

Muhammad Farhan Azmine

540-557-8751 muhammadfarhan@vt.edu [\[Website-Portfolio\]](#) [\[Github\]](#) [\[linkedin.com/Muhammad Farhan Azmine\]](#)

Education

Virginia Tech (GPA: 4.00 / 4.00)

PhD in Computer Engineering (Direct PhD; MS completed)

Blacksburg & Alexandria, VA

Aug 2022 – May 2027 (Expected)

- **Relevant Coursework:** Advanced Digital Design, Advanced Computer Architecture, Testing VLSI Techniques, VLSI Device Modeling, Advanced Analog IC Design, Deep Learning, Advanced Machine Learning, Computation in Data Science

Publications

- **Muhammad F. A.,** Li, R., Sharma, G., & Yi, Y. (2025). SpikeSpec: On-Chip Learning Neuromorphic Accelerator for Spectrum Sensing. *IEEE Transaction CAD*, Feb 2025.
- Li, R., **Muhammad F. A.,** Sharma, G., & Yi, Y. (2025). Efficient Digital Architecture of Spiking Encoders. In *Proc. ISQED 2025*.
- Lin, C., **Muhammad F. A.,** Liang, Y., & Yi, Y. (2024). Neuro-Inspired AI Accelerator for 6G Networks. *Front. Comput. Neurosci.*
- Lin, C., **Muhammad F. A.,** & Yi, Y. (2023). Accelerating Wireless Communications with FPGA-Based AI. In *ICCAD 2023*.

Recent- Tapeout

- Taped out a RV32I RISC-V CPU on Sky130 PDK (21.8K cells, 334.9K μm^2 , 15.9 mW), with full ISA verif. & post-layout validation.

Research & Design Experience

Research Assistant (Functioning as RTL Engineer – Complex Digital System Design & Verification)

June 2023 – present

Supervisor : Dr. Yang (Cindy) Yi

RISC-V 5 stage Pipelined CPU Tapeout on Sky130 PDK with Genus & Formal Verification Framework with JasperGold

[\[Github link\]](#)

Fall 2025

- Designed and verified using SystemVerilog a 5 stage pipelined RV32I RISC-V CPU with forwarding circuit to solve hazard, taped out on Sky130 PDK (21.8K cells, 334.9K μm^2 total area, 15.9 mW total power), meeting timing across FF/SS/TT corners in Cadence Genus.
- Developed a UVM-lite environment with abstracted register/memory monitors and property-based formal checks for 40+ RV32I instructions using Cadence JasperGold and Xcelium, achieving 100% proof convergence, 98% coverage, and 80% debug effort reduction.
- Reduced task flow by 80% by creating a Makefile script to automate C code disassembly into LLVM IR for R32IV CPU using the risc64-gcc-elf toolchain in Ubuntu Linux 24.04 OS.

RTL Design and Verification of Direct-Mapped Cache Memory with RAM and Control Unit using Xilinx Vivado

[\[Github link\]](#)

Fall 24

- Designed and verified in SystemVerilog an integrated direct-mapped cache memory system comprising cache memory, RAM, and a control unit, with a 15-bit address input for a 128-bit cache line and 1024 entries.
- Ensured 100% efficient cache hit/miss synchronization using read/write handshake signal verification and an FSM-based controller.
- Performed cache hit/miss determination through comparison of tag bits and valid bit.

PPA optimization of Spiking Neural Network RTL architecture design with On-Chip learning for Spectrum-Sensing

[\[Github link\]](#)

Fall 22 - Spring 24

- Achieved 100% SystemVerilog Post-Si validation through hardware–software co-integration on Zynq CPU, enabling real-time sample delivery to RTL (PL) and result capture via ZynQ AXI-UART (PS), with a custom C++ parser for CSV input processing and output generation.
- Reduced RTL area by 60% in SystemVerilog using resource-shared adders and LUTs via serialization with SIPO shift registers.
- Throughput increase by 58 MHz with critical path balancing between priority encoder & exponential approximator
- Reduced latency by 50% using simple dual port memory ram in read-then-write mode for weight learning update

RTL integration of Ethernet-MAC IP interface with AI Accelerator inference chip for OFDM symbol detection

[\[Github link\]](#)

Spring 23-Fall 23

- Boosted design frequency by 100 MHz by implementing Clock Domain Crossing to achieve 200 MHz frequency for target accelerator through synchronizing with Ethernet PHY communication at 125 MHz using Ping-pong buffer, CDC AXI-handshake IPs and Asynchronous FIFOs
- Improved SystemVerilog verification coverage by 30% through modifying BIST testbench in frame data transfer between Ethernet-PHY and target accelerator by creating over 4 protocol-variant Ethernet frame stimulus patterns including error-injection & backpressure scenarios
- Reduced IP area usage in SystemVerilog RTL by 33.3% through DSP48E1 IP integration at RTL-level for inference MAC operation

Technical Skills

- **Techniques:** RTL design, STA, PPA, Clock-Domain-Crossing, UVM, DFT, FPGA-IP Integration, ASIC implementation
- **Languages:** SystemVerilog, Verilog, Matlab (Script), Python (OOP) [\[Github link\]](#), C++, Java [\[Github link\]](#), Tcl, Linux Shell
- **Tools:** Cadence Suite (Xcelium, Genus & JasperGold), TensorFlow [\[Github link\]](#), PyTorch, Scikit-learn [\[Github link\]](#), Vivado, Quartus
- **Concepts:** UVM, SVA Formal Verification, AXI-DMA, AXI4, UART, Ethernet-TCP/UDP, SPI, VGA, RISC V ISA, Cache memory mapping