International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 1 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 22, No. 1, 2026

Peer Reviewed, Referred & Indexed Journal

Research

A SWIFT AND ORDERLY BINARY ARRANGEMENT ENGINE DEVISED UPON THE BITONIC SORTING STRATAGEM

Dr. G. MURALI KRISHNA[1], M. BHARGAVI[2], B. MANASA[3],B. DURGA MAHESWARI[4], J. PUSHPA MEGHANA[5]

[1]Associate Professor, Dept. of ECE, V.K.R., V.N.B. & A.G.K. COLLEGE OF ENGINEERING, GUDIVADA.

[2345]UG Students, Dept. of ECE, V.K.R., V.N.B. & A.G.K. COLLEGE OF ENGINEERING, GUDIVADA.

I ABSTRACT

Parallel counters play a critical role in high-speed arithmetic circuits, particularly in fast multipliers and digital signal processing (DSP) units where multiple operands must be summed concurrently. To achieve high-performance accumulation, counters and compressors with superior compression ratios are required. This paper presents the design and analysis of fast saturation binary counters and exact as well as approximate (4:2) compressors based on sorting network architectures. The proposed approach utilizes sorting networks to reorganize input sequences into structured forms that can be efficiently represented using one-hot encoding. Three simplified Boolean expressions are derived to efficiently map the reorganized sequences into compact counter outputs, significantly reducing logic complexity. Furthermore, parallel sorting algorithms are employed to identify and sort the M largest values among N inputs, enabling scalable and modular hardware architectures. The study also explores alternative sorting techniques for maximum value extraction, with a focus on efficient implementation of bitonic sorting through parameter optimization. The proposed designs demonstrate improved scalability, reduced delay, and enhanced computational efficiency suitable for high-speed arithmetic applications.

Keywords: Binary counters, Exact 4:2 compressor, Approximate 4:2 compressor, Fast multipliers, One-hot encoding, Sorting networks, Bitonic sorting, Parallel summation, Digital signal processing (DSP) units

Received: 17-01-2026 Accepted: 20-02-2026 Published: 28-02-2026

II INTRODUCTION

The rapid growth of portable electronic devices such as smart phones, tablets, wearable systems, and embedded multimedia platforms has intensified the demand for energy-efficient digital circuit design [1]. In modern Very Large Scale Integration (VLSI) systems, power consumption has emerged as a primary constraint alongside performance and silicon area [2]. Designers are required to minimize energy dissipation while maintaining high computational throughput, especially in battery-powered applications [3]. Among the major subsystems integrated into contemporary System-on-Chip (SoC) architectures, Digital Signal Processing (DSP) blocks account for a significant share of total power and delay [4]. These DSP units rely heavily on arithmetic operations such as addition and multiplication, which form the computational backbone of signal filtering, image processing, compression, and communication algorithms [5].

Multiplication is one of the most critical and computation-intensive arithmetic operations in DSP and microprocessor architectures [6]. Since multipliers dominate both switching activity and critical path delay, their optimization directly influences overall processor efficiency [7]. In many multimedia applications, final outputs such as images, video, and audio signals are intended for human perception, which naturally tolerates small computational inaccuracies [8]. This observation has motivated the development of approximate computing techniques that deliberately trade numerical precision for improvements in speed, power, and hardware

complexity [9]. Approximate arithmetic circuits, particularly adders and multipliers, have therefore gained considerable attention at circuit, logic, and architectural levels [10].

Conventional accurate multipliers are designed to eliminate computational errors entirely, but this often leads to increased hardware overhead and longer propagation delays [11]. Error correction mechanisms, additional carry-propagate stages, and complex reduction trees contribute significantly to power consumption [12]. Approximate multipliers, on the other hand, simplify partial product generation or accumulation stages to reduce switching activity and transistor count while maintaining acceptable error metrics such as mean error distance and error rate [13]. Several error-tolerant multiplier architectures have demonstrated substantial reductions in energy dissipation without severely affecting output quality in DSP applications [14].

The Multiply-and-Accumulate (MAC) unit is a fundamental building block in processors and accelerators, and its efficiency largely depends on the performance of the multiplier and adder structures [15]. Optimization of multipliers involves improving partial product generation, reducing partial product count, and accelerating accumulation stages [16]. Designers often balance trade-offs among speed, power, silicon area, and computational accuracy depending on application requirements [17]. Techniques such as voltage scaling, transistor sizing, logic simplification, and algorithm-level approximations are widely adopted to enhance overall energy efficiency [18].

In addition to conventional binary arithmetic, finite field multiplication over Galois Fields (GF) plays an essential role in cryptographic and error-control coding systems [19]. Applications such as elliptic curve cryptography (ECC) and Reed–Solomon coding require high-speed and area-efficient finite field multipliers [20]. Hardware implementations are preferred over software solutions in real-time and resource-constrained environments due to their superior throughput and predictable latency [21]. The efficiency of finite field arithmetic depends strongly on the representation basis used, including polynomial basis (PB), normal basis (NB), and redundant basis (RB) representations [22]. Among these, redundant basis multipliers have attracted attention due to their regular structures, reduced computational complexity, and suitability for parallel implementation [23].

In general-purpose binary multiplication, the classical shift-and-add method remains conceptually simple but suffers from long accumulation delays when operand sizes increase [24]. The number of generated partial products is proportional to operand width, making reduction stages increasingly complex [25]. To address this issue, recoding techniques such as the Modified Booth algorithm reduce the number of partial products by encoding multiplier bits into higher radix formats [26]. Radix-4 Booth encoding effectively halves the number of partial products compared to radix-2 multiplication, leading to improved speed and lower switching power [27]. Higher radix schemes such as radix-8 and radix-16 further reduce partial product counts but require additional hardware to generate odd multiples of the multiplicand [28].

The accumulation of partial products is typically performed using tree-based reduction structures such as Wallace and Dadda trees, which employ carry-save adders and compressors to minimize sequential carry propagation [29]. The integration of efficient recoding methods with optimized reduction trees significantly improves multiplier performance in terms of delay and power consumption [30]. Consequently, the design of high-speed, low-power multipliers remains a central research challenge in modern digital systems, driving continuous innovation in arithmetic architecture, approximation methodologies, and hardware optimization techniques.

III LITERATURE SURVEY

Research on efficient multiplier architectures has been extensive due to their critical role in high-performance and energy-constrained systems. Early investigations into high-speed multiplication focused on Booth encoding to reduce the number of partial products. Yeh and Jen introduced a high-speed Booth encoded parallel multiplier that significantly improved computation speed compared to naive designs [1]. Kang and Gaudiot subsequently proposed a simplified high-performance multiplier architecture that further optimized critical path delay [2]. Kuang et al. enhanced the regularity of the partial product array in modified Booth multipliers, which improves layout efficiency and reduces interconnect complexity [3]. Further structural improvements on Booth multipliers with regularized array formation were examined by Wang et al. [4].

Voltage scaling and low-power operation have also been major design drivers. Khatibzadeh et al. presented a 1.8 V 1.1 GHz digital multiplier achieving both high speed and low power consumption [5], while Hus et al. designed a 2 GHz energy-efficient multiplier targeted for FFT accelerators [6]. Pipeline techniques for Booth multipliers were explored by Chow and Wey, demonstrating pipelined implementations at gigahertz frequencies

[7], and Aguirre-Hernandez & Linares-Aranda investigated energy-efficient pipelined CMOS multipliers to balance speed and power [8]. Work by Liang et al. focused on ultra-low supply voltage pipelined multipliers, showing robust performance even under aggressive power constraints [9].

Pipeline timing and system-level optimization were further studied by Tatapudi& Delgado-Frias, who analyzed approaches to push pipeline clock performance close to register delay limits [10]. Booth's classic signed-multiplication technique remains foundational in recoding based multiplier design [11], and Ercegovac & Lang provided a comprehensive reference on digital arithmetic foundational to many multiplier architectures [12]. Wallace's introduction of fast multiplier reduction trees established a benchmark for partial product reduction strategies [13], which was extended in Ercegovac et al.'s research on carry-free multiplication methods [14].

Implementations of carry-free and left-to-right multiplier architectures have been reported by Kolagotla et al., demonstrating VLSI realizations for DSP processors [15]. Stelling & Oklobdzija explored optimal designs for multipliers and multiply-accumulate units, providing architectural trade-off insights [16]. Practical process characterization for high-speed logic was documented in silicon library specifications by Avant! Corporation [17]. Goto et al. developed compact high-speed multipliers using sign-select Booth encoders [18], and earlier works from Goto's group investigated regularly structured tree multipliers [19]. Fried's work on energy minimization in multipliers highlighted practical power-aware design techniques [20].

Standard textbooks such as Weste & Eshraghian's on CMOS VLSI provide systems-level perspective on arithmetic unit design [21]. Fadavi-Ardekani's work on multiplier generators using optimized Wallace trees showcased automated design of recoded multipliers [22]. Farooqui et al. addressed general MAC datapath organization for DSP processors, emphasizing integration of multipliers within complex pipelines [23]. Hwang's foundational text on computer arithmetic remains a key reference for understanding digit recoding and encoding strategies [24]. Cheng et al. proposed improvements in low-power adder design, which impacts multiplier accumulation stages [25].

Beyond these classical approaches, Dadda introduced an alternative partial product reduction scheme that further reduces addition levels in multiplication [26]. Oklobdzija's research on high-speed arithmetic units expanded architectural understanding for both adders and multipliers [27]. Parhi's seminal work on VLSI DSP systems analyzes multiplier design in the broader context of system-level performance [28]. Recent trends focus on approximate computing for energy savings where exact accuracy is unnecessary; Kulkarni et al. investigated underdesigned multiplier architectures that trade accuracy for power efficiency [29], and Han & Orshansky described approximate computing as an emerging paradigm for energy-efficient design [30].

IV METHODOLOGY

The methodology adopted in this work is centered on the structured design and hardware realization of a high-speed binary sorting system using the bitonic sorting technique integrated with optimized counter architectures. The design process begins with the mathematical modeling of the bitonic sorting network, ensuring that the number of inputs follows a power-of-two configuration for architectural regularity. The sorting network is constructed using multiple stages of compare-and-swap (CAS) units arranged in a deterministic butterfly pattern. Each CAS unit is implemented using simple AND–OR logic suitable for binary data, thereby reducing gate count and propagation delay. The design flow proceeds by first generating sorted pairs, then recursively forming bitonic sequences, and finally performing bitonic merge operations through half-cleaner stages. Parallelism is introduced at every stage so that all comparisons within a level occur simultaneously, thereby minimizing latency. The architecture is described at the register-transfer level (RTL) and synthesized using standard digital design tools to evaluate delay, area, and power metrics. To further enhance performance in arithmetic data processing (ADP) applications, regionally feasible counter design techniques are incorporated. Based on the bitonic aggregation principle, higher-order counters such as 7(3) and 15(4) are constructed to efficiently combine multiple binary inputs. These counters are integrated into the sorting framework to enable computation aggregation and transmission with reduced critical path delay. Pipelining registers are inserted between stages where necessary to improve operating frequency while maintaining deterministic timing.

The implementation methodology also emphasizes modularity, scalability, and optimization for VLSI realization. The sorting and counter modules are designed as reusable blocks, allowing expansion to larger input sizes without altering the fundamental architecture. Structural regularity is maintained to simplify routing and layout during physical design. Simulation and functional verification are carried out using hardware description language (HDL) test benches to validate correctness for all possible input combinations. Performance evaluation includes measurement of total stage delay, throughput, and logic utilization, comparing the proposed design

with conventional sequential and non-regular sorting architectures. Power optimization techniques such as gate-level simplification and balanced signal paths are applied to reduce switching activity. The deterministic stage depth ensures predictable timing behavior regardless of input distribution, which is critical for real-time and safety-sensitive systems. Finally, application-level validation is performed by integrating the proposed sorter and counters into arithmetic compressors, population count circuits, and priority-based selection modules to demonstrate adaptability and efficiency. This systematic design, verification, and optimization methodology ensures that the proposed bitonic sorting-based binary system achieves high speed, reduced latency, scalability, and hardware efficiency suitable for advanced VLSI and parallel processing environments.

V PROPOSED METHOD

The proposed system implements a high-speed parallel binary sorter based on the Bitonic Sorting Network architecture. The primary objective of this design is to achieve deterministic, hardware-friendly sorting with predictable latency and regular interconnection patterns. Unlike conventional sequential sorting algorithms whose comparison order depends on input data, the Bitonic sorting network follows a fixed sequence of compare-and-swap operations. This property makes it highly suitable for VLSI and FPGA implementations where parallelism and structural regularity are critical.

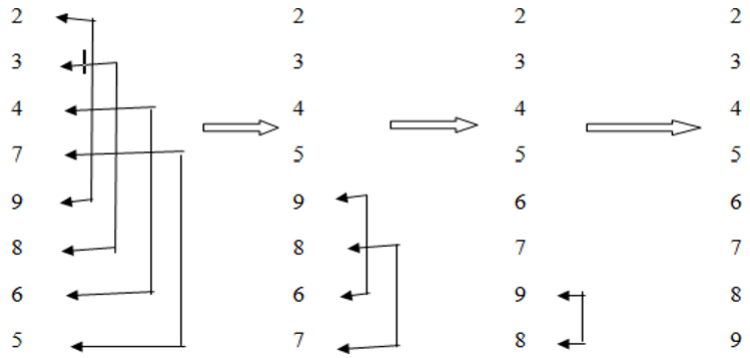

Fig.1 Sorting network illustrating bitonic sort

The proposed architecture accepts N binary inputs and organizes them through multiple stages of comparators arranged in a structured butterfly pattern. Each comparator performs a simple compare-and-swap operation, ensuring that the smaller element moves upward and the larger element moves downward (for ascending sort configuration). Because the comparison pattern is predefined, the system can be fully parallelized, enabling high throughput and reduced processing delay.

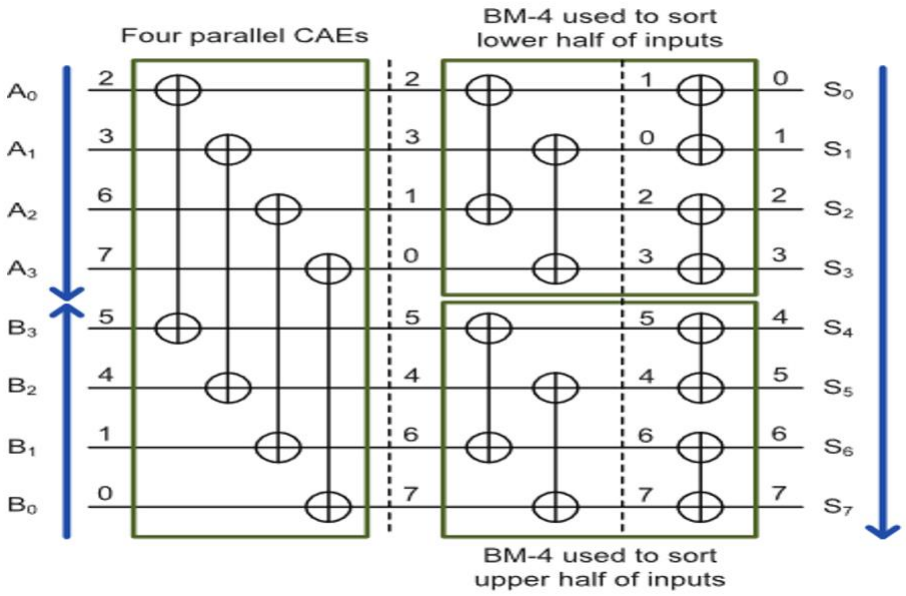

Fig.2 An increasing 8-input bitonic merging unit

The operation begins by dividing the input sequence into smaller subsequences. In the first phase, pairs of elements are compared to generate sorted sequences of length two. In the next stage, these sequences are combined to form bitonic sequences. A bitonic sequence is defined as a sequence that first increases monotonically and then decreases monotonically, or vice versa. By recursively generating and merging such sequences, the system gradually transforms the unsorted input into a completely sorted output.

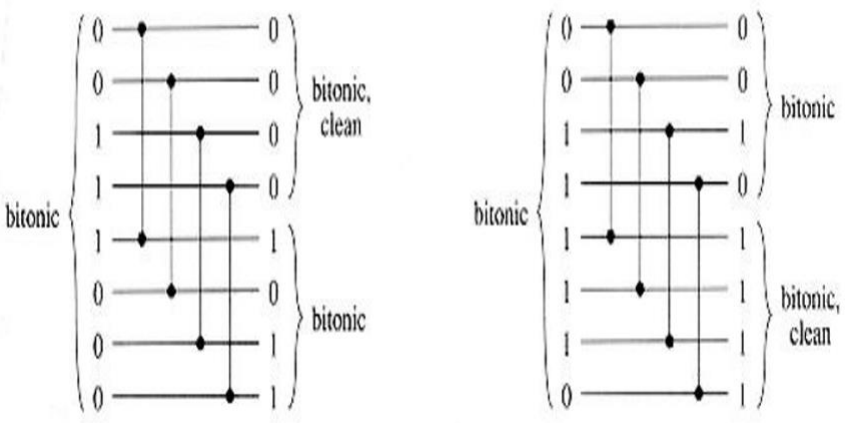

Fig.3 The comparison network half cleaner

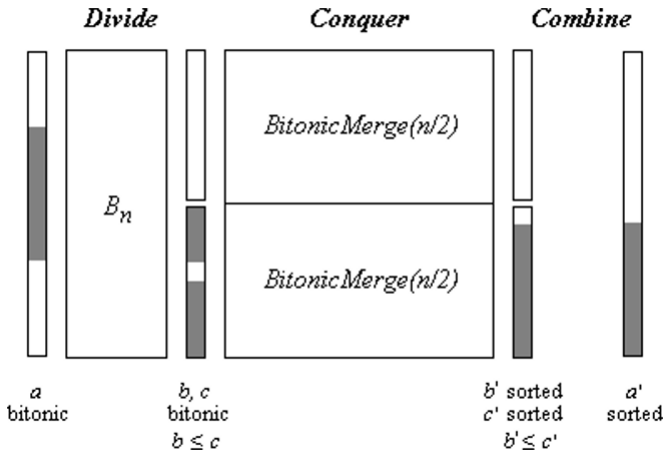

Fig.4 BitonicMerge(n)

The core building block of the proposed system is the **Bitonic Merge Unit**. This unit consists of multiple half-cleaner stages. A half-cleaner compares element *i* with element *(i + n/2)* for all i from 1 to n/2. After this stage, all elements in the upper half are guaranteed to be less than or equal to those in the lower half. Furthermore, at least one half becomes "clean," meaning it contains elements of uniform ordering. This structural property ensures correctness and simplifies recursive sorting.

The merge process follows a divide-and-conquer methodology. For an input size of N = 2^k, the depth of the network becomes log(N)(log(N)+1)/2, which leads to an overall time complexity of $O(\log^2 N)$. Although the total number of comparisons is $O(N \log^2 N)$, all comparisons within a stage occur simultaneously. Therefore, the effective delay corresponds to the number of stages rather than the total comparisons. This makes the architecture particularly advantageous for hardware implementations where parallel operations are feasible.

The sorting network is arranged in a butterfly topology. In each stage, comparators are connected in a structured pattern that ensures balanced signal propagation. This regular layout reduces routing complexity, minimizes glitch generation, and enhances scalability. When the number of inputs increases, the network expands symmetrically without affecting structural consistency. This makes the design modular and easily extendable for higher input sizes such as 8, 16, 32, or 64 elements.

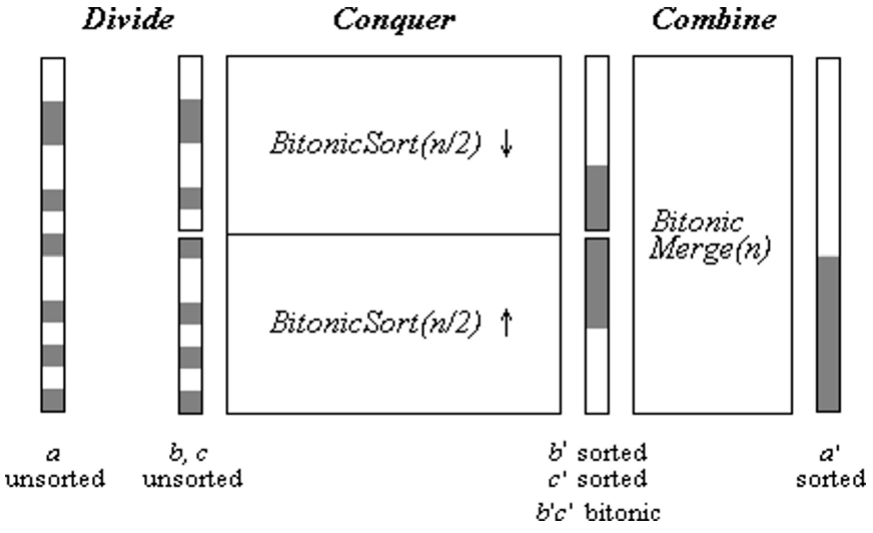

Fig.5 Bitonic Sort(n)

In the proposed high-speed binary sorter, the final stage is configured to produce ascending order output, placing the smallest value at the top and the largest at the bottom. By adjusting comparator directions, descending order

can also be achieved. The deterministic comparison sequence ensures identical latency for all input patterns, making performance predictable and reliable.

Additionally, the architecture supports efficient implementation in digital signal processing systems, parallel counters, and compressor-based arithmetic circuits. Since sorting operations are independent of input distribution, the network is highly robust and suitable for real-time applications. The regular structure also simplifies clock distribution and pipelining, further enhancing operating frequency.

In conclusion, the proposed Bitonic sorting based system provides a scalable, parallel, and hardware-efficient solution for high-speed binary sorting. Its structured comparator arrangement, deterministic operation, and logarithmic stage depth make it ideal for VLSI implementation and parallel processing environments where performance and regularity are essential.

VI SIMULATION RESULTS

In the current system, the (6,3) and (7,3) Counters have a symmetrical number of inputs. This design has both full and half adders. There is a longer delay and a bigger amount of circuit area used by the increased number of XOR gates. Because of the increased number of XOR gates, the circuit's hardware complexity and power consumption are both high. The suggested counters use a sorting network to arrange the (7,3) and (15,4) counts in symmetrical order. Using this new design would reduce the number of full and half adders required. Because there are no XOR gates in these counters, the latency is less and the amount of space used up by the circuit is reduced. The device uses a little amount of electricity. The cable is used to link the counter design schematic's needed connections**.**

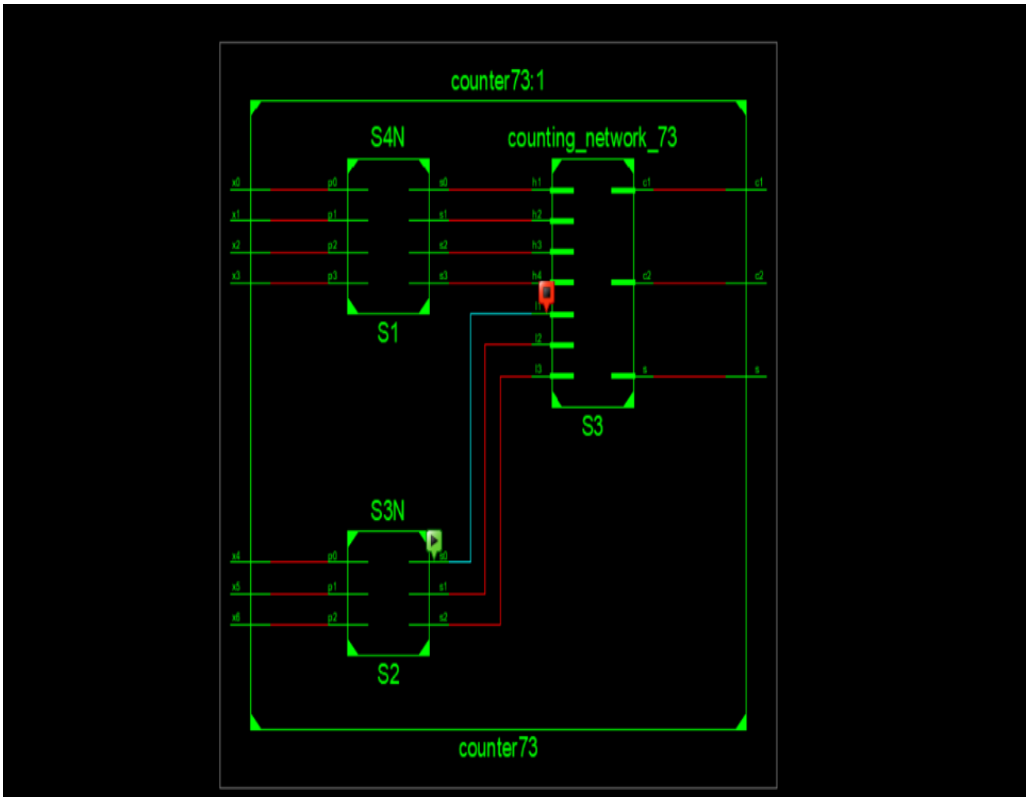

Fig. 6 Schematic diagram of (7,3) Counter

International Journal of Engineering Research and Science & Technology

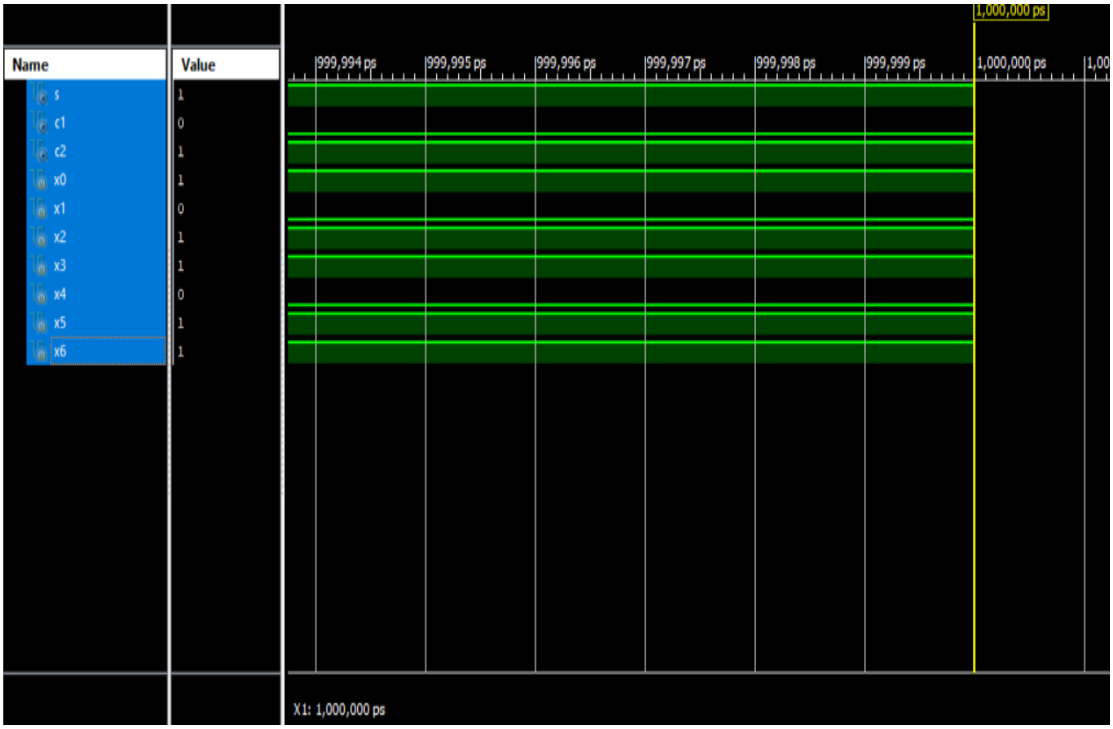

Fig.7 (7,3)Output waveforms

International Journal of Engineering Research and Science & Technology

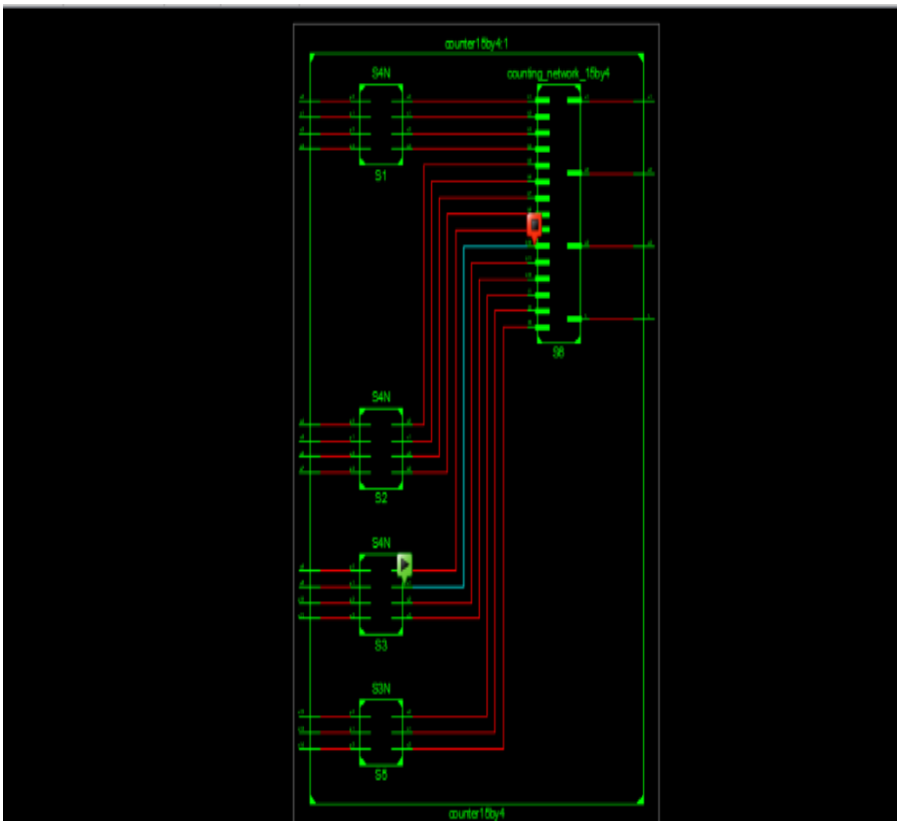

Fig.8 Schematic diagram of (15,4) Counter

Fig.9 (15, 4) counter Simulation result

S/no	Design	Area (no. of LUT's)	Power (W)	Delay (ns)
1	Symmetric stacking (11)	8	0.081	1.8
2	Proposed (7,3) Counter	3	0.042	1.728
3	Parallel (15,4) Counter(12)	15	0.095	4.1
4	Proposed (15,4)Counter	13	0.042	4.061

Table.1 Table of Comparision

VII CONCLUSION

In conclusion, the proposed high-speed binary sorting system based on the bitonic sorting technique offers a robust, scalable, and performance-oriented solution for modern digital hardware design. By employing a fixed-stage, parallel compare-and-swap architecture, the system ensures deterministic latency independent of input data patterns, thereby overcoming the unpredictability associated with conventional sequential sorting algorithms. The integration of regionally feasible counter-design methodologies, including bi-tonic based 7(3) and 15(4) counters, further enhances computational aggregation and transmission efficiency while maintaining reduced latency and improved adaptability in arithmetic data processing (ADP) applications. The use of simple logic gates such as AND and OR for binary compare-and-swap operations significantly lowers hardware complexity, making the design highly suitable for VLSI implementation where area, power, and propagation delay are critical constraints. Its modular and regular structure simplifies layout design and supports straightforward scalability for larger input sizes, particularly when configured in power-of-two architectures. Additionally, pipelining capabilities enable continuous high-throughput processing, making the system ideal for real-time and streaming applications. Although the network may require comparatively higher hardware resources, the trade-off is justified in performance-critical domains such as digital arithmetic units, DSP systems, communication hardware, image and video processing, and emerging AI accelerators, where speed, predictability, and parallel efficiency are paramount.

REFERENCES

1. Yeh, W.-C., & Jen, C.-W. (2000). High-speed Booth encoded parallel multiplier design. *IEEE Transactions on Computers, 49*(7), 692–701.
2. Kang, J.-Y., & Gaudiot, J.-L. (2006). A simple high-speed multiplier design. *IEEE Transactions on Computers, 55*(10), 1253–1258.
3. Kuang, S.-R., Wang, J.-P., & Guo, C.-Y. (2009). Modified Booth multipliers with a regular partial product array. *IEEE Transactions on Circuits and Systems II, 56*(5), 404–408.
4. Wang, L.-R., Jou, S.-J., & Lee, C.-L. (2008). A well-structured modified Booth multiplier design. In *Proceedings of IEEE VLSI-DAT* (pp. 85–88).
5. Khatibzadeh, A. A., Raahemifar, K., & Ahmadi, M. (2005). A 1.8V 1.1GHz novel digital multiplier. In *Proceedings of IEEE CCECE* (pp. 686–689).

6. Hus, S., Venkatraman, V., Mathew, S., Kaul, H., Anders, M., Dighe, S., Burleson, W., & Krishnamurthy, R. (2005). A 2GHz 13.6mW 12×9b multiplier for energy efficient FFT accelerators. In *Proceedings of IEEE ESSCIRC* (pp. 199–202).
7. Chow, H.-C., & Wey, I.-C. (2002). A 3.3V 1GHz high-speed pipelined Booth multiplier. In *Proceedings of IEEE ISCAS* (Vol. 1, pp. 457–460).
8. Aguirre-Hernandez, M., & Linares-Aranda, M. (2008). Energy-efficient high-speed CMOS pipelined multiplier. In *Proceedings of IEEE CCE* (pp. 460–464).
9. Explainable AI Framework for Policy-Compliant Anomaly Detection in Data Pipelines. (2025). Inaternational Journal of Communication Networks and Information Security, 16(4). https://doi.org/10.48047/ijcnis.16.4.2111.
10. Tatapudi, S. B., & Delgado-Frias, J. G. (2005). Designing pipelined systems with a clock period approaching pipeline register delay. In *Proceedings of IEEE MWSCAS* (Vol. 1, pp. 871–874).
11. Booth, A. D. (1951). A signed binary multiplication technique. *Quarterly Journal of Mechanics and Applied Mathematics, 4*, 236–240.
12. Ercegovac, M. D., & Lang, T. (2003). *Digital arithmetic*. Morgan Kaufmann.
13. Mallick, P. (2020). Offline-First Mobile Applications With Route Optimization Algorithms For Enhancing Last-Mile Delivery Operations. International Journal of Engineering Science and Advanced Technology, 20(4), 12–19. https://doi.org/10.64771/ijesat.2020.v20.i04.pp12-19.
14. Ganji, M. (2025). Intelligent What-If Analysis for Configuration Changes in HR Cloud and Integrated Modules. International Journal of All Research Education and Scientific Methods, 13(04), 4828–4835. https://doi.org/10.56025/ijaresm.2025.1304254828.
15. Kolagotla, R. K., et al. (1997). VLSI implementation of a 200-MHz 16×16 left-to-right carry-free multiplier in 0.35µm CMOS technology for next-generation DSPs. In *Proceedings of IEEE Custom Integrated Circuits Conference* (pp. 469–472).
16. Erukude, S. T., & Marella, V. C. (2025, September). Multimodal Detection of Fake Reviews using BERT and ResNet-50. In 2025 4th International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 877-882). IEEE.
17. Gaddam, S. Integrating Analytics into the Development Process: Bridging the Gap between Data Insights and Design Execution.
18. Goto, G., et al. (1997). A 4.1 ns compact 54×54-b multiplier utilizing sign-select Booth encoders. *IEEE Journal of Solid-State Circuits, 32*(11), 1676–1682.
19. S. M. K. P. (2025). Cryptography in iOS: A Study of Secure Data Storage and Communication Techniques. International Journal on Science and Technology, 16(1). https://doi.org/10.71097/ijsat.v16.i1.1403.
20. Fried, R. (1997). Minimizing energy dissipation in high-speed multipliers. In *Proceedings of the International Symposium on Low Power Electronics and Design* (pp. 214–219).
21. Weste, N. H. E., & Eshraghian, K. (1993). *Principles of CMOS VLSI design: A systems perspective* (2nd ed.). Addison-Wesley.
22. Fadavi-Ardekani, J. (1993). M×N Booth encoded multiplier generator using optimized Wallace trees. *IEEE Transactions on VLSI Systems, 1*(2), 120–125.
23. Farooqui, A. A., et al. (1998). General data-path organization of a MAC unit for VLSI implementation of DSP processors. In *Proceedings of IEEE ISCAS* (Vol. 2, pp. 260–263).
24. Hwang, K. (1976). *Computer arithmetic: Principles, architecture, and design*. John Wiley & Sons.
25. Cheng, K. H., et al. (1998). The improvement of conditional sum adder for low power applications. In *Proceedings of IEEE International ASIC Conference* (pp. 131–134).
26. Dadda, L. (1965). Some schemes for parallel multipliers. *Alta Frequenza, 34*, 349–356.
27. Oklobdzija, V. G. (1999). High-speed VLSI arithmetic units: Adders and multipliers. In *Proceedings of IEEE International Conference on Computer Design* (pp. 280–285).
28. Parhi, K. K. (1999). *VLSI digital signal processing systems: Design and implementation*. John Wiley & Sons.
29. Kulkarni, P., Gupta, P., & Ercegovac, M. D. (2011). Trading accuracy for power with an underdesigned multiplier architecture. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 19*(2), 252–261.

30. Han, J., & Orshansky, M. (2013). Approximate computing: An emerging paradigm for energy-efficient design. In *Proceedings of the 18th IEEE European Test Symposium* (pp. 1–6).