Journal of VLSI circuits and systems
RESEARCH ARTICLE
Journal of VLSI Circuits and Systems, ISSN: 2582-1458 Vol. 8, No. 2, 2026 (pp. 15-22)
WWW.VLSIJOURNAL.COM

A Low Power and Area Efficient Leading Zero Counter

N. Hemanth[1], Sonali Agrawal[1], M. Vinodhini[1,*]
[1]Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Bengaluru, Karnataka, India.

KEYWORDS:
Leading Zero Counter (LZC),
Zero Detection Logic,
Logic Minimization,
Hierarchical Architecture,
Floating Point,
Arithmetic Operations,
Parallel Processing,
Data-Level Pipelining,
High-Performance Computing

ARTICLE HISTORY:
Received: 31.01.2026
Revised: 21.02.2026
Accepted: 06.03.2026

DOI:
https://doi.org/10.31838/jvcs/08.02.02

Abstract

The latest and up-to-date arrangements for arithmetic logic units are necessary for modern processors to function effectively. Leading zero counters (LZCs) are indispensable for floating-point operations because they accelerate computational throughput and allow for parallel processing. Advancements in ASIC technology have enabled the development of LZC systems that process information rapidly and use very little energy. The high-speed, energy-efficient performance of Application Specific Integrated Circuits (ASICs) is why they are preferred for creating LZCs in modern computer systems. By reducing the number of transistors required, the proposed LZC architecture results in smaller area and power consumption than other embedded multiplier designs. Of the four techniques, the 32-bit LZC results in a 28.72% space reduction and a 16.44% decrease in power consumption. Similar to 64-bit LZC, the design achieves a 27.65% reduction in area and a 21.64% reduction in power consumption. These changes cause only a small increase in delay compared to LZC done in the past. The design was implemented using Verilog Hardware Description Language (HDL) and processed using the Xilinx Vivado application on the 45 nm technology node. Consequently, the architecture is scalable and easily adapted to modern processing requirements. This architecture significantly reduces energy consumption, contributing to energy-efficient computing systems and supporting sustainable digital infrastructure.

Authors' e-mail ID: nunnahemanth2001@gmail.com, a_sonali@blr.amrita.edu, m_vinodhini@blr.amrita.edu

Authors' ORCID ID: 0009-0009-1536-9218, 0000-0001-9830-8721, 0000-0002-3718-2122

How to cite this article: Hemanth N, Agrawal S, Vinodhini M. A Low Power and Area Efficient Leading Zero Counter. Journal of VLSI Circuits and Systems, Journal of VLSI Circuits and System, Vol. 8, No. 2, 2026 (pp. 15-22)

Introduction

Today, data processing relies heavily on leading zero counters (LZCs) to ensure efficiency and accuracy.[1-3] They determine the number of leading zeros in binary numbers, a critical function in arithmetic operations. The need to process and identify leading zeros is high in floating-point calculation, changing data formats, and high-performance computing (HPC), where both quick work and accuracy are key.[4,5] LZCs support automatically adjusting the exactness of data, leaving calculation precision unchanged. As a result, LZCs play a valuable role in modern hardware by being used in deep learning accelerators, signal processing systems, security and encryption systems, and communication systems.[6,7]

Today's technical advances in computer architecture make it easier for system designers to adjust a system's performance and energy use.[8-10] New digital designs feature lookup tables (LUTs), efficient carry chains, best logic, and flush flip-flops.[11] Such features allow us to work quickly while managing low energy use.[12] They matter for HPC systems and also for real-time embedded platforms, AI, and cloud computing. Low-power ASIC technology makes LZCs essential components in low-energy devices, like mobile processors, IoT sensors, and equipment used in biology or medicine. Additionally, placing LZCs into advanced arithmetic units increases the precision and speed of floating-point operations. This also boosts the effectiveness of mixed-precision and adaptive-precision computing, both of which are essential in recent systems.[13,14]

Latency is further reduced, and design resources are saved by incorporating parallel processing into the latest

LZC designs.[15] Implementing these methods enables faster data computation by dealing with several data segments at a time.[16] Because traditional bit-counting methods are inefficient, modern approaches build tree-like, nested counting structures to achieve faster results.[10],[17] In addition, strategies such as clipped counting halt counting as soon as a bit-level threshold is met, and these strategies are used together with methods that adjust resource use to maximize efficiency. The enhancements are designed to meet the needs of particular applications. In addition, advanced LZC systems rely on pipelining and speculative execution to enhance throughput without significantly increasing energy consumption. These techniques are built into hardware to support the latest versions of CPUs, GPUs, and AI inference processors.[18] We expect LZCs to remain important in new areas such as embedded computing, quantum computing, and the development of neuromorphic processors.[10] These systems benefit from the fast, energy-efficient operation of LZCs.[19]

The paper introduces a new LZC design to enhance the precision and performance of leading-zero detection in arithmetic systems. A major task of an LZC is to locate the MSB, especially in the course of normalizing a floating-point value. LZCs operate by counting the leading zeros of a number, which directly determines how the mantissa is placed. They are valuable for properly identifying bits and supporting rounding, which makes floating-point operations much more precise. Minimizing power consumption in arithmetic logic unit components directly supports green computing initiatives and reduces the environmental footprint of high-performance computing systems.

Section II then provides examples of related work, differentiating the uses of LZCs in the field of digital computing. Section III explains the design for the LZC and its detailed implementation approach. In Section IV, the highlights of the analysis and simulation are presented using key metrics such as area, power, and delay. The final part of the paper, Section V, recapitulates the key results and points out potential future directions for growth.

Although hierarchical LZC architectures have been widely explored, this work introduces a simplified logic formulation and modular construction to reduce their hardware overhead. Compared with conventional design methods that build higher-bit LZCs with cascaded 2-bit blocks, the proposed design uses a direct 4-bit LZC formulation based on NOR-dominant logic with multiplexer-based selection, thereby eliminating intermediate validity-propagation stages and reducing overall transistor count. The proposed 4-bit unit requires only 28 transistors, compared to 50 in currently available designs, enabling hierarchical area and power savings in 8-, 16-, 32-, and 64-bit implementations. Furthermore, the architecture employs a hierarchical framework that is scalable with reduced inter-stage logic overhead, and its efficiency is verified through an in-depth evaluation using an ASIC-based technique.

Related Work

The 4-bit LZC circuit shown in Figure 1 is composed of two cascaded 2-bit LZC circuits. The input bits are processed in each 2-bit unit to give validity (V) and zero-count (Z) signals. These intermediate outputs are merged using additional logic gates, thus giving the final 4-bit LZC result, as in the referred design.[19] Figures 2 and 3 show the 8-bit and 16-bit LZCs, respectively.

LZCs are especially important for computations and normalizations performed using floating-point numbers in high-speed processing elements of a digital system. Classical LZC designs relied on serially counting the number of leading zeros.[3,20] However, serial methods introduced significant delays and were therefore not appropriate for high-performance processing. One way this disadvantage has been overcome is by reducing the latency of the counting task and, at the same time, improving the overall processing efficiency, so that LZCs can support modern, high-speed computational demands much more effectively.

The evolution of LZC architectures added parallel counting capability that enabled many input bits to be processed in parallel, and thus, delay was reduced with many fewer gates, as compared to traditional serial algorithms.[15] This development significantly improved

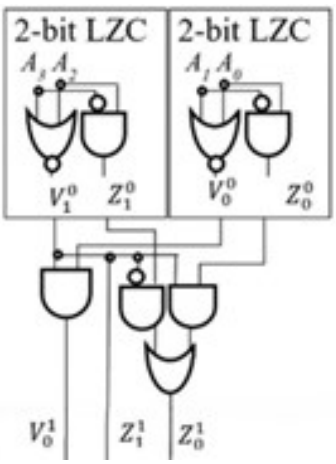

Fig. 1: Design of the 4-Bit Existing LZC.[19]

performance and provided a path to faster computation in areas that required it. Further efficiencies were obtained by using tree-like structures, which split the logic operations into distinct hierarchical levels, allowing faster, more efficient counting.[10] The ability to offer both high speed and precision make LZCs suitable for fast data processing jobs, since they are organized hierarchically. Also, contemporary high-performance LZCs have been implemented on FPGAs, where device-specific LUT and carry-chain capabilities are used to deliver highly efficient and reliable implementations.[21]

High-speed computation is achieved according to the LZC design; the rapid flow of intermediate results is facilitated by the carry-chain logic, which provides low-latency operations.[22] This aspect qualifies FPGA implementation of LZCs as being very appropriate in use in applications that involve real-time responsiveness and efficient signal processing, where speed and power consumption are also key determinants of the overall effectiveness of the system in question.[18,23]

LZC structure minimization focuses on saving hardware resources without sacrificing speed.[11,24] The key performance metrics tend to include LUTs, flip-flops, and multiplexers in FPGA implementations.[18] Specific architectures achieve better performance and efficiency by incorporating straightforward logic with simple bitwise operations, in a high-density, compact structure, so resource utilization is managed through a reduced hardware design, resulting in better overall performance.[14]

Known designs of LZCs implemented with FPGAs can dynamically adjust to different inputs (e.g., irregular leading zero distributions).[10] System optimization is obtained in real time using the FPGA reprogramming and does not necessitate a hardware change. Also, the ongoing technological improvements in FPGAs contribute to the improved performance of LZC; hence, these FPGAs are very useful in designing modern computation-intensive and performance-sensitive apps where the adaptability and efficiency features matter.[25]

Recent designs with zero counters (LZC) are a mix of both energy saving and parallel processing designs to provide a reasonable balance among the cost of performance, time delay, and energy consumption.[10,11,14] Hybrid systems combine both combinational logic and sequential logic properties, and their circuits have all the benefits over combinational and sequential systems, e.g., computational throughput with a reduced hardware area and total energy consumption. [9],[24]

Input								[8]			
A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	V	Z_2	Z_1	Z_0
1	X	X	X	X	X	X	X	0	0	0	0
0	1	X	X	X	X	X	X	0	0	0	1
0	0	1	X	X	X	X	X	0	0	1	0
0	0	0	1	X	X	X	X	0	0	1	1
0	0	0	0	1	X	X	X	0	1	0	0
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	0	0	1	X	0	1	1	0
0	0	0	0	0	0	0	1	0	1	1	1
0	0	0	0	0	0	0	0	1	1	1	0

Fig. 2: Truth Table of Existing 8-Bit LZC.[19]

These synergetic solutions allow LZCs to be well-suited to address the needs of contemporary applications that are characterized by both speed of operation and low-power consumption. Figure 2 shows the truth table of the present 8-bit LZC.[19]

The concept of LZC designs is still shifting toward increased scalability and flexibility, common in next-generation processing devices.[10] In future implementations, it is expected that machine learning employing optimization and advanced low-power operations will also be integrated to further reduce delay and overhead area.[7] Such innovations will guarantee that LZCs will form a cornerstone to arithmetic units, providing support to floating-point flexibility and competitiveness in HPC and embedded systems applications.[7]

Proposed Architectures For Leading Zero Counters

An improved architecture-based design of the LZC on ASIC serves as a time-saving method of application when the incoming data consists entirely of zeros. In this design, the output count Z is irrelevant when the validity signal V is true. Although the proposed design necessitates the generation of Z outputs only, it introduces the least extra is 1. The value of Z1 is 1 only when both input bits X3 and X4 are 0.

The multiplexer (MUX) selection line is derived from Z1 in the circuit topology. The MUX thus plays a central role in determining Z0, thereby guaranteeing correct counting behaviour in accordance with the current input conditions.

Also, the overflow flag V is only raised to high when all constituent input bits are set to zero, thus representing a complete leading-zero situation. Such an overflow-handling mechanism is essential to maintaining computational accuracy in binary arithmetic and allows the system to effectively handle edge cases.

Governing mathematical equations of this functionality are represented as follows:

- logic compared to the traditional methods and, at the same time, delivers high efficiency and reliable operation in actual implementations.

The current methodology is not a traditional paradigm, as it uses 4-bit LZCs as its basic building blocks. In the 8-bit LZC architecture presented here, two 4-bit processors are connected and their outputs are combined through supplementary logic gates. Auxiliary logic is applied at every hierarchical level, thus allowing scalability to the synthesis of larger LZCs, such as 16, 32, and 64-bit versions. The hierarchical structure provides effective growth by means of progressively incorporating smaller LZC modules into higher-bit configurations.

$$Z_1 = X_3 + X_4 \quad (1)$$

$$Z_0 = (Z_1?X_2 : X_4)$$
$$= (Z_1 \cdot X_2 + Z_1 \cdot X_4) \quad (2)$$

The 4-bit LZC referred in Fig. 3 produces two output bits Z1, Z0, as well as an overflow indicator V. Figure 4 presents the input and output bit relationships for the LZC. The key observation regarding output Z1 is that it remains 0 when at least one of the input bits X3 or X4

$$V = X_1 + X_2 + X_3 + X_4 \quad (3)$$

The logical framework represented in the LZC circuitry is formally defined by equations (1), (2), and (3), hence ensuring the effectiveness of the detection of leading zeros in binary input streams.

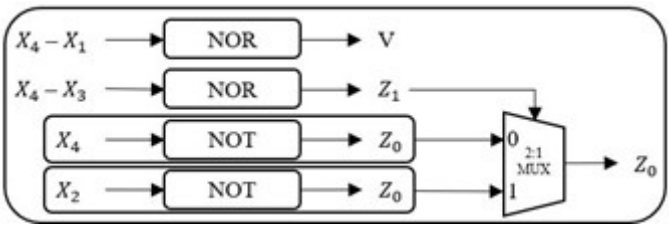

Fig. 3: Design of the 4-Bit LZC.

X_4	X_3	X_2	X_1	Z_1	Z_0	V
1	X	X	X	0	0	0
0	1	X	X	0	1	0
0	0	1	X	1	0	0
0	0	0	1	1	1	0
0	0	0	0	1	1	1

Fig. 4: Truth Table of Proposed 4-Bit LZC.

A. 8-bit LZC

Figure 5 represents the 8-bit LZC architecture as a hierarchical and modular design. The 8-bit input is split into two 4-bit blocks, and each of them is handled by a single 4-bit LZC block. Each of these modules identifies the first position of a binary 1 in their respective input streams and produces a 2-bit output (Z 1, Z 0) and a validity signal (V). All 4-bit LZs internally use simple logic gates, including NOR, AND, OR, and inverters, to do effective and high-speed zero-counting.

Figure 6 illustrates the relationship between the input bits and the resulting output signals of the proposed eight-bit LZC. The eight-bit input in this arrangement is divided into two separate four-bit blocks, each of which is processed independently by a separate four-bit LZC module. The modules produce intermediate products that are a two-bit zero-count (Z1, Z0) and a validity signal (V). The two module outputs are then supplied to a high-level combinational logic circuit that combines and adds the outputs to get the total number of leading zeros in the overall eight-bit input. Finally, the top-level logic generates the three-bit (Z2, Z1, Z0) zero count and a final valid signal, and, as such, it accurately captures the number of leading zeros.

B. 16-Bit LZC

The hierarchical and organised design used to achieve effective design of a LZC is depicted in Figure 7 and was specifically created to handle 16-bit input data. This approach uses a modular paradigm to make it scalable, so that latency is minimized and to have better performance. The 16-bit input is divided into 2 8-bit sectors

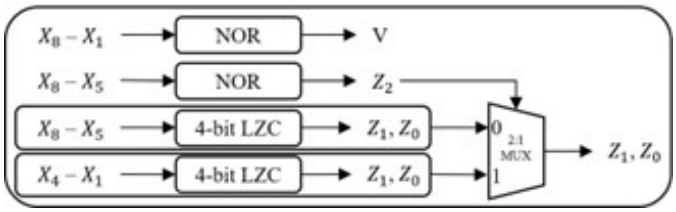

Fig. 5: Design of the 8-Bit LZC.

Input								Proposed			
A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	V	Z_2	Z_1	Z_0
1	X	X	X	X	X	X	X	0	0	0	0
0	1	X	X	X	X	X	X	0	0	0	1
0	0	1	X	X	X	X	X	0	0	1	0
0	0	0	1	X	X	X	X	0	0	1	1
0	0	0	0	1	X	X	X	0	1	0	0
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	0	0	1	X	0	1	1	0
0	0	0	0	0	0	0	1	0	1	1	1
0	0	0	0	0	0	0	0	1	1	1	1

Fig. 6: Truth Table of Proposed 8-Bit LZC.

that in turn can be divided in 2 4-bit blocks. At the lowest level, the 4-bit LZC modules compute the local numbers of leading zeros. Such intermediate values are then fed back through the hierarchy and summed together at the 8-bit level, with the results of that stage being filtered by more complex logic to produce the final leading-zero count with high efficiency.

This hierarchical organization simplifies logic complexity at every stage while enabling parallel processing, which effectively minimizes critical path delay. By applying a recursive and structured methodology, the LZC design attains greater flexibility, allowing adaptation to different bit-widths with only minor architectural adjustments. The modular nature of the design further promotes the reusability of smaller LZC units when constructing larger systems. Consequently, the architecture achieves both high performance and optimized resource utilization, making it highly suitable for advanced high-speed arithmetic computations and signal processing applications where efficiency and scalability are of primary importance.

Hierarchical LZC methodology represents a well-balanced combination of speed, area, and power consumption and therefore makes it a strong solution to next-generation systems in digital systems.

C. 32-bit LZC

Figure 8 illustrates the operation of a 32-bit LZC implemented using two 16-bit sub-units interconnected. These subunits work together to detect leading zeros in a 32-bit input with a high degree of efficiency. The architecture uses combinational logic gates to support

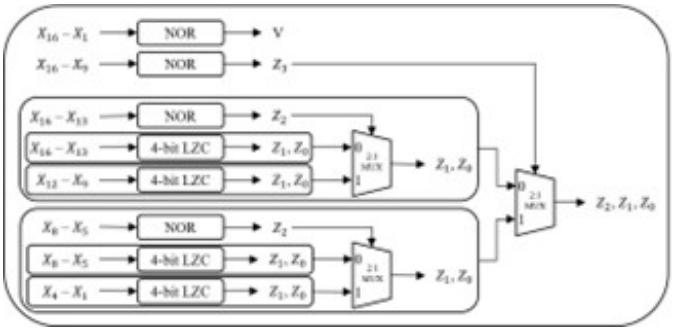

Fig. 7: Design of the 16-Bit LZC.

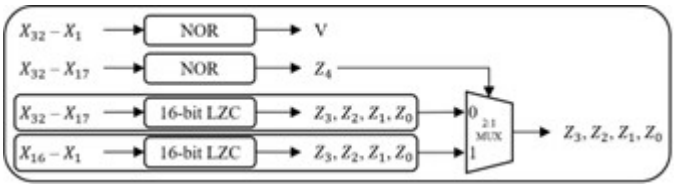

Fig. 8: Design of the 32-Bit LZC.

the lowest propagation delay and maximize the allocation of hardware resources. The validity flag (V) serves to show that the higher-order bits are all zero, and the output bits Z4–Z0 show the location of the first nonzero bit. This high-level architecture supports fast performance with reduced power usage, making it especially appropriate for floating-point arithmetic and data compression, as well as signal processing applications.

In addition, the 32-bit LZC modular design supports scalability by allowing a smooth upgrade to increased bit-widths. This flexibility increases the suitability of the architecture to meet the future processing needs of high-performance computing units and embedded systems.

D. 64-Bit LZC

The 64-bit LZC architecture, as shown in Figure 9, makes use of hierarchical logical constructs to easily detect leading zeros in input data, hence allowing faster processing and better performance in high-speed digital applications.

The architecture includes two LZC 32-bit units, which are further hierarchically divided into 16-bit, 8-bit, and 4-bit modules. The modular structure enables effective aggregation of the results, which increases the processing rate and significantly decreases the latency of the whole process.

In general, the hierarchical LZC implementation framework between 4-bit to 64-bit designs has systematic tradeoffs in speed, area, and power consumption. Its modular architecture ensures flexibility across a range of digital applications, making it a powerful solution for next-generation processors that require the efficiency of floating-point computation, real-time signal processing, and low-power embedded systems.

Results and Analysis

The evaluation of the design was based on a dual-flow methodology. Functional verification was performed with the Xilinx Vivado software, where simulation waveforms (Figures 10–13) proved the logical correctness of the bit count of leading zeroes for different bit-widths.

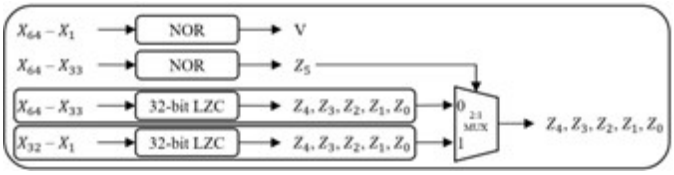

Fig. 9: Design of the 64-Bit LZC.

For the performance characterization, the Verilog HDL codes were synthesized by Cadence Genus with a target of a technology node of 45 nm ASIC. The implementation of advanced 45nm technology demonstrates a commitment to sustainable microelectronics design through improved power-performance tradeoffs. This ASIC flow was used to achieve technology-normalized metrics for area, power, and critical path delay, ensuring that the results reported represent actual physical implementation costs rather than FPGA-specific resource utilization.

Table 1 gives a comparison between the existing[17,19,25] and proposed 4-bit, 8-bit, 16-bit, 32-bit and 64-bit architectures are analyzed for the following performance parameters, such as power, area, and critical path delay.

The proposed architecture shows a small penalty in delay for some bit widths, and this is mainly as a result of the use of high fanout NOR-gates and hierarchical aggregation stages, where additional capacitive loading is experienced along critical paths. This architecture decision is deliberate because it optimizes the reduction of transistor count and minimizes switching activity, resulting in significant area and power savings. Consequently, the proposed LZC represents an intentional tradeoff between energy efficiency and silicon footprint and absolute minimum delay. Such characteristics make the design especially suitable for embedded processors, low-power floating-point units, and energy-constrained accelerators, where area and power are the primary design concerns. By achieving up to 21.64% power reduction and substantial area optimization, the design promotes sustainable semiconductor technology and resource-efficient hardware development.

Table 1: Comparison Table of Area, Power, and Delay

n	DESIGN	AREA (μm²)	POWER (μW)	DELAY (ps)
4	New [19]	21.87 24.46	0.60 0.60	1404 1234
8	New	41.70	0.72	2219
	[25]	42.41	0.75	2036
	[19]	52.15	0.79	1660
	[17]	42.08	0.75	2242
16	New	86.70	0.93	2451
	[25]	86.53	0.95	2818
	[19]	113.97	1.03	1923
	[17]	85.45	0.86	3070
32	New	189.93	1.39	3658
	[25]	179.02	1.30	3122
	[19]	239.94	1.51	2229
	[17]	176.80	1.08	4500
64	New	410.89	2.21	4112
	[25]	368.71	2.06	3510
	[19]	494.21	2.46	2573
	[17]	354.14	1.58	5523

Across all the above operand widths, i.e., 4-64 bits, the proposed hierarchical LZC shows uniform reduction in the hardware cost and power consumption in comparison to the previous hardware designs with little overhead in delay. Area improvement varies between about 1–3% in the worst case (very much for some 16- and 32-bit comparisons) and 27–31% in the best case (measured for 32- and 64-bit implementations) with the average area reduction of about 18–20% for all bit-widths considered. Power saving exhibits a comparable behaviour, ranging between 0 and 3% in the worst case and almost 21% in the best case, resulting in a mean power saving between 10 and 12%. These advantages come with point-of- optimization with respect to critical path delay, ranging from a negligible added delay in the case of small configurations, to higher overheads for large word sizes and so constitute an explicit tradeoff towards area and energy efficiency. Overall, the results show that although peak improvements come at higher bit-widths, the architecture offers steady improvements at an average level in all configurations, and thus, it is suitable for energy and area constrained arithmetic units.

As shown in Figure 1 and Table 2, the 4-bit LZC design has a clear optimization over the traditional version in Ref.[19] The previous 4-bit LZC design used two 2-bit LZCs, three 2-input AND gates, one inverter, and a 2-input OR gate,

Table 2: Comparison Table for Number of Transistors in[19] and New 4-Bit LZCs

Design	*n*	Gates	Transistors	Total
[19]	2-bit LZC	2 input NOR	4	12
		inverter	2	
		2 input AND	6	
	4-bit LZC	2 * 2-bit LZC	2 * 12 = 24	50
		3 * 2 input AND	3 * 6 = 18	
		inverter	2	
		2 input OR	6	
new	4-bit LZC	4 input NOR	8	28
		2 input NOR	4	
		2 * inverter	2 * 2 = 4	
		multiplexer	12	

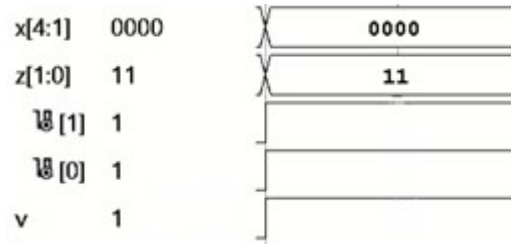

Fig. 10: Simulated waveforms of the 4-bit LZC.

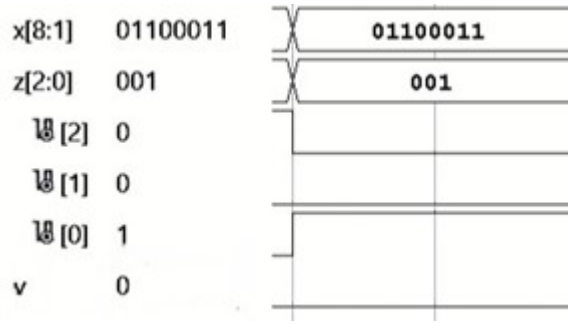

Fig. 11: Simulated waveforms of the 8-bit LZC

and used 50 transistors. The proposed design, on the other hand, saves hardware cost by implementing a 4-input NOR, a 2-input NOR, two inverters, and a multiplexer, requiring only 28 transistors. This represents a reduction of 22.

As shown in Figure 10, the simulated waveforms confirm the correct operation of the 4-bit LZC in all input transitions. The number of leading zeros in the input word is accurately represented in the number of leading zeros expressed in the output count and correctly expressed in the validity signal.

Simulation waveforms of the 8-bit leading-zero counter, as represented in Figure 11, support the appropriate hierarchical functionality that has been realized with the combination of fewer bit submodules. The obtained results indicate that the leading-zero count and signal selection are correct.

Waveform analysis verifies the functional correctness of the 16-bit leading-zero counter designed using a modular hierarchical approach, as illustrated in Figure 12, which demonstrates the appropriate response to various forms of input as well as being consistent in timing features.

The modeling of the 32-bit LZC shown in Figure 13 highlights the roles of effective level scaling and enabling signal transfer. The resultant waveforms confirm proper accounting for the leading-zero phenomenon, and the pair-to-pair timing geometry remains even across a wide range of input widths.

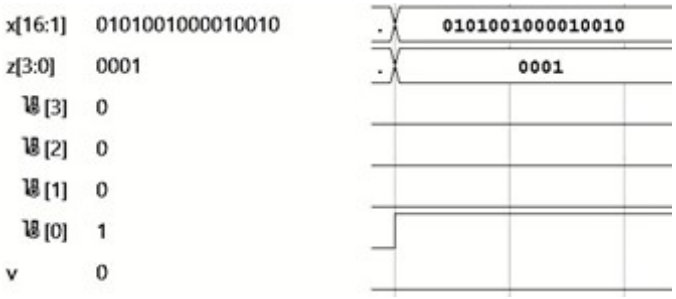

Fig. 12: Simulated waveforms of the 16-bit LZC

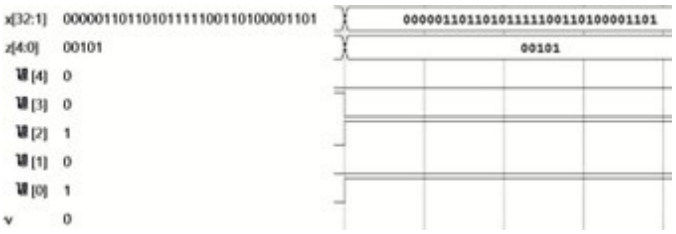

Fig. 13: Simulated waveforms of the 32-bit LZC.

Conclusion

A compact LZC architecture, which has been developed keeping scalability and modularity. The design with 4-bit LZC blocks and their hierarchical extensions, which support operand sizes of 8, 16, 32, and 64 bits and achieve consistent reductions in hardware resource utilization. Experimental results obtained using Xilinx Vivado and Cadence Genus (45 nm) demonstrate savings of up to 31.44% and power savings of up to 21%, with average improvements of 18–20% in area and 10–12% in power, and only with moderate increases in delay with respect to operand width. In the worst case, improvements remain within 1–3%, indicating a deliberate tradeoff favoring area and energy efficiency over absolute minimum delay. To minimize hardware footprint, the design guarantees a reduced transistor count, enabling the architecture to be used in space- and energy-sensitive devices such as IoT devices, embedded processors, and mobile systems. This scalable, high-performance LZC architecture enables the development of energy-efficient processors for next-generation computing platforms, contributing to sustainable industrial innovation.

The design has a slight delay overhead (especially with larger operand sizes), but the tradeoff heavily biases it in favor of applications that focus more on area and energy consumption. The current study may be enhanced in the future with fieldwork on delay-based optimizations, adaptive bit-width control of precision, and the implementation of the design to handle subnormal numbers and signed floating-point. The proposed LZC can be a viable solution for next-generation arithmetic and computing systems due to its compactness, energy efficiency, and scalability.

Acknoweldgement

The authors gratefully acknowledge Sri Mata Amritanandamayi Devi (AMMA), Chancellor, Amrita Vishwa Vidyapeetham, for her inspiration and for providing financial support for the Article Processing Charge (APC) of this publication.

References

1. H. Nair and A. Chalil, "FPGA implementation of area and speed efficient CORDIC algorithm," *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, 2022, pp. 512–518.
2. M. D. Ercegovac and T. Lang, *Digital Arithmetic*. San Francisco, CA: Morgan Kaufmann, 2004.
3. M. Schmookler and K. Nowka, "Leading zero anticipation and detection— A comparison," *IEEE Transactions on Computers*, vol. 49, no. 6, pp. 641–648, Jun. 2000.
4. A. Balachandar, A. Patel and S. R. Ramesh, "Design of a Vedic Multiplier based 64-bit Multiplier Accumulator Unit," 2024 5th International Conference on Innovative Trends in Information Technology (ICITIIT), Kottayam, India, 2024, pp. 1–7.
5. S. F. Oberman and M. J. Flynn, "Design issues in division and other floating-point operations," IEEE Transactions on Computers, vol. 51, no. 7, pp. 771–779, July 2002.
6. A and P. E. "Error resolving in SRAM emulated FPGA using CRC," *2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Jalandhar, India, 2023, pp. 647–651.
7. V. Sze, Y. Chen, T. Yang and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2290–2329, Dec. 2017.
8. S. N. S. Vishnu, A. Gandluru and R. S. R, "32-Bit RISC Processor Using VedicMultiplier," *2022 3rd International Conference for Emerging Technology (INCET)*, Belgaum, India, 2022, pp. 1-5, doi: 10.1109/INCET54531.2022.9824747.
9. M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," 2014 IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers, San Francisco, CA, USA, 2014.
10. K. Mai et al., "Low-power arithmetic design in nanoscale CMOS," IEEE Journal of Solid-State Circuits, vol. 42, no. 11, pp. 2400–2410, Nov. 2007.
11. Gregory Morse, Tamas Kozsik, and Peter Rakyta, "Minimal Path Delay Leading Zero Counters on Xilinx FPGAs," In Computational Science – ICCS 2023: 23rd International Conference, Prague, Czech Republic, July 3–5, 2023, Proceedings, Part III. Springer-Verlag, Berlin, Heidelberg, 626–640.
12. Ravindran and R. P, "FPGA implementation of approximate multiplier with edge detection application," *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2022, pp. 287–291.
13. N. Sriman and B. T. Sundari B, "VLSI Implementation of Stochastic MAC Unit for 2D Convolution," *2024 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, Mangalore, India, 2024, pp. 171–175.
14. M. Olivieri, F. Pappalardo, S. Smorfa, and G. Visalli, "Analysis and implementation of a novel leading zero anticipation algorithm for floating-point arithmetic units," IEEE Trans. Circuit Syst. II, Exp. Brief, vol. 54, no. 8, pp. 685–689, Aug. 2007.
15. P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, vol. C-22, no. 8, pp. 786–793, Aug. 1973.
16. G. Zhang, W.-W. Hu, and Z.-C. Qi, "Parallel error detection for leading zero anticipation," *Journal of Computer Science and Technology*, vol. 21, no. 6, pp. 901–906, 2006.
17. M. Saahithya, N. Hemanth, and M. Vinodhini, "Optimized leading zero count structure for arithmetic computations," *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, Greater Noida, India, 2024, pp. 1–5.
18. S. Ullah, T. D. A. Nguyen, and A. Kumar, "Energy-efficient lowlatency signed multiplier for FPGA-based hardware accelerators," IEEE Embedded Syst. Lett., vol. 13, no. 2, pp. 41–44, Jun. 2021.
19. S. Perri, F. Spagnolo, F. Frustaci, and P. Corsonello, "Design of leading zero counters on FPGAs," *IEEE Embedded Systems Letters*, vol. 15, no. 3, pp. 149–152, Sept. 2023.
20. L. M. G, B. Velivela, H. S. Shaik, and B. Pasam, "FPGA implementation of leading zero counter for floating point operations," *2024 4th International Conference on Artificial Intelligence and Signal Processing (AISP)*, Vijayawada, India, 2024, pp. 1–4.
21. J. L. V. R. Kumari, V. K. Kumar, M. Abhignya, and P. S. R. Krishna, " Implementation of leading zero count (LZC) for arithmetic and logic unit (ALU) operations," *2024 4th International Conference on Intelligent Technologies (CONIT)*, Bangalore, India, 2024, pp. 1–6.
22. J. Miao and S. Li, "A design for high speed leading-zero counter," *2017 IEEE International Symposium on Consumer Electronics (ISCE)*, Kuala Lumpur, Malaysia, 2017, pp. 81–82.
23. G. Dimitrakopoulos, K. Galanopoulos, C. Mavrokefalidis, and D. Nikolos, "Low-power leading-zero counting and anticipation logic for high-speed floating point units," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 7, pp. 837–850, Jul. 2008.
24. Zejia Lyu, Jizhong Shen, "An efficient algorithm for estimating gatelevel power consumption in large-scale integrated circuits," Microelectronics Journal, Vol. 146, 2024, 106143, ISSN 1879-2391.
25. Zahir, A. Ullah, P. Reviriego, and S. R. U. Hassnain, "Efficient leading zero count (LZC) implementations for Xilinx FPGAs," *IEEE Embedded Systems Letters*, vol. 14, no. 1, pp. 35–38, Mar. 2022.