Jamshed Ashurov

☎ +1 571 315 8339 | @ jamshed00ashurov@gmail.com | Portfolio | LinkedIn | GitHub

Education

UC San Diego — La Jolla, California, USA
BS in Computer Engineering — *Sep 2018 – June 2022*

UC San Diego — La Jolla, California, USA
MS in Computer Engineering — *Sep 2022 – March/June 2024 (Expected)*

Industry Experience

Ampere Computing — Portland, Oregon, USA
CPU Architect Intern — *June 2023 – Sep 2023*

- Collaborated closely with the Architecture and Design team to assess the Architectural Vulnerability Factor(AVF) of various microprocessor structures using advanced emulation techniques via Cadence Palladium.
- Engineered and implemented an automated solution, streamlining the evaluation process and creating an adaptable infrastructure for potential integration with forthcoming chip designs.
- Collaborated with the Memory Controller Team to apply machine learning methods for memory controller performance optimization.
- Led the establishment of an experimental framework and selection of suitable machine learning models for single-performance metric optimization and multi-objective optimization problems.

Nanome Inc. — San Diego, California, USA
Software Engineering Intern — *Jun 2019 – Sep 2019*

- Developed AR application utilizing C# on iOS devices, to aid in analyzing 3D protein structures without the requirement of a VR setup.
- Architected a robust local database system, integrating it into the application's framework to efficiently store and manage data elements retrieved from an online database source.
- Implemented preliminary scripts, fostering cross-platform versatility and laying the foundation for the application's potential expansion to other platforms.

Research Experience

UC San Diego Research Assistant — Jan 2022 - Present

Characterizing WebAssembly Performance in the Era of Serverless Computing

Jamshed Ashurov, Zixuan Wang, Jishen Zhao

- *ISSTA 2023 SRC*

Teaching Experience

UC San Diego Graduate Teaching Assistant — Jan 2023 – Present

Computer Networks — Professor Aaron Schulman

- Designed and developed the source code and the testing framework for students to implement Sliding Window Protocol and TCP Congestion Control mechanisms.
- Assisted students with understanding/debugging in building a simple router.

Projects

Advanced Microprocessor Design — *System Verilog, Verilator, Python, Bash*

- Improved 5-stage pipelined MIPS 32 processor with advanced optimizations in System Verilog, achieving significant performance gains across all benchmarks. Utilized bash and Python scripts to analyze and collect performance data.
- Implemented 8KB 8-component TAGE branch predictor, reducing mispredictions by 94.3%(99.1%) on average across all benchmarks compared to the baseline predictor.
- Developed a victim cache using NRU replacement policy, resulting in an average CPI reduction of 10.2%(19.9%) compared to the baseline across all benchmarks.

- Implemented DRRIP cache replacement policy, reducing dcache misses by 31.4%(80.3%) compared to the baseline direct mapped cache across all benchmarks.
- Modified the baseline in-order pipeline to scalar OOO processor with Partial Ordering for memory disambiguation policy. Achieved an average CPI reduction of 8.9%(13.15%) across all benchmarks compared to the baseline.

Intellectual Robotic Chess Player Arm *Python, TensorFlow, Keras, ONNX, Raspberry Pi*

- Implemented a three-way network communication system involving a laptop, robotic arm, and camera.
- Utilized computer vision techniques to capture and process real-time images of the chessboard.
- Utilized a range of convolutional neural network (CNN) models, including Xception, MobileNet, SqueezeNet, and NASNetMobile, to perform chessboard detection.
- Employed different runtimes, such as TensorFlow, Keras, and ONNX, to optimize the detection process.
- Integrated an AI chess engine, Stockfish, to generate the next move and coordinated the Raspberry Pi to control the robotic arm and execute the determined move.

Parallel Computing *CUDA*

- Developed and optimized CUDA-based convolutional layer for the LeNet-5 architecture, significantly improving inference performance while maintaining high accuracy.
- Converted 2D convolution operations into tiled matrix multiplications for better cache utilization to achieve faster execution time.

Snek Compiler *Rust, x86*

- Developed a compiler for the dynamically typed language Snek in Rust which incorporated essential language features such as arithmetic and binary operators, functions, dynamic data allocation, and other fundamental elements.
- Integrated the Mark and Compact garbage collection algorithm to efficiently manage memory and remove garbage data.
- Orchestrated the transition to Administrative Normal Form and subsequently to Intermediate Representation for effective code transformation and optimization.
- Introduced optimization techniques, including constant folding and variable propagation, to enhance the binary file size and improve program execution efficiency.

Dual Core Machine Learning Hardware Accelerator for Attention Mechanism

- Integrated an attention mechanism, enabling the computation of a weighted sum of values based on query-value similarity for efficient performance.
- Utilized RTL design, verification, logic synthesis, and place-and-route (PnR) methodologies to generate gate-level netlist and layout via Synopsis Design Compiler (DC) and Cadence Innovus.
- Implemented advanced low-power techniques, including pipelining, loop unrolling, asynchronous FIFO, and clock gating, to optimize the power consumption of the design.

Fault Tolerant and Scalable Surfstore *Golang, gRPC, Python, Bash*

- Developed a robust file-sharing application, closely resembling the functionalities of Dropbox.
- Implemented consistent hashing for even data distribution, minimal movement during node scaling, and fault tolerance for seamless operations across numerous nodes.
- Employed the RAFT consensus algorithm to establish leader election, log replication, and fault recovery mechanisms, ensuring high availability and data integrity within the distributed system.

Skills

Programming: C, C++, Java, Python, System Verilog, Rust, Golang, C#, Shell, Tcl, CUDA, Haskell, MATLAB, x86/ARM/MIPS Assembly.

Tools: Cadence Palladium/Virtuoso/Innovus, Synopsis Verdi/Design Compiler, ModelSim, Quartus, Questa, Verilator, Unity3D, Fusion360.

Technical Skills: Microprocessor Design, Compiler Design, Parallel Computing, Logic Design, Performance Profiling, Software Development, Reliable and Scalable System Development, Machine Learning, Universal Verification Method(UVM), Analog Circuit Analysis.

Soft Skills: Leadership, Teamwork, Public Speaking, Coaching, Teaching.

Additional Skills: Ballroom and Latin Dancing, Martial Arts, Rock Climbing.

Languages: Tajik (Native), Russian (Native), English (Professional).