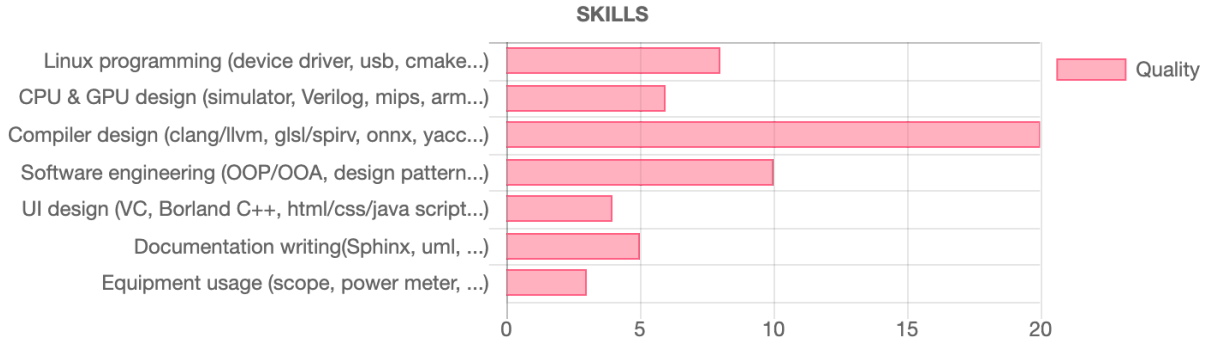
I am a compiler developer with solid experience in LLVM CPU and GPU backends, the LLD linker, NPU/ONNX, C++, OpenGL/GLSL, simulators, and more. I enjoy working on compilers and related technologies.

RESUME

QUALIFICATION

Over 20 years of experience in C/C++ programming, with 13 years focused on compiler



MY OPEN SOURCE PROJECT

I'm proud that my work is featured in the official LLVM documentation under http://llvm.org/docs/tutorial/#external-tutorials.

Tutorial: Create an LLVM Backend Compiler http://jonathan2251.github.io/lbd/index.html

Tutorial: Create an LLVM Backend Toolchain http://jonathan2251.github.io/lbt/index.html

The Concept of a GPU Compiler http://jonathan2251.github.io/lbd/gpu.html

EDUCATION

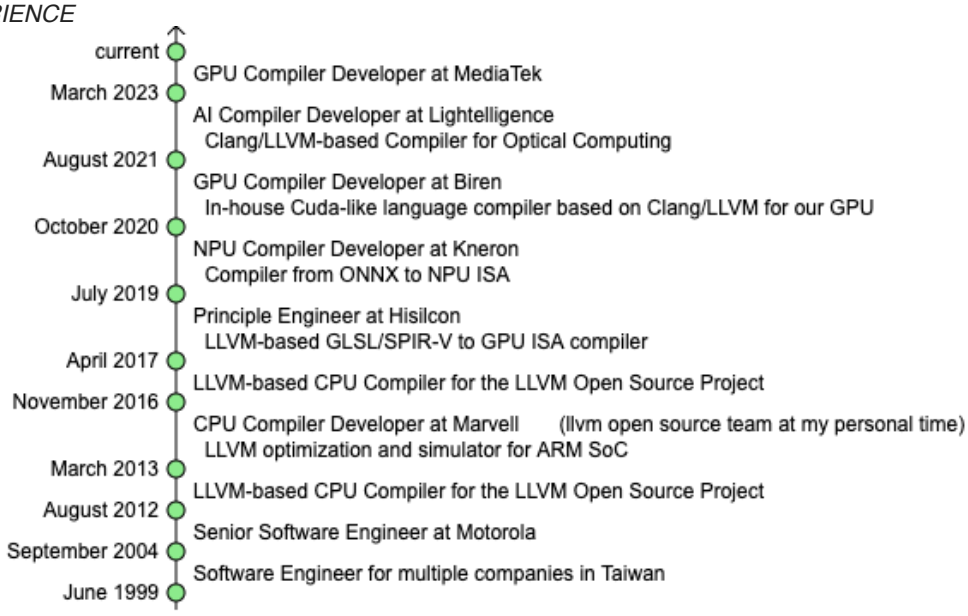
Master's Degree, Information Science, National Taiwan Normal University (國立台灣師範大學), Taipei — June 1999

Bachelor's Degree, Industrial Engineering, National Taiwan University of Science and Technology (國立台灣科技大學), Taipei — June 1994

LICENSE

National Senior Technician Certificate in Information Technology (國家高考資訊技師), Taiwan — 1995

EXPERIENCE



Master's Thesis

The Researches of Column Sort and Related Problems

Conference Paper: Search for "行排列法簡化步驟之研究" on the above link.

PhD Study Proposal

The Researches of Sorting Network and Related Algorithm

OTHER WORK

Took a course in image processing and developed Jpeg decoder

Web and javascript: As my resume and my personal web site

Graphivz: as some graph diagrams used in this CV. Source code: mywork_1.gv and study_and_apply.gv

ACHIEVEMENT

Lightelligence

Developed a RISC-V backend compiler for Lightelligence's optical NPU:

1. Built full RISC-V toolchain (GCC, LLVM, QEMU/Gem5); evaluated vendors and costs.
2. Led Aurora software development; implemented compiler backend.
3. Created TaskGraph in C++ compiler with Runtime integration for DL graph support.

Biren

Built an in-house Cuda-like compiler using Clang/LLVM for our GPU:

1. Implemented GPU codegen for tensor ops and usharpid.
2. Optimized performance and resolved bugs.
3. Proposed parallelism via async{...}.

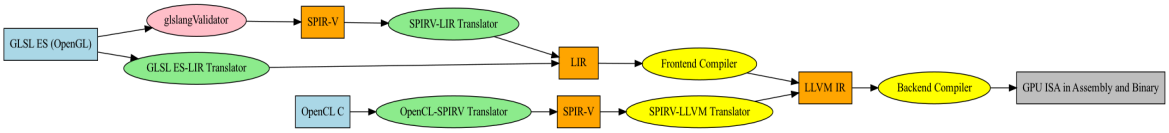
Kneron

NPU Compiler Developer:

1. Rebuilt the top two layers to support a unified graph across NPUs.
2. Added input support for encrypted ONNX and config files.
3. Validated MLIR integration solutions.

Hisilcon

Scope of GPU Compiler Work:



Compared our GPU compiler with the ARM-licensed version (yellow nodes); ~20% of frontend and 50% of backend modified, based on lines of code.

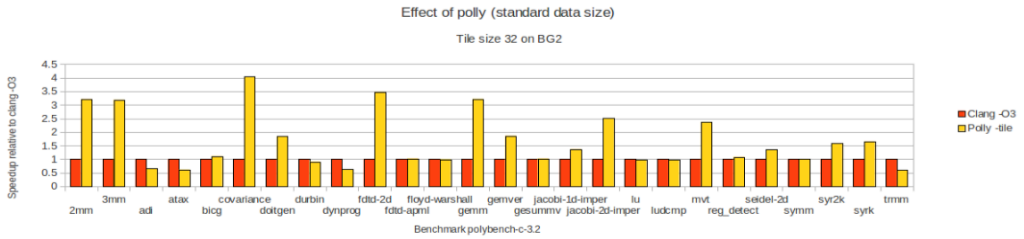
My Contributions:

1. Implemented ~80% of texture-related frontend/LLVM backend per OpenGL ES 3.2; wrote documentation.
2. Supported and reviewed the remaining 20% with team and texture lead.
3. Developed Prefetch-Sample optimization for driver-level texture sampling.
4. Added compiler support for Vulkan load/store ops with RGBA fixed-point formats (32, 16, 11, 10, 2 bits) and NaN/Inf handling; documented feature.

Marvell

LLVM Optimization and Simulator for ARM SoC:

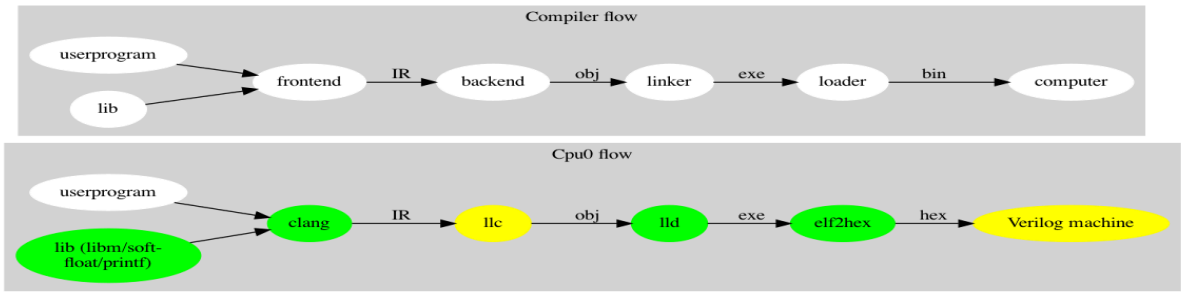
1. Built a semi-automated system for benchmarking and reporting on the GCC toolchain.
2. Introduced Polly (loop optimization) and polyhedral model to improve LLVM/GCC at Marvell.



3. Developed a co-simulator for Marvell's 64-bit ARM CPUs, including a DSL that reduced C++ verification code.
4. Migrated CSim from Make to CMake for a simpler, cross-platform build system.

LLVM Open Source Project

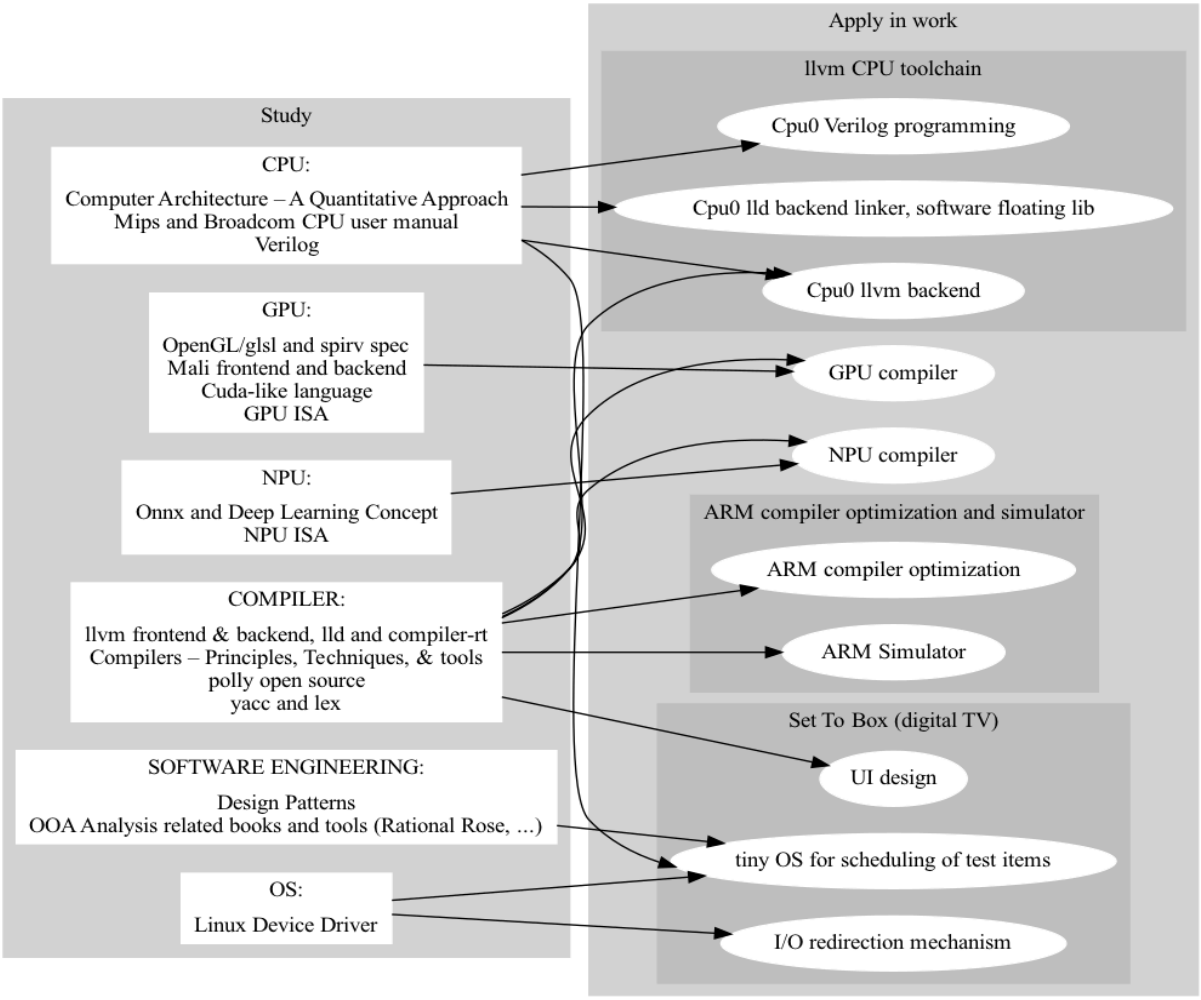
The lower half of diagram below illustrates the workflow of my LLVM backend. The yellow and green sections represent components I implemented, as documented in my tutorials.



Motorola

Developed the software framework for Set-Top Box systems.

Learning Beyond School and Applying It at Work



References

Recommendation Letter from Former Manager: https://jonathan2251.github.io/ws/en/RL_Marvell.pdf