Mingkai Zheng

Email: mz588@cornell.edu | Phone: +1 (646) 784-2461 | LinkedIn | Personal Website|GitHub

EDUCATION BACKGROUND

Cornell University 08/2021- 12/2022
Master of Engineering in Electrical and Computer Engineering, Overall GPA: 3.925/4.0
University of Liverpool (UoL) 09/2016 – 07/2020
Bachelor of Engineering in Electrical Engineering, Overall GPA: 3.94/4.0, In-major GPA: 3.97/4.0

PROGRAMMING SKILLS

Programming: Java, Python, C/C++, HTML, CSS, JavaScript, MATLAB, SystemVerilog HDL, RISC-V ISA
Version Control Tools and Frameworks: Git, ReactJS, Docker, MongoDB, gRPC, Envoy

WORK EXPERIENCE

Software Development Engineer Intern, Biren Technology, Shanghai 12/2020 – 03/2021

- Responsible for writing and maintaining Python scripts for various purposes, including comparing the data generated by C model and RTL design, generating a systemic report based on the comparison result, replacing the data stream generated from the simulator with macros defined in the C/C++ program, and a web creeper which reminds engineers when their testing jobs running in Jinkins finished.
- Increased familiarity with Linux OS commands, version control tools, and project development procedures.

PROJECT EXPERIENCES

Movie Tickets Reservation Platform
Team Leader, Datacenter Computing final project, Cornell University 03/2022 – 05/2022

- Led a team of 3 to design and implement an online movie ticket reservation system using microservices. The main functionalities include signing up and logging in to the system, reserving movie tickets for different movies in different theaters, cancelling reservations, and making, modifying, deleting personal movie reviews.
- Designed and implemented the frontend webpages by using **ReactJS**. Client-side stores local data by using **Redux**.
- Implemented the middle-tier logics (microservices) in Python.
- All microservices live in **Docker** containers. The microservices communicate with each other through **gRPC**.
- Applied **Enovy** proxy to realize load-balancing.
- Deployed the backend database by using **MongoDB**.

Implementation of a Push Box 3D Game in JavaScript
Team Leader, Computer Graphics final project, Cornell University 11/2021 – 12/2021

- Led a team of 4 and realized the backbone of a Push Box games by using **WebGL** framework.
- Included real-time animations such as floating gems and procedurally generated trees with leaves that periodically grow, change their color from green to yellow, and fall off to the ground.
- Implemented texture mapping and various shader effects written in **openGL** and **GLSL**.

Design of a 5-stage Pipeline Processor and Multicore Processor Based on RISC-V ISA
Team Leader, Computer Architecture course project, Cornell University 10/2021 – 12/2021

- Led a team of 4 and implemented a RISC-V based 5-stage pipeline processor having basic integer arithmetic operations, stalling logics, and fully bypassing logics written by using SystemVerilog HDL and Python.
- Implemented a 2-way set associative, write back, write allocate cache with LRU replacement policy and a direct-mapped, write through, write no allocate cache simulator in Python for verification by using PyMTL library.
- Designed a multicore processor system by composing single-core processors with specially design caches.
- Implemented **parallel quick sort and merge sort** in C language which are executable on our processor.

Improvement on xv6 Operating System
6.s081: Operating System Engineering, lab assignments, MIT 05/2021 – 07/2021

- Self-studied the operating system course provided by MIT online.
- Implemented two system call functions, including trace function that tracks the usage of a specific system call in processes, and sysinfo which collects the information of a running system, which are useful for debugging.
- Improved the speed of fork function by using lazy allocation and copy-on-write scheme, as new page table will not be allocated if the process calls exec right after fork which dumps the data in the page table of parent process.
- Realized preemptive multithreading implementation with a FIFO scheduler from scratch.

Bear Map Application
CS61B: Data Structure and Algorithms course project, UC Berkeley 08/2020 – 11/2020

- Implemented the N-Dimensional Tree for searching nearest point on map by using Java
- Implemented a map application with many features including highlighting the shortest path between two selected locations on map, autocompleting search entry, and label all locations for which the user is searching.