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) *Blacksburg & Alexandria, VA*
PhD in Computer Engineering (Direct PhD; MS completed) *Aug 2022 – Dec 2026 (Expected)*

Bangladesh University of Engineering and Technology (BUET) *Dhaka, Bangladesh*
Bachelors of Science in Electrical & Electronics Engineering *Jan 2013 – Sep 2017*

- **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*.

Work Experience

Research Assistant *(Functioning as RTL Engineer – Complex Digital System Design & Verification)* *June 2023 – present*
Supervisor : Dr. Yang (Cindy) Yi

RISC-V Single-Cycle CPU Verification Framework with property based formal verification module Cadence JasperGold

[Github link] *Fall 25*

- **40+ RISC-V ISA instructions formally verified** through **SystemVerilog property checks**, achieving **100% proof convergence** on properties in **Cadence JasperGold** and **98% functional coverage** in **Cadence Xcelium** simulation, MET timing check in FF, SS, Typical cornet test in **Genus**
- **1:1 property-to-ISA mapping accelerated debug by 45%**, supported by waveform tracing in **Cadence SimVision & Xcelium**.
- **80% regression debug effort reduced** by abstracting register/memory monitors integrated into a **UVM-lite** style environment.

Spiking Neural Network RTL architecture design with On-Chip learning for Spectrum-Sensing in ZC702 Xilinx FPGA

[Github link] *Fall 22 - Spring 24*

- Achieved **100% SystemVerilog Post-Si validation** through hardware–software co-integration on Zynq SoC, 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.
- Ensured **0% state-space explosion** of updated weight register by coverage analysis & property check with **concurrent & immediate assertions**
- 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
- Improved performance accuracy **by 3.88%** through **priority encoder and fixed-point exponential approximator** for weight update engine

RTL integration of Ethernet-MAC IP interface with RNN inference chip in VC-707 FPGA for OFDM symbol detection

[Github link] *Spring 23-Fall 23*

- **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
- **Increased data transfer throughput 5x** by Ethernet-MAC IP integration with target accelerator design at 125 MHz frequency
- **Performed 100% AI algorithm verification** using **C++ simulator** $\langle 16, 10\rangle$ fixed-point format with template libraries like **std::vector**
- **Reduced IP area usage in SystemVerilog RTL by 33.3%** through **DSP48E1 IP** integration at RTL-level for inference MAC operation
- **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**

Technical Skills

- **Techniques:** RTL design, STA, Power optimization, 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, PyTorch, Vivado, Modelsim, Quartus
- **Concepts:** UVM, SVA Formal Verification, AXI-DMA, AXI4, UART, Ethernet-TCP/UDP, SPI, VGA, RISC V ISA, Cache memory mapping