Nahjay Battieste

954-494-7138 | nahjaybattieste@gmail.com | www.linkedin.com/in/nahjay-battieste-a84655224 | https://github.com/Nahjay
Portfolio | https://nahjay.github.io/

Education

University of Florida — **Aug. 2024 – Expected May 2027**
Bachelor of Science in Computer Engineering | GPA: 3.93 — *Gainesville, Florida*

Broward College — **Aug. 2022 – May 2024**
Associates of Arts | GPA: 3.95 — *Coconut Creek, Florida*

Technical Skills

Languages: C, C++, SystemVerilog, Python, Rust, VHDL, Assembly, Bash, PowerShell
Hardware / FPGA: Vivado (ILA), Quartus (Timing Analyzer), Questa, Cadence Virtuoso, UVM, STA, AXI4/AXI4-Stream, SystemRDL
Systems & Tools: Linux, Git, Perforce, Docker, GNU Make, CMake, GDB, Pytest

Experience

SmartSystems Lab | University of Florida — **Oct. 2024 – Current**
Undergraduate Researcher — *Gainesville, Florida*

- Architected the streaming frontend and control FSM for an RGB 3×3 convolution accelerator on Zynq-7000: line-buffer window generators feed ping-pong FIFOs, overlapping frame N+1 load with frame N compute at 1 window/cycle.
- Integrated three 3×3 systolic PE arrays (27 MACs/cycle) with an AXI4-Stream clock-domain-crossing output FIFO: ∼165 MHz post-route fMax (4.46 GMAC/s peak) and 94% MAC utilization at 64×64 frames in simulation.
- Built a Vivado/ModelSim/Python flow benchmarking fMax, resources, power, and accuracy against a NumPy golden model; verified bit-exact output over 24 scenarios and traced scaling limits to flip-flop FIFOs (85% of registers) and `fifo_sel` fanout.

Digital Design (EEL4712C) | University of Florida — **Jan. 2026 – Dec. 2026**
Undergraduate Teaching Assistant — *Gainesville, Florida*

- Taught digital design concepts (FSMs, datapath/control separation, pipelining, testbenching with assertions and CRV) in SystemVerilog; led labs, debugged student RTL through simulation, synthesis, and FPGA deployment, and graded assignments.

NVIDIA — **May 2026 – Aug. 2026**
GPU Firmware Engineer Intern, Chips Team — *Santa Clara, California*

- Rearchitected the settings (register-configuration) model for the GPU hardware-initialization boot microcode across NVIDIA's next-generation Rubin-architecture client GPUs (desktop, workstation, and notebook SKUs), consolidating structure definitions out of 40+ per-module C header files into a single schema-generated header derived from a unified configuration schema.
- Redesigned the microcode's build and linking process so configuration data could be packaged directly into the microcode's own data memory alongside its execution sequence, letting settings and code move and deploy together as one containerized unit for the first time.
- Built a build-time "burn" capability that collects finalized per-chip configuration values and writes them into the reserved data-memory region of the packaged microcode image, producing a fully self-contained, chip-specific boot image.

NVIDIA — **May 2025 – Aug. 2025**
GPU Firmware Engineer Intern, Infrastructure Team — *Santa Clara, California*

- Built a remote pipeline that validates firmware on target hardware, automating GPU ROM data extraction and Pytest suite execution.
- Created a tool that binary-searches Perforce changelists to find commits behind firmware build failures or performance degradation.
- Extended my 2024 containerized firmware build environment so VBIOS builds that could take hours now take minutes.

NVIDIA — **May 2024 – Aug. 2024**
GPU Firmware Engineer Intern, Infrastructure Team — *Santa Clara, California*

- Built a wizard and supporting scripts (Python, Batch, Bash, PowerShell, Docker) that generate custom containerized environments for building firmware packages and microcode components, cutting ramp-up time for new engineers and development time for existing ones.
- Optimized the boot sequence in C for next-generation GPUs, removing obsolete code and integrating new architectural features.
- Resolved MASM build issues by updating assembler tooling, fixing ROM data table directives, and improving assembly performance.

Cell Antenna Corporation — **May 2023 – Apr. 2024**
Electrical Engineer Intern — *Coral Springs, Florida*

- Built a full-stack direction-finding system (FastAPI, Rust, JavaScript) visualizing real-time antenna-array signal data.
- Developed a Python/DearPyGui GUI and an asynchronous Rust (Actix-Web) API to control a Yocto Linux-based SDR and a GPS device.
- Wrote and optimized time-critical Assembly firmware for an Atmel microcontroller, synchronized to a GPS PPS signal at microsecond precision; developed a Rust program to process raw IQ data from an FPGA-based SDR.

Projects

UF/Apple-Sponsored Hardware Design Contest, BNN FPGA Accelerator | *SystemVerilog*

- Optimized a parameterized FPGA BNN accelerator to 670.69 MHz fMax on Xilinx UltraScale via pipelining, retiming, and LUT reduction strategies, targeting the SFC 784→256→256→10 topology for MNIST digit recognition.
- Designed full RTL in SystemVerilog for a pipelined binary neural network classifier with AXI4-Stream config, input, and output interfaces; implemented a custom configuration manager, neuron processors (PN=8, PW=8), and memory hierarchy across 3 hidden/output layers.
- Developed a parameterized SystemVerilog testbench with functional coverage, assertions, and a behavioral reference model; achieved 100% verification pass rate across 50 test cases with measured average throughput of 21,545 outputs/sec.

RAVEN: RISC-V-Controlled FPGA Neural Processing Unit | *SystemVerilog, C, Python, SystemRDL, Vivado, Questa*

- Led control and runtime for a 5-person team building an INT8 NPU on the Ultra96-V2 (8×8 MAC array, 6.4 GMAC/s peak, 100 MHz timing closure); integrated a MicroBlaze V RISC-V core, AXI4-Lite interconnect, AXI DMA, and interrupt controller.
- Designed the NPU control plane in SystemVerilog: a SystemRDL-generated 36-register CSR block (RTL, C headers, docs), a depth-4 command queue, and a control FSM with bank interlock, watchdog, and 11 error codes.
- Built a ≤48 KiB bare-metal C runtime (HAL, drivers, model loader) running AD01, DS-CNN, and MobileNetV1-VWW from one binary.
- Overlapped DMA and compute across ping-pong banks, cutting command issue from 22 MMIO writes to ∼5 per tile (≤2 μs interrupt-to-ISR latency); verified with a 657-check testbench and on-board fault injection (0 mismatches vs. TFLite-Micro INT8 reference).

Leadership

FPGAtors | University of Florida — **Aug. 2026 – Current**
Secretary — *Gainesville, Florida*

- Handled event planning, club liaison work, and slide decks, and contributed to technical workshop planning and development.