

Mahmoud Abumandour

+ 1 (788) 320-8958 | British Columbia, Canada | mahmoud_abumandour@sfu.ca | [LinkedIn](#) | [GitHub](#)

EDUCATION

Simon Fraser University **Sep 2024 – Present**
Ph.D. in Computer Science *BC, Canada*

Simon Fraser University **Sep 2022 – Aug 2024**
Master of Science in Computer Science (GPA: 4.0) *BC, Canada*
Thesis: **Resilient Neural Networks at the Edge: Uncovering and Mitigating Bit-Flip Vulnerabilities**

Mansoura University **Sep 2017 – Jul 2022**
Bachelor of Computer and Communication Engineering (GPA: 3.96, ranked first) *Mansoura, Egypt*

PROFESSIONAL EXPERIENCE

Simon Fraser University **BC, Canada**
Graduate Research Assistant **Sep 2022 – Present**

- Devised the first semi-black box bit flip attack against quantized DNNs (degrades DenseNet121 from 85% to 10% with only 14 bit flips out of 58 million)
- Implemented a software redundancy defense in TensorFlow, reducing attack success probability from 99% with 32 bit-flips to 0.9% while incurring less than 1% performance overhead on average

Teaching Assistant **Jan 2023 – Present**

- Conduct tutorials and lab sessions, grading assignments and exams, and providing support during office hours
- Courses: Intro to computer systems, Principles of compiler design, Computer architecture

Intel Corporation **Santa Clara, CA (Remote)**
Graduate CPU Architecture Intern **Jan 2024 – May 2024**

- Researched, modelled, and assessed CPU front-end features, including instruction prefetching and caching
- Performed in-depth workload analysis to categorize based on instruction icache footprint and branch behavior
- Conducted comparative studies between functional and cycle-accurate simulators and identified sources of miscorrelation

Google Summer of Code (RTEMS) **Remote**
Student Developer **May 2022 – Sep 2022**

- Achieved 8x speedup over the previous release notes generator by using a multi-threaded architecture
- Automated manual Markdown and RST to PDF generation after fetching release data from RTEMS bug tracker

Master Micro **Cairo, Egypt**
Software Engineering Intern **Oct 2021 – Feb 2022**

- Designed a database format for the main design lookup table file, reducing average query time by 50% over a binary format
- Participated in code review and testing. Increased the testing coverage of the data querying subsystem by 10%

Google Summer of Code (QEMU) **May 2021 – Aug 2021**
Student Developer

- Implemented multi-core, multi-level cache performance emulation of user-space and full-system workloads
- Improved the system call tracing by making its reports more script-friendly for post-processing

PROJECTS

- [C Compiler in Rust & LLVM](#): Compiled a subset of C and supported two backends: LLVM and native x86_64
- [Fuzzing with RISC-V Emulation](#): Developed a RISC-V 64-bit emulator for userspace fuzzing. Increased test generation throughput by over 16x (linearly with available resources) over single-core performance
- [Database Engine \(RheaDB\)](#): Implemented a disk-oriented DBMS with SQL support, in-memory pool caching, B+ Tree indexing, and JDBC driver
- [AES Encryption Core](#): Designed a low-power AES encryption core for FPGA. Reduced area and power consumption by more than 80% over a high-throughput pipelined design
- [Hyperthreaded, Software-Interlocked MIPS Processor](#): A multi-threaded five-stage pipelined MIPS core for FPGA and a custom assembler with software interlocking, achieving 5x more throughput over single-threaded execution

SKILLS

Programming Languages: C++, C, x86 Assembly, Rust, Python, Bash Scripting, Java

Tools: Gem5, TensorFlow(lite/micro), LLVM, Git, Docker, Valgrind, LLVM

Platforms: Linux, QEMU, FPGA, ARM Cortex M4, AVR

Hardware Design Tools: Xilinx Vivado, ModelSim, SystemVerilog, VHDL

OPEN-SOURCE CONTRIBUTIONS

- **SerenityOS**: Defined a global OS versioning API. Increased user-space utilities POSIX compliance. Improved the SerenityOS DBMS SQL compliance by supporting INSERTs with multiple tuples and table-description statements
- **QEMU**: Modernized the usage of locking and memory allocation APIs by using scope-based locks and automatically freed allocations. Redefined plugins' command line syntax adhering to modern QEMU CLI syntax