IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

8-Bit Ripple Carry Adder Area, Power And Delay Analysis

Muktha S Patil, Aishwarya N Joshi, Ankita M Kunchaganoor, Chandrika Ashok

Assistant Professor, Student, Student, Student
Department of Electronics and Communication
Tontadarya College of Engineering, Gadag, Karnataka, India

Abstract **-** This paper presents an in-depth evaluation of an 8-bit Ripple Carry Adder (RCA) architecture, a fundamental building block in digital arithmetic operations. The design is implemented in Verilog and synthesized using Cadence Genus, targeting an ASIC-based workflow. Key performance parameters—including power consumption, chip area, and propagation delay—are extracted and analysed. While the RCA architecture benefits from a straightforward structure and energy efficiency, it exhibits higher latency due to serial carry propagation. The study provides practical insights into the trade-offs inherent in RCA-based adder design, especially for low-power or area-sensitive VLSI applications.

Index terms **- Ripple Carry Adder (RCA), Power Analysis, Area Utilization, Timing Optimization**

I. INTRODUCTION

Adders play a critical role in digital circuits, serving as core components in ALUs, CPUs, and data processing units. The Ripple Carry Adder (RCA) is one of the simplest multi-bit adder designs, constructed by connecting individual full adder units in a chain. In this configuration, each bit's carry output is passed to the next stage, resulting in a delay that scales linearly with bit-width.

Despite its relatively slow performance, the RCA remains a popular choice in designs where simplicity, area efficiency, and low power consumption are priorities. With growing demand for optimized digital systems in portable and embedded environments, understanding the behaviour and trade-offs of RCA-based adders becomes essential. This paper focuses on designing and analyzing an 8-bit RCA using a standard ASIC flow, employing Cadence Genus for synthesis and performance extraction.

II. RELATED WORK

The Ripple Carry Adder (RCA) remains one of the most fundamental and widely used adder architectures in digital system design, primarily due to its structural simplicity and ease of implementation. It is commonly employed in arithmetic logic units (ALUs) and embedded processors, particularly in applications where power efficiency and minimal area are more critical than speed.

Several studies have explored ways to enhance the RCA's performance characteristics. For example, [1] proposed a delay-line clocking strategy aimed at improving RCA speed without significantly increasing power consumption. This approach demonstrated the impact of clock management and control logic on traditional adder efficiency.

Further investigations into RCA optimization at the circuit level are presented in [2], where various full adder implementations were analysed to minimize energy consumption while preserving functional

correctness. These studies highlight how even slight improvements in full adder design can influence the overall performance of an RCA.

Technology-specific evaluations have also been conducted. For instance, [3] examined the behaviour of RCA designs in advanced technology nodes such as 14nm and 45nm. The results showed that while RCAs maintain their advantage in power and area, the propagation delay becomes increasingly problematic as feature sizes shrink.

In [4], the use of FinFET-based logic in RCA implementation demonstrated that transistor-level advancements could help mitigate some timing limitations, though not eliminate them entirely. Research like [5] assessed the RCA's role in ultra-low power applications, especially under subthreshold voltage operation, reaffirming its viability in energy-constrained environments such as IoT devices and wearable electronics.

Approximate computing techniques were applied to RCA designs in [6], where slight compromises in computational accuracy led to substantial gains in power efficiency and speed. This is particularly relevant in signal processing and machine learning accelerators, where perfect accuracy may not always be required.

FPGA and ASIC implementation studies, such as those in [6] and [6], compared RCA synthesis outcomes across different platforms. These works emphasized the influence of
synthesis tools, physical libraries, and switching activity on the adder's final performance metrics.

Despite ongoing research, many prior studies either focus on sub-components of the RCA or analyse it alongside other adder architectures. Few have presented a dedicated, synthesis-driven evaluation of an 8-bit RCA using standardized ASIC flows. This study addresses that gap by providing a complete design-to-silicon evaluation of an 8-bit Ripple Carry Adder using Cadence Genus, highlighting its practical characteristics in terms of area, power, and delay.

III. ARCHITECTURE OVERVIEW

In this section, the structural differences between the Ripple Carry Adder (RCA) and the Carry Lookahead Adder (CLA) are discussed, focusing on their design principles, scalability, and performance characteristics.

A. Ripple Carry Adder (RCA)

The Ripple Carry Adder is the most fundamental and straightforward form of a multi-bit adder, constructed by cascading multiple full adders in sequence. In an 8-bit RCA, eight full adders are connected in series, where the carry output from each stage becomes the carry input to the next. This sequential propagation of the carry introduces a cumulative delay, which increases linearly with the number of bits, making the design slower for higher bit-widths. Despite this drawback, the RCA is highly area-efficient and simple to implement, which makes it suitable for low-power and small-area applications. It is often used in systems where speed is not the primary concern, but design simplicity and resource utilization are critical..

Each full adder computes:

- **Sum**: Sum = $Ai \oplus Bi \oplus Cin$
- **Carry**: $\text{Cout} = (\text{Ai} \cdot \text{Bi}) + (\text{Bi} \cdot \text{Cin}) + (\text{Ai} \cdot \text{Cin})$

While the RCA is easy to implement and occupies less area, the carry propagation delay through all stages limits its speed for wider bit-widths.

C. Design Implementation

The 8-bit Ripple Carry Adder (RCA) was implemented in Verilog HDL and synthesized using Cadence Genus for ASIC design evaluation. The design consisted of eight cascaded full adders, where each stage generated a sum and propagated the carry forward.

A Verilog testbench was used to verify functional correctness through simulation. Switching activity was recorded using VCD (Value Change Dump) files for accurate power estimation during synthesis.

Synthesis was performed with a 45nm standard cell library, applying consistent constraints such as clock period and I/O drive strength. Post-synthesis reports provided key metrics, including area, power, and delay, which were analySed to evaluate the RCA's efficiency and propagation delay characteristics.

Figure-1 shows the block diagram of 8-bit RCA:

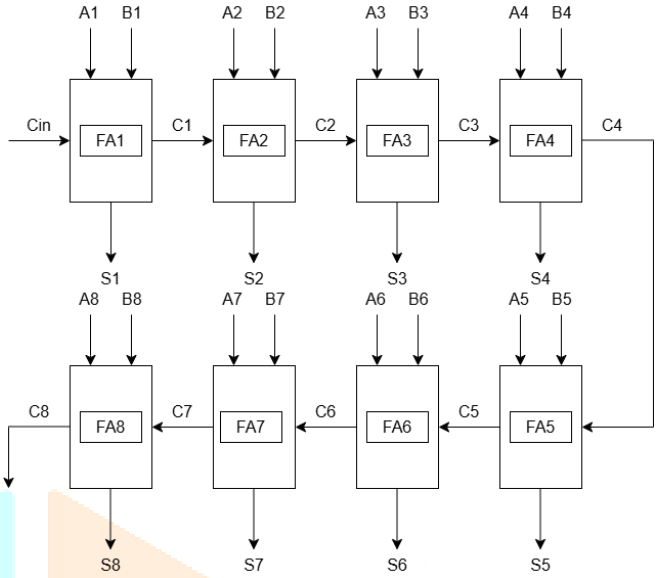

Figure 1 8-bit Ripple Carry Adder (RCA)

IV. METHODOLOGY

This section describes the comprehensive design and evaluation methodology used to compare the 8-bit Ripple Carry Adder (RCA) and Carry Lookahead Adder (CLA) architectures. The process includes design entry using VHDL, functional simulation, synthesis using Cadence Genus, and extraction of key performance metrics such as area, power, and delay using an ASIC design flow.

A. VHDL-Based RTL Design

The Ripple Carry Adder (RCA) architecture was described using VHDL (VHSIC Hardware Description Language). The 8-bit RCA design was implemented by cascading eight full adder modules, where each module computes a sum and passes its carry output to the next stage. The design accepts 8-bit input vectors (A and B), along with a single carry-in (Cin), and produces an 8-bit sum output (S) and a carry-out (Cout). The VHDL code was developed in the Cadence Design Entry Environment, with modular coding practices to ensure clarity, reusability, and ease of verification.

B. Simulation and Verification Setup

Functional verification was performed using Cadence simulation tools to validate the behavioural correctness of the RCA and design before synthesis.

- A structured VHDL testbench was created to apply a range of input patterns, including both exhaustive and randomized cases, ensuring comprehensive coverage.
- Simulations were conducted to cover all possible input transitions, focusing on edge cases and critical corner scenarios to assess design robustness.
- Generated waveforms were analysed to confirm the accurate generation of carry and sum outputs throughout the simulation cycles.

After successful functional verification, Value Change Dump (VCD) files were generated by enabling signal activity recording. These VCD files captured switching activity during simulation and were later utilized for accurate power estimation.

C. Synthesis using Cadence Genus (GUI Mode)

The validated RTL designs were synthesized using the **Cadence Genus Synthesis Solution (GUI mode)** targeting a 45nm standard cell technology library. The synthesis flow included:

- **Design import** of the VHDL files into Genus via GUI interface.
- Setting **synthesis constraints**, such as:
  - Target clock period (e.g., 1 ns),
  - Input/output drive strengths,
- Performing **logic optimization**, technology mapping, and netlist generation.
- Importing **VCD files** into Genus for power analysis based on actual switching activity.

Reports were generated for each architecture, which included metrics on area, power, and delay. These reports served as the foundation for architectural comparison under consistent design and technology parameters.

D. Extracted Performance Metrics

Post-synthesis, the following performance metrics were extracted and documented for both adder designs:

- **Area**: Total cell area consumed in μm^2, obtained from Genus area reports.
- **Power Consumption**: Includes static (leakage) and dynamic power, derived from toggling information in the VCD files.
- **Timing Delay**: Critical path delay (worst-case delay from input to output), derived from Genus timing analysis.

These metrics enabled a structured evaluation of each design's efficiency, independent of any implementation bias.

V. RESULTS

To validate the functional correctness of the 8-bit Ripple Carry Adder (RCA) design, simulations were carried out using Cadence SimVision. The waveform output shown in Figure 2 illustrates the timing behaviour of the RCA for various combinations of input operands (A, B) and carry-in (Cin). It can be observed that the output sum and carry-out (Cout) follow the expected binary addition results. Each bit addition is sequentially dependent on the previous carry, confirming the inherent carry propagation delay in an RCA architecture.

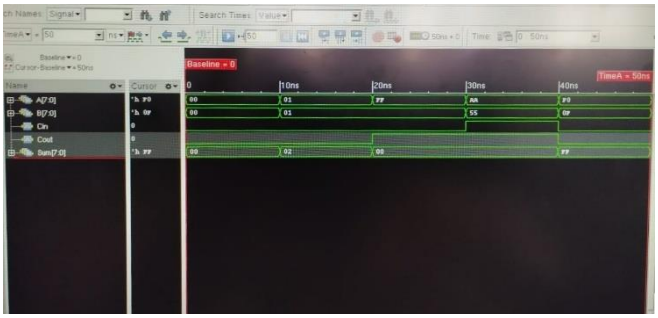

Figure 2 Simulation waveform of 8-bit Ripple Carry Adder showing inputs, sum, and carry outputs in Cadence SimVision.

Furthermore, the schematic diagram in Figure 3 shows the hierarchical design structure consisting of eight full adder modules cascaded together. This confirms the structural implementation of the RCA design, where the carry output from each full adder is passed to the next in the chain.

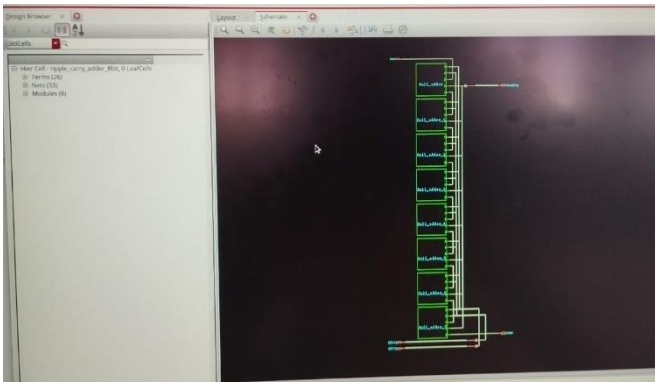

Figure 3 RTL schematic of the 8-bit Ripple Carry Adder showing chained full adder modules.

The 8-bit Ripple Carry Adder (RCA) was implemented in Verilog and synthesized using the Cadence Genus tool targeting a 45nm GPDK technology library. The synthesized design occupies an area of approximately 92.45 µm². Power analysis based on switching activity from simulation indicates a total power consumption of around 6.21 µW, which includes 5.72 µW dynamic power and 0.49 µW leakage power.

Overall, the RCA design demonstrates a good balance between power consumption and silicon area, making it suitable for low-power and area-constrained digital applications where speed requirements are moderate.

Table 1 8-bit RCA results

Parameter	Value
Cell Area	92.45 μm^2
Dynamic Power	5.72 µW
Leakage Power	0.49 µW
Total Power	6.21 µW

VI. CONCLUSION

This study successfully implemented and analysed an 8-bit Ripple Carry Adder (RCA) using Verilog and synthesized it with Cadence Genus targeting a 45nm technology node. The RCA design offers a simple and compact solution with low power consumption and moderate area usage. However, the linear carry propagation leads to increased delay, which limits its speed performance for high-frequency applications. Despite this, the RCA remains an effective choice for applications where low power and minimal area are prioritized over maximum speed. Future work could explore more advanced adder architectures to address the speed limitations while balancing power and area constraints.

References

[1] S. Jalaja, T. R. Dinesh Kumar, S. Sivasaravana Babu, P. Vignesh, R. Vignesh, and P. Prem Kumar, "Ripple Carry Adder Technique for High-Speed and Delay-Line Clocking Based on Low-Power Control," *IEEE Transactions on VLSI Systems*. [Online]. Available: https://ieeexplore.ieee.org/document/10547742.

[2] B. Koyada, N. Meghana, M. O. Jaleel, and P. R. Jeripotula, "A Comparative Study on Adders," *Proc. 2017 Int. Conf. Wireless Commun., Signal Processing and Networking (WiSPNET)*, Chennai, India, Mar. 2017, pp. 2225–2230. [Online]. Available: https://ieeexplore.ieee.org/document/8300155

[3] K. Kadam and S. S. Shetkar, "Design and implementation of power efficient 4-bit ripple carry adder using 14 nm FinFET technology," in *Proceedings of ICCCE 2021*, Springer Nature Singapore, 2021. [Online]. Available: https://www.springerprofessional.de/en/design-and-implementation-of-power-efficient-4-bit-ripple-carry-/20404622

[4] P. Balasubramanian, C. Dang, D. L. Maskell, and K. Prasad, "Approximate ripple carry and carry lookahead adders — A comparative analysis," in *Proceedings of the 2017 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems*

(VLSID), IEEE, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8190125

[5] R. Uma and P. Dhavachelvan, "Performance of Adders with Logical Optimization in FPGA," in *Advances in Computer Science, Engineering & Applications*, D. C. Wyld, J. Zizka, and D. Nagamalai, Eds. Berlin, Heidelberg: Springer, 2012, pp. 245–254. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-30157-5_25

[6] B. Jovanović and M. Jevtić, "Optimization of the Binary Adder Architectures Implemented in ASICs and FPGAs," in *Soft Computing Applications*, V. E. Balas, J. Fodor, A. R. Várkonyi-Kóczy, J. Dombi, and L. C. Jain, Eds. Berlin, Heidelberg: Springer, 2013, pp. 295–308. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-33941-7_27