Journal of Electronics and Informatics (ISSN: 2582-3825)
www.irojournals.com/iroei/

FPGA Based 32-Bit Hybrid Ripple Ling Carry Adder

Neenu Joseph[1], Ashker Assis[2], Arjun Bibin[3], Aromal A.[4], Thanzeel A R.[5]

[1]Associate Professor, [2-5]Student, Dept. of ECE, Albertian Institute of Science and Technology, Kerala, India

E-mail: [1]neenujoseph@aisat.ac.in, [2]ashkerassis1@gmail.com, [3]arjunbibin2003@gmail.com, [4]aromalanil32@gmail.com, [5]thanzeel072@gmail.com

Abstract

The 32-bit Hybrid Ripple Ling carry adder uses a Ling-based parallel prefix adder for the upper 16 bits and a ripple-carry adder (RCA) for the lower 16 bits to optimize performance, power, and area efficiency for VLSI designs. While the Ling adder speeds up carry propagation for larger bits, the RCA minimizes area and power for smaller bit-widths. This hybrid structure offers high-speed, low-power operation while saving 12% power, 30%–40% area, and a worst-case delay of 0.182 ns in a 28 nm process. The adder employs a Ling-based parallel prefix topology for the higher-order bits. A re-expression of the carry-lookahead technique is used to construct ling adders, which reduce the complexity of the carry-propagation process and allow for a higher speed carry calculation. The Ling adder's parallel prefix character suggests that multiple bits' carry signals are computed in parallel, significantly reducing the total delay. For the higher-order bits, where the carry chain length would normally be the dominant critical path, this speedup is especially important. The hybrid adder increases the global speed of the addition operation by reducing the critical path by using a Ling adder for the upper bits. The Xilinx Zynq-7000 SoC is used for the implementation.

Keywords: Ripple-Ling hybrid carry adder, Ripple-carry adder (RCA), Carry-lookahead adder, Ling-based parallel prefix adder, Carry propagation, Critical Path.

1. Introduction

Basic ripple-carry adders (RCAs) are one of the simplest and most uncomplicated adder structures. They are so because they calculate the sum and carry propagate from bit to bit in a

DOI: https://doi.org/10.36548/jei.2025.2.008
Received: 24.04.2025, received in revised form: 29.05.2025, accepted: 17.06.2025, published: 01.07.2025

simple manner, making them quite easy to implement using not many logic gates. RCAs have poor performance because carry propagation is sequential. In a ripple-carry adder, the carry-out of a bit addition is a function of the carry-in to the preceding bit. This implies that, for an N-bit adder, the carry must propagate through N stages, and hence there is a delay that grows linearly with the number of bits. Therefore, ripple-carry adders are inefficient for large word sizes where speed becomes an important issue. Conversely, the Ling adder offers a high-speed option by altering the carry-lookahead method. Ling adders redefine the carry propagation equations so that fewer operations are needed to calculate the carry signals. This yields quicker carry propagation and overall lower delay than ripple-carry adders. Nevertheless, this speed comes at a price: Ling adders take more gates, and more complex, and draw more power because of their parallel prefix design. They also take up more space on the chip, which makes them poor for designs where area and power efficiency matter. The secret to contemporary adder design is how to balance speed, power usage, and area. This is very difficult in leading nanometer process technologies, like 180 nm, 28 nm, or even 14 nm nodes.

As circuits shrink, power and area become increasingly important, but performance needs to be preserved to address the growing needs of high-performance computing and data processing. Hybrid adder architectures, such as the 32-bit Ripple-Ling adder, are where such requirements are met. The hybrid architecture leverages the strengths of both ripple-carry adders and Ling-based adders while overcoming their respective weaknesses. Particularly, the hybrid adder is divided into two parts: the lower bits are serviced by a ripple-carry adder while the higher bits are serviced by a Ling-based parallel prefix structure. In the lower bits, the application of a ripple-carry adder minimizes design complexity. As carry propagation through these bits does not dramatically affect the critical path delay, a straightforward ripple-carry structure is adequate. This also decreases the area and power consumption overall since ripple-carry adders take fewer gates, and switching activity is less compared to adders of higher complexity. In the higher bits, where the carry propagation delay matters more, the hybrid design incorporates a Ling-based parallel prefix adder. The parallel prefix structure computes the carry signals for the upper bits in a faster and more efficient manner, ensuring that the critical path is minimized.

The parallel prefix speed compensates for the lower-speed ripple carry structure in the lower bits, and thereby provides a high-speed overall adder without the unnecessary area and power overhead of employing a full Ling adder for all 32 bits. This hybrid design optimizes

contemporary applications, where power and area are limited, and high-speed performance is paramount. The hybrid Ripple-Ling adder structure enjoys considerable area savings of about 10% to 30% over a full Ling adder, thus constituting a better area-conscious alternative. In terms of power consumption, the hybrid design achieves power savings of about 10% due to the reduced utilization of the more power-hungry Ling structure compared to the straightforward ripple-carry structure. Along with area and power reduction, the hybrid adder is also efficient in terms of speed. The critical path delay is reduced because of the rapid carry calculation in the higher-order bits, so the hybrid adder to be used for high-performance VLSI designs. When utilized in complex processes such as the 28 nm node, the design attains a maximum delay of 0.183 ns demonstrating its ability to accommodate the speed demands of contemporary processors and digital systems. In summary, the hybrid Ripple-Ling adder exemplifies the trade-off between performance, power, and area that is desirable in contemporary digital circuit. By fusing the advantages of ripple-carry and Ling adders, this hybrid architecture achieves effective power and area utilization without loss of speed, which making it suitable for sophisticated VLSI systems in various applications.

2. Related work

Hybrid and optimized adder circuits have greatly improved to meet speed, power, and area requirements, which are the primary drivers in contemporary VLSI system design. Initial work such as Ling's high-speed binary adder [8], Sklansky's conditional-sum logic [9], and the Brent-Kung layout for parallel adders [10] introduced fundamental architectural concepts that are still utilized today. These layouts paved the way for parallel computation and effective carry propagation, reducing computation time in arithmetic units. Subsequently, studies shifted towards maximizing energy and performance in more compact technologies. Zlatanovici and Nikolic came forward with power-performance optimal 64-bit carry-lookahead adders (CLAs) [1] and further expanded on this with energy-delay optimization in CMOS designs [11], proving delay and power trade-offs. Augmenting this, Zeydel et al. concentrated on energy-efficient techniques for high-performance VLSI adders [7], extending the limits of low-energy designs. To address the need for power-efficient and high-speed architectures, a number of hybrid solutions have appeared. He and Chang proposed a hybrid carry-lookahead/carry-select adder for optimized power-delay product [2], whereas Bhattacharyya et al. presented a low-power, high-speed 1-bit full adder based on hybrid logic [3]. Likewise, Goel et al. focused on

energy efficiency and robustness in full adders based on hybrid CMOS logic styles [4], which enhanced process variation tolerance in deep-submicron nodes.

At the device level, Shams et al. compared various 1-bit CMOS full adder cells on power, delay, and energy parameters [5]. Mahmoud and Bayoumi previously proposed a small 10-transistor full adder with competitive performance [14], while Fang et al. presented a dynamic ripple-carry adder with less delay and area [15]. In contrast, Choi and Hwang investigated wave-pipelined structures at high clock speeds for ripple-carry adders [16], presenting the enhancements by timing control as opposed to structural modification.

Innovations have also reached new paradigms of computing. Kaushik and Bodapati recently introduced IMPLY-based carry and carry-select adders for in-memory computing [6], combining logic and storage to reduce data movement and delay. Morgenshtein et al. presented full-swing gate diffusion input logic-based CLAs for ultra-low power VLSI [12], whereas Pashaeifar et al. investigated approximate reverse carry propagate adders for DSP with a trade-off of precision for energy efficiency [13]. Overall, the discipline progressed from basic logic designs to sophisticated designs optimized for energy, delay, and scalability across a range of systems from deep-submicron CMOS to new logic-in-memory paradigms. All these attempts indicate a continued push toward adders that are responsive to the requirements of performance-critical and energy-constrained applications.

3. Methodology

In the implementation of a hybrid adder, the process starts with the acceptance of two 32-bit binary inputs, A and B. They are divided into two categories: lower-order and higher-order bits, which are processed separately in an effort to maximize both speed and efficiency. The adder is then separated into four different blocks of summation, where each block sums a particular range of bits. This separation allows for selective optimization in processing the different parts of the input data, trading off simplicity for lower-order bits and speed for higher-order bits.

The hybrid adder structure starts with the acceptance of two 32-bit binary input values, A and B. These values are divided into two groups: the lower-order bits and the higher-order bits. Lower-order bits, usually the initial 16 bits, are handled by a Ripple Carry Adder (RCA), while the higher-order bits, the other 16 bits, are handled by Ling Carry Logic. The reason for

this splitting is to enable the adder to maximize its performance for various parts of the input data, where the higher-order bits require speed and the lower-order bits benefit from simplicity. For the lower-order bits, the Ripple Carry Adder (RCA) is a simple and effective architecture that performs well with small bit-widths. The sum for each bit in this range is calculated by the equation.

$$Si = Ai \oplus Bi \oplus C(i-1) \quad (1)$$

Although simple, the RCA has carry propagation delays that are a concern when dealing with large bit widths. For the low-order bits, such delays are not a concern, and the ripple carry structure is sufficient.

For the high-order bits, the design resorts to Ling Carry Logic, a more sophisticated method of carry generation and propagation. Ling Carry Logic is a variant of conventional carry-lookahead logic, intended to reduce the delay of carry propagation by computing carry signals in parallel instead of sequentially. The most important characteristic of Ling Carry Logic is the generation of carry generate (G) and carry propagate (P) signals per bit, which are utilized to compute the carry for several bit positions at the same time. This carry computation in parallel considerably minimizes the delay incurred during carry propagation through the high-order bits and is therefore a suitable option where speed is of prime importance. The last step in the design of the hybrid adder is the employment of a multiplexer (MUX) to combine the outcomes of the low-order RCA and high-order Ling logic sections. The MUX chooses the corresponding carry values from both structures accordingly using the generated carry signals so that the carry information is correctly propagated over the whole 32-bit sum. This process ensures that the final sum is calculated correctly and that the carry propagation is consistent between the lower-bit and higher-bit sections.

The output of the hybrid adder is the ultimate 32-bit sum, generated by combining the sum outcomes of the low-order and high-order blocks. The implementation achieves a balance of speed, power, and area using RCA for the lower-order bits, where ease is adequate, and Ling Carry Logic for the upper-order bits, where speed is most important. Utilization of a MUX to control carry propagation between the two parts guarantees that the ultimate sum is calculated in an efficient manner without adding notable delays or complexity. In current VLSI (Very Large-Scale Integration) applications, where power consumption, area, and delay are to be minimized, this hybrid adder circuit is especially effective. Through judicious choice of the

right adder structure for varying regions of the bit-width, the design is a high-performance one that effectively manages large binary numbers without sacrificing the speed-resource trade-off. With the hybrid strategy, both the lower-order and higher-order bits are managed optimally, leading to a sturdy and efficient adder suitable for modern digital systems. Figure 1 depicts the block diagram of the suggested system.

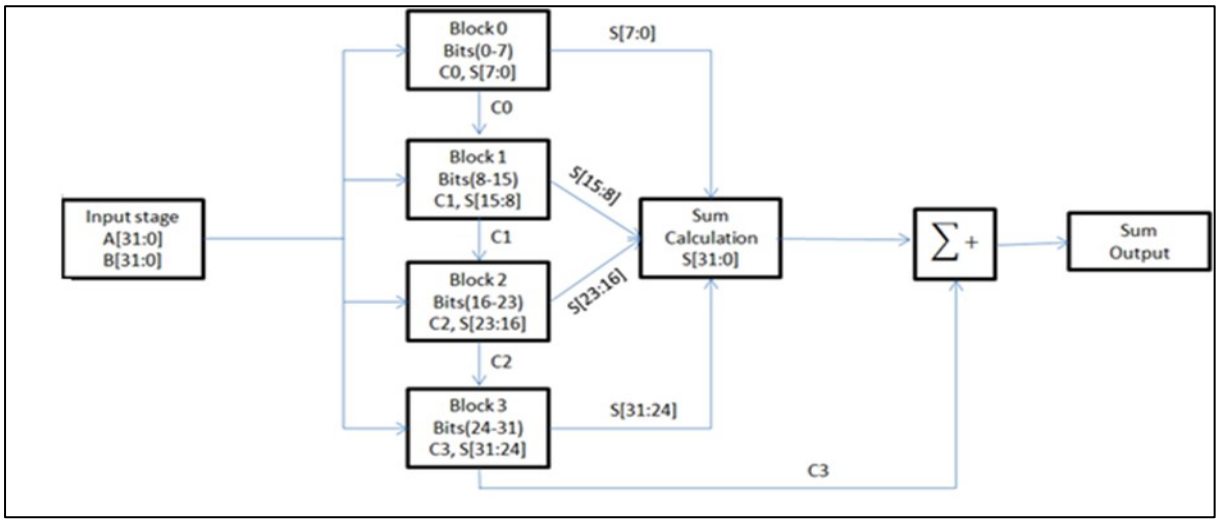

Figure 1. Block Diagram

Using a Ling Carry Logic for the high-order bits and a Ripple Carry Adder (RCA) for the low-order bits, the hybrid adder maximizes performance. This design accelerates carry propagation in the higher bits while decreasing delay in the lower bits, striking a balance between speed and simplicity. The multiplexer (MUX) optimizes speed, power, and area by integrating carry values from both sections, guaranteeing accurate carry propagation and effective computation of the final 32-bit sum.

3.1 Carry Generation

The block diagram illustrates a hierarchical 32-bit adder system aimed at calculating the sum of two 32-bit inputs, A [31:0] and B [31:0]. The design divides the inputs into two parts: low-order bits and high-order bits. This partitioning provides a parallel computation strategy, enhancing the efficiency of the addition operation as well as efficiently handling carry propagation. Stage one receives the low-order bits and inputs them into a ripple carry adder. The ripple carry adder calculates the partial sum of the lower half of the inputs and produces a ripple carry output, which is the carry signal generated from the low-order addition. This carry output is important since it is passed to the subsequent stage to be computed further.

The high-order bits are handled in stage two by a Ling carry adder, accepting as inputs both the high-order bits and the ripple carry output. The Ling carry adder is highly efficient in calculating the partial sum for the high-order bits while handling the carry input from the low-order calculation. The improved carry-handling mechanism saves time and ensures error-free performance in large bit-width addition. Lastly, the ripple carry adder and Ling carry adder partial sums are input into the final sum calculation block. The outcomes are aligned here, summed up, and finalized to generate the entire 32-bit sum output. Through the division of the addition process into several stages and the usage of dedicated adders, this architecture finds a compromise between speed, complexity, and the usage of resources to boost performance for 32-bit addition operations. The 32-bit Hybrid Ripple-Ling Carry Adder is an advanced arithmetic circuit aimed at improving addition speed with moderate hardware complexity.

The adder uses two proven techniques: Ripple Carry Adder (RCA) for low-order bits and Ling Carry Logic for high-order bits, thus achieving both power efficiency and delay as optimal. The traditional Ripple Carry Adder is straightforward, with the sum and carry of each bit being computed sequentially, carrying the carry from one stage to the next. However, as the width of the bits' increases, the delay also accumulates, making it inefficient for high-speed use. To alleviate this, the hybrid adder employs Ling Carry Logic for bits 11 through 31, significantly curtailing the delay by reengineering the mechanism of carry propagation.

Ling's approach modifies the conventional carry equation to generate intermediate carry signals to facilitate faster computation. Instead of having to wait for a ripple of a carry signal through several stages, the Ling Carry Adder calculates carry dependencies simultaneously, reducing the critical path delay considerably. With this improved structure, the high-order bits may be calculated efficiently without the long delays of ripple propagation. The choice of using an RCA for bits 0-10 and a Ling-based scheme for bits 11-31 is deliberate.

An RCA will be adequate for lower-order bits since the effect of delay is not quite as important. An RCA would, however, not be adequate for higher-order bits, where compounded delay has more of an effect. Ling's method prevents the calculation of carry from becoming a bottleneck at the expense of hardware complexity. Selective application in this way gives the Ripple-Ling Hybrid Carry Adder the best trade-off between hardware complexity and speed of computation. The final sum output, S[31:0], is generated by the combination of the partial sums of the two parts. The Ling Carry Logic generates the end carry signals quickly so that there is smooth fusion between the two parts. The structure is more efficient compared to the traditional

Ripple Carry Adder, making it applicable in high-speed computing applications such as microprocessors, digital signal processors (DSPs), and arithmetic logic units (ALUs).

With this hybrid approach, the adder benefits from the low power and simplicity of an RCA for the lower bits and the speed of Ling's approach for the higher bits. This makes the 32-bit Hybrid Ripple-Ling Carry Adder a suitable choice for contemporary computer systems that require high-speed arithmetic computations at a cost-effective hardware level. Its ability to minimize carry propagation delay without compromising the structure of logic optimization qualifies it as a design option for modern digital designs.

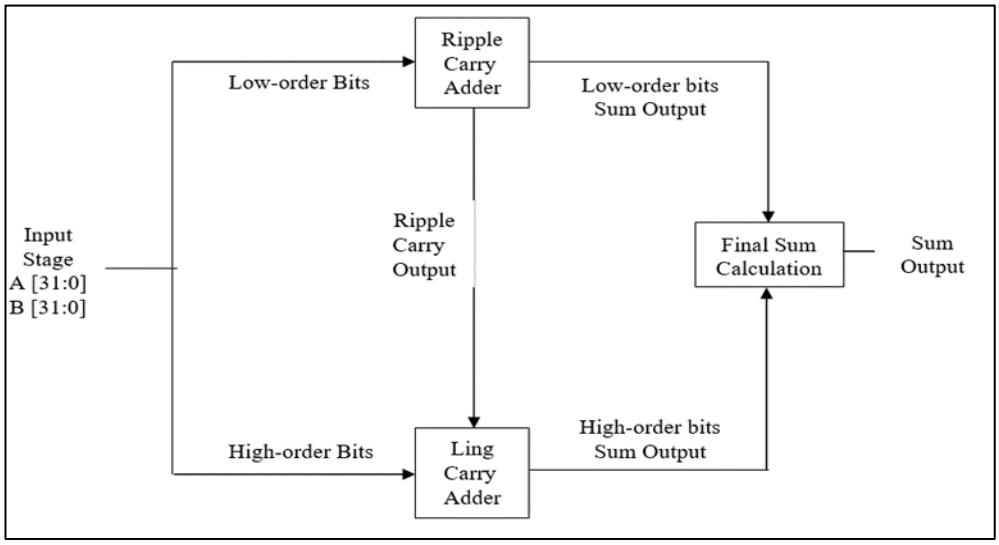

Figure 2. Hybrid Ripple Ling Carry Adder

The hybrid adder architecture depicted in Figure 2 divides the 32-bit inputs A and B into low-order and high-order bits. The low-order bits are processed by a ripple carry adder, while the high-order bits are handled by a ling carry adder. In the final sum calculation step, the outputs from the two adders are then combined to create the final sum output.

3.2 Hardware

The EDGE Spartan 6 FPGA board is one of the most capable and general-purpose development boards, featuring the Xilinx Spartan 6 FPGA at its core, which is highly reliable and economical.

The board comes with an impressive range of 26 external I/O pins, providing vast connectivity possibilities, making suitable for connecting with diverse peripherals like sensors,

actuators, and communication modules. These I/O pins can be configured as digital inputs or outputs, enabling smooth integration into a wide variety of embedded systems, control, and automation applications. The availability of USB UART and USB JTAG interfaces facilitates easy and efficient programming, debugging, and data transfer, giving developers the necessary tools for fast prototyping. One of the key strengths of the EDGE Spartan 6 FPGA board lies in its wireless communication capabilities.

It is WiFi and Bluetooth enabled, making it extremely versatile for IoT applications in which remote control, wireless data transmission, and monitoring are critical. These features enable developers to create connected systems that take advantage of cloud-based services and real-time data analysis, enhancing system efficiency and flexibility.

Besides high I/O capabilities, the board comes with SPI FLASH memory for non-volatile storage of bit streams and custom firmware. This provides persistence of data across power cycles, which is essential for applications that must run continuously. The presence of an ADC (Analog-to-Digital Converter) and DAC (Digital-to-Analog Converter) allows for analog signal processing, enabling the board to communicate with real-world analog inputs and outputs, such as sensors and actuators

3.3 Software

Xilinx Vivado is a complete and capable software package with which digital circuits and systems are developed for Xilinx FPGAs (Field-Programmable Gate Arrays) and SoCs (System-on-Chips) in particular. It offers a good collection of tools and features that highly simplify the design process, allowing engineers to easily design, simulate, and realize sophisticated hardware systems.

The user-friendly GUI simplifies FPGA design by providing an easier access platform for both professional and amateur engineers, facilitating a more efficient and accelerated process. Underneath, Vivado accommodates standard hardware description languages (HDLs) such as VHDL and Verilog, which are required to describe lower-level digital systems. This ensures that engineers with digital design experience in conventional digital design approaches can easily transition to FPGA design.

Through the creation of powerful synthesis, optimization, and debugging capabilities, Vivado guarantees that designs are effectively placed on Xilinx hardware with minimal

resource usage. Such optimization features are essential in areas where performance and efficiency are critical, including high-speed data processing, real-time systems, and embedded applications

4. Results and Discussion

The provided simulation output shows the function of a 4-bit Carry Lookahead Adder (CLA), which is an accelerated version compared to the Ripple Carry Adder (RCA). The CLA enhances speed in addition since it calculates the carry bits concurrently instead of cascading them step by step. The waveform presents input values `A[3:0]`, `B[3:0]`, and `Cin` along with the resulting sum `Sum[3:0]` and carry-out `Cout`.

At various instances of time, the input values vary, and their respective sum and carry-out values are calculated. At 40 ns, the inputs are `A = 1000 (8)`, `B = 0101 (5)`, and `Cin = 1`. The resultant sum is `8 + 5 + 1 = 14 (1110)`, which is observed as `Sum = 1110`. The carry-out `Cout = 1` establishes the fact that the addition exceeded 4-bit storage. Again, at 50 ns, for `A = 1111 (15)`, `B = 1111 (15)`, and `Cin = 1`, the sum is `15 + 15 + 1 = 31 (11111)`. `Sum` holds only the lower four bits (`1110`), and the fifth bit (`1`) is in `Cout`. In contrast with the RCA, where all the carries have to propagate sequentially from stage to stage, in the CLA, carry signals are precomputed, thus greatly reducing the delay. By precalculating carry signals, the Carry Lookahead Adder (CLA) output in Figure 3 shows faster computation. As can be seen, the carry-out is appropriately indicated, and the sum is 31 (11111) for A = 15, B = 15, and Cin = 1.

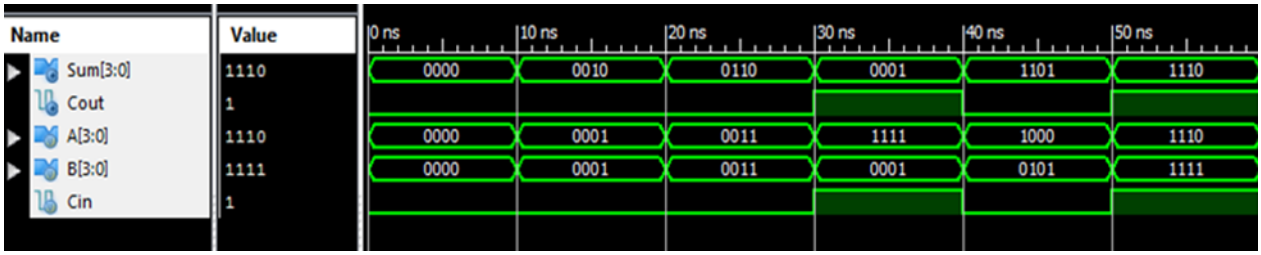

Figure 3. CLA

4.1 Ripple Carry Adder

The simulation output of a 4-bit Ripple Carry Adder (RCA), is a basic digital circuit that performs binary addition. Here, `a[3:0]` and `b[3:0]` are the two 4-bit input operands, and

`cin` is the carry-in bit. The outputs are `sum[3:0]`, a 4-bit sum, and `cout`, which is a carry-out in case the sum is more than four bits. At 0 ns, all the inputs are set to zero, so `sum = 0` and `cout = 0`. With the passage of time, various input values are used, and their respective sum and carry-out values are calculated. At **20 ns**, the inputs are `a = 3 (0011)`, `b = 3 (0011)`, and `cin = 0`, and the expected sum is `6 (0110)`, as shown in waveform figure 4. The carry-out is still `0`, since there is no overflow of more than four bits. At 40 ns, the inputs switch to `a = 8 (1000)`, `b = 5 (0101)`, and `cin = 0`.

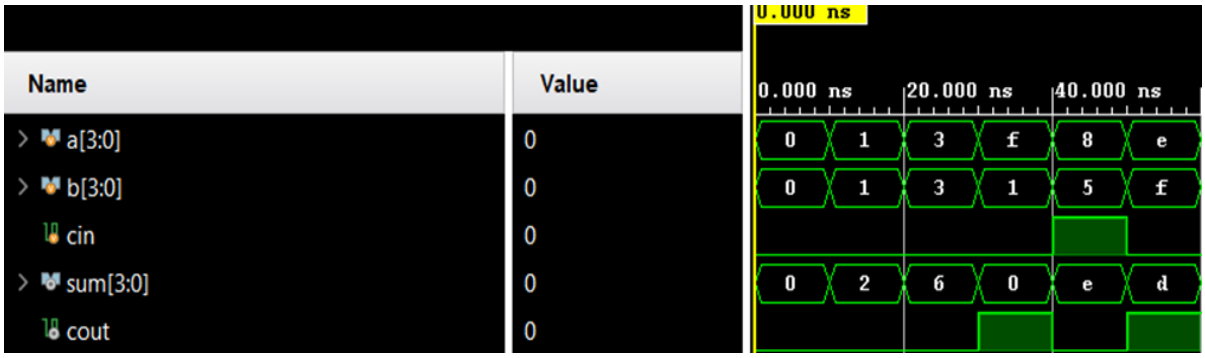

Figure 4. Ripple Carry Adder

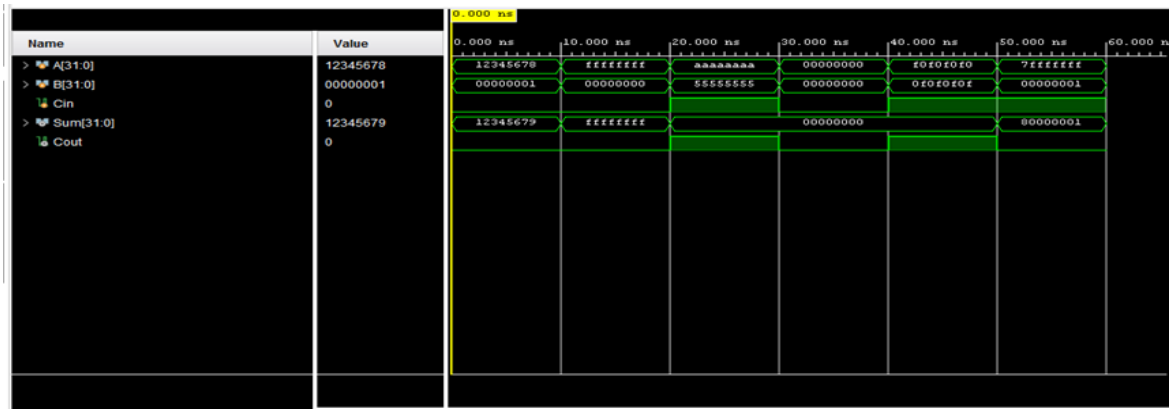

Figure 5. Hybrid Ripple Ling Carry Adder (HRLCA)

The given simulation output, Figure 5, demonstrates the working of a Hybrid Ripple-LingCarry Adder. It is a hybrid between the classical Ripple Carry Adder (RCA) and the Carry Lookahead Adder (CLA), intended to balance the speed vs. hardware complexity trade-off. The waveform illustrates the operation of this hybrid adder as it performs binary addition on two 32-bit inputs, A[31:0] and B[31:0], along with carry-in (Cin), and generates a 32-bit sum (Sum[31:0]) and carry-out (Cout).

The above-mentioned adders can be used in DSPs, microprocessors, cryptographic algorithms, low-power embedded systems and graphic processing units (GPUs). The

advantages of this adder include reduced critical path delay, area efficiency, power efficiency and cost efficiency ensuring versatility in a range of manufacturing technologies from traditional to advanced nodes (28 nm, 14 nm). The comparative analysis of three adders, CLA, RCA and hybrid adders are below in table 1.

Table 1. Comparative Analysis

SL No	Parameter	CLA	RCA	RLHCA
1.	Speed	Fast	Slow	Faster than CLA
2.	Power (μW)	24.324	22.548	21.638
3.	Area (μm^2)	273.6	254.7	232.5

5. Conclusion

The 32-bit Hybrid Ripple Ling Carry Adder introduces a new method to speed, power, and area optimization in digital circuits through the synthesis of ripple carry and Ling carry adders' strengths. The hybrid design is particularly suited to provide equal measures of performance and complexity for applications in need of both efficient and rapid processing, including high-end computing and digital signal processing (DSP). In this hybrid architecture, the adder splits the 32-bit word into high-order and low-order parts. The lower-order bits are processed through a ripple carry adder (RCA) structure, which is simple and has low resource requirements. RCAs carry the carry signal sequentially from one bit to another, which is adequate for smaller bit-widths where delay is not a significant issue. By using this form for the lower bits, the design minimizes complexity and conserves area and power since there are fewer logic gates needed.

The high-order bits, however, are computed through a Ling carry logic structure, which is an improvement over standard carry-lookahead adders. Ling logic alters how carry signals are produced and passed, allowing for quicker computation of carry signals in parallel across multiple bits. This is very useful in high-order bits where the carry propagation delay would otherwise constrain the entire performance of the adder. By reducing the delay of the critical path using Ling carry logic, the adder is made much faster while retaining a cost-effective design for the low-order bits.

The combination of these two techniques the simplicity of ripple carry for low-order bits and the speed of Ling carry for high-order bits allows the adder to strike an optimal balance between performance, power consumption, and area. The hybrid design not only speeds up the summation process but also decreases the total area by as much as 30%-40% over conventional Ling adders since the ripple-carry component of the design demands fewer interconnections and gates. Moreover, power utilization is decreased by around 12%, and this adder is best suited for today's Very-Large-Scale Integration (VLSI) designs where efficiency plays an important role and is synthesized on the Xilinx Zynq-7000 SoC.

References

[1] Zlatanovici, Radu, and Borivoje Nikolic. "Power-performance optimal 64-bit carry-lookahead adders." In ESSCIRC 2004-29th European Solid-State Circuits Conference (IEEE Cat. No. 03EX705), IEEE, (2003): 321-324.

[2] He, Yajuan, and Chip-Hong Chang. "A power-delay efficient hybrid carry-lookahead/carry-select based redundant binary to two's complement converter." IEEE Transactions on Circuits and Systems I: Regular Papers 55, no. 1 (2008): 336-346.

[3] Bhattacharyya, Partha, Bijoy Kundu, Sovan Ghosh, Vinay Kumar, and Anup Dandapat. "Performance analysis of a low-power high-speed hybrid 1-bit full adder circuit." IEEE Transactions on very large scale integration (VLSI) systems 23, no. 10 (2014): 2001-2008.

[4] SGoel, Sumeer, Ashok Kumar, and Magdy A. Bayoumi. "Design of robust, energy-efficient full adders for deep-submicrometer design using hybrid-CMOS logic style." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 14, no. 12 (2006): 1309-1321.

[5] Shams, Ahmed M., Tarek K. Darwish, and Magdy A. Bayoumi. "Performance analysis of low-power 1-bit CMOS full adder cells." IEEE transactions on very large scale integration (VLSI) systems 10, no. 1 (2002): 20-29.

[6] Kaushik, Nandit, and Srinivasu Bodapati. "IMPLY-based high-speed conditional carry and carry select adders for in-memory computing." IEEE Transactions on Nanotechnology 22 (2023): 280-290.

[7] Zeydel, Bart R., Dursun Baran, and Vojin G. Oklobdzija. "Energy-efficient design methodologies: High-performance VLSI adders." IEEE Journal of solid-state circuits 45, no. 6 (2010): 1220-1233.

[8] Ling, Huey. "High-speed binary adder." IBM Journal of Research and Development 25, no. 3 (1981): 156-166.

[9] Sklansky, Jack. "Conditional-sum addition logic." IRE Transactions on Electronic computers 2 (2009): 226-231.

[10] Brent, and Kung. "A regular layout for parallel adders." IEEE transactions on Computers 100, no. 3 (1982): 260-264.

[11] Zlatanovici, Radu, Sean Kao, and Borivoje Nikolic. "Energy–delay optimization of 64-bit carry-lookahead adders with a 240 ps 90 nm CMOS design example." IEEE Journal of Solid-State Circuits 44, no. 2 (2009): 569-583.

[12] Morgenshtein, Arkadiy, Viacheslav Yuzhaninov, Alexey Kovshilovsky, and Alexander Fish. "Full-swing gate diffusion input logic—case-study of low-power CLA adder design." Integration 47, no. 1 (2014): 62-70.

[13] Pashaeifar, Masoud, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram. "Approximate reverse carry propagate adder for energy-efficient DSP applications." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 26, no. 11 (2018): 2530-2541.

[14] Mahmoud, Hanan A., and Magdy A. Bayoumi. "A 10-transistor low-power high-speed full adder cell." In 1999 IEEE International Symposium on Circuits and Systems (ISCAS), vol. 1, IEEE, (1999): 43-46.

[15] Fang, Chih-Jen, Chung-Hsun Huang, Jinn-Shyan Wang, and Ching-Wei Yeh. "Fast and compact dynamic ripple carry adder design." In Proceedings. IEEE Asia-Pacific Conference on ASIC, IEEE, (2002): 25-28.

[16] Choi, H., and S. H. Hwang. "Design of wave-pipelined 900 MHz 16b ripple-carry adder using modified NPCPL." In 1996 IEEE International Symposium on Circuits and Systems. Circuits and Systems Connecting the World. ISCAS 96, vol. 4, IEEE, (1996): 182-184.