Performance Analysis Of Power Gating Design In Low Power VLSI Circuits

S Pranesh[1],V Premsai[2],L J Saurav[3],S vasanthiriya[4]
[1, 2, 3, 4] Dept of ECE
[1, 2, 3, 4] Muthayammal Engineering College,Namakkal,Tamilnadu,India.

Abstract- *In modern digital systems, power efficiency is a critical design parameter, especially in applications like mobile devices, IoT, and embedded systems. Clock gating is a widely-used technique for reducing dynamic power consumption by disabling the clock signal in inactive circuit modules. This project explores the design and implementation of a clock-gated 8-bit adder using Verilog. The adder is a fundamental component in arithmetic and logic units (ALUs) and is integral to many digital circuits. By incorporating clock gating into the adder design, significant power savings can be achieved during idle states without compromising functional accuracy. The design process involves integrating clock gating logic into a ripple-carry adder architecture and implementing it using hardware description language (HDL). The Verilog design is simulated and validated using ModelSim, ensuring accurate functionality and verifying the clock gating's effect on power dissipation. Additionally, synthesis is performed using industry-standard tools to assess the area, power, and timing trade-offs. The results demonstrate a measurable reduction in power consumption compared to a conventional adder, making the design suitable for low-power applications. This document provides a comprehensive overview of the design methodology, implementation details, and performance analysis of the clock-gated adder. The findings underline the importance of power optimization techniques in digital design and highlight the potential for further exploration in power-efficient arithmetic units*

Keywords- ALU, HDL, IOT

I. INTRODUCTION

Digital design optimization is crucial for improving performance, power efficiency,area utilization, and reliability in modern electronic systems . As technology scales down and digital circuits become more complex, optimization techniques have become essential to meet stringent design constraints in applications such as mobile devices, embedded systems, and high-performance computing. The primary objectives of digital design optimization include reducing power consumption enhancing performance, minimizing silicon area, and ensuring reliability. Various techniques are employed to achieve these goals, supported by electronic design automation (EDA) tools

This paper explores the significance of digital design optimization with a particular focus on clock gating, a fundamental power optimization technique . By selectively disabling the clock signal in inactive components, clock gating effectively reduces dynamic power consumption, making it a widely used strategy in modern processors, system-on-chips (SoCs), and application-specific integrated circuits (ASICs) [5]. The implementation of a clock-gated 8-bit adder in Verilog serves as a case study to evaluate the impact of clock gating on power efficiency. The study aims to compare the power consumption, timing performance, and area utilization of a conventional adder versus a clock-gated counterpart, highlighting the trade-offs and benefits of this technique .
The findings of this research contribute to the ongoing efforts in low-power digital design by demonstrating the practical advantages of clock gating. The results will help designers make informed decisions in optimizing arithmetic circuits for power-sensitive applications. By integrating power-efficient techniques into digital architectures, this work aligns with the growing demand for energy-conscious computing solutions in modern electronic systems .

II. LITERATURE SURVEY

Rabaey et al. discussed fundamental principles of digital circuit design and emphasized power-efficient design methodologies. They highlighted the importance of reducing switching activity to minimize dynamic power dissipation, a key challenge in modern low-power designs. Weste and Harris further elaborated on CMOS VLSI design strategies, providing insights into optimization techniques at different abstraction levels.

Chandrakasan et al. introduced various low-power design techniques, including supply voltage scaling and energy-efficient computation methods. Their research provided a foundation for later advancements in low-power digital circuits. Pedram and Rabaey examined power-aware design methodologies, with a particular focus on architectural-level power optimization.

Clock gating, as a widely adopted power reduction technique, has been analyzed in several studies. Kang and Leblebici provided an in-depth explanation of clock gating mechanisms and their implementation in CMOS circuits. Yang explored various gating techniques, including AND-based and latch-based clock gating, demonstrating their impact on dynamic power reduction.

Gonzalez and Horowitz investigated energy dissipation in general-purpose processors, highlighting the role of clock gating in reducing unnecessary switching activity. Their study showed that clock gating could reduce dynamic power consumption by 30% to 70%, depending on the circuit activity. Chandrakasan et al. further validated these findings by implementing clock gating in low-power CMOS digital designs, demonstrating significant energy savings.

Recent research has explored the integration of clock gating with other power optimization techniques. Kim et al. [9] proposed adaptive clock gating, which dynamically adjusts gating control based on workload variations, achieving further power efficiency improvements. Similarly, Das et al. introduced machine learning-based clock gating techniques that predict idle states more accurately, optimizing power savings in real-time applications.

The implementation of clock gating in arithmetic circuits has also been investigated. Liu et al. designed a clock-gated adder and compared its power consumption with a conventional adder. Their findings indicated a significant reduction in switching activity without compromising performance. In another study, Zhang et al. analyzed the trade-offs between power, performance, and area (PPA) in clock-gated arithmetic units, emphasizing the importance of careful design considerations.

These studies collectively establish the effectiveness of clock gating as a power-saving technique. The ongoing advancements in clock gating methodologies, including adaptive and machine learning-based approaches, continue to enhance power efficiency in modern digital circuits. This research builds on these existing works by implementing and evaluating a clock-gated 8-bit adder in Verilog, providing insights into its practical benefits and trade-offs.

III. METHODOLOGY

The proposed system is an 8-bit clock-gated adder designed to reduce dynamic power consumption by incorporating clock gating into its operation. The system uses a ripple-carry adder architecture as the arithmetic unit, integrated with a clock gating mechanism to selectively disable the clock signal for idle components. This ensures that power is consumed only when the adder is active, significantly improving energy efficiency.

System Architecture

The proposed system consists of the following key components:

Ripple-Carry Adder

- A simple and robust adder structure is chosen for its ease of implementation.
- Adds two 8-bit binary numbers along with an optional carry input to produce an 8-bit sum and a carry output.
- The operation is sequential, propagating the carry bit through all stages.

Clock Gating Logic

- Implements an AND-based clock gating mechanism:
    - The clock signal is gated (enabled or disabled) using an AND gate.
    - An enable signal determines whether the clock signal is propagated to the adder logic.
- When the enable signal is inactive, the clock signal is blocked, preventing unnecessary toggling of flip-flops

.

Control Unit

- Generates the enable signal based on the system's operational state.
- Monitors external inputs or system requirements to determine whether the adder is active or idle.
- Ensures proper synchronization and prevents glitches during transitions

.

Functional Verification

- The design is modeled using Verilog HDL at the RTL level .
- A comprehensive testbench is developed to verify functionality under various input conditions .
- Functional correctness is ensured before proceeding to synthesis.

Power Analysis

- Power estimation is performed using Synopsys Design Compiler and Xilinx Vivado tools .
- Power savings achieved through clock gating are compared against a conventional non-gated adder .

Timing and Performance Analysis

- Simulation is carried out using ModelSim to analyze waveforms and validate clock gating effectiveness [.
- Performance metrics such as propagation delay, setup time, and hold time are measured [14].

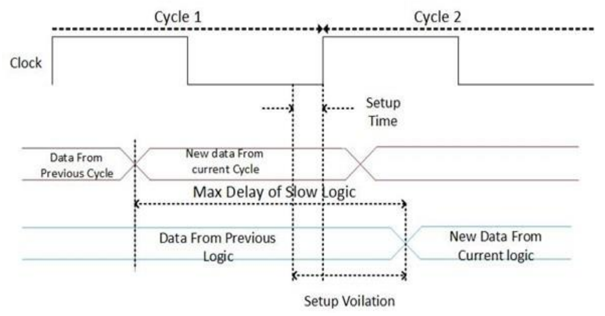

Figure 1:TimingDefinitionofSetupTime

Scalability and Integration

- The architecture is tested for scalability by extending it to higher bit-width adders [15].
- Feasibility for integration into low-power embedded systems is assessed [16].

IV. SOFTWARE DESCRIPTION FOR MODELSIM

ModelSim is a powerful Hardware Description Language (HDL**)** simulation tool used for verifying and debugging digital circuits described in Verilog, VHDL, or SystemVerilog. It is widely employed in the design and development of digital systems to ensure that the logic functions correctly before hardware implementation. Below is a detailed description of ModelSim and its role in this project.

Overview of ModelSim

- Purpose**:** Simulates HDL designs to validate functionality, timing, and logic before synthesis or deployment.
- Supported Languages: Verilog, VHDL, and SystemVerilog

Key Features:

- Integrated development environment (IDE) with waveform viewing and debugging capabilities.
- Supports both behavioral and timing simulation.
- Compatibility with various EDA tools for synthesis and verification.
- Provides tools for debugging with breakpoints, variable tracking, and waveform analysis.
- Relevance to the Project

In this project, ModelSim is used to:

- Simulate the Clock-Gated Adder Design:
  - o Verify that the adder performs correct addition operations.
  - o Validate the functionality of the clock gating mechanism in both active and idle states.
- Generate and Analyze Waveforms:
  - o Observe signal transitions (clock, enable, inputs, and outputs).
  - o Confirm that the clock gating logic effectively disables unnecessary toggling in idle states.
- Debug the HDL Code:
  - o Identify and fix errors in the Verilog implementation.

Debug the clock gating logic, ensuring smooth transitions without glitches or timing violations.

Steps for Using ModelSim

Installing ModelSim:

- Download and install the appropriate version of ModelSim (Intel FPGA Edition, Xilinx Edition, or SE/DE based on your FPGA platform or preferences).
- Configure the tool with the necessary environment variables (e.g., PATH) for easy execution.

Setting Up the Project

1. Create a New Project:
   - o Launch ModelSim and create a new project.
   - o Specify the project directory and HDL language (Verilog in this case).
2. Add Design Files:
   - o Import the Verilog modules (ClockGatedAdder and Testbench) into the project.
   - o Ensure all dependencies and libraries are included.

3. Compile the Design:
    - Use the Compile option to compile the Verilog files.
    - Fix any syntax or semantic errors detected during compilation.

Simulating the Design

1. Set Up the Simulation Environment:
    - Select the Testbench as the top module for simulation.
    - Initialize the simulation using the Simulate menu.
2. Run the Simulation:
    - Run the simulation for a specified duration or until a particular condition is met.
    - Use run commands (run 100ns, run -all, etc.) in the ModelSim console.
3. View Waveforms:
    - Open the Waveform Viewer to observe signals like clock, enable, inputs (a, b, cin), and outputs (sum, cout).
    - Analyze whether the gated clock behaves as expected, blocking unnecessary transitions during idle states.

Debugging

- Use breakpoints, watch windows, and the waveform viewer to debug issues in the design.
- Verify edge cases (e.g., carry-in propagation, clock transitions) using simulation tools.

Features Used in This Project

1. Waveform Analysis:
    - Visualize signal activity and transitions.
    - Confirm the correctness of clock gating and arithmetic operations.
2. Debug Console:
    - Execute commands for step-by-step simulation.
    - Check intermediate values of signals.
3. Simulation Control:
    - Use commands like restart, force, and run to control simulation flow.
4. Log Files:
    - Generate detailed logs of simulation results for documentation and troubleshooting.

Outputs from ModelSim:

- Simulation Waveforms:
    - Illustrates the clock gating effect (clock transitions disabled during idle states).
    - Shows correct addition outputs for various input combinations.
- Debugging Reports:
    - Identifies and resolves issues in the Verilog design.
    - Validates the enable signal logic for smooth transitions between active and idle states.
- Functional Verification:
    - Confirms that the design operates correctly under all test cases defined in the Testbench.

Advantages of Using ModelSim

- Ease of Use: Intuitive interface for waveform viewing and debugging.
- Comprehensive Analysis: Offers detailed insights into signal behavior and logic transitions.
- Integration with Synthesis Tools: Compatible with popular synthesis tools for seamless verification and implementation.

V. CONCLUSION

The implementation of a clock-gated 8-bit adder demonstrates the effectiveness of incorporating low-power design techniques into digital circuits. By selectively disabling the clock signal for idle components, the clock gating mechanism significantly reduces dynamic power consumption without affecting the functional correctness or performance of the adder.

Through simulation and analysis using ModelSim, the functionality of the clock-gated adder was validated under various input scenarios. The results confirmed the expected power savings during idle states, highlighting the practical advantages of clock gating in energy-efficient design. Additionally, synthesis results verified that the design meets timing, area, and power requirements, making it suitable for real-world applications.

REFERENCES

[1] M. Morgenshtein, A. Fish, and I. Wagner, “Gate-Diffusion Input (GDI) – A Power-Efficient Method for Digital Combinatorial Circuits,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, no. 5, pp. 566–581, 2002.

[2] R. Zimmermann and W. Fichtner, “Low-power logic styles: CMOS versus pass-transistor logic,” IEEE Journal

of Solid-State Circuits, vol. 32, no. 7, pp. 1079–1090, 1997.

[3] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits: A Design Perspective, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2003.

[4] S. Borkar, "Design challenges of technology scaling," IEEE Micro, vol. 19, no. 4, pp. 23–29, 1999.

[5] K. Roy and S. C. Prasad, "Low-power CMOS VLSI circuit design," Wiley-Interscience, 2000.

[6] S. Mutoh et al., "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," IEEE Journal of Solid-State Circuits, vol. 30, no. 8, pp. 847–854, 1995.

[7] N. H. E. Weste and D. M. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Boston, MA, USA: Pearson/Addison-Wesley, 2010.

[8] Y. Taur and T. H. Ning, Fundamentals of Modern VLSI Devices, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[9] D. Harris and S. Harris, Digital Design and Computer Architecture, 2nd ed. Burlington, MA, USA: Morgan Kaufmann, 2012.

[10] Synopsys, "Design Compiler User Guide," Synopsys Inc., 2021.

[11] Mentor Graphics, "ModelSim Simulation and Debugging Tool User Manual," Mentor Graphics, 2020.

[12] Xilinx, "Vivado Design Suite User Guide: Synthesis," Xilinx Inc., 2021.

[13] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," IEEE Transactions on Computers, vol. C-35, no. 8, pp. 677–691, 1986.

[14] D. Markovic, C. C. Wang, L. P. Alarcon, T. T. Liu, and J. M. Rabaey, "Ultralow-power design in near-threshold region," Proceedings of the IEEE, vol. 98, no. 2, pp. 237–252, 2010.