RoBA Multiplier-Driven FIR Filter Synthesis: Uniting Efficiency and Speed for Enhanced Digital Signal Processing

Bandi Sathwik[1], D.Sudha[2], Y Aruna Suhasini Devi[3], Jagadeesh Bodapati [4]

[1]*UGC Student, Dept. of ECE, CMR College of Engineering & Technology, Telangana.*
[2]*Professor,Dept. of ECE, CMR College of Engineering & Technology, Telangana.*
[3]*Associate Professor,Dept. of ECE, CMR College of Engineering & Technology, Telangana.*
[4]*Professor , Dept. of ECE, BVC College of Engineering,Rajamundry, Andhra Pradesh.*

KEYWORDS:
Approximation multiplier,
Energy-efficient computation,
Accuracy trade-off,
Performance evaluation,
FIR filter,
Multiplication strategies,
Speed optimization.

ARTICLE HISTORY:
Received 09.04.2024
Accepted 13.05.2024
Published 03.06.2024

DOI:
https://doi.org/10.31838/jvcs/06.02.03

Abstract

The main objective of this project includes, enhance the computational efficiency of DSP systems which make them operate more efficiently. Traditional DSP computations can be slow, so sometimes it's more important to be fast than completely accurate. To craft a Digital Signal Processor (DSP) systems even faster and save energy, researchers have designed circuits that approximate calculations, sacrificing a bit of precision. By this proposed solution, we designed a RoBA multiplier type for filters that rounds numbers to the closest whole number. By simplifying the multiplication process, this approximation reduces system's size and speeds up the operation. Considering the fact that the multiplier typically functions at a slower pace compared to other components, this adjustment is anticipated to enhance the filter's overall efficiency. We compared the Vedic multiplier to the proposed ROBA multiplier. The acquired results showcase the power, area, number of slice LUTs, number of bonded Input/Output Buffers, and delay related to this multiplier and FIR filters were greatly decreased, while the multiplier speed increased proportionally.

Author e-mail: bandisathwiksunny@gmail.com, dsudha@cmrcet.ac.in, yarunasuhasini@cmrcet.ac.in, bjagadeesh2020@gmail.com

How to cite this article: Sathwik B, Sudha D, Devi ASY, Bodapati J. RoBA Multiplier-Driven FIR Filter Synthesis: Uniting Efficiency and Speed for Enhanced Digital Signal Processing, Journal of VLSI Circuits and System Vol. 6, No. 2, 2024 (pp. 23-30).

1 INTRODUCTION

Digital filters are like super-smart tools used in electronics. They're better at reducing noise compared to old-fashioned analog filters. With analog filters, the noise can get worse, but digital filters keep it under control. They also do math without making mistakes, which helps keep the noise low. Our way of doing things makes sure we use resources wisely while keeping the output signal clear and responsive.

Digital filters are like mathematical tools that help clean up signals, like sound or images, in electronic devices. They work by doing basic math operations like adding, multiplying, and delaying the signal. There exist two primary categories: FIR and IIR. Finite Impulse Response, or FIR, is a sort of filter that is easy to understand and use. It doesn't use feedback and keeps the signal's timing consistent to avoid messing it up. FIR filters are good for fast processing, which makes them popular in things like digital sound processors. In Digital Signal Processing, we often need to add and multiply numbers quickly. To do this, we use special circuits like parallel prefix adders for fast addition and modified multipliers with fewer parts to reduce delays.[2]

Filters, such as FIR (Finite Impulse Response) filters, function well with finite precision since we only have to worry about precisely creating a certain amount of bits when using them. However, feedback loops in IIR (Infinite Impulse Response) filters might be problematic.FIR filters, however, don't need feedback to work properly, which makes them simpler to use. In FIR filters, we can even use fractional arithmetic, which means working with numbers that aren't whole, like 1.5 or 0.75. And multipliers are really important in DSP because they're used a lot for these filtering operations.

In multiplication, an operation requiring a considerable quantity of hardware, the main areas of interest are higher speed, lower cost, and less VLSI space. Propagation latency and consumption of power are the two primary, albeit usually contradictory, design objectives.[2,5,25,26] Given these constraints, building low power multipliers[23,25,26] is particularly desired. The main objective within the context of this project is to provide an approximate, high-speed, low-power/energy multiplier that is suitable for error. Here is an overview of the contributions made by this work:

1) Three hardware setups for the approximation multiplication method are shown, which may be used for both signed and unsigned operations.[1]

2) Proposing an entirely new ROBA multiplication strategy by adjusting the standard multiplication approach. The approximation multiplier of the suggested FIR filter is likewise area-efficient and is built by algorithmically altering the traditional multiplication technique, assuming rounded input values.[1] The structure of the rest of this document is as follows. Section II discusses the prior works about approximate multipliers.[23] In Section III, the Existing Methodology of Vedic Multiplier is studied. Section IV describes the proposed methodology and finally Results and Conclusion are mentioned and compared the Existing and Proposed Methodologies.

2 LITERATURE SURVEY

Significant progress has been made in the domain of approximate computation, especially concerning multiplier design, as demonstrated by Gupta et al.[3] introduced various approximate adder types that enhance power efficiency,[23,25,27] area utilization, and performance by streamlining the logic employed in traditional mirror adders. Meanwhile, Mandeni et al.[4] proposed a bioinspired approach utilizing OR gates to approximate addition results within the multiplier, particularly focusing on the lower bits of inputs. In a parallel effort, novel metrics and techniques were introduced by researchers[5-7] to assess the effectiveness of approximate adder modeling. Building on these foundations, Kulkarni et al.[6] presented an approximation multiplier based on erroneous building blocks, yielding notable power savings compared to correct multipliers.

Expanding upon the concept of approximation multipliers, scientists have investigated different approaches and applications. For instance, an approximate signed Booth multiplier proposed by leveraging broken-array multiplier (BAM) techniques demonstrated substantial power and area savings compared to conventional Booth multipliers.[5,23] Furthermore, the literature has delved into error-resilient systems, with studies like that of,[8] which introduced an accuracy-configurable multiplier architecture (ACMA) featuring carry in prediction strategies, significantly reducing latency while maintaining accuracy.

In pursuit of more efficient multiplication operations, researchers have proposed innovative approaches or example, dynamic segment method (DSM)[16] as well as the utilization of logarithmic approximation.[12] DSM, as described by prior works,[16,17] operates on truncated values derived from input operands’ leading one bits, offering a promising avenue for reducing complexity and improving efficiency. Additionally, Bhardwaj et al.[11] introduced the approximation Wallace tree multiplier (AWTM), employing carry-in forecast techniques to reduce critical paths and enhance performance in real-time image processing applications.

Notably, recent studies[12,18,26] have focused on refining approximation strategies specifically for unsigned integers, demonstrating improvements in mean error rates. Unlike previous approaches, these methodologies offer unique insights into operand breakdowns and hardware expansion, leading to more accurate approximations. Moreover, the proposed solutions aim to address the challenges posed by multiplication operations involving signed integers, indicating paths for additional investigation and optimization in this domain.

3 EXISTING SYSTEM

1] VEDIC MULTIPLIER

Here, the Physical Architecture related to Vedic Multiplier Module’s for 2X2, 4X4, also 8X8 bit Operations are illustrated in below sections. This design utilizes the “Urdhva-Tiryagbhyam” sutra of Binary Numbers multiplication. The advantages of this existing multiplier is that, it enables generation of partialized product, at a same time addition, where Parallel Processing can be achieved by this means. Hence the project mainly focuses on the capability of reducing the significant latency.[20]

2] DIVINE MULTIPLICATION ALGORITHM FOR 8X8 BIT INPUTS

An 8X8 Bit Vedic-Multiplier depicted in Figure 3.1 block diagram can be constructed by assembling four of 4X4 bit multiplier modules, as described earlier. For an 8x8 multiplication operation with input bit representations A = (A7) (A6) (A5) (A4) (A3) (A2) (A1) (A0), and also B = (B7) (B6) (B5) (B4) (B3) (B2) (B1) (B0), the resulting product will span 16 bits, denoted as (S15) (S14) (S13) (S12) (S11) (S10) (S9) (S8) (S7) (S6) (S5) (S4) (S3) (S2) (S1) (S0). So to promote the computation, the distribution of the inputs A and B into two equal halves is done as AH-AL and BH-

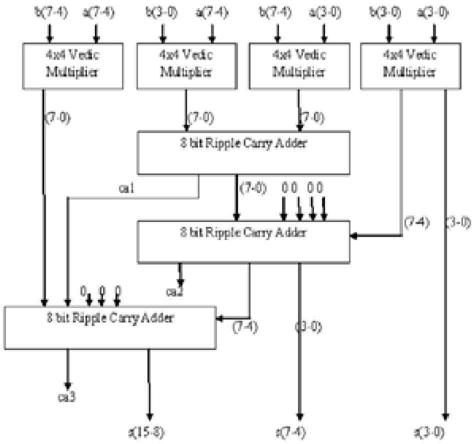


Fig. 3.1: An 8X8 Bit Vedic Multiplier's Architecture

BL, respectively. Utilizing the 4-bit multiplier blocks and Vedic multiplication principles, multiplication is performed four bits at a time. The outputs from the 4X4 bit multipliers are sequentially summed up using three 8-bit Ripple-Carry Adders, as shown as in Figure 3.1. This method offers adaptability in managing input bit pairs by organizing AH-AL and BH-BL pairs accordingly.[20]

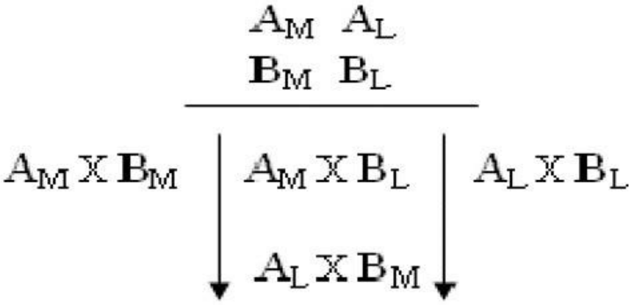


Fig. 3.2: General characterization of Vedic Multiplication.

Suppose we have two N-bit binary numbers as A, B where A = (A) (AN) (A2) (A1), B = (B) (BN) (B3) (B2) (B1). When multiplying these two numbers, here the 'N' can be any positive integer, the resulting product should have twice of no. of Bits, denoted as S (N + N). Hence final result would be consisting of bits S3 S2 S1.[20,21]

Step 1: To compute the multiplication, we first divide both the multiplicand(A) and the multiplier(B) divided evenly into two segments, each containing bits from N to (N/2) + 1 and from N/2 to 1, respectively. The First portion represents the Most Significant bits (MSB), and the Second portion represents the Least Significant bits (LSB). Step 2: Represent the components of A as AM and AL, and the components of B as BM and BL. At this stage, A is presented as AM AL and B as BM BL. Step 3: Multiplication process of A by B follows a general format, as illustrated in Figure 3.2, depicting the overall Vedic Multiplication Representation.[22]

4 PROPOSED SYSTEM

An algorithm for multiplying the RoBA Multiplier

The primary objective of the proposed approximation multiplier is to capitalize on the simplicity inherent in 2n numbers. To elucidate the operational mechanics of this multiplier, we begin by identifying the rounded numbers Ar and Br, corresponding to inputs A and B, respectively. The multiplication of A by B is delineated as the product of (Ar – A) and (Br – B), as well as Ar × B, Br × A, and Ar × Br. Notably, the calculation of Ar × Br, Ar × B, and Br × A can be efficiently accomplished through shift operations alone. However, tackling the term (Ar − A) × (Br − B) poses a more intricate challenge. Recognizing that the discrepancy between exact and rounded numbers typically exerts minimal influence on the outcome, we propose streamlining the multiplication process by omitting this term. Consequently, the multiplication operation is effectively conducted utilizing the expression: A × B ~= Ar × B + Br × A − Ar × Br. [1] The process of duplication involves a sequence of three steps and two expansion/subtraction tasks. Achieving the closest values for An and B, approximated as 2n, necessitates careful matching. When An (or B) nears 3 × 2p−2, where p is a positive integer exceeding one, two values closest to 2n emerge with absolute differences of 2p and 2p−1. Opting for the larger value (except when p = 2) minimizes hardware complexity in computing the nearest adjusted value, albeit both options yield comparable effects on the proposed multiplier's accuracy.[23] Utilizing these values in assembly scenarios facilitates shorter justification expressions.[2] The premise that numbers akin to 3 × 2p−2 remain unaffected by disentanglement and assembly serves as a foundational principle. Additionally, the proposed approximate multiplier emphasizes three as the closest motivator. In prior research, where assessed outcomes deviated from the actual result, the resultant output from the RoBA multiplier may vary, either surpassing or falling short of the true result, contingent upon the relative magnitudes of Ar, Br, An, and B. In instances where one operand (An) falls short of its corresponding balanced value while the other operand (B) exceeds its balanced counterpart, the estimated outcome is expected to exceed the correct result. This discrepancy arises from the negative expansion of (Ar - A) × (Br - B). Conversely, when both An and B surpass Ar and Br or when they both fall below Br, the anticipated result will fall below the actual value. Notably, the RoBA multiplier's effectiveness is limited to scenarios involving positive inputs, as negative inputs do not benefit from its enhancements.

Therefore, our suggestion entails resolving any disparities between the expansion yield and the noncomparable values between the pair input sources before commencing the enlargement process. Subsequently, the task concerning unsigned numbers must be concluded, leading to the alignment of the unsigned result with the optimal sign. A detailed exposition on the hardware utilization of the proposed approximate multiplier ensues promptly thereafter.

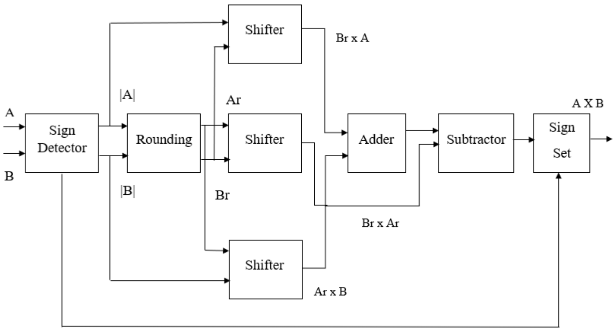


Fig. 4.1: Block Diagram for the Proposed Multiplier.

Implementation of proposed Multiplier in Hardware:

$$
\begin{aligned}
A_r[n-1] &= \overline{A[n-1]} \cdot A[n-2] \cdot A[n-3] \\
&\quad + A[n-1] \cdot \overline{A[n-2]} \\
A_r[n-2] &= (\overline{A[n-2]} \cdot A[n-3] \cdot A[n-4] \\
&\quad + A[n-2] \cdot \overline{A[n-3]}) \cdot \overline{A[n-1]} \\
&\vdots \\
A_r[i] &= (\overline{A[i]} \cdot A[i-1] \cdot A[i-2] + A[i] \cdot \overline{A[i-1]}) \cdot \prod_{i=i+1}^{n-1} \overline{A[i]} \\
&\vdots \\
A_r[3] &= (\overline{A[3]} \cdot A[2] \cdot A[1] + A[3] \cdot \overline{A[2]}) \cdot \prod_{i=4}^{n-1} \overline{A[i]} \\
A_r[2] &= A[2] \cdot \overline{A[1]} \cdot \prod_{i=3}^{n-1} \overline{A[i]} \\
A_r[1] &= A[1] \cdot \prod_{i=2}^{n-1} \overline{A[i]} \\
A_r[0] &= A[0] \cdot \prod_{i=1}^{n-1} \overline{A[i]}. \qquad (3)
\end{aligned}
$$

Referring to (2), we present proposed multiplier's block diagram, illustrating the data flow within its enhanced structure. Initially, signals from input sources are organized, with each negative signal receiving both internal and external consideration. The modifying block then assigns the nearest power of 2 for each input. As the most significant bit in the two's complement representation of an n-bit integer is zero, it's evident that the output of this block has a width of n bits. By applying the following equation to each output bit of the modifying block, we can determine the closest approximation of data A:

With regard to the mentioned context, Ar[i] assumes one of two potential states. At the beginning scenario, A[i] is devoid of any bits to its left, a condition also applicable to A[i − 1]. Alternatively, in the second scenario, if A[i] and all preceding bits are zero, both A[i − 1] and A[i − 2] are set to one. Addressing these conditions involves the utilization of three barrel shifters, tasked with computing Ar × Br, Ar × B, and Br × An subsequent to necessary adjustments. The determination of the shifting ratio is facilitated by selecting the An (or B) operand based on logAr 2 − 1 (or logBr 2 − 1). These shifter blocks possess a data bit width of n and yield outputs of 2n. The summation of Ar × B and Br × A is executed through a singular 2n-bit Kogge-Stone adder. This adder's outcome, in conjunction with the result of Ar × Br, feeds into the subtractor block, culminating in the final approximation of the proposed multiplier's output. The subtractor's functionality is contingent upon the chosen data schemes for Ar and Br. Furthermore, corresponding output schemes are detailed in Table I. Given the nature of the input data and outputs, a straightforward circuit design was deemed necessary. Parallel Prefix Adder (PPA) is a key method for achieving parallel addition in microprocessors, DSPs, and mobile devices, optimizing area and power efficiency. Built upon the foundation of the Carry Look Adder, PPA simplifies logic and reduces latency. It operates in three main phases: 1) computing carry generation and propagation signals based on input bit count, 2) simultaneously calculating all carry signals, and 3) evaluating the final sum of input values.

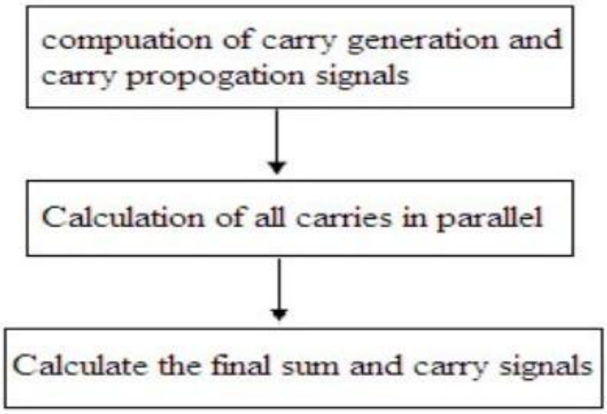

compuation of carry generation and carry propogation signals
Calculation of all carries in parallel
Calculate the final sum and carry signals

Fig. 4.2: Mechanism of a Parallel Prefix Adder

The three procedures or phases that are employed

Table-1 : Comparison Table

System Name	Vedic Multiplier (Existing System)	RoBA Multiplier (Proposed System)
Number of slice LUT's	196	192
Number of bonded IOB's	132	129
DELAY	**22.950ns**	**13.793ns**

in the parallel prefix addition are as follows: 1. Compute the propagation signal and carry creation: The signals produced by the preceding carry, which is computed to the next bit and is referred to as the propagate signal, are the ones below.

Gi is equivalent to Ai. Bi
Pi is equivalent to Ai ⊕ Bi.

2. Compute every carry signal:

$P_{i:j} = P_{i:k}.\ P_{k-1:j}$
$G_{i:j} = G_{i:k} + P_{i:k}.\ G_{k-1}$

3. Compute the Final Sum:

$S_i = P_i \oplus G_{i-1:0}$

There are several varieties of Parallel-Prefix adders, some of which are discussed in this work. The Kogge-Stone adder, Brent-Kung adder, Ladner-Fischer adder, Han-Carlson adder, S. Knowles adder, and Sklansky Conditional-Sum adder are the few Parallel-Prefix adders. Of the adders listed above, Kogge-Stone is frequently encountered and efficiently used.

The Implementation of the FIR filter:

Since the period of the FIR system's impulse response is finite, it contains a finite number of nonzero terms. The input samples from the past and present are the only ones that have an impact on the FIR filter's response.

Typical FIR filter:

$$y[n] = \sum_{k=0}^{N-1} b_k\, x(n-k)$$

The conventional architecture of the FIR filter is depicted in the Figure. To create a FIR, three basic building elements are required: signal delay, addition, and multiplication. The formula for the FIR filter is:

The filter order, denoted by N, consists of filter coefficients represented by bk, and the resulting output signal is denoted by y[n]. Taps or tapped delay lines, denoted by x[n - k], represent past values related to Input Signal x[n]. The standard architecture of a simple multiplier structure for FIR filters involves creating unfinished products through multiplication. When the multiplier digit is 1, the multiplicand is copied down to represent the product; otherwise, the product is zero. This process increases delay and area, impacting FIR filter performance. The speed of the RoBA Multiplier affects the performance of the FIR filter, as it represents slowest component. Hence, using the RoBA Multiplier, which accumulated less space and also faster compared to conventional multipliers, is recommended.

Kogge-Stone multiplier

The Kogge Stone Adder is a parallel-prefix form carry look-ahead adder. Harold S. Stone and Peter M. Kogge developed the Kogge-Stone adder, which was first released in 1973. The KS adder design is a fast adder design because it generates a carry signal in O(log2 n) time and has the best performance in VLSI implementations.[20,27]

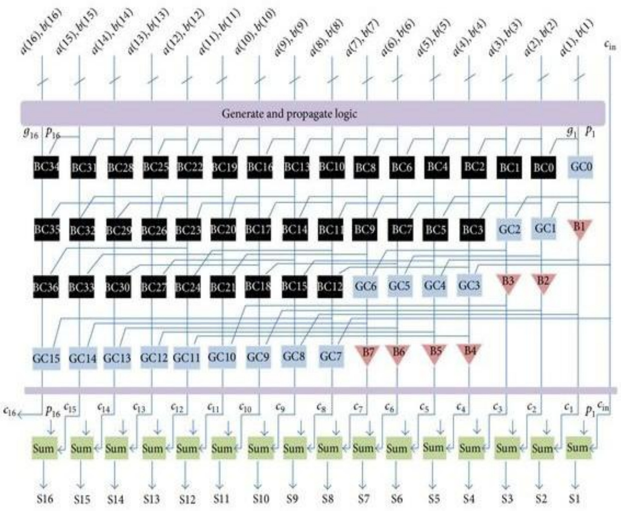

a(16), b(16)
a(15), b(15)
a(14), b(14)
a(13), b(13)
a(12), b(12)
a(11), b(11)
a(10), b(10)
a(9), b(9)
a(8), b(8)
a(7), b(7)
a(6), b(6)
a(5), b(5)
a(4), b(4)
a(3), b(3)
a(2), b(2)
a(1), b(1)
c_{in}
Generate and propagate logic
g_{16} P_{16}
g_1 P_1
BC34 BC31 BC28 BC25 BC22 BC19 BC16 BC13 BC10 BC8 BC6 BC4 BC2 BC1 BC0 GC0
BC35 BC32 BC29 BC26 BC23 BC20 BC17 BC14 BC11 BC9 BC7 BC5 BC3 GC2 GC1 B1
BC36 BC33 BC30 BC27 BC24 BC21 BC18 BC15 BC12 GC6 GC5 GC4 GC3 B3 B2
GC15 GC14 GC13 GC12 GC11 GC10 GC9 GC8 GC7 B7 B6 B5 B4
c_{16} P_{16} c_{15} c_{14} c_{13} c_{12} c_{11} c_{10} c_9 c_8 c_7 c_6 c_5 c_4 c_3 c_2 c_1 P_1 c_{in}
Sum
S16 S15 S14 S13 S12 S11 S10 S9 S8 S7 S6 S5 S4 S3 S2 S1

Fig. 4.3: 16-bit Kogge Stone Adder illustration

The extensive surface area and low fan-out of the KS adder contribute to its good performance.

In high performance 32-bit, 64-bit, and 128-bit adders, the Kogge Stone Adder is frequently used since it significantly reduces the pathway with the highest timing delay. In Figure 4.2, each vertical level originates, propagates, and produces bits.

To facilitate produce the sum, generate bits are created in final stage and XOR'ed[28] with the initial propagate bits.

5 RESULTS

TOP MODULE OF PROPOSED (RoBA) MULTIPLIER

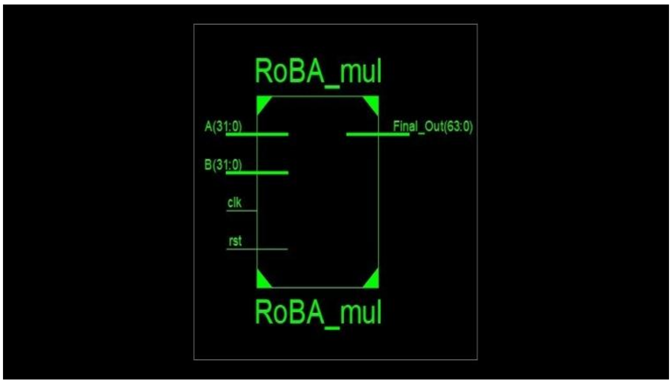


Fig. 5.1: Top Module of a RoBA Multiplier.

At the core of the proposed (Rounding-Based Approximate) Multiplier lies its primary module, functioning as the central control unit overseeing the entirety of the approximate multiplication operation. This pivotal component orchestrates a network of subordinate modules tasked with handling input data, executing approximation algorithms, performing rounding operations, and generating output. Designed to cater to the demands of digital signal processing tasks, the RoBA Multiplier embodies various approximation techniques aimed at optimizing computational speed and energy efficiency,[24] albeit at the expense of absolute precision. Leveraging the principles of rounding-based approximation, this multifaceted module strikes a delicate equilibrium between computational accuracy and resource allocation, rendering it well-suited for real-time processing scenarios where swift performance and minimal energy consumption[24] are paramount considerations.

RTL SCHEMATIC VIEW OF THE RoBA MULTIPLIER

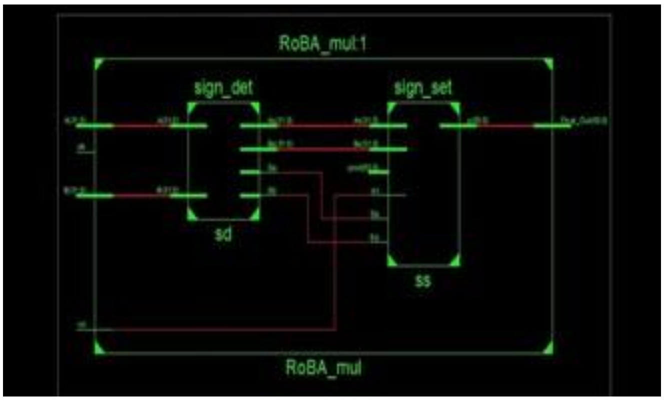


Fig. 5.2: RTL Schematic view of RoBA Multiplier

In the Register Transfer Level (RTL) perspective of a multiplier with Rounding Based Approximation, the focus lies in detailing the functional actions of the multiplication process at a low level of abstraction, typically involving registers and data transfers. This type of multiplier prioritizes efficiency over precise accuracy in its hardware implementation. The RTL view illustrates how input variables are stored in registers, whereby the execution of the multiplication operation using approximate methods like truncated or rounded multiplication, and the recording of the outcome into registers. It also encompasses any control logic needed for rounding decisions and data flow management within the multiplier circuitry. Essentially, the RTL view offers insights into the operational aspects of the approximate multiplier, tailored for hardware implementation and optimization.

WAVEFORMS OF A RoBA MULTIPLIER

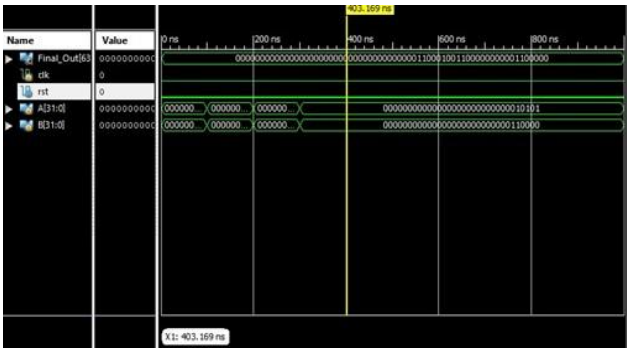


Fig. 5.3: The output waveform of a RoBA Multiplier.

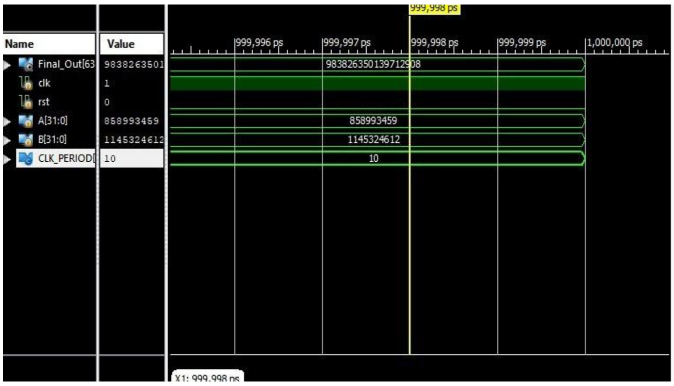


Fig. 5.4: Output Waveforms Radix unsigned form.

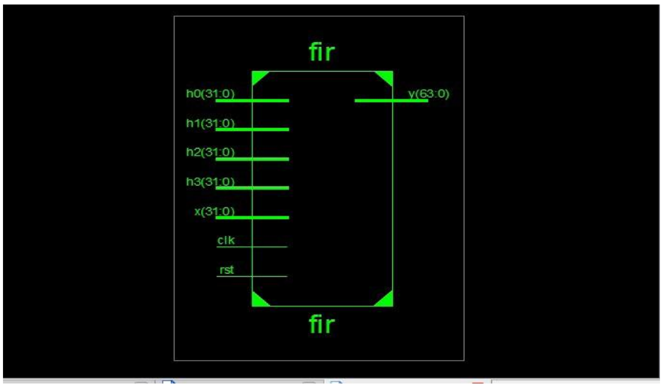
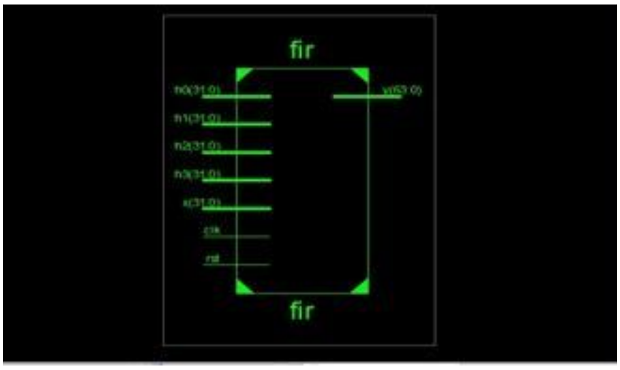


Figure 5.5: Top Module of FIR Filter.

TOP MODULE OF FIR FILTER WITH RoBA MULTIPLIER:



The top module of a Finite Impulse Response (FIR) filter with Rounding Based Approximate Multiplier serves as the interface between the filter's input and output and its internal processing components. It typically includes input and output ports for data communication, control signals for filter configuration, and connections to the rounding-based approximate multiplier unit. This module coordinates the flow of data through the filter, manages multiplier operations with rounding-based approximation for computational efficiency, and ensures accurate output generation while optimizing resource utilization. Additionally, it may incorporate features for coefficient storage, filter configuration, and performance monitoring to facilitate seamless integration into larger digital signal processing systems.

RTL SCHEMATIC VIEW OF RoBA MULTIPLIER INTEGARTED FIR FILTER.

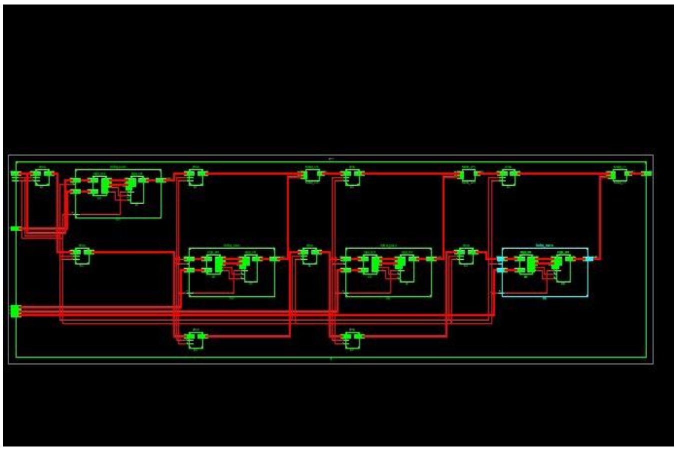


Fig. 5.6: RTL Schematic view of FIR Filter with RoBA Multiplier

The RTL schematic view of FIR filter integrated with RoBA Multiplier typically includes components such as registers, adders, multipliers, and rounding units. In this design, the approximate multiplier operates by compromising accuracy for decreased hardware complexity and power consumption. It employs simplified multiplication algorithms or approximate arithmetic techniques to perform multiplication operations with fewer resources. The rounding unit is responsible for rounding the approximate multiplier's output to the nearest integer, ensuring compatibility with fixed-point arithmetic. This RTL schematic offers a balance between computational efficiency and acceptable filtering performance for applications where stringent accuracy requirements are not critical.

CONCLUSION

This study compares the attributes of suggested Rounding Based Approximate Multiplier (RoBA) with the Vedic Multiplier. Kogge-Stone Adder, Barrel Shifter, and Subtractor are made use in formation of the RoBA Multiplier, which offers notable benefits in areas of space efficiency, consumption of power, Reduction in delay, number of Slice LUTs, and number of bonded IOBs. The FIR Filter's area was lowered in consequences of rounding-based estimate. The proposed multipliers' efficacy was assessed by contrasting their efficiency with a few exact and approximation multipliers employing different structural properties. The effectiveness of the suggested multipliers was observed during comparing them to a few precise and Approximate Multipliers that used various structural features. The Xilinx ISE Simulator was utilized to simulate and implementation of FIR filter and RoBA multiplier, ultimately yielding the aforementioned outcomes.

REFERENCES

1. Zendegani, R., Kamal, M., Bahadori, M., Afzali-Kusha, A., & Pedram, M. (2017). RoBA Multiplier: A Rounding-Based Approximate Multiplier for High-Speed yet Energy-Efficient Digital Signal Processing. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 25(2), 393-401. https://doi.org/10.1109/tvlsi.2016.2587696.

2 Alioto (2016) provided a tutorial elucidating ultra-low power24] VLSI circuit design, demystifying complex concepts and mechanisms in "The Circuits and Systems by IEEE Transactions" I: Regular Papers.

3 Gupta, V., Mohapatra, D., Raghunathan, A., & Roy, K. (2013). Low-Power Digital Signal Processing Using Approximate Adders. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32(1), 124–137. https://doi.org/10.1109/tcad.2012.2217962

4 Mahdiani, H. R., Ahmadi, A., Fakhraie, S. M., & Lucas, C. (2010). Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications. IEEE Transactions on Circuits and Systems I: Regular Papers, 57(4), 850–862. https://doi.org/10.1109/tcsi.2009.2027626.

5 Venkatesan, R., Agarwal, A., Roy, K., & Raghunathan, A. (2011). MACACO: Modeling and analysis of circuits for approximate computing. 2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). https://doi.org/10.1109/iccad.2011.6105401.

6 Farshchi, F., Abrishami, M. S., & Fakhraie, S. M. (2013). New approximate multiplier for low power digital signal processing. The 17th CSI International Symposium on Computer Architecture &amp; Digital Systems (CADS 2013). https://doi.org/10.1109/cads.2013.6714233.

7 Kelly et al. (2009) introduced approximate signed binary integer multipliers for arithmetic data value speculation, presented at the Conference on Design, Automation & Test in Europe.

8 Kyaw et al. (2010) presented a low-power, high-speed multiplier tailored for error-tolerant applications, documented in the proceedings of the IEEE International Conference on Electron Devices and Solid-State Circuits.

9. Momeni, A., Han, J., Montuschi, P., & Lombardi, F. (2015). Design and Analysis of Approximate Compressors for Multiplication. IEEE Transactions on Computers, 64(4), 984–994. https://doi.org/10.1109/tc.2014.2308214.

10. Bhardwaj, K., & Mane, P. S. (2013). ACMA: Accuracy-configurable multiplier architecture for error-resilient System-on-Chip. 2013 8th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC). https://doi.org/10.1109/recosoc.2013.6581532.

11. Bhardwaj, K., Mane, P. S., & Henkel, J. (2014). Power- and area-efficient Approximate Wallace Tree Multiplier for error-resilient systems. Fifteenth International Symposium on Quality Electronic Design. https://doi.org/10.1109/isqed.2014.6783335.

12. Mitchell, J. N. (1962). Computer Multiplication and Division Using Binary Logarithms. IEEE Transactions on Electronic Computers, EC-11(4), 512–517. https://doi.org/10.1109/tec.1962.5219391.

13. Mahalingam, V., & Ranganathan, N. (2006). Improving Accuracy in Mitchell's Logarithmic Multiplication Using Operand Decomposition. IEEE Transactions on Computers, 55(12), 1523–1535. https://doi.org/10.1109/tc.2006.198.

14. The Nangate 45nm Open Cell Library provided essential resources for integrated circuit design.

15. AMETANI, A., T. KAWAMURA, and H. SHOKI. "An advanced state of art design of QCA based applications." International Journal of communication and computer Technologies 11.1 (2023): 85-94.

16. Uvarajan, K. P., and K. Usha. "Implement A System For Crop Selection And Yield Prediction Using Random Forest Algorithm." International Journal of communication and computer Technologies 12.1 (2024): 21-26

17. Narayanamoorthy, S., Moghaddam, H. A., Liu, Z., Park, T., & Kim, N. S. (2015). Energy-Efficient Approximate Multiplication for Digital Signal Processing and Classification Applications. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 23(6), 1180–1184. https://doi.org/10.1109/tvlsi.2014.2333366.

18. Hashemi, S., Bahar, R. I., & Reda, S. (2015). DRUM: A Dynamic Range Unbiased Multiplier for approximate applications. 2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). https://doi.org/10.1109/iccad.2015.7372600.

19. Lin, C.-H., & Lin, I.-C. (2013). High accuracy approximate multiplier with error correction. 2013 IEEE 31st International Conference on Computer Design (ICCD). https://doi.org/10.1109/iccd.2013.6657022.

20. Kahng, A. B., & Kang, S. (2012). Accuracy-configurable adder for approximate arithmetic designs. Proceedings of the 49th Annual Design Automation Conference. https://doi.org/10.1145/2228360.2228509.

21. Nittala, V. B., Bomma, A., & Reddy, M. R. (2022). Energy Efficient Approximate 8-bit Vedic Multiplier. International Journal for Research in Applied Science and Engineering Technology, 10(9), 1453–1465. https://doi.org/10.22214/ijraset.2022.46861.

22. In 2021, Prem Kumar, Shiva Nand Singh proposed "64-bit Vedic Multiplier with Modified Architecture and Improved Performance using Verilog". International Journal of Advanced Research, Ideas and Innovations in Technology, ISSN: 2454-132X.

23. Jagadeesh Bodapati, Oggi Sudahkar, Atava G V Karthik Raju, "An Improved Design of Low-Power High-Speed Accuracy Scalable Approximate Multiplier" Journal of VLSI Circuits and Systems, ISSN: 2582-1458 Vol. 4, No. 1, pp. 7-11, feb - 2022.

24. Bodapati, D. J., Raju, A. G. V. K., & Yenugu, D. (2019). Estimation of Energy in Complementary Metal Oxide Semiconductor using VLSI Design. International Journal of Innovative Technology and Exploring Engineering, 8(10), 4421–4423. https://doi.org/10.35940/ijitee.j9838.0881019.

25. In 2023, T. Vaani, V. Nagamani, N. Joshna, R. Rangaswamy, Y. Mallikarjuna Reddy, S. Girish Babu proposed "VLSI De- sign of a Novel Area Efficient FIR Filter Design using RoBA Multiplier.", Advanced Engineering Science, Vol. 55, Issue 02, 2096-3246.

26. Jagadeesh Bodapati, Dr. Vidushi Sharma, "A Novel Design of Low Power Explicit Pulse Triggered D-Flip Flop" Jour of Adv Research in Dynamical & Control Systems, Vol. 10, 06-Special Issue, pp no :486-491,may 2018.

27. Kurshid, Bimer, et al. "The Potential of Ultra-Wideband Printed Rectangular-Based Monopole Antennas." National Journal of Antennas and Propagation 5.2 (2023): 14-20

28. Dr Jagadeesh Bodapati, Oggi Sudhakar, Thoram Saran kumar, I Rama Satya nageswara Rao, "A Self-Controllable Voltage Level Technique for Low Leakage Power Sram Cell Using Finfet TechnologY" The International journal of analytical and experimental modal analysis, Volume XIII, Issue XI,pp no:1760-1766 , ISSN NO: 0886-9367,November/2021.

29. Jagadeesh Bodapati, Dr. Vidushi Sharma, "Estimation Of Low Power Consumption in Synchronous Digital System Using Cmos Digital Vlsi Circuits" International Journal of Advanced in Management, Technology and Engineering Sciences Volume 7 Issue 1, pp no:73-78,ISSN: 2249-7455, jan 2017.

30. Dr Jagadeesh Bodapati, Oggi Sudhakar, Atava G V Karthik Raju,"Design of Full Adder With Area Efficient Using Xor/Xnor Gates" International Journal of Management, Technology And Engineering Volume IX, Issue IV, pp no:552-560 ISSN NO : 2249-7455,APRIL/2019.