計組 期末 project demo

Q0(40%):

編譯環境

Q1(15%):

enable L3 跑 hello world 修改option, cache, cache\_config, xBar.py

* Cache:

一張含有 文字 的圖片

自動產生的描述

* Cache Config:

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

BaseCPU:

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

XBar:

一張含有 文字 的圖片

自動產生的描述

Option.py



Q2(15%):

full:

./build/X86/gem5.opt configs/example/se.py -c ./quicksort --cpu-type=TimingSimpleCPU --caches --l2cache --l3cache --l3\_assoc=1 --l1i\_size=32kB --l1d\_size=32kB --l2\_size=128kB --l3\_size=1MB --mem-type=NVMainMemory --nvmain-config=../NVmain/Config/PCM\_ISSCC\_2012\_4GB.config

2 way

./build/X86/gem5.opt configs/example/se.py -c ./quicksort --cpu-type=TimingSimpleCPU --caches --l2cache --l3cache --l3\_assoc=2 --l1i\_size=32kB --l1d\_size=32kB --l2\_size=128kB --l3\_size=1MB --mem-type=NVMainMemory --nvmain-config=../NVmain/Config/PCM\_ISSCC\_2012\_4GB.config

去 stats.txt 看 L3 overall cache miss rate

去 terminal 看 active\_energy

都要截圖。(共4張)

Two:

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

Full :

一張含有 文字, 報紙, 收據 的圖片

自動產生的描述一張含有 文字 的圖片

自動產生的描述

結論：full miss rate 比較低。

Q3(15%):

replacement\_policy = Param.BaseReplacementPolicy(LFURP(),"Replacement policy")

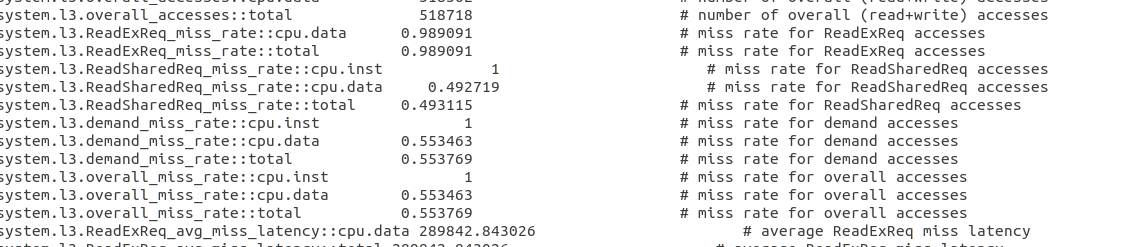
編譯 scons 執行 quicksort. 利用quick sort去比較預設(LRU)跟LFU的差別，比較miss rate。

* 執行指令：

./build/X86/gem5.opt configs/example/se.py -c ./quicksort --cpu-type=TimingSimpleCPU --caches --l2cache --l3cache --l1i\_size=32kB --l1d\_size=32kB --l2\_size=128kB --l3\_size=1MB --mem-type=NVMainMemory --nvmain-config=../NVmain/Config/PCM\_ISSCC\_2012\_4GB.config

* 預設(LRU):
* 修正程式碼：

一張含有 文字 的圖片

自動產生的描述

* 更改更改policy後的(LFU)
* 修正程式碼：

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

去 config.ini 看設置 L3 replacement policy.

一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述

更改前 更改後

一開始預設的LRU的miss rate會比較低(0.5多)，到後面會被更改到0.7多。

Q4(15%):

我不會qaq