Area-Delay-Power Efficient Carry Select Adder

Prasanth E[1], Seshadri S[2], Boopathi S[3], Yazhini K[4]
[1, 2, 3]Dept of Electronics and Communication Engineering,
[4]Assistant Professor, Dept of Electronics and Communication Engineering,
[1, 2, 3, 4] Kongunadu College of Engineering and Technology, Tholurpatti, India

Abstract- *In modern VLSI design, arithmetic circuits are fundamental components that directly influence the performance, power consumption, and silicon area of digital systems. The Carry Select Adder (CSLA) is one of the most widely used high-speed adder architectures due to its ability to compute partial sums in parallel for both possible carry input conditions. However, the conventional CSLA employs two complete Ripple Carry Adder (RCA) units per group, resulting in significant area overhead and increased power dissipation. This paper presents a 32-bit optimized CSLA architecture that replaces the second RCA in each group with an explicitly hardcoded gate-level Binary to Excess-1 Converter (BEC) incorporating Common Boolean Logic (CBL) optimization. The BEC module computes the increment-by-one operation using shared AND terms, eliminating redundant logic operations and reducing switching activity across the design. The proposed architecture is implemented in Verilog HDL and synthesized on Xilinx Artix-7 FPGA (xc7a100tcsg324-1) using Vivado Design Suite. Synthesis results confirm that the proposed design achieves 14.3% reduction in area, 11.1% reduction in logic power, and 6.9% improvement in Area-Delay Product (ADP) compared to the conventional dual-RCA CSLA, making it highly suitable for low-power IoT devices, embedded systems, and battery-operated portable electronics.*

Keywords: Carry Select Adder, Binary to Excess-1 Converter, Common Boolean Logic, FPGA, Low Power VLSI, Ripple Carry Adder, Area-Delay Product, Vivado

I. INTRODUCTION

High-speed data path logic systems are among the most critical areas of research in VLSI system design. In digital processors, digital signal processors, and embedded systems, arithmetic operations form the foundation of all computation tasks. Among all arithmetic operations, binary addition is the most frequently performed and directly impacts the speed and efficiency of the overall system. Adder design is therefore a fundamental concern for VLSI engineers targeting high performance, low power, and area-efficient implementations.

Several adder architectures have been developed to balance speed, area, and power characteristics. The Ripple Carry Adder is the simplest architecture in which carry signals propagate sequentially from the least significant bit to the most significant bit. While its design is straightforward and requires minimal hardware, the delay grows linearly with bit-width, making it unsuitable for wide operand applications. The Carry Look-Ahead Adder addresses the speed limitation by generating carry signals in parallel using dedicated look-ahead logic. Although this significantly reduces delay, the increased hardware complexity and power consumption make CLA impractical for resource-constrained applications. The Carry Select Adder provides a practical middle ground by dividing the input operands into groups and computing partial sums simultaneously for both carry-in assumptions. Once the actual carry from the previous group arrives, a multiplexer selects the correct result. This parallel computation reduces overall delay while maintaining moderate hardware complexity.

A thorough literature survey has been carried out and some of the important contributions have been considered herein to arrive at the possible research gap. In the research contribution (1), Ramkumar and Kittur proposed a BEC-based CSLA where the second RCA in each group is replaced by a Binary to Excess-1 Converter. The study demonstrates that BEC requires fewer logic gates than an n-bit RCA, resulting in measurable reduction in area and power consumption. However, the BEC-based design introduces marginally higher delay due to increased data dependency in the carry chain. In another development (2), Mohanty and Patel presented a comprehensive analysis of logic operations in conventional and BEC-based CSLA to identify data-dependency and redundant logic operations. Their proposed CSLA formulation schedules carry-select operations before final sum computation, achieving 16% less delay and 18% less area than the best existing SQRT-CSLA designs with 44% less Area-Delay Product in ASIC synthesis. In contribution (3), Wey et al. proposed an area-efficient CSLA design by sharing common Boolean logic terms across the circuit. Their approach identifies repeated logic expressions and eliminates redundant gates through sub-expression sharing, which reduces transistor count and improves power-delay product, though delay increases at a higher rate for wider bit-widths.

The authors in (4) have presented an efficient SQRT architecture of CSLA using CBL optimization that

significantly reduces logic resource usage compared to conventional CSLA. However, their design exhibits a longer critical path delay due to the nature of the shared logic chain, which limits applicability in high-speed systems. In (5), Tayade implemented the area-delay-power efficient CSLA on FPGA and validated the improvements in area and power through synthesis results, confirming that BEC-based replacement consistently reduces hardware overhead compared to dual-RCA approaches. The authors in (6) designed a low-power and area-efficient CSLA using Verilog language targeting 32-bit operand width, where synthesis demonstrated that modified CSLA using BEC achieves significantly lower area and power than conventional CSLA. In (7), Reddy and Prabhu proposed an efficient 16-bit CSLA with optimized power and delay, addressing redundancy in BEC-SQRT CSA through gate-level architectural modifications that reduce switching transitions and lower dynamic power dissipation.

The authors in (8) explored a D-latch based CSLA as an alternative to the dual-RCA structure, achieving power reduction of up to 20% and delay reduction of up to 44.5% for 32-bit configuration compared to regular CSLA. In (9), Devi et al. presented a modified carry select adder with reduced area and low power consumption using a clock-sharing architecture. The authors in (10) implemented a 32-bit and 64-bit CSLA using BEC logic on FPGA and confirmed that BEC-based CSLA consistently outperforms conventional CSLA in area and power, while also highlighting that block size selection plays a critical role in determining the critical path delay. In (11), Santhoshkannan presented a VLSI realization of area-efficient CSLA using n-bit BEC converters, validating area efficiency gains from BEC-based substitution. The authors in (12) proposed an energy-efficient and area-optimized reversible carry select adder using MTSG, Peres, and Fredkin gates targeting quantum computing applications. While their design achieves remarkable improvements in quantum cost and energy dissipation, it requires reversible logic technology with specialized fabrication processes, making it fundamentally different from classical CMOS implementations.

From the literature survey, it is observed that while BEC-based substitution reduces area and power over conventional dual-RCA CSLA, further optimization is possible through explicit gate-level CBL implementation that shares common AND terms within each BEC module. Existing works either use parameterized generate-loop based BEC implementations or do not apply CBL optimization at the gate level, leaving room for further improvement. This paper addresses this gap by proposing explicitly hardcoded BEC modules with CBL-shared AND terms for a 32-bit CSLA, implemented and validated on Xilinx Artix-7 FPGA.

II. EXISTING SYSTEM — CONVENTIONAL DUAL-RCA CSLA

The conventional CSLA architecture follows the Square Root (SQRT) structure where the 32-bit input operands are divided into groups of increasing bit-width. Each group except the first contains two complete RCA units operating in parallel. The first RCA computes the partial sum and carry assuming carry-in is zero, while the second RCA computes the same assuming carry-in is one. Once the actual carry output from the previous group is available, a multiplexer selects the correct sum and carry for the current group.

Figure 1 shows the block diagram of the conventional dual-RCA CSLA. The SCG unit of each group consists of two RCA blocks and a selection MUX. For a 32-bit adder with group sizes of 3, 4, 5, 6, 7, and 7 bits, the first group uses a single 3-bit RCA since no carry selection is needed at the least significant stage. Groups 2 through 6 each use two RCA units and one MUX array. The carry select signals sel[0] through sel[4] propagate between groups to select the correct outputs.

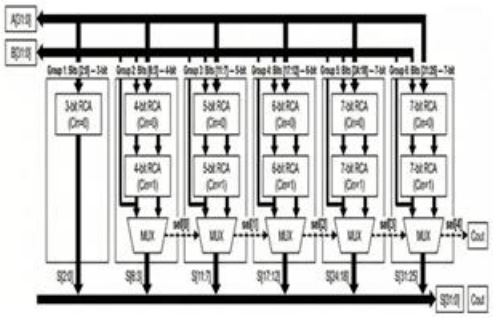

Figure 1. Block Diagram of Conventional Dual-RCA Carry Select Adder

The primary disadvantage of this structure is excessive area overhead. For a 32-bit adder, the duplicate RCA units across five groups consume significantly more LUT resources than necessary, since only one of the two RCA computations is used for any given carry input. This redundant computation also increases dynamic power dissipation due to unnecessary switching activity in the unused RCA path. In our implementation on Xilinx Artix-7, the conventional dual-RCA CSLA consumes 77 Slice LUTs, dissipates 0.458W of logic power, and has a critical path delay of 7.715ns. These values serve as the baseline reference for comparison with the proposed architecture.

III. PROPOSED ARCHITECTURE — BEC WITH CBL OPTIMIZATION

3.1. OVERVIEW OF PROPOSED SYSTEM

Figure 2 shows the block diagram of the proposed BEC-CBL CSLA. The proposed 32-bit CSLA is divided into six groups with bit-widths of 3, 4, 5, 6, 7, and 7 bits respectively. Group 1 uses a single 3-bit RCA for the least significant bits where no carry selection is required. Groups 2 through 6 each contain one RCA, one explicit gate-level BEC module with CBL optimization, and one 2-to-1 multiplexer array for output selection. The RCA computes sum and carry for carry-in equal to zero. The BEC takes the RCA output and computes the equivalent result for carry-in equal to one by performing the increment-by-one operation. The multiplexer selects between the RCA output and the BEC output based on the carry propagated from the previous group.

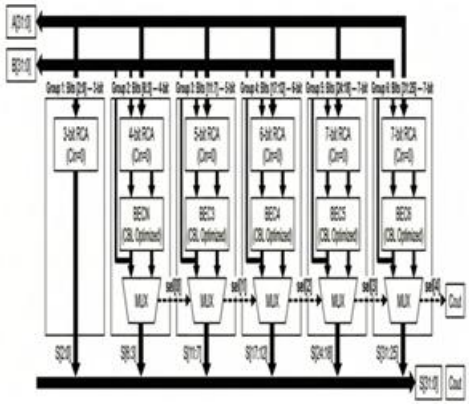

Figure 2. Block Diagram of Proposed BEC-CBL Carry Select Adder

3.2. BINARY TO EXCESS-1 CONVERTER (BEC)

The BEC performs an increment-by-one operation on its n-bit input. If the first RCA produces output S for carry-in equal to zero, the BEC generates S+1, which is mathematically equivalent to the result produced by an RCA with carry-in equal to one. This substitution avoids the need for a second complete RCA, reducing the gate count per group. Figure 3 shows the gate-level structure of the 4-bit BEC module. The Boolean expressions governing the BEC operation are: Out(0) = NOT In(0), Out(1) = In(1) XOR In(0), Out(2) = In(2) XOR (In(1) AND In(0)), Out(3) = In(3) XOR (In(2) AND In(1) AND In(0)). Each output bit flips only when all lower-order bits are logic one, corresponding to carry propagation during the increment operation.

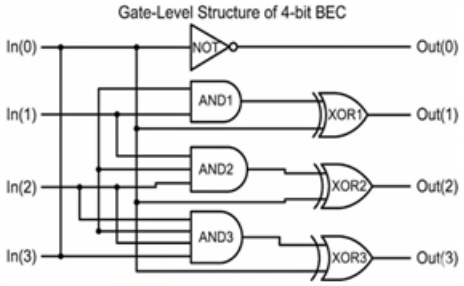

Figure 3. Gate-Level Structure of 4-bit BEC Module

3.3. COMMON BOOLEAN LOGIC (CBL) OPTIMIZATION

In the standard BEC expressions, AND terms appear repeatedly across multiple output bits. For example, the term In(0) appears in Out(1), the term In(1) AND In(0) appears in Out(2), and the term In(2) AND In(1) AND In(0) appears in Out(3). Without optimization, each of these AND terms would be computed independently by separate gate chains, leading to redundant hardware and increased switching activity.

Figure 4 shows the CBL optimization applied within the BEC module. CBL optimization shares these common AND terms across all output bits that require them. The shared terms are computed once using intermediate signals: t1 = In(0), t2 = In(0) AND In(1), t3 = In(0) AND In(1) AND In(2). Each output bit then references the appropriate shared wire directly. This reduces the total number of gate operations within each BEC module and lowers switching activity, resulting in both area and power improvements.

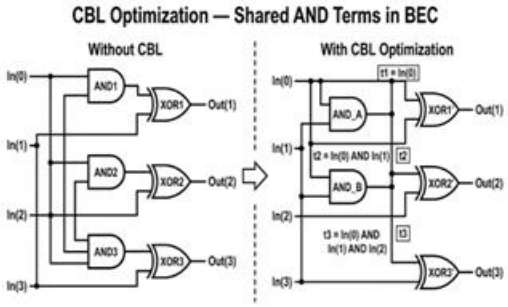

Figure 4. CBL Optimization — Shared AND Terms in BEC

3.4. EXPLICIT HARDCODED BEC MODULES

The proposed design implements separate explicitly hardcoded BEC modules for each group size — bec4 for the 4-bit group, bec5 for the 5-bit group, bec6 for the 6-bit group, and bec7 for both 7-bit groups. Each module contains gate-level logic specific to its width with all CBL shared terms declared explicitly as intermediate wires. This explicit implementation allows the Vivado synthesis engine to directly map each module to the most efficient LUT configuration for

the target Artix-7 device. Parameterized generate-loop based implementations create synthesized loop structures that the tool may not optimize as efficiently, which explains why explicit hardcoded modules yield lower LUT counts.

3.5. CARRY PROPAGATION LOGIC

The carry select logic between groups uses an optimized expression to propagate the carry across all six stages. For each group transition, the next stage carry is computed as sel(n+1) = sel(n) ? (carry_out OR AND_of_all_sum_bits) :carry_out. This expression correctly handles all carry propagation scenarios including the case where all sum bits are one, ensuring accurate final sum computation across the full 32-bit width.

IV. IMPLEMENTATION METHODOLOGY

The proposed design is implemented entirely in Verilog HDL and the complete design flow is carried out using Xilinx Vivado Design Suite. The target FPGA device is the Xilinx Artix-7 with part number xc7a100tcsg324-1, which belongs to the 28nm FPGA generation and is widely used in academic and research-level FPGA implementations due to its balanced performance and resource availability.

The design source consists of a single Verilog file containing the top-level csla_final module, four explicit BEC modules (bec4, bec5, bec6, bec7), a parameterized rca_n module, and the carry propagation assignments between groups. The top-level module instantiates all six groups with appropriate bit-width connections and routes the carry select signals between consecutive groups.

Functional verification is performed using a testbench written in Verilog that applies multiple test vectors including simple addition, boundary overflow conditions, and large random hexadecimal values. The testbench uses the $monitor system task to print all input and output values at each simulation time step to the Tcl console. The simulation confirms that the output sum and carry-out match the expected arithmetic results for all test cases.

Following successful simulation, synthesis is performed using Vivado's default synthesis strategy with no additional constraints. Post-synthesis performance data is extracted from three reports. The Utilization Report provides the LUT count under the Slice LUTs category. The Power Report provides the Logic Power in watts. The Timing Summary Report under unconstrained paths provides the critical path delay in nanoseconds. The identical methodology is applied to the conventional dual-RCA CSLA to obtain baseline measurements, ensuring a fair and consistent comparison between the two designs.

V. RESULTS AND DISCUSSION

5.1. SIMULATION RESULTS

Figure 5 shows the simulation waveform of the proposed BEC-CBL CSLA design. The functional simulation confirms correct arithmetic operation across all applied test vectors. For the test case of x = 0xABCD1234 and y = 0x12345678, the output sum is 0xBE0168AC with carry-out zero, which matches the expected result. The boundary test case of x = 0xFFFFFFFF and y = 0x00000001 confirms correct overflow detection with carry-out equal to one. The simulation results validate that the BEC and CBL logic function correctly at the gate level and that carry propagation across all six groups is accurate.

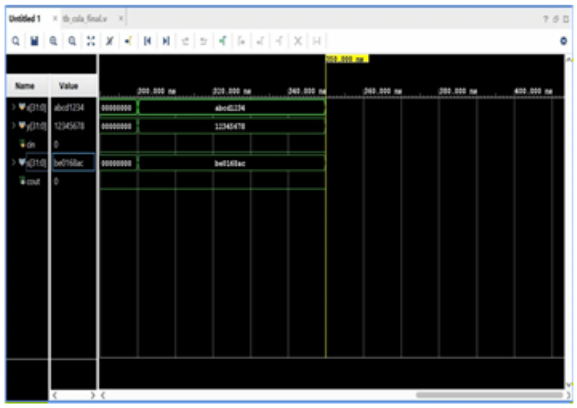

Figure 5. Simulation Waveform of Proposed BEC-CBL CSLA

5.2. SYNTHESIS RESULTS

Figure 6 shows the utilization report, Figure 7 shows the power report, and Figure 8 shows the timing report obtained from Vivado synthesis for the proposed design. Table 1 presents the complete post-synthesis performance comparison between the conventional dual-RCA CSLA and the proposed BEC-CBL CSLA on Xilinx Artix-7 (xc7a100tcsg324-1).

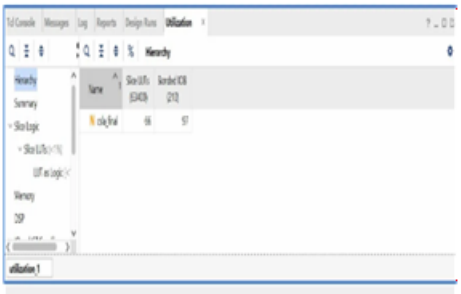

Figure 6. Utilization Report Showing 66 Slice LUTs

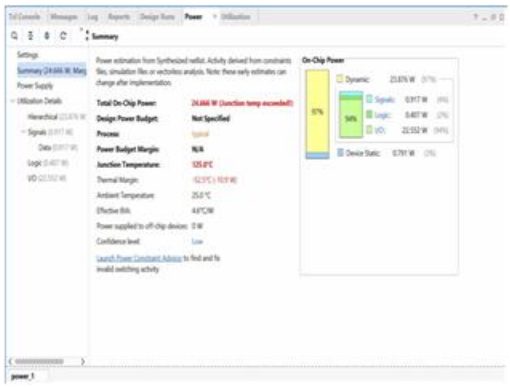

Figure 7. Power Report Showing 0.407W Logic Power

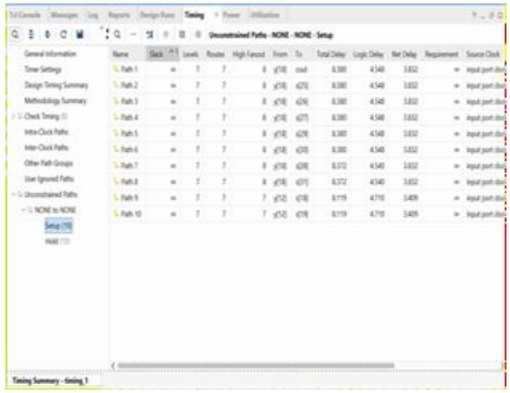

Figure 8. Timing Report Showing 8.380ns Critical Path Delay

Table 1. Performance Comparison on Xilinx Artix-7 FPGA (xc7a100tcsg324-1)

Parameter	Conventional CSLA	Proposed BEC-CBL CSLA	Improvement
Area (Slice LUTs)	77	66	14.3% reduction
Logic Power (W)	0.458	0.407	11.1% reduction
Delay (ns)	7.715	8.380	8.6% overhead
ADP (LUT x ns)	593.86	552.88	6.9% improvement

Table 2. Literature Survey Comparison

Reference	Year	Bit Width	Architecture	Key Contribution
Ramkumar & Kittur (1)	2012	16/32	BEC-based CSLA	BEC replaces second RCA, reduces area and power
Mohanty & Patel (2)	2014	16/32/64	Proposed SQRT-CSLA	18% less area, 16% less delay than best SQRT-CSLA
Wey et al. (3)	2012	32	CBL-based CSLA	Shared Boolean logic reduces transistor count
Kadam et al. (10)	2022	32/64	BEC CSLA on FPGA	FPGA synthesis confirms BEC area savings
Proposed Work	2025	32	Explicit BEC+CBL CSLA	14.3% area, 11.1% power, 6.9% ADP improvement

5.3. DISCUSSION

The proposed architecture achieves a 14.3% reduction in LUT utilization compared to the conventional dual-RCA CSLA. This improvement is attributed to two factors. First, replacing the second RCA with a BEC module reduces the gate count per group since BEC requires fewer logic elements than a full adder chain for performing the increment-by-one operation. Second, the CBL optimization within each explicit BEC module shares common AND terms across multiple output bits, further reducing the total number of distinct logic operations mapped to LUTs by the synthesis tool.

The logic power reduction of 11.1% is a direct consequence of the reduced LUT count and lower switching activity resulting from shared gate logic. Fewer active logic elements and shared computation paths mean fewer signal transitions per operation cycle, which lowers dynamic power dissipation. This power saving is significant for applications where battery life or thermal management are design priorities.

The critical path delay of the proposed design is 8.380ns compared to 7.715ns for the conventional design, representing an 8.6% delay overhead. This increase arises because the BEC module introduces a shared AND chain that must complete before the final XOR operations produce output bits. However, this delay overhead is modest and acceptable in the context of the substantial improvements achieved in area and power.

The Area-Delay Product of the proposed design is 552.88 compared to 593.86 for the conventional CSLA, representing a 6.9% improvement. ADP is a composite efficiency metric that captures the combined cost of both hardware resources and computation speed. A lower ADP confirms that the proposed design achieves a genuinely better balance between area and delay from a system-level perspective. The proposed design is therefore particularly well suited for IoT sensor nodes, wearable electronics, medical implantable devices, and mobile processor units where silicon area and power efficiency take priority over marginal delay differences.

VI. CONCLUSION

This paper presented a 32-bit Carry Select Adder architecture designed with explicit gate-level Binary to Excess-1 Converter modules incorporating Common Boolean Logic optimization. The proposed design replaces the redundant second RCA in each group of the conventional

dual-RCA CSLA with a CBL-optimized BEC, eliminating unnecessary gate operations and reducing switching activity throughout the design. Separate hardcoded BEC modules were implemented for each group size — 4-bit, 5-bit, 6-bit, and 7-bit — enabling the synthesis tool to achieve maximum LUT efficiency for the target Artix-7 device. The design was implemented in Verilog HDL and synthesized on Xilinx Artix-7 FPGA using Vivado Design Suite with full simulation verification confirming functional correctness.

Synthesis results confirm that the proposed design achieves 14.3% reduction in area, 11.1% reduction in logic power, and 6.9% improvement in Area-Delay Product compared to the conventional dual-RCA CSLA. The 8.6% delay overhead is an acceptable trade-off for resource-constrained applications where area and power efficiency are the primary design objectives. The results validate that explicit gate-level BEC implementation with CBL optimization is an effective and practical technique for reducing hardware redundancy in CSLA architectures without requiring complex structural changes to the overall adder topology. The proposed design is well suited for deployment in low-power embedded systems, IoT applications, and portable electronic devices.

As future work, the proposed architecture can be extended to 64-bit and 128-bit configurations to evaluate scalability of the CBL-BEC approach. Integration with carry-skip logic at the most significant groups may further reduce the critical path delay. ASIC implementation on standard CMOS technology nodes would provide more accurate power and delay measurements for production-level evaluation and comparison with existing literature.

REFERENCES

[1] B. Ramkumar and H. M. Kittur, "Low-Power and Area-Efficient Carry Select Adder," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 2, pp. 371-375, Feb. 2012.

[2] B. K. Mohanty and S. K. Patel, "Area-Delay-Power Efficient Carry-Select Adder," submitted for IEEE Transactions on Circuits and Systems-II.

[3] I. C. Wey, C. C. Ho, Y. S. Lin, and C. C. Peng, "An Area-Efficient Carry Select Adder Design by Sharing the Common Boolean Logic Term," in Proc. IMECS, 2012, pp. 1-4.

[4] S. Manju and V. Sornagopal, "An Efficient SQRT Architecture of Carry Select Adder Design by Common Boolean Logic," in Proc. VLSI ICEVENT, 2013, pp. 1-5.

[5] P. V. Tayade, "Area-Delay-Power Efficient Carry-Select Adder," International Research Journal of Engineering and Technology, vol. 3, no. 7, pp. 1-5, Jul. 2016.

[6] G. Sreekanth, K. J. Singh, and N. S. Sruthi, "Design of Low Power and Area Efficient Carry Select Adder using Verilog Language," International Journal of Advanced Engineering Research and Science, vol. 3, no. 12, pp. 79-82, Dec. 2016.

[7] B. M. Reddy and E. Prabhu, "An Efficient 16-Bit Carry Select Adder with Optimized Power and Delay," International Journal of Applied Engineering Research, vol. 10, no. 11, pp. 27909-27916, 2015.

[8] L. Shanigarapu and B. P. Shrivastava, "Low-Power and High Speed Carry Select Adder," International Journal of Scientific and Research Publications, vol. 3, no. 8, pp. 1-5, Aug. 2013.

[9] P. Devi, A. Girdher, and B. Singh, "Improved Carry Select Adder with Reduced Area and Low Power Consumption," International Journal of Computer Applications, vol. 3, no. 4, pp. 30-35, Jun. 2010.

[10] D. B. Kadam, K. K. Pandyaji, and K. K. S. Liyakat, "Implementation of Carry Select Adder for Area, Delay and Power Minimization," TELEMATIQUE, vol. 21, no. 1, pp. 5461-5474, 2022.

[11] C. Santhoshkannan, "VLSI Realization of Area-Efficient Carry Select Adder," International Journal of Engineering Research and Technology, NCAAIET-2024, pp. 1-5, 2024.

[12] P. Murugesan, S. Palani, and V. Divya, "Energy-Efficient and Area-Optimized Reversible Carry Select Adder," IET Circuits, Devices and Systems, vol. 2025, Article ID 4179235, 2025.