Harshith Kantamneni

kantamneniharshith@gmail.com | +1 (414) 916-5799 | linkedin.com/in/hk4231 | github.com/Drogon4231 | Madison, WI 53726

M.S. ECE student specializing in GPU Architecture, Deep Learning Inference, and Parallel Computing
Seeking full-time roles in: GPU Performance Engineering, Deep Learning Systems, and HW-SW Co-Design

Skills

- **Languages:** C, C++, Python, CUDA
- **Tools:** PyTorch, TensorRT, Nsight Compute, gem5, McPAT, Slurm, ModelSim, Git
- **Domains:** GPU Architecture, Inference Performance Optimization, Parallel Computing, ML for Hardware

Education

University of Wisconsin-Madison — *Madison, USA*
M.S. in Electrical and Computer Engineering — Sep 2024 – Dec 2025

- Relevant Coursework: High Performance Computing, Machine Learning, Advanced Computer Architecture, Fault-Tolerant Computing, Introduction to Artificial Neural Networks

Vellore Institute of Technology — *Amaravati, India*
B.Tech in Electronics and Communication Engineering — Nov 2020 – May 2024

Certifications

- **NVIDIA Deep Learning Institute (DLI) – Getting Started with Accelerated Computing using CUDA C++**, 2025

Projects

ML-Guided CUDA Kernel Optimizer — Jan 2025 – May 2025
Python, PyTorch, CUDA, Slurm — GitHub

- Developed PyTorch model to predict CUDA kernel launch parameters (grid/block sizes) from workload dimensions.
- Reduced kernel tuning overhead by $>$**95%**, achieving up to **30% runtime improvement** on matrix workloads.
- Executed large-scale kernel benchmarks using **CUDA event timers** and automated Slurm scripts on HPC clusters.
- Demonstrated that learned configurations generalize across problem scales and GPU architectures.

ML-Assisted Task Graph Partitioning — Jan 2025 – May 2025
Python, XGBoost, pandas, Slurm — GitHub

- Modeled task graph features to predict optimal partitioning for GPU-accelerated runtime scheduling.
- Reduced configuration search time by **25% with $<$5% error**, expediting design-space exploration and CI evaluation.
- Executed large-scale runtime simulations on Slurm-managed clusters, automating performance profiling pipelines.

Architectural Performance Modeling using gem5 — Sep 2025 – Present
gem5, Python, McPAT, Slurm

- Profiled and compared **TimingSimple**, **Minor**, and **O3 CPU** models in gem5 for CPI, latency, and power trends.
- Executed **IAXPY, SAXPY, and DAXPY** workloads under different cache hierarchies and clock domains.
- Automated simulation and power extraction flows using Python and McPAT for rapid SoC-level performance analysis.
- Correlated instruction mix and performance counters to identify microarchitectural trade-offs in pipeline design.

5-Stage Pipelined RISC CPU (WISC-F24) — Aug 2024 – Dec 2024
Verilog, ModelSim

- Designed pipelined CPU with hazard detection, forwarding, and branch handling logic.
- Verified cycle accuracy with full instruction coverage using waveform-based debugging in ModelSim.
- Applied microarchitecture design concepts directly relevant to GPU SM and CPU pipeline development.