Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 46 No. 04 (2025)

Efficient Physical Design Techniques for High-Speed, Area-Optimized 12x12 Multiplier Using CSA

Eswararao Boddepalli[1], S V Jagadeesh Chandra[2], E. V. Krishna Rao[3], K Srinivasa Yadav[2], P Sasanka[5]

[1]Department of ECE, NIT Meghalaya, Shillong, India.
[2]Department of Physics, GITAM (Deemed to be) University, Visakhapatnam-530045, Andhra Pradesh, India
[3]Department of ECE, Lakireddy Bali Reddy College of Engineering, Mylavaram, India
5Edubot Technologies, USA

Abstract:

Designing a high-speed, area-efficient 12-bit by 12-bit multiplier architecture tailored for performance-driven digital systems. Traditional multiplication schemes often encounter challenges in minimizing computational delay and silicon footprint simultaneously. The proposed design adopts a carry save addition (CSA)-based array multiplier structure, which effectively reduces carry propagation delay by deferring carry processing to the final computation stage. The implementation utilizes CMOS-compatible components, including dynamic flip-flops and optimized logic cells, to achieve a compact layout. Furthermore, hierarchical design strategies and advanced physical layout techniques are employed to improve signal integrity and manufacturability. The resulting multiplier demonstrates notable enhancements in speed and area efficiency, making it a strong candidate for integration into high-performance digital processing architectures.

Key Terms: Carry Save Adder, Array Multiplier, CMOS Design, Area Optimization and Signal integrity.

I. Introduction:

A 12-bit multiplier is a digital circuit designed to compute the product of two 12-bit binary numbers, producing a 24-bit output. The architecture illustrated follows an array multiplier approach, constructed in a systematic and modular fashion. In this configuration, a grid of processing elements—each comprising a full adder (FA) and supporting logic—handles carry and partial product signals. The array is organized such that each row corresponds to a bit from the multiplier and each column to a bit from the multiplicand. Partial products are generated using bitwise AND operations between corresponding bits of the multiplicand and multiplier [1]. These partial products are then summed using a matrix of full adders, which propagate sum and carry signals through the array. The initial columns are responsible for raw partial product generation, while subsequent columns handle their accumulation. To finalize the result, a fast carry-propagate adder is employed in the lower section of the circuit, efficiently summing the remaining bits to produce the complete 24-bit output, labeled P0 through P23 (with some outputs not shown in the illustration) [2]. Owing to its parallel structure and systematic organization, this multiplier architecture is well-suited for high-speed digital applications such as processors and digital signal processing units.

Three widely used multiplier architectures in digital design are the Wallace Tree multiplier, the Braun multiplier, and the Array Multiplier with Carry Save Addition (CSA). Each offers distinct architectural advantages suited to

various design requirements. The Wallace Tree multiplier is recognized for its high-speed performance, primarily due to a reduction in sequential addition stages. However, its complex structure can introduce routing challenges and complicate physical implementation. The Braun multiplier, in contrast, features a more straightforward layout, but tends to occupy more silicon area and is less scalable for higher bit-width operations [3]. The Array Multiplier with Carry Save Addition provides a compelling balance between design simplicity and computational performance. It is well-suited for implementation using standard digital components and integrates efficiently with pipelining techniques. By employing CSA, the architecture mitigates the delays typically introduced by carry propagation in traditional adders, enabling faster multiplication operations. The design is composed of fundamental digital building blocks such as full adders, dynamic flip-flops, and AND gates. These components are readily implemented using standard CMOS logic, with AND gates constructed from basic NAND gates and inverters, ensuring structural simplicity.

To further enhance computational speed and throughput, both pipelining and parallel processing techniques are applied. Pipelining enables multiple stages of the multiplication process to execute simultaneously, thereby reducing overall latency. Parallelism in the accumulation of partial products allows faster management of carry bits, improving performance without introducing significant architectural complexity. Special consideration is given to propagation delay, particularly within the adder and logic gate stages, to maintain optimal signal timing throughout the design. Additionally, design strategies focus on minimizing silicon area to support compact and efficient integration. Overall, the Array Multiplier with Carry Save Addition offers an effective compromise between speed, area efficiency, and ease of implementation, making it an ideal candidate for high-performance digital multiplier applications.

II. Design:

A. Full Adder

1) Design Approach:

At the beginning of the project, the design team focused on making each part of the multiplier as efficient as possible. A particular priority was the full adder, a key building block in digital circuits as shown in fig-1(a). The initial design chosen was a mirror adder topology [4], known for its simplicity and effectiveness. However, it had a significant drawback; it required relatively large transistor dimensions to operate reliably, which led to increased power consumption and chip area. To improve this, the existing full adder design was taken as a starting point, and efforts were made to optimize it for better performance and lower power usage.

2) Schematic Optimization:

The first step in optimization involved reducing the sizes of the transistors in the schematic. This was done in multiple stages. After each stage of size reduction, simulations were conducted to verify that the circuit's behavior remained correct, with no significant issues in output shape or timing. Once the basic size reductions proved successful, further refinements were applied by adjusting the sizes of specific transistors individually. Rather than applying the same change across all components, each transistor was fine-tuned based on its role in the circuit. This targeted optimization helped achieve an effective balance between performance, area, and power efficiency.

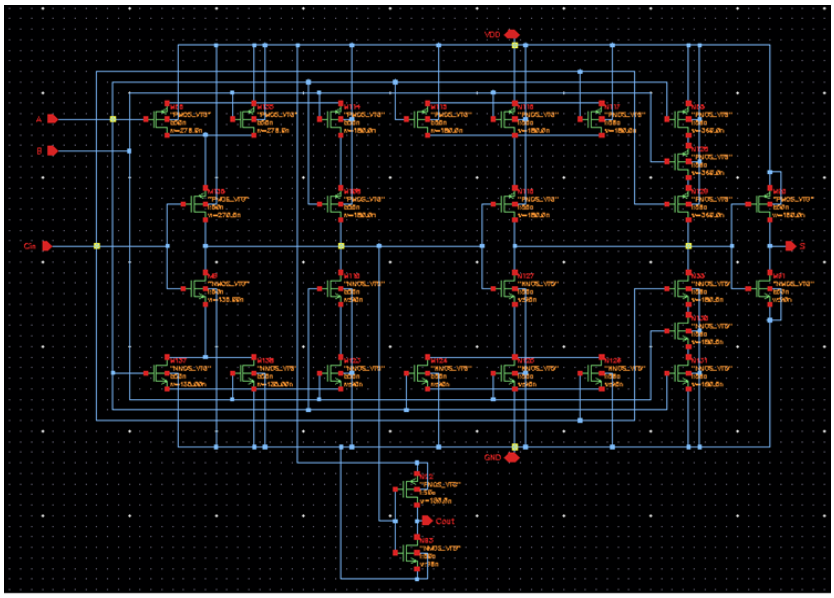

Fig-1: Full Adder Schematic diagram

After the schematic was finalized, the next step was to modify the corresponding physical layout. Using the original layout as a base, adjustments were made to shrink the areas allocated to wells and diffusion regions, in line with the reduced transistor sizes. This layout revision led to a smaller footprint for the full adder on the silicon chip, contributing to overall area savings. Once the layout was finalized and validated, the design was ready to be integrated with other circuit components such as logic gates and memory elements, preparing it for use in the larger digital system [5].

B. Dynamic Flip-Flop:

1) Design:

For the sequential logic portion of the project, a dynamic D-type flip-flop was selected due to its simplicity, efficiency, and compact layout. The chosen design follows a master-slave configuration, where the master and slave latches are controlled by complementary clock signals. The architecture is built using transmission gates and inverters, forming a structure similar to a multiplexer-based D flip-flop [6].

This approach offers several advantages:

i) Compact area: The use of transmission gates instead of larger logic gates helps reduce the overall footprint of the flip-flop on silicon.

ii) CMOS compatibility: The entire design is implemented using basic CMOS components, making it easy to integrate with other digital logic circuits.

iii) Performance: The simplicity of the design contributes to low propagation delay and efficient switching behavior.

Because of its clean and effective design, no further modifications were necessary, and it was used directly in the system as originally designed.

2) Schematic:

The schematic clearly illustrates the flip-flop's internal structures. **Two stages** (master and slave) work together to latch the input data on one clock edge and release it on the next. **Transmission gates** act as clock-controlled switches to transfer and isolate signals at the appropriate times [7]. Inverters are used for buffering and feedback within each latch stage to maintain data integrity. These schematic forms the logical foundation of the dynamic flip-flop and ensures proper data capture and hold behavior shows below figure-4.

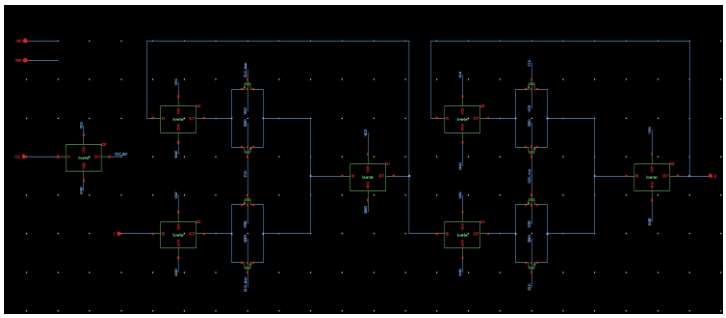

Fig-4: Schematic for Multiplexer based Master Slave D Flip-Flop

C. Full Adder with Integrated AND Gate:

This design enhances a standard full adder by integrating an AND gate at one of its inputs. The AND function is implemented using a NAND gate followed by an inverter, a common logic-level conversion where the inverted NAND output produces the desired AND behavior [8]. The two primary inputs (A and B) feed into the NAND gate, and its output passes through the inverter before connecting to input B of the full adder. The remaining full adder inputs are assigned as follows: the carry-in (Cin) signal connects directly to its designated port, while the sum output (Sum Out) is linked to the full adder's input A. The circuit's final outputs—Sum Out and carry-out (Cout)—deliver the computed results based on these logic operations.

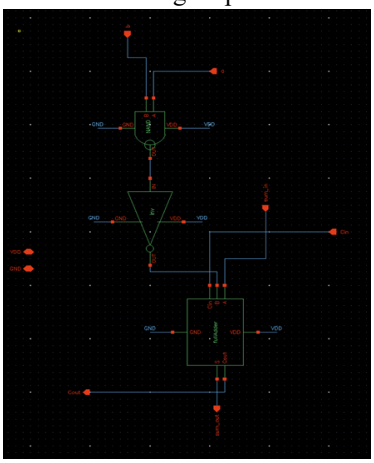

Fig-6: Schematic of the Full Adder with AND Gate

The schematic captures this configuration, with the NAND-inverter combination driving the full adder's input B. For the physical layout, the NAND gate's output is routed to the inverter using a metal-1 layer, and the inverter's output connects to the full adder via a metal-2 layer with an intermediate via. Input and output ports are strategically placed to ensure proper connectivity and functionality. This approach maintains logical integrity while leveraging efficient routing techniques for optimal performance.

D. 12-Bit Array Multiplier Using Carry-Save Addition:

The 12-bit array multiplier was designed using carry-save addition to efficiently handle partial product accumulation while minimizing propagation delays. The architecture was inspired by a reference 4×4 array multiplier, which consists of 16 identical cells, each containing an AND gate (for generating partial products) and a full adder (for bit-wise addition with carry propagation) [9]. At the edges of the array, flip-flops were used to buffer inputs (via D pins) and store outputs (via Q pins). A fast carry-propagate adder (CPA), composed of three additional full-adder cells, was integrated at the final stage to compute the complete sum. Scaling this design to 12×12 bits required expanding the core array to 144 cells (12 rows × 12 columns) and extending the CPA to 11 full-adders to accommodate the larger bit-width. The peripheral logic, including input/output flip-flops, was also adjusted to ensure proper signal synchronization across the expanded architecture [10, 11].

The schematic implementation followed a hierarchical approach, leveraging modular repetition of the basic AND with full-adder cell for scalability. The carry-save technique was employed to reduce intermediate delays by deferring carry propagation until the final CPA stage.

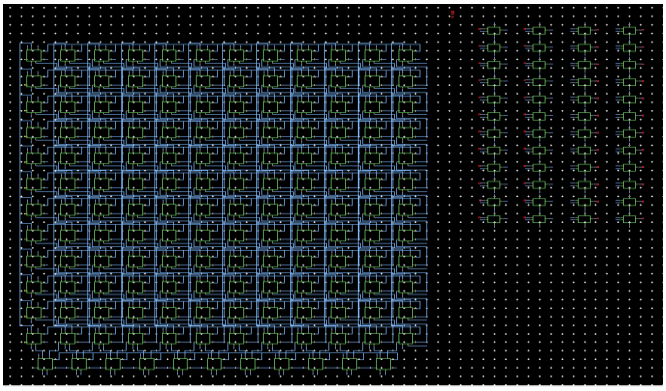

Fig-7: Schematic for 12x12 Multiplier

The top-level symbol abstracted the internal complexity, exposing only the necessary input and output ports for integration into larger systems [12, 13, 14]. Key optimizations included buffered I/O for signal stability and careful routing of carry chains to minimize critical path delays. The complete schematic is illustrated in Figure-7, highlighting the structured arrangement of cells and the final CPA [15, 16].

For the physical layout, several practical adjustments were made to enhance manufacturability and performance. The flip-flops storing outputs were resized in height to simplify placement and routing during integration. Power distribution was optimized using two dedicated Metal-4 (M4) rails: one for VDD to ensure uniform voltage supply across the array, and another for GND to provide low-resistance return paths and reduce noise. Signal integrity was preserved by shielding critical nets, such as high-speed carry lines, to mitigate crosstalk. The final layout, shown in Figure-9, reflects these design choices while maintaining compactness and efficiency.

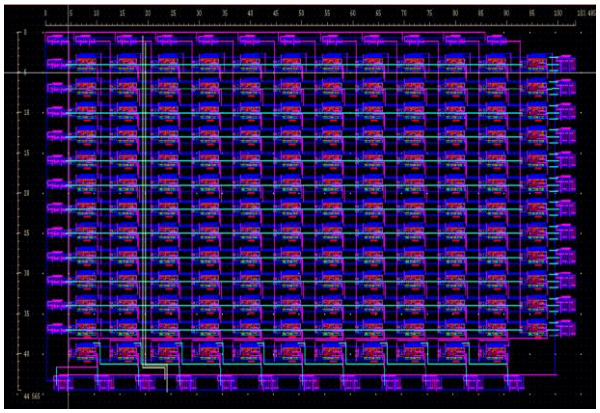

Fig-9: Layout for 12x12 Multiplier using CSA

III. **Testing of the design**:

A. Testbench:

The testbench is designed to thoroughly evaluate the functionality and performance of the multiplier circuit. It consists of a single multiplier unit, 24 capacitors (each connected to an output pin), 24 input pins linked to all input nodes, and 24 output pins connected to every output. This setup ensures comprehensive testing by allowing individual monitoring of each input and output. The capacitors help stabilize the outputs by filtering noise and transient signals, ensuring accurate measurements. The testbench structure, as illustrated in Figure 10, provides a controlled environment to verify the multiplier's behavior under different conditions, including signal transitions, delays, and noise interference.

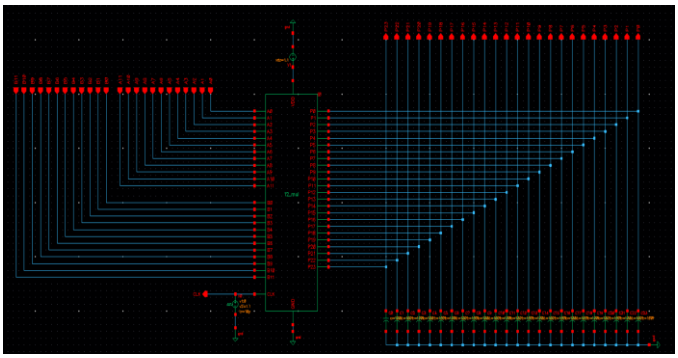

Fig-10: Schematic of Test Bench for 12x12 Multiplier

B. Pre-Layout Testing:

Before physical layout implementation, the multiplier underwent pre-layout simulation to verify its logical correctness. Eight distinct input combinations were tested with a clock period of 10 ns, where inputs switched every clock cycle. To ensure accurate results, the simulation included a waiting period between input changes, allowing the multiplier's internal registers to clear previous data. This step was crucial to prevent residual values from affecting subsequent computations. The simulation confirmed that the 12x12 multiplier functioned as intended, producing correct outputs for all tested input patterns. This phase ensured that the design met logical specifications before proceeding to physical implementation.

C. Post-Layout Testing:

After completing the physical layout, the multiplier underwent rigorous post-layout testing to validate its performance under real-world conditions. The first test replicated the pre-layout simulation setup to confirm that the layout retained correct functionality, with results matching expectations. The second test focused on propagation delay by applying varying inputs along the critical path to identify the worst-case delay. The third test assessed noise immunity by introducing AC bias to the inputs, verifying the circuit's stability under signal interference (Figure 14). Finally, power consumption was evaluated by simulating multiple input variations to measure current draw under different operational conditions (Figure 15). These tests ensured that the multiplier not only functioned correctly but also met timing, noise, and power requirements for reliable operation.

D. Propagation Delay:

Propagation delay in digital circuits refers to the time it takes for a signal to travel from the input to the output of the circuit. In the case of the array multiplier using carry-save addition, the critical path meaning the longest signal path that determines the circuit's overall speed is from input A0 to output P23. This path is particularly important because any delay along it directly affects how quickly the multiplier can produce a correct result. According to the timing information given, the output transitions from a low (logical 0) to a high (logical 1) state. To determine the propagation delay, we subtract the earlier signal transition time (10.0745 nanoseconds) from the later one (11.7121 nanoseconds), resulting in a delay of 1.6376 nanoseconds. This value represents the time required for a signal to propagate through the critical path of the array multiplier and is a key factor in assessing the performance and speed of the digital circuit.

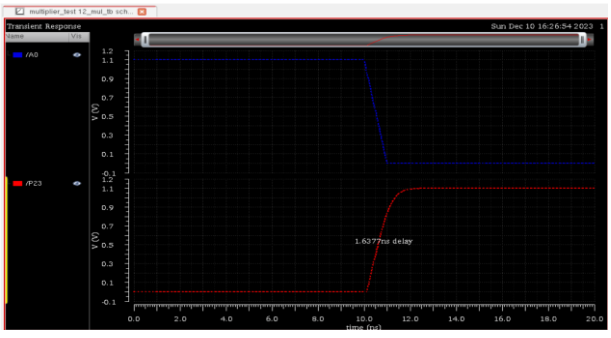

Fig-11: Simulation waveforms of Critical Path

E. Measured Area:

The measured area of a circuit refers to the physical space occupied by the circuit, which is typically represented by the bounding box of the layout diagram. In this case, the dimensions of the bounding box are shown in Figure 9 of the layout diagram. The bounding box has a length of 44,565 nanometers (nm) and a width of 103,485 nanometers (nm). To calculate the total area of the bounding box, we multiply the length by the width, resulting in an area of 4,611,809,025 square nanometers (nm^2). This value represents the total physical area occupied by the array multiplier circuit in the layout and is important for understanding the spatial efficiency and integration of the circuit on a chip.

F. Area Delay Product (ADP):

The Area Delay Product (ADP) is a metric used to evaluate the efficiency of a digital circuit, considering both the propagation delay and the physical area of the circuit, along with its power consumption. The ADP combines these factors to give an overall sense of the circuit's performance in terms of speed, area, and power usage. To calculate the ADP, you multiply the propagation delay (in nanoseconds), the bounding box area (in square nanometers), and the average power (in milliwatts). In this case, the propagation delay is 1.6376 ns, the bounding box area is 4,611,809,025 nm^2, and the average power is 0.3586 mW. Using the formula:

$$ADP = PD\ (ns) \times Box\ Area\ (nm^2) \times Power\ (mW)$$

Substituting the given values:

$$ADP = 1.6376\ ns \times 4{,}611{,}809{,}025\ nm^2 \times 0.3586\ mW$$
$$= 3{,}153{,}952$$

Therefore, the Area Delay Product (ADP) is 3,153,952. This number provides a way to assess the trade-off between speed, area, and power consumption, helping engineers optimize the design for better performance while minimizing resource usage.

IV. Future Scope:

Future work can focus on making the 12x12 multiplier faster and reducing delays by using newer manufacturing technologies and improving key parts of the design. Adding pipelining and parallel processing can help the multiplier handle more operations at once, which is important for powerful computing systems. The design can also be expanded to support larger multipliers like 16-bit or 32-bit to be used in more complex devices. Using new types of chips, such as FinFET or FD-SOI, could improve performance and make the design smaller. Testing the multiplier on different platforms like FPGA and ASIC will help in real-world use. Adding features to detect and correct errors can make the multiplier more reliable, especially for important fields like aerospace and medical electronics. These improvements aim to make the multiplier faster, easier to scale, and more reliable for future digital systems.

V. Conclusion:

The research successfully demonstrates an efficient 12x12 bit multiplier design leveraging carry save addition (CSA) to achieve high-speed operation. By integrating CMOS-compatible components such as dynamic flip-flops and optimized logic gates, and employing hierarchical and advanced physical layout techniques, the multiplier achieves reduced critical path delays and a compact silicon footprint. These improvements validate the effectiveness of the proposed design methodologies in balancing speed and area requirements. The resulting multiplier architecture is highly suitable for integration into complex digital systems requiring fast and efficient arithmetic operations, such as processors and digital signal processors. This work provides a solid foundation for future developments including scaling to higher bit-width configurations and exploring advanced fabrication technologies to further enhance design performance.

VI. References:

[1] Leela, S. N., Chandrika, D. K., Swetha, K., Kalali, D. G., & Shanthi, G. (2024, March). A Novel Design of High-Speed Multiplier Using Hybrid Adder Technique. In 2024 3rd International Conference for Innovation in Technology (INOCON) (pp. 1-5). IEEE.

[2] Nagaraj, S., Reddy, K. S., Varma, K. D., Sri, K. R., Likhitha, K., & Vamsi, K. (2024, September). Performance Analysis of Vedic Multiplier using Different Adders. In 2024 5th International Conference on Smart Electronics and Communication (ICOSEC) (pp. 158-167). IEEE.

[3] Ram, G. C., Subbarao, M. V., Varma, D. R., & Kumar, M. P. (2023, January). Delay enhancement of Wallace tree multiplier with binary to excess-1 converter. In 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 113-117). IEEE.

[4] Kumari, A., Kharwar, S., Kumar, A., & Singh, S. (2021, August). Design and Implementation of 12-bit Vedic Multiplier using Optimized Decoder-based Adder. In 2021 International Conference on Simulation, Automation & Smart Manufacturing (SASM) (pp. 1-5). IEEE.

[5] Kalyan, P. S., Reddy, P. R., Chakka, S. V., & Vinodhini, M. (2025, January). An Efficient Digital Vedic Multiplier with an Enhanced Adder. In 2025 8th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech) (pp. 1-5). IEEE.

[6] Ram, G. C., Rani, D. S., Balasaikesava, R., & Sindhuri, K. B. (2016, May). Design of delay efficient modified 16-bit Wallace multiplier. In 2016 IEEE international conference on recent trends in electronics, information & communication technology (RTEICT) (pp. 1887-1891). IEEE.

[7] Habibi, A., & Wintz, P. A. (2006). Fast multipliers. IEEE Transactions on Computers, 100(2), 153-157.

[8] Naik, A., Deka, D., & Pal, D. (2020, February). ASIC implementation of high-speed adaptive recursive karatsuba multiplier with square-root-carry-select-adder. In 2020 IEEE 11th Latin American Symposium on Circuits & Systems (LASCAS) (pp. 1-4). IEEE.

[9] Patel, D. K., Chouksey, R., & Saxena, M. (2016, February). Design of fast FIR filter using compressor and Carry Select Adder. In 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 460-465). IEEE.

[10] Sankanatti, S., & Praveen, S. (2021, July). Design and synthesis of Karatsuba multiplier using square root carry select adder (SRCSA). In 2021 6th International Conference on Communication and Electronics Systems (ICCES) (pp. 67-72). IEEE.

[11] Teja, D. S., Jagadeesh, P., & Jency, S. (2024, April). Implementation of High Speed and Low Area Novel 16-Bit Vedic Mathematic Algorithm Using Sklansky Adder in Comparison with Carry Select Adder. In 2024 Ninth International Conference on Science Technology Engineering and Mathematics (ICONSTEM) (pp. 1-6). IEEE.

[12] Swetha, S., & Shreya, T. (2025, March). Design and Analysis of 4-bit and 8-bit Vedic Multipliers Using Variable Bit CSLA. In 2025 3rd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT) (pp. 264-268). IEEE.

[13] Pavaiyarkarasi, R., Yashwini, T. S., & Udhayashankar, S. (2022, November). High speed and low power 8 bits-Dadda Multiplier using square root Carry Select Adder with Binary to Excess one Converter. In 2022 International Interdisciplinary Humanitarian Conference for Sustainability (IIHC) (pp. 503-509). IEEE.

[14] Prasad, Y. B., Chokkakula, G., Reddy, P. S., & Samhitha, N. R. (2014, February). Design of low power and high speed modified carry select adder for 16 bit Vedic Multiplier. In International Conference on Information Communication and Embedded Systems (ICICES2014) (pp. 1-6). IEEE.

[15] Paradhasaradhi, D., Prashanthi, M., & Vivek, N. (2014, March). Modified wallace tree multiplier using efficient square root carry select adder. In 2014 international conference on green computing communication and electrical engineering (ICGCCEE) (pp. 1-5). IEEE.

[16] Sari, L. K., & Adiono, T. (2019, July). Fully combinational 8× 8 bits multiplier using 130 nm technology. In 2019 International Conference on Electrical Engineering and Informatics (ICEEI) (pp. 201-205). IEEE.