Design of a Power-Efficient Carry Select Adder Using Reversible Logic and Online Testing

A. Vamshidhar Reddy[1], Jahnavi Kommireddy[2], M. Jyothi[3], K. Divya Sree[4], Chikatla Sriram[5]

[12345] Department of ECE, CMR Technical Campus, Hyderabad, Telangana, India

Corresponding mail: jahnavikommireddy1@gmail.com

ABSTRACT: Arithmetic addition is a fundamental operation in digital systems, and its delay, area, and power requirements influence overall circuit performance. A conventional Carry Select Adder (CSLA) improves speed by computing results for alternative carry-in values in parallel, but duplicated adder hardware can increase area and power. This paper presents the project design of a power-efficient CSLA that combines hybrid adder structures with reversible logic and an online parity-based error-detection mechanism. The architecture described in the project uses Carry Lookahead Adder (CLA) logic for lower-order sections and Kogge–Stone Adder (KSA) logic for higher-order sections, alongside reversible gates including Peres, Toffoli, and Feynman gates. The design workflow covers RTL development, simulation, synthesis, physical-design analysis, and verification of arithmetic outputs and parity-based fault indications. The source project reports qualitative improvements in propagation delay and power efficiency and discusses the associated area trade-off. Numerical performance values are not stated here because the source report does not provide a consistent, directly comparable summary table.

Keywords— *Carry Select Adder, Reversible Logic, Carry Lookahead Adder, Kogge–Stone Adder, Online Testing, Parity Checking, Low-Power VLSI, Cadence.*

1. INTRODUCTION

Modern processors, digital signal-processing units, communication systems, and embedded devices depend on fast arithmetic circuits. Addition is used in many datapath operations, so adder architecture has a direct influence on circuit timing and energy use. A Ripple Carry Adder (RCA) is simple but has a carry path that grows with the number of bits. A Carry Select Adder reduces this waiting time by calculating candidate results for carry-in values of zero and one in parallel, then selecting the result corresponding to the actual carry.

The parallel computation in a conventional CSLA comes at the cost of duplicated hardware and can increase area and power. The project therefore investigates a hybrid CSLA architecture that combines faster carry computation with reversible logic and online checking. The stated goal is to improve propagation delay while keeping area overhead controlled and supporting fault detection during operation.

1.1 Project Objectives

The main objectives are to study conventional CSLA limitations; reduce carry-propagation delay through CLA and KSA structures; explore reversible gates for arithmetic and selection functions; evaluate area, power, and timing through the design tool flow; and include parity-based online checking to indicate faults during circuit operation.

1.2 Problem Statement

Conventional CSLA designs trade speed for duplicated adder units, increasing hardware complexity, area, and power. The design challenge is to reduce critical-path delay without allowing resource use to grow unnecessarily, while also checking output integrity during normal operation.

2. LITERATURE SURVEY

The project report identifies several established directions for adder optimization. Ripple Carry Adders provide a compact baseline but suffer from serial carry propagation. Conventional CSLA architectures improve speed through parallel candidate calculations, although duplicated blocks increase area. Binary-to-Excess-1 Converter (BEC)-based approaches can reduce duplicated hardware, but may add delay in some configurations. Carry Lookahead Adders compute carries using generate/propagate relationships, while Kogge–Stone adders use parallel-prefix structures to reduce carry depth.

Hybrid designs assign different adder structures to different bit groups to balance speed and resource use. Reversible logic, motivated by the relationship between information loss and energy dissipation, uses gates with one-to-one input/output mappings. The project also draws on parity-based checking as a way to detect inconsistencies in circuit outputs. These approaches introduce design complexity and must be assessed together rather than treating speed, area, power, and testability as independent goals.

3. PROPOSED ARCHITECTURE

The proposed design is a hybrid Carry Select Adder using reversible logic. As described in the project report, CLA logic is assigned to lower-order bit positions and Kogge–Stone logic to higher-order positions within the CSLA framework. The partitioning is intended to shorten the critical carry path while retaining a modular structure.

Each CSLA block calculates two candidate results using assumed carry-in values of zero and one. When the carry from the preceding block becomes available, selection logic forwards the appropriate sum and carry. In the proposed reversible implementation, selection and arithmetic functions use reversible gates such as Peres, Toffoli, and Feynman gates. This design approach is intended to preserve information and support the project's online-testing mechanism.

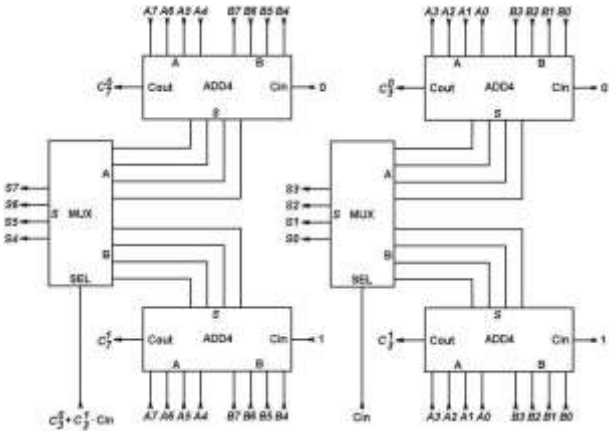


Fig. 1. Carry Select Adder block structure shown in the source project.

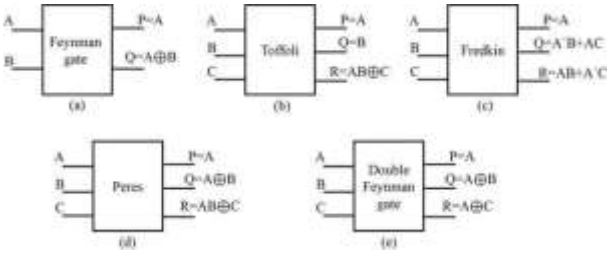


Fig. 2. Reversible gate structures referenced in the proposed design.

3.1 Reversible Logic and Online Testability

Reversible logic maps input combinations uniquely to output combinations. The source project discusses Peres, Toffoli, and Feynman gates for constructing arithmetic and selection functions. The online-checking approach uses parity preservation/checking to identify an unexpected output condition during operation. The parity checker provides an error indication; it is not, by itself, a guarantee that every possible multiple-bit fault will be detected.

3.2 Design Trade-offs

Parallel candidate computation and advanced carry structures can reduce delay, but duplicated adder blocks and additional reversible/checking logic can increase area. The intended design target is therefore a balanced power-performance-area result, with the trade-offs examined through simulation and synthesis reports rather than assuming that every metric improves simultaneously.

4. DESIGN AND IMPLEMENTATION METHODOLOGY

The project describes an RTL-based implementation and verification flow using Cadence simulation and synthesis tools, together with Xilinx ISE Design Suite 14.7 in the implementation workflow. The process begins with defining the arithmetic behavior and reversible-gate structure, followed by RTL coding, testbench preparation, functional simulation, synthesis, and inspection of implementation reports.

Simulation applies input combinations and edge conditions to check that sum and carry outputs match expected arithmetic behavior. The online test logic is also evaluated through parity-related output behavior. Synthesis and physical-design stages provide information about hardware utilization, power estimates, timing, placement, and routing. These stages allow the design to be examined beyond functional correctness.

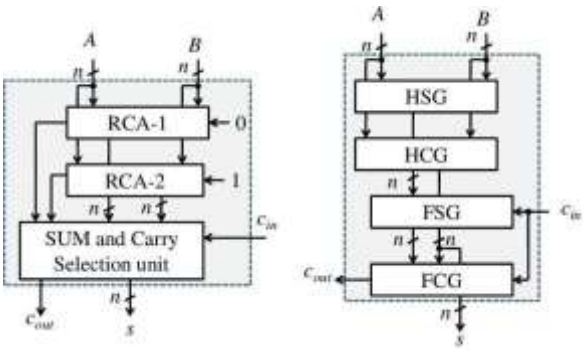


Fig. 3. Design/simulation environment screenshot included in the source report.

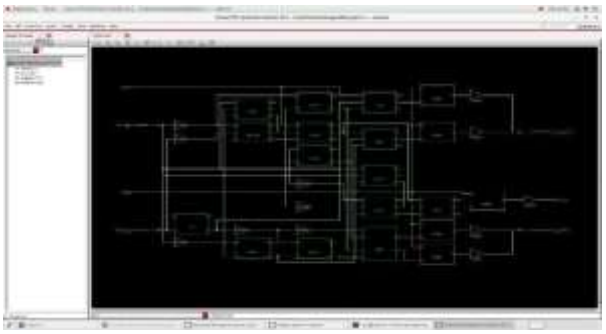


Fig. 4. Physical-design view included in the source report.

5. RESULTS AND DISCUSSION

The source report presents simulation waveforms, synthesis/area and power reports, and a physical layout view as evidence used to assess the design. It states that the simulated outputs match expected results for the tested input combinations and that output transitions remain stable in the examined scenarios. It also describes parity-based checking as a means of flagging faults during operation.

The report qualitatively attributes reduced propagation delay to the hybrid carry architecture and discusses lower power dissipation as a motivation and reported outcome of reversible logic. It also acknowledges a moderate area increase associated with multiple adder blocks and additional logic. Because the original report does not present a clear consolidated comparison table with baseline and proposed numerical values under the same conditions, this paper does not claim a specific percentage improvement or provide unverified numerical measurements.

Generated by: Genus(TM) Synthesis Solution 20.11-s111_1
Generated on: Jul 16 2024 11:20:24 am
Module: pro
Operating conditions: fast (balanced_tree)
Wireload mode: enclosed
Area mode: timing library

Instance	Module	Cell Count	Cell Area	Net Area	Total Area	Wireload
pro		75	0.000	0.000	0.000	<none> (D)
c1	cnot	5	0.000	0.000	0.000	<none> (D)
c3	cnot_7	5	0.000	0.000	0.000	<none> (D)
c8	cnot_6	5	0.000	0.000	0.000	<none> (D)
c10	cnot_4	5	0.000	0.000	0.000	<none> (D)
c12	cnot_3	5	0.000	0.000	0.000	<none> (D)
f1	fre	3	0.000	0.000	0.000	<none> (D)
f2	fre_10	3	0.000	0.000	0.000	<none> (D)
f3	fre_9	2	0.000	0.000	0.000	<none> (D)
f4	fre_8	3	0.000	0.000	0.000	<none> (D)
p1	per	1	0.000	0.000	0.000	<none> (D)
p2	per_3_12	1	0.000	0.000	0.000	<none> (D)
p3	per_11	1	0.000	0.000	0.000	<none> (D)
p4	per_3	1	0.000	0.000	0.000	<none> (D)
z1	ccu	18	0.000	0.000	0.000	<none> (D)
h1	tof	2	0.000	0.000	0.000	<none> (D)
h2	tof_15	2	0.000	0.000	0.000	<none> (D)
h3	tof_14	2	0.000	0.000	0.000	<none> (D)
h4	tof_13	2	0.000	0.000	0.000	<none> (D)
h5	tof_5_16	5	0.000	0.000	0.000	<none> (D)
h6	tof_5	5	0.000	0.000	0.000	<none> (D)

(D) = wireload is default in technology library

Fig. 5. Area-analysis report reproduced from the source project.

Instance: /pro
Power Unit: W
PDB Frames: /stim#0/frame#0

Category	Leakage	Internal	Switching	Total	Row%
memory	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
register	5.55915e-09	4.29346e-05	2.97003e-06	4.59102e-05	8.81%
latch	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
logic	5.35106e-08	6.94941e-05	4.02902e-04	4.72450e-04	90.69%
bbox	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
clock	0.00000e+00	0.00000e+00	2.61360e-06	2.61360e-06	0.50%
pad	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
pm	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
Subtotal	5.90697e-08	1.12429e-04	4.08486e-04	5.20974e-04	100.00%
Percentage	0.01%	21.58%	78.41%	100.00%	100.00%

Fig. 6. Power-analysis report reproduced from the source project.

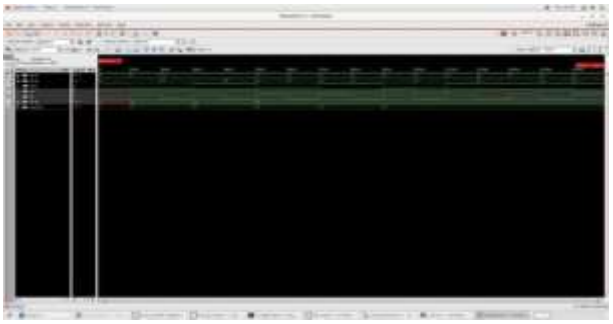


Fig. 7. Simulation output screenshot reproduced from the source project.

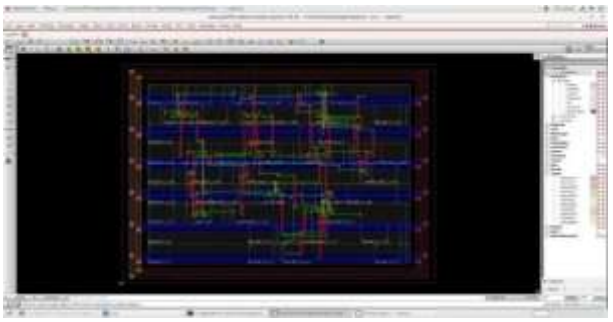


Fig. 8. Design/layout screenshot reproduced from the source project.

5.1 Interpretation

The reported evidence supports functional verification and a qualitative discussion of timing, power, and area. A stronger quantitative evaluation for publication would include the exact bit width, target device/library, clock or timing constraints, synthesis settings, and a table comparing RCA, conventional CSLA, BEC-based CSLA, and the proposed architecture using the same tool conditions. These details are not consolidated in the supplied report, so they are identified here as reporting limitations rather than filled with assumed values.

6. CONCLUSION AND FUTURE SCOPE

This paper has summarized a power-efficient Carry Select Adder project that combines a hybrid CLA/Kogge–Stone architecture with reversible logic and parity-based online checking. The approach addresses the delay and resource-use limitations of conventional CSLA designs while adding a mechanism intended to identify output inconsistencies. The source project describes successful functional simulation and qualitative benefits in propagation delay and power, alongside a potential area overhead from parallel blocks and checking logic.

Future work can provide a more complete quantitative comparison across baseline architectures, optimize reversible gate count and block partitioning, extend testing to larger bit widths, and evaluate the parity mechanism against defined single- and multiple-fault models. Hardware validation and implementation on newer platforms could further clarify practical performance and scalability.

ACKNOWLEDGMENT

The authors express their sincere gratitude to the Department of Electronics and Communication Engineering, CMR Technical Campus, Hyderabad, Telangana, India, for providing the facilities and academic support for this project. The source project also acknowledges the guidance of Mr. A. Vamshidhar Reddy, Assistant Professor, and support from the department and institution.

REFERENCES

[1] R. Landauer, “Irreversibility and Heat Generation in the Computing Process,” IBM Journal of Research and Development, vol. 5, no. 3, pp. 183–191, 1961.

[2] C. H. Bennett, “Logical Reversibility of Computation,” IBM Journal of Research and Development, vol. 17, no. 6, pp. 525–532, 1973.

[3] M. Morris Mano and M. D. Ciletti, Digital Design, 5th ed. Pearson Education, 2013.

[4] J. M. Rabaey, A. Chandrakasan, and B. Nikolić, Digital Integrated Circuits: A Design Perspective, 2nd ed. Pearson, 2003.

[5] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs. Oxford University Press, 2000.

[6] H. Thapliyal and M. B. Srinivas, “Novel Reversible Logic Gate and Its Application,” Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2005.

[7] E. Fredkin and T. Toffoli, “Conservative Logic,” International Journal of Theoretical Physics, vol. 21, pp. 219–253, 1982.

[8] M. Nielsen and I. Chuang, Quantum Computation and Quantum Information. Cambridge University Press, 2000.

[9] Xilinx Inc., Xilinx ISE Design Suite 14.7 User Guide, 2013.