International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

FPGA Implementation of Adaptive Hold Logic Vedic Fused Dot Product Floating-Point Multiplier Using Razor Flip-Flop

G. Erna[1], V. Saidulu[2], G. Srihari[3], K. Babu Rao[4]

Submitted: 17/07/2023 **Revised**: 07/09/2023 **Accepted**: 26/09/2023

Abstract: In the recent technology of high-speed digital signal processing system has been increases with the explosive growth in mobile computing and portable multimedia application. In this digital signal processing method of arithmetic operation will have more number of floating point multipliers it will used in FFT and DCT based applications, but it will have more latency and less throughput and also take more area consumptions. Thus, this proposed work introduced to Reduced the latency in floating point multiplication with fused method, therefore here fused method integrated with Vedic multiplier using AHL and Razor flip flop. Here, this proposed work will Designed at 16-bit size in Verilog HDL, and it will Synthesized in Xilinx FPGA S6LX9-2TQG144, and finally compared all the parameters in terms of area, delay and power.

Keywords: *Vedic Fused Dot Product, NBTI and PBTI, Adaptive Hold Logic, Razor Flip Flop, FPGA, CSA*

1. Introduction

The growing need for mobile devices has resulted in the implementation of energy reduction strategies. The working of a VLSI device depends primarily on the method of digital multiplication by multipliers. Previously, the ageing-aware multiplier design was used. The multiplier's latency and velocity have both increased as a result of the PMOS transistor's low saturation thermal instability impact. For improved throughput and decreased performance loss, this led to the creation of an ageing-aware 8-bit Booth multiplier [1]. But in the Booth multiplier, the number of partial products is higher and the complexity of producing product bits is increased concurrently. As a result, this research suggests using a low-power Vedic multiplier and a razor flip-flop in oxide multiplier design with the novel Adaptive keep Logic (AHL) circuit. The Vedic multipliers can overcome problems of high-power dissipation and prevent failures. The Vedic multiplier also benefits from the extremely mild increase in area and door delays when compared with other multipliers as the number of bits is rising. That is why time, space, and power are efficient. Also, for the carry-save adder, this architecture can be implemented. The Vedic multiplier implementation will reduce the energy consumption on the AHL circuit to a degree and also reduces the timeframe. Finally, the experimental the 2×2 and 4×4 modules can be used for the Vedic multiplier. However, the ripple carrying adder can be used for efficient implementation as 16×16 and 16×16 multipliers. By removing the tri-state gates and using conditioned hardware applied without a buffer, the output of the conscious ageing multiplier can be further improved [2]. This decreases the area and power consumption of the proposed circuit. The Vedic fixed-point multipliers suffered from an unstable temperature so it will produce errors.The incorrect values are found by the use of razor approaches results confirm that the Vedic AHL multiplier can provide both less power and delay in conjunction with the fusion of the dot product and the Vedic multiplier. The main objective is the power reduction with a small area and overhead delay because the efficiency of digital filters depends on the area, power, and delay of multipliers. AHL can be used to minimize output degradation by the Vedic multiplier design [3].

2. Literature Survey

Hani H. Saleh and Earl E. Swartzlander, Jr. published an article on very large-scale integration (VLSI) in IEEE Transactions in 2008 at Intel Corporation. Some important details regarding the (ADP) were provided for the Fused Dot Product Floating point. The time necessary to execute single-precision floating-point multiplication and addition operations on two pairs of data by a floating-point fused dot-product unit is only 150% that of a traditional floating-point multiplication [4] to [5].

[1]*Associate Professor, Department of Electronics and Communication Engineering, PACE Institute of Technology & Sciences (UGC Autonomous) Ongole-523272 Andhra Pradesh, India, ernamist@gmail.com*
ORCID: 0000-0002-4427-9002
[2]*Senior Assistant Professor, Department of Electronics and Communication Engineering, Mahatma Gandhi Institute of Technology, Jawaharlal Nehru Technological University, Hyderabad, India, vsaidulu_ece@mgit.ac.in*
[3]*Associate Professor, School of Technology, The Apollo University, Chittoor. Email: srihari_g@apollouniversity.edu.in*
[4]*Professor, Department of Electronics and Communication Engineering, Usha Rama College of Engineering and Technology (UGC Autonomous), Telaprolu, ungutur mandal, krishan District, Andhra Pradesh, India, baburaokodavati@gmail.com*

The paper "Leading Zero Detection and Detect - A Comparison of Methods" IBM Server Development and 21BM Austin Research Laboratory Austin, Texas, USA, Martin S. Schmooklerl and Kevin J. Nowka In high-speed floating point computers, the design of the leading zero anticipators (La) or detector (LZD) is crucial to the normalization of results for addition and fused multiplication-addition. This essay establishes the analysis and offers some alternate configurations and applications from the content of existing knowledge. It demonstrates how design decisions are frequently influenced by the overall layout of the addition unit, how subtraction is handled when the exponents are the same, and how the probable 15% one-bit error has been identified and corrected.

Honey Durga Tiwari et al (2008) a new algorithm based on an ancient Indian Vedic mathematics formula for reduced-bit multiplication is proposed. The Vedic formulation was discussed in great depth, Urdhva Triyakbhyam, and Nikhil. Urdhva Triyakbhyam's general mathematical formulation is valid to all situations of multiplication. This Sutra has been used to create multiplier architecture, which is close to the conventional multiplier array in which many supplements are needed to reach the final product. Because of its composition, there is a high delay in carrying propagation when large numbers are multiplied. The Nikhil Sutra, which reduces the multiplication of two large numbers by two small numbers, was discovered to be the solution to this problem. The proposed algorithm derives its sense from this Sutra and is further configured with some general arithmetical operations, such as extension and bit moving, to reap the benefits of a bit decrease in multiplication. By minimizing specific multiplication by 4 *4 bits to 2 * 2 bits, the algorithm's computational efficiency was demonstrated. The results of the FPGA implementation show that the proposed design needs much less time and space than conventional stand and array-multiplier concepts, making it suitable for a variety of DSP applications.

Cong Liu et al. (2014) suggest a new projected multiplier for high-performance DSP applications that uses less power and has a shorter critical path than conventional multipliers. For quick partial product accumulation, this multiplier uses a newly created estimated adder that restricts carry propagation to the nearest neighbors. By employing variable numbers of most significant bits (MSBs) for error reduction, configurable error recovery can achieve various levels of accuracy. Most of the errors are minor because the estimated multiplier has a small mean error gap. A 16-bit anticipated multiplier built in a 28nm CMOS system reduces latency and power by 20% and as much as 69%, in contrast to the Wallace multiplier. The idea for an approximate multiplier offers processing precision comparable to that found in conventional accurate multipliers, but with considerable power and efficiency advantages, thanks to the extensive use of error recovery.

3. Existing Work

3.1 The Enhanced Architecture Of Floating-Point Fused Dot-Product Unit

A majority of work has reached into the blocked floating-point fused multiply-add (FMA) unit, adding of scaling factor and to avoid the overflow during adding of partial products. The architecture of floating-point units is very similar to that of discrete hovering adders and multipliers. A fused multiply-add unit can not only minimize the latency of a multiplication and addition process, but it can also prevent the floating-point adder resolution and improve the the floating-point multiplication. FMA units allow high-throughput scanning and FPGA-based implementations. FMA units are used in electronic signal processing and computer applications for partition , argument elimination, and that is why FMA has become an important unit in many consumer products, including IBM, HP, and Intel [6].

High-throughput scanning and FPGA-based implementations were possible with FMA units. FMA units are used in electronic signal processing, image processing and video games for partition and argument elimination, which is why FMA has become an important unit in many consumer products, including such IBM, HP and Intel. To get the dot-product with traditional floating-point adders and multipliers, there are multiple alternatives. In synchronization, two multipliers and an adder are used. Many DSP algorithms can benefit from the numerical operations performed by this unit. The FDP is especially helpful to the multiplication of complex operands. The FFT butterfly operation, the DCT butterfly operation, vector multiplication, and the wavelet transform, for example, could all benefit greatly from the speed increase given by this unit [7].

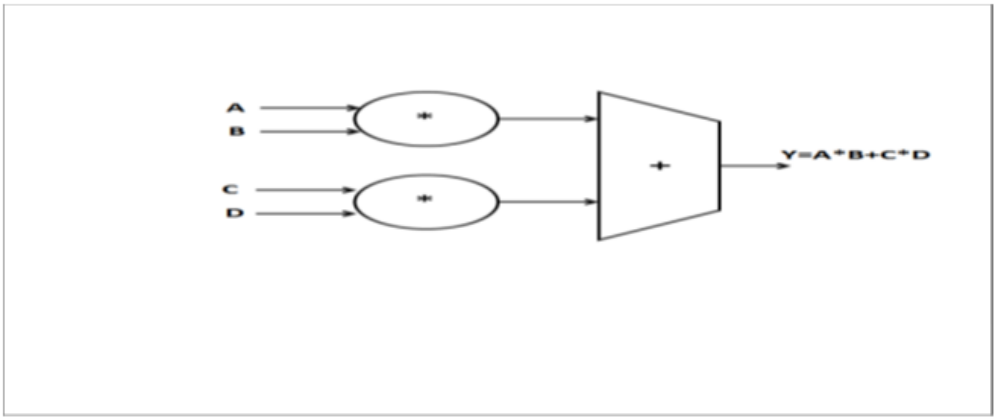


Fig 1. Traditional Parallel Dot Product

This is necessary, for example, when evaluating the FFT and DCT butterfly operations are performed in decimation-in- time (DIF) and decimation-in-frequency (DIF) and it improved the image accuracy that means avoid the noise in the more number of pixels in the 2-dimensional graphic with the helping of zero-padding techniques in butterfly operation in FFT technology. The dot product is calculated using two multiplications and addition in traditional floating-point hardware. Alternatively, two independent floating-point multipliers followed by a floating-point adder could be used to perform the multiplications in parallel, and this is more expensive in terms of transistor area and power consumption.

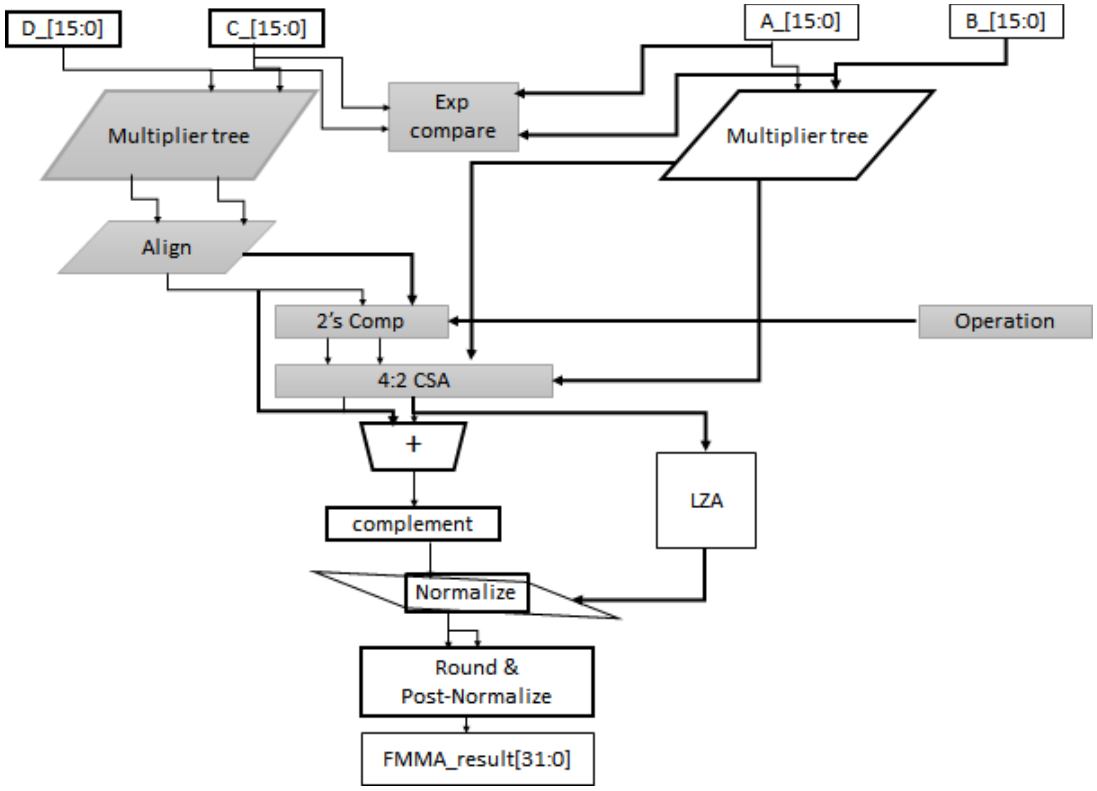


Fig 2. Floating Point fused Dot Product Unit

This device could be used by the system to replace the floating-point adder and multiplier, despite its unappealing appearance. To avoid multiplication trees, if both operands B and D are set to a module, the data is only supplemented and transferred by simple multiplexers for operands A and C the added frequency is more like a discrete floating-point adder with a multiplexer delay. The device can also be used to multiply C * D by adjusting A or B to zero and bypassing the orientation circuit with data forwarding

multiplexers. In this context, there will be an additional delay of two multiplexer actions [8]

3.2 A. Exponent Compare Circuit

The exponent comparisons circuit for the floating-point fused two-term dot-product unit bases itself on the exponent compare circuit for the FMA. An additional exponent adder (an 8-bit IEEE floating-point single-precision adder) is included in the fused DP exponent comparator circuit depicted in Figure 3 to combine the exponents of the C and D sources. Because the fused DP unit extra exponent adder and the FMA unit original exponent adder operate in tandem, the mixed DP unit exponential comparator circuit's delay is practically the same as the FMA exponent comparing circuit's delay.

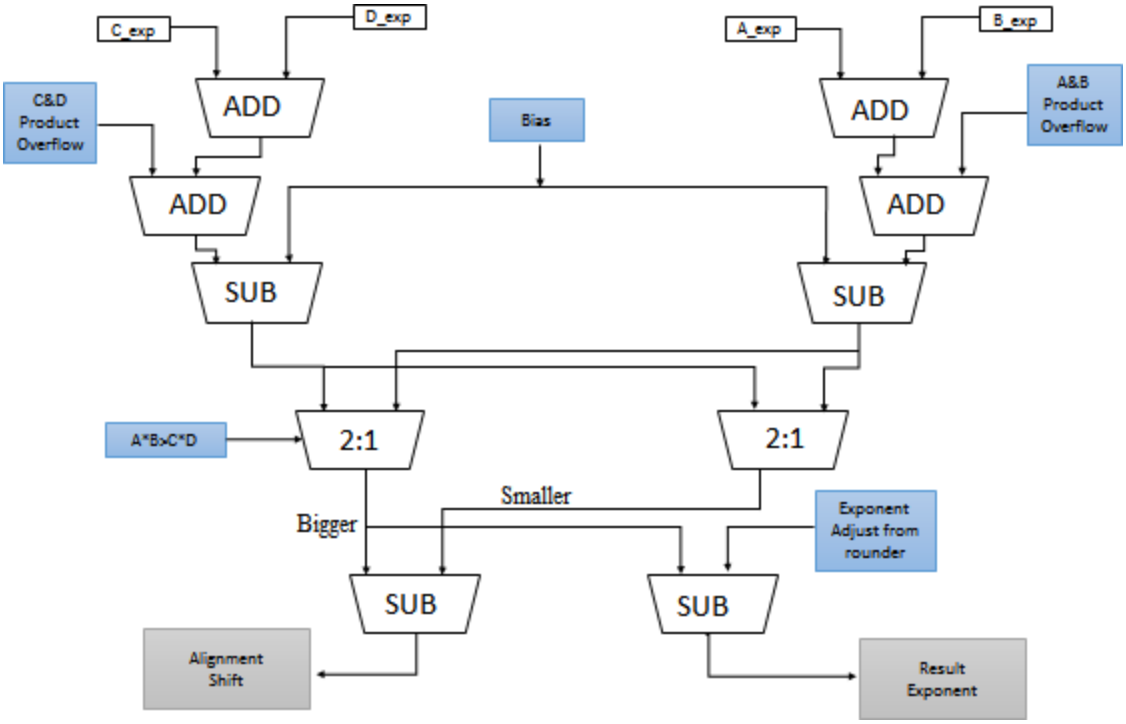


Fig 3: Exponent Compare Circuit

3.3 B Leading Zero Anticipated

This circuit left-shifts the sum such that the leading zero appears in the left-most digit to comply with the IEEE floating-number notation specification. It does this by counting the number of successful zeros in the significant adder result. Additionally, it produces round and sticky bits for the round device as well as exponent correction values.

The fundamental building block of a leading zero anticipator (LZA) circuit is a pre-encoder. The LZA circuit is required for result normalization in fused DP unit subtraction operations with large denial (the result contains numerous leading zeros). The standardization and rounding circuits of the fused DP units are equivalent to those of the FPA unit.

The alignment circuit of the floating-point fused two-term dot-product unit served as the foundation for the balancing circuit of the FMA unit. The fused DP alignment circuit consists of two sizable alignment shifter sub-blocks. This method provides the number and carries outputs of the "C*D" significant multiplication result. The number and carry outputs of the notable multiplications of "C*D" and "A*B" are compared by the respective magnitudes of "A*B" and "C*D," as well as the outputs of each multiplying [9].

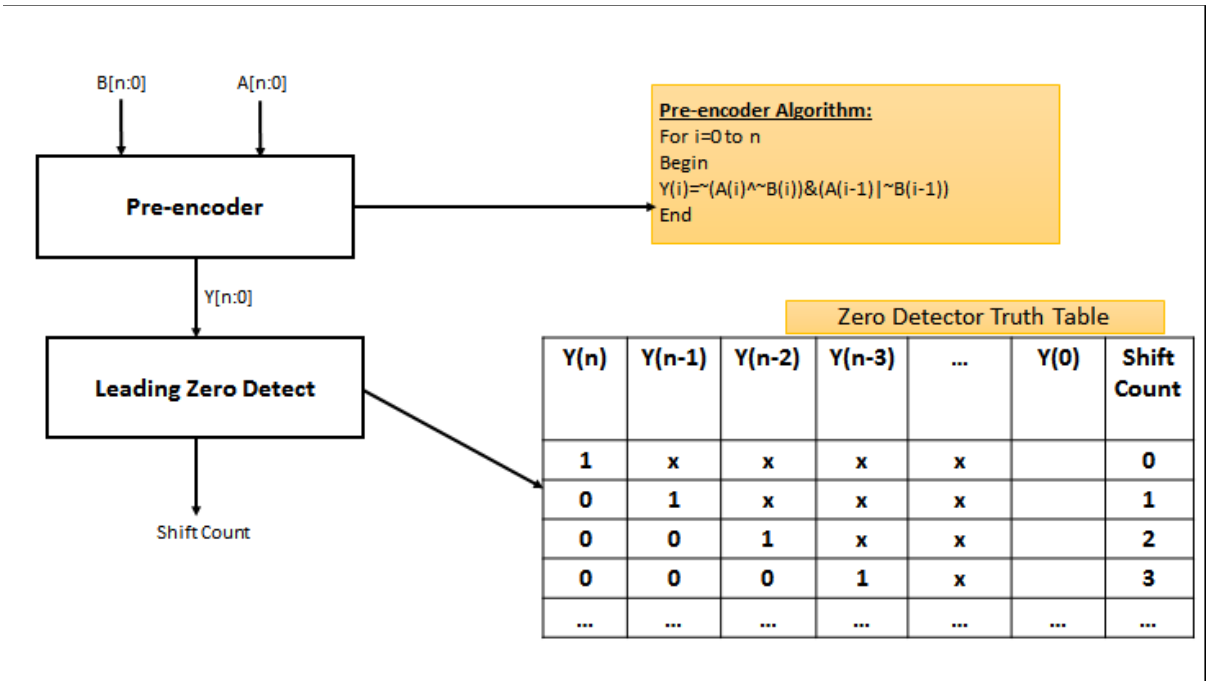


Fig 4. LZA circuit Concept

The "Alignment of a Fused Floating-Point Dot-Product Unit with two Terms to change the exponent, the exponential changes the logic in the circuit, which removes or subtracts the carry out from the main addition. Because the four major bits generate a carry out of up to three, two carry out bits are used for the change. The normalization change number is subtracted in the event of major cancellation. The collection bits and carry-outs from addition and subtraction are used to find exclusions".

3.4 Timing Violation With The Instability Of Temperature Or Variable Latency Delay

The circuit is divided into two components by varying architecture produce two paths with effect of suddenly increased temperature in the existing work. When the PMOS transistor is under negative bias (V gs =- V dd), negative temperature bias instability (NBTI) occurs. The interaction between the inversion layer holes and the hydrogen-passive Si atoms in this state breaks the Si-H bond formed during the oxidation process, permitting H or H2 molecules to form. The surface traps are cleared at the point where these molecules scatter. Increased limit voltage (Vth) and reduced number of switches speed of the circuit result from collected charge transfer between the silicon and the gate oxide interface.

1) Shorter Paths:

Shorter paths can be completed in a single cycle. The entire application of variable-latency designs is better than conventional designs when shorter paths are activated repeatedly. Several variable-latency adders, for example, were suggested using the prediction technique in conjunction with error detection and recovery. To increase the accuracy of the hold logic and maximize the efficiency of the variable-latency circuit, a short path activation function method has been developed. To coordinate operations on non-uniform latency too high. Longer paths

2) Longer paths

Longer paths, on the other hand, allow multiple cycles to complete. The longer path also changed dynamically to increase with the temperature as shown in the below figure [10].

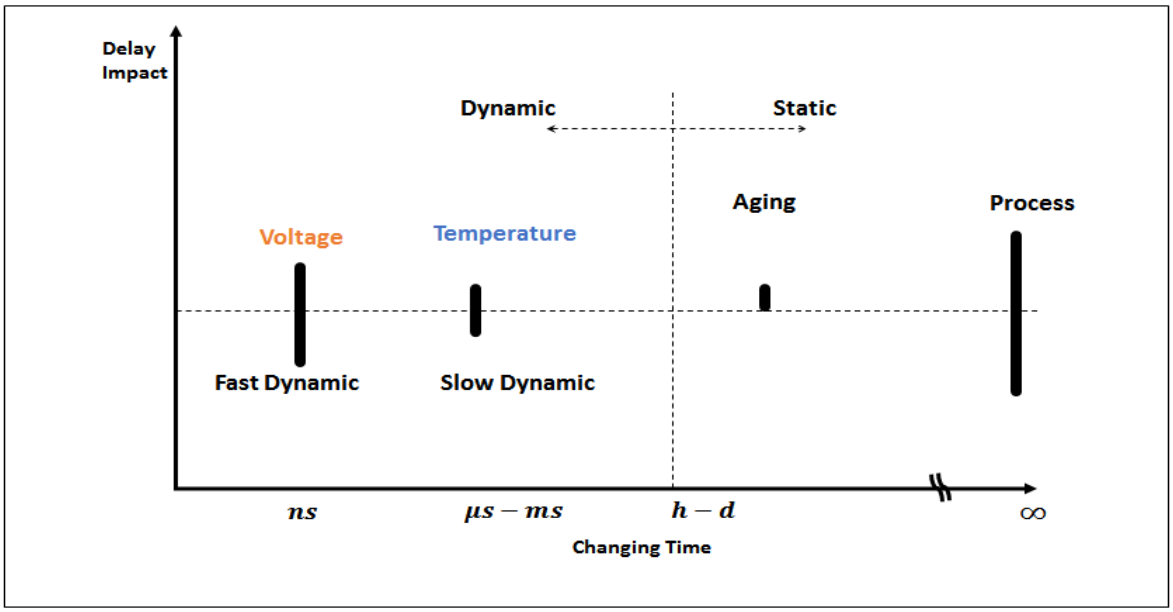


Fig 5. Timing Violation with multiple threshold Voltage along with Temperature effect

3.5 Disadvantages of Existing Work

In the treatment of floating-point computing data, there are two types of errors: propagation errors, which are mostly caused by mistakes in the input data and operation types, and reducing errors, which are brought on by the rounding of operation results.

- Less Security
- More Area and More Power

4. Proposed Methodology

4.1 Vedic Multiplier

It is constructed on Vedic multiplication formulas (sutras) and handles more straightforward operations rather than modest mathematical operations. Both a modest and dominant method is used to perform the basic arithmetic operation. It is a distinct computational methodology that is based on basic principles and rules. It is based on the 16 sutras, which were historically used for the multiplication of two numbers, and it can handle a wide range of mathematical challenges.

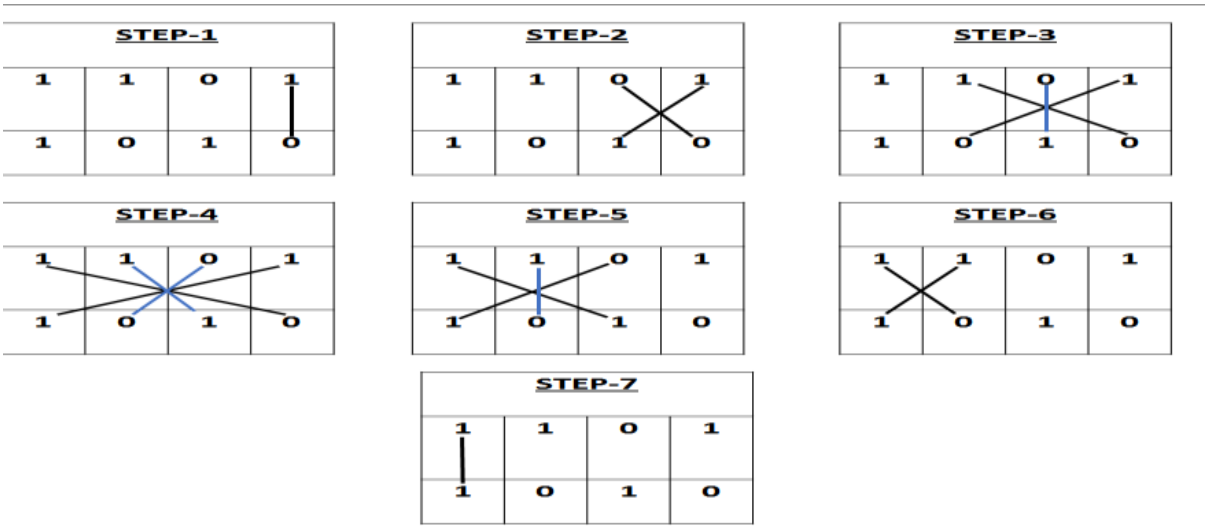


Fig 6. Line diagram for multiplication of two 4 - bit numbers

.Urdhva Tiryakbhyam sutras are being used in the Vedic multiplier in this study. Urdhva denotes down or vertically, and transverse is denoted by Thiryakbhyam. In these sutras, the biased products are defined in parallel and their summation occurs in the multiplication that is shown in Figure 4.3. At that point, the multiplier is self-regulating in the processor at the clock frequency. Besides, the microprocessor operations are normally carried out at a higher clock frequency, increasing the processing power.

The Vedic multiplier is normally implemented in the 2x2 module, so two 2x2 modules are needed to implement the 4x4 Vedic multiplier, while four 2x2 modules are required for 8x8. The implementation starts first with the 2x2 bit multiplier design. For multiplication, the "Urdhva

Tiryakbhyam Sutra" or "Vertically and Crosswise Algorithm" is used effectively to construct the digital multiplier architecture.

4.2 Computation Methods In Parallel In Vedic Multiplier

1. CP D0 = D0 * Eo= P
   E0
2. CP D1D0 = D1 * E0+D0 * E1 = Q
   E1 E0
3. CP D2 D1 D0= D2*E0+D0*E2+D1*E1= R
   E2 E1 E0
4. CP D3 D2 D1 D0=D3*E0+D0*E3+E2*E1+E1*E2= S
   E3 E2 E1 E0
5. CP D3 D2 D1 =D3*E1+D1*E3+D2*E2= T
   E3 E2 E1
6. CP D3 D2 =D3*E2+D2*E3 = Y
   D3 D2
7. CP D3 =D3*D3=G
   D3

In the example of a multiplier, the basic blocks are 2x2 bit multipliers, which are used to create a 4x4 block, which is then used to create a 16X16bit block, and finally a 16x16 bit multiplier. System family Spartan 3E, device xc3s500, kit fg320 with speed grade -4 has been chosen for synthesis. Let's start with the formation of a 2x2 bit multiplier.

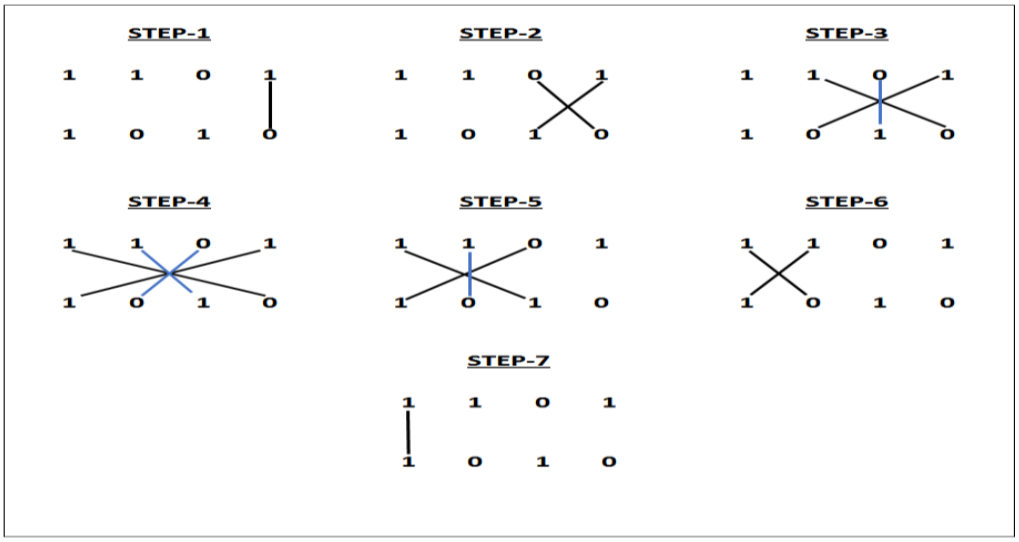


Fig 7. Flow Chart of Urdhya Tiryakbhyam Sutra

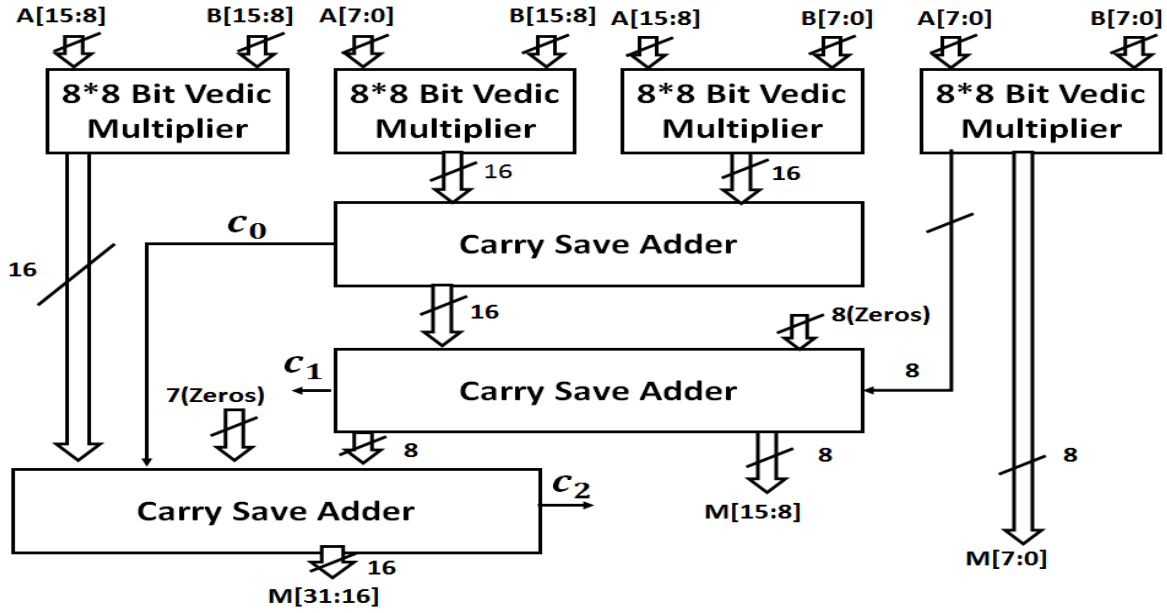


Fig 8. An Implementation of 16 x16 bit Vedic multiplier using carry save adder

Carry save adders are the best choice for this kind of addition because it occurs frequently that we need to combine several operands. A carry-saving adder performs numerous partial additions by building a ladder of stand-alone full adders. The main principle is that the carry is sent to the next column since it has a larger power of 2 than the other moves. For the creation of the 8x8 Vedic multiplier, the 4x4 Vedic multiplier is used as a basic building block diagram. By using the 8x8 Vedic multiplier as the fundamental building block, a further design of 16x16 is put into effect. The inputs for carrying skip adders of various sizes are multipliers. Comparing the BEC to the standard Vedic multiplier in Figure 7 will reduce the number of gates used. It includes four equal-sized groups, each of which consists of an 8*8 Vedic multiplier with inputs divided by the Urdhva-tiryagbhyam sutra. Results of Vedic Using the Carry save adder to do additions will save you time and logic. So in the fused dot product, Vedic multiplier the 16-bit adder easily skips the carry to maintain the high speed [11].

5. Design the Vedic 16-Bit Floating Point Fused Dot Product Multiplier Using Adaptive Hold Logic

Proposed to reduce traditional circuits' waste timing. The circuit is divided into two components by varying architecture produce two paths with effect of suddenly increased temperature in the existing work. With the explosive growth in mobile computing and portable multimedia applications, the recent technology of high-speed digital signal processing systems has been growing. This digital signal processing method of arithmetic operation will have more number floating-point multipliers it will use in FFT and DCT based applications, but it will have more latency and less throughput and also take more area consumptions [7] to [9]. Thus, this proposed work introduced to reduce the latency in floating-point multiplication with the fused method, therefore here fused method is integrated with Vedic multiplier using AHL and Razor flip flop. Here, this proposed work will be designed at 16-bit size in Vedic fused dot product[12].

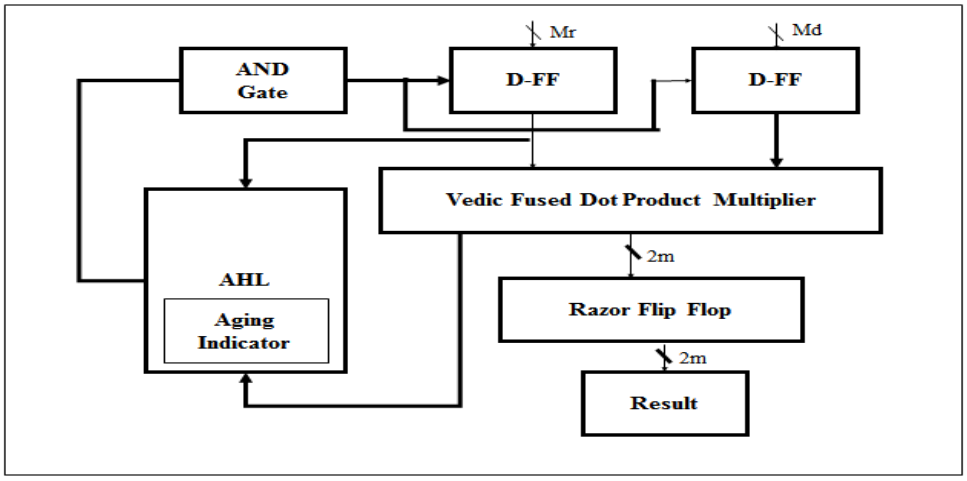


Fig 9. Aging-aware Vedic Floating-Point Fused Dot Product Multiplier

Figure 7 consists of sub-hardware components likely Adaptive hold Logic and Razor flip flop then it is possible to avoid aging problems in the conventional multiplier and detect the errors using Razor flip flops[13].

5.1 Adaptive Hold Logic

Fig 9 shows the architecture's overall process. A MUX, D flip-flop, and declining indicator make up the AHL.

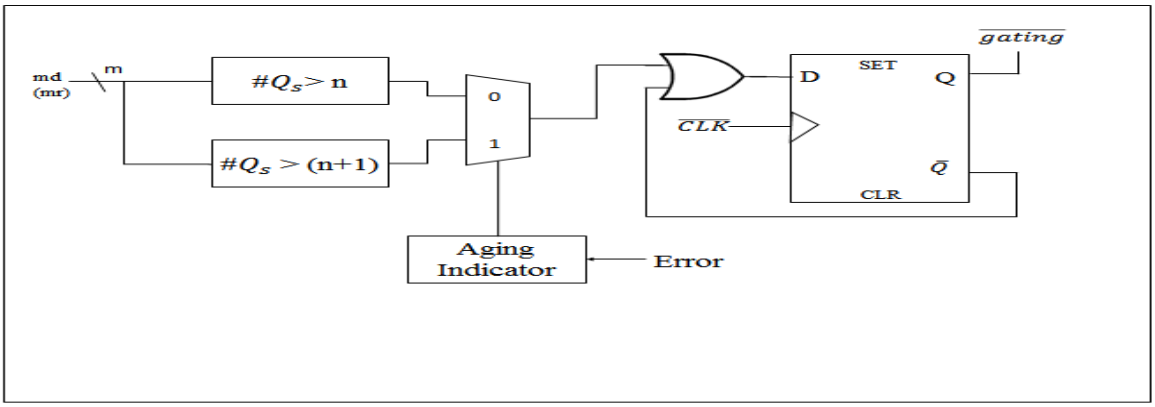


Fig 10. Diagram of AHL

The AHL circuit includes an aging indicator, two measurement blocks, a mux, and a D flip flop. The ageing indicator indicates whether the device has lost a significant amount of productivity as a result of aging. The aging indicator is set up as simple timers that indicate the number of faults over the subject of a few procedures before returning to zero at the end. The feature architecture design with Vedic fused floating dot product will be unable to perform these functions if the cycle time is too short, resulting in time abuses [14]. The AHL and multipliers both start working at the same time at first. The input bit magnitude from the multipliers is used in the decision block. The judgment block determines whether the operation can be finished in one or two iterations by calculating the number of zeros. The outcome is forwarded to the cutter flip flop by the multiplication. The path delay failure is detected by the razor flip flop, which confirms the error to AHL [14] to [15]. The very next input pattern will be gated by the AHL. After just a few cycles, the error will be checked. As a result, the correct product obtained. Usually, the AHL will guarantee that the or sometimes two-cycle operations are completed properly. These temporal violations are caught by the Cutter flip-flops, which generate error signals. When errors arise regularly and exceed that one threshold, the part of the circuit scheduling has greatly improved due to the ageing effect, and the ageing indicator is output 1. If the output is zero, the ageing impact is still significant, and that no corrective action is required [16].

If the number of the binary integers of zeros (multiplicator for Vedic Fused Dot multipliers) is greater than n (a positive number), outputs 1 is generated in the first computing framework on the AHL circuit, and if the multiplicand number of zeros is greater than n + 1, output 1 is generated in the next computation block on the AHL circuit. They both decide whether an input pattern requires one or several cycles, but only one cycle at a time is selected [17].

5.2 Razor Approach

The razor flip flop is the most useful in the digital system design because it finds the met stability of timing violation in the presented affected with the ageing factor in the Vedic fused floating-point dot product multiplier. When a problem happened, the Cutter flip-flop sets the error signal to 1 to tell the system it intends to re-run the action and to alarm the AHL circuit.

In this case, the Razor was used as a technological solution to a complex offence. The Razor Flip Flop is a flip flop that resembles a razor blade. A 1-bit Razor flip-flop consists of a key

flip-flop, a shadowy latch, a XOR and a multiplier. The main flip-flop captured the execution result of the combination circuit

using a normal clock signal, while the functioning result of the shadow latch is recorded using a postponed clock signal, which is less than the normal clock signal. The current proposal's delay is longer than the processing times if the shadow latch's lock bit differs from the main flip-flop's, and the primary flip-flop detects an incorrect conclusion [18] to [20].

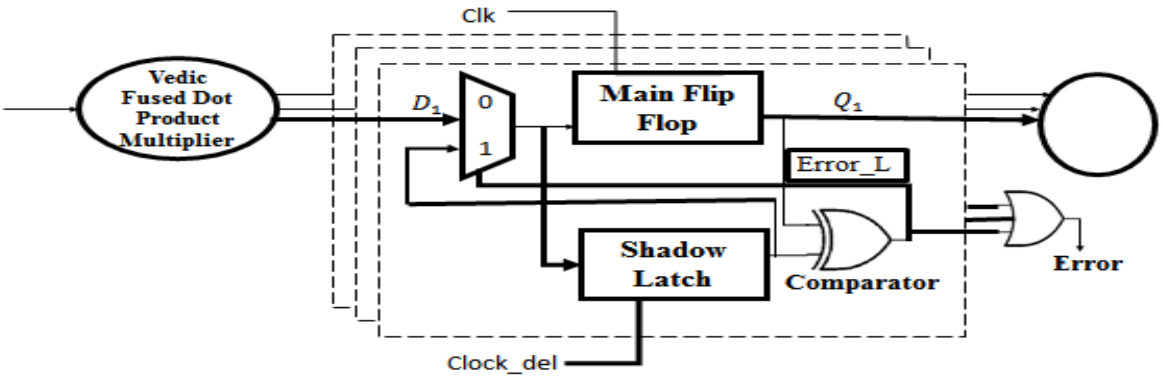


Fig 11. Block Diagram of the Razor Flip-Flop

The sharpener switch eventually decides whether a one-cycle trend will sequence. The timing violations are shown in the waveform of which was by given figure 10.

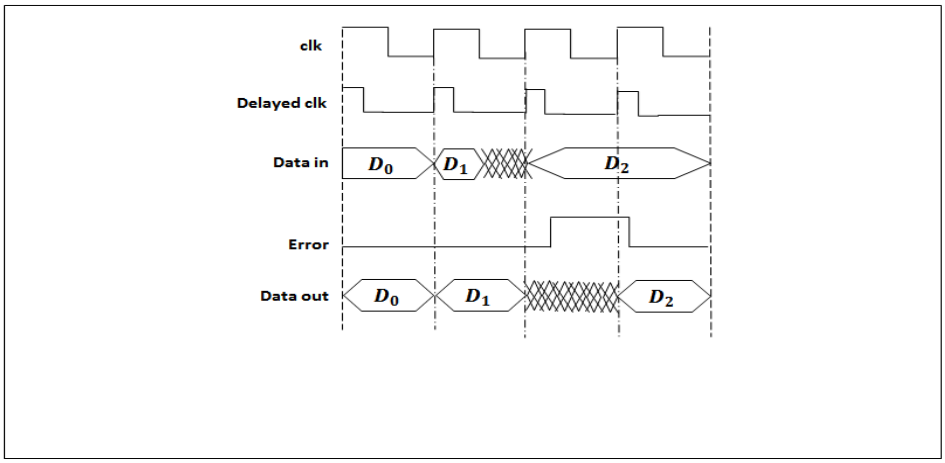


Fig 12. Waveform of the Razor Flip-Flop

6. Experimental Results and Discussions

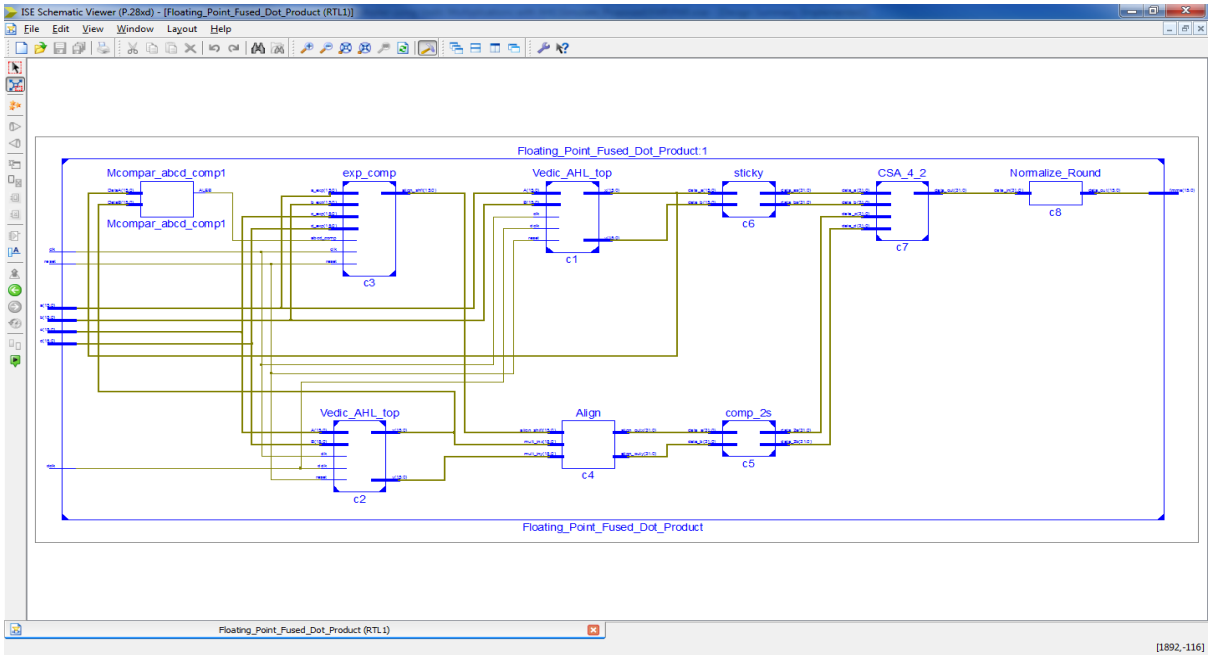
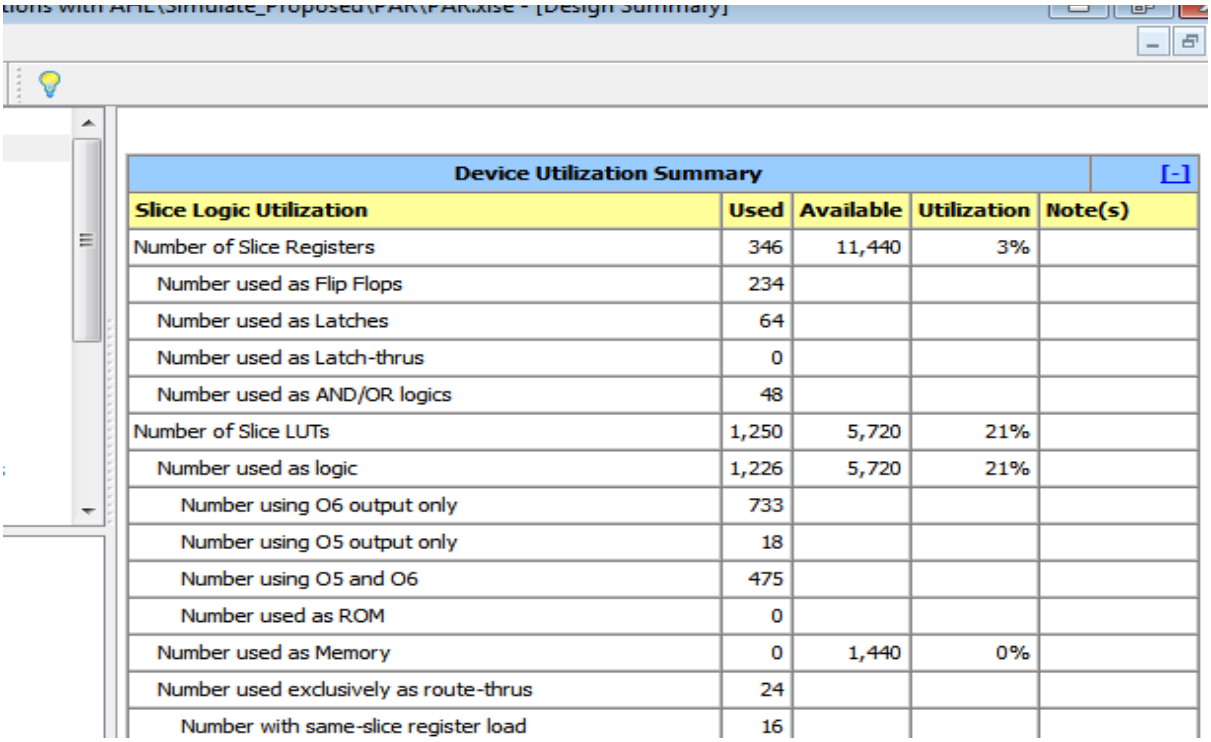


Fig 13. RTL Schematic of Vedic AHL using Razor Flip flop

The RTL schematic was implemented and synthesized using highly efficient adaptive Hold Logic-based floating point Fused Dot Product using Vedic Multiplier, which generates the Device utilization summary, Power report, and timing summary in nanoseconds with fewer logic blocks because of used new method to compare with Floating Point Fused Dot Product Multiplier. The Hardware design of the Fused Dot Product of floating Point Vedic Multiplier using adaptive Hold Logic integrated with Razor Flip Flop was performed on Vertex-5 Based on the FPGA Developed Board by XILINX to evaluate the area and speed. The proposed method improves the performance in the speed and minimizes the late.



Device Utilization Summary				
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	346	11,440	3%	
Number used as Flip Flops	234			
Number used as Latches	64			
Number used as Latch-thrus	0			
Number used as AND/OR logics	48			
Number of Slice LUTs	1,250	5,720	21%	
Number used as logic	1,226	5,720	21%	
Number using O6 output only	733			
Number using O5 output only	18			
Number using O5 and O6	475			
Number used as ROM	0			
Number used as Memory	0	1,440	0%	
Number used exclusively as route-thrus	24			
Number with same-slice register load	16			

Fig 14. Device Utilization of Vedic AHL using Razor Flip flop

As can be shown, the proposed Fused Vedic Multiplier has 346 slices with a total availability of 11,440. The total number of LUTs used is 1250, with 1226 logics and an availability value of 5,720. According to the utilization review, proposed Vedic AHL Multiplier registers have a utilization of 3%, and slices LUTs have a utilization of 21%. The proposed Fused Vedic Razor multiplier design with blocks can be seen in Figure 2.18. CSA adder, AHL, Razor Flip Flop, and, Multiplication of mantissa and addition of exponent with normalization and detect the errors in the parts of the proposed architecture.

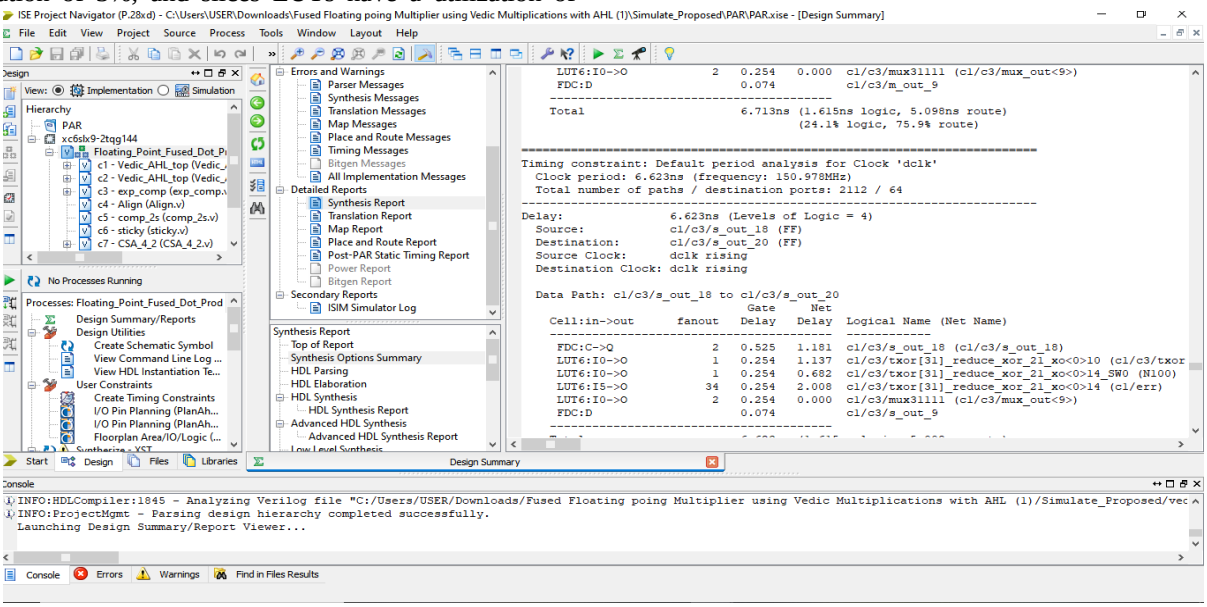


Fig 15. Delay Synthesis Report of Vedic AHL using Razor Multiplier

In the above figure of 13 shows the delay summary of the Vedic AHL multiplier. The delay consists of gate delay and net delay. The proposed system produced less delay compared to the existing method because AHL Vedic Multipliers utilized fewer hardware components. Finally, the total delay is 6.62ns including logic and routing hardware components.

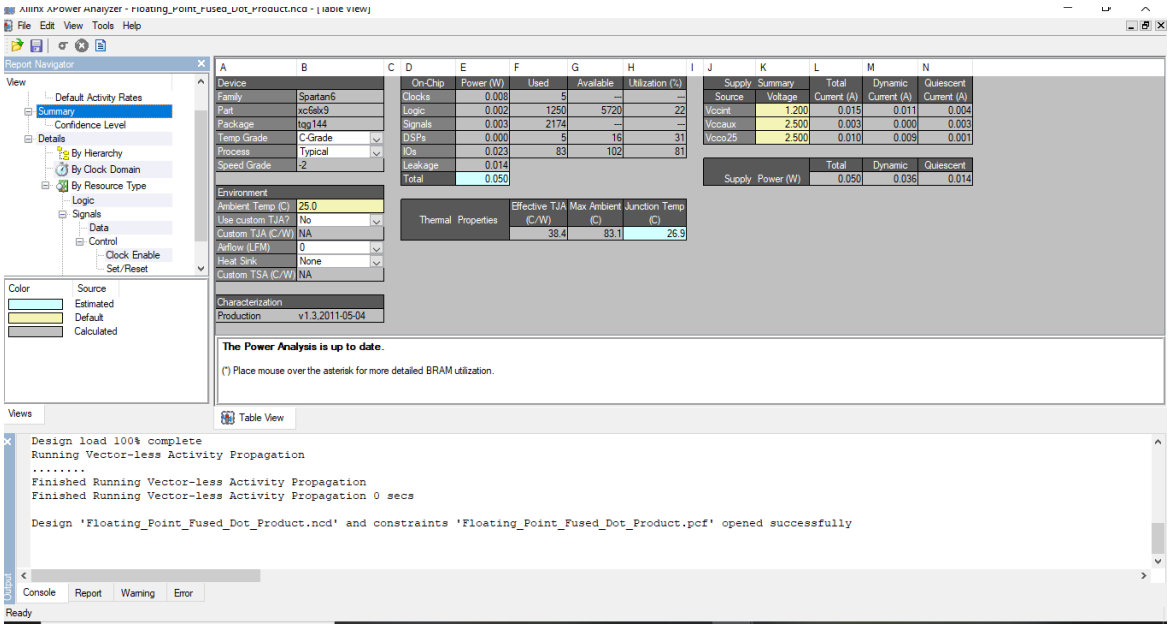


Fig 16. Power Report of Vedic AHL using Razor Multiplier

The Proposed Vedic Fused dot Product utilize less power because of implemented with vertical and crosswise algorithm and its combination of logic power of 0.000, signal power of 0.010, Ios power of 0.020, and dynamic power is 0.264 becomes a total power is 0.041.

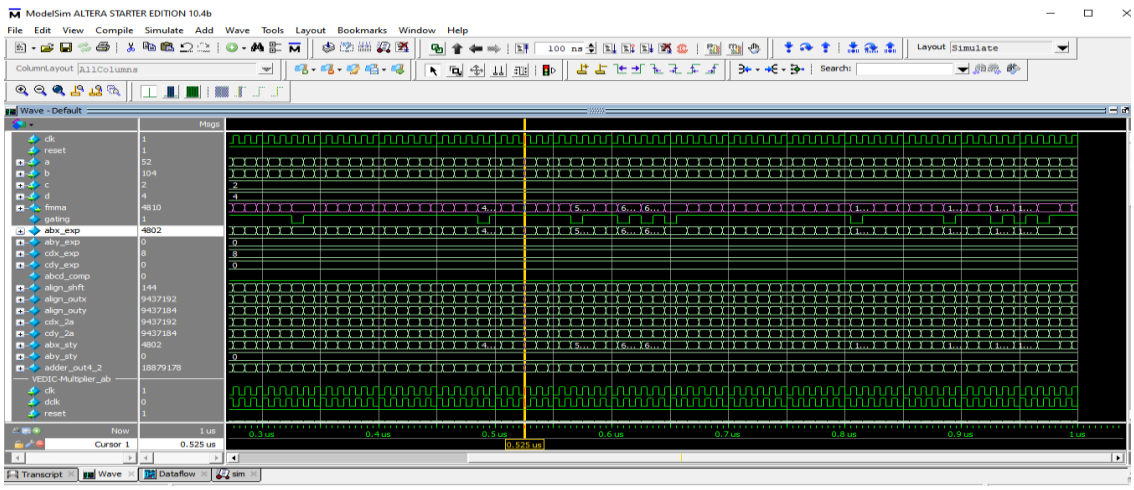


Fig 17. Simulation of Vedic AHL using Razor Multiplier

7. Performance Evaluation

The results of the proposed Vedic Fused Dot Product Multiplier with AHL are compared to those of the Fused Floating Point Dot Product Multiplier and the column multiplier without buffer in this portion. Table 1 summarizes the results of the comparison

Table 1. Outlines the results of the comparison

Parameters	Fused Dot Product (Existing)	Vedic Fused Dot Product (Proposed)	Reduction %
Number of slices	684	346	49
Number of LUTs	2448	1250	48
Number of occupied slices	846	224	73
IOB	83	80	3
Delay(ns)	35.587	6.623	81
Power consumption(mw)	0.050	0.014	72

In the above table, the existing multiplier is compared to the proposed multipliers. By comparing the number of flip flops, LUTs, and slices used in fused floating-point Vedic AHL multiplier has fewer flip flops, LUTs, and slices. It is widely used in unsigned multipliers because it reduces the number of partial products due to Vedic algorithms.

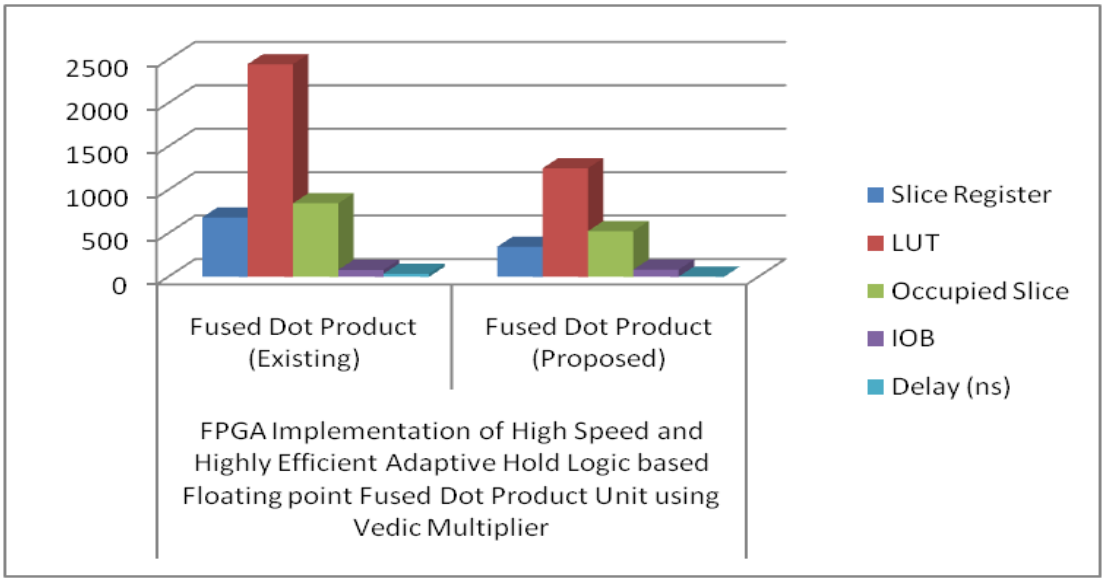


Fig 18. Performance Comparison between Fused Vedic AHL with Razor Multipliers and existing multipliers in 16× 16 multipliers

This is because, in a Fused Floating Point Dot Product Multiplier without a buffer, additional circuits for AHL and Razor flip-flops are needed to ensure the multiplier's correct operation after deterioration. The proposed Vedic Fused Dot Product Multiplier with AHL takes lessor hardware components than the Fused Floating Point Dot Product Multiplier without a buffer. This is because AHL and Razor flip-flops take up less space in proposed multipliers. In this proposal, the power utilization is 72% is a reduction in all logic elements, and at the same

maintained less delay because we proposed Urdhvatriyabham sutra with this effect to get the less critical path compared to conventional Fused dot Product. so possible minimized latency in the percentage is 81.

8. Conclusion

The Vedic fused dot product multiplier implemented Xilinx-14.2 and synthesized with Vertex-5 technology standard cell library. The simulation is used to calculate the elevated speed and power consumption. The outcomes of the fused Vedic dot product floating-point unit designs are compared in Table 1. The result estimates are dependent on the proportion of the standard design's performance. To minimize the change number, the improved floating-point fused dot product unit implemented a new orientation and switching scheme. To minimize the size of the large addition, early normalization is used. A Fused dot product Vedic multiplier using AHL and Razor flip flop is designed for the unsigned numbers. The performance metrics of the proposed fused dot product Vedic multiplier such as power is reduced by 72%, the area is reduced by 48% and latency is reduced by 81% when compared to the conventional method. Hardware complexity is minimized using the Vedic Fused dot product. The proposed results show that the fused dot product configuration of the Vedic multiplier decreases the delay compared to the floating dot multiplier of the fused dot product. Since the failure error that occurred due to metastability (time violation) is also minimized by the planned reliable Vedic multiplier using adaptive keep logic, it is preferred for the emerging VLSI architecture.

References

[1] McGeer, Patrick, Jagesh Sanghavi, Robert Brayton, and Alberto Sangiovanni Vincentelli. "ESPRESSO-SIGNATURE: A new exact minimizer for logic functions." In *Proceedings of the 30th international design automation conference*, pp. 618-624. 1993.

[2] Betz, Vaughn, Jonathan Rose, and Alexander Marquardt. *Architecture and CAD for deep-submicron FPGAs*. Vol. 497. Springer Science & Business Media, 2012.

[3] Keote, R. S., and P. T. Karule. "Performance Analysis of Fixed Width Multiplier using Baugh Wooley Algorithm." *International Organization of Scientific Research Journal of Very Large-Scale Integration and Signal Processing* 8, no. 3 (2018): 31-38.

[4] Asadee, P. "A High-Speed Multiplication Algorithm Using Modified Partial Product Reduction Tree." *International Journal of Electrical and Computer Engineering* 4, no. 3 (2010): 518-524.

[5] Keote, R. S., and P. T. Karule. "Performance Analysis of Fixed Width Multiplier using Baugh Wooley Algorithm." *International Organization of Scientific Research Journal of Very Large-Scale Integration and Signal Processing* 8, no. 3 (2018): 31-38.

[6] de Angel, Edwin. "Low power digital multipliers." In *Application Specific Processors*, pp. 91-120. Boston, MA: Springer US, 1997.

[7] Sheplie, M. "High performance array multiplier." *IEEE transactions on very large scale integration systems* 12, no. 3 (2004): 320-325.

[8] Tiri, Kris, and Ingrid Verbauwhede. "A digital design flow for secure integrated circuits." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25, no. 7 (2006): 1197-1208.

[9] Bae, Kiseok, Sangjae Moon, and Jaecheol Ha. "Instruction fault attack on the miller algorithm in a pairing-based cryptosystem." In *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp. 167-174. IEEE, 2013.

[10] Iwamura, Jun, Shinji Taguchi, Suganuma Kazuo, Kimura Minoru, Tango Hiroyuki, Ichinose Kazuaki, and Sato Tai. "A high speed and low power CMOS/SOS multiplier-accumulator." *Microelectronics Journal* 14, no. 6 (1983): 49-57.

[11] Singh, Raminder Preet Pal, Parveen Kumar, and Balwinder Singh. "Performance analysis of 32-bit array multiplier with a carry save adder and with a carry-look-ahead adder." *International Journal of Recent Trends in Engineering* 2, no. 6 (2009): 83.

[12] Saligram, Rakshith, and T. R. Rakshith. "Optimized reversible vedic multipliers for high speed low power operations." In *2013 IEEE Conference on Information & Communication Technologies*, pp. 809-814. IEEE, 2013.

[13] Kim, Hyung-Ock, and Youngsoo Shin. "Semicustom design methodology of power gated circuits for low leakage applications." *IEEE Transactions on Circuits and Systems II: Express Briefs* 54, no. 6 (2007): 512-516.

[14] Cao, Y. "Predictive Technology Model (PTM) and NBTI Model." (2013).

[15] Yang, Hao-I., Shyh-Chyi Yang, Wei Hwang, and Ching-Te Chuang. "Impacts of NBTI/PBTI on timing control circuits and degradation tolerant design in nanoscale CMOS SRAM." *IEEE Transactions on Circuits and Systems I: Regular Papers* 58, no. 6 (2011): 1239-1251.

[16] Abrishami, Hamed, Safar Hatami, Behnam Amelifard, and Massoud Pedram. "NBTI-aware flip-

flop characterization and design." In *Proceedings of the 18th ACM Great Lakes symposium on VLSI*, pp. 29-34. 2008.

[17] Calimera, Andrea, Enrico Macii, and Massimo Poncino. "Design techniques for NBTI-tolerant power-gating architectures." *IEEE Transactions on Circuits and Systems II: Express Briefs* 59, no. 4 (2012): 249-253.

[18] Du, Kai, Peter Varman, and Kartik Mohanram. "High performance reliable variable latency carry select addition." In *2012 design, automation & test in Europe conference & exhibition (DATE)*, pp. 1257-1262. IEEE, 2012.

[19] Yang, Hao-I., Shyh-Chyi Yang, Wei Hwang, and Ching-Te Chuang. "Impacts of NBTI/PBTI on timing control circuits and degradation tolerant design in nanoscale CMOS SRAM." *IEEE Transactions on Circuits and Systems I: Regular Papers* 58, no. 6 (2011): 1239-1251.

[20] Chen, Yiran, Hai Li, Cheng-Kok Koh, Guangyu Sun, Jing Li, Yuan Xie, and Kaushik Roy. "Variable-latency adder (VL-adder) designs for low power and NBTI tolerance." *IEEE transactions on very large scale integration (VLSI) systems* 18, no. 11 (2009): 1621-1624.

[21] Su, Yu-Shih, Da-Chung Wang, Shih-Chieh Chang, and Malgorzata Marek-Sadowska. "Performance optimization using variable-latency design style." *IEEE transactions on very large scale integration (VLSI) systems* 19, no. 10 (2010): 1874-1883.

[22] Latha, S. ., Gundavarapu, M. R. ., Kumar, N. P. ., Parameswari, D. V. L. ., & Reddy, B. R. K. . (2023). Technology for Kisan Samanvayam: Nutrition Intelligibility of Groundnut Plant using IoT-ML Framework. International Journal on Recent and Innovation Trends in Computing and Communication, 11(3), 273–282. https://doi.org/10.17762/ijritcc.v11i3.6345

[23] Auma, G., Goldberg, R., Oliveira, A., Seo-joon, C., & Nakamura, E. Enhancing Sentiment Analysis Using Transfer Learning Techniques. Kuwait Journal of Machine Learning, 1(3). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/129