Below, you can find the details of most (if not all) of my projects. Thanks for visiting!

Resume // Email // LinkedIn

All My Past & Current Projects

Personal Projects

Running DOOM (and Snake) on the DE1-SoC Source code:

- FPGA modules for DOOM and Snake, and C++ source code for Snake
- C DOOM source code Summary:
- Modified the original DOOM source code (in C) to be compilable on the DE1-SoC's ARM Cortex A9 processor
- Loaded a Linux kernel onto the DE1-SoC and constructed a loadable kernel module to reserve memory space on the CPU for shared access w/ FPGA
- Developed SystemVerilog FPGA modules to carry out hardware-accelerated graphics rendering functions via CPU shared memory
- Added functionality to custom-ported DOOM code to take advantage of these hardware-accelerated FPGA modules
- Integrated FPGA system with Intel Quartus' Platform Designer for modularized, swappable design
- Debugged the system at integration time with GNU GDB (CPU side) and Intel Signal Tap logic analyzer (FPGA side)

Python Typing Game with a GUI, Scores, and Live Feedback

- Built a live-responding graphical typing game to expand Python knowledge and exercise programming best practices
- Leveraged TkInter library knowledge to construct an efficient, responsive GUI for the typing game
- Designed optimized algorithms to compute and display user statistics in real time without consuming significant system resources
- Employed unit tests, virtual environments, and Docstrings to ensure code correctness, readability, and changeability

FORM Swim Projects

At form, I worked with the Software QA team to continuously test & validate FORM's embedded firmware and mobile software. More interestingly, I proposed and led the development of an internal tool -- QA Terminal.

Sorry, the source code for this project is not available.

QA Terminal

- Led the development of this Python-based GUI application to combine 10+ critical testing functions and accelerate testing
- Demonstrated a passion for learning by independently mastering 4 new BLE, GUI, and cloud libraries to innovate new functionality
- Developed a proprietary Bluetooth Low-Energy (BLE) Python API for FORM devices to facilitate remote control & file transfer
- Leveraged AWS services and embedded C++ utilities to enable rapid firmware deployment and verification
- Optimized and increased USB-Serial data processing speed by 80% (compared to previous tools) to enable faster insights

UBC Solar Projects

Github CI/CD Pipeline Automations

- Eliminated manual file update work by developing automation tools with Git and GitHub Actions
- Automatically managed file version descriptions by creating and executing Python scripts in a Linux environment
- Enhanced and formalized build process for embedded C source files by using PlatformIO build tools
- Ensured code quality and correctness by communicating decisions, considerations, and drawbacks via code-reviews

Ventilation Intakes for a Battery Pack

- Led the design of custom 3D-printed ventilation intakes for the Solar car battery to ensure sufficient cooling airflow through the battery.
- Applied the engineering design process to settle on one design that conforms to the tight geometry inside the car while moving a sufficient amount of air.
- Continuously presented design features, considerations, and drawbacks to team leads and executives during “design reviews”.

Waterproof Cover for a Vital Circuit

- Led the design of a custom 3D-printed water-tight enclosure to cover a vulnerable & critical circuit board on the Solar car battery.
- Applied problem-solving skills by resolving difficult design elements demanded by electronics team.
- Efficiently communicated design choices, feasibility, and performance to team executives.

UBC Course Projects

(Notes: To comply with UBC course policy, I have hidden the source code for my course projects. However, they are available upon request!)

Hardware Acceleration on an FPGA Embedded Processor — CPEN311

- Utilized Intel FPGA IP to construct an embedded Nios-II processor that can execute C programs on an Altera FPGA chip
- Engineered hardware SystemVerilog component to perform Direct Memory Access to on- and off-chip RAM
- Accelerated matrix-multiplication speed by implementing another hardware component to compute without CPU overhead
- Validated functionality of embedded processor by RTL simulation with Modelsim and then synthesizing it onto the DE1-SoC FPGA
- Designed C program to manage matrix multiplication on existing neural network weights to interpret binary images, and ntegrated hardware components with embedded processor to enable starting & stopping hardware accelerator in C code

Unix-like Memory System -- CPEN212

- Implemented a model memory management system in C++ to run on a Linux machine
- Engineered heap allocation algorithm to allocate, free, reallocate, coalesce manually-managed memory
- Engineered virtual memory system to provide virtual addresses to users and optimize RAM usage by offloading to disk
- Utilized GNU Make and GDB to compile and debug the memory system, ensuring 90%+ code coverage

Real-Time Client-Server IoT — CPEN221

- Constructed a client-server model to allow socket communication over a network of simulated IoT entities
- Established an abstract model of client-server interactions in UML to provide a starting point for implementation
- Developed multithreaded server implementation to maximize service-handling capability for the local machine
- Implemented data-processing algorithm with real-time guarantees to ensure high “quality of service” metrics for clients
- Increased server capability further by offloading excessive server load onto AWS Lambda

Reduced Instruction-Set Computer (RISC) — CPEN211

- Programmed a Turing-complete RISC CPU in SystemVerilog by applying embedded software, digital logic, and hardware principles
- Reduced debugging time by 70% through adopting the pair-programming technique with my partner
- Enabled adoption of a custom 16-bit instruction-set architecture by deploying a software finite state machine controller
- Executed our program embedded onto an Intel SoC FPGA to carry out high-coverage unit & integration testing suites
- Increased CPU performance by 180% relative to others and ranked top 10 in class by optimizing RISC execution steps

Computations & Simulations with Graphs — CPEN221

- Practiced Java object-oriented programming concepts to create two robust implementations of mathematical graphs
- Applied leadership skills by setting team meetings and splitting tasks to teammates

Arduino-Driven Autonomous Claw, APSC101

- Constructed an embedded Arduino program to autonomously retrieve objects with sensors, actuators, and switches
- Achieved a correct, well-rounded solution by utilizing C programming skills and the formal engineering design process
- Scored 100% at the class competition by testing and fine-tuning the embedded sensing algorithms

Simon Says on Arduino, APSC160

- Implemented embedded C program on an Arduino to play a game of Simon Says with LEDs, switches, and buttons
- Accelerated coding process and eliminated debugging time by creating program flowcharts