Design of 8-Bit Wallace Tree Multiplier Using 4:2 Compressor

***Abstract*—Multipliers are the fundamental arithmetic units that significantly influence the performance of architectures across a wide range of applications. The efficiency of these multipliers depends on the effective reduction of partial products while maintaining optimal power and delay constraints. In this project, we have implemented the schematic and layout of an 8-bit Wallace tree multiplier circuit using 4-2 compressors. We have verified the functionality of our design for different input combinations. Our design achieves a worst-case propagation delay of 1.42 ns, occupies an area of 1916.57μm^2 and consumes 609.038 fJ of energy. Additionally, we have developed a Multiply and Accumulate (MAC) unit based on this multiplier and thoroughly evaluated its functionality and performance in terms of area, power and delay metrics.**

***Index Terms*—Multiplier, 4:2 compressor, Multiply and Accumulate, energy, delay, area.**

I. Introduction

Multipliers circuits are extensively utilized in applications such as digital signal processing, machine learning, image processing, cryptography etc. They are repeatedly called upon for complex mathematical operations like convolution, matrix multiplication and transform computations. As a result, optimizing multiplier designs is critical for enhancing the overall performance and energy efficiency of these systems.

Tree multiplier architectures typically operate in three main phases: partial product generation, partial product reduction and the final addition to produce the result. Among these three phases, the partial product reduction phase is the most power-intensive and significantly contributes to the overall critical path delay, making it a crucial focus for optimization in multiplier design [1]. Wallace tree multiplication algorithm is beneficial in terms of speed of operation since it reduces the number of partial products as much as possible on each layer [2].

In this project, we have used 4-2 compressors for partial product reduction. By using these compressors, we can efficiently reduce multiple partial products into fewer bits in each stage of the reduction process, thereby reducing the overall delay of the multiplier. In the next sections, we will explore the design of our multiplier circuit and the 4-2 compressor. We will also present a thorough evaluation of our design in terms of various performance metrics like area, power and propagation delay.

II. Topology Selection for different circuit elements

We have used 45 *nm* CMOS technology to implement the primitive gates. Additionally, we have used CMOS as well as transmission based logic to based design our ciruits as explained below.

A. NAND, NOR and NOT gates

The primitive gates like NAND, NOR and NOT are designed using standard CMOS logic with NMOS and PMOS transistors. CMOS gates offer full rail-to-rail output, high noise margin and low power consumption. Hence, they are preferred for the design of these primitive gates.

B. XOR gate and 2:1 Multiplexer

The XOR gate and 2:1 MUX are designed using transmission gates since they require fewer number of transistors as compared to CMOS based logic. Additionally, transmission gates have minimal leakage and static power dissipation during steady states. Hence, we have used transmission based design for XOR and 2:1 MUX.

III. Wallace Tree Multiplier Design

We have implemented the Wallace tree multiplier using a Partial Product Generator circuit, 4:2 compressors, full and half adders and finally a Carry Look-ahead adder. This section explores the design and implementation of each of these blocks.

A. 4:2 Compressor

A 4:2 compressor takes four input bits and an additional carry input bit C_{in} from the previous column of partial products. It compresses these inputs to produce two outputs, namely Sum and $Carry$ along with an intermediary carry bit C_{out}, which serves as the carry input C_{in} for the next column [3]. The input–output relation of 4:2 compressor can be expressed mathematically as given by equation (1).

$$A_1+A_2+A_3+A_3+A_4+C_{in} = Sum+2(Carry+C_{out}) \quad (1)$$

Here, we note that $Carry$ and C_{out} are the outputs having significance of 1 bit higher than the five input bits A_1, A_2, A_3, A_3, A_4 and C_{in}. Moreover, in contrast to a full

adder where both the Sum and the C_{out} are influenced by the C_{in} from the previous stage, a 4:2 compressor decouples C_{out} from C_{in}. This independence allows parallel computation of carry outputs, reducing the overall propagation delay [4].

We have implemented the first design of the 4:2 compressors suggested by Priyadharshni et al. [3]. The circuit diagram is shown in Fig. 1. Since XOR gate is susceptible to more delay and area, this compressor circuit is designed by minimizing the number of XOR gates to 3 against 4 XOR gates used in traditional 4:2 compressor. Besides, 2:1 Multiplexers (MUX), which are made using transmission gates, are used to reduce the delay further. In Fig. 1, we can see that there are two interim signals Si and $Condition$ which are given by equations (2) and (3) respectively.

$$S_i = A_1 \oplus A_2 \oplus A_3 \tag{2}$$

$$Condition = A_4 \oplus C_{in} \tag{3}$$

The expression for Sum, $Carry$ and C_{out} are given by equations (4), (5) and (6). Sum is determined based on S_i and A_4. The expression in equation (4) is equivalent to the conventional expression for Sum but it is optimized using MUX logic. Similarly, $Carry$ is also generated based on the MUX logic. Finally, a dual stage NAND based model is used to produce C_{out}.

$$Sum = \overline{S_i}(A_4 \oplus C_{in}) + S_i(A_4 \odot C_{in}) \tag{4}$$

$$Carry = (A_4 \oplus C_{in})C_{in} + (A_4 \odot C_{in})S_i \tag{5}$$

$$C_{out} = \overline{(A_1A_2)} \cdot \overline{(A_2A_3)} \cdot \overline{(A_1A_3)} \tag{6}$$

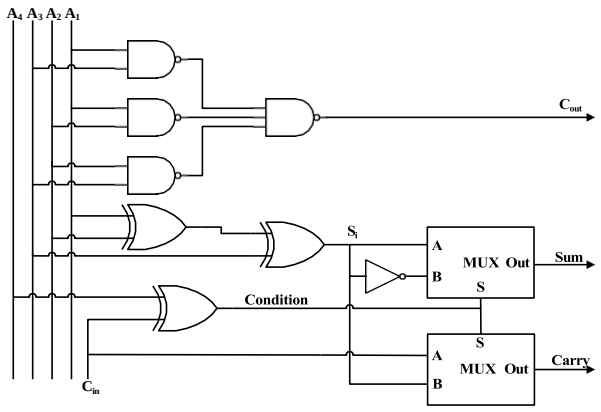

Fig. 1. 4:2 Compressor design

B. Partial Product Generator

The Partial Product Generator produces the partial product for each bit of the multiplier and the multiplicand using AND operation. Using two 8-bit inputs, a total of 64 partial products are obtained which are given as inputs to the proceeding circuits for partial product reduction. Each block in the Partial Product Generator consists of an array of 8 AND gates which generate the partial products from $A_i \cdot B_0$ to $A_i \cdot B_7$. We have used 8 such blocks to generate the partial products for all 64 combinations of the input bits.

C. Carry Look-ahead Adder

In the last stage of our Wallace tree multiplier design, we use three cascaded 4-bit Carry Look-ahead (CLA) adders to form a 12-bit Carry Look-ahead adder. Unlike other adder topologies like Ripple Carry adder, a Carry Look-ahead adder computes carry signals in parallel using generate and propagate logic, eliminating the delay caused by sequential carry propagation. The expressions for the propagate signal P_i and the generate signal G_i are given in equations (7) and (8) respectively.

$$P_i = A_i \oplus B_i \tag{7}$$

$$G_i = A_i \cdot B_i \tag{8}$$

These signals are used to generate the Sum and $Carry$ for the CLA adder as given below in equations (9) and (10).

$$S_i = P_i \oplus C_i \tag{9}$$

$$C_{i+1} = G_i + P_i \cdot C_i \tag{10}$$

D. 8-bit Wallace Tree Multiplier

The schematic of the 8-bit Wallace tree multiplier implemented using 4:2 compressors is shown in Fig. 2. The 4-2 compressors are used in Stage 1 and Stage 2, along with full adders and half adders for partial product reduction. In the final stage, a 12-bit Carry Look-ahead Adder (CLA) is used to generate the output product.

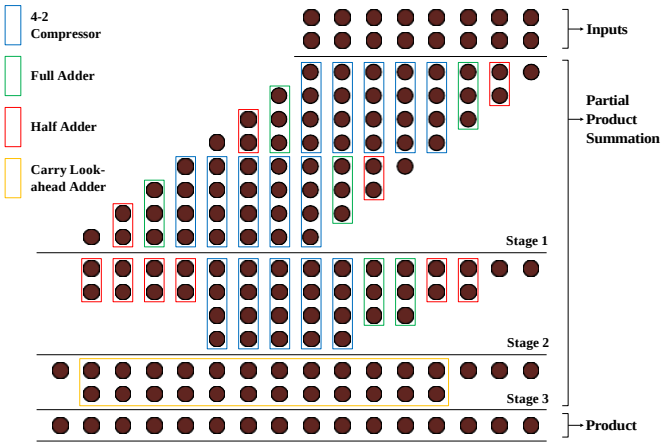

Fig. 2. 8-bit Wallace tree multiplier design

IV. Sizing Strategy

The width of NMOS and PMOS are sized for the primitive gates by equalizing the propagation delays when the output transitions from low to high (t_{plh}) and from high to low (t_{phl}) for the inputs which lie in the critical path (farthest from the output). This strategy helps in minimizing power consumption by ensuring symmetrical rise and fall times on the output

signal, leading to better overall circuit performance. The width of NMOS and PMOS for different logic gates used in our design are shown in Table I. Their corresponding delay is also mentioned for $V_{dd} = 1.1\ V$.

TABLE I
WIDTH OF PMOS AND NMOS FOR DIFFERENT GATES

Gate	PMOS Width *nm*	NMOS Width *nm*	Delay *ps*
Inverter	175	120	21
NAND 2I	180	240	34.25
NAND 3I	180	360	41.2
NAND 4I	180	480	49.8
NOR 2I	340	120	36.3
NOR 3I	470	120	44.83
NOR 4I	590	120	54.55
Transmission Gate	240	240	15.92

V. SIMULATION RESULTS

The schematic of our Wallace tree multiplier as implemented in *Cadence Virtuoso* is shown in Fig. 3.

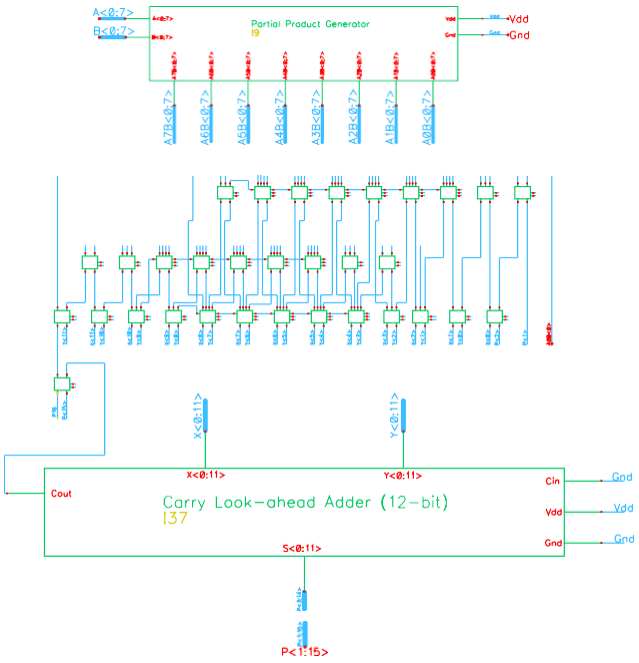

Fig. 3. Wallace tree multiplier schematic

A. Functionality Verification

To perform functional verification of our design, we have implemented a test bench circuit where the Wallace tree multiplier is powered with $1.1\ V$ supply and all the outputs are connected to $2\ fF$ capacitors. The rise/fall time of all the input bits is set to $50\ ps$.

The waveform for different input combinations and the corresponding outputs are shown in Fig. 4. We have verified our outputs for the input combinations.

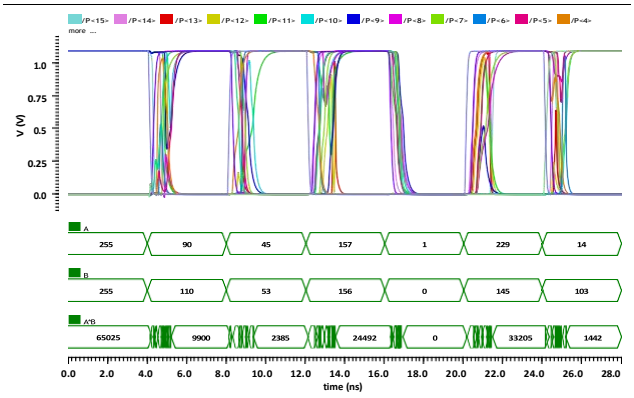

Fig. 4. Waveform showing output for different input combinations

B. Layout and Total Area

The layout of the Wallace tree multiplier circuit is shown in Fig. 5. The total area of our layout is approximately **1916.571525 μm^2**.

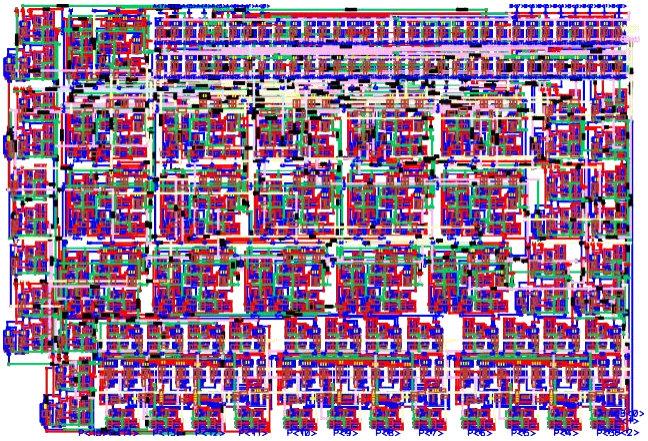

Fig. 5. Layout of Wallace tree multiplier

After designing the layout, we perform parasitic extraction for energy and latency analysis.

1) DRC and LVS checks: Our layout has successfully passed DRC and LVS checks as shown in Fig. 6 and Fig. 7 respectively.

C. Worst Case Propagation Delay and Corresponding Input

The propagation delay is measured for the time taken by the input to transition to 50% of V_{dd} to the time taken by the

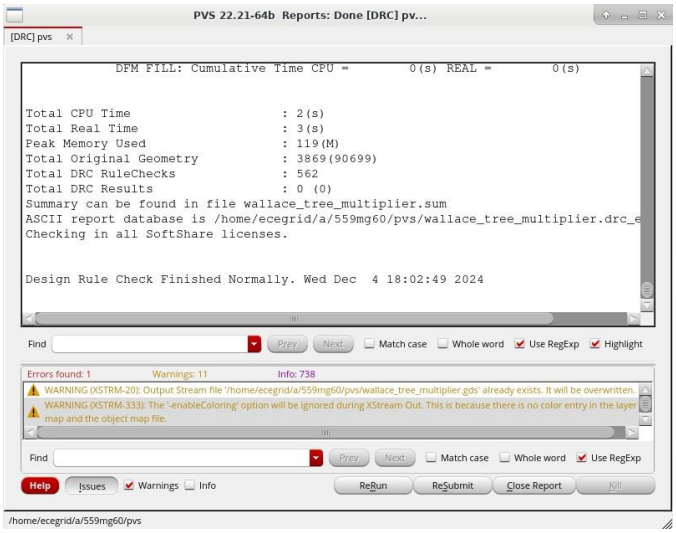

Fig. 6. DRC report for Wallace tree multiplier.

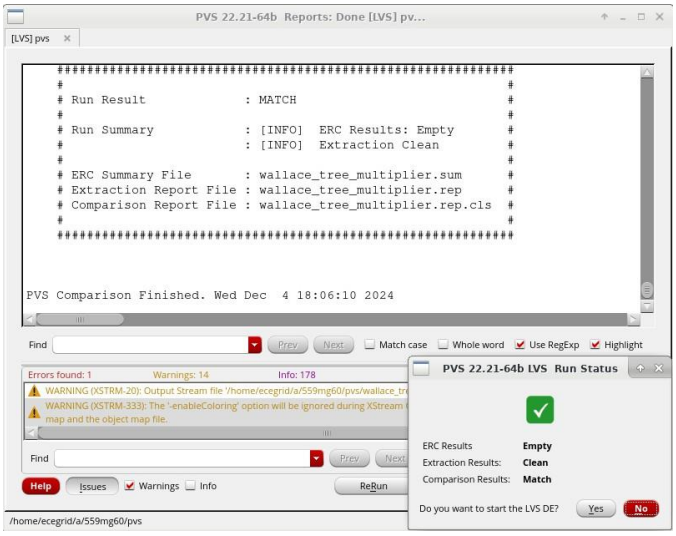

Fig. 7. LVS report for Wallace tree multiplier

output to stabilize to 50% of V_{dd}. The worst case propagation delay depends on several factors like the critical path for the output signal, transistor sizing, layout implementation, parasitic capacitance and resistance extracted from layout, interconnect capacitance and resistance, power supply, signal crosstalk and so on.

The maximum propagation delay of our Wallace tree multiplier arises from the output bit P (8) as shown clearly in Fig. 8. We can also see from Fig. 2 that P (8) occurs approximately in the middle of our Wallace tree architecture, hence it typically encounters the most number of carries and additions due to overlapping partial products. The bit P (8) becomes high for the input A = 157 and B = 156. The maximum delay for this combination is **1.42482 ns**.

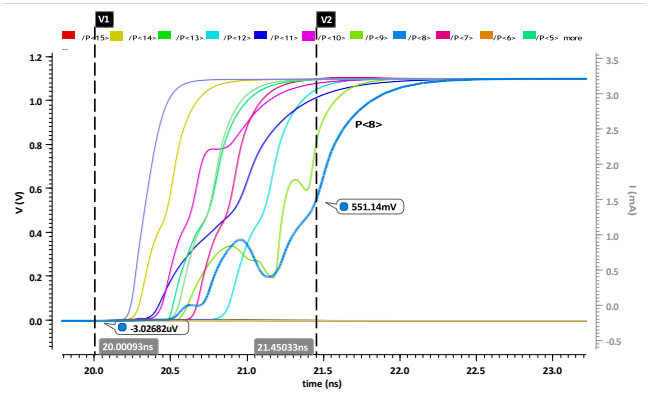

Fig. 8. Graph showing output transitions for maximum delay

D. Energy Analysis

Using our worst-case input pair having A = 157 and B = 156, we calculate the maximum energy consumption for our design which comes out to be around **609.038 fJ**, which can be verified from Fig. 9.

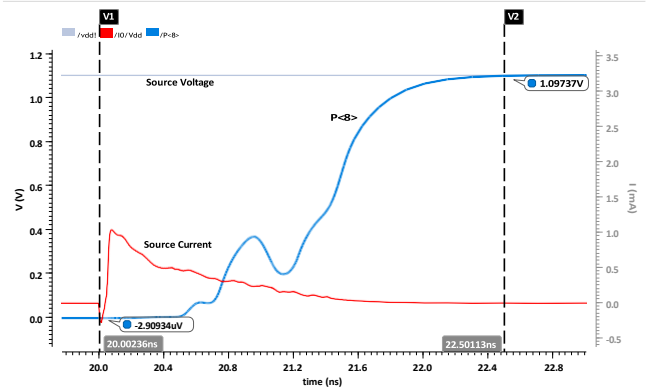

Fig. 9. Graph used for energy calculation in Wallace tree multiplier

E. Results summary and analysis

A summary of our Energy-Delay-Area analysis is shown in Table II.

By varying the supply voltage from 1 V to 1.2 V, we have calculated the values of Energy, Delay and Energy-Delay Product (EDP) as depicted in Table III. From this table, we can see that EDP is minimum at 1.2 V.

TABLE II
SUMMARY OF RESULTS FOR WALLACE TREE MULTIPLIER

Area μm^2	Delay ns	Energy fJ
1916.571525	1.42482	609.038

TABLE III
VARIATION OF ENERGY AND DELAY WITH POWER SUPPLY

Supply Voltage V	Energy fJ	Delay ns	EDP $ns \times fJ$
1.0	495.1	1.74	861.47
1.1	609.04	1.42	864.83
1.2	732.0	1.16	849.12

VI. MULTIPLY AND ACCUMULATE MODULE

A Multiply and Accumulate module (MAC) multiplies two numbers and adds the product to an accumulator. It consists of consists of a multiplier, adder and accumulator. MAC unit is a fundamental block in the computing devices, especially Digital Signal Processor (DSP) [5].

We have implemented a MAC module using our Wallace tree multiplier, a 16-bit Carry Look-ahead adder and a 17-bit Parallel Input Parallel Output (PIPO) shift register which works as an accumulator. The schematic of our MAC unit is shown in Fig. 10.

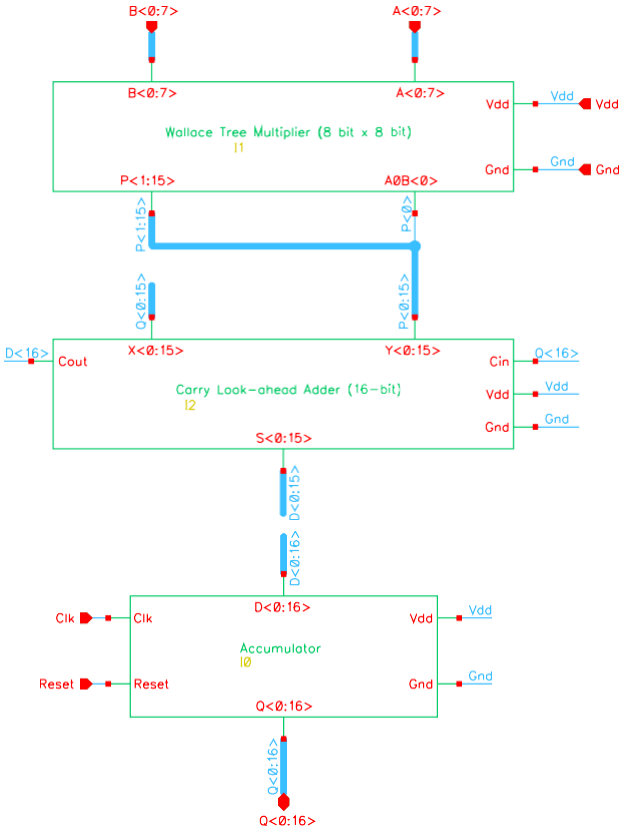

Fig. 10. Schematic of MAC unit

A. Accumulator

We have designed the accumulator unit using a 17-bit Parallel Input Parallel Output (PIPO) shift register. The accumulator stores the output from the CLA adder (which includes 16 output bits and 1 carry bit) and sends it back to the CLA in the next clock cycle so that the new output from the multiplier can be added with the value present in the accumulator. The accumulator can also be set to 0 asynchronously using a *Reset* pin.

The value of each bit in the accumulator is stored using a D Flip-Flop. The schematic for the D Flip-Flop is adapted from [6] and is shown in Fig. 11.

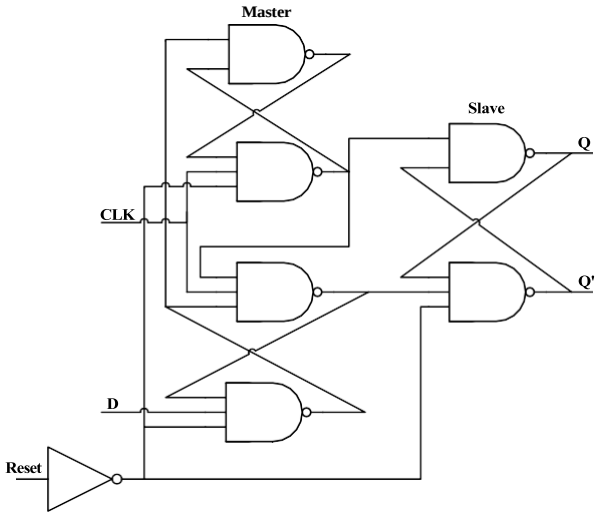

Fig. 11. D Flip-Flop Circuit

B. Simulation Results

1) Functional Verification: The MAC unit is verified for different input combinations as shown in Fig. 12. From this graph we can see that when Reset is 0, then at the rising edge of the Clock the output of the multiplier gets added with the previous value stored in the accumulator, which is the output given by the MAC unit.

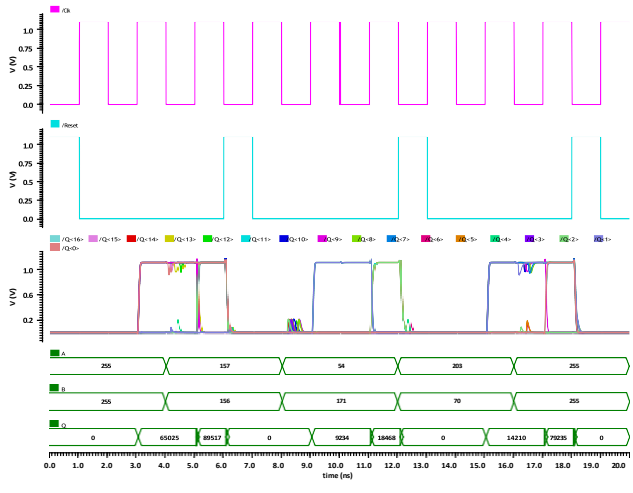

Fig. 12. Graph showing functional verification for MAC

2) Layout and Total area: The layout of our MAC unit is shown in Fig. 13. Total area of the layout is **2954.770250 μm^2**.

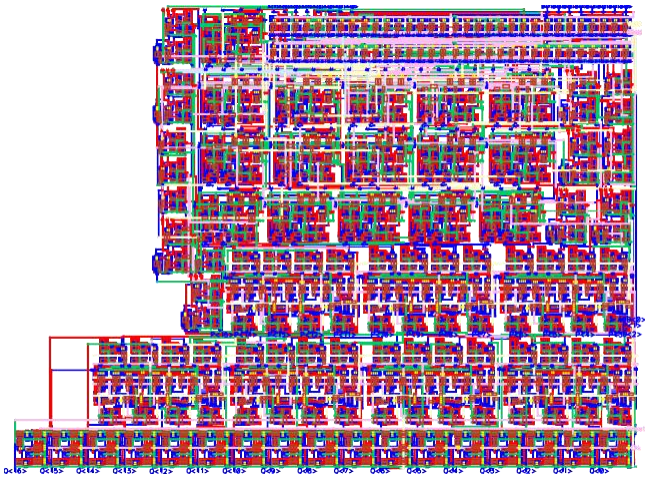

Fig. 13. Layout of MAC

The results for DRC and LVS reports are shown in Fig. 14 and Fig. 15 respectively.

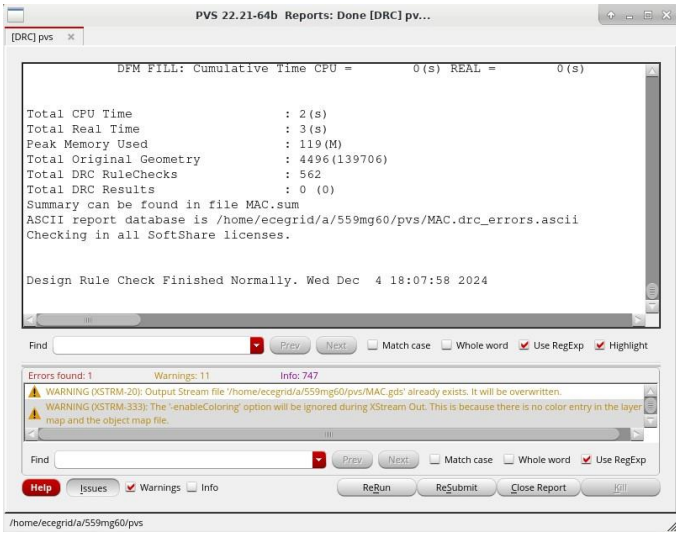

Fig. 14. DRC report for MAC

3) Clock-to-Q Delay: For calculating the Clock-to-Q delay of the MAC unit, we first store the highest output produced by the Wallace tree multiplier in MAC, that is 65025 produced when the inputs A and B are both equal to 255. To this value, we add the output produced by the worst case input pair, $A = 157$ and $B = 156$ for the Wallace tree multiplier. Hence, the output produced by MAC is $65025 + (157 \times 156) = 89517$. The time taken by the MAC to produce this output after the rising edge of the clock gives our Clock-to-Q delay which is **850.85 ps** as shown in Fig. 16.

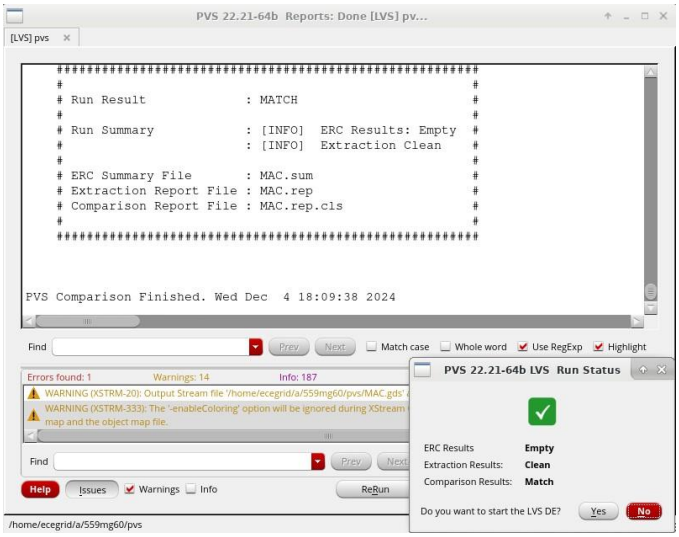

Fig. 15. LVS report for MAC

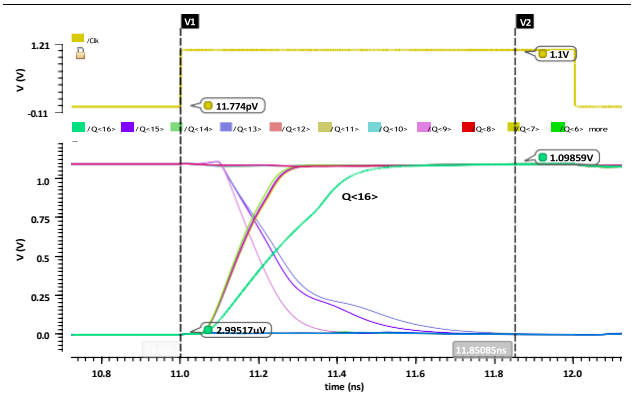

Fig. 16. Waveforms showing Clock-to-Q delay of MAC

C. Energy

The energy of the MAC unit is calculated by integrating the current and voltage for the time interval in which the Wallace tree produces a stable output to the the time when the accumulator produces a stable output after the rising edge of the Clock as given in Fig. 17. The total energy consumed for one MAC operation is **1.297 pJ**.

The simulation results for MAC module are summarized in Table IV.

TABLE IV
SUMMARY OF RESULTS FOR MAC MODULE

Area μm^2	Delay *ps*	Energy *pJ*
2954.770250	850.85	1.297

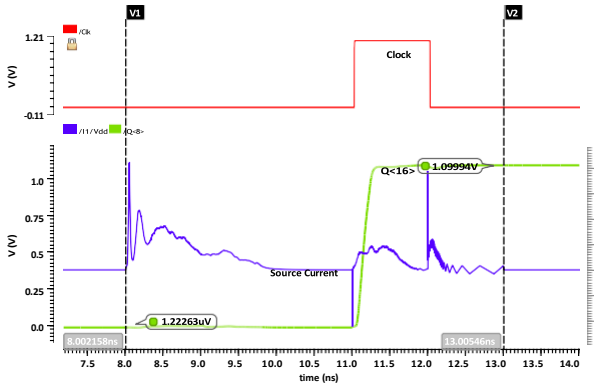

Fig. 17. Waveform used for energy calculation in MAC

VII. CONCLUSION

We have successfully demonstrated the implementation of an 8-bit Wallace tree multiplier and a corresponding Mul- tiply and Accumulate module. The Wallace tree multiplier efficiently performs multiplication by reducing partial prod- ucts through hierarchical stages of half adders, full adders and 4:2 compressors, followed by a final Carry Look-ahead adder to produce the result. The MAC module combines this multiplier with an accumulator to enable repeated multiply- and-accumulate operations. The functionality of the designs was rigorously verified for various input combinations and key implementation metrics, including area, worst-case delay and energy consumption, were obtained to evaluate performance and efficiency.

REFERENCES

[1] P. Hasini and T. K. Murthy, “A novel high-speed transistorized 8×8 multiplier using 4-2 compressors,” *International Journal of Engineering Research and General Science*, vol. 3, no. 2, pp. 359–365, 2015.

[2] C. S. Wallace, “A suggestion for a fast multiplier,” *IEEE Transactions on Electronic Computers*, no. 1, pp. 14–17, 1964.

[3] M. Priyadharshni, A. Chathalingathu, S. Kumaravel, A. Manoharan, S. Veeramachaneni, and S. N. Mahammad, “Logically optimal novel 4:2 compressor architectures for high-performance applications,” *Arabian Journal for Science and Engineering*, vol. 45, pp. 6199–6209, 2020.

[4] M. Kumar and J. Nath, “Design of an energy-efficient 4-2 compressor,” in *IOP Conference Series: Materials Science and Engineering*, vol. 225, no. 1, IOP Publishing, 2017, p. 012136.

[5] M. S. Kumar, D. A. Kumar, and P. Samundiswary, “Design and performance analysis of multiply-accumulate (MAC) unit,” in *2014 International Conference on Circuits, Power and Computing Technologies (ICCPCT-2014)*, IEEE, 2014, pp. 1084–1089.

[6] G.-M. Sung, C.-C. Huang, X. Xiao, and S.-Y. Hsu, “10-bit 5 MS/s successive approximation register analog-to-digital converter with a phase-locked loop and modified bootstrapped switch for a BLDC motor drive,” *Electronics*, vol. 11, no. 4, p. 624, 2022.