Arpit Sakhreliya

+1 (240) 462-4258 | arpits@umd.edu | LinkedIn | Portfolio

Education

MEng. Electrical and Computer Engineering, University of Maryland, College Park **August 2024 - Exp. May 2026**

Courses: Computer Architecture | CMOS VLSI Design | Embedded Systems Design | Domain-Specific Architecture Design

BE Electrical Engineering, Gujarat Technological University, India **October 2018 - July 2022**

Courses: Analog and Digital Systems | Semiconductor Devices and Technology | Signal Processing | Embedded Systems

Skills

HDL: Verilog, SystemVerilog
Programming: Python, Perl, TCL, C++, Shell scripting, Linux commands
Tools: Cadence Virtuoso, Innovus, Xilinx Vivado, Synopsys Design Compiler, PrimeTime, Spectre
Protocols: AXI, AHB, APB, UART, SPI, I2C, MESI, MOESI
Concepts: RTL design, UVM, timing and waveform analysis (STA, SDC, CDC), Clock Tree Synthesis, dataflow optimization, pipelining, power, performance, and area (PPA) optimization, gate-level synthesis, P&R, cycle-accurate simulation

Work Experience

Teaching Assistant | Computer Organization | UMD **January 2025 - June 2025**

- Instructed 20 students on **MIPS and RISC-V microarchitecture topics**, **pipelining**, hazards, and cache coherence. Mentored 4 semester projects focused on ISA, datapath and control-logic design, memory subsystems, and hierarchy.

Junior Research Fellow | SysIDEA Robotics Lab | IIT Gandhinagar, India **October 2021 - April 2024**

- Developed and verified pipelined control datapaths on NI sbRIO for a multi-rate actuator system, using clock-domain partitioning and clock distribution with **STA and CDC analysis** to achieve **30 kHz inner- and 5 kHz outer-control loops**.
- Implemented fixed-point, pipeline-optimized compute blocks for Kalman filter, disturbance observer, control loop, and sensor datapaths, **achieving sub-1us latency and timing closure at 40 MHz**.
- Engineered a 20 kHz ARM-based IMU acquisition system, processing raw data into quaternion representations while ensuring timing consistency and data integrity over SPI; **acknowledged in a journal paper** (link).

Projects

Multiply-Accumulate (MAC) Unit Design | Cadence Virtuoso at 14nm | Spectre **January 2025 - May 2025**

- Designed a MAC unit using a 6-bit unsigned **Dadda multiplier, Kogge-Stone adder, and 19-bit synchronous register** file in Cadence, **achieving 2.225 GOPS throughput.**
- Validated **ASIC layout-to-schematic with DRC/LVS**, resolving violations to ensure manufacturable, error-free design.
- Analyzed Spectre transient and critical-path timing; applied **layout-aware floorplanning and placement**, along with transistor sizing and logic optimization, to **close timing (899 ps)** and reduce power **from 0.43 pJ/op to 0.291 pJ/op**.

FPGA-based Edge AI Accelerator | Xilinx Vivado | Nexys FPGA **August 2024 - October 2024**

- Implemented a 16-bit floating-point ALU on the Nexys 4 DDR FPGA using an **FSMD architecture in Verilog,** supporting FMA, Kalman filtering, FFT, and a **4x4 systolic-array** multiplier for GPU-style parallel edge-ML workloads.
- Defined a SIMD instruction set, improving instruction throughput and **reducing data latency from 5 ms to 3.8 ms.**
- Built and debugged pre-silicon **RTL testbenches** with SVA and functional coverage to verify control and interface logic.

Out-of-Order RISC-V cycle-accurate Simulator | Tomasulo Algorithm **August 2024 - October 2024**

- Simulated a **cycle-accurate, out-of-order RV32I-F processor in C** using Tomasulo's Algorithm, incorporating register renaming, instruction scheduling, out-of-order execution, and in-order commit across a **seven-stage pipeline**.
- Engineered a centralized instruction queue, reorder buffer, and memory-conflict queue with address-based disambiguation to manage **data and control hazards**, enhancing instruction throughput within the pipeline.
- Integrated multi-cycle functional units with a shared writeback port, reducing **structural hazards**; validated cycle accuracy and execution **correctness using benchmarks and trace matching.**

Cache Simulation and Performance Analysis | MESI coherence protocol | CACTI **August 2024 - October 2024**

- Modeled a **trace-driven multi-core cache simulator** with configurable cache parameters, implementing MESI snoopy coherence and LRU replacement for set-associative caches; tracked miss rates and demand fetches.
- Verified state transitions, invalidations, and memory consistency using **automated tests and trace-based validation.**
- **Characterized cache architectures using CACTI** by evaluating access latency, hit/miss rates, AMAT, and energy-delay product under **SPEC CPU2017 benchmark workloads to analyze energy-performance trade-offs.**