Mahmoud Abumandour

+ 1 (788) 320-8958 |BC, Canada | mahmoud_abumandour@sfu.ca | Website | Linkedin | GitHub

EDUCATION

Simon Fraser University **Sep 2024 – Present**

Ph.D. in Computer Science (GPA: 4.0) *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 over 180 students) *Mansoura, Egypt*

EXPERIENCE

Simon Fraser University **BC, Canada**

Graduate Research Assistant ***Sep 2022 – Present***

- Devised the first semi-black box Bit Flip Attack (BFA) against quantized DNNs (destroys Llama 3 2B with 25 bit flips with over 80% probability)
- Explored hardware/software defenses & Implemented Chimera, a memory allocator that randomizes model data memory layout, eliminating BFA against DNNs by preventing precise data positioning on vulnerable DRAM pages.

Simon Fraser University **BC, Canada**

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)**

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 cache 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 release data fetching from RTEMS bug tracker and Markdown to RST & PDF generation

Master Micro **Cairo, Egypt**

Software Engineering Intern ***Oct 2021 – Feb 2022***

- Designed a range format for an EDA design lookup table file, reducing query time by 50% over a binary format

Google Summer of Code (QEMU)

Student Developer ***May 2021 – Aug 2021***

- 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 functional 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 RISC Processor**: A multi-threaded five-stage pipelined RISC 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, LLVM, Git, Docker, Valgrind, perf, PyTorch, Tensorflow
Platforms: Linux, QEMU, FPGA, ARM Cortex M4, Raspberry Pi
Hardware Design Tools: Xilinx Vivado, ModelSim, SystemVerilog, VHDL

OPEN-SOURCE CONTRIBUTIONS

- **RISC-V Newlib:** Profiled and optimized the newlib standard library implementation for RISC-V. Used QEMU, Spike, Gem5, and a RISC-V Raspberry Pi board for benchmarking.
- **SerenityOS**: Defined a global OS versioning API. Increased user-space utilities POSIX compliance. Improved the SerenityOS DBMS SQL support
- **QEMU**: Modernized the usage of locking and memory allocation APIs by using scope-based locks and automatically freed allocations. Redefined plugins' configuration syntax adhering to modern QEMU CLI syntax