

期末PROJECT



評分標準

- (Q1) GEM5 + NVMAIN BUILD-UP (40%)
- (Q2) Enable L3 last level cache in GEM5 + NVMAIN (15%) (看到 log 裡面有 L3 cache 的資訊)
- (Q3) Config last level cache to 2-way and full-way associative cache and test performance (15%)
 - 必須跑benchmark quicksort在 2-way 跟 full way (直接在 L3 cache implement，可以用 miss rate 判斷是否成功)
- (Q4) Modify last level cache policy based on frequency based replacement policy (15%)
- (Q5) Test the performance of write back and write through policy based on 4-way associative cache with isscc_pcm(15%)
 - 必須跑 benchmark multiply 在 write through 跟 write back (gem5 default 使用 write back，可以用 write request 的數量判斷 write through 是否成功)
- Bonus (10%)
 - Design last level cache policy to reduce the energy consumption of pcm_based main memory
 - Baseline:LRU

繳交方式

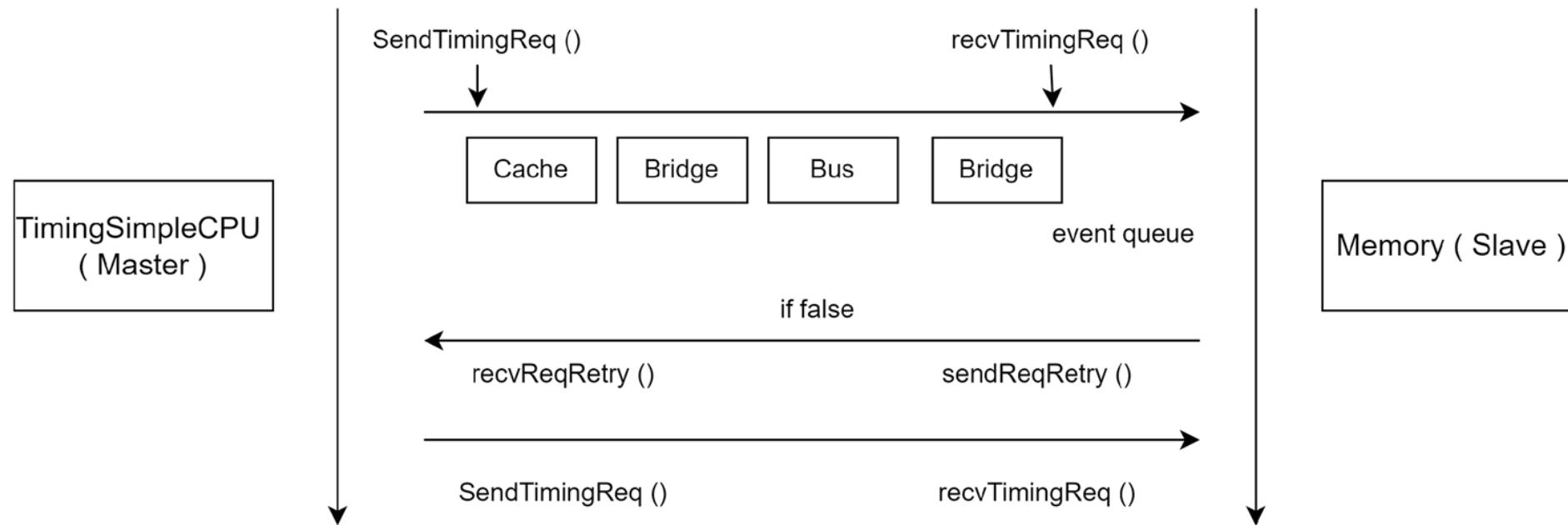
- (Q3) 直接在 I3 cache implement，並繳交兩份 log 檔案，一份是 2-way，一份是 full-way，需在 log 檔裡看到 I3 cache 的資訊
- (Q4) 繳交兩份 log 檔案，一份是原始 policy，一份是 frequency based policy
- (Q5) 繳交兩份 log 檔案，一份 write through，一份是 write back
- (bonus) 繳交兩份 log 檔案，一份是 baseline (hello)，一份是修改過後的方法
- log 檔案包含 m5out 中的 stat.txt 檔案，以及 nvmain 輸出的 log
- “修改過後”的 code 需要繳交，可以上傳 github，或是一起放在壓縮檔繳交，不需要額外的書面報告
- demo 時會額外問與 gem5，nvmain 的相關問題，且為了測試 code 是否能正常執行，會要求 demo 時編譯一次

常見問題

- 記憶體空間不足
- Ubuntu 版本不對，有問題的話建議用 18.04
- 執行 benchmark 的時候，因為 gem5 本身速度緩慢，會誤以為程式卡住，但其實他仍然有在執行
- 跑完程式後，可以從 m5out 的資料夾去查看 stat.txt
- 更改完 gem5 的檔案記得要編譯再重新執行
- gem5 有 debug flag 的功能

```
build/X86/gem5.opt --debug-flags=Exec configs/learning_gem5/part1/simple.py | head -n 50
```

Gem5 Memory system



NVMAIN+GEM5教學

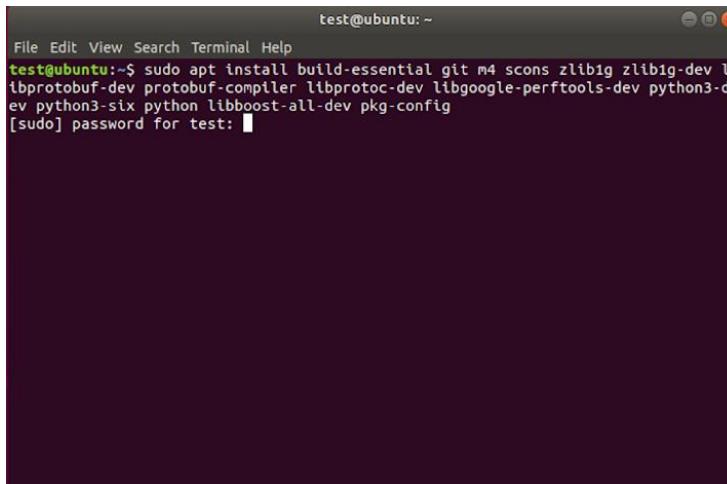


VIRTUAL MACHINE

- 推薦使用Ubuntu 18.04
- Ubuntu 20.04 g++太新 nvmain編譯會出問題
- Ubuntu 16.04 gem5不支援
- 其他硬體規格
 - 記憶體:4GB
 - 硬碟:20GB

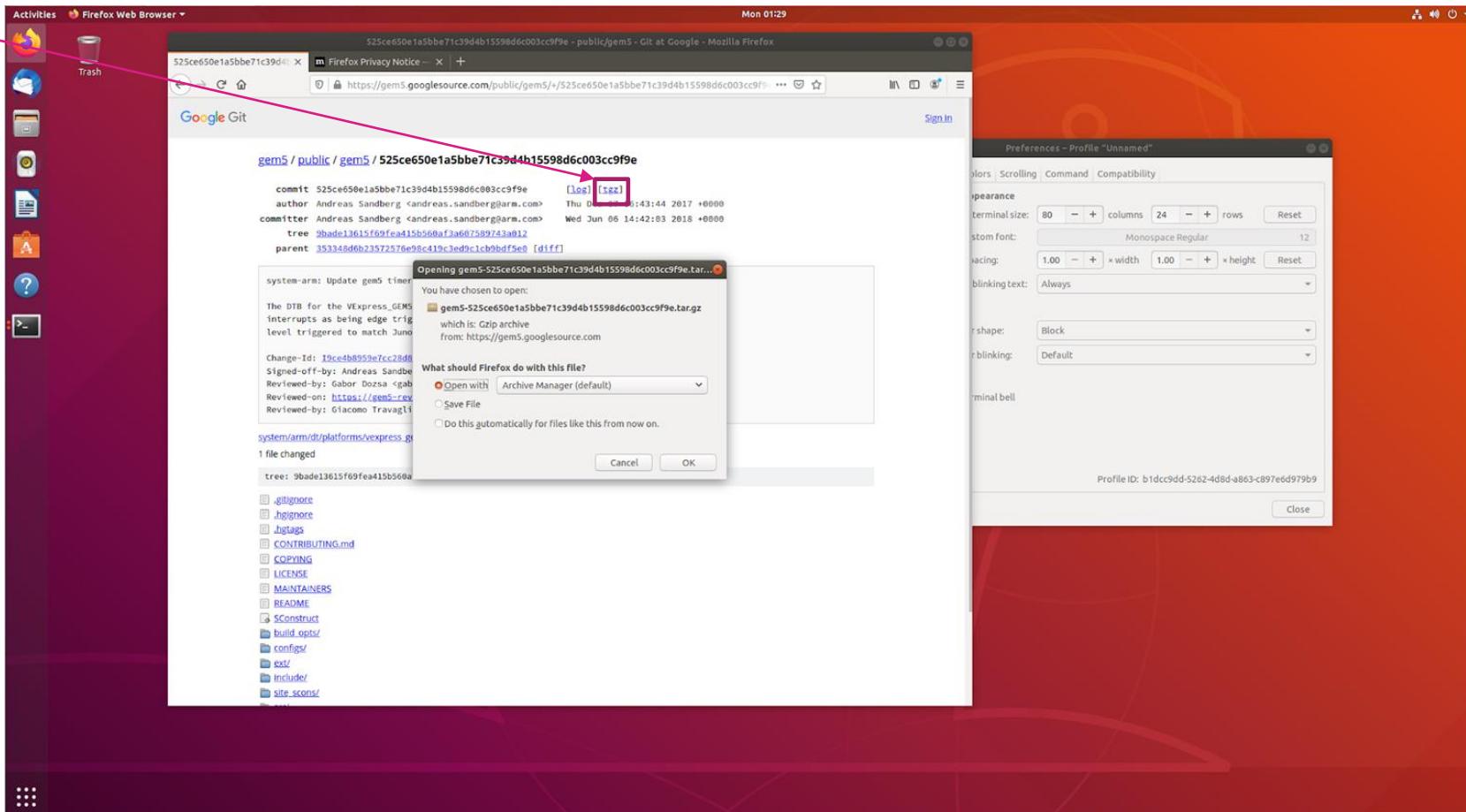
■ 安裝編譯工具

■ sudo apt install build-essential git m4 scons zlib1g zlib1g-dev libprotobuf-dev protobuf-compiler libprotoc-dev libgoogle-perftools-dev python3-dev python3-six python libboost-all-dev pkg-config

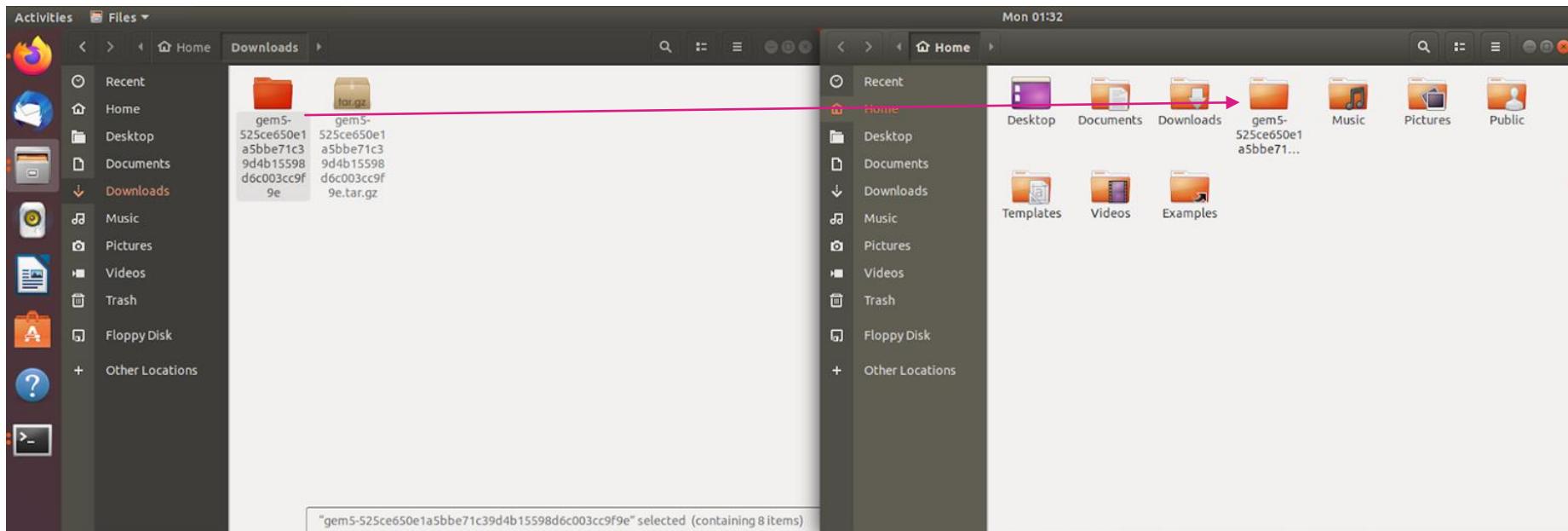


安裝GEM5

- <https://gem5.googlesource.com/public/gem5/+/525ce650e1a5bbe71c39d4b15598d6c003cc9f9e>
- 點tgz下載(不推薦下載最新版的GEM5，後面編譯會出問題)

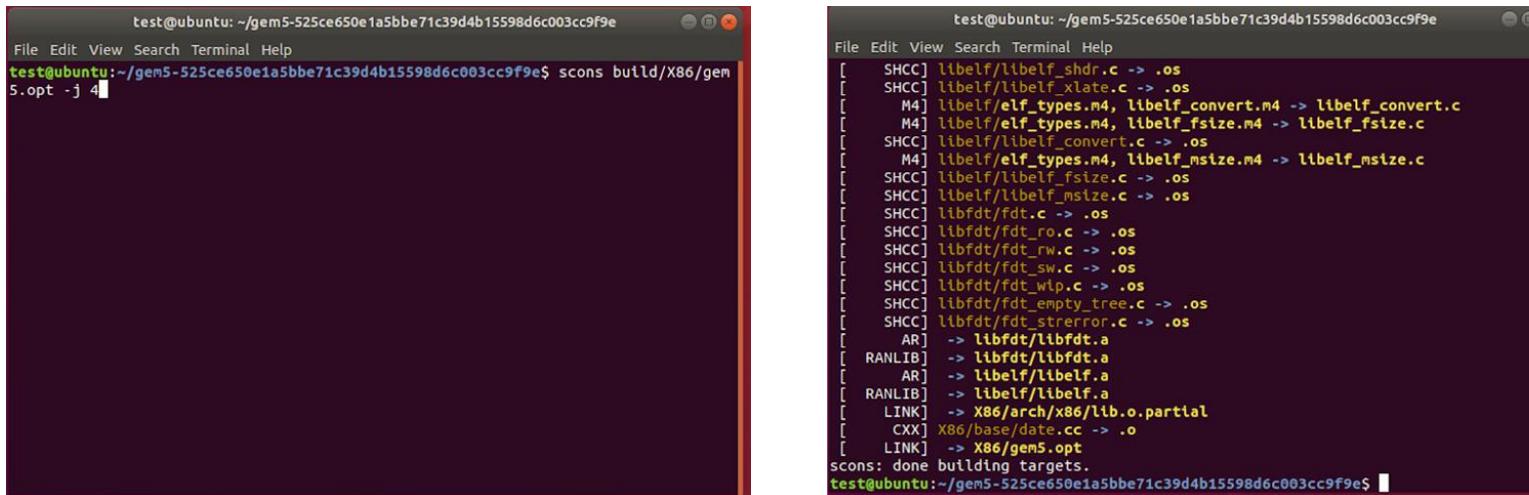


把解壓後的GEM5資料夾丟進HOME



編譯 GEM5

- 在GEM5目錄下scons build/X86/gem5.opt (後面可以加 –j num num為multithread數量來加速)
- 這安裝要大概20分鐘



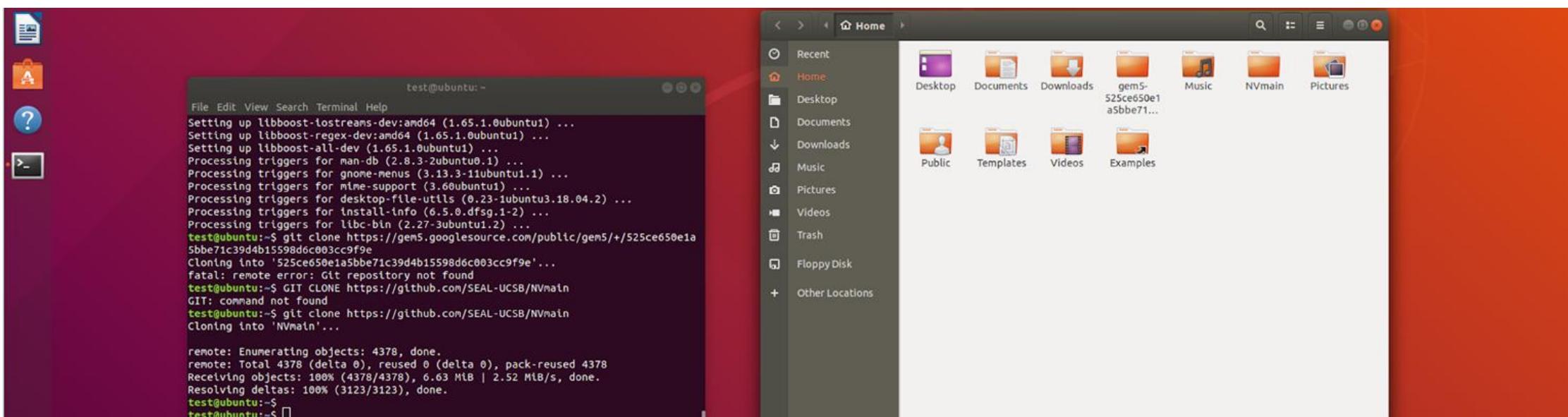
```
test@ubuntu: ~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e$ scons build/X86/gem5.opt -j 4
```

```
[ SHCC] libelf/libelf_shdr.c -> .os
[ SHCC] libelf/libelf_xlate.c -> .os
[ M4] libelf/elf_types.m4, libelf_convert.m4 -> libelf_convert.c
[ M4] libelf/elf_types.m4, libelf_fsize.m4 -> libelf_fsize.c
[ SHCC] libelf/libelf_convert.c -> .os
[ M4] libelf/elf_types.m4, libelf_msize.m4 -> libelf_msize.c
[ SHCC] libelf/libelf_fsize.c -> .os
[ SHCC] libelf/libelf_msize.c -> .os
[ SHCC] libfdt/fdt.c -> .os
[ SHCC] libfdt/fdt_ro.c -> .os
[ SHCC] libfdt/fdt_rw.c -> .os
[ SHCC] libfdt/fdt_sw.c -> .os
[ SHCC] libfdt/fdt_wip.c -> .os
[ SHCC] libfdt/fdt_empty_tree.c -> .os
[ SHCC] libfdt/fdt_strerror.c -> .os
[ AR] -> libfdt/libfdt.a
[ RANLIB] -> libfdt/libfdt.a
[ AR] -> libelf/libelf.a
[ RANLIB] -> libelf/libelf.a
[ LINK] -> X86/arch/x86/lib.o.partial
[ CXX] X86/base/date.cc -> .o
[ LINK] -> X86/gem5.opt
scons: done building targets.
test@ubuntu:~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e$
```

用GIT 安裝 NVMAIN

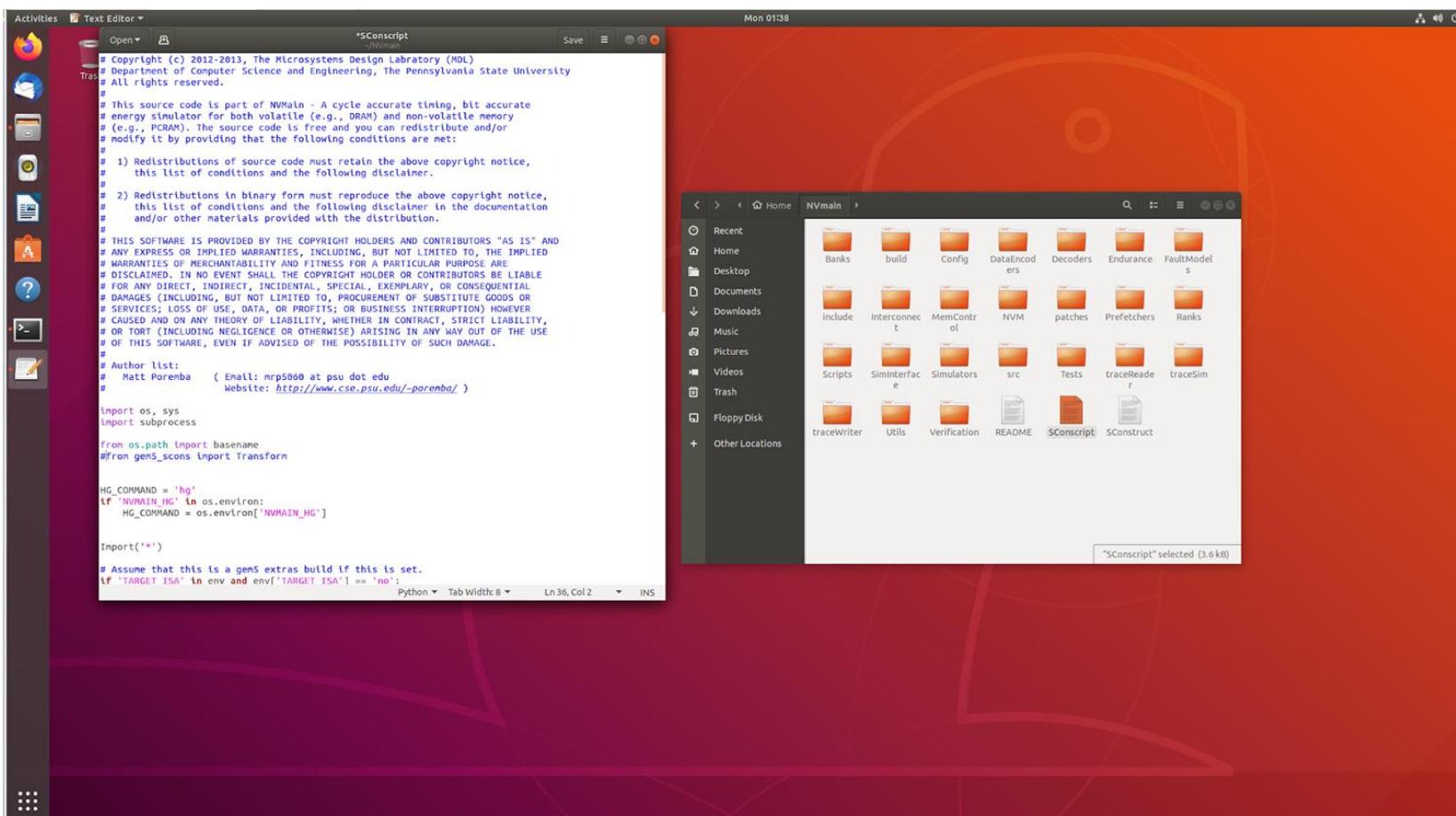
■ git clone <https://github.com/SEAL-UCSB/NVmain>

■ 把nvmain跟gem5放到同一個目錄下



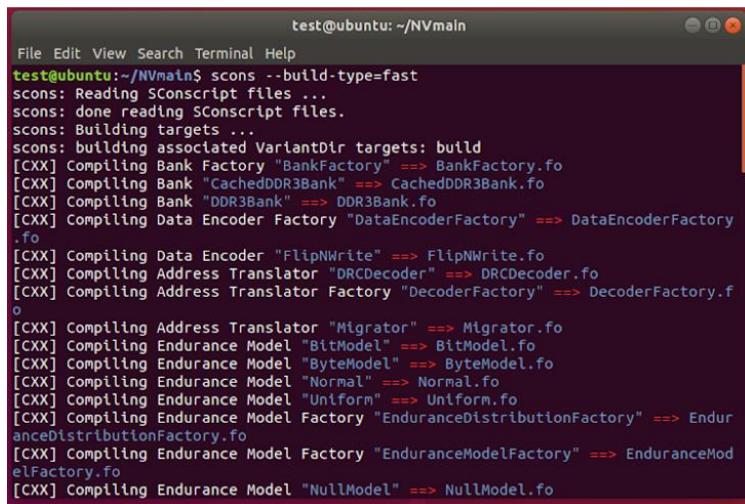
修改 SCONSCRIPT

■ 進入nvmain資料夾 點開 SConscript 把36行的from gem5_scons import Transform註解掉(記得要存檔)



編譯NVMAIN

■ 在NVmain目錄下 scons --build-type=fast



```
test@ubuntu:~/NVmain$ scons --build-type=fast
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
scons: building associated VariantDir targets: build
[CXX] Compiling Bank Factory "BankFactory" ==> BankFactory.fo
[CXX] Compiling Bank "CachedDDR3Bank" ==> CachedDDR3Bank.fo
[CXX] Compiling Bank "DDR3Bank" ==> DDR3Bank.fo
[CXX] Compiling Data Encoder Factory "DataEncoderFactory" ==> DataEncoderFactory.fo
[CXX] Compiling Data Encoder "FlipNWrite" ==> FlipNWrite.fo
[CXX] Compiling Address Translator "DRDecoder" ==> DRDecoder.fo
[CXX] Compiling Address Translator Factory "DecoderFactory" ==> DecoderFactory.fo
[CXX] Compiling Address Translator "Migrator" ==> Migrator.fo
[CXX] Compiling Endurance Model "BitModel" ==> BitModel.fo
[CXX] Compiling Endurance Model "ByteModel" ==> ByteModel.fo
[CXX] Compiling Endurance Model "Normal" ==> Normal.fo
[CXX] Compiling Endurance Model "Uniform" ==> Uniform.fo
[CXX] Compiling Endurance Model Factory "EnduranceDistributionFactory" ==> EnduranceDistributionFactory.fo
[CXX] Compiling Endurance Model Factory "EnduranceModelFactory" ==> EnduranceModelFactory.fo
[CXX] Compiling Endurance Model "NullModel" ==> NullModel.fo
```

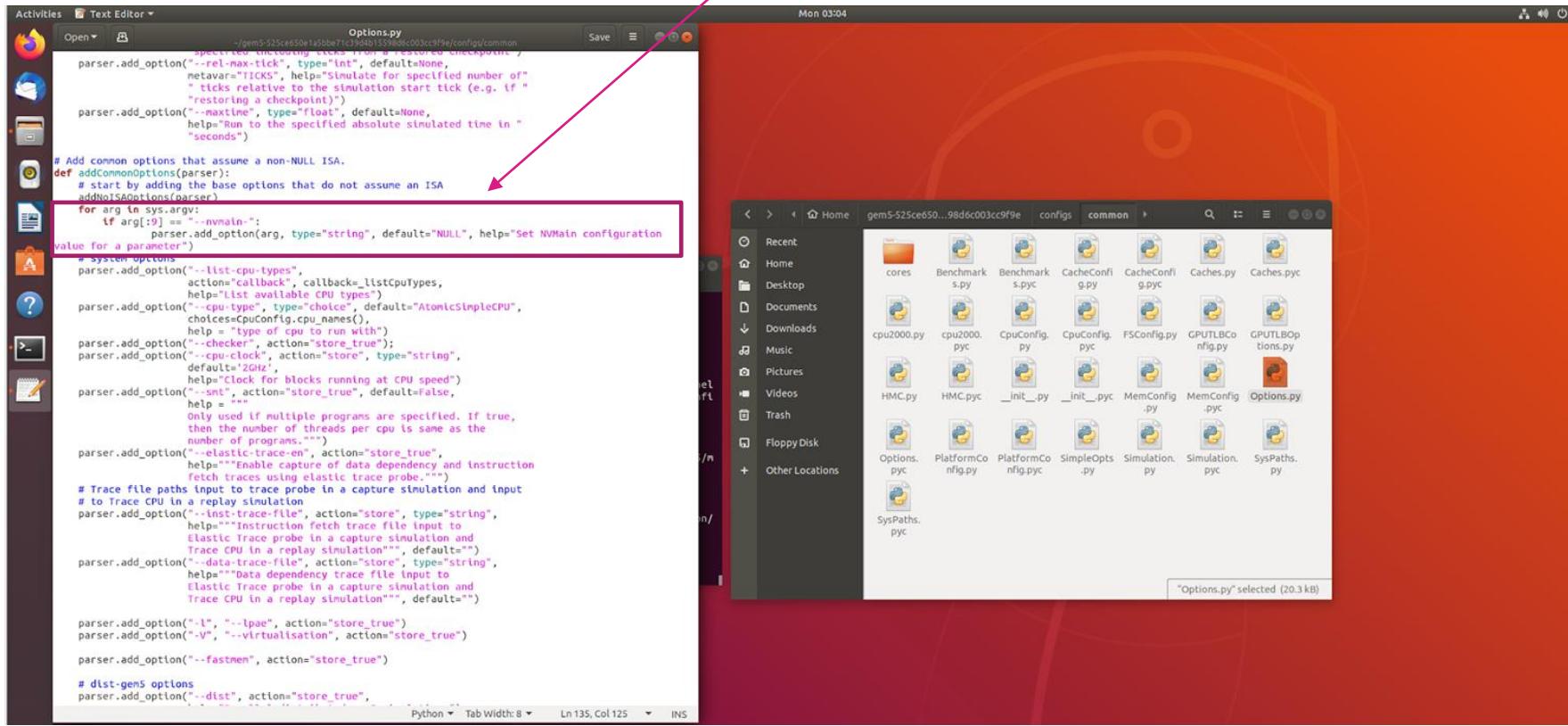
修改GEM5 OPTIONS

■ 在gem5/configs/common/Options.py中第133行加入下段程式(記得要存檔)

■ for arg in sys.argv:

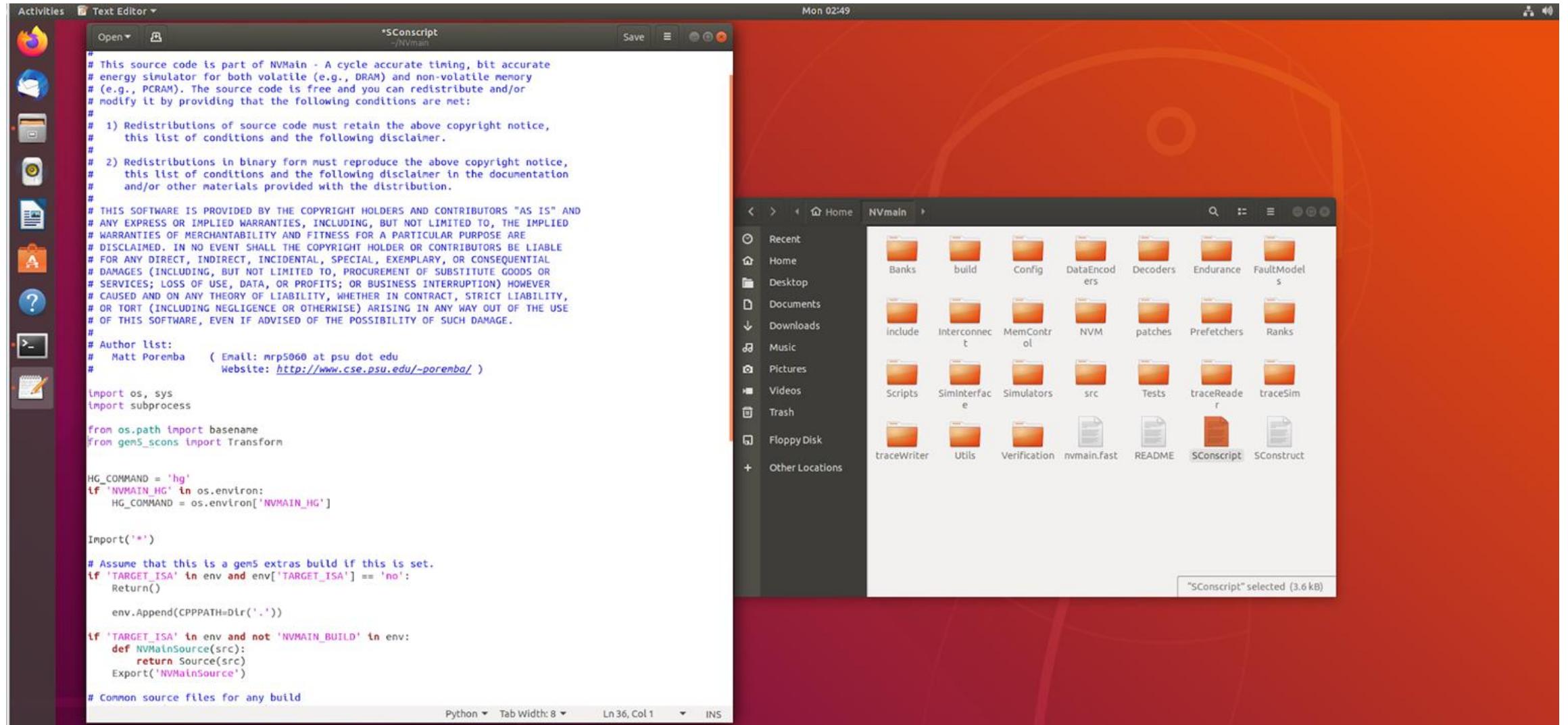
■ if arg[:9] == "--nvmain-":

■ parser.add_option(arg, type="string", default="NULL", help="Set NVMain configuration value for a parameter")



還原 SCONSCRIPT

■ 還原前面nvmain sconsctip註解掉的指令



A screenshot of an Ubuntu desktop environment. On the left, a terminal window titled "Text Editor" displays the content of an "SConscript" file. The file contains Python code for a build system, including comments about redistribution and copyright. On the right, a file browser window titled "NVmain" shows a directory structure with various subfolders like Banks, build, Config, DataEncoders, Decoders, Endurance, FaultModels, include, Interconnect, MemControl, NVM, patches, Prefetchers, Ranks, Scripts, SimInterface, Simulators, src, Tests, traceReader, traceSim, traceWriter, Utils, Verification, nvmain.fast, README, SConstruct, and SConscript. A tooltip at the bottom right of the file browser says "SConscript selected (3.6 kB)".

```
# This source code is part of NVMain - A cycle accurate timing, bit accurate
# energy simulator for both volatile (e.g., DRAM) and non-volatile memory
# (e.g., PCRAM). The source code is free and you can redistribute and/or
# modify it by providing that the following conditions are met:
#
# 1) Redistributions of source code must retain the above copyright notice,
#    this list of conditions and the following disclaimer.
#
# 2) Redistributions in binary form must reproduce the above copyright notice,
#    this list of conditions and the following disclaimer in the documentation
#    and/or other materials provided with the distribution.
#
# THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
# ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
# WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
# DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
# FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
# DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
# SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
# CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
# OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
# OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Author list:
# Matt Poremba ( Email: mrp5060 at psu dot edu
# Website: http://www.cse.psu.edu/~poremba/ )

import os, sys
import subprocess

from os.path import basename
from gen5_scons import Transform

HG_COMMAND = 'hg'
if 'NVMIN_HG' in os.environ:
    HG_COMMAND = os.environ['NVMIN_HG']

Import('*')

# Assume that this is a gem5 extras build if this is set.
if 'TARGET_ISA' in env and env['TARGET_ISA'] == 'no':
    Return()

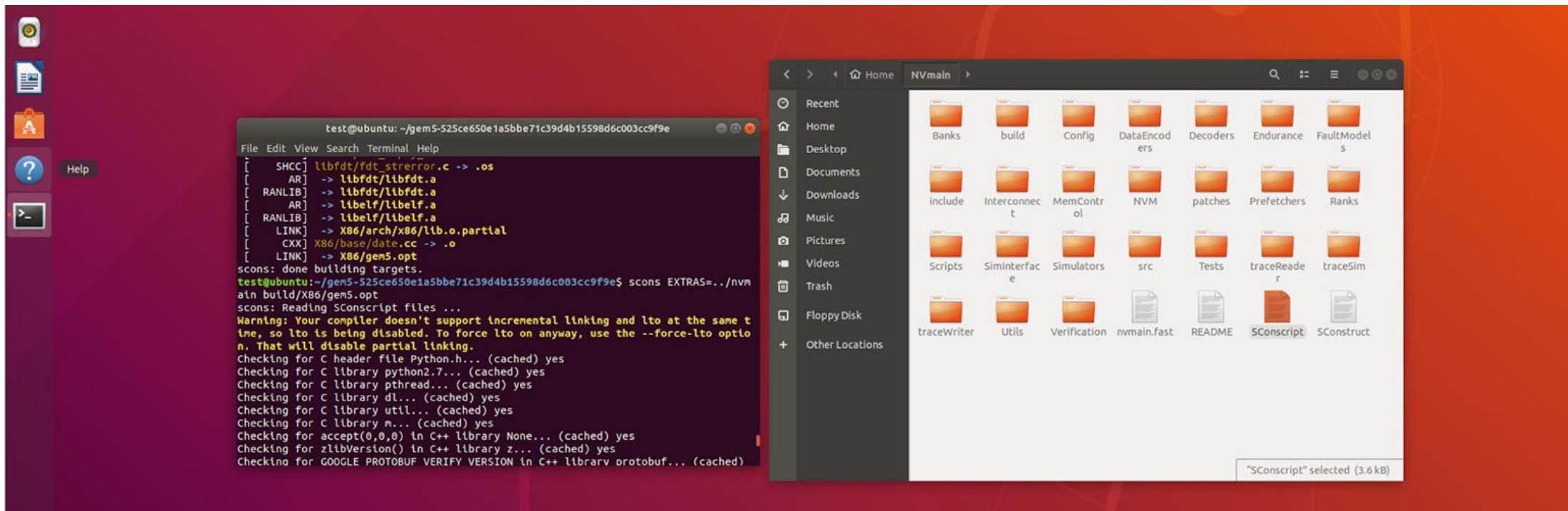
env.Append(CPPPATH=Dir('.'))

if 'TARGET_ISA' in env and not 'NVMIN_BUILD' in env:
    def NVMainSource(src):
        return Source(src)
    Export('NVMainSource')

# Common source files for any build
```

混和編譯 GEM5

- 在GEM5目錄下 混合NVMMAIN編譯GEM5
- scons EXTRAS=../NVmain build/X86/gem5.opt



測試HELLO WORLD

■ ./build/X86/gem5.opt configs/example/se.py -c tests/test-progs/hello/bin/x86/linux/hello --cpu-type=TimingSimpleCPU --caches --l2cache --mem-type=NVMMainMemory --nvmain-config=../NVmain/Config/PCM_ISSCC_2012_4GB.config

Gem5 執行檔

程式執行檔

nvm

```
test@ubuntu: ~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e
File Edit View Search Terminal Help
10.defaultMemory.channel0.RFCFS-WQF.averageTotalLatency 0
10.defaultMemory.channel0.RFCFS-WQF.measuredLatencies 0
10.defaultMemory.channel0.RFCFS-WQF.measuredQueueLatencies 0
10.defaultMemory.channel0.RFCFS-WQF.measuredTotalLatencies 0
10.defaultMemory.channel0.RFCFS-WQF.simulation_cycles 1188
10.defaultMemory.channel0.RFCFS-WQF.wakeupCount 0
10.defaultMemory.totalReadRequests 0
10.defaultMemory.totalWriteRequests 0
10.defaultMemory.successfulPrefetches 0
10.defaultMemory.unsuccessfulPrefetches 0
test@ubuntu:~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e$ ./build/X86/gem5.opt configs/example/se.py -c tests/test-progs/hello/bin/x86/linux/hello --cpu-type=TimingSimpleCPU --caches --l2cache --mem-type=NVMMainMemory --nvmain-config=../NVmain/Config/PCM_ISSCC_2012_4GB.config
gem5 Simulator System. http://gem5.org
gem5 is copyrighted software; use the --copyright option for details.

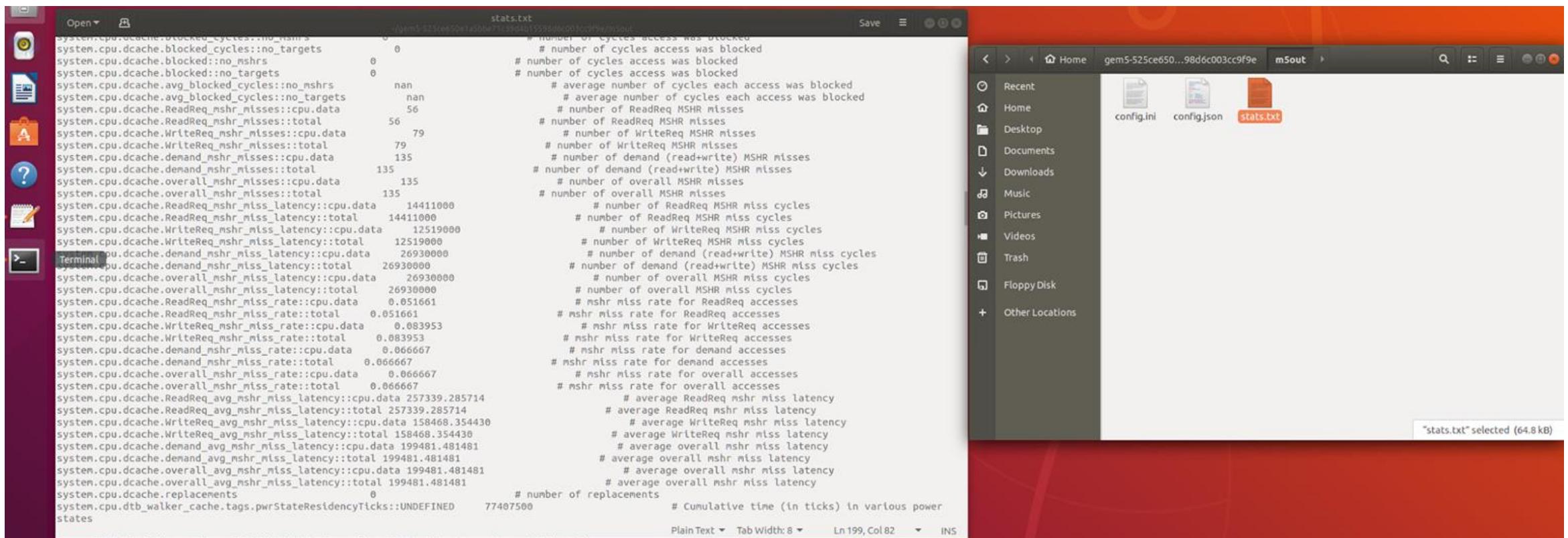
gem5 compiled May 10 2021 02:54:14
gem5 started May 10 2021 03:05:52
gem5 executing on ubuntu, pid 21001
command line: ./build/X86/gem5.opt configs/example/se.py -c tests/test-progs/hello/bin/x86/linux/hello --cpu-type=TimingSimpleCPU --caches --l2cache --mem-type=NVMMainMemory --nvmain-config=../NVmain/Config/PCM_ISSCC_2012_4GB.config
```

```
test@ubuntu: ~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e
File Edit View Search Terminal Help
Config: Warning: Key Decoder is not set. Using '' as the default. Please config
ure this value if this is wrong.
Config: Warning: Key RankType is not set. Using '' as the default. Please config
ure this value if this is wrong.
Config: Warning: Key BankType is not set. Using '' as the default. Please config
ure this value if this is wrong.
NVMMain: the address mapping order is
    Sub-Array 1
    Row 6
    Column 2
    Bank 4
    Rank 5
    Channel 3
defaultMemory.channel0.RFCFS-WQF capacity is 4096 MB.
Creating 4 command queues.
**** REAL SIMULATION ****
info: Entering event queue @ 0. Starting simulation...
Hello world!
Exiting @ tick 77407500 because exiting with last active thread context
10.defaultMemory.channel0.RFCFS-WQF.channel0.rank0.bank0.subarray0.subArrayEnergy 4.33674nJ
10.defaultMemory.channel0.RFCFS-WQF.channel0.rank0.bank0.subarray0.activeEnergy 4.1412nJ
```

Hello world!

OUTPUT

■ 可以由gem5/m5out中的stat.txt查看cache hit 的次數



The screenshot shows a Linux desktop environment with a terminal window displaying the contents of the 'stats.txt' file from the 'm5out' directory. The terminal window has a dark theme and shows the following text:

```
system.cpu.dcache.blocked_cycles::no_mshrs          0           # number of cycles access was blocked
system.cpu.dcache.blocked_cycles::no_targets         0           # number of cycles access was blocked
system.cpu.dcache.blocked::no_mshrs                  0           # number of cycles access was blocked
system.cpu.dcache.blocked::no_targets                0           # number of cycles access was blocked
system.cpu.dcache.avg_blocked_cycles::no_mshrs       nan        # average number of cycles each access was blocked
system.cpu.dcache.avg_blocked_cycles::no_targets     nan        # average number of cycles each access was blocked
system.cpu.dcache.ReadReq_mshr_misses::cpu.data      56          # number of ReadReq MSHR misses
system.cpu.dcache.ReadReq_mshr_misses::total          56          # number of ReadReq MSHR misses
system.cpu.dcache.WriteReq_mshr_misses::cpu.data      79          # number of WriteReq MSHR misses
system.cpu.dcache.WriteReq_mshr_misses::total          79          # number of WriteReq MSHR misses
system.cpu.dcache.demand_mshr_misses::cpu.data        135         # number of demand (read+write) MSHR misses
system.cpu.dcache.demand_mshr_misses::total            135         # number of demand (read+write) MSHR misses
system.cpu.dcache.overall_mshr_misses::cpu.data        135         # number of overall MSHR misses
system.cpu.dcache.overall_mshr_misses::total            135         # number of overall MSHR misses
system.cpu.dcache.ReadReq_mshr_miss_latency::cpu.data 14411000    # number of ReadReq MSHR miss cycles
system.cpu.dcache.ReadReq_mshr_miss_latency::total     14411000    # number of ReadReq MSHR miss cycles
system.cpu.dcache.WriteReq_mshr_miss_latency::cpu.data 12519000    # number of WriteReq MSHR miss cycles
system.cpu.dcache.WriteReq_mshr_miss_latency::total    12519000    # number of WriteReq MSHR miss cycles
system.cpu.dcache.demand_mshr_miss_latency::cpu.data   26930000    # number of demand (read+write) MSHR miss cycles
system.cpu.dcache.demand_mshr_miss_latency::total      26930000    # number of demand (read+write) MSHR miss cycles
system.cpu.dcache.overall_mshr_miss_latency::cpu.data  26930000    # number of overall MSHR miss cycles
system.cpu.dcache.overall_mshr_miss_latency::total      26930000    # number of overall MSHR miss cycles
system.cpu.dcache.ReadReq_mshr_miss_rate::cpu.data    0.051661   # mshr miss rate for ReadReq accesses
system.cpu.dcache.ReadReq_mshr_miss_rate::total        0.051661   # mshr miss rate for ReadReq accesses
system.cpu.dcache.WriteReq_mshr_miss_rate::cpu.data   0.083953   # mshr miss rate for WriteReq accesses
system.cpu.dcache.WriteReq_mshr_miss_rate::total        0.083953   # mshr miss rate for WriteReq accesses
system.cpu.dcache.demand_mshr_miss_rate::cpu.data     0.066667   # mshr miss rate for demand accesses
system.cpu.dcache.demand_mshr_miss_rate::total          0.066667   # mshr miss rate for demand accesses
system.cpu.dcache.overall_mshr_miss_rate::cpu.data     0.066667   # mshr miss rate for overall accesses
system.cpu.dcache.overall_mshr_miss_rate::total          0.066667   # mshr miss rate for overall accesses
system.cpu.dcache.ReadReq_avg_mshr_miss_latency::cpu.data 257339.285714 # average ReadReq mshr miss latency
system.cpu.dcache.ReadReq_avg_mshr_miss_latency::total 257339.285714 # average ReadReq mshr miss latency
system.cpu.dcache.WriteReq_avg_mshr_miss_latency::cpu.data 158468.354430 # average WriteReq mshr miss latency
system.cpu.dcache.WriteReq_avg_mshr_miss_latency::total 158468.354430 # average WriteReq mshr miss latency
system.cpu.dcache.demand_avg_mshr_miss_latency::cpu.data 199481.481481 # average overall mshr miss latency
system.cpu.dcache.demand_avg_mshr_miss_latency::total 199481.481481 # average overall mshr miss latency
system.cpu.dcache.overall_avg_mshr_miss_latency::cpu.data 199481.481481 # average overall mshr miss latency
system.cpu.dcache.overall_avg_mshr_miss_latency::total 199481.481481 # average overall mshr miss latency
system.cpu.dcache.replacements                      0           # number of replacements
system.cpu.dtb_walker_cache.tags.pwrStateResidencyTicks::UNDEFINED 77407500  # Cumulative time (in ticks) in various power states
```

The terminal window also shows the file's size: "stats.txt selected (64.8 kB)".

ACTIVE ENERGY

■ 在執行完程式後就可以看到

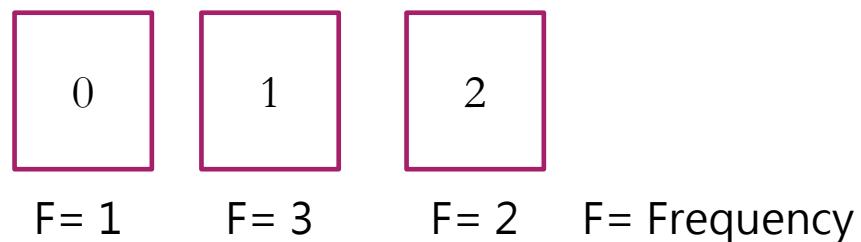
```
test@ubuntu: ~/gem5-525ce650e1a5bbe71c39d4b15598d6c003cc9f9e
File Edit View Search Terminal Help
Channel 3
defaultMemory.channel0.RRFCFS-WQF capacity is 4096 MB.
Creating 4 command queues.
**** REAL SIMULATION ****
info: Entering event queue @ 0. Starting simulation...
Hello world!
Exiting @ tick 77407500 because exiting with last active thread context
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.subArrayEnergy 4.33674nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.activeEnergy 4.1412nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.burstEnergy 0.195536nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.writeEnergy 0nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.refreshEnergy 0nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.cancelledWrites 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.cancelledWriteTime 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.pausedWrites 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.averagePausesPerRequest 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.measuredPauses 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.averagePausedRequestProgress 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.measuredProgresses 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.reads 121
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.writes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.activates 51
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.precharges 50
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.refreshes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.worstCaseEndurance 1844674407370955161
5
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.averageEndurance 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.actWaits 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.actWaitTotal 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.actWaitAverage -nan
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.worstCaseWrite 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.num00Writes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.num01Writes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.num10Writes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.num11Writes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.averageWriteTime 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.measuredWriteTimes 0
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.mlcTimingHisto {}
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.cancelCountHisto {}
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.wpPauseHisto {}
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.subarray0.wpCancelHisto {}
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.bankEnergy 4.33674nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.activeEnergy 4.1412nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.burstEnergy 0.195536nJ
10.defaultMemory.channel0.RRFCFS-WQF.channel0.rank0.bank0.refreshEnergy 0nJ
```

■ ENABLE L3 CACHE簡易提示

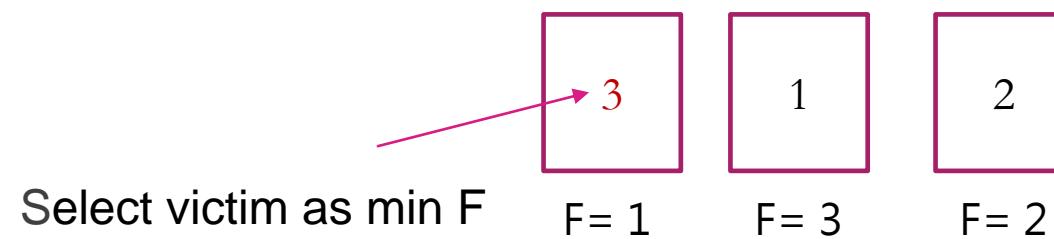
- 需要修改的檔案 在gem5資料夾中的
 - Options.py
 - Caches.py
 - Xbar.py
 - BaseCPU.py
 - CacheConfig.py
- 前面四個檔案只是增加L3 cache的parameter，照著L2 cache的設定去做模仿就可以。
- CacheConfig.py需要讓L3 cache連接整個Gem5系統，這邊要注意 L2跟L3這兩個cache的關係，**要讓系統在已使用L2 cache的情況下才能使用L3 cache**，所以修要注意修改的時候有沒有滿足這個條件。
- 細節code的部分，同學可以上網找資源關鍵字 Gem5 L3 cache之類的。

FREQUENCY BASED REPLACEMENT POLICY

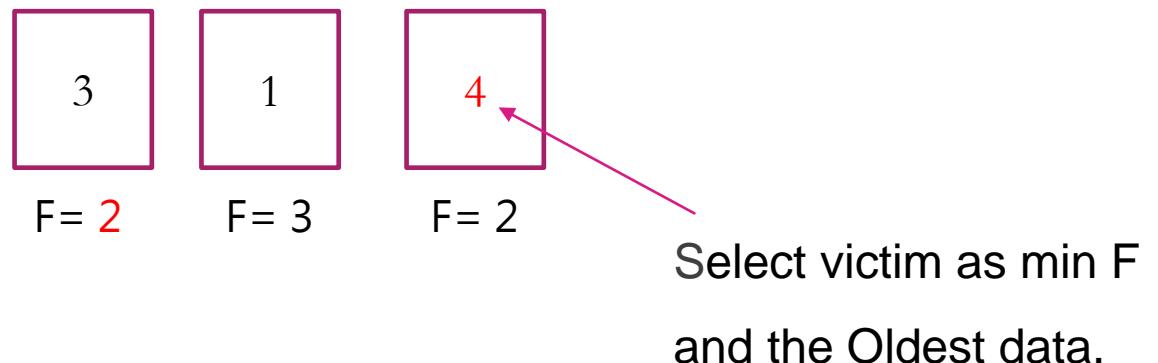
- Cache size = 3 full associative blocks
- block request 0 , 1 , 2 , 1, 2, 1



- block request 3



- block request 3, 4



■ WRITE BACK AND WRITE THROUGH 提示

- 先了解什麼是write back 和 write through (應該上課會教)
- 可以去mem/cache的資料夾下找到兩個檔案一個叫做cache.cc另一個叫做base.cc，可以研究一下gem5底層的code是如何把資料寫入Memory。