Zhinan (Mike) Liu

Seattle, WA, 98115 | zliu24601@gmail.com | (206) 714-7829 | zliu43.github.io

Summary

MSEE candidate specializing in **embedded firmware, C/C++, and low-level optimization**. Experienced in building **performance-critical systems**, developing pipelines at scale, and applying **hardware/software integration skills** on microcontrollers, RTOS, and FPGA-based systems.

Education

University of Washington – MS in Electrical Engineering **Sep 2023 - Dec 2025 (Expected)**
Relevant Coursework: Computer Architecture, Digital Systems Design with FPGAs, Data Structures & Algorithms, Embedded Software Design

University of Washington – BS in Biochemistry **Sep 2016 - June 2020**

Experience

Data Analyst, Harborview Medical Center– Seattle, WA Nov 2020 – June 2024

- Engineered high-performance data pipelines **(Python, SQL, Jenkins)**, automating dataset generation and reducing runtime by 90%.
- Applied statistical modeling and algorithm design **(Python, R)** to large-scale clinical datasets, improving predictive accuracy of geriatric readmission models (F1 = 0.83)
- Designed modular, testable code frameworks to ensure reproducibility and scalability across large datasets
- Collaborated with cross-functional teams; contributed to 11 peer-reviewed publications.

Research Assistant, Lieber Lab – Seattle, WA August 2016 – August 2020

- Developed and optimized large-scale genomic analysis pipelines **(Python, Nextflow),** processing 400GB/week efficiently.
- Built automation for CRISPR data analysis, demonstrating ability to handle high-throughput, performance-critical systems.
- Implemented low-level optimization strategies (parallelization, memory-efficient algorithms) to accelerate analysis runtimes.
- Documented pipeline design and maintained version control in Git, improving collaboration and long-term reproducibility.

Projects

esp_simd: High-Level SIMD Library for ESP32-S3 https://github.com/zliu43/esp_simd
High-level C library wrapping Xtensa SIMD for vector ops on ESP32-S3; designed for safe alignment/saturation and drop-in use with esp-idf.

- Implemented with hand-written, branchless ASM with zero-overhead loops
- Ops: Basic math ops including add/sub/mul/dotp across int8/16/32 & float32; benchmarks show ~5-10x speedups vs. scalar for int types and ~3-5x speedup for float types
- CMake examples and unit tests; ready for integration, including with esp-idf functions

Tech: C/C++, Xtensa ASM, esp-idf, CMake

Skills

Programming Languages: C, C++, Python, Assembly (RISC-V, ARM, Xtensa), SystemVerilog, Bash
Embedded Systems & Firmware: ESP32, STM32, FreeRTOS, UART, I2C, SPI
Software Engineering & Tools: CMake, GCC, Git, Docker, Linux/Unix, Yocto, Microsoft Azure