Jack Poli

Email: jpoli1040@gmail.com | Phone: (815) 245-1477
LinkedIn: https://www.linkedin.com/in/jack-poli-42764a34a/
GitHub: https://github.com/jackp7

Education

University of Illinois Urbana-Champaign – Bachelor of Science in Computer Engineering
Graduated: December 2024

Technical Skills

Languages: C, C++, Python, JavaScript, React.js, SystemVerilog, Assembly (x86)

Tools & Platforms: Git, GitHub, GitLab, Docker, Linux/Unix

Areas: Operating Systems, Computer Architecture, Web Development

Projects

Competition Website – JavaScript, React.js & Python

- Built a full-stack web application to manage player eliminations, time until elimination, and results for a public online video game event.
- Created a responsive front-end with React.js for player registration and leaderboard tracking.
- Developed a Python backend to periodically fetch player statistics via a public game API, apply MMR-based elimination logic, and persist tournament state in AWS S3. Included automated elimination logic based on Eastern Time using a timezone-aware scheduler.
- Deployed the site on AWS using EC2 and S3, with automated scripts to fetch data, update standings, and trigger eliminations.

Custom Operating System – C & Assembly

- Designed and implemented a Unix-like OS kernel supporting multitasking across 3 independent terminals and up to 6 concurrent processes.
- Built core kernel components including paging-based memory management, process scheduling, interrupt handling, and syscall interface.
- Integrated a basic file system for user-level program execution with support for command-line I/O.
- Debugged over 2000 lines of low-level C and assembly code using QEMU and GDB in a bare-metal x86 development environment.
- Demonstrated system stability running long-lived user processes and responsive multiterminal support.

Out-of-Order Processor – SystemVerilog

- Designed and implemented a 5-stage, out-of-order RISC-style CPU with support for up to 32 in-flight instructions, using Tomasulo's algorithm and branch prediction.
- Built custom hardware modules including 16-entry reservation stations, a 32-entry reorder buffer, and a dynamic register renaming system to resolve hazards.
- Added three advanced features to the processor: a split load/store queue for improved memory-level parallelism, a two-level branch predictor with a branch target buffer for higher accuracy, and support for floating-point instructions in the ISA.
- Simulated and verified using extensive testbenches, performance benchmarks (e.g., CoreMark), and real-world workloads.
- Collaborated via GitHub in a structured sprint-based workflow with biweekly reports and peer code reviews.

Additional Information

- Open to travel and relocation as needed.
- Strong collaboration skills from group projects and version control practices using Git in team settings.
- Comfortable working in containerized environments with Docker and on Linux-based systems.
- Eager to explore new technologies in embedded systems, compilers, or performance optimization.
- Comfortable learning on the fly and diving deep into technical documentation and low-level tools.