Jim Palomo

jimppalomo@gmail.com | (847) 345-2180
jimpalomo@github.io | linkedin.com/in/jim-palomo/ | github.com/jimpalomo

Education

Chicago, IL **University of Illinois at Chicago (UIC)** **May 2022**

- Bachelor of Science in Computer Engineering. GPA: 3.79
- Undergraduate Coursework: Data Structures, Embedded Systems, Logic Design, Circuit Analysis, Computer Architecture, Discrete and Continuous Signals and Systems, Digital Systems Design.
- Filipinos in Alliance: Allstate Hot Chocolate Run 5k/15k; (Self) 34th Annual Hunger Walk

Skills

- **Languages:** C, C++, Python, ARM Assembly, MIPS Assembly
- **Software/Tools:** Linux, Git, SSH, Catch Framework, Valgrind, MARS

Internships

Computer Architecture Intern **University of Illinois at Chicago (UIC)** **Jun - Aug 2019**

- Developed on an open-source simulation platform for computer system architecture called gem5
- Established connections between different CPU chip-sets such as ARM & x86 with memory controllers, caches, and interconnects
- Communicated with an engineering professor and a Ph.D. student on project deadlines

Projects

Cache Simulator - Python **Nov 2020**

- Led a team to simulate four different types of cache configurations: Simple (1 block, block size 64 B), Direct Map (4 sets, block size 16 B), Fully Associative (4-way, block size 8 B), Set Associative (2-way 4-set, block size 8 B)
- The simulator in Python takes in hexadecimal machine code inputs from MIPS and determines program mode according to user input
- Programmed the simulator to display detailed step-by-step cache information: memory breakdown, LRU (least recently used) specifics, hit or miss results

8-Bit CPU Design - Python, MIPS, CircuitVerse **Oct - Nov 2020**

- Designed a custom ASIC-style ISA with a team featuring nine unique 8-bit instructions with binary width determination instruction
- Developed the following components: 64-byte ROM instruction memory, ALU schematic, Control unit logic, and CPU Datapath
- Led the software development portion that simulated the ISA in Python and MIPS which takes in binary machine code through a text file that outputs expected results

Back-End Navigation - C++, XML, Valgrind, GNU Make **Apr - May 2020**

- Designed an application that allows the user to observe the back-end functions of GPS oriented maps through loading a map, building a graph, and finding the shortest path between two separate locations
- Integrated Dijstrakia's algorithm to find the shortest path among two points
- Implemented map data from openstreetmap.org of UIC's East Campus containing over 18,000 nodes which covers 34 buildings
- Debugged using VSCode, used Valgrind to check for memory leaks, (data structures) Map, Graph, Stack, Vector, Set, Queue

DIVVY Data Hashing - C++, CSV, Valgrind, GNU Make **Apr 2020**

- Developed an application that hashes station and trip data from DIVVY bike-sharing company
- Created a hashmap with separate hash functions for over 1500 trips and 580 bike IDs
- Added multiple commands: search by station id, abbreviation, trip id, bike id, nearby stations, and similar trips
- Debugged using VSCode, used Valgrind to check for memory leaks, (data structures) Vector, Hashmap