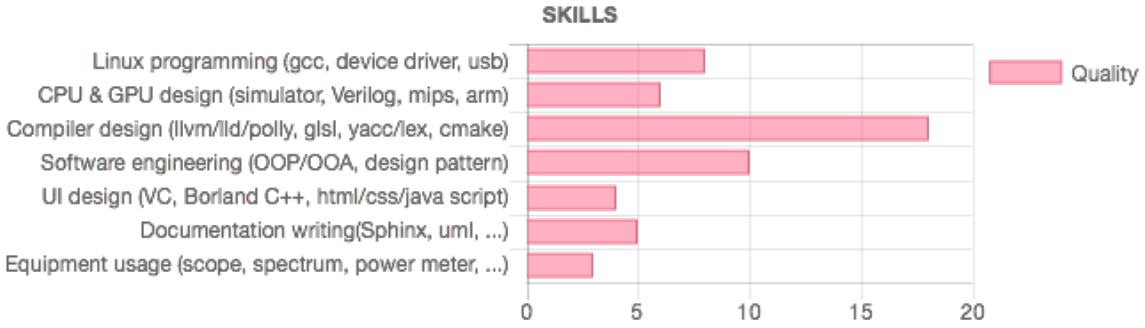
Chung-Shu Chen 陳鍾樞

I am a compiler developer with good experience in llvm cpu and gpu backend, lld linker, npu/onnx, c++, OpenGl/glsl and simulator, ..., and enjoy with compiler.

CV (DETAIL RESUME)

QUALIFICATION

Over 20 years experience in c/c++ programming, 7 years compiler toolchain related experience and research in parallel processing for master degree.



MY OPEN SOURCE PROJECT

I am proud of my work is accepted by LLVM documentation, appears at 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 GPU compiler http://jonathan2251.github.io/lbd/gpu.html

EDUCATION

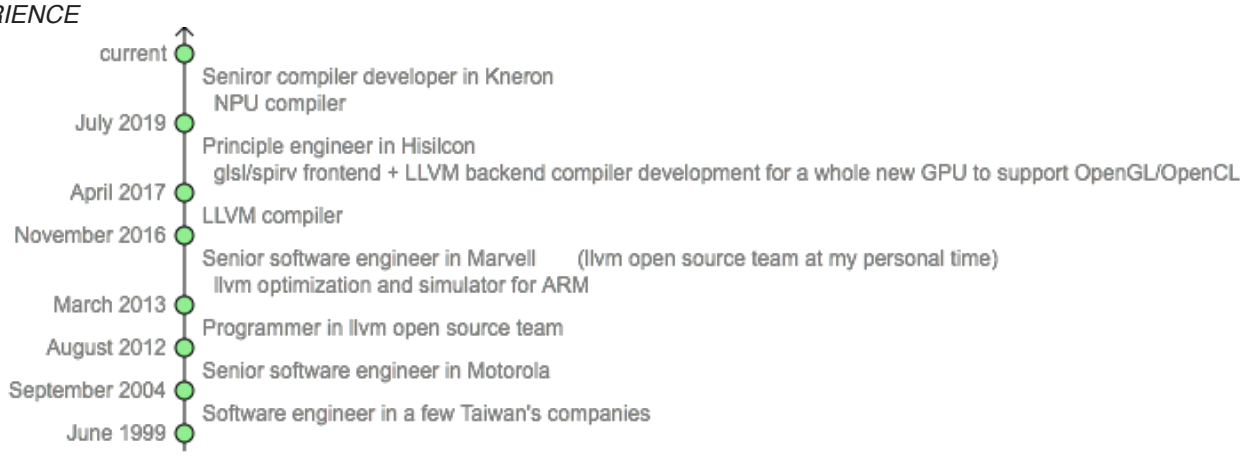
1997-1999 Master, June 1999, National Taiwan Normal University (國立台灣師範大學), Taipei, Major: Information Science.

1991-1994 B.S., June 1994, National Taiwan Technology University of Science and Technology (國立台灣科技大學), Taipei, Major: Industry Engineer.

LICENSE

Taiwan National Computer Engineer license, 1995 高考資訊技師及格.

EXPERIENCE



September 2004 - June 1999:

Proton 2014/3 – 2014/9 Manager Digital TV programming, Abocom 2013/6 – 2014/3, Senior Engineer 802.11b programming,

DBTEL 2011/11 – 2013/6 Engineer DECT wireless phone programming, Symmetry 2011/2 – 2011/11 Engineer, SGSN and GGSN for GPRS&3G programming,

Cando 2010/7 – 2011/2 Engineer CAM programming, Spirox 2009/12 – 2010/7, Engineer CAM programming, Intech 2009/6 – 2009/12 Engineer CAM programming

THESIS OF MASTER DEGREE

The Researches of Column Sort and Related Problems

PROPOSAL OF PHD STUDY

The Researches of Sorting Network and Related Algorithm

OTHER WORK

Take course "Image processing" and program: 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

Hisilcon

Implement compiler (fontend + llvm backend) for 80% of texture related API and optimization by myself and document writing.

Code generation for load/store with RGBA fixed floating point format of vulkan (32, 16, 11, 10 and 2 bits; NaN Infinity transfer) alone and document writing.

To support an our new designed GPU for cell phone, ported from ARM. 20% of frontend is changed, 50% of backend is changed.

Marvell

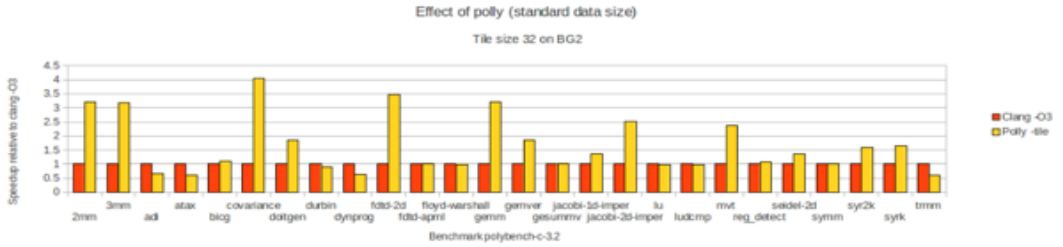
Implement semi-auto software system of running benchmark and generating report for gcc toolchain.

Purpose is to make sure the code change of gcc has no side effect in optimization (expain in flow chart)

Implemented bash script which can compare any two version of gcc toolchain and generate report. Similar as above.

Introduce polly for Marvell llvm toolchain optimization. Polly is a software for loop optimization.

The advantage is that it might be 500% speed up in some numerical application program, such as matrix multiplication, or other kind of matrix operation. Sadly, Marvell's CPUs are not used in numerical applications. But maybe it will have for 64 bits ARM in future.



Implement co-simulator(our name:Csim) for a few Marvell's ARM based cpu.

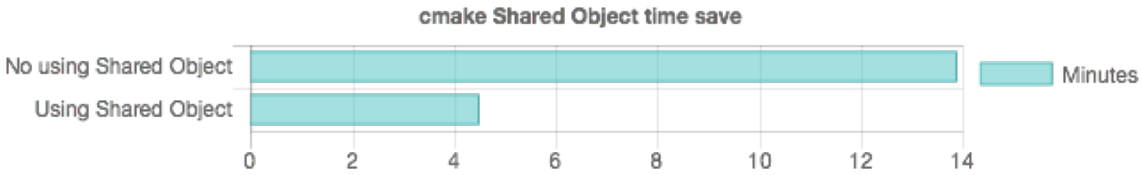
Apply compiler code generation idea to propose a solution and Csim team adopted it.

ARM spec → script → generate c++ .h .cpp

Mouse hover the above (in html) for explanation, obviously the script is simple, match ARM spec and easy to read more then the c++ program.

Complete cmake and adjust python programs to build Csim with cmake. It saves effort in maintaining both make and windows project file. In addition, cmake also support iMac user.

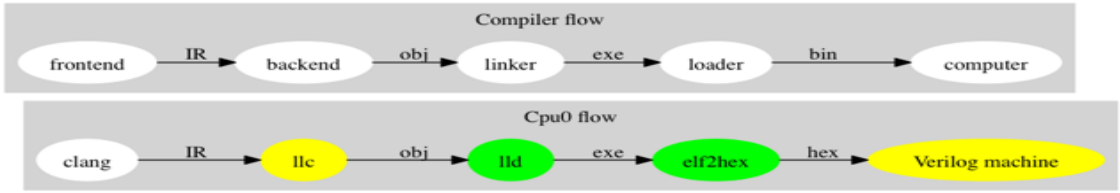
The version of cmake in Csim server is 2.6.x, however, the cmake 2.8.8 provides shared object to save the unnecessary compilation.



Report performance comparing between cmake and ninja for other manager's concern. The result is "no difference"

LLVM open source project

The upper half of the following figure is the work flow that software package of a computer program be generated and executed. IR stands for Intermediate Representation. The lower half is the llvm's work flow. The yellow and green parts are implemented in my books (listed as my open source project above), Tutorial: Create an LLVM Backend compiler and Tutorial: Create an LLVM Backend toolchain respectively.

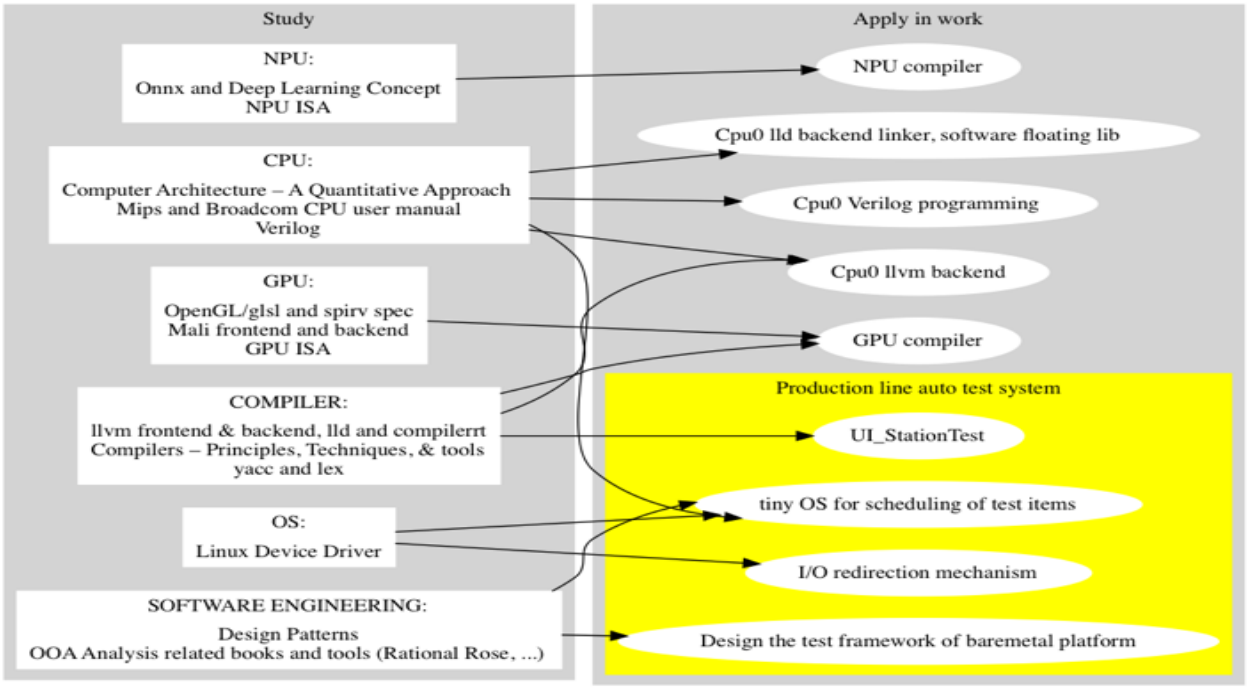


Mortorola

Design QIP7kP1 and P2 board level production line testing code. Over 5 millions units of QIP7k model of Set Top Box are tested with this code on production line and shipping to market. Other major products DCX33, DCX34 and DCX35 (over 10 millions of box) are ported from QIP7kP1 & QIP7kP2.

Combine mips cpu and software engineer knowledge, lead and cowork with other four members to create a frame work for board level testing code as manager request.

Learning after school & applying in work



"Tiny OS": implement test items scheduling mechanism through timer interrupt routing with c++ "observe pattern" and mips assembly language to solve the halting problem in some testing program.

After this implementation, the code has run more stable and fast since the testing driver code will hang on especially in new hardware and software of new model of set top box.

Implement the I/O redirection mechanism to make the new version of code porting task easier in Motorola.

References

[My former manager's recommendation letter: https://jonathan2251.github.io/ws/en/RL_Marvell.pdf](https://jonathan2251.github.io/ws/en/RL_Marvell.pdf)