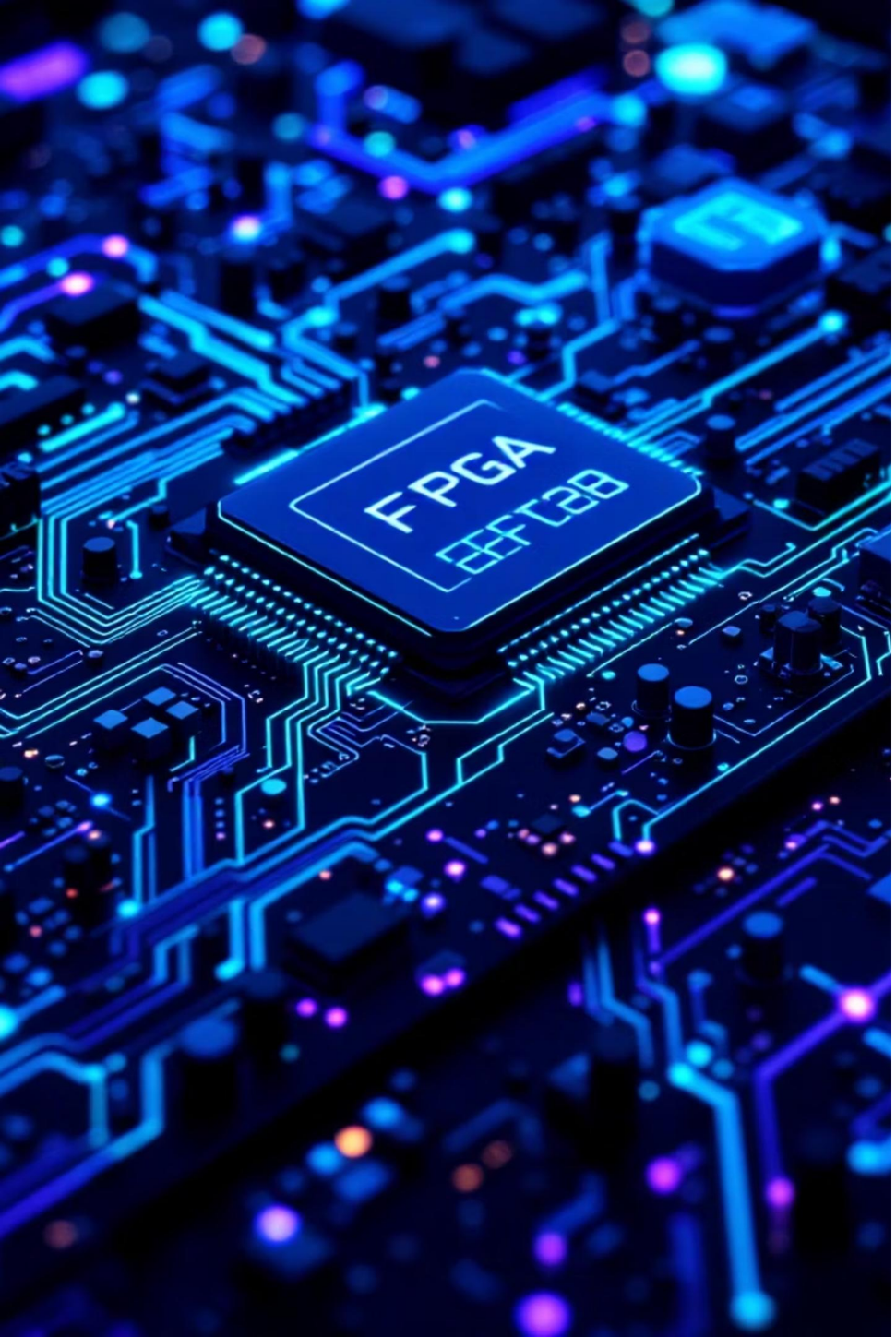


LLM-Augmented FPGA Timing Closure

Toward Intelligent Static Timing Analysis Agents

Saher Elsayed | University of Pennsylvania | Architecture 2.0 Workshop @ ASPLOS 2026

The Problem: Timing Closure Bottleneck

Modern FPGA designs contain millions of logic elements running at 500+ MHz. Achieving timing closure — the state where all paths meet their timing requirements — is one of the most challenging, time-consuming stages of FPGA development.

Weeks of Manual Iteration

Experienced engineers spend weeks per design iterating on timing constraints and fixes.

Deep Expert Knowledge

Requires mastery of constraint patterns, violation types, and RTL restructuring techniques.

Knowledge Bottleneck

Experiential know-how is hard to codify or automate — locked in individual engineers.

Key Question: Can LLMs + RAG replace manual expert knowledge for timing diagnosis?

Background: LLMs in EDA — Opportunity and Gap

What LLMs Already Do in EDA

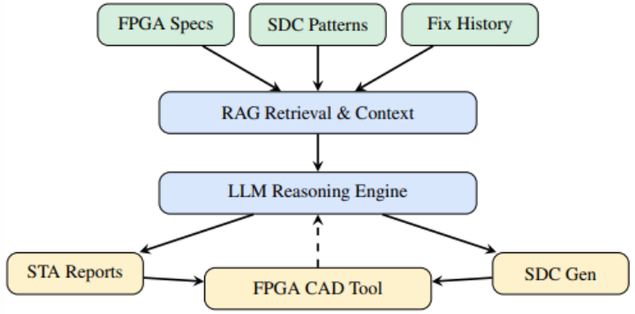
- RTL generation (Blocklove et al., TODAES 2025)
- HLS optimization (TimelyHLS, IEEE COINS 2025)
- Code debugging and synthesis guidance

Critical Gaps for FPGA Timing Closure

- General-purpose LLMs lack FPGA-specific timing knowledge
- Hallucination risks → incorrect SDC constraints
- No practical deployment path with CAD tool integration
- Timing closure remains largely unexplored territory

TIMINGLLM fills this gap with RAG-grounded LLM reasoning for FPGA STA.

TIMINGLLM Framework Architecture



Three primary components working together to deliver intelligent timing closure:

Domain Knowledge Base

- FPGA device timing models (logic: 0.15–0.20ns, DSP: 1.0–1.5ns, memory: 1.2–1.8ns)
- 234 curated SDC constraint patterns (CDC, async resets, multi-cycle arithmetic)
- 1,523 anonymized historical fix records from production designs

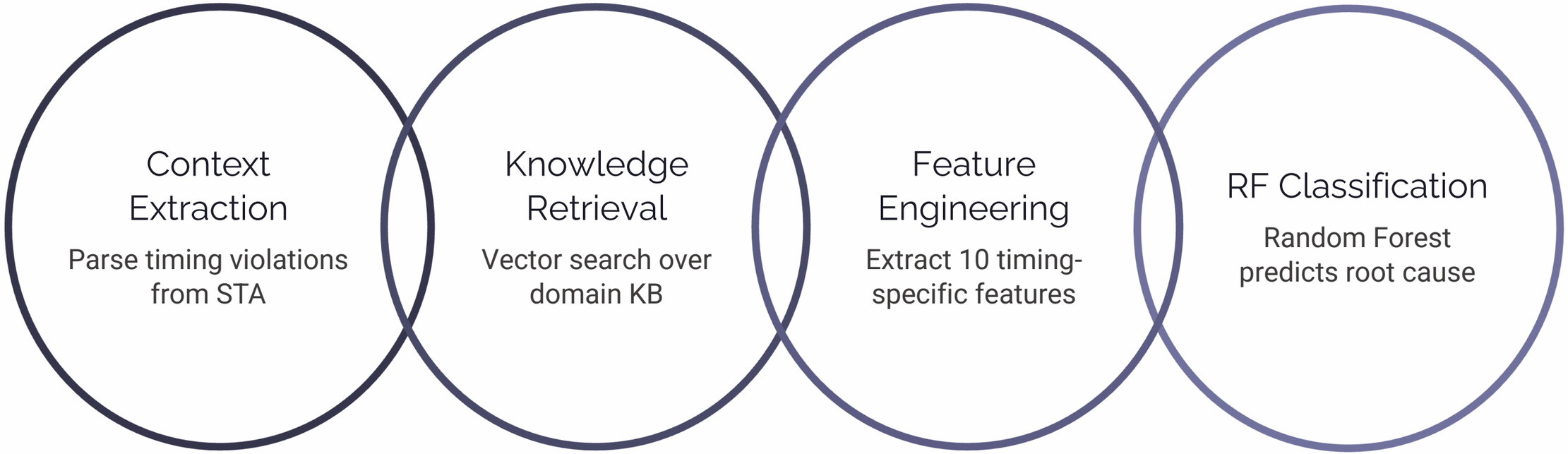
RAG-Augmented Reasoning Engine

- Context extraction from STA reports
- Vector database retrieval
- 10-feature Random Forest root-cause classifier
- Template-based SDC fix generation

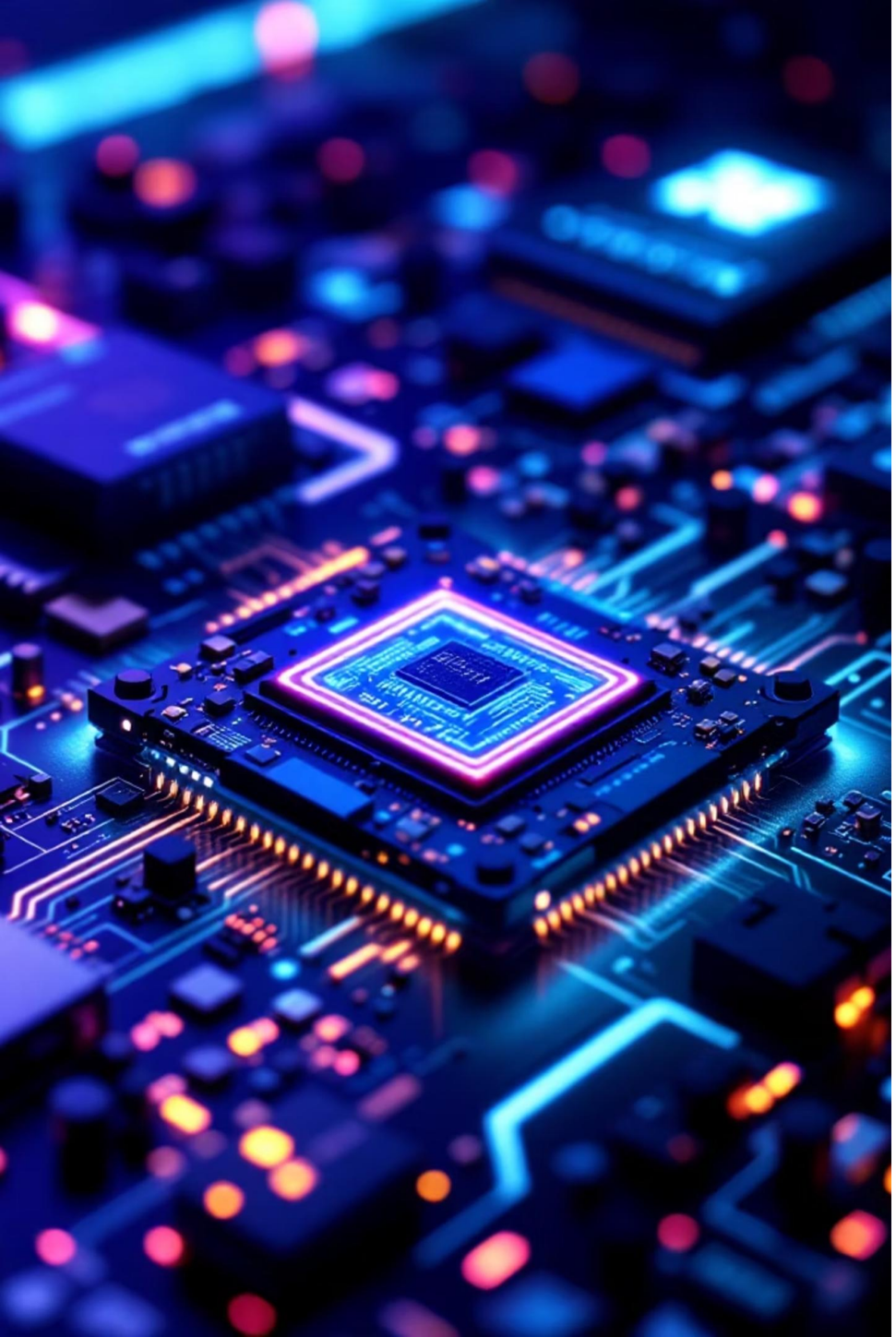
Tool Integration Layer

- Ingests STA timing reports
- Outputs SDC constraint fixes
- CAD feedback loop with place-and-route tools

5-Step Reasoning Pipeline



This end-to-end pipeline transforms raw STA timing reports into actionable SDC constraint fixes. Ablation result: Removing RAG causes a **−19% F1 drop** (0.84 → 0.68), confirming domain grounding is essential to accurate diagnosis.



Experimental Setup

12 FPGA designs across 3 application domains — target frequencies 300–450 MHz, 658 total violations across 5 categories, validated with 5-fold cross-validation.

🌐 Networking

- 100G Ethernet
- PCIe Gen5
- Switch Fabric
- RDMA NIC

📡 Signal Processing

- FFT
- MIMO Beamformer
- Radar DSP
- Filter Bank

🤖 AI Accelerator

- CNN Engine
- Transformer
- GEMM
- Quantized NN

Violation Classification Results

Violation Type	F1 Score
Clock Domain Crossing	100%
Multicycle Path	100%
Architectural Bottleneck	97%
Missing Constraint	70%
Hold Violation	24% ⚠️
Overall	**82%**

82%

Overall F1

82% Precision, 84% Recall

100%

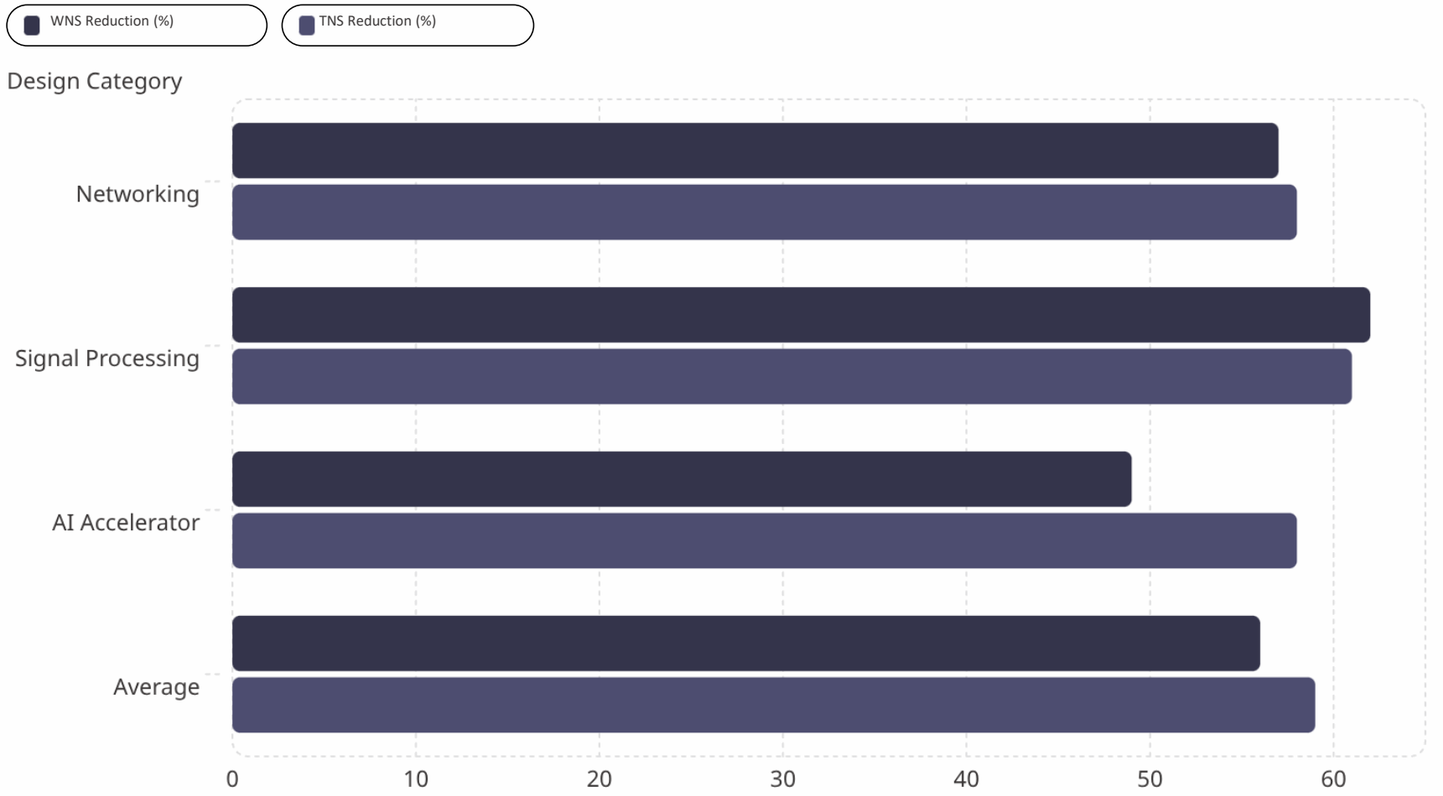
CDC & Multicycle

Perfect detection for top violation classes

Hold violations remain challenging — physical placement details are not captured in the current feature set.

Timing Slack Improvement Results

LLM-generated SDC fixes reduce worst negative slack (WNS) and total negative slack (TNS) across all design categories:



📈 **Signal Processing** achieves the highest improvement (62%) due to regular datapath structures that map well to constraint-based fixes.

⚠️ **AI Accelerators** (49%) include complex memory patterns that may require RTL changes beyond constraint fixes alone.

Extended Analysis: Scalability & Cross-Domain Transfer

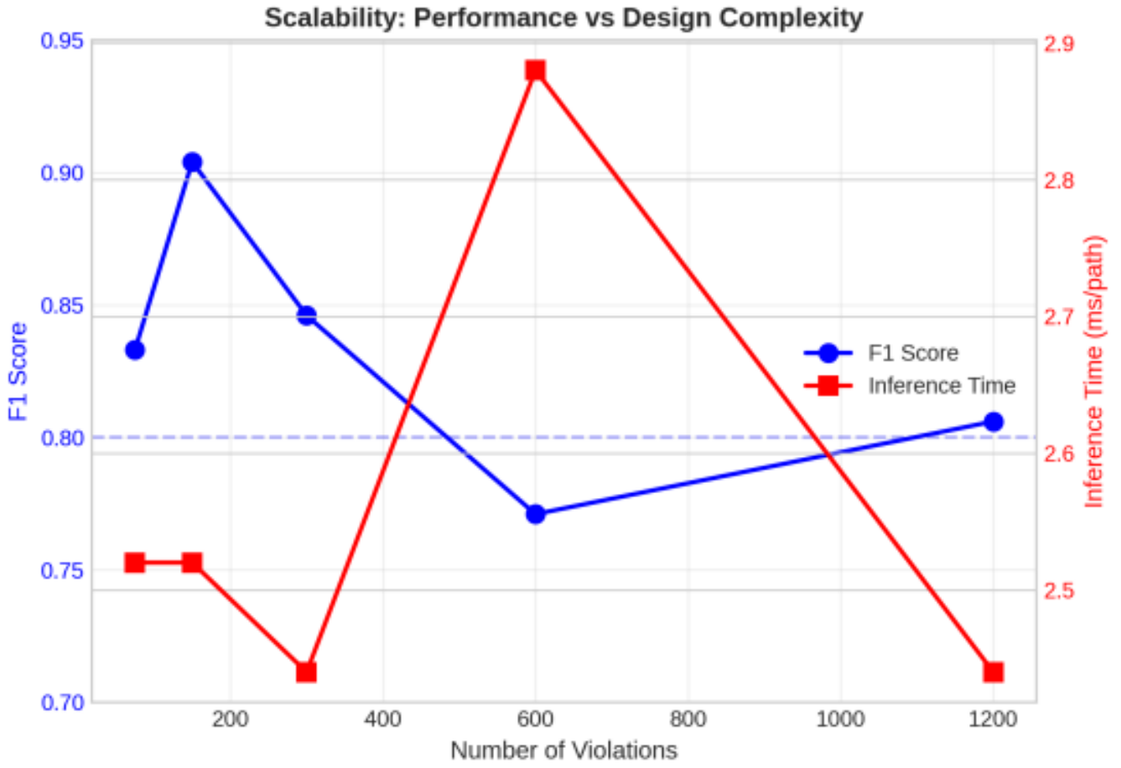
Scalability

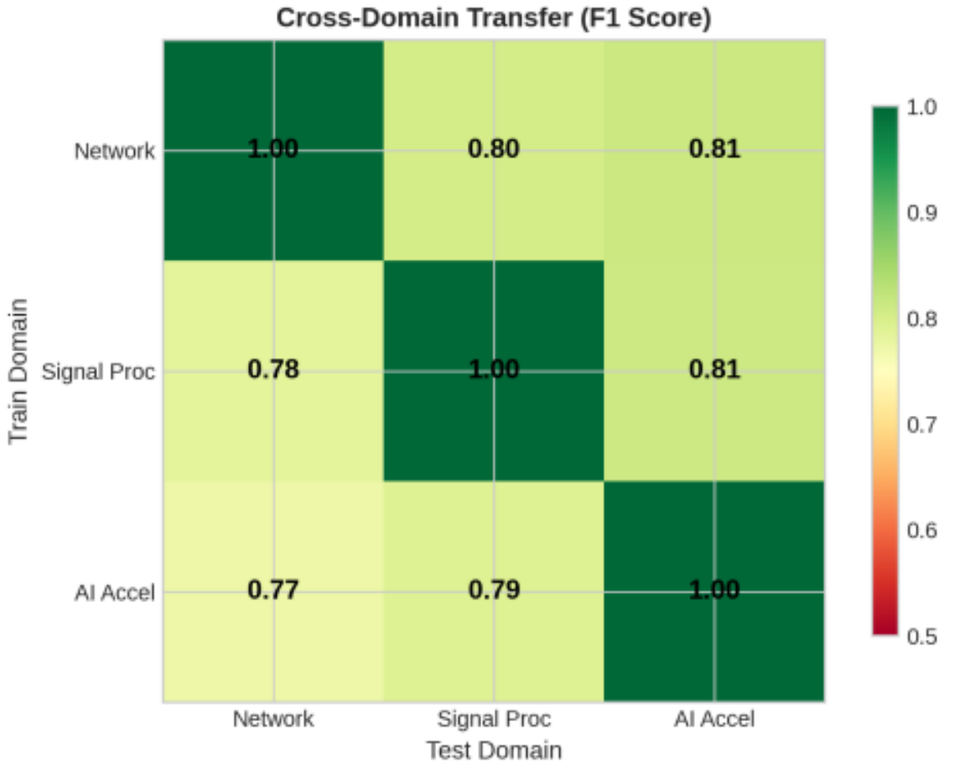
- Evaluated from **75 to 1,200 violations**
- F1 scores remain stable between **0.77–0.90**
- Inference time consistently **sub-3ms per violation** regardless of design size
- Demonstrates practical applicability to **large industrial designs**

Cross-Domain Transfer (F1 Matrix)

Train → Test	Networking	Signal Proc	AI Accel
Networking	1.00	0.80	0.81
Signal Proc	0.78	1.00	0.81
AI Accel	0.77	0.79	1.00

In-domain: near-perfect (1.0 F1). Cross-domain: ~20% degradation. Best transfer: Networking → AI Accel (18.7% degradation) due to shared CDC patterns.


Scalability: Performance vs Design Complexity
F1 Score
Inference Time (ms/path)
Number of Violations
0.95
0.90
0.85
0.80
0.75
0.70
2.9
2.8
2.7
2.6
2.5
200
400
600
800
1000
1200
F1 Score
Inference Time


Cross-Domain Transfer (F1 Score)
Train Domain
Test Domain
Network
Signal Proc
AI Accel
1.00
0.80
0.81
0.78
1.00
0.81
0.77
0.79
1.00
1.0
0.9
0.8
0.7
0.6
0.5

Limitations, Future Directions & Conclusion

Current Limitations

- Hold violation F1 only 24% — physical placement details missing from feature set
- ~20% cross-domain degradation limits out-of-box generalization
- 12-design evaluation is encouraging but not exhaustive
- No baseline comparison against pure ML or rule-based systems yet
- Fix implementation cost (power, resource utilization) not evaluated

Future Directions

- Placement-aware features to improve hold violation detection
- Multi-agent architecture with specialized agents per violation class
- Domain-specific LLM fine-tuning on FPGA corpora
- Cross-vendor evaluation (Intel Agilex vs. Xilinx Ultrascale+)
- RTL restructuring recommendations beyond constraint-only fixes
- Public benchmark dataset release

TIMINGLLM: Key Results at a Glance

82%	100%	56%	+19%
Overall F1 Score	CDC & Multicycle F1	Avg WNS Reduction	RAG F1 Contribution

<3ms

Inference per Violation

TIMINGLLM establishes the foundation for autonomous timing closure agents that can significantly reduce FPGA design iteration cycles.