**Project #3**

1. Project Objective
   1. Based on the single-cycled CPU simulator from project #1, implement a simulator with memory hierarchy, Translation-Lookaside Buffer (TLB), and virtual page table mechanism. **The memory size, page size, cache total size**, **block size** and **set associativity of the cache should be configurable.**
   2. Design and submit a test case to verify the functionality of the memory hierarchy configuration.
2. About the Simulator

The simulator is similar to that of project 1 except for the followings:

* 1. The executable should be named **CMP (which stands for Cache\_Memory\_Pagetable)**.
  2. All instructions access through **virtual address**
  3. Both instruction cache and data cache have one level cache. In other words, your simulator has to simulate two caches, L1 instruction cache and L1 data cache.
  4. Cache Organization
     1. Both instruction cache and data cache use **write-back/allocate** policy.
     2. Both instruction cache and data cache use the following replacement policy to deal with cache misses: For the cache line under consideration, replace the **invalid** set if possible; otherwise, replace the **LRU** set. Pick the **least indexed** set to be the victim in case that there is more than one candidate.
     3. The **default** instruction cache is **of 16 bytes, 4-way associative. The block size for instruction cache is 4 bytes**.
     4. The **default** data cache is **of 16 bytes, direct map. The block size for data cache is 4 bytes.**
  5. Cache Initialization

The valid bit of each cache block is set to be false before the simulation begins. All other contents are initialized as “don’t cares” (x’s).

* 1. TLB Organization
     1. There should be two TLBs, one for IPageTable and one for DPageTable.
     2. The TLBs is direct mapped and its size is a quarter of the page table size, i.e.,#TLB\_entries =1/4\*(#page\_table\_entries).
     3. TLBs adopt the same replacement policy as that for cache. In other words, replace the invalid entry if possible; otherwise, replace the LRU entry.
  2. Page Table Organization
     1. Although theoretically we should have page table cover the full 32-bit virtual space, for simplicity of verification you are required to calculate the page table size from the default disk size in this project, i.e.,

#page\_table\_entries = disk\_size / page\_size.

* + 1. You have to map virtual address (VA) to physical address (PA)
    2. At page fault, we assume that the virtual address (VA) is exactly the disk address.
    3. The defaultdata page size is **16 bytes** and the default instruction page size is **8 bytes.**
  1. Page Table Initialization

The valid bit of each page table block is initialized to be false before simulation begins. All other contents are “don’t cares” (x’s).

* 1. Memory Organization
     1. Both instruction memory and data memory use **write-back/allocate** policy.
     2. Memory replacement policy for page faults: If memory space is available, place data to the first available page closest to the page zero; otherwise, replace the **LRU** set. Pick the **least indexed** set to be the victim in case of tie
     3. The **default** instruction memory size is **64 bytes** andthe **default** data memory size is **32 bytes.**
  2. Memory Initialization
     1. All memory contents are initialized to 0’s.
  3. Disk Initialization
     1. Assume that both the instruction disk and data disk are of **1K bytes size.**
     2. All other addresses not covered by the image are assumed to have been initialized to 0’s.

For configurability, the executable takes arguments **from the command line**. **All size parameters should be multiple of four.** Note that if no command line parameters are set, the default configuration should be used for simulation. Other specifications are the same as *project\_1.pdf*. The parameters should be of the following order:

1. The instruction memory (I memory) size, in number of bytes
2. The data memory (D memory) size, in number of bytes
3. The page size of instruction memory (I memory), in number of bytes
4. The page size of data memory (D memory), in number of bytes
5. The total size of instruction cache (I cache), in number of bytes
6. The block size of I cache, in number of bytes
7. The set associativity of I cache
8. The total size of data cache (D cache), in number of bytes
9. The block size of D cache, in number of bytes
10. The set associativity of D cache
11. Input Format

It is the same as that of Project 1.Please refer to the specification of Project 1 and *Appendix B*, “*Sample Input*.”

1. Output Requirement

For each test case, **report.rpt** and **snapshop.rpt** should be generated.

* 1. **snapshot.rpt**  
     The requirement is the same as that for project 1. Please refer to *project\_1* and *Appendix C-1*, “*Sample Output for Project 1*.”
  2. **report.rpt:** For details please refer to *Appendix C-3*, “*Sample Output for Project 3*.”

**report.rpt** should contain the following information for total memory access: total hit / miss number of Icache, Dcache, Ipagetable, Dpagetable.

1. Test case design

Design a test case to verify if your simulator handles every cache event correctly with the default configuration. By cache events, we mean read hit, read miss, write hit and write miss.

**Note that your test case should not execute more than 30,000 cycles and should not cause initialization error or it will be deemed invalid. Any error which cause simulator to halt (defined in appendix of proj1 ) should not be in your test case. (No address overflow error、No misalignment error)**

**The definition of initialization is the action that load your iimage.bin /dimage.bin files into your I-Memory/D-memory.**

1. Evaluation and grading

Same as Project1

Note that for all evaluations, verifications will be done on **nthucad (on backend ic17 )workstation**. Furthermore, we will evaluate your project using scripts. Please make sure that your project can be executed by the script provided by TA’s. **If it cannot run through the script, you will lose all your points even if your program or result is correct.**

1. **Etiquette**
   1. **Do not copy others’ works, or you will fail this course.**
   2. **No acceptance of late homework.**
   3. **Please constantly check the class website announcements for any possible updates.**