International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 3, May-Jun-2026, ISSN (Online): 2395-566X

Efficient MAC Architecture Using Different Parallel Adders

Mr.A.Prasad[1], Mrs.N.Swarupa Rani[2], Batchu Bala Bhargavi[3], Chinni Yamini[4], Chityala Lakshmi Devi[5], Chilakala Anjali[6]

[1]Assosiate Professor, Department of ECE, KITS-Markapur
[2]Assistant Professor, Department of ECE,KITS-Markapur
[3,4,5,6] Students, Department of ECE,KITS-Markapur

Abstract- Because of its capability to do arithmetic operations at fast speeds, the Multiply-Accumulate (MAC) Unit is an essential part of all digital signal processor applications. An 8-bit MAC Unit that can do addition and multiplication is the target of this study. While the MAC Unit uses the same multiplier, it incorporates other adders, including the Kogge-Stone, Ladner-Fischer, Carry Look-Ahead, and Ripple Carry adders. Xilinx ISE was used to implement the structures that were created in Verilog Hardware Description Language (HDL), and ModelSim was used for simulation.

Keywords- MAC Unit, Kogge-Stone, Ladner-Fischer, Verilog HDL, Multiplier.

I. INTRODUCTION

The process of designing intricate integrated circuits (IC) is involved in very large-scale integrated circuits (VLSI). Because integrated circuit technology made it possible to fit millions, if not thousands, of transistors onto a single chip, it radically altered chip architecture. The significance of using HDL in the design of more sophisticated digital systems has grown. [1]. Power dissipation is one of the important design elements in very large scale integration (VLSI). To keep up with Moore's law and provide lighter, more reliable consumer electronics, very large scale integration (VLSI) designs that use less power are essential. An increasing number of consumer, intermedial, and mechanical devices rely on digital signal processing (DSP) to process signals utilizing various components such as chips, Field Programmable Gate Arrays (FPGAs), and Custom ICs [2,3]. Digital signal processing systems cannot function without Multiplier Accumulator (MAC) units that are fast, have high throughput, and operate very well. Designers may improve power consumption by embedding MAC units into VLSI processors. This is especially important for battery-operated devices [4]. The MAC Unit follows the "multiply and then accumulate" procedure. Multipliers, adders, and registers form its backbone, as the name implies.

Video coding, digital filtering, and voice processing are a few of the many uses of digital signal processing systems that need MAC operations. Using fast and regular sum propagate adders, this paper investigates the MAC unit's design. Asymptotically equivalent sum-propagate adders may be developed with the introduction of the sum-propagation addition formulation. There are, nevertheless, observable performance gaps in latency due to differences in their underlying architecture. This research makes use of the Kogge-Stone and Ladner-Fischer parallel adders. Because they rely on a sequence of AND-XOR operations to function, these adders aren't always the fastest. This is why these carry-propagate adders also use AND-OR operations.

These adders use prefix calculation in parallel. The performance difference between different adder designs is successfully minimized throughout their operation [5]. Future technological implementations or FPGA platforms can benefit from this parallel-prefix computation when designing adders, especially when developing addition-related operations like in-memory or approximate adders, and when optimizing AND-XOR in logic synthesis flows that deal with addition [5]. The article is organized into five sections; the first portion provides an overview of the MAC unit's functions and uses. In Section 2, we provide a literature review. The approach is detailed in Section 3. Section 4 presents the findings and discussions that were acquired and simulated using the XILINX (ISE) tool, and Section 5 confers the conclusion of this study endeavor.

II. LITERATURE REVIEW

In [1], we see a suggested MAC unit design that minimizes hardware requirements while achieving maximum speed via the use of the Spurious Power Suppression Technique (SPST), which leverages the Booth multiplier. For the purpose of verification, this design utilizes the ModelSim and synthesizes it using the Xilinx-ISE tool. With regard to basic ideas, another MAC Unit was proposed by [6]. Designed for use in high-speed digital signal processing applications, this work's MAC Unit is Xilinx ISE 6.2i generated and implemented using the xc3s1000-5fg456 FPGA Xilinx device. The code for the unit is written in Verilog. The work was completed using minimum period values of 4.5 ns and maximum frequency values in MHz. According to a research by [7], the fastest multiplication method is the Kasturba method. A MAC Unit with high-speed capabilities was created using this method. To further facilitate command of the circuits, the design included reset and clock capabilities.

The design was synthesized using Xilinx ISE - 9.2i and the ISE Simulator. [8] A 32x32 bit signed/unsigned MAC Unit, modeled in VHDL, was created and implemented in this study. A modified version of the Booth method that incorporates time division multiplexing (TDM) and sign correction circuitry yields a multiplier with a latency that is similar to six XOR gates, and it reaches high speed. Due to its well-organized data paths, it ran efficiently and quickly when implemented in silicon. According to [9], there is another approach. The 32-bit MAC Unit was investigated in this study. It was modelled in Verilog HDL and built using the Carry Select Adder and Vedic Multiplier. The results were compared to a MAC unit that was built using the Ripple Carry Adder. The former was found to have less delay, significantly less power consumption, and faster speeds. The Cadence Genus tool was used to do the synthesis of this study. Based on virtuosity, the authors of [10] were able to develop and construct a pipelined MAC Unit for use in high-speed DSP applications using 180nm technology.

In order to minimize power consumption and processing time, this research examined several MAC designs to identify the optimal architectures for the specified performance. Spice code, created using HSPICE, was used to test the functioning of the complete framework. In this code, all the blocks in the circuits were specified as subcircuits. In [11], an alternative plan was detailed.

The purpose of this study was to create a MAC Unit using 45nm CMOS technology that has better latency performance, less size, and more power economy. The creation of the hybrid adder cell made all of this possible. This study's findings demonstrate that a MAC unit using a hybrid-logic adder achieves unprecedented levels of operating speed, power efficiency, and transistor count reduction. The traditional CMOS logic architecture resulted in a 68% reduction in area usage and a 92% improvement in power consumption. The paper [12] describes the design of a 64-bit MAC Unit that was simulated and synthesized using Verilog-HDL, ModelSim6.4b, and Xilinx-ISE 10.1e. The design included the Wallace multiplier and the Carry Save Adder.

The authors claim that this MAC Unit can be used in systems with high-performance processors, requiring a large number of bits for operations, due to its reduced delay, 217MHz frequency, and 177.732mW total power dissipation. in [13] The authors of this study used Verilog HDL to create a 32-bit MAC Unit with a Vedic multiplier and reversible logic gates; they simulated it using ModelSim and Xilinx 14.2; they synthesized it using Cadence and Xilinx. The investigation on this Vedic Multiplier found that it excels in many important areas, including speed, delay, area, and complexity, and it made use of the Urdhava Triyagbhayam sutra. These logics were successfully integrated into this work from a variety of mathematical, scientific, and engineering domains. In [14], an alternative method was laid forth. A Virtex-6 family field-programmable gate array (FPGA), an Array Multiplier, a Ripple Carry Array Multiplier, a Wallace tree multiplier, and a DADDA multiplier are all part of the MAC Unit that this study proposes using, which were all synthesized in Xilinx ISE 13.2. Superior area and latency performance were attained by this design in comparison to traditional MAC Units. With a larger amount of bits, this study may be further studied.

III. METHODOLOGY

There is a great deal of variety in the designs of MAC units. The MAC unit consists of two sub-modules: the adder and the multiplier. The MAC Unit's latency may be evaluated using these submodules since using various adders with the same multiplier has a significant impact on the computer's performance and efficiency. A rapid multiplier and an accumulator, which is the sum of the products that came before it, are the fundamental components of a typical MAC unit. A specific point in the memory stack is used to retrieve the two

inputs that are supplied to the MAC Unit. The first block of the MAC Unit, the multiplier, receives the inputs after fetching them and conducts the standard multiplication. The product is then passed on to the adder, which will accumulate the result. It is moved to a different part of memory. There is only one clock cycle required to complete the whole operation [15]. Verilog HDL is used to code the MAC Unit. To create and construct the MAC Unit, we used the Xilinx ISE Design tool (Version 7.1i). The multiplier is initialized first in the Verilog code for the MAC Unit. Then come the adders: Kogge-stone Ladner Fischer, RCA, and CLA. After that come the registers, which contain the D flip-flops. The instantiation technique is used to finish the MAC Unit and receive the final output. In Fig. 1, we can see the MAC Unit's design. The multiplier and multiplicand are the two components that make up the multiplier.

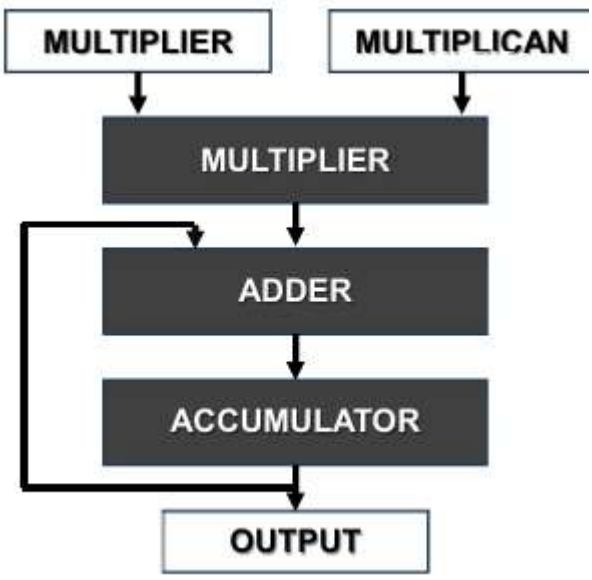
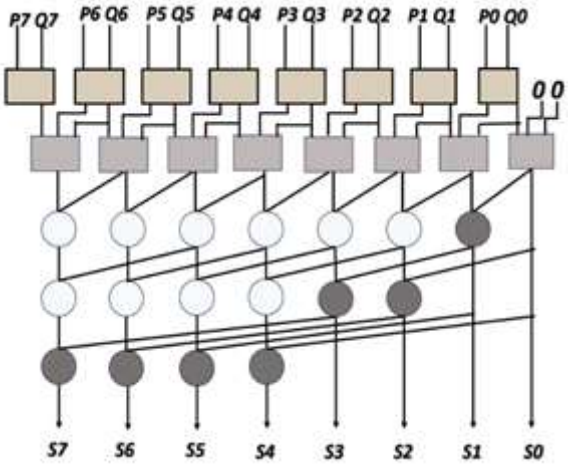
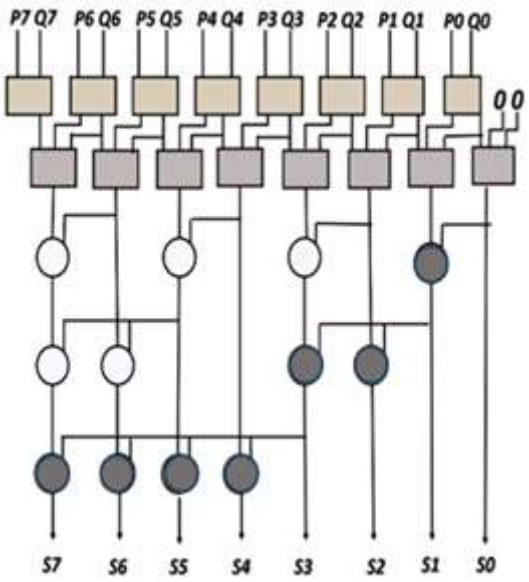


Fig. 1 MAC Unit Layout B. Adders

Propagate Adders that Add Prefixes in Parallel The main goal of this study is to change the carry-propagate operation of binary addition to a sum-propagate operation. This method eliminates the need for an intermediary carry by making each sum bit Si directly reliant on the sum bit Si-1 that came before it [5]. To do this, the AND operation must come before the XOR operation; for example, p + qr should be written as p+(qr)).



(a) Kogge – Stone



Ladner – Fischer

Figure 2: Prefix Adders in Parallel [3] These adders' adding process is free of carry. The addition of bits at one location in a carry-free addition does not influence or propagate a carry to the subsequent bit location. To illustrate, while adding two 1s in binary, a carry is produced (1 + 1= 10 in binary), however these processes make sure that carries like that don't impact the entire addition. The addition accurately depicts the non-redundant results without additional bits or excessive complexity, even if it is carry-free. One way to construct a binary adder is using the ripple-carry adder, which is the simplest possible arrangement. Because the carry bits are calculated sequentially in classic carry-propagate adders, their

performance may be subpar. Parallel prefix carry computation, on the other hand, provides a more organized and consistent method. A huge performance boost is achieved by computing the carry in parallel.

We can easily develop sum propagate adders by taking use of the features of parallel prefix processing. In order to reduce the delay to O (log2 n), where 'n' is the bit count, the parallel prefix circuits use a tree-like topology. The topologies proposed by Ladner-Fischer and Kogge-Stone form the basis of multiple parallel prefix sum propagate adders. Kogge-stone is useful for high-performance arithmetic circuits as it produces the carry in O(logn), making it the quickest addition algorithm in terms of design time. Because of its uniform structure, the Kogge stone can easily adapt to new technologies. Its little fan-out and logic depth is an additional benefit [16].

The Ladner-Fischer is easily able to speed up binary addition due to its flexible structure, making it a common choice for high performance arithmetic operations. This adder reduces the operation's footprint while simultaneously increasing speed. In VLSI Design and DSP, the Kogge Stone and Ladner Fischer adders are used because to their rapid and precise performance. A Kogge-Stone or Ladner-Fischer adder typically has three steps for calculation: (i)pre-processing, (ii)prefix, and (iii)final computation [17]. Verilog HDL was used to simulate the Kogge-Stone and Ladner Fischer adders. Codes for the various models were generated according to the logic gates that made up their structures. Afterwards, the constituent modules were instantiated according to their different designs, as illustrated in Figure 2(a) and 2(b), and the whole layout of both adders was programmed.

• A long-chain adder known as the Ripple-Carry Adder (RCA) may be constructed from a network of N full-adders to create an N-bit RCA. Using the carry-out from each previous full-adder as the carry-in to the next stage, this adder develops a bigger system by reusing the full-adder modules numerous times. Carriers, contrary to schematic convention, are assumed to move from least significant to most significant columns, usually in a right-to-left fashion. At the very right full-adder, the carry-in is zeroed out. O(N) is the carry-propagation time that determines how fast the RCA functions.

Despite appearances, this adder is really bit-parallel; nonetheless, the calculation of the output bits is sequential. • Keep CLA on hand to use as a look-ahead More complicated systems employ CLA since the RCA's propagation latency is a big drawback. The goal of implementing CLA is to reduce carry logic across a set of bits in order to achieve two-level logic. To implement carry look-ahead, we need to define two variables: "carry generate," sometimes known as Gi, and "carry propagate," often called Pi. These two variables, Gi and Pi, are used to get the adder's total and carry. One major benefit of CLA is the rapid addition logic it gives. Consider an n-bit CLA; to evaluate its carry bits, you'll need [n(n+1)]/2 AND gates and n OR gates.

Multiplier: The multiplier works by using the concept of traditional multiplication. There are two 4-bit inputs and one 8-bit output with this. To get the output, we multiply the two inputs. The MAC Unit IV's adder takes the final output as one of its inputs.

IV. RESULT AND DISCUSSIONS

The Fig.3 shows the MAC Unit configuration that was produced from the simulation.

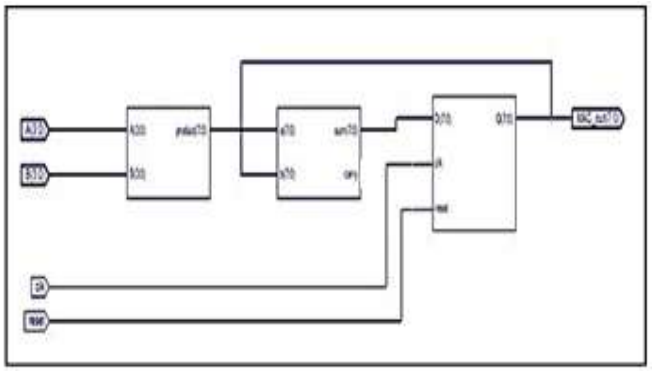


Fig. 2 Schematic of the MAC Unit

A multiplier, register, and adder are all part of the MAC Unit's internal processing in this architecture. Each unit also has many inputs, such as reset, clk, A, and B. Below in Fig. 4 is the MAC Unit with Kogge adder's output based on simulation. To aid with simulating the Verilog code, ModelSIM - 6.6d Simulator is used. At startup, with the rising edge of the clock and the reset set to high, the intended output is generated. The first value delivered to input A is 1111, while the second value provided to input B is 1000. With the reset turned all the way to high, the accumulator read nothing but zeros. Following an initialization of the reset to low (zero), the value at point B was changed from 1000 to 1111 in the following set of inputs.

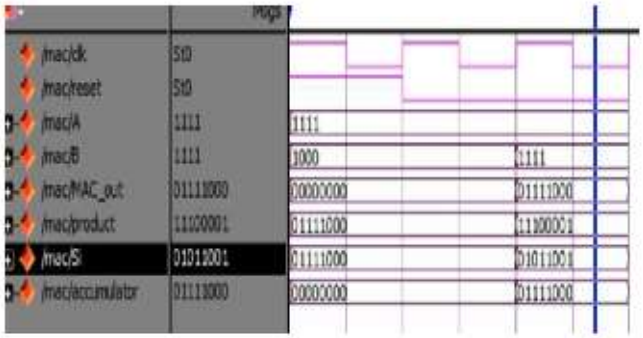


Fig. 3 Simulation-based output of MAC Unit

utilizing Kogge-Stone The sum Si (01011001) was finally calculated by adding the previously stored value (01111000) to the product (11100001) and then adding it to the sum (01011001). After that, the MAC Unit's output, MAC_out, is set to this total. As mentioned before, the output of the MAC Unit was obtained in a similar fashion for the other sets of values, which included of different adders.

TABLE I DESIGN UTILIZATION SUMMARY OF MAC UNIT WITH KOGGE-STONE

Logic Utilization	Used
Number Of Slices	31
Number Of 4-Input LUTs	57
Number Of Bonded IOBs	18

The Xilinx tool also yielded the MAC Unit's design description. The xc3s400 Field-Programmable Gate Array (FPGA) from Xilinx's Spartan-3 series is the intended device for this design. Slice count, 4-input LUT count, and Bonded IOB count were all part of the device usage summary that was included in the design summary. Table I displays the estimated values.

Fig. 4 Simulation-based output of MAC Unit

in terms of Ladner-Fischer Figure 5 below shows the MAC Unit's output from the Ladner Fischer adder simulation. Time, static, place and route, post-place and route, and synthesis reports are among them. Through the synthesis report, we were able to ascertain the overall MAC Unit delay. Kogge-Stone and Ladner-Fischer both achieved delays of 6.46 ns and 6.39 ns, respectively.

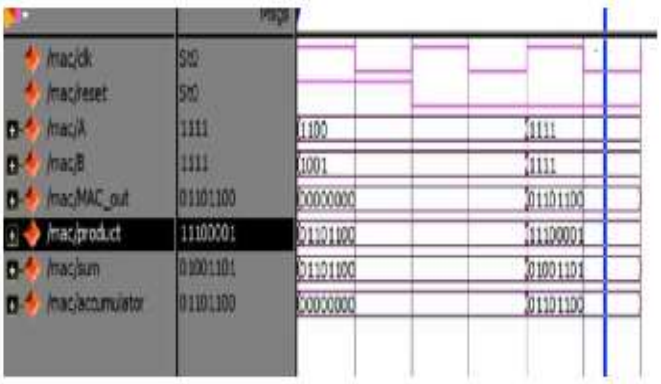


Fig. 5 Simulation-based output of MAC Unit with RCA

Different adders but the same multiplier were also included into the MAC Unit's architecture. Both RCA and CLA were included. Figures 6 and 7 show the results of the MAC Unit's simulation using these adders.

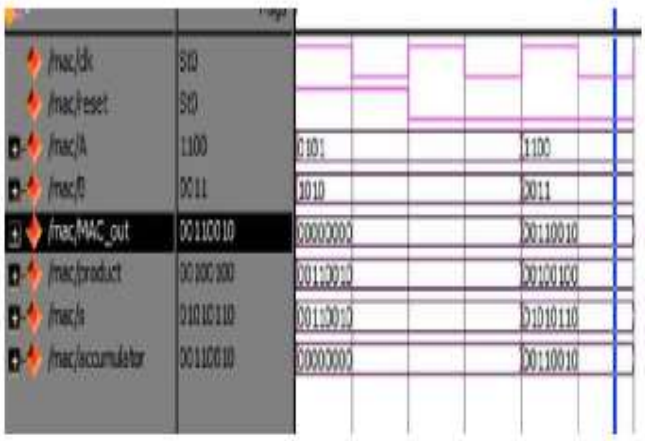


Fig. 6 Simulation-based output of MAC Unit with CLA

Table 2 compares and contrasts the adders Kogge Stone and Ladner-Fischer with the design utilization summaries that were also collected. Additionally, the synthesis report for these adders was created and used to determine their latency.

TABLE II DESIGN UTILIZATION SUMMARY OF VARIOUS ADDERS (COMPARATIVE RESULTS)

Logic Utilization	RCA	CLA	Kogge Stone	Ladner Fischer
Number of occupied Slices	10	13	18	14
Number Of 4-Input LUTs	17	22	31	24
Number Of Bonded IOBs	26	26	32	32
Delay	17.10	15.13	13.30	11.82

TABLE III PERFORMANCE OF THE MAC UNIT WITH DIFFERENT ADDERS (COMPARATIVE RESULTS)

Logic Utilization	RCA	CLA	Kogge Stone	Ladner Fischer
Number of occupied Slices	33	29	31	31
Number Of 4-Input LUTs	61	53	57	57
Number Of Bonded IOBs	18	18	18	18
Delay	6.49	6.46	6.46	6.39

Table 3 shows the results of a comparison between the logic consumption tables of the MAC Units that used the same multiplier but different adders.

V. CONCLUSION

utilizing a comparison analysis based on delay performance, the MAC Unit was thoroughly analyzed utilizing several adder designs. Because its individual adders follow a parallel prefix structure, the Ladner Fischer adder showed the shortest delay of 11.82 ns, while the Kogge-Stone adder showed the second-shortest delay of 13.30 ns. Kogge-Stone and carry look-ahead adder had a latency of 6.46 ns, while Ladner-Fischer obtained the MAC unit's lowest delay of 6.39 ns. Using Kogge-Stone and CLA in the MAC Unit setup makes these adders work better for quicker carry propagation. The RCA, on the other hand, has the slowest propagation because of its linear carry chain.

REFERENCES

1. D. H. S. S. K. B.Hemalatha, "Design of MAC Unit for DSP Applications using Verilog HDL," 2019.
2. Sudhanya, P., and SP Joy Vasantha Rani. "Wire-length and run-time optimization in fpga placement using hybrid iterative algorithms." Journal of Circuits, Systems and Computers 30, no. 05 (2021): 2150081.
3. Sudhanya, P., Joy Vasantha Rani, S.P. (2020). "Adaptive Particle Swarm Optimization Based Wire-length Minimization for Placement in FPGA",New Trends in Computational Vision and Bio-inspired Computing. Springer, Cham.
4. R. Chikkani, B. M. and Y. J. M Shirur, "VLSI Implementation of Multiply and Accumulate Unit using Distributed Arithmetic," Bioscience Biotechnology Research Communication, vol. 13, pp. 212-217, 2020.
5. G. Dimitrakopoulos, K. Papachatzopoulo and V. Paliouras, "Sum Propagate Adders," IEEE Transactions on Emerging Topics on Computing, 2021.
6. M. Sultana, B. Arunalatha and P. Ramyaraju, "MAC Unit Design using Multiplier & Ripple Carry Adder," International Journal of VLSI System Design and Communication Systems (IJVDCS), vol. 03, no. 10, pp. 1538 1540, 2015.
7. N. Khare, D. Rao, and R. Mohan, "VLSI Implementation of High-Speed MAC Unit Using Karatsuba Multiplication Technique," Journal of Network Communications and Emerging Technologies (JNCET), vol. 6, no. 1, 2016.
8. A. Farooqui and V. Oklobdzija, "General data-path organization of a MAC unit for VLSI implementation of DSP processors," in IEEE, Monterey, CA, USA, 2002.
9. K. B. R. Dinesh, R. Vinoth and M. Kasyap, "Design and Implementation of High Speed 32-bit MAC Unit," in Journal of Physics: Conference Series, 2023.
10. M. S. Shanthala, C. Raj and S. Kulkarni, "Design and VLSI Implementation of Pipelined Multiply Accumulate Unit," in International Conference on Emerging Trends in Engineering and Technology, ICETET, 2009.
11. A. Laxman, N. S. S. Reddy and B. R. Naik, "Design and implementation of hybrid logic based MAC unit using 45 nm technology," ELSEVIER, vol. 6, 2023.
12. N. P. R. Reddy and K. M. Devi, "Design of High Performance 64 bit MAC Unit," International Journal for Advanced Research In Science and Technology (IJARST), vol. 10, no. 11, 2020.

13. R. Anitha, N. Deshmukh, S. K. Sahoo, S. P. and J. R. I., "A 32 BIT MAC Unit Design Using Vedic Multiplier and Reversible Logic Gate," in IEEE, International Conference on Circuit, Power and Computing Technologies [ICCPCT], 2015.
14. M. SaiKumar, D. A. Kumar and D. Samundiswary, "Design and Performance Analysis of Multiply-Accumulate (MAC) Unit," in IEEE, International Conference on Circuit, Power and Computing Technologies [ICCPCT], 2014.
15. A. Laxman, D. N. S. S. Reddy and D. B. R. Naik, "Design and Implementation of Hybrid Multiplier for DSP Applications," International Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 10.
16. D. Harris, "A taxonomy of parallel prefix networks," in Signals, Systems, and Computers," in Conference Record of Thirty-Seventh Asilomar, 2003.