Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 6, June : 2024

A Novel Architecture for Three-Operand Adder for Area Efficient Low-Power Computations

Devi Agraharapu
M-Tech Student
Aditya College of Engineering, Surampalem
devirao3rd@gmail.com

Y Sugandhi Naidu
Associate Professor
Aditya College of Engineering, Surampalem
Sugandhi_ece@acoe.edu.in

ABSTRACT: The three-operand binary adder plays a crucial role in cryptographic systems and pseudorandom bit generators (PRBGs), supporting modular arithmetic operations. Although the carry-save adder (CS3A) is traditionally used for three-operand addition, its sequential carry propagation introduces significant delay. In contrast, the Han-Carlson adder (HCA), a parallel prefix two-operand adder, can also handle three-operand addition with a shorter critical path delay, despite its increased hardware complexity. To address these challenges, this paper introduces a novel adder architecture designed for enhanced speed and efficiency. This architecture leverages pre-computation of bitwise addition combined with carry-prefix logic for executing three-operand binary addition operations. By adopting this approach, the design minimizes area usage, reduces power consumption, and achieves significantly lower adder delay compared to conventional methods like the HC3A. The proposed architecture has been successfully implemented and evaluated for both 32-bit and 64-bit adders, demonstrating superior performance across various metrics. This innovation represents a promising advancement in the field of adder design, particularly beneficial for applications requiring high-speed arithmetic operations with minimal computational overhead.

KEYWORDS: Three-operand adder, carry save adder (CSA), Han-Carlson adder (HCA), modular arithmetic.

I. INTRODUCTION

To achieve optimal system performance and ensure robust physical security, it is crucial to integrate cryptographic algorithms into hardware systems [1]–[3]. Modular arithmetic, which includes operations such as modular exponentiation, multiplication, and addition, serves as a foundational component in numerous cryptographic algorithms [4]. Thus, the effectiveness of cryptographic algorithms heavily relies on the efficient execution of congruential modular arithmetic operations. Among these operations, the Montgomery algorithm [5]–[7] stands out as highly efficient for implementing modular multiplication and exponentiation. At its core, the algorithm relies on three-operand binary addition [6]–[8], a pivotal technique that significantly enhances both computational efficiency and security in cryptographic applications. This method plays a critical role in minimizing computation time and resource utilization, thereby bolstering the overall performance and reliability of cryptographic systems. Three-operand binary addition plays a pivotal role in the arithmetic operations of linear congruential generator (LCG)-based pseudorandom bit generators (PRBGs) such as coupled LCG (CLCG) [9], modified dual-CLCG (MDCLCG) [10], and coupled variable input LCG (CVLCG) [11]. Among these methods, the modified dual-CLCG (MDCLCG) stands out for its superior security and high randomness. It ensures polynomial-time unpredictability and security, particularly effective for n ≥ 32-bit applications. Notably, the security of

MDCLCG scales with larger operand sizes, reinforcing its robustness in cryptographic applications. However, the hardware architecture of MDCLCG poses challenges due to its components, which include four three-operand modulo-2n adders, two comparators, and four multiplexers [10]. While these components contribute to its security and randomness, they also lead to linear increases in both area requirements and critical path delays. This characteristic highlights the trade-offs between security features and hardware complexity in implementing MDCLCG. Thus, optimizing the implementation of the three-operand adder holds the potential to significantly enhance the performance of MDCLCG, effectively addressing both area efficiency and critical path delays. By refining how the three-operand adder is integrated into the MDCLCG architecture, it becomes feasible to mitigate the linear increases in hardware requirements and delay associated with its current configuration. This optimization effort is crucial for maintaining the algorithm's robust security features while improving its overall operational efficiency in practical cryptographic applications. Therefore, strategic enhancements in the implementation of the three-operand adder are essential steps towards maximizing the effectiveness and applicability of MDCLCG across diverse computing environments and security requirements.

In modular arithmetic, the execution of three-operand binary addition can be accomplished through either two two-operand adders or a single three-operand adder. Among these methods, the carry-save adder (CS3A) is widely acknowledged for its efficiency and is commonly employed in cryptographic algorithms [5]–[8], [12]–[14] and pseudorandom bit generator (PRBG) techniques [9]–[11]. However, despite its area efficiency, the CS3A suffers from extended carry propagation delays during the ripple-carry phase. This delay significantly impacts the performance of MDCLCG and other cryptographic architectures, especially when deployed on resource-constrained IoT-based hardware devices. To address these challenges, advancements in the design and implementation of three-operand adders are crucial. By minimizing carry propagation delays and optimizing hardware utilization, it becomes possible to enhance the operational efficiency and overall performance of cryptographic algorithms like MDCLCG. This optimization is essential for meeting the stringent requirements of modern cryptographic standards while ensuring compatibility with diverse hardware environments, including IoT devices. Thus, ongoing research and development efforts aim to refine three-operand adder designs to strike a balance between area efficiency, delay reduction, and cryptographic robustness, thereby advancing the capabilities of cryptographic systems in various practical applications.

One approach to mitigate critical path delays in three-operand binary addition involves employing a parallel-prefixed two-operand adder such as the Han-Carlson (HCA). By utilizing the HCA, the critical path delay can be reduced to O(log2 n). However, this improvement comes at the cost of increased hardware area complexity, scaling up to O(n log2 n) [15]. The HCA offers a trade-off between critical path delay reduction and increased hardware resource utilization. This trade-off is particularly relevant in cryptographic applications where minimizing delay is crucial for maintaining performance while managing the hardware footprint effectively. Despite the increased area complexity, the HCA's capability to streamline critical path operations underscores its potential to enhance the efficiency of three-operand binary addition in scenarios requiring high-speed arithmetic operations. Efforts in refining parallel-prefixed adder designs continue to explore optimizations that balance computational efficiency with hardware constraints. Such advancements are pivotal in advancing the capabilities of cryptographic systems, ensuring they meet the stringent demands of modern cryptographic algorithms across diverse computational platforms.

Hence, the development of a streamlined VLSI architecture capable of efficiently executing rapid three-operand binary addition with minimal hardware resources is imperative. This paper introduces an innovative approach for achieving efficient and high-speed addition using a novel adder technique. The method leverages pre-computed bitwise addition followed by carry-prefix computation logic to handle three-operand addition operations. This approach yields substantial reductions in gate area consumption and propagation delay when compared to existing methods like the HCA-based three-operand adder (HC3A). Moreover, the proposed adder architecture has been implemented using Verilog HDL and synthesized with a commercially available 32nm CMOS technology library. To evaluate its performance, metrics such as area-delay and power-delay products are measured and compared against those of conventional CS3A and HC3A three-operand adder techniques. By optimizing the hardware design through innovative methodologies, this research aims to advance the efficiency and effectiveness of cryptographic arithmetic operations. These improvements are pivotal for enhancing the overall performance and scalability of cryptographic systems across various computational platforms and applications. The findings contribute to the ongoing evolution of VLSI design practices, offering insights into achieving superior performance metrics crucial for modern cryptographic algorithms.

II. LITERATURE REVIEW

Three-operand binary addition plays a pivotal role in the realm of congruential modular arithmetic structures [5]–[8] and is integral to the operation of LCG-based pseudorandom bit generators (PRBGs) like CLCG [9], MDCLCG [10], and CVLCG [11]. This operation can be implemented in two primary configurations: either using two stages of two-operand adders or a single stage with a dedicated three-operand adder. Among the techniques available, the carry-save adder (CSA) stands as a widely embraced method for executing three-operand binary addition [9]–[14]. This approach efficiently computes the sum of three operands by breaking down the addition process into two distinct stages. By first generating partial sums without the carry propagation, followed by a final addition stage, the CSA minimizes critical path delays and optimizes hardware utilization. This dual-stage methodology is particularly advantageous in cryptographic applications where speed and efficiency are paramount, enabling faster computation of modular arithmetic operations crucial for ensuring data security and integrity. The versatility of the CSA extends to various computational environments, from high-performance computing to embedded systems and IoT devices. Its adoption underscores a balance between computational efficiency and hardware complexity, essential for meeting the rigorous demands of modern cryptographic algorithms and PRBG methods. Ongoing research continues to refine and optimize CSA designs, aiming to enhance performance metrics such as area efficiency, power consumption, and overall computational speed in cryptographic applications.

The process begins with the first stage, where a series of full adders simultaneously compute the "carry" and "sum" bits using three binary inputs: ai, bi, and ci. This stage operates in parallel to efficiently generate intermediate results. Following this, the second stage employs a ripple-carry adder to finalize the n-bit "sum" and produce the one-bit "carry-out" signal at the output of the three-operand addition. Here, the "carry-out" signal propagates sequentially through each of the n full adders in the ripple-carry stage. As a consequence, the delay in processing increases linearly with the length of the binary inputs. This dual-stage approach effectively balances between parallel processing in the initial stage for rapid computation of intermediate results and sequential processing in the ripple-carry stage to ensure accurate final summation across all bits. However, the linear increase in delay highlights a critical consideration in designing efficient arithmetic units, particularly in applications where speed and timing are critical factors. Future advancements aim to optimize this process, seeking to minimize delay while maintaining computational accuracy and efficiency in diverse computing environments.

The architecture of the three-operand carry-save adder is depicted in Figure 1, emphasizing the components and pathways involved in its operation. A significant aspect highlighted in the diagram is the critical path delay, denoted by the dashed line. This delay predominantly stems from the carry propagation process within the ripple-carry stage. In this architectural layout, the carry-save adder operates in a structured manner to manage the addition of three operands efficiently. The initial stage typically involves concurrent computation of partial sums and carries, optimizing performance through parallel processing. Subsequently, these partial results are fed into the ripple-carry stage, where the final addition and carry propagation occur sequentially across the full width of the operands. The dashed line in Figure 1 underscores the path through which signals experience the most significant delay, directly impacting the overall speed and efficiency of the addition process. Understanding and mitigating this delay is crucial for optimizing the adder's performance, especially in applications demanding high-speed arithmetic operations, such as cryptography and digital signal processing. Continued research focuses on refining the design of carry-save adders to reduce critical path delays and enhance overall computational efficiency. These efforts aim to advance the capabilities of arithmetic units in modern computing systems, ensuring they meet the stringent demands of diverse computational tasks across various technological domains.

The Han-Carlson adder (HCA) is renowned for its efficiency in terms of speed and gate complexity relative to other two-operand adder techniques. It boasts the lowest Area Delay Product (ADP) and Power-Delay Product (PDP) among its peers, making it a favorable choice for high-performance arithmetic operations. Figure 2 illustrates how the HCA facilitates three-operand addition effectively in two distinct stages. The architecture of the HCA-based three-operand adder (HC3A) is elaborated in detail in [15], emphasizing its design principles and operational stages. In HC3A, the addition process unfolds through a meticulously crafted sequence that optimizes both speed and resource utilization. Key to its operation is the propagate chain within the PG logic of the Han-Carlson adder, which dictates the maximum combinational path delay. This approach ensures that the HC3A efficiently handles the addition of three operands by leveraging the inherent parallelism and optimized propagation paths characteristic of the HCA. By breaking down the addition into stages, HC3A mitigates delay while maintaining accuracy, crucial for applications demanding rapid and reliable arithmetic computations. Ongoing advancements in the field seek to refine the PG logic and overall architecture of HC3A, aiming to further minimize path delays and enhance computational efficiency. These efforts are pivotal in advancing the capabilities of arithmetic units in modern computing systems, aligning them with the evolving demands of diverse computational tasks across various technological domains.

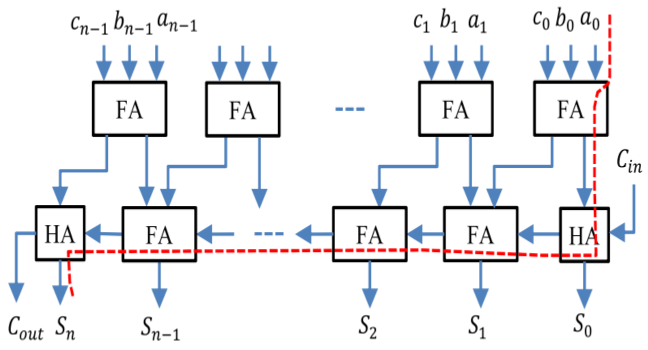


Fig. 1. Three-operand carry-save adder (CS3A).

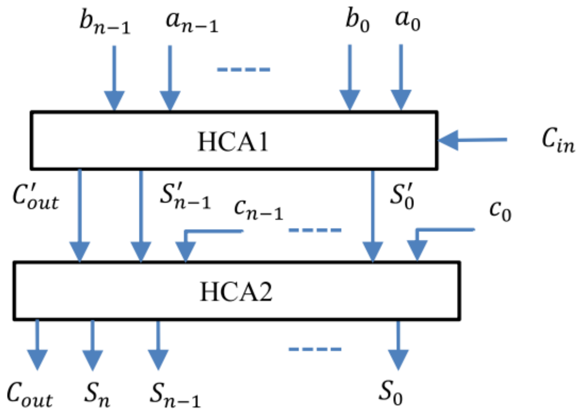


Fig. 2. Block level architecture of HCA-based three-operand adder (HC3A).

The HC3A, utilizing the Han-Carlson adder, effectively reduces critical path delays in comparison to traditional three-operand carry-save binary adders [15]. Nonetheless, with increasing bit lengths, its area requirement escalates proportionately, approximating O(n log2 n). To address this inherent trade-off between area utilization and delay, the subsequent section presents a novel approach aimed at achieving both high speed and efficiency in three-operand addition, accompanied by an optimized VLSI architecture. This new technique aims to optimize the balance between hardware complexity and performance metrics such as delay and area efficiency. By innovatively integrating pre-computed bitwise addition and advanced carry-prefix computation logic, this approach seeks to minimize both area usage and propagation delays associated with traditional adder designs. This innovation is crucial for enhancing the scalability and applicability of three-operand addition techniques in modern computational contexts, particularly in cryptographic systems and other applications requiring rapid and reliable arithmetic operations. Furthermore, the proposed VLSI architecture, implemented using Verilog HDL and synthesized with cutting-edge CMOS technology, aims to demonstrate superior performance characteristics compared to existing methodologies like HC3A and CS3A. By evaluating metrics such as area-delay product and power-delay product, this research contributes to advancing the state-of-the-art in VLSI design, offering insights into enhancing computational efficiency across diverse computing platforms.

III. PROPOSED METHODOLOGY

This section presents a novel technique and its corresponding VLSI architecture designed specifically for executing three-operand addition within modular arithmetic frameworks. The proposed approach innovatively adopts a parallel prefix structure, distinguishing itself from conventional three-stage prefix adders by expanding into four distinct stages. These stages encompass essential functionalities such as bit-addition logic, base logic, PG (propagate and generate) logic, and sum logic, each playing a critical role in the overall operation of the adder.

Detailed explanations and logical expressions for each of these stages are provided below:

Bit-Addition Logic: This initial stage handles the fundamental addition of individual bits from the operands. It computes the sum and carry-out bits based on the input bits ai, bi, and ci.

Base Logic: Following the bit-addition stage, the base logic integrates the results from the bit-addition stage to establish the foundation for subsequent computations. It prepares the data for further processing within the adder architecture.

PG Logic (Propagate and Generate): The PG logic stage is pivotal in determining the propagation and generation of carry signals throughout the adder structure. It manages the flow of carry information across different sections, optimizing the propagation paths to minimize delay and improve efficiency.

Sum Logic: Finally, the sum logic stage synthesizes the results computed from earlier stages into the final output of the three-operand addition operation. It consolidates the carry and sum bits to produce the accurate result of the arithmetic addition process.

This innovative architecture aims to enhance both speed and efficiency in three-operand addition operations, addressing challenges related to area utilization and propagation delay. By delineating the functionality and logical operations of each stage, this approach provides a comprehensive framework for advancing the state-of-the-art in VLSI design for modular arithmetic applications.

Stage-1: Bit Addition Logic:

$$S'_i = a_i \; xor \; b_i \; xor \; c_i$$
$$cy_i = a_i . b_i + b_i . c_i + c_i . a_i$$

Stage-2: Base Logic:

$$G_{i:i} = G_i = S'_i . cy_{i-1}$$
$$G_{0:0} = G_0 = S'_0 . C_{in}$$
$$P_{i:i} = P_i = S'_i \; xor \; cy_{i-1}$$
$$P_{0:0} = P_0 = S'_0 \; xor \; C_{in}$$

Stage-3: PG (Generate and Propagate) Logic:

$$G_{i:j} = G_{i:k} + P_{i:k} . G_{k-1:j}$$
$$P_{i:j} = P_{i:k} \, . P_{k-1:j}$$

Stage-4: Sum Logic:

$$S_i = (P_i \; xor \; G_{i-1:0})$$
$$S_0 = P_0$$
$$C_{out} = G_{n:0}$$

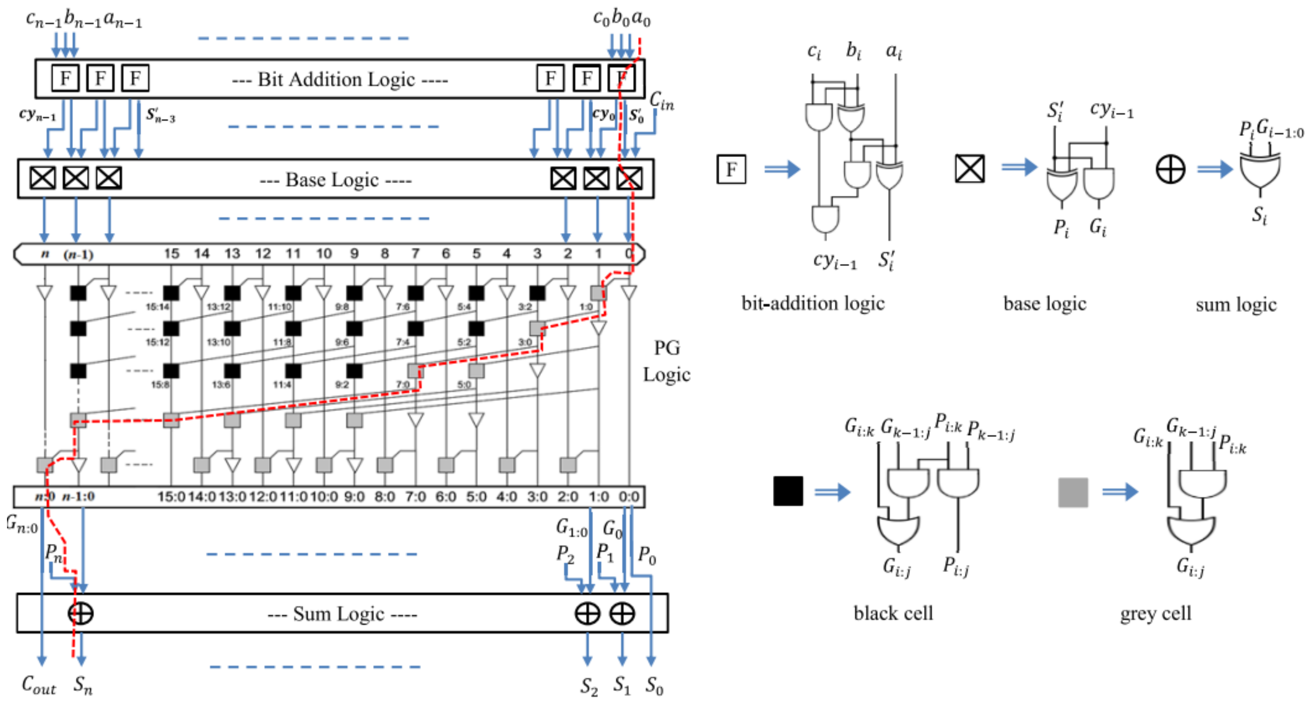
The VLSI architecture proposed for the three-operand binary adder and its internal structure are presented in Figure 3. This innovative adder technique processes the addition of three n-bit binary inputs through a meticulously designed sequence spanning four distinct stages. In the initial stage, known as the bit-addition logic, an array of full adders is employed to execute bitwise addition on the three n-bit binary input operands. Each full adder within this stage computes essential signals: "sum (Si)" and "carry (cyi)," as depicted in Figure 3(a). This stage focuses on generating precise sum and carry signals for subsequent processing steps. The logical expressions governing the computation of "sum (Si)" and "carry (cyi)" signals are meticulously detailed in Stage-1. Figure 3(b) provides a visual representation of the logical diagram illustrating the intricate workings of the bit-addition logic. This diagram serves as a blueprint for understanding how individual bits from the operands are combined to derive the correct sum and carry results. Moving forward, subsequent stages in the adder architecture, including base logic, PG (propagate and generate) logic, and sum logic, build upon the outputs generated in the bit-addition stage. These stages collectively optimize the addition process, ensuring efficient handling of carry propagation and synthesis of final output results.

The proposed VLSI architecture not only enhances the speed and efficiency of three-operand binary addition operations but also offers insights into advancing the state-of-the-art in modular arithmetic computations. By detailing each stage's functionality and logical operations, this approach lays the groundwork for developing robust and scalable VLSI designs suitable for modern computational applications.

The initial stage of the proposed three-operand binary adder focuses on computing the "sum (Si)" and "carry" signals using full adders. Subsequently, in the second stage, known as the base logic stage, these signals are utilized to derive the generate (Gi) and propagate (Pi) signals. This process involves combining the "sum (Si)" output bit from the current full adder with the "carry" output bit from the adjacent full adder on the right. Figure 3(a) illustrates this computation using a configuration known as the "squared saltire-cell," which consists of a total of n + 1 saltire-cells within the base logic stage. These saltire-cells visually depict how the generate (Gi) and propagate (Pi) signals are calculated based on the inputs from the previous stage. Furthermore, Figure 3(b) presents the logic diagram of the saltire-cell, detailing the intricate logical operations involved in generating the Gi and Pi signals. This diagram serves as a comprehensive reference for understanding the internal workings of the base logic stage and how it contributes to the overall operation of the three-operand adder architecture. By delineating these stages and their respective components, the proposed VLSI architecture aims to optimize both speed and efficiency in three-operand binary addition operations. This systematic approach not only enhances computational performance but also provides a foundation for further advancements in VLSI design tailored for modular arithmetic applications and beyond.

The proposed three-operand adder technique incorporates an external carry-input signal (Cin) to facilitate the addition process. This additional signal, Cin, plays a crucial role in the base logic stage by influencing the calculation of G0 (S0 · Cin). This computation occurs within the first saltire-cell of the base logic stage, where the interaction between Cin and the sum output (S0) is pivotal for generating the initial generate (G0) signal. Moving to the third stage of the adder architecture, known as the "generate and propagate logic" (PG), the focus shifts to pre-computing the carry bit. This stage utilizes a combination of black and grey cell logics to determine the carry generate (Gi: j) and propagate (Pi: j) signals. Figure 3(b) provides a detailed logical diagram depicting how these black and grey cells interact to calculate the carry signals effectively. The logical operations within these cells are designed to optimize the carry computation process, ensuring accurate and efficient handling of carry propagation across the adder structure. This strategic approach enhances the overall performance of the three-operand adder by minimizing delays and optimizing resource utilization. By integrating external carry-input capability and refining carry computation through advanced PG logic, the proposed VLSI architecture aims to advance the state-of-the-art in modular arithmetic operations. These innovations not only bolster the speed and efficiency of three-operand addition but also pave the way for future developments in VLSI design for high-performance computing applications.

The proposed adder architecture integrates multiple prefix computation stages, specifically (log2 n + 1) stages, to optimize the addition process. This design choice effectively manages the critical path delay, which predominantly hinges on the efficiency of the carry propagate chain within the adder structure. In the final stage, known as the sum logic stage, the adder computes the "sum (Si)" bits using a defined logical expression: Si = (Pi ⊕ Gi-1:0). Here, Pi represents the carry propagate bits, while Gi-1:0 encompasses the carry generate bits derived from preceding stages. This logical operation ensures accurate summation of the operands, synthesizing the final result efficiently. Moreover, the carryout signal (Cout) is directly derived from the carry generate bit Gn:0, encapsulating the overall carry status of the addition operation. This streamlined approach enhances computational accuracy while minimizing propagation delays, critical for maintaining high-speed arithmetic operations. By leveraging advanced prefix computation stages and meticulous logical expressions, the proposed VLSI architecture aims to optimize both performance metrics and resource utilization in three-operand addition. These innovations contribute to advancing the capabilities of modular arithmetic in computational applications, fostering robust solutions for contemporary computing challenges.

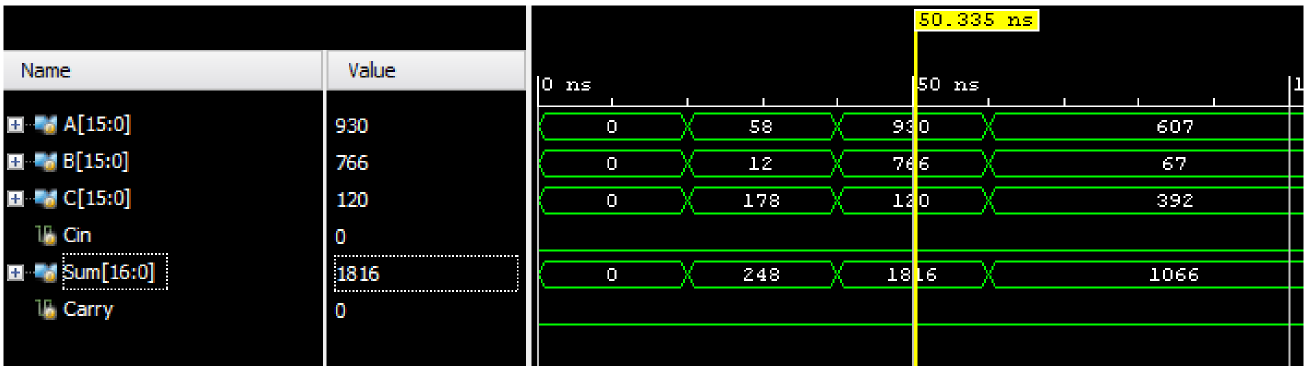


(a) (b)

Fig. 3. Proposed three-operand adder; (a) First order VLSI architecture, (b) Logical diagram of bit addition, base logic, sum logic, black-cell and grey-cell.

IV. RESULTS & DISCUSSION

The methodology proposed here is implemented using Verilog HDL coding and simulated using the Xilinx Vivado Software tool. The simulation outcomes are depicted in the subsequent figures.



```
Name        Value    0 ns                         50 ns
A[15:0]     930      0      58      930      607
B[15:0]     766      0      12      766      67
C[15:0]     120      0      178     120      392
Cin         0
Sum[16:0]   1816     0      248     1816     1066
Carry       0
```

Fig 4: 16-bit adder simulation result

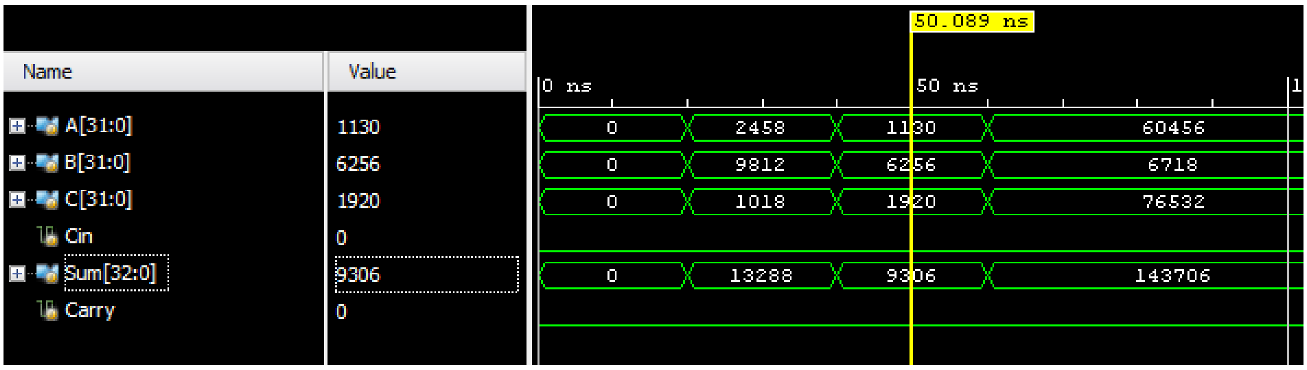


Fig 5: 32-bit adder simulation result

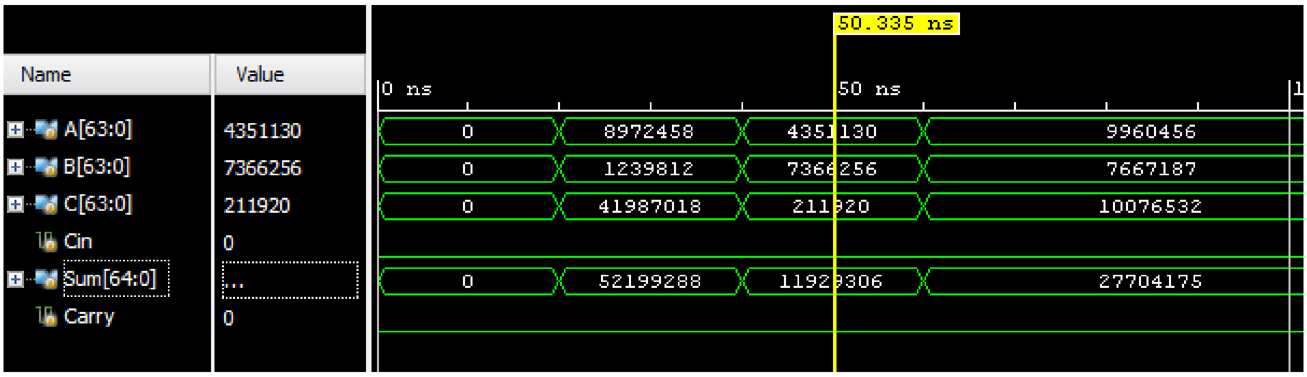


Fig 6: 64-bit adder simulation result

The outcomes of the proposed adder are juxtaposed with those of existing designs such as the Koggestone and Han-Carlson adders. The respective findings are presented in the table below.

	Hancarlson Adder	Koggestone Adder	Proposed Adder
Static Power	0.166	0.167	0.162
Dynamic Power	10.633	10.983	10.069
Total On-chip Power	10.799	11.15	10.231
No.of Utilized LUT's	31	46	24

V. CONCLUSION

This study introduces an innovative approach to enhance the speed and efficiency of three-operand binary addition through a novel adder technique and its tailored VLSI architecture. This development addresses the critical need for optimizing modular arithmetic computations crucial in cryptography and pseudorandom bit generator (PRBG) applications. The proposed three-operand adder leverages a parallel prefix structure organized into four stages, specifically designed to manage the addition process efficiently across three input operands. The key innovation lies in optimizing both delay and area utilization within critical stages such as PG (propagate and generate) logic and bit-addition logic. This optimization strategy aims to achieve significant reductions in overall area footprint and power consumption, essential for enhancing computational efficiency in modern computing environments. To ensure a comprehensive evaluation, this research extends the concept to include the development of a hybrid Han-Carlson three-operand adder (HHC3A). This approach maintains consistency in coding style across implementations of HHC3A, HC3A, and CS3A using Verilog HDL. By adopting this standardized methodology, the study facilitates fair and accurate comparisons between different adder topologies, providing insights into their respective strengths and performance metrics. By advancing the state-of-the-art in VLSI design for three-operand addition, this research contributes to the ongoing evolution of cryptographic algorithms and PRBG methods. These advancements are poised to impact various computational domains, ensuring robust and efficient solutions capable of meeting the escalating demands of modern computing applications.

REFERENCES

[1]. M. M. Islam, M. S. Hossain, M. K. Hasan, M. Shahjalal, and Y. M. Jang, “FPGA implementation of high-speed area-efficient processor for elliptic curve point multiplication over prime field,” IEEE Access, vol. 7, pp. 178811–178826, 2019.

[2]. Z. Liu, J. GroBschadl, Z. Hu, K. Jarvinen, H. Wang, and I. Verbauwhede, “Elliptic curve cryptography with efficiently computable endomorphisms and its hardware implementations for the Internet of Things,” IEEE Trans. Comput., vol. 66, no. 5, pp. 773–785, May 2017.

[3]. Z. Liu, D. Liu, and X. Zou, “An efficient and flexible hardware implementation of the dual-field elliptic curve cryptographic processor,” IEEE Trans. Ind. Electron., vol. 64, no. 3, pp. 2353–2362, Mar. 2017.

[4]. B. Parhami, Computer Arithmetic: Algorithms and Hardware Design. New York, NY, USA: Oxford Univ. Press, 2000.

[5]. P. L. Montgomery, “Modular multiplication without trial division,” Math. Comput., vol. 44, no. 170, pp. 519–521, Apr. 1985.

[6]. S.-R. Kuang, K.-Y. Wu, and R.-Y. Lu, “Low-cost high-performance VLSI architecture for montgomery modular multiplication,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 2, pp. 434–443, Feb. 2016.

[7]. S.-R. Kuang, J.-P. Wang, K.-C. Chang, and H.-W. Hsu, “Energy-efficient high-throughput montgomery modular multipliers for RSA cryptosystems,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 21, no. 11, pp. 1999–2009, Nov. 2013.

[8]. S. S. Erdem, T. Yanik, and A. Celebi, “A general digit-serial architecture for montgomery modular multiplication,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 5, pp. 1658–1668, May 2017.

[9]. R. S. Katti and S. K. Srinivasan, “Efficient hardware implementation of a new pseudo-random bit sequence generator,” in Proc. IEEE Int. Symp. Circuits Syst., Taipei, Taiwan, May 2009, pp. 1393–1396.

[10]. A. K. Panda and K. C. Ray, “Modified dual-CLCG method and its VLSI architecture for pseudorandom bit generation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 3, pp. 989–1002, Mar. 2019.

[11]. A. Kumar Panda and K. Chandra Ray, “A coupled variable input LCG method and its VLSI architecture for pseudorandom bit generation,” IEEE Trans. Instrum. Meas., vol. 69, no. 4, pp. 1011–1019, Apr. 2020.

[12]. N. Weste and K. Eshraghian, Principles of CMOS VLSI Design—A Systems Perspective. Reading, MA, USA: Addison-Wesley, 1985.

[13]. T. Kim, W. Jao, and S. Tjiang, “Circuit optimization using carry-saveadder cells,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 17, no. 10, pp. 974–984, Oct. 1998.

[14]. A. Rezai and P. Keshavarzi, “High-throughput modular multiplication and exponentiation algorithms using multibit-scan–multibit-shift technique,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 9, pp. 1710–1719, Sep. 2015.

[15]. A. K. Panda and K. C. Ray, “Design and FPGA prototype of 1024- bit Blum-Blum-Shub PRBG architecture,” in Proc. IEEE Int. Conf. Inf. Commun. Signal

Process. (ICICSP), Singapore, Sep. 2018, pp. 38–43.

[16]. T. Han and D. A. Carlson, “Fast area-efficient VLSI adders,” in Proc. IEEE 8th Symp. Comput. Arithmetic (ARITH), May 1987, pp. 49–56.

[17]. D. L. Harris, “Parallel prefix networks that make tradeoffs between logic levels, fanout and wiring racks,” U.S. Patent 0 225 706 A1, Nov. 11, 2004.

[18]. H. Ling, “High-speed binary adder,” IBM J. Res. Develop., vol. 25, no. 3, pp. 156–166, Mar. 1981.

[19]. R. Jackson and S. Talwar, “High speed binary addition,” in Proc. Conf. Rec. 38th Asilomar Conf. Signals, Syst. Comput., vol. 2. Pacific Grove, CA, USA, Nov. 2004, pp. 1350–1353.

[20]. K. S. Pandey, D. K. B. N. Goel, and H. Shrimali, “An ultra-fast parallel prefix adder,” in Proc. IEEE 26th Symp. Comput. Arithmetic (ARITH), Kyoto, Japan, Jun. 2019, pp. 125–134.

[21]. F. Jafarzadehpour, A. S. Molahosseini, A. A. Emrani Zarandi, and L. Sousa, “New energy-efficient hybrid wide-operand adder architecture,” IET Circuits, Devices Syst., vol. 13, no. 8, pp. 1221–1231, Nov. 2019.

[22]. S. Muthyala Sudhakar, K. P. Chidambaram, and E. E. Swartzlander, “Hybrid Han-Carlson adder,” in Proc. IEEE 55th Int. Midwest Symp. Circuits Syst. (MWSCAS), Boise, ID, USA, Aug. 2012, pp. 818–821.