



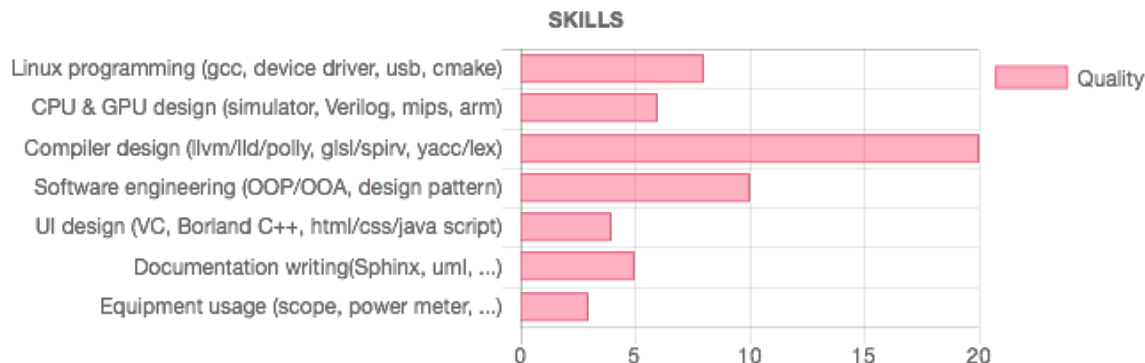
## 陳鍾樞

我是位有經驗的compiler開發者，開發過llvm cpu and gpu backend, lld linker, npu/onnx, c++, OpenGL/glsl compiler與simulator，...，對寫compiler感到快樂。

## 履歷

### 資格

二十年c/c++軟體開發經驗，九年編譯器相關工具開發經驗，碩士時研究平行處理。



### 我的開法原始碼專案

很高興我的作品已被LLVM接受，出現在 <http://llvm.org/docs/tutorial/#external-tutorials>

如何建立LLVM後端編譯器 <http://jonathan2251.github.io/lbd/index.html>

如何建立LLVM後端系統工具 <http://jonathan2251.github.io/lbt/index.html>

GPU編譯器概念 <http://jonathan2251.github.io/lbd/gpu.html>

### 學歷

1997-1999 碩士班，六月 1999，國立台灣師範大學，台北，主修：資訊科學。

1991-1994 學士班，六月 1994，國立台灣科技大學，台北，主修：工業工程。

### 證照

1995年 高考資訊技師及格。

### 經驗



### 碩士論文

[The Researches of Column Sort and Related Problems](#)

論文期刊：上述鏈結網頁搜尋"行排列法簡化步驟之研究"

### 博士班研究計畫

[The Researches of Sorting Network and Related Algorithm](#)

### 其餘作品

修影像處理課程與撰寫：[Jpeg decoder程式](#)

**Graphviz:** 如此詳細履歷裡的一些圖學的圖。原始碼：[mywork\\_1.gv](#) and [study\\_and\\_apply\\_ch1.gv](#)

- Gpu CodGen for tensor instructions and usharpid handling.
- Gpu Optimatization and bug fix.
- CodGen for paralel proccessing of our Cude-like language `async{...}`.

- Re-implement top 2 layers of our npu compiler for our common graph data structure to handle onnx.
- Implement compiler input interface to support encryption-onnx format.
- Confirm solution for MLIR supporting.

GPU Just In Time Compiler Flow

```
graph LR; A([frontend(opengl, vulkan and opencl compiler)]) -- "llvm IR" --> B([backend]); B -- "obj" --> C([linker]);
```

The flowchart illustrates the GPU Just In Time Compiler Flow. It starts with a box labeled "frontend(opengl, vulkan and opencl compiler)", which outputs "llvm IR" to a box labeled "backend". The "backend" box then outputs "obj" to a final box labeled "linker".

獨立完成80% texture相關的API與optimization (frontend + llvm backend)與document撰寫。  
指導別的工程師完成其餘20% texture相關的API, [80 APIs totally here](#), 核對並與texture的架構leader一起合作。

完成Prefetch-Sample optiomization, 讓driver在載入glsl bin與執行sampling指令前就可驅動2D sampling指令。

獨立完成GPU對vulkan load/store RGBA 固定浮點格式(32, 16, 11, 10 and 2 bits; NaN Infinity)支援的指令生成與document撰寫。

Effect of polly (standard data size)

Tile size 32 on BG2

Speedup relative to clang-O3

Clang-O3  
Polly-32e

Benchmark polybench-c-3.2

Benchmark	Clang-O3	Polly-32e
2mm	1.0	3.0
3mm	1.0	3.0
atak	1.0	0.5
biog	1.0	0.5
covariance	1.0	4.0
datagen	1.0	1.5
durbin	1.0	0.5
dynprog	1.0	0.5
ftds-2d	1.0	3.4
ftds-apr1	1.0	1.0
floyd-warshall	1.0	1.0
gemm	1.0	3.0
gemver	1.0	1.5
jacobi-2d-imper	1.0	1.0
jacobi-2d-imper	1.0	1.5
lu	1.0	2.5
ludcmp	1.0	1.0
mvt	1.0	2.5
reg_detect	1.0	1.0
seidel-2d	1.0	1.0
symm	1.0	1.0
syr2k	1.0	1.0
syrk	1.0	1.0
trmm	1.0	0.5

Compiler flow

```

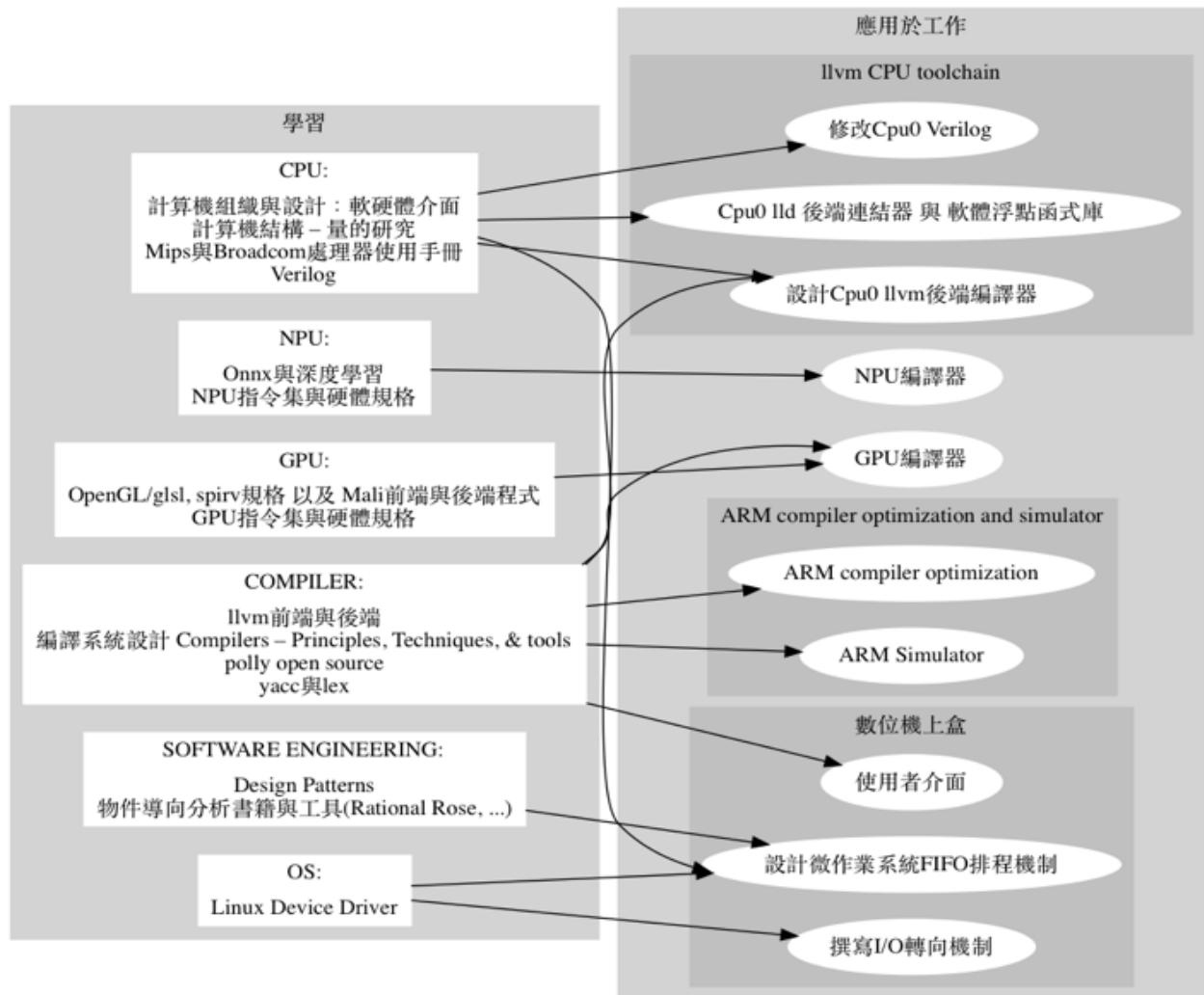
graph LR
    frontend -- IR --> backend
    backend -- obj --> linker
    linker -- exe --> loader
    loader -- bin --> computer
  
```

Cpu0 flow

```

graph LR
    clang -- IR --> llc
    llc -- obj --> lld
    lld -- exe --> elf2hex
    elf2hex -- hex --> Verilog_machine[Verilog machine]
  
```

## Page 2 of 3



## 推薦函

前主管推薦函: [https://jonathan2251.github.io/ws/ch1/RL\\_Marvell.pdf](https://jonathan2251.github.io/ws/ch1/RL_Marvell.pdf)