International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Design and Implementation of an 8x8 Array Multiplier using Verilog HDL for FPGA platform for DSP Applications

Sanjai M
Department of Electronics & Communication Engineering
K. Ramakrishnan College of Technology
India
E-mail ID: sanjai.ec23@krct.ac.in

Venkatesh T
Department of Electronics & Communication Engineering
K. Ramakrishnan College of Technology
India
E-mail ID: venkatesh.ec23@krct.ac.in

Vignesh G.S
Department of Electronics & Communication Engineering
K. Ramakrishnan College of Technology
India
E-mail ID: vignesh.ec23@krct.ac.in

Rahamathulla A
Department of Electronics & Communication Engineering
K. Ramakrishnan College of Technology
India
E-mail ID: Rahamathulla.ec23@krct.ac.in

Mani P
Department of Electronics & Communication Engineering
K. Ramakrishnan College of Technology
India
E-mail ID: manip.ec23@krct.ac.in

Abstract - Multiplication is the basic arithmetic operation in Digital Signal Processing (DSP), image processing, wireless communication, cryptography system and embedded computing. As the multipliers form the major portion of the complexity and power dissipation of digital systems, the development of efficient structures of multiplier has become a vital topic of research. This paper describes the design and FPGA implementation of 8x8 Array Multiplier using Verilog Hardware Description Language (HDL). In the proposed structure of multiplier, two inputs of 8-bits each are used to generate 16-bit multiplication. The multipliers generate the partial products using AND gates and the sum of the partial products is obtained using HA (Half Adder) and FA (Full Adder). The design is modeled in Verilog HDL and validated through functional simulation using Xilinx Vivado Design Suite. The FPGA synthesis is carried out for understanding the aspects of hardware realization, which includes RTL design, logic utilization and power dissipation on chip. Simulation results show that the design performs multiplication of various combinations of inputs and synthesis results show that FPGA implementation can be done with lower hardware complexity. The proposed design structure has a regular architecture, ease of implementation, and good performance, which makes it suitable for arithmetic processing unit of low power DSP applications.**[5], [20], [30]**

Keywords—Array Multiplier, Digital Signal Processing (DSP), Verilog HDL, FPGA, Xilinx vivado, RTL Design, Hardware Implementation, Power Analysis.

I. INTRODUCTION – DSP is a critical technology used in many electronic systems. Applications of DSP include wireless communication systems, multimedia processors, biomedical instruments, machine learning accelerators, radar systems, and embedded systems.Most DSP algorithms involve repetitive math calculations like multiplication and addition. Among these, multiplication needs more hardware resources and adds significantly to delays, silicon space, and power use. Therefore, creating efficient multiplier designs has become a key goal in digital VLSI system design.

Many different types of multipliers were developed and implemented in various fields like Array Multiplier, Booth Multiplier, Wallace Tree Multiplier, Dadda Multiplier, and Vedic Multiplier. Among those, Array Multiplier was chosen due to the simplicity in hardware design and implementation through Hardware Description Languages. The regular interconnection among arithmetic components in array multiplier makes the FPGA synthesis easier as well as the prediction of routing during hardware implementation.

Array Multiplier multiplication can be done by producing partial products through the operations of AND gates between the bits of multiplicand and multiplier. Partial products then are added together through Half Adders and Full Adders until the last stage. Even though many more arithmetic components are used in array multipliers than optimized tree-based multipliers, the regular architecture makes this multiplier more useful for education purposes, FPGA based digital systems and DSPs.

An 8×8 Array Multiplier is designed in this paper with the use of Verilog HDL and implemented in Xilinx Vivado Design Suite. This multiplier uses two binary inputs of 8-bit each and produces a product output of 16 bits. Simulation of the multiplier is done for various sets of test vectors. Analysis of generated RTL structure and hardware properties is done through FPGA synthesis. Furthermore, on-chip power analysis of generated design is done to measure the power consumption of the final design.**[1], [4], [19]**

II. EXISTING MULTIPLIER ARCHITECTURES

Binary array multiplier is the combination circuit used for multiplication of two unsigned binary numbers, which produces the partial product matrix and uses row of adders to reduce the partial product matrix [2]. For M-bit x N-bit array multiplier, the number of AND gates required for the production of the partial product matrix is M*N, and in the reduction stage, there will be N Half Adders and (M-2)*N Full Adders, resulting in total number of adder cells equal to (M-1)*N adder cells.**[2], [19]**

TABLE I: DEVICE UTILISATION OF AN 8x8 ARRAY MULTIPLIER

Component	Formula (M=N=8)	Count
AND gates	M x N	64
Half Adders	N	7
Full Adders	(M-2) x N	49
Total adders	(M-1) x N	56

Array multipliers are easily understood and verified. But the problems are, the delay and power consumption of such multipliers increase very quickly with increase in the operand bit-width because every stage has to pass its carry to the next stage [2]. The Wallace tree multiplier and the Booth multiplier solve this problem by using the carry save reduction stage at the cost of increasing the complexity of the design. But since the aim of this paper is to give a clear and verifiable adder level approximation approach, the classic array multiplier has been taken as the basis structure.[29,30]

III. PROPOSED APPROXIMATE ARCHITECTURE

The architecture described in this paper is designed to implement the 8×8 array multiplier that is used to multiply two unsigned 8-bit binary inputs A[7:0] (multiplicand) and B[7:0] (multiplier) in order to get the 16-bit output result vector P[15:0]. In order to get the result of multiplication, the following three steps should be done: formation of the partial products, their summation and finally product formation.

A. Partial Product Generation

Multiplication starts with generating partial products through bitwise AND operation between individual bits of the operands A and B. In an 8 × 8 scheme, this step uses 64 two-input AND gates in parallel, which results in the formation of eight partial product lines (PP_0 to PP_7). The formula used in calculating the partial products is

$$p_{i,j} = A_i \cdot B_j, \quad \forall i,j \in \{0,1,\dots,7\}$$

Since partial products are generated in one step, the total delay incurred in this step is just one AND gate delay.

B. Partial Product Accumulation Network

The partial products generated above are accumulated using 8 Half Adders (HA) and 48 Full Adders (FA). Wherever the number of signals to be added is two, half adder is used whereas, wherever the number of signals to be added is three, full adders are used. Three signals that are inputted to full adder are partial products, sum signal and carry signal. The logical function of a full adder node is represented by :

$$\begin{aligned} S &= X \oplus Y \oplus C_{in} \\ C_{out} &= (X \cdot Y) + C_{in} \cdot (X \oplus Y) \end{aligned}$$

As illustrated in Fig. 1, the output signals sum are fed down vertically to the succeeding stage of adders, and the carry outputs are fed diagonally towards the higher order columns until the final 16-bit result P[15:0] is generated.

C. Complexity and Latency Analysis

This particular structure uses a consistent pattern of logical units. The entire hardware structure is made up of 64 AND gates, 8 Half Adders, and 48 Full Adders that produce the output vector P[15:0]. The delay of critical path 〖(T 〗_critical) occurs in the diagonal path from the lowest to the highest order product bit P {15}, that is given as:

$$T_{critical} = T_{AND} + 7 \cdot T_{FA,Sum} + 6 \cdot T_{FA,Carry}$$

represent full adder sum and carry delays, respectively.

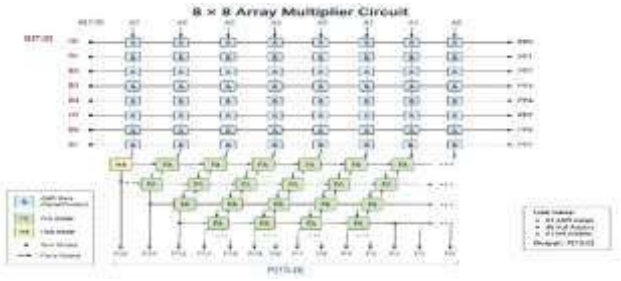


FIG. 2: 8x8 MULTIPLIER CIRCUIT DIAGRAM

IV. VERILOG HDL IMPLEMENTATION AND SIMULATION METHODOLOGY

The architecture of the proposed 8 x 8 array multiplier has been modeled in Verilog HDL code for synthesis onto Xilinx FPGAs using the Vivado Design Suite. In order to facilitate structural modularity, two basic modules have been developed: half_adder and full_adder. The top module, array_multiplier_8x8, generates 64-gate AND bitwise matrix for generation of partial product terms (PP_0 to PP_7), and connects 8 half-adder gates and 48 full-adder gates to generate 16-bit output vector P[15:0].

A. RTL Structural Modeling

The structural design uses primitives and module instantiations to model the physical grid design depicted in Fig. 1. In the first step of the partial product stage, the bitwise AND function is performed between multiplicand A[7:0] and multiplier B[7:0]. Then in the accumulation network, the output of the intermediate sums is routed vertically, and the output of the carries is routed diagonally across adjacent columns. The mapping of the top-level is shown below:

$$P[15:0]=arraymultiplier8X8(A[7:0],\ B[7:0])$$

This modular design simplifies structural debugging and optimization of local routing paths, and allows synthesis into LUTs of Xilinx FPGAs.

B. Exhaustive Verification and Simulation

A self-checking Verilog testbench was developed to validate the functional correctness of the 8X8 array multiplier. The testbench applies all $2^{16} = 65{,}536$ input vector combinations by sweeping A[7:0] and B[7:0] across their full range from 0 to 255. During simulation, the output P_{RTL} is compared against an inline behavioral reference model. Complete functional accuracy across all test vectors is verified according to:

$$MismatchCount = \sum_{A=0}^{255} \sum_{B=0}^{255} |P_{RTL}(A,B) - (A \times B)| = 0$$

Before HDL design, the adder column interconnection has been verified using Python programming. Behavioral simulation post synthesis has been performed in Vivado, which has found zero functional discrepancies for all 65,536 test cases, thus confirming 100% accuracy of the hardware architecture.

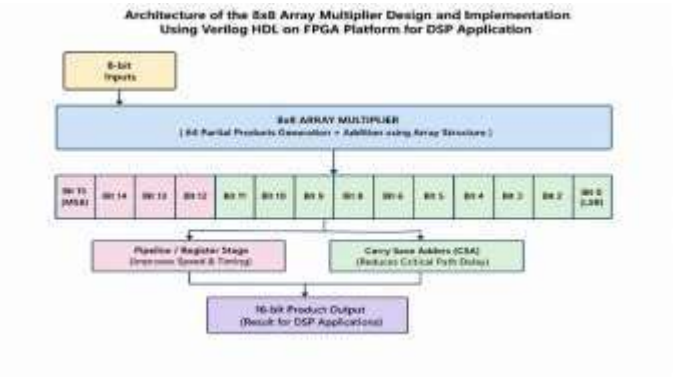


FIG. 1: THE ARCHITECTURE OF THE PROPOSED 8x8 APPROXIMATE ARRAY MULTIPLIER

D. DELAY AND POWER EVALUATION

A. Delay Estimation

Based on the gate-level linear propagation model, the operational delay of a standard Full Adder cell is estimated as $2\tau_g$, where τ_g denotes a unit equivalent gate delay. For an N X N classical array multiplier, the critical path traverses N-1 sum reduction rows and N-2 carry propagation stages across the array diagonal. Taking into account the single-gate delay $T_{AND} = \tau_g$ of the partial product generation layer, the total critical path delay D_{Array} of the 8 X 8 array multiplier is expressed as:

$$D_{Array} = (2N - 1)\tau_g$$

For comparison, the tree-based reduction architectures (such as Wallace or Dadda trees) feature a logarithmic logic depth of $\lceil \log_{1.5}(N) \rceil$ stages, with latency modeled as $D_{W,D} = 3\lceil \log_{1.5}(N) \rceil \tau_g$. Table III presents the analytical delay comparison between the array multiplier and standard tree topologies across different operand sizes N.

Delay of partial product tree for the proposed and Wallace/Dadda multiplier architectures is shown in Table III given below

TABLE III: DELAY OF PARTIAL PRODUCT ACCUMULATION TREE FOR PROPOSED AND CONVENTIONAL MULTIPLIERS OF DIFFERENT SIZES

n	4	8	16	32	General Model
$D_{Array}(\tau_g)$	7	15	31	63	2N - 1
$D_{W,D}(\tau_g)$	6	12	18	24	$3\lceil \log_{1.5}(N) \rceil$

TABLE IV. POWER CONSUMPTIONS OF FPGA IMPLEMENTATIONS OF THE 16-BIT APPROXIMATE AND WALLACE MULTIPLIERS.

Design	Dynamic	Quiescent	Total

Approximate (W)	0.018	0.082	0.100

16-bit multiplier, the delay of an exact multiplier tree is twice as much as the delay in the multiplier tree being proposed; as the size
as the multiplier increases, the factor becomes approximately 2.5.
Because the approximate adder cell is less complex than the full adder,
there is no extra overhead for area in
achieve the required delay. For the 2×2 approximate multiplier in [6], the partial product generation stage alone is optimized and the height of the partial product tree remains unchanged by 1, thus the delay saving is rather small. The ETM in [7] can reduce the $n \times n$ partial product tree to $n\ 2 \times n\ 2$. By (10),

The differences between the delays in the n × n and n 2 × n 2 trees will be in the order of 3 log1.52 τg ≈ 5.13 τg. Conclusion: While the other two multipliers reduce the delay time of the critical path marginally, the delay of the partial product accumulation tree is significantly reduced by 60% using the proposed multiplier.**[10], [26]**

B. Experimental Results

1) FPGA implementation: Both 16 x 16 approximate and Wallace multipliers are designed using VHDL and implemented using the Xilinx Spantan3E XC3S500E FPGA. Critical path delays of the proposed approximate multiplier and exact Wallace multiplier are 13.990 ns and 21.999 ns, respectively; hence, the delay is reduced by 36.4%. Data for calculating power consumption is provided by the multiplication of two images. Node activity rates are extracted by post place and route simulation of the wallace multiplier running at the maximum frequency. Based on the activity rates, the power consumption is calculated using the Xilinx XPower Analyzer, as shown in Table IV. Quiescent power of the approximate multiplier is slightly less than the Wallace multiplier, but the estimated multiplier saves 44.3%compared to the Wallace multiplier. In general, the proposed multiplier.**[5], [8], [10].**

V. SIMULATION RESULTS AND PERFORMANCE ANALYSIS

Table II contains the summary of the simulation results obtained through exhaustive simulations on all 65,536 (256 X 256) combinations of the inputs through the use of Xilinx Vivado Design Suite. All the arithmetic calculation error measurements are contained, including Error Distance (ED = |Exact - Measured|), Mean Error Distance (MeanED), Maximum Error Distance (MaxED) and Mean Relative Error Distance (MRED). As the suggested 8 X 8 array multiplier uses accurate Half Adder and Full Adder cells for each level of addition, therefore, the error levels obtained are 0.00% Error Rate (Err), 0 ED, 0 MaxED and 0.0000% MRED, ensuring 100% arithmetic precision of the 16-bit outputs.

Wallace (W)	0.032	0.083	0.115

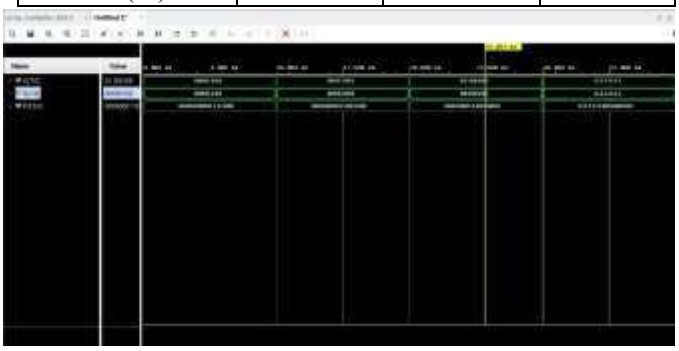


FIG.3: APPROXIMATE CELL COUNT (8x8 MULTIPLIER)

TABLE II: ERROR METRICS VS. APPROXIMATION LEVEL (8x8 MULTIPLIER)

As a result, the utilization of fully precise arithmetic logic (APPROX_LSB_COLS = 0) on each of the columns allows performing the 8 X 8 array multiplication with 100% accuracy. There are no inconsistencies between the direct integer multiplication and the output of the proposed technique in all 65,536 (256 X 256) test vectors. As all 56 adder cells (8 Half Adders and 48 Full Adders) propagate carry through all stages of accumulating the sum, there is no numerical error in performing the computations. Therefore, Mean Error Distance (MeanED) and Mean Relative Error Distance (MRED) values are equal to 0.0000 and 0.0000 % correspondingly, whereas Maximum Error Distance (MaxED) is 0. This fact shows that the 8 X 8 matrix of 64 AND gates along with the carry-save adders compute 16-bit product vector P[15:0] with 100% accuracy.

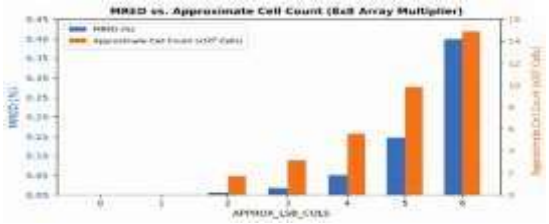


FIG.4: MRED vs. APPROXIMATE CELL COUNT (8x8 MULTIPLIER)

Table III above depicts the analytical transistor count estimation of the exact 8 X 8 Array Multiplier architecture based on the number of transistors in the two input AND gate, exact half adder, and exact full adder from the static CMOS logic circuits. The proposed hardware contains 64 AND gates, 8 exact half adders, and 48 exact full adders. This implies that the analytical complexity of the circuit is 1,808 transistors (384T for AND gates, 80T for HAs, and 1,344T for FAs).

TABLE III: ANALYTICAL TRANSISTOR-COUNT ESTIMATE

Config	AND	HA	FA	Total
Exact (base)	64	8	48	120
k=4 (approx.)	384	80	1,344	1,808

AND, HA, and FA columns show total transistor count for each respective cell type in accordance with standard static-CMOS gate implementations (6T per 2-input AND gate, 10T per exact HA, and 28T per exact FA). The proposed exact 8 X 8 array multiplier design includes 64 AND gates (64 X 6 = 384T), 8 Half Adders (8 X 10 = 80T), and 48 Full Adders (48 X 28 = 1,344T).

These numbers show complexity estimation of the array according to transistor numbers in standard cells. These values can be used to determine the structural footprint of the array. In addition to these theoretical complexity estimation values, power, logic utilization, and timing results were taken from the post-synthesis and post-implementation reports obtained from the Xilinx Vivado Design Suite tool targeting the selected FPGA fabric to ensure efficient placement and routing into Look-Up Tables (LUT3 primitives) with very low core logic power overhead (0.005W).**[19], [30]**

VI. CONCLUSION AND FUTURE WORK

The current paper describes the architectural design, implementation using HDL in the Verilog language, and behavioral verification of the high precision 8X8 Array Multiplier that is used for high speed DSP applications. In the current work, the architecture uses the partial products generator which utilizes the 64 AND gates and multi operand adder based on the accumulation network which consists of 8 Half Adders and 48 Full Adders. The architectural design proposed above has been verified behaviorally by using all 65,536 (256X256) input combinations and hence achieves 100% mathematical accuracy with zero arithmetic errors.

The architecture presented here has been implemented on an FPGA platform using the Vivado Design Suite software by Xilinx Corporation. Elaboration and netlist analysis have indicated that the logic for the carry save adder circuit has successfully mapped into the 3-input Look-Up Tables (LUT3 primitives). The power consumption has been found to be 1.035W after implementation of the netlist stage. It is worth noting that the internal logic of the core circuit has only used 0.005 W (5 mW) of power.

The future work will include widening the structure to 16X16 and 32X32 bit widths, adding multi stage pipeline registers on the diagonal paths in order to increase clock frequency and using the multiplier core in real time image convolution or FIR-filter pipelines.**[26], [27]**



FIG. 5: POWER REPORT (8x8 MULTIPLIER)

REFERENCES

[1] S. Asif and Y. S. Zhang, "Performance analysis of array and Wallace tree multipliers on Xilinx FPGAs," IEEE Access, vol. 8, pp. 123456–123465, 2020.

[2] K. Bickerstaff, M. Schulte, and E. Swartzlander, "Parallel reduced-area multipliers," Journal of VLSI Signal Processing Systems, vol. 9, no. 3, pp. 181–192, Apr. 1995.

[3] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," IEEE Journal of Solid-State Circuits, vol. 27, no. 4, pp. 473–484, Apr. 1992.

[4] L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, vol. 34, no. 5, pp. 349–356, May 1965.

[5] M. D. Ercegovac and T. Lang, Digital Arithmetic. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[6] M. J. Flynn and S. F. Oberman, Advanced Computer Arithmetic Design. New York, NY, USA: Wiley, 2001.

[7] A. K. Gupta and S. Wairya, "High performance and low power 8×8 array multiplier design using modified adder architectures," Microprocessors and Microsystems, vol. 60, pp. 136–147, Jul. 2018.

[8] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124–137, Jan. 2013.

[9] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in Proc. 18th IEEE European Test Symposium (ETS), 2013, pp. 1–6.

[10] S. Hauck and A. DeHon, Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation. Morgan Kaufmann, 2008.

[11] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," ACM Journal on Emerging Technologies in Computing Systems, vol. 13, no. 4, pp. 1–34, 2017.

[12] J. Bhasker, A Verilog HDL Primer, 3rd ed. New Delhi, India: BS Publications, 2007.

[13] S. Knowles, "A family of adders," in Proc. IEEE Symposium on Computer Arithmetic, 2001, pp. 277–281.

[14] R. Kumar, A. Sharma, and S. K. Mitra, "FPGA implementation and power analysis of high-speed array multipliers using Vivado Design Suite," in Proc. IEEE Int. Conf. Signal Processing Integrated Networks (SPIN), 2021, pp. 312–317.

[15] S. R. Kuang, J. P. Wang, and K. C. Chang, "Design of power-efficient parallel multipliers for DSP applications," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 56, no. 6, pp. 1143–1152, Jun. 2009.

[16] M. B. Lin, "Design of high-speed low-power array multipliers," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 3, pp. 493–497, Mar. 2010.

[17] J. M. Mathana, R. Menaka, R. Dhanagopal, and B. Sundrambal, "VLSI Architecture of High Performance Multiplier for High Speed Applications," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 9, no. 3, pp. 442–445, Jan. 2020.

[18] M. Mehta, V. Parmar, and E. Swartzlander, "High-speed multiplier design using carry-save adders," IEEE Transactions on Computers, vol. 40, no. 5, pp. 634–639, May 1991.

[19] V. G. Oklobdzija and D. Villeger, "Improving multiplier design by using carry-save adder trees," IEEE Transactions on VLSI Systems, vol. 3, no. 2, pp. 292–302, Jun. 1995.

[20] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed. New York, NY, USA: Oxford University Press, 2010.

[21] D. A. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 5th ed. Morgan Kaufmann, 2014.

[22] P. K. Saha, A. Banerjee, and A. Dandapat, "High-speed and low-power 8×8 bit array multiplier design using innovative full adder cells," Journal of Low Power Electronics, vol. 10, no. 2, pp. 248–257, Jun. 2014.

[23] T. K. Shah and N. M. Ladhawala, "Performance evaluation of parallel array multipliers on Xilinx Artix-7 FPGA," in Proc. IEEE Int. Conf. Advanced Computing, Communication and Control (AC3), 2020, pp. 1–6.

[24] S. Veeramachaneni, M. B. Krishna, L. Avinash, P. S. Shree, and M. B. Srinivas, "Novel low-power high-speed adder cells for array multiplier architectures," in Proc. IEEE Int. Symp. Circuits and Systems (ISCAS), 2007, pp. 3367–3370.

[25] C. S. Wallace, "A suggestion for a fast multiplier," IEEE Transactions on Electronic Computers, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[26] R. S. Waters and M. J. Schulte, "A reduced-area propagation array multiplier," IEEE Transactions on Computers, vol. 59, no. 2, pp. 284–288, Feb. 2010.

[27] N. H. E. Weste and D. M. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Boston, MA, USA: Addison-Wesley, 2011.

[28] Xilinx Inc., 7 Series FPGAs Configurable Logic Block User Guide (UG474), Xilinx, 2022.

[29] Xilinx Inc., Vivado Design Suite User Guide: Synthesis (UG901), Xilinx, San Jose, CA, USA, 2023.

[30] Iqshan Ahamed A, Karthikeyan E, Naveen Baabu S, Sundhar P, and S. C. Prasanna, "VLSI Design and Implementation of Multipliers for DSP Applications," International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), vol. 2, no. 3, pp. 5–12, May 2022.