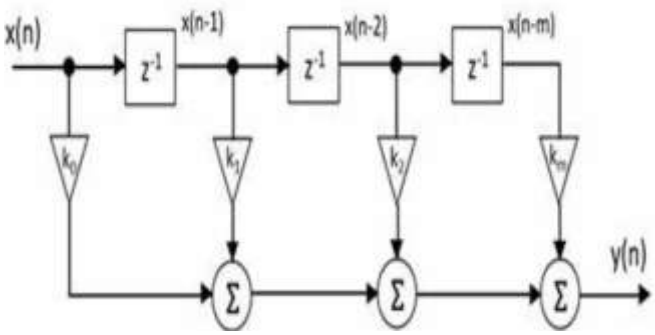


Figure.3 Proposed system

www.ijprems.com
editor@ijprems.com
INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642
e-ISSN : 2583-1062
Impact Factor : 7.001

The methodology adopted in this project revolves around the development of a novel approach to approximate multiplier design, focusing on leveraging accuracy-adjustable carry look-ahead (CLA) approximation techniques to enhance energy efficiency. The process begins with an in-depth review of existing literature in approximate computing and multiplier design, laying the groundwork for identifying innovative solutions. Building upon this understanding, novel accuracy adjustable CLA approximation techniques are formulated to enable flexible adjustment of computational accuracy, facilitating trade-offs between accuracy and energy efficiency. Subsequently, a reconfigurable multiplier architecture is conceptualized, integrating the proposed approximation techniques and allowing seamless switching between exact and inexact operating modes based on specific application requirements. This architecture is then translated into hardware description language (HDL) to facilitate FPGA-based implementation and simulation using tools like Xilinx 12.1. Through rigorous simulation and validation processes, the performance metrics of the proposed design, including power consumption, delay overhead, and accuracy levels, are thoroughly assessed and validated against established benchmarks. Comparative analyses with existing designs further elucidate the advantages of the proposed approach. Iterative optimization cycles are conducted to fine-tune design parameters and maximize efficiency, guided by feedback from simulation results and validation tests. Ultimately, this methodology aims to advance the state-of-the-art in approximate computing techniques, particularly in multiplier design, enabling energy-efficient computation with adjustable accuracy levels to meet diverse application requirements

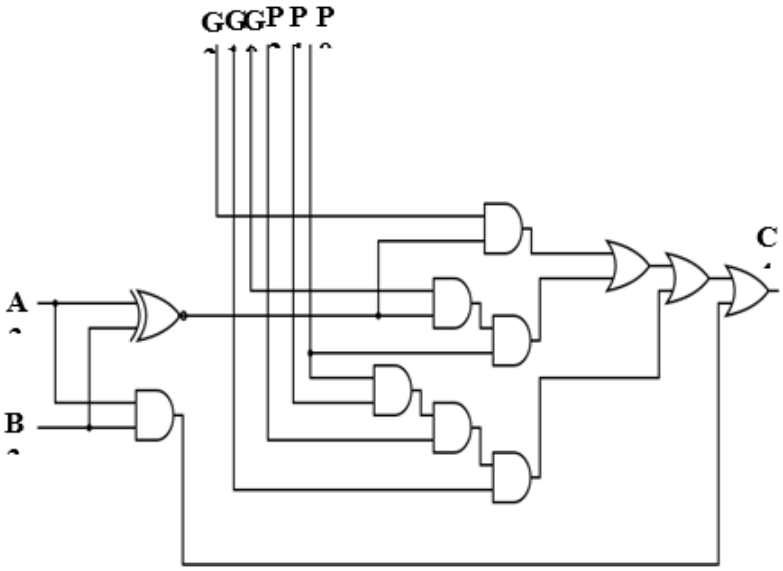


Fig. 4. Logic design of the voter

It is clear that computing imprecise C_{i+1} is faster and consumes less power compared to computing C_{i+1} precisely. Based on this segmentation, two exact and approximate operating modes are realized for the proposed reconfigurable approximate adder. Thus, compared to the conventional CLA, only one multiplexer is added in the circuitry for computing C_{i+1}. The inputs of this multiplexer are approximate and augmenting carry outputs with the operating mode signal as the selector. The operating mode signal determines the generation of the carry output in the exact or approximate mode. As an example, the gate-level netlist of calculating C_4 in the proposed structure which is called RAPCLA is shown in Fig. 1. To avoid unnecessary power consumption of the circuit of the augmenting part, the power gating technique is invoked. In order to remove the power consumption of augmenting part when the carry generation block is in the approximate operating mode, this part is power gated using p-MOS headers.

In the proposed adder, by exploiting a multiplexer in each carry generator block of the conventional CLA, a reconfigurable CLA is obtained. If all carry generators of an adder are based on the proposed structure, and all the operating signals of these units are controlled by only one signal for the operating mode, all output bits of the adder in the approximate mode may be inaccurate. One may increase the accuracy of the approximate mode by exploiting the exact carry generator for the most significant group of bits of the adder similar to the approach suggested. In fact, the technique may be utilized in designing CLAs with different levels of accuracy. For structures with adjustable accuracy, the adder should be partitioned into some segments with their own operating mode signals. At this point, the voter can still tolerate another fault that does not overlap with the existing fault (i.e for some other combination of 'M1,M2,M3'). Given a repair mechanism such as CRAM scrubbing is active to correct the existing fault, it is also possible to tolerate overlapping non-concurrent faults

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001

4. SOFTWARE DESCRIPTION

4.1 VERILOG

In electronics, a hardware description language (HDL) is a specialized computer language used to program the structure, design and operation of electronic circuits and most commonly, digital logic circuits.

A hardware description language enables a precise, formal description of an electronic circuit that allows for the automated analysis, simulation, and simulated testing of an electronic circuit. It also allows for the compilation of an HDL program into a lower level specification of physical electronic components, such as the set of masks used to create an integrated circuit.

A hardware description language looks much like a programming language such as C, it is a textual description consisting of expressions, statements and control structures. One important difference between most programming languages and HDLs is that HDLs explicitly include the notion of time.

HDLs form an integral part of Electronic Design Automation (EDA) systems, especially for complex circuits, such as microprocessors.

Motivation

Due to the exploding complexity of digital electronic circuits since the 1970s (see Moore's law), circuit designers needed digital logic descriptions to be performed at a high level without being tied to a specific electronic technology, such as CMOS or BJT. HDLs were created to implement register-transfer level abstraction, a model of the data flow and timing of a circuit.

There are two major hardware description languages: VHDL and Verilog. There are different types of description in them "dataflow, behavioral and structural".

Structure of HDL

HDLs are standard text-based expressions of the structure of electronic systems and their behavior over time. Like concurrent programming languages, HDL syntax and semantics include explicit notations for expressing concurrency. However, in contrast to most software programming languages, HDLs also include an explicit notion of time, which is a primary attribute of hardware. Languages whose only characteristic is to express circuit connectivity between a hierarchy of blocks are properly classified as netlist languages used in electric computer-aided design (CAD). HDL can be used to express designs in structural, behavioral or register-transfer-level architectures for the same circuit functionality in the latter two cases the synthesizer decides the architecture and logic gate layout.

HDLs are used to write executable specifications for hardware. A program designed to implement the underlying semantics of the language statements and simulate the progress of time provides the hardware designer with the ability to model a piece of hardware before it is created physically. Its executability that gives HDLs the illusion of being programming languages, when they are more precisely classified as specification languages or modeling languages. Simulators capable of supporting discrete-event (digital) and continuous-time (analog) modeling exist, and HDLs targeted for each are available.

4.2 DESIGN USING HDL

As a result of the efficiency gains realized using HDL, a majority of modern digital circuit design revolves around it. Most designs begin as a set of requirements or a high-level architectural diagram. Control and decision structures are often prototyped in flowchart applications, or entered in a state diagram editor. The process of writing the HDL description is highly dependent on the nature of the circuit and the designer's preference for coding style. The HDL is merely the 'capture language', often beginning with a high-level algorithmic description such as a C++ mathematical model. Designers often use scripting languages such as Perl to automatically generate repetitive circuit structures in the HDL language. Special text editors offer features for automatic indentation, syntax-dependent coloration, and macro-based expansion of entity/architecture/signal declaration.

The HDL code then undergoes a code review or auditing. In preparation for synthesis, the HDL description is subject to an array of automated checkers. The checkers report deviations from standardized code guidelines, identify potential ambiguous code constructs before they can cause misinterpretation, and check for common logical coding errors, such as dangling[jargon] ports or shorted outputs. This process aids in resolving errors before the code is synthesized.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642
www.ijprems.com
editor@ijprems.com
e-ISSN : 2583-1062
Impact Factor : 7.001

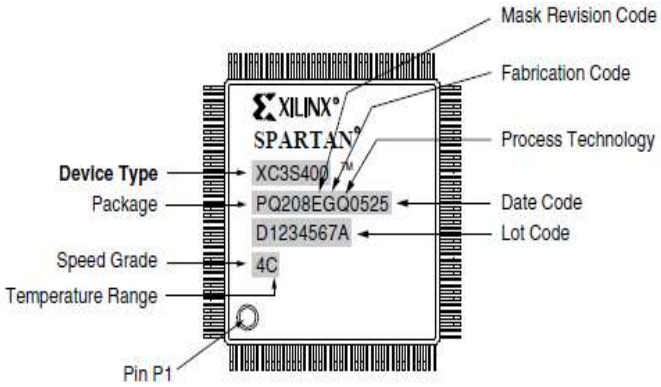


Figure 5. Spartan-3 Package XC3S400-4PQ208C

4.3 CONFIGURABLE LOGIC BLOCK

The Configurable Logic Blocks (CLBs) constitute the main logic resource for implementing synchronous as well as combinatorial circuits. Each CLB comprises four interconnected slices as shown in Fig.6.3. These slices are grouped in pairs. Each pair is organized as a column with an independent carry chain.

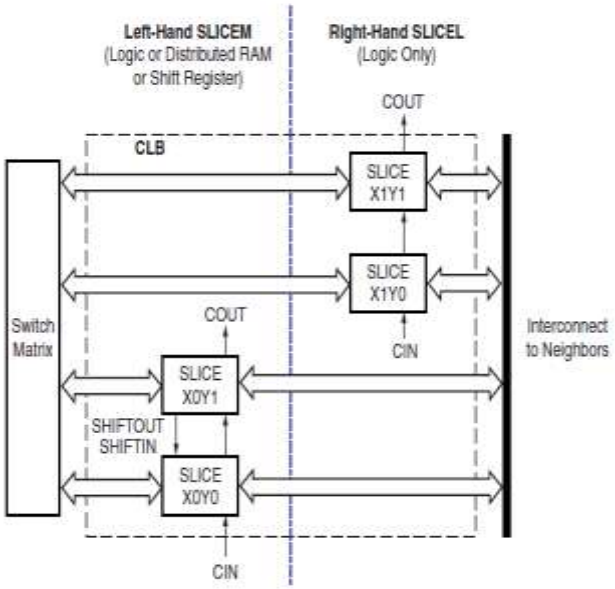


Figure .6 Arrangement of Slices within the CLB

5. SIMULATION RESULTS

The proposed FIR filter was coded in Verilog and simulated using Xilinx ISE 12.1 with the Spartan-3E FPGA platform for evaluation. This evaluation focused on key performance metrics, including slice count, lookup tables (LUT) utilization, and delay to assess the hardware efficiency and speed of the design. The two multiplier variants—one optimized for speed and power efficiency and the other for balanced accuracy and performance—were tested separately to compare their behavior under different configurations.

1. Slice Count and Area Efficiency

The design's slice count was significantly lower than traditional multipliers due to the novel dual sub-adder architecture, which effectively reduces the hardware complexity by splitting the precise adder into two approximate sub-adders. This reduction in slice count results in a more area-efficient multiplier, making it suitable for resource-constrained environments like embedded systems. The first variant, which maximizes approximation, demonstrated a further decrease in slice usage compared to the more accurate variant.

2. LUT Utilization

LUT usage was measured to analyze how efficiently the multiplier maps onto the FPGA's configurable logic. The approximate adder structure led to a reduction in LUT count, as it avoids complex addition stages typically required in precise designs. The accuracy configuration module also played a role in maintaining optimal LUT usage by dynamically managing approximation based on the application's requirements. The less accurate variant showed a more significant reduction in LUT usage, benefiting applications prioritizing power and speed.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001

Delay Analysis

The delay performance was also measured, showing a noticeable reduction compared to conventional multipliers. The dual sub-adder approach helps to minimize critical path delays, allowing faster propagation of partial sums.

The high-speed variant achieved the lowest delay, making it highly suitable for applications where rapid computation is critical. However, the balanced variant showed slightly higher delay due to additional error recovery steps but still performed efficiently within acceptable limits.

The Spartan-3E FPGA results thus demonstrate that the proposed approximate multiplier offers substantial improvements in area, power efficiency, and speed while retaining configurable accuracy, making it an ideal choice for error-resilient applications requiring high performance with minimal resource utilization.

The findings indicate that this design could be a practical solution for real-world scenarios where computational efficiency outweighs the need for exact precision.

The approximate adder structure led to a reduction in LUT count, as it avoids complex addition stages typically required in precise designs. The accuracy configuration module also played a role in maintaining optimal LUT usage by dynamically managing approximation based on the application's requirements. The less accurate variant showed a more significant reduction in LUT usage, benefiting applications prioritizing power and speed.

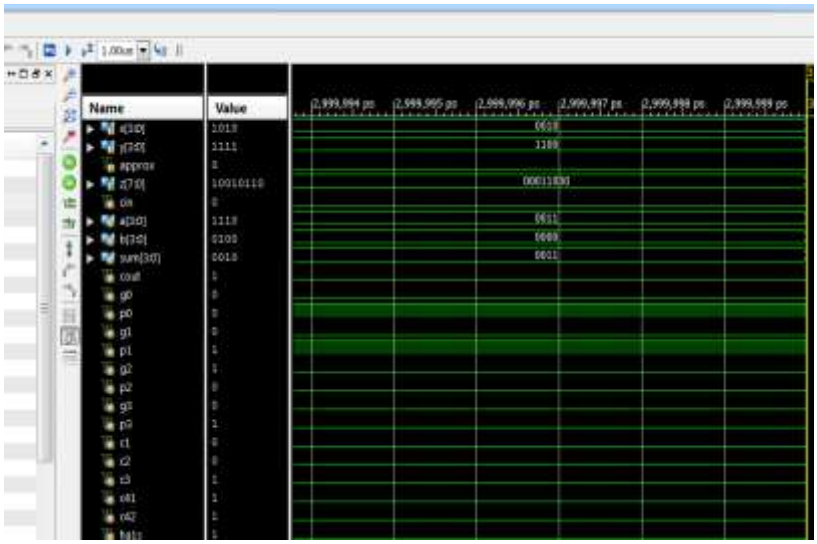


Fig 7. simulation result

Product Version:	ISE 12.1	• Warnings:	13 Warn
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	5	960	
Number of 4 input LUTs	10	1920	
Number of bonded IOBs	14	66	

Detailed Reports				
Report Name	Status	Generated	Errors	Warnings
Synthesis Report	Current	Thu Nov 23 12:40:14 2017	0	13 Warnings (1 new)
Translation Report				
Map Report				
Place and Route Report				
Power Report				

Fig.8 Design Summary of Existing

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 05, May 2025, pp : 632-642

e-ISSN : 2583-1062
Impact Factor : 7.001

Fig.9. Design Summary of Proposed

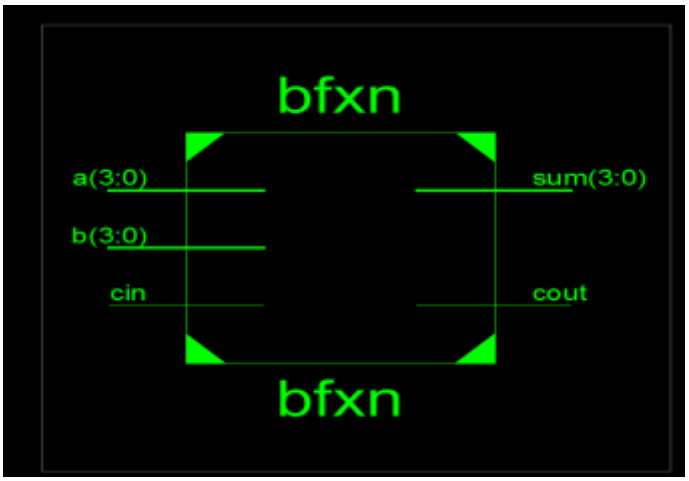


Fig.10 RTL Schematic

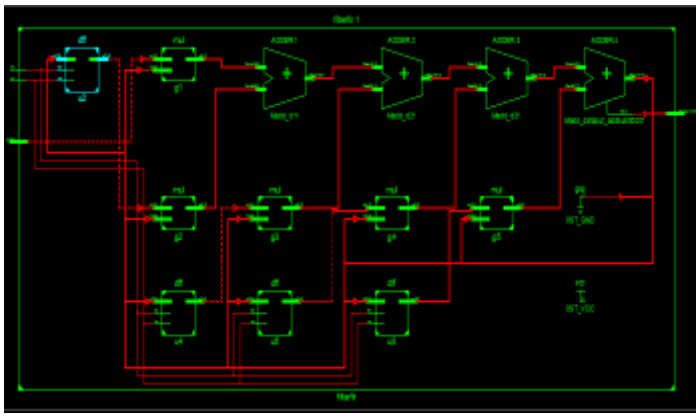


Fig.11 Gate level Netlist

6. CONCLUSION

In conclusion, the development of the configurable Finite Impulse Response (FIR) filter utilizing advanced VLSI techniques marks a significant stride towards efficient and versatile signal processing solutions. Through the use of an Approximate Carry Look-Ahead Adder, the proposed system achieves notable improvements in performance metrics such as speed, area efficiency, and power consumption.The utilization of an Approximate Carry Look-Ahead Adder accelerates addition operations, thereby bolstering the overall throughput of the filter. These advancements not only optimize the filter's performance but also make it well-suited for real-time signal processing applications across various domains.Moreover, the configurability of the FIR filter empowers users to tailor its behavior to specific application requirements, enhancing its adaptability and versatility. By allowing adjustments to critical parameters such as tap coefficients and filter length, the system can efficiently address a wide range of signal processing tasks with precision and efficacy.

7. REFERENCES

[1] L. Bandil and B. C. Nagar, "Dynamic Input Pruning-Based Low-Power and Error-Optimized Approximate Adder," 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kamand, India, 2024, pp. 1-4, doi: 10.1109/ICCCNT61001.2024.10725204.

[2] Kumari and R. P. Palathinkal, "Design and Analysis of Energy Efficient Approximate Multipliers for Image Processing and Deep Neural Network," in IEEE Transactions on Circuits and Systems I: Regular Papers, doi: 10.1109/TCSI.2024.3479279.

[3] Chieh-Wei Wu Wu, L. -C. Wang, W. -X. Chen, Y. -X. Hong and S. -Y. Lin, "A Low-area Hardware and Architecture Design of Truncation- and Rounding-Based Approximate Multiplier for Artificial Intelligence Applications," 2024 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan), Taichung, Taiwan, 2024, pp. 561-562, doi: 10.1109/ICCE-Taiwan62264.2024.10674167.

[4] J. Vafaei and O. Akbari, "HPR-Mul: An Area and Energy-Efficient High-Precision Redundancy Multiplier by Approximate Computing," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 32, no. 11, pp. 2012-2022, Nov. 2024, doi: 10.1109/TVLSI.2024.3445108.

[5] S. Sivasubramani, B. Paikaray, M. Kuchibhotla, A. Haldar, C. Murapaka and A. Acharyya, "Area Efficient Skyrmion Logic based Approximate Adder Architecture Design Methodology," in IEEE Transactions on Emerging Topics in Computing, doi: 10.1109/TETC.2024.3434723.

[6] Srinivasulu et al., "Efficient Approximate Adders for Fast Arithmetic in Energy-Saving Applications," 2024 16th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Iasi, Romania, 2024, pp. 1-4, doi:

[7] J. S. Ubhi, J. K. Yadav and C. Goyal, "Design of CNFET Based Approximate Full Adder for Motion Detector Application," 2024 1st International Conference on Smart Energy Systems and Artificial Intelligence (SESAI), Mauritius, 2024, pp. 1-5, doi: 10.1109/SESAI61023.2024.10599443.

[8] B. Y. Kong, "Low-Latency PAPR Reduction Architecture for Discrete Multitone Based on Approximate Midrange," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, doi: 10.1109/TVLSI.2024.3430094.

[9] M. -G. Lin, J. -P. Wang, C. -Y. Chang and A. -Y. A. Wu, "Approximate Adder Tree Design with Sparsity-Aware Encoding and In-Memory Swapping for SRAM-based Digital Compute-In-Memory Macros," 2024 IEEE 6th International Conference on AI Circuits and Systems (AICAS), Abu Dhabi, United Arab Emirates, 2024, pp. 362-366, doi: 10.1109/AICAS59952.2024.10595967.

[10] F. Seiler and N. TaheriNejad, "Accelerated Image Processing Through IMPLY-Based NoCarry Approximated Adders," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 71, no. 11, pp. 5141-5154, Nov. 2024, doi: 10.1109/TCSI.2024.3426926.

[11] Z. Cui, K. Chen, B. Wu, C. Yan, Y. Gong and W. Liu, "A Time Efficient Comprehensive Model of Approximate Multipliers for Design Space Exploration," 2024 IEEE 31st Symposium on Computer Arithmetic (ARITH), Malaga, Spain, 2024, pp. 116-123, doi: 10.1109/ARITH61463.2024.00028.

[12] N. Kaushik, H. Krishna and S. Bodapati, "High-Speed Serial and Semi-Parallel IMPLY-based Approximate Adders through Memristors for In-Memory Computing," 2024 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, Singapore, 2024, pp. 1-5, doi: 10.1109/ISCAS58744.2024.10558550.

[13] L. H. Krishna, N. Kaushik and S. Bodapati, "Energy Efficient Accurate and Approximate Modified Adders for Ternary Multipliers," 2024 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, Singapore, 2024, pp. 1-5, doi: 10.1109/ISCAS58744.2024.10558406.

[14] P. Goswami and R. Biswas, "High Speed Fault Tolerant Approximate Adder for Low Power Application," 2024 IEEE 4th International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI SATA), Bangalore, India, 2024, pp. 1-5, doi: 10.1109/VLSISATA61709.2024.10560161