STM JOURNALS

Journal of
VLSI Design Tools & Technology

ISSN: 2249-474X (Online)
ISSN: 2321-6492 (Print)
Volume 15, Issue 3, 2025
September–December
DOI (Journal): 10.37591/JoVDTT

https://journals.stmjournals.com/jovdtt

Research JoVDTT

RTL-to-GDSII Flow Optimization for Low-Power 32-Bit RISC-V Processor

Abhishek Patel[1,*], Pankaj Verma[2]

Abstract

This study presents the implementation and optimization of a 32-bit RISC-V processor, transitioning from Register Transfer Level (RTL) design to final GDSII using Synopsys Fusion Compiler over a 32 nm technology node. The processor architecture is based on the RV32I base instruction set and incorporates a 5-stage pipeline to achieve a balanced trade-off between performance and design complexity. The design methodology involved RTL synthesis, gate-level netlist generation, and successive physical design stages including floor planning, placement, clock tree synthesis (CTS), routing, and signoff verification. To enhance the design's power, advanced optimization techniques such as Concurrent Clock and Data Optimization (CCDO), clock gating, and self-gating were employed during the physical design phase. These techniques collectively contributed to a substantial improvement in power efficiency and timing closure. Clock network optimization was carefully conducted to reduce skew, latency, and unnecessary dynamic switching activity. The final physical implementation demonstrated a notable total power reduction of approximately 15.9%, while ensuring performance integrity and meeting stringent timing constraints. The outcomes validate the effectiveness of integrating Synopsys Fusion Compiler in achieving a streamlined RTL-to-GDSII flow and highlight its potential for next-generation processor designs in energy-sensitive applications.

Keywords: RTL-to-GDSII, RISC-V processor, Synopsys fusion compiler, power optimization, clock network design, clock tree synthesis (CTS), physical design flow, low-power design, EDA tools

INTRODUCTION

Modern processor design demands high performance, low power consumption, and rapid development cycles. The RISC-V architecture, known for its open-source nature and modular extensibility, has emerged as a popular choice in both academic and industrial sectors. This study focuses on the RTL-to-GDS implementation of a 32-bit RISC-V processor [1]. The design is based on the RV32I instruction set architecture, a widely adopted 32-bit standard that is celebrated for its simplicity and modularity using industry-grade electronic design automation (EDA) tools. Specifically, Synopsys Fusion Compiler was used to transition the processor from RTL to a manufacturable GDSII layout, while incorporating power-aware optimization techniques.

***Author for Correspondence**
Abhishek Patel
E-mail: abhi.stp011@gmail.com

[1]Student, School of VLSI design and Embedded System, National Intitute of technology, Kurukshetra, Haryana, India
[2]Assistant Professor, Department of Electronics and Communication , National Intitute of technology, Kurukshetra, Haryana, India

Received Date: May 24, 2025
Accepted Date: June 21, 2025
Published Date: December 13,2025

Citation: Abhishek Patel, Pankaj Verma. RTL-to-GDSII Flow Optimization for Low-Power 32-bit RISC-V Processor. Journal of VLSI Design Tools & Technology. 2025; 15(3): 1–10p.

In physical design, the clock network and switching activity are major contributors to dynamic power. Therefore, optimizing both clock paths and data paths is critical. Techniques such as clock gating and concurrent clock and data optimization (CCDO) are used to reduce unnecessary toggling and improve timing closure [2]. This study documents the design journey and quantifies the power savings achieved by these optimizations.

PHYSICAL DESIGN

Physical design represents a critical phase in converting a functional design into a manufacturable

integrated circuit (IC) [3]. This intricate process begins with transforming a high-level hardware description, such as Register Transfer Level (RTL) code, into a gate-level netlist. The subsequent steps include floorplanning, placement, routing, clock tree synthesis (CTS), and signoff verification. Each stage plays a pivotal role in optimizing the chip's performance, power efficiency, and area utilization, collectively referred to as PPA metrics (Figure 1).

Physical Design Flow

- *Partitioning:* Partitioning is the process of dividing a large RTL circuit (or netlist) into smaller, manageable sub-blocks or regions before floorplanning and placement [3]. It is especially useful in complex chips like processors or SoCs that contain multiple functional modules**.**
- *Floorplanning and Placement:* Floorplanning is the initial step where major functional blocks are strategically arranged on the chip to define their layout [3]. Effective floor planning minimizes congestion and ensures better timing performance. Following this, placement algorithms arrange standard cells in a manner that reduces wirelength and avoids excessive signal delays.
- *Clock Tree Synthesis (CTS):* CTS is a critical stage that establishes a reliable clock distribution network across the chip [3]. The goal is to deliver the clock signal to all parts of the IC with minimal skew and latency, enabling synchronized operations. Proper CTS ensures the chip operates reliably at its intended frequency.
- *Routing:* During routing, interconnections between the logic elements are created, considering factors such as resistance, capacitance, and signal delay [3]. Modern ICs rely on multiple layers of metal interconnects to accommodate the high density of connections, ensuring optimal signal integrity and performance.

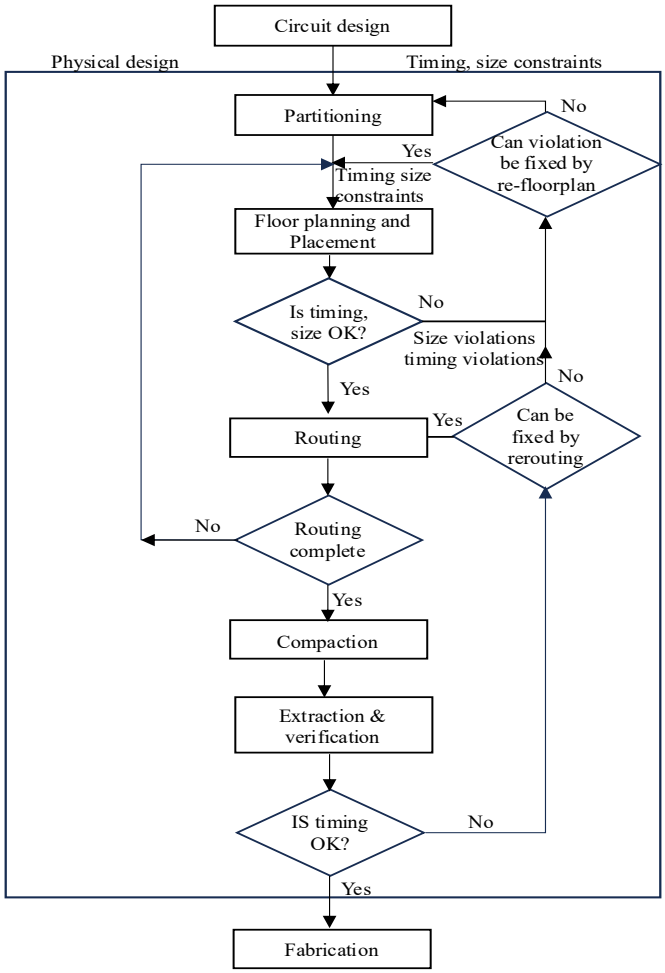


Figure 1. Physical design cycle.

- *Signoff and Verification:* Before the design is finalized for manufacturing, the chip undergoes rigorous verification processes to confirm it adheres to performance, power, and manufacturability requirements. Key checks include timing closure, design rule compliance, and electromigration analysis. This ensures that the design is robust and ready for fabrication without unexpected issues.

POWER CONSIDERATION IN PHYSICAL DESIGN

During the physical design flow of a chip, designers must address various forms of power consumption that arise due to the nature of digital switching and the characteristics of modern CMOS technologies [4]. These power losses are typically categorized as dynamic and static power losses (Figure 2).

Dynamic Power

Dynamic power primarily originates from the charging and discharging of internal capacitances and short-circuit current flow during state transitions [4].

- *Switching Power:* It is dissipated every time a logic gate changes its output. The primary reason behind this is the charging and discharging of capacitive loads associated with interconnects and gate inputs.
- *Short-Circuit Power:* During the switching transition, there is a small-time window when both the pull-up and pull-down transistors in a CMOS gate are conducting. This creates a direct path from the power supply to ground, causing a brief but non-negligible power loss.

Static Power

Static power, also known as leakage power, is the power consumed by the chip even when no active switching is occurring due to aggressive device scaling, which results in thinner gate oxides and lower threshold voltages [4].

- *Subthreshold Leakage:* Subthreshold leakage occurs when the transistor is off (i.e., $V_{GS}<V_{TH}$) but a small amount of current still flows between the source and drain. This happens due to carrier diffusion and is more prominent in transistors with low threshold voltage.
- *Gate Leakage:* As device geometries shrink, gate oxide layers become ultra-thin, leading to tunnelling currents through the oxide, commonly referred to as gate leakage.

POWER OPTIMIZATION TECHNIQUES USED IN DESIGN FLOW

In the physical design flow, various strategies are integrated to minimize dynamic, static, and short-circuit power components. These techniques are applied across different stages of the flow, from RTL synthesis to signoff. Below is an overview of key power optimization techniques:

Clock Gating

Clock gating is one of the most commonly used dynamic power-saving techniques [5]. The idea is to disable the clock signal to modules that are not currently active, preventing unnecessary switching activity. Since the clock network contributes significantly to overall power due to its high toggle rate, gating idle portions reduces power significantly.

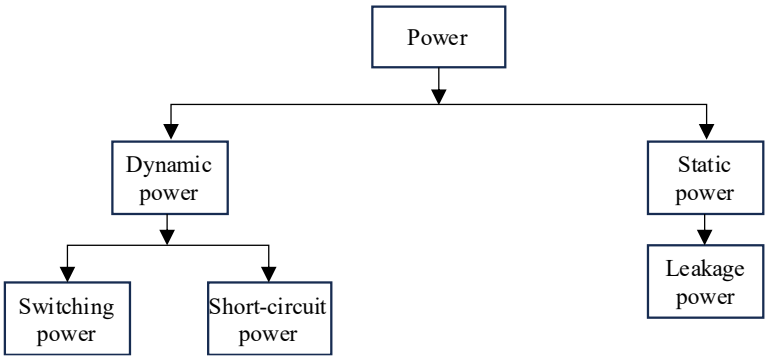


Figure 2. Power considerations in VLSI physical design

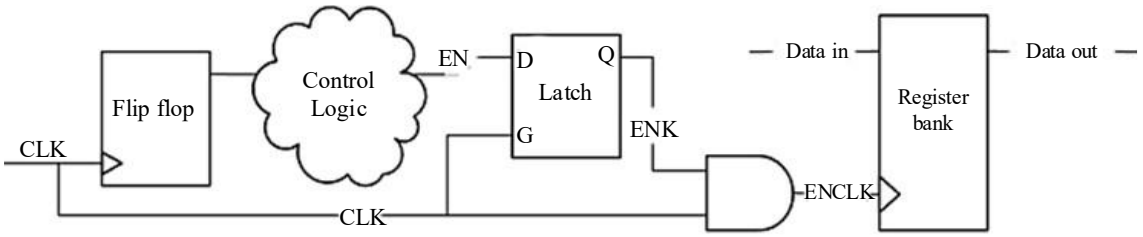


Figure 3. Latch-based clock gating.

In latch-based clock gating, the AND gate blocks unnecessary clock pulses by maintaining the clock signal's value after the trailing edge (Figure 3).

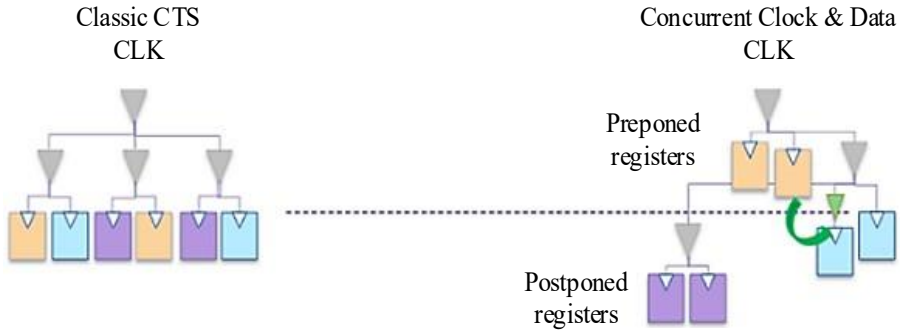
Self Gating

Self-gating is a low-power design technique where logic within a module itself determines whether it needs to disable its clock signal based on its own internal conditions or data activity. Rather than relying on external control signals (like in regular clock gating), self-gating logic is embedded inside the module [6]. The module "self-analyses" its need to operate, and if it does not need to update, it disables its clock locally. Registers with an enable condition that cannot be inferred from the existing logic can only be gated using the self-gating technique.

Concurrent Clock and Data Optimization

One of the most significant contributors to dynamic power is the switching activity of the clock network. The clock tree drives a large number of sequential elements and toggles constantly, making it a major consumer of dynamic power. Therefore, optimizing the clock network is critical for reducing power consumption and ensuring timing closure. One of the modern techniques used for this purpose is Concurrent Clock and Data Optimization (CCDO) [5].

Concurrent Clock and Data Optimization is typically implemented during or just before the CTS phase in modern EDA tools like Synopsys Fusion Compiler. In essence, while classic CTS focuses solely on distributing the clock signal, CCDO aims to align clock distribution with data path characteristics, offering better power efficiency, enhanced performance, and improved timing robustness. For designs where power and timing margins are tight, especially in deep sub-micron technologies, CCDO presents a compelling improvement over the traditional method (Figure 4) [7].



- Clock tree is built while ignoring the data paths
- Goal: Minimize skew
  - Allows full clock-cycle reg-to-reg timing
- Data path is optimized post-CTS -Clock tree remains untouched

- Clock tree is built with full knowledge of data path timing
- Goal: Meet setup/hold timing
  - Reg-to-reg timing may be larger or smaller than a clock-cycle
- Data path is optimized post-CTS - Clock tree remains untouched

Figure 4. Classic CTS vs CCD optimization [8].

METHODOLOGY

The RTL-to-GDSII flow in Fusion Compiler is a structured process that transforms a high-level design description into a physical layout ready for manufacturing [8]. This comprehensive flow integrates logical synthesis, physical design, and optimization into a single, unified environment. Below is a step-by-step explanation of the Fusion Compiler's compile flow:

RTL Description and Synthesis

The RISC-V processor, described in Verilog and based on the RV32I instruction set, was synthesized using Fusion Compiler targeting a 32 nm technology node. The design flow began with importing RTL files, SDC constraints, and standard cell libraries [9].

During synthesis, the RTL was translated into a gate-level netlist, with optimizations applied for timing, power, and area using techniques such as gate resizing, retiming, and logic restructuring [8]. Design constraints, including clock frequency, area limits, and transition thresholds, were specified to guide the synthesis process and ensure that the resulting netlist met both functional and performance requirements ahead of physical implementation (Figure 5).

Floorplanning and Placement

After synthesis is floorplanning, which lays out the initial placement of macros, standard cells, and I/O pins within the chip area [8]. In this phase, the power grid is created, and regions for standard cells and macros are defined. The placement stage involves placing the standard cells in the predefined floorplan regions. The tool performs global placement first, where cells are positioned approximately to optimize timing and connectivity [10]. This is followed by detailed placement, where the tool refines cell positions to adhere to design rules and improve timing closure (Figure 6).

Clock Tree Synthesis (CTS)

After placement, the tool moves to clock tree synthesis (CTS) [8]. In this step, the tool builds a clock distribution network to deliver the clock signal uniformly across the design. The clock tree is optimized for minimal skew and low latency while meeting timing and power constraints [11].

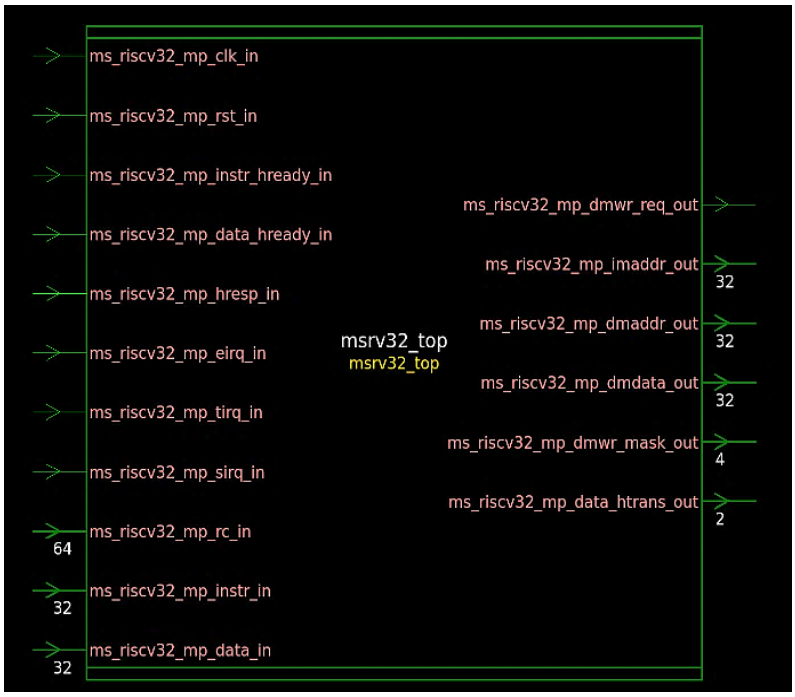


Figure 5. Schematic of the synthesized netlist.

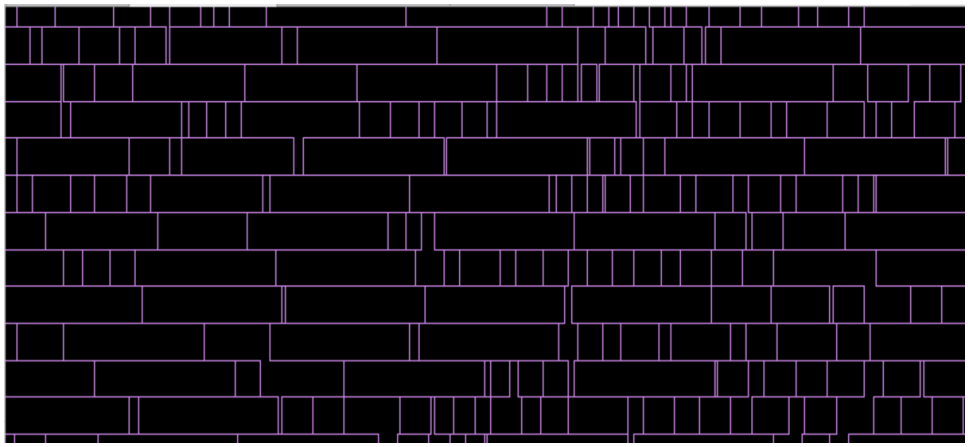


Figure 6. Legalized placed cells.

Routing and Signoff

The routing phase connects the placed cells and macros according to the synthesized netlist [8]. During this stage, the tool performs global routing to determine approximate routes for signals, followed by detailed routing to implement the connections precisely.

With routing complete, the design undergoes signoff checks, which involve design Rule Checking (DRC) and Layout vs Schematic (LVS) verification. After that, a clean final layout is produced, ready for fabrication.

RESULTS

Figure 7 shows the complete final layout of the RISC-V processor with completed clock tree synthesis, and all the routing of the signal nets is done at this stage.

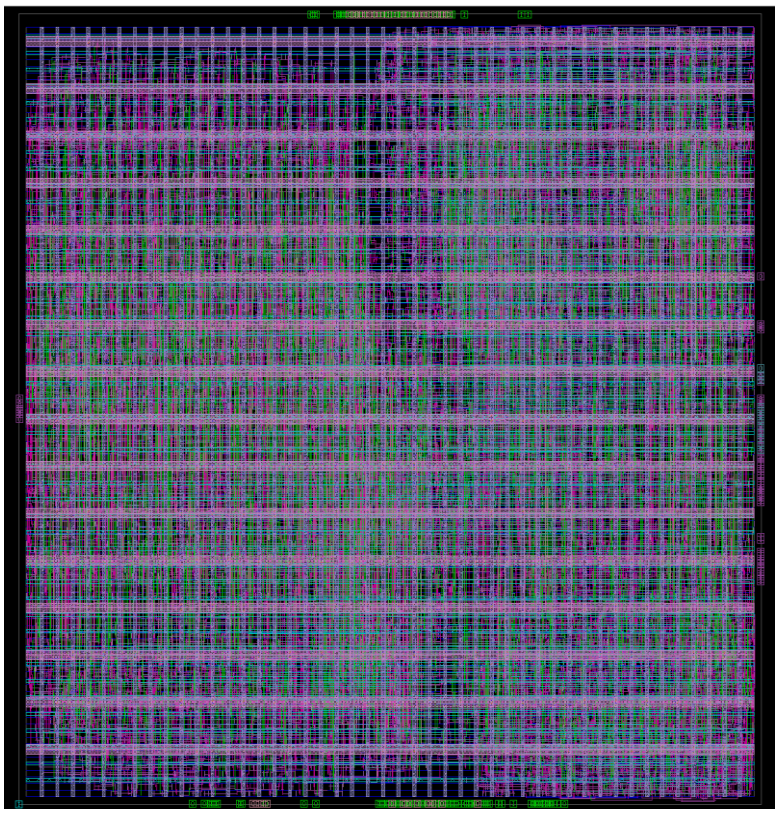


Figure 7. Final layout of design (32-bit RISC-V processor).

Power Report With and Without Optimization

The power consumption analysis of the 32-bit RISC-V processor provides valuable insights into its energy efficiency and design optimization. The report categorizes power consumption into four primary components: internal power, switching power, leakage power, and total power (Tables 1 and 2).

Table 1. With optimization.

Power Group	Internal power	Switching power	Leakage power	Total power
io_pad	0.00e+00	0.00e+00	0.00e+00	0.00e+00
memory	0.00e+00	0.00e+00	0.00e+00	0.00e+00
black_box	0.00e+00	0.00e+00	0.00e+00	0.00e+00
clock_network	3.27e+08	6.78e+07	1.23e+06	3.96e+08
register	1.31e+08	7.55e+06	1.80e+07	1.57e+08
sequential	1.96e+06	1.61e+06	3.17e+04	3.60e+06
combinational	3.12e+07	5.95e+07	5.52e+05	9.12e+07
Total	4.91e+08	1.36e+08	1.98e+07	6.48e+08

Table 2. Power comparison.

Power group	Internal power	Switching power	Leakage power	Total power
io_pad	0.00e+00	0.00e+00	0.00e+00	0.00e+00
memory	0.00e+00	0.00e+00	0.00e+00	0.00e+00
black_box	0.00e+00	0.00e+00	0.00e+00	0.00e+00
clock_network	3.14e+08	2.69e+07	1.35e+06	3.43e+08
register	1.31e+08	7.11e+06	9.45e+06	1.47e+08
sequential	1.42e+06	5.01e+05	3.80e+04	1.96e+06
combinational	3.39e+07	5.66e+07	5.53e+05	9.10e+07
Total	4.80e+08	9.11e+07	1.80e+07	5.45e+08

Table 3. Timing summary.

Power type	Without optimization	With optimization	Improvement
Internal Power	4.91×10^{8} pW	4.80×10^{8} pW	Decreased slightly
Switching Power	1.36×10^{8} pW	9.11×10^{7} pW	Reduced significantly
Leakage Power	1.98×10^{7} pW	1.80×10^{7} pW	Reduced slightly
Total Power	6.48×10^{8} pW	5.45×10^{8} pW	~15.9% decrease

Power Analysis and Comparison: Pre- and Post-Optimization

The key observation is accumulated in Table 3.

The comparison between the two power reports, before and after applying power optimization techniques, reveals notable reductions across all major components of dynamic and static power consumption (Table 3) [12].

The histogram in Figure 8 shows the power comparison of the Pre- and Post-Optimization flow.

Internal Power: There is a minor reduction in internal power, indicating efficient transistor switching and reduced short-circuit power due to better logic structuring and transistor sizing.

Switching Power: A notable reduction *(~33%)* was observed after optimization. This directly correlates to clock gating and self-gating techniques, minimizing unnecessary signal transitions and toggling activity [13].

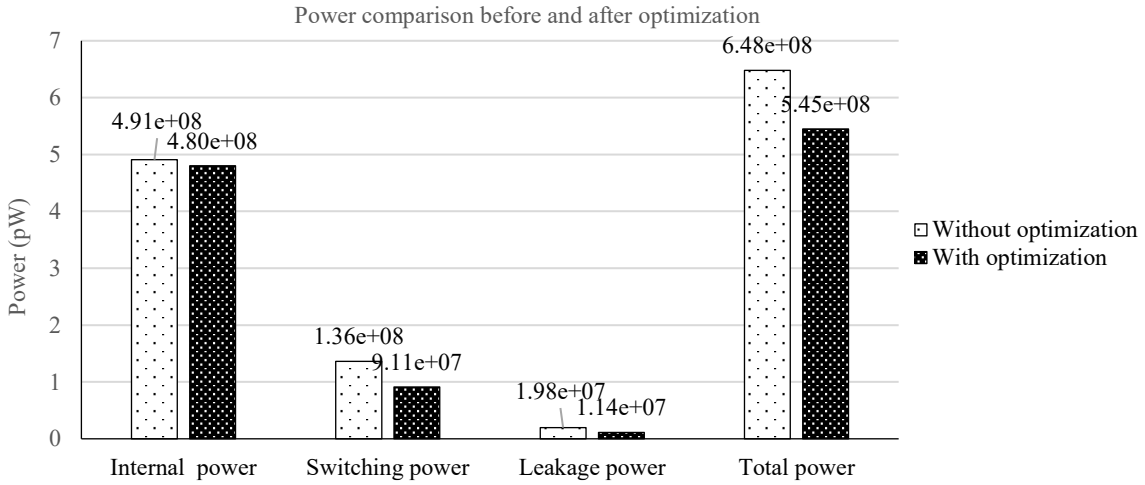


Figure 8. Power comparison histogram.

Table 4. Area summaries.

Context	WNS (before) [ns]	WNS (after) [ns]	TNS [ns]	NVE
func.ff_m40c (Setup)	7.22	7.26	0.00	0
func.ss_125c (Setup)	1.77	1.68	0.00	0
func.ss_m40c (Setup)	0.02	0.01	0.00	0
Design (Setup)	0.02	0.01	0.00	0
func.ff_m40c (Hold)	0.04	0.03	0.00	0
func.ss_m40c (Hold)	0.11	0.03	0.00	0
Design (Hold)	0.04	0.03	0.00	0

Table 5. Area Summaries.

Metric	Before optimization	After optimization
Cell Area (netlist)	28705.82 µm²	28972.16 µm²
Cell Area (netlist and physical only)	28705.82 µm²	28972.16 µm²

Leakage Power: Nearly 9% decrease in leakage power is observed, which improves the energy efficiency and thermal performance of the system, leading to lower standby power consumption and enhanced reliability.

Total Power: Overall, the optimizations brought down the total power consumption by over *15.9%* which is significant and demonstrates the efficiency of the applied techniques.

Timing and Area Comparison

The Quality of Results comparison reveals that the design remains well-optimized and functionally correct in both pre- and post-optimization stages. After applying power optimization techniques such as clock gating, self-gating, and CCDO, minor adjustments are observed in timing and area, with no violations (NVE=0) in either case (Table 4) [14, 15].

- WNS (Worst Negative Slack) across all corners remains positive in both versions, confirming that the design meets timing.
- There is a slight decrease in hold slack, particularly in func.ss_m40c, from 0.11 to 0.03, indicating tighter hold margins post-optimization, but still well within acceptable limits.
- Total cell area increased slightly (~0.9%) from 28705.82 to 28972.16 µm², which is expected due to the insertion of additional clock-gating logic or buffer cells to aid power reduction (Table 5).

Post-optimization, the design maintains full timing integrity and exhibits only a marginal area increase while achieving significant power savings (as seen in earlier power reports). This confirms that the applied power optimization strategies did not compromise the timing or structural quality of the design.

CONCLUSION

The successful RTL-to-GDSII implementation of the 32-bit RISC-V processor using Synopsys Fusion Compiler was achieved with both functional correctness and design closure. Initially, the design process emphasized completing the flow from synthesis to final signoff without explicit optimizations. However, further refinement through dynamic power optimization techniques, including Clock Gating, Self-Gating, and Concurrent Clock and Data Optimization (CCDO), significantly improved the processor's power efficiency. After implementing these techniques, a notable reduction in internal, switching, leakage, and overall power consumption was observed, validating the effectiveness of power-conscious design practices. This enhanced outcome highlights the critical importance of targeted optimizations in achieving energy-efficient, high-performance VLSI designs. The study demonstrates a comprehensive understanding of both the physical design process and advanced power-saving methodologies, aligning well with the modern semiconductor industry's goals.

REFERENCES

1. Aradhya HVR, Kanase G, Vinayakgouda Y. RTL to GDSII of Harvard structure RISC processor. In: 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT); 2021. p. 1–4.
2. Chong AB, Tan SH, Koh JI, Mohamad HI, Loo GH, Chin KCC. Clock optimization techniques. In: 2022 IEEE 5th International Conference on Electronics Technology (ICET); 2022. p. 321–326.
3. Sherwani NA. Algorithms for VLSI Physical Design Automation. 3rd ed. Boston (MA): Springer; 1999.
4. Rabaey JM. Digital Integrated Circuits. 2nd ed. Upper Saddle River (NJ): Prentice Hall; 2002.
5. Jayasurya K, Ajith R, Premananda BS. Area and power optimized RTL to GDSII flow of a telecommunication receiver core. In: 2024 5th International Conference on Circuits, Control, Communication and Computing (I4C); 2024. p. 207–212.
6. Shohal A, Kaur J. Efficient RTL to GDSII flow for finite state machine integration: A physical design approach. In: 2024 IEEE 5th India Council International Subsections Conference (INDISCON); 2024. p. 1–5.
7. Maestre S, Gumera A, Hora J, de Guzman MJ. Improving digital design PPA (performance, power, area) using iSpatial physical restructuring. In: 2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC); 2022. p. 647–651.
8. Synopsys Inc. Fusion Compiler User Guide. Mountain View (CA): Synopsys; 2023.
9. Weste NHE, Harris D. CMOS VLSI Design: A Circuits and Systems Perspective. 4th ed. Boston (MA): Pearson; 2010.
10. Hiremath S, Vinayakgouda Y. RTL to GDSII implementation of advanced high-performance bus lite. In: 2021 6th International Conference on Communication and Electronics Systems (ICCES); 2021. p. 303–306.
11. Jain R, De S, Shreeharsha KG. RTL to GDSII implementation of RADIX-4 Booth multiplier using opensource EDA tools. In: 2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS); 2023. p. 1–4.
12. Acharya D, Mehta US. Performance analysis of RTL to GDS-II flow in opensource tool Qflow and commercial tool Cadence Encounter for synchronous FIFO. In: 2022 IEEE International Conference of Electron Devices Society Kolkata Chapter (EDKCON); 2022. p. 199–204.
13. Pala V, Makhe V, Bhuva K, Parekh R. RTL to GDSII flow implementation of 8-bit Baugh-Wooley multiplier. In: 2022 IEEE International Conference on Nanoelectronics, Nanophotonics, Nanomaterials, Nanobioscience and Nanotechnology (5NANO); 2022. p. 1–6.

14. Kachhadiya RJ, Agrawal Y, Parekh R. Implementation of ALU using RTL to GDSII flow and on NEXYS 4 DDR FPGA board. In: 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC); 2021. p. 481–488.
15. Fauzan M, Yanuarsyah RF, Hibatullah MH, Arisaputra R, Syafalni I, Sutisna N, **et al.** Physical design of RISC-V based system-on-chip using OpenLane. In: 2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS); 2024. p. 529–533.

