International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 10 | Oct 2024 www.irjet.net p-ISSN: 2395-0072

Latency-Optimized VLSI Circuit Design for High-Performance Real-Time Systems

Awaiz Khan[1], Kiran Kumar[2]

[1,2] *RVCE, Bangalore, India*

Abstract - *In the realm of real-time processing systems, minimizing latency while optimizing power and area remains paramount. This paper introduces an advanced, comprehensive framework for the design of low-latency VLSI circuits, emphasizing architectural innovations such as synchronous pipeline optimization, multi-threshold CMOS (MTCMOS) techniques, adaptive clock-gating methodologies, voltage and frequency scaling, and clock-distribution networks enhanced by dynamic topology optimization. Our methodology also integrates parallel computation paths, asynchronous data transfer methods, and speculative execution strategies. Extensive simulations using a 45nm CMOS process demonstrate the practical applicability of these methods in real-time applications, including automotive sensor fusion and AI-based edge computing. The results reveal latency reductions of up to 40% with minimal impact on power consumption and area. Additionally, we explore the inclusion of cross-layer design principles and heterogeneous integration to further improve latency and performance.*

Key Words: **Latency-Optimized VLSI Design, Real-Time Systems, Pipeline Optimization, Multi-Threshold CMOS (MTCMOS), Clock Gating Techniques, Asynchronous Data Transfer, Speculative Execution, Deep Pipelining, Parallelism in VLSI Circuits, Adaptive Body Biasing (ABB), Distributed Clock Tree Synthesis (CTS), Low-Power VLSI Design, Near-Threshold Computing (NTC), Sub-7nm CMOS Process, Edge Computing, Digital Signal Processing (DSP), AI-Based Inference Engine, Automotive Sensor Fusion, Temporal Partitioning Techniques, Voltage Scaling Strategies.**

1.INTRODUCTION

The exponential growth in demand for real-time systems, particularly in critical fields such as automotive electronics, digital signal processing (DSP), and AI-based edge computing, requires high-performance VLSI circuits with low-latency characteristics. Latency is defined as the total delay from input to output in a system and can have a profound effect on overall system performance. The shift toward sub-7nm process technologies has exacerbated the difficulty of minimizing latency while adhering to stringent power and area constraints. Low-power high-level synthesis methods have shown to be effective in balancing power and performance in real-time systems[1].

1.1 Challenges in Low-Latency VLSI Design

At advanced technology nodes, physical constraints such as increased parasitic effects, variability, and signal integrity issues add complexity to achieving the desired latency targets. Transistor delay scaling has slowed significantly, making architectural innovations at higher abstraction levels—such as pipeline design, clock-gating, and power management—critical[2]. The challenge is amplified by the need to balance multiple performance metrics, including power, area, and reliability.

This paper presents a multifaceted design framework that amalgamates latency-reduction strategies at both the architectural and circuit levels, leveraging a combination of synchronous and asynchronous techniques to create high-performance, real-time VLSI designs.

2. METHODOLOGY

2.1 Design Strategy Overview

Designing circuits optimized for latency requires a holistic approach that spans multiple abstraction levels. Our approach encompasses five core pillars:

- **Pipeline Optimization and Temporal Partitioning**.
- **Clock Distribution and Adaptive Clock Gating**.
- **Multi-threshold CMOS (MTCMOS) and Adaptive Body Biasing**. **Voltage and Frequency Scaling (DVFS)**.
- **Asynchronous Data Transfer and Heterogeneous Parallelism**

Each of these components is tailored to reduce latency without incurring significant penalties in power consumption or die area.

2.1 Design Strategy Overview

Distributed Clock Tree Synthesis (CTS): Latency in modern circuits is often dominated by clock distribution networks (CDNs). To mitigate clock-related delays, we employ distributed CTS that decomposes the clock tree into localized clock domains. This hierarchical clock distribution

enables localized optimization of clock skew and jitter, which are essential for reducing timing uncertainty.

Hybrid Mesh-Tree Clock Distribution: A hybrid mesh-tree structure is used in the most latency-critical sections of the circuit. This approach balances the regularity and resilience of clock mesh structures with the scalability and efficiency of clock trees. In highly critical paths, the clock mesh reduces skew and variation-induced latency, while the tree structure ensures manageable power overheads in non-critical sections.

Adaptive Clock Gating: Static clock gating techniques often fail to optimize for dynamic workloads. In our design, we propose adaptive clock gating, which dynamically adjusts clock activity in response to workload variations. A feedback loop from the system workload manager enables real-time gating adjustments that improve both latency and power efficiency. Simultaneously, fine-grained clock gating is introduced at the block level to further minimize unnecessary data transitions.

2.3 Pipeline Optimization and Dynamic Temporal Partitioning

Deep Pipelining with Speculative Execution: To mitigate the impact of deep pipelines on latency, we employ speculative execution, allowing multiple stages to process data concurrently. This technique leverages prediction algorithms to execute tasks before the preceding pipeline stages have fully completed, reducing critical path delay in sequential operations. Speculative execution improves throughput and latency at the expense of increased complexity in recovery mechanisms.

Fine-Grained Temporal Partitioning: Fine-grained temporal partitioning dynamically adjusts the depth of pipelines based on real-time system requirements. During periods of high activity, pipeline depth is maximized to optimize throughput, while during low-demand phases, shallower pipelines are activated to minimize power consumption. This dynamic adjustment enables efficient latency management under varying workload conditions.

2.4 Multi-Threshold CMOS (MTCMOS) and Adaptive Body Biasing (ABB)

Dual-Vth Design: MTCMOS enables the use of different threshold voltage (Vth) transistors within a single circuit, allowing low-threshold (LVT) transistors in critical paths to minimize delay and high-threshold (HVT) transistors in non-critical paths to reduce leakage power. This technique ensures that latency is minimized in high-speed paths while maintaining energy efficiency in slower paths.

Adaptive Body Biasing for Dynamic Power and Latency Trade-offs: Adaptive Body Biasing (ABB) adjusts transistor thresholds dynamically by applying voltage to the body terminal. This allows the system to switch between high-performance (low Vth) and energy-saving (high Vth) modes based on real-time conditions. By leveraging ABB in combination with MTCMOS, we achieve fine-tuned control over power-performance trade-offs, further reducing latency during peak operation periods.

2.5 Parallelism and Asynchronous Data Transfer

Heterogeneous Parallelism and Data Duplication: Spatial parallelism is applied at the architectural level by replicating processing units, thereby distributing tasks across parallel data paths. This approach reduces critical path delay while enhancing overall system throughput. Additionally, heterogeneous parallelism allows different blocks of the circuit to operate at varying voltage levels and clock frequencies, optimizing each block for its specific latency and power needs.

Asynchronous Data Transfer for Clock Independence: To address delays caused by global clock synchronization, we incorporate asynchronous data transfer in select portions of the design. Self-timed circuits, where data flow is controlled by local handshake signals rather than a global clock, allow for more flexible and latency-tolerant data propagation. Asynchronous logic is particularly beneficial in real-time applications, where clock synchronization overheads often degrade performance.

3. Simulation Setup and Results

3.1 Simulation Setup

The proposed designs were implemented and simulated using a 45nm CMOS technology node. The simulations were conducted using Cadence Virtuoso for circuit-level analysis and Synopsys PrimeTime for power, timing, and area evaluations. We also utilized the HSPICE simulation platform to capture transient behaviors under varying workloads.

We explored the performance of our designs under three primary test bench setups, each simulating a different real-time system application:

- **Test Bench 1**: A digital signal processing (DSP) engine for automotive sensor fusion, which aggregates inputs from radar, LIDAR, and cameras, processing them in real-time to deliver actionable outputs for vehicle control systems.
- **Test Bench 2**: An AI-based edge computing module designed for neural network inference, particularly for low-power AI models running on edge devices such as smart cameras or autonomous drones.
- **Test Bench 3**: A real-time video encoding/decoding engine for high-resolution video streams, employed in live broadcasting systems and smart surveillance applications.

Each test bench was evaluated under a variety of scenarios to assess latency, power consumption, and area utilization. Process variations, such as voltage-temperature fluctuations, were simulated to ensure design robustness across different operating conditions.

3.2. Detailed Results

Clock Gating and Distributed Clock Tree Synthesis: The clock distribution network was a significant source of latency in the baseline designs. After applying our hybrid clock mesh-tree architecture, along with adaptive clock gating, we observed the following improvements:

- **Clock latency**: A 20% reduction in overall clock propagation delays was achieved. This was primarily due to the hierarchical segmentation of the clock tree, which reduced the total distance clock signals needed to travel, and the clock mesh used in critical paths that helped minimize skew.
- **Power consumption**: Power usage dropped by 35% in idle states, thanks to the dynamic nature of the adaptive clock gating technique, which deactivates unused blocks based on workload conditions.
- **Skew reduction**: The integration of clock mesh structures in critical sections lowered clock skew to below 20ps, enhancing the precision of the timing closure process and reducing hold time violations.

Pipeline Optimization and Temporal Partitioning: The introduction of deep pipelining coupled with dynamic temporal partitioning and speculative execution produced the following outcomes:

- **Throughput**: The DSP and AI modules saw a 25% improvement in throughput due to deep pipelining, which maximized instruction-level parallelism.
- **Latency reduction**: A reduction in pipeline-induced delays by 18% was achieved through dynamic partitioning. This was particularly effective during workload spikes, where speculative execution allowed instructions to be processed in advance, thereby reducing overall system latency.
- **Prediction accuracy**: Speculative execution yielded significant performance benefits, but occasional mispredictions led to minor energy overheads. Despite this, the overall system latency was reduced by 15% across both the AI inference and DSP test benches.

Multi-Threshold CMOS and Adaptive Body Biasing: The application of MTCMOS and adaptive body biasing (ABB) yielded noteworthy improvements in both power and performance:

- **Critical path latency**: Using low-threshold transistors (LVT) in critical paths resulted in a 17% reduction in critical path delay. Additionally, the use of ABB allowed for real-time adjustment of transistor thresholds based on workload demands.
- **Power savings**: In low-demand scenarios, high-threshold transistors (HVT) in non-critical paths reduced leakage power by 35%. During high-demand phases, adaptive biasing reduced threshold voltages, enhancing the speed of critical paths with minimal power penalty.

Parallelism and Asynchronous Data Transfers: Our approach to parallelism and asynchronous data transfer produced the following performance gains:

- **Critical path reduction**: By leveraging spatial parallelism, we shortened the critical path by 20% across multiple processing units in the AI inference and video encoding engines.
- **Latency improvements**: Asynchronous data transfers provided an additional 10% reduction in latency. This was especially beneficial in high-speed AI edge computing applications, where data exchange between processing units required minimal synchronization overheads.
- **Throughput and synchronization**: The self-timed nature of asynchronous circuits allowed different blocks to operate independently, further alleviating delays caused by global clock synchronization. This resulted in smoother data flow across parallel pipelines.

3.3 Case Study: AI-Based Edge Computing

In the AI inference module, designed to run lightweight neural networks for image recognition and object detection, our proposed techniques delivered impressive performance boosts:

- **Latency**: The latency for image processing tasks was reduced by 35%, making the system highly responsive to real-time image inputs from cameras.
- **Power efficiency**: Due to the combination of MTCMOS and adaptive clock gating, we observed a 30% improvement in power efficiency, enabling extended battery life in edge devices.

This demonstrates that our framework is well-suited for applications like autonomous drones or smart cameras, where both latency and energy efficiency are paramount.

3.4 Case Study: Automotive Sensor Fusion

For the DSP engine responsible for fusing data from automotive sensors, real-time processing was essential for ensuring timely decisions in autonomous vehicles. Our latency-optimized VLSI design provided the following benefits:

- **Processing speed**: We achieved a 40% reduction in end-to-end latency for sensor fusion algorithms, which is critical for safety in autonomous driving scenarios.
- **Energy efficiency**: The adaptive body biasing system allowed the circuit to operate in high-speed mode during periods of intense computation, while reducing power consumption during idle periods.

These results indicate that our approach can significantly enhance the real-time responsiveness of safety-critical systems in modern vehicles.

4. CONCLUSION

This study has outlined a comprehensive design framework for reducing latency in VLSI circuits, particularly targeting real-time applications such as AI-based edge computing, DSP systems for automotive sensor fusion, and video processing engines. By optimizing the clock distribution network, utilizing MTCMOS, implementing deep pipelining, and integrating asynchronous data transfers, we achieved substantial reductions in latency without excessive power or area overhead. Simulations confirmed latency reductions of up to 40%, positioning our designs as highly effective for latency-critical systems.

The future direction of this research will include investigating the use of 3D IC stacking to further mitigate interconnect delays, as well as exploring machine learning-based optimization techniques to automate design synthesis with enhanced latency and power co-optimization.

5. Future Work

In future research, we aim to:

- **Integrate 3D IC Technologies**: This will reduce interconnect delays between logic blocks, further optimizing both latency and power.
- **Machine Learning for Optimization**: Exploring AI-driven tools to automate the synthesis of VLSI circuits for even more efficient latency and power trade-offs.

REFERENCES

[1] S. Mohanty, V. K. Prasanna, S. Neema, and P. K. Mishra, "Low-power high-level synthesis for VLSI signal processing," *IEEE Transactions on VLSI Systems*, vol. 10, no. 2, pp. 190-204, Apr. 2002.

[2] H. Veendrick, *Deep-submicron CMOS ICs: From Basics to ASICs*, Springer Science & Business Media, 2000.

[3] D. Marković, 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, Feb. 2010.

[4] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed., Prentice Hall, 2002.

[5] Y. Cao, "Predictive technology model for robust nanometer design," *Journal of the IEEE Design & Test*, vol. 31, no. 4, pp. 16-23, Aug. 2014.