Andrew Helbig

(530) 616-1752
ajhelbig@ucdavis.edu

Objective

To work on interesting problems with a team of driven individuals.

Education

Bachelors of Science, Computer Science — Expected Graduation June 2024

University of California, Davis

Current GPA: 3.5/4.0

Skills

Intermediate C/C++ programming

Basic Bash/Shell Scripting

Basic CMake/make

Basic GIt

Basic Python programming

Most experience with Ubuntu Linux

Some experience working on Mac and Windows

Interests

Operating systems, computer architecture, parallel computing, parallel algorithms, scalable high performance software, graphics, game development, embedded systems.

Relevant Classes

ECS 36C Data Structures, Algorithms, & Programming

- Fell in love with efficient algorithms
- Implemented Ford-Fulkerson min/max flow algorithm and applied it to a scheduling problem using bipartite matching
- Grade Received: A

ECS 50 Computer Organization & Machine-Dependent Programming

- Fell in love with low level systems programming
- Wrote a tic tac toe game on Professor Nita's bare metal RISC-V game console simulator
- Grade Received: A+

ECS 154A Computer Architecture

- Created a single cycle 16 bit cpu in Logisim
- Wrote a simple compiler for the single cycle cpu assembly instructions
- Grade Received: A

ECS 154B Computer Architecture

- Built and simulated a 5 stage RISK-V CPU using Chisel
- Simulated 6 different array sum algorithm configurations using native multi threading and GEM5
- Grade Received: A

ECS 150 Operating Systems & Systems Programming

- Wrote a user-level threads library with preemption and round robin scheduler
- Wrote a simple file system using a file allocation table similar to MSFAT
- Grade Received: A

Relevant Current and Future Classes

- ECS 158 Parallel Architecture - Spring 2023 Registered
- ECS 201C Parallel Architecture - Spring 2023 Registered
- ECS 120 Theory of Computation - Spring 2023 Registered

Projects

Graph Box

Just a summer passion project to learn how to use cmake and an external graphics api called raylib using C++. The first goal of the project was to create a simple graph sandbox that allows for the quick and easy generation of graphs that can then be used for studying graph algorithms. I got as far as implementing a complete graph generator, grid graph generator, naive MST, and maze generation algorithms. The secondary goal of the project was to write clean code and structure my git repo to be as easy as possible to clone and start tinkering with. I feel I succeeded in this second goal, as all that is needed to get the program running after being cloned is the execution of the run script.

Graph Box repo: https://github.com/ajhelbig/graph_box.git