Sarcouncil Journal of Multidisciplinary

ISSN(Online): 2945-3445

Volume- 05| Issue- 08| 2025

Research Article **Received:** 25-06-2025 | **Accepted:** 16-07-2025 | **Published:** 04-08-2025

Distributed Flat So C Timing Analysis Using Hyper Grid: A Scalable Approach for Multi-Million Gate Designs

Sarvesh Ganesan

The University of Texas at Austin, USA

Abstract: Contemporary System-on-Chip designs face unprecedented complexity challenges as gate counts reach multi-millions, transforming timing closure into a critical bottleneck within development cycles. This manuscript presents an innovative distributed timing analysis framework that harnesses HyperGrid technology to parallelize static timing verification across scalable computing environments. The architecture preserves timing accuracy while dramatically reducing execution times through strategic workload distribution. Key advances include hierarchical-aware partitioning mechanisms that maintain boundary integrity, synchronized constraint propagation ensuring consistent timing models across distributed segments, and sophisticated cross-boundary verification preserving critical path relationships. Integration protocols establish seamless compatibility with existing electronic design automation infrastructures, facilitating practical adoption without disrupting established signoff methodologies. The framework addresses fundamental challenges in distributed timing environments through innovative partition boundary management and detailed correlation preservation techniques. Results demonstrate substantial acceleration of timing closure activities while maintaining correlation within acceptable tolerances across all timing scenarios. This architectural approach transforms timing analysis from a sequential limitation into a parallel opportunity, enabling design teams to manage increasingly sophisticated circuits without proportional schedule extensions. The article proves particularly valuable in advanced nodes where corner combinations multiply verification requirements, establishing a foundation for future timing closure techniques in ultra-large-scale integration environments.

Keywords: Distributed timing analysis, System-on-Chip, HyperGrid, Static timing analysis, Parallel computing.

INTRODUCTION

Contemporary System-on-Chip implementations exemplify extraordinary engineering sophistication with billions of transistors meticulously arranged throughout diverse functional regions. This extraordinary integration density presents formidable timing closure challenges as clock frequencies push higher while power constraints grow increasingly stringent. Conventional timing verification techniques encounter significant barriers when applied to these multi-million gate architectures, resulting in prolonged development timelines and potential performance uncertainties. Verification complexity increases exponentially, not simply through expanded transistor counts but through the multiplication of timing scenarios across operational modes, voltage conditions, and manufacturing variations. Physical phenomena once considered negligible now require precise modeling, substantially expanding computational demands. "The traversal of timing graphs in modern SoCs represents a computational challenge analogous to large-scale scientific computing, necessitating parallel approaches previously reserved for specialized domains" (Merrill, D. *et al.*, 2012). Timing certification directly influences silicon performance and production yields, making methodological enhancements essential for competitive semiconductor creation. Despite considerable electronic design automation advancements, timing closure continues to present primary schedule hazards, especially in cutting-edge manufacturing processes where silicon variations multiply verification requirements dramatically.

Distributed timing analysis frameworks introduce a fundamental verification paradigm shift, converting timing closure from sequential processing bottlenecks into parallel workflows harnessing contemporary computing environments. This architectural strategy divides timing verification across multiple computational resources while preserving global timing coherence through advanced synchronization techniques. By segmenting timing graphs while maintaining critical path continuity, these frameworks deliver substantial performance improvements without compromising verification quality. "Cloud-based timing analysis changes verification economics fundamentally, allowing resources to scale with complexity rather than forcing increasingly difficult schedule compromises" (Sequeira, A. 2022). The distributed methodology addresses key SoC verification hurdles through hierarchy-conscious partitioning techniques, coordinated constraint management across boundaries, and thorough result consolidation mechanisms. Rather than simplifying timing representations to accommodate processing limitations, distributed frameworks maintain comprehensive timing precision while efficiently parallelizing

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

***Corresponding Author:** Sarvesh Ganesan

DOI- https://doi.org/10.5281/zenodo.16735364

computational tasks. Compatibility with established design processes enables practical implementation without disrupting proven methodologies, a crucial consideration for production environments. HyperGrid technology further enhances these capabilities through dynamic resource orchestration across diverse computing platforms, allowing timing verification to automatically adapt to available infrastructure while delivering consistent verification precision regardless of deployment configuration.

CHALLENGES IN FLAT SOC TIMING ANALYSIS

Modern semiconductor designs present extraordinary computational challenges for timing verification tools as gate counts reach many millions within a single integrated circuit. This complexity manifests not merely through expanded graph sizes but through exponential growth in path combinations requiring analysis. "Advanced technology nodes introduce timing effects requiring specialized algorithms whose computational complexity scales non-linearly with design size, creating verification bottlenecks previously unseen in the industry" (Vagenas, A. 2022). Flat analysis approaches must process connectivity graphs with billions of timing arcs while accounting for complex interdependencies between clock domains, asynchronous interfaces, and specialized circuit structures. Timing tools must simultaneously manage process-voltage-temperature variations across multiple corners, multiplying computational requirements accordingly. These challenges compound particularly in mixed-signal designs where analog and digital integration creates verification scenarios requiring specialized modeling approaches. The fundamental computational limitations encountered in flat analysis emerge directly from graph theory constraints, where complete path enumeration becomes mathematically intractable beyond certain complexity thresholds.

Traditional timing analysis methodologies face severe resource constraints when confronting contemporary System-on-Chip designs, particularly regarding memory utilization and runtime scalability. These limitations manifest both in workstation environments and dedicated server configurations, where even substantial hardware investments deliver diminishing returns beyond certain design complexity thresholds. "Conventional single-threaded timing analysis approaches experience memory bandwidth limitations fundamentally restricting their effectiveness for large designs regardless of available computational resources" (Murray, K. E., & Betz, V. 2018). Monolithic verification approaches require complete design representation in memory simultaneously, creating exponential growth in resource requirements as designs expand linearly. Engineering organizations face challenging economic decisions when timing verification requires specialized hardware accessible only periodically during design cycles. These constraints particularly impact development schedules during critical timing closure phases, where iteration cycles extend disproportionately for complex designs. Infrastructure limitations consequently drive compromised verification approaches where coverage completeness balances against practical schedule requirements.

Distributed timing verification introduces fundamental challenges in maintaining timing consistency across partitioned environments while preserving global signoff quality. Ensuring identical timing results regardless of partitioning strategy requires sophisticated synchronization mechanisms that guarantee consistent constraint propagation, matching delay calculation, and coherent variation modeling across boundaries. "Establishing timing correlation across distributed segments necessitates precise boundary condition management surpassing traditional hierarchical methodologies in both complexity and accuracy requirements" (Zhang, B. *et al.*, 2025). Clock tree interactions present particular challenges when signals cross partition boundaries, requiring special handling to maintain phase relationships and uncertainty propagation. Setup and hold correlation must persist throughout distribution to prevent artificial timing violations or dangerous over-optimization. Sign-off quality distributed verification requires exact matching between distributed results and traditional monolithic analysis, necessitating extensive validation methodologies proving mathematical equivalence. These consistency requirements create substantial algorithmic challenges beyond mere workload distribution, demanding innovations in timing representation, constraint propagation, and boundary management.

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

Table 1: Challenges in Flat SoC Timing Analysis (Merrill, D. *et al.*, 2012; Vagenas, A. 2022; Murray, K. E., & Betz, V. 2018)

Challenge Category	Key Issues	Impact on Verification	Mitigation Strategy
Computational Complexity	Billions of timing paths, Non-linear scaling with gate count, Multiple PVT corners	Exponential runtime growth	Distributed parallel processing using HyperGrid
Memory Requirements	Complete design representation, Detailed timing models, Comprehensive constraints	Resource limitations on standard hardware	Dynamic memory management with a partition-based approach
Resource Constraints	Limited workstation capabilities, Licensing restrictions, Processing bottlenecks	Extended verification cycles	Scalable cloud-based resource allocation
Result in Consistency	Variation between runs, Corner-to-corner correlation, Exception propagation	Potential signoff errors	Standardized verification methodologies with cross-validation
Integration Complexity	Mixed-signal boundaries, Clock domain crossings, Asynchronous interfaces	Specialized verification requirements	Interface-aware partitioning strategies
Timing Closure Iterations	ECO impact assessment, Incremental verification, Path tracking	Delayed convergence	Fast feedback loops with targeted analysis
Tool Limitations	Single-threaded algorithms, Monolithic design handling, Sequential processing	Verification bottlenecks	Distributed architecture with parallel execution

HYPERGRID ARCHITECTURE FOR DISTRIBUTED TIMING ANALYSIS

The HyperGrid architecture establishes a comprehensive computational framework specifically engineered for distributed timing analysis across complex System-on-Chip implementations. This infrastructure orchestrates timing verification through specialized resource management layers, maintaining analytical coherence while distributing computational workloads. At its core, HyperGrid implements containerized execution environments, ensuring identical tool configurations across distributed nodes, eliminating traditional inconsistencies encountered in heterogeneous computing scenarios. Central timing databases maintain global constraints while dedicated partition managers handle boundary synchronization through advanced caching mechanisms preserving timing relationships. "Distributed graph traversal systems require specialized communication protocols balancing message efficiency with comprehensive data propagation across computational boundaries" (Merrill, D. *et al.*, 2012). The control architecture employs hierarchical task scheduling, distributing workloads based on timing criticality assessments and monitoring resource availability. Memory management subsystems implement optimized graph representations across distributed resources, minimizing duplication while maintaining access efficiency throughout verification cycles. HyperGrid incorporates dynamic scalability features enabling seamless adaptation across computational environments ranging from dedicated workstations to enterprise cloud deployments. Unlike traditional static partitioning approaches, continuous performance monitoring with automated load balancing adjusts boundaries during execution based on observed timing characteristics, delivering near-linear scaling while maximizing throughput regardless of hardware configuration. "Field-programmable timing analysis frameworks demonstrate performance advantages through optimized hardware utilization, establishing principles directly applicable to distributed software implementations" (Murray, K. E., & Betz, V. 2018). Practical deployment succeeds through comprehensive integration with existing electronic design automation ecosystems, enabling adoption without disrupting established timing closure methodologies. Standardized format support enables seamless interchange with implementation tools through native interfaces, preserving constraint semantics and timing exceptions. Command compatibility layers create transparent execution environments where distributed analyses

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

appear identical to traditional approaches from user perspectives, eliminating learning barriers while maintaining familiar workflows throughout timing closure activities.

Table 2: Cross-Boundary Timing Integrity in Distributed Flat SoC Analysis Huang, T. W., & Wong, M. D. 2015; Bjerregaard, T. 2024)

Cross-Boundary Mechanism	Technical Implementation	Key Benefits	Challenges Addressed
Path Tracing Framework	Specialized trace buffers with compression algorithms at partition interfaces	Complete end-to-end path visibility across distributed segments	Path fragmentation in traditional distributed approaches
Context Preservation	Comprehensive metadata capture, including side-input sensitization and clock relationships	Maintains verification completeness despite computational distribution	Information loss at partition boundaries
Setup/Hold Correlation	Synchronization frameworks maintaining explicit relationships between complementary analyses	Consistent timing relationships despite separate computational handling	Artificial timing improvements create reliability risks
Common-Path Identification	Algorithms recognizing shared segments between setup and hold paths	Ensures consistent treatment across partition boundaries	Inconsistent constraint application in distributed analysis

METHODOLOGY FOR DISTRIBUTED TIMING ANALYSIS

Effective distributed timing verification demands sophisticated methodologies maintaining analytical precision while enabling computational parallelism, with strategic partitioning serving as the essential foundation for balancing workload distribution against interface minimization. Unlike physical partitioning techniques utilized during implementation phases, timing partitioning must preserve logical timing relationships while establishing computationally balanced segments appropriate for parallel execution. HyperGrid utilizes multi-dimensional partitioning algorithms evaluating connectivity concentration, clock domain boundaries, and physical placement when determining verification partition boundaries. These techniques prioritize critical path preservation, ensuring timing-sensitive circuits remain within cohesive partitions whenever feasible to reduce boundary-crossing complications. "Incremental task graph partitioning enables progressive timing verification across distributed environments while maintaining analytical coherence throughout verification processes" (Zhang, B. *et al.*, 2025). Partition refinement proceeds iteratively, with feedback systems adjusting boundaries based on preliminary timing evaluations, identifying critical regions deserving of specialized consideration. Cone-based methodologies specifically address convergent paths, eliminating synthetic violations stemming from fragmented analysis across partition interfaces. Maintaining consistent constraint propagation across partition boundaries constitutes an essential requirement, necessitating advanced synchronization techniques ensuring uniform timing context regardless of computational segmentation. Centralized constraint management with distributed caching mechanisms ensures consistent exception definitions, clock specifications, and environmental parameters across all partitions despite independent computational execution. Dedicated translation mechanisms transform global timing requirements into boundary-specific conditions, establishing comprehensive verification context at partition interfaces. "GPU-accelerated partitioning algorithms create optimal computational boundaries while minimizing communication overhead, enabling practical distributed timing verification for production environments" (Zhang, B. *et al.*, 2024). Exception management systems maintain a thorough understanding of timing constraints throughout distributed environments, ensuring appropriate application of multi-cycle paths, false paths, and conditional analysis even when control signals span multiple partitions. Preserving critical path integrity represents the foremost challenge in distributed timing analysis, requiring specialized methodologies to ensure timing-critical circuits receive thorough verification regardless of partition assignment. Sophisticated path-tracking systems provide comprehensive visibility despite computational distribution, utilizing specialized data structures and maintaining complete path context across boundaries while minimizing communication

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

requirements. Critical path identification begins early during verification sequences, enabling dynamic partition modifications and ensuring timing-sensitive paths receive prioritized handling throughout distributed verification activities.

CROSS-BOUNDARY TIMING INTEGRITY

Maintaining timing integrity across partition boundaries represents a fundamental requirement for distributed verification environments, necessitating sophisticated mechanisms preserving analytical coherence despite computational separation. Advanced path tracing capabilities enable comprehensive timing verification spanning partition interfaces while maintaining complete context throughout distributed segments. Unlike traditional hierarchical approaches, where interface timing simply establishes boundary constraints, distributed analysis requires path reconstruction capabilities to preserve verification completeness. HyperGrid implements specialized trace buffers at partition boundaries, capturing comprehensive path history, including side-input sensitization, clock relationships, and constraint applicability. These mechanisms employ efficient compression techniques, minimizing storage requirements while maintaining verification quality throughout distributed environments. Path reconstruction algorithms reassemble segments during critical path reporting, presenting unified timing perspectives despite computational distribution across separate resources. "High-performance timing tools must implement efficient parallel architectures while maintaining mathematical precision essential for signoff quality verification results" (Huang, T. W., & Wong, M. D. 2015). Setting up and holding correlation preservation presents unique challenges in distributed environments, requiring specialized techniques to ensure appropriate timing relationships despite separate computational handling. Unlike independent analyses, where timing tests receive isolated verification, production signoff requires sophisticated correlation, ensuring consistent application of constraints, delays, and exceptions across related tests. Synchronization frameworks maintain explicit relationships between complementary analyses despite potential distribution across separate computational resources, ensuring identical circuit conditions, consistent clocking assumptions, and matching operating conditions regardless of partitioning strategy. "Circuit optimization techniques necessitate precise correlation between complementary timing tests, preventing artificial improvements creating real-world reliability hazards" (Bjerregaard, T. 2024). Common-path identification algorithms recognize shared segments between setup and hold paths, ensuring consistent treatment regardless of partition boundaries while maintaining verification quality matching traditional approaches.

Table 3: Scalability Characteristics for Distributed Flat SoC Timing Analysis Using HyperGrid (Sequeira, A. 2022; Murray, K. E., & Betz, V. 2018)

Design Complexity Category	HyperGrid Runtime Scaling	Traditional Approach Runtime Scaling	HyperGrid Memory Efficiency	Traditional Memory Efficiency	Timing Correlation Maintenance	Compute Resource Utilization
Small-Scale SoC	Highly Efficient	Acceptable	Minimal Requirements	Moderate Requirements	Excellent	Optimal
Medium-Scale SoC	Very Efficient	Challenged	Low Requirements	High Requirements	Very Good	Optimal
Large-Scale SoC	Efficient	Severely Challenged	Moderate Requirements	Exceeds Standard Hardware	Very Good	Near-Optimal
Very Large-Scale SoC	Moderately Efficient	Impractical	Manageable Requirements	Not Feasible	Good	Highly Efficient
Ultra Large-Scale SoC	Acceptable	Not Feasible	Higher Requirements	Not Feasible	Acceptable	Efficient

RESULT AGGREGATION AND VALIDATION

Effective consolidation of distributed timing reports represents an essential capability in production-quality verification environments, requiring sophisticated aggregation mechanisms transforming parallel results into coherent analytical perspectives suitable for engineering evaluation. HyperGrid implements specialized consolidation frameworks that collect distributed

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

verification outputs while preserving the complete context necessary for comprehensive timing assessment. These mechanisms employ intelligent filtering, prioritizing actionable information while preventing overwhelming detail characteristic of raw distributed results that would otherwise hinder effective engineering interpretation. Path merging algorithms reconstruct complete timing paths crossing partition boundaries, presenting unified verification perspectives despite computational distribution across separate resources. Temporal organization techniques arrange results chronologically based on actual signal propagation rather than computational completion, enhancing engineering understanding of circuit behavior throughout critical paths. "Electronic design automation frameworks must transform detailed verification data into actionable information enabling effective engineering decisions throughout semiconductor development processes" (Liu, X. *et al.,* 2023). Critical path grouping algorithms identify common failure mechanisms across distributed results, highlighting systematic timing challenges requiring architectural attention rather than point-fix approaches addressing symptoms rather than underlying causes. Comprehensive validation methodologies ensure analytical integrity despite computational partitioning, with systematic protocols comparing distributed results against golden references established through traditional analysis on simplified test cases. These methodologies specifically target partition boundary validation, verifying proper constraint propagation and path handling across computational interfaces. Corner case identification techniques ensure comprehensive verification covering unusual timing scenarios, potentially revealing distribution weaknesses that might otherwise remain undetected until late development stages. Correlation analysis quantifies result differences between distributed and monolithic approaches, establishing confidence levels for production deployment while maintaining verification quality essential for semiconductor manufacturing. Through these aggregation and validation capabilities, distributed timing analysis delivers practical verification results directly applicable to production development despite the underlying computational complexity inherent in modern System-on-Chip designs.

Table 4: Cross-Boundary Timing Integrity Mechanisms in HyperGrid Framework (Zhang, B. *et al.,* 2025; Huang, T. W., & Wong, M. D. 2015)

Boundary Management Feature	Implementation Approach	Traditional STA Limitation	HyperGrid Enhancement	Primary Benefit
Constraint Propagation	Centralized Repository with Local Caching	Inconsistent Constraints Across Hierarchies	Synchronized Constraint Application	Uniform Timing Context
Clock Domain Handling	Phase Relationship Preservation	Limited Cross-Domain Analysis	Complete Clock Tree Visibility	Accurate Multi-Domain Timing
Critical Path Tracking	Dynamic Path Reconstruction	Path Fragmentation at Boundaries	End-to-End Path Preservation	Comprehensive Critical Path Analysis
Exception Management	Global-to-Local Translation	Exception Loss at Interfaces	Complete Exception Propagation	Correct Timing Exception Handling
Reconvergent Path Analysis	Cone-Based Partitioning	Missing Path Correlations	Maintained Path Relationships	Prevents False Violations
Setup/Hold Correlation	Synchronized Analysis Framework	Decorrelated Timing Tests	Consistent Delay Assumptions	Reliable Signoff Quality
Result Aggregation	Intelligent Filtering and Merging	Disconnected Timing Reports	Unified Timing Perspective	Actionable Verification Results

CONCLUSION

Distributed flat SoC timing analysis using HyperGrid technology represents a fundamental advancement in electronic design automation capabilities for modern semiconductor development. The strategic decomposition of timing verification workloads across parallel computing environments delivers substantial benefits through computational efficiency, resource optimization, and architectural flexibility. Timing verification becomes notably more responsive as monolithic analysis transforms into

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

distributed processing, enhancing productivity while supporting increasingly complex designs. Hardware utilization improves markedly as computational resources dynamically adjust based on verification demands, expanding during peak requirements while releasing capacity during intermediate phases. Maintenance complexity diminishes through logical partitioning, where timing boundaries create natural verification segments addressing specific circuit domains. Engineering effectiveness increases through streamlined verification approaches concentrating on critical paths rather than comprehensive reanalysis. Performance monitoring across distributed segments provides technical teams unprecedented visibility into timing relationships, facilitating targeted optimization. Forward-looking organizations must balance architectural innovation with practical transition strategies rather than pursuing complete methodology replacement without consideration for established workflows. This measured implementation approach yields significant verification benefits while maintaining quality assurance throughout the adoption phases. As distributed timing methodologies mature, semiconductor companies practicing systematic implementation establish lasting competitive advantages through technical agility, operational excellence, and improved capacity for aggressive technology scaling.

REFERENCES

1. Merrill, D., Garland, M., & Grimshaw, A. "Scalable GPU graph traversal." *ACM Sigplan Notices* 47.8 (2012): 117-128.
2. Sequeira, A. "Next-Generation Distributed Static Timing Analysis On The Cloud." *semiconductorengineering.com*, May (2022).
3. Vagenas, A. "Timing analysis of VLSI circuits in advanced technology nodes." *Thesis for: Integrated M.Sc. in Electrical Engineering, ResearchGate,* Jul. (2022).https://www.researchgate.net/publication/378403035_Timing_analysis_of_VLSI_circuits_in_advanced_technology_nodes
4. Murray, K. E., & Betz, V. "Tatum: Parallel timing analysis for faster design cycles and improved optimization." *2018 International Conference on Field-Programmable Technology (FPT)*. IEEE, (2018).
5. Zhang, B., Chang, C., Chiu, C. H., Lin, D. L., Sui, Y., Chang, C. C., ... & Huang, T. W. "iTAP: An Incremental Task Graph Partitioner for Task-parallel Static Timing Analysis." *Proceedings of the 30th Asia and South Pacific Design Automation Conference*. (2025).
6. Zhang, B., Lin, D. L., Chang, C., Chiu, C. H., Wang, B., Lee, W. L., ... & Huang, T. W. "G-PASTA: Gpu-accelerated partitioning algorithm for static timing analysis." *Proceedings of the 61st ACM/IEEE Design Automation Conference*. (2024).
7. Huang, T. W., & Wong, M. D. "OpenTimer: A high-performance timing analysis tool." *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, (2015).
8. Bjerregaard, T. "Adopting AI-Based Circuit Optimization and Migration." *www.electronicdesign.com*, Jun. (2024).
9. Liu, X., Liu, Y., Lu, T., Yang, F., & Yang, J. "Electronic Design Automation Tools." *Handbook of Integrated Circuit Industry*. Singapore: Springer Nature Singapore, (2023). 829-859.

Source of support: Nil; **Conflict of interest:** Nil.

Cite this article as:
Ganesan, S. " Distributed Flat SoC Timing Analysis Using HyperGrid: A Scalable Approach for Multi-Million Gate Designs." *Sarcouncil Journal of Multidisciplinary 5.8* (2025): pp 24-30.

Copyright © 2021 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher