HW4: Memory

***Due May 6th, 2023, 11:59pm***

## extra credit available for early submissions!

There are 2 parts to this assignment: Part 1: written exercises for Ch5; Part 2: cache simulation with MARS. Basic rules regarding assignment implementation and submission:

* Total points of this assignment: 100.
* **Team Allowed. Maximum 2 per Team**. You must include names and G#’s of **BOTH** group members in **ALL** submitted files**.**

***Submission Instructions***

* Submit to **gradescope**. A link to gradescope is available from Blackboard. There will be two assignments on gradescope, one for Part 1 and one for Part 2. You will lose "off the top" points if you fail to follow the submission instructions.
* For both Part1 and Part 2, follow the submission instruction from gradescope. You will need to upload a **PDF** and **specify on which page** we can find your answer. Make sure your answer is legible.
  + **Up to -5 points** for incorrect format or not specifying pages.
* For teams with two people:
  + For each part, only **one** team member should submit to Gradescope.
  + You won’t be able to specify the team when you submit. Instead, you must **include names and G#’s of both group members** in all submitted files**.** We will use the provided group information to assign the same grade to both members.

# Plagiarism is not permitted in any form. I enforce the university honor code.

***Part 1. Written Exercise for Memory (60%)***

# Notes:

* **A large portion of (or all) points will be taken off if you do not include detailed calculation in your answer.** You must show steps to justify your answer.
* **Answers must be legible**, especially if you scan to generate your submission.

1. (12 pts) Explain how a 32-bit byte memory address should be divided into Tag/Index/Offset fields for each of the cache configurations below. **Note**: 1KB = 210 bytes, 1 word = 4 bytes. You must include a brief explanation or calculation steps to show how to decide the length of each field. You get at most 50% of the credit if you give the lengths without an explanation.
   1. A direct-mapped cache with cache block size = 16 words and cache size = 256KB.
      * The offset is 6 bits, the cache block holds 16 words and each word is 4 bytes. 16\*4 is 64 and 2^6 is 64, therefore the size of the offset is 6 bits.
      * The index is 12 bits as the cache size is 256 KB and each block is 64 bytes. 254 KB / 64 bytes is 2^12.
      * The tag is the remaining bits so 32-6-12 which equates to 14 bits.
   2. A fully associative cache with cache block size = 16 words and cache size = 256KB.
      * The offset is 6 bits, the cache block holds 16 words and each word is 4 bytes. 16\*4 is 64 and 2^6 is 64, therefore the size of the offset is 6 bits.
      * The index is nonexistent in a fully associative cache, so it is 0 bits.
      * The tag is the remaining bits which is 32-6 which equates to 26 bits.
   3. A four-way set associative cache with cache block size = 16 words and cache size = 256KB.
      * The offset is 6 bits, the cache block holds 16 words and each word is 4 bytes. 16\*4 is 64 and 2^6 is 64, therefore the size of the offset is 6 bits.
      * The index is found by finding the number of sets there are, number of sets is number of blocks / associativity which is 2^12 / 4 which is 2^10. Therefore, the number of index bits is 10.
      * Tag is the remaining bits and it would be 32-10-6 which is 16 bits.
2. (10 pts) Assume that main memory accesses take 60ns while L1 cache has the following features: L1 cache hit time: 0.5ns, L1 cache miss rate: 8%.
   1. What is the average memory access time if we only have L1 cache?
      * Average Memory Access Time (AMAT) = Hit Time + Miss Rate x Miss Penalty
      * 0.5ns + 60ns \* 0.08 = 5.3 ns
   2. Now suppose we add an L2 cache with hit time as 5ns. What would the L2 miss rate need to be in order for the average memory access time to be improved compared with L1 cache only?
      * 5ns + 60ns \* ?
      * ? is the miss rate and the miss rate would have to be lower than 0.005 or .5 percent in order for the L2 cache average memory access time to be improved compared to with L1 cache only. 5ns + 60ns \* 0.005 = 5.3 ns.
3. (15 pts) Consider a **direct-mapped** cache with a total size of 4 blocks and block size as 1 word. Given the initial cache status as below, fill the table for a sequence of five accesses (given as byte addresses in **hexadecimal**). Also fill the final cache table to describe the final cache status after the sequence completes. Assume that **0** in **Valid** column means an invalid block.

# Initial cache Final cache

|  |  |  |
| --- | --- | --- |
| **Index** | **Valid** | **Tag** |
| **0** | **1** | **0x2500** |
| **1** | **0** | **0x6018** |
| **2** | **1** | **0x1647** |
| **3** | **0** | **0xA100** |

|  |  |  |
| --- | --- | --- |
| **Index** | **Valid** | **Tag** |
| **0** | 1 | 0x2B01 |
| **1** | 1 | 0x6300 |
| **2** | 1 | 0x3061 |
| **3** | 0 | 0xA100 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Access** | **Byte Address** | **Index** | **Tag** | **Hit/Miss?** |
| **1** | **0x25001** | 0 | 0x2500 | Hit |
| **2** | **0x63004** | 1 | 0x6300 | Miss |
| **3** | **0x2B013** | 0 | 0x2B01 | Miss |
| **4** | **0x3061A** | 2 | 0x3061 | Miss |
| **5** | **0x63007** | 1 | 0x6300 | Hit |

1. (23 pts) Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. The table will be updated with accesses. Assume the following configurations:

* Page table is indexed by VPN (virtual page number): row 0 is for VPN 0, etc.
* Assume initially the largest PPN (physical page number) in use is **8**. If a new page needs to be brought in from the disk, we always assign the next physical page number to it. Hence the very next page that we bring in from disk would have a PPN 9, then PPN 10, etc.
* TLB is **fully associative** with **4** entries and true **LRU replacement**.

Fill the table below for a sequence of virtual addresses (given as VPNs in the table). For each address, determine whether it is a TLB miss/hit, and (if it is a TLB miss) whether it is a page table hit/page fault. Show how TLB is updated for every access as well as the final Page Table.

Initial Page Table: (all other entries are invalid)

|  |  |  |
| --- | --- | --- |
| Index (VPN) | Valid | Physical Page Number or in Disk |
| 0 | 1 | 5 |
| 1 | 0 | Disk |
| 2 | 1 | 3 |
| 3 | 0 | Disk |
| 4 | 1 | 8 |
| 5 | 0 | Disk |
| 6 | 1 | 2 |
| 7 | 1 | 1 |

Initial TLB:

|  |  |  |  |
| --- | --- | --- | --- |
| Valid | Tag (VPN) | Physical Page Number | LRU  (**1 as most recent**) |
| 1 | 0 | 5 | 3 |
| 1 | 2 | 3 | 4 |
| 1 | 4 | 8 | 2 |
| 1 | 7 | 1 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Access** | **Address (VPN)** | **TLB Hit/Miss; Page Table Hit/**  **Page Fault** | **TLB** | | | |
| **Valid** | **Tag** | **PPN** | **LRU** |
| **1** | **0x6** | TLB Miss  Page Table Hit | 1 | 0 | 5 | 4 |
| 1 | 6 | 2 | 1 |
| 1 | 4 | 8 | 3 |
| 1 | 7 | 1 | 2 |
| **2** | **0x5** | TLB Miss  Page Fault | 1 | 5 | 9 | 1 |
| 1 | 6 | 2 | 2 |
| 1 | 4 | 8 | 4 |
| 1 | 7 | 1 | 3 |
| **3** | **0x4** | TLB Hit  Page Table Hit | 1 | 5 | 9 | 2 |
| 1 | 6 | 2 | 3 |
| 1 | 4 | 8 | 1 |
| 1 | 7 | 1 | 4 |
| **4** | **0x2** | TLB Miss  Page Table Hit | 1 | 5 | 9 | 3 |
| 1 | 6 | 2 | 4 |
| 1 | 4 | 8 | 2 |
| 1 | 2 | 3 | 1 |
| **5** | **0x5** | TLB Hit  Page Table Hit | 1 | 5 | 9 | 1 |
| 1 | 6 | 2 | 4 |
| 1 | 4 | 8 | 3 |
| 1 | 2 | 3 | 2 |

# Final Page table:

(**use as many rows as you need**)

|  |  |  |
| --- | --- | --- |
| Index | Valid | Physical Page Number or in Disk |
| 0 | 1 | 5 |
| 1 | 0 | Disk |
| 2 | 1 | 3 |
| 3 | 0 | Disk |
| 4 | 1 | 8 |
| 5 | 1 | 9 |
| 6 | 1 | 2 |
| 7 | 1 | 1 |
| 8 |  |  |
| 9 |  |  |

## Part 2. MARS Memory Simulation (40%)

For this assignment, you will use MARS Data Cache Simulator to check and compare cache performance for different cache configurations and better understand the memory behavior. **Warm up**:

* Read MARS Tutorial Part 2 Activity 1: Running the Data Cache Simulator tool.
* Try Step 1-7 with the provided **row\_major.asm**.

**Simulation**: With the provided **row\_major.asm**, simulate the following cache configurations. For each simulation, record the final hit rate at the end of the simulation. **NOTE**:

* Make sure the simulation always starts with an empty cache. You may need to reset the MIPS program and reset the tool before you can re-run it.
* Feel free to adjust the Run Speed slider to any speed anytime you want to.
* Another MARS tool, Memory Reference Visualization, can also provide helpful insight about the memory behavior of the program. You can link both to the MIPS simulation.

Required configurations to simulate:

* Group 1, Directed-mapping:
  + Number of Blocks: 8, Cache block size: 2 words, LRU
  + Number of Blocks: 8, Cache block size: 4 words, LRU (this is the default setting)
  + Number of Blocks: 8, Cache block size: 8 words, LRU
  + Number of Blocks: 8, Cache block size: 16 words, LRU
* Group 2, Directed-mapping:
  + Number of Blocks: 8, Cache block size: 8 words, LRU
  + Number of Blocks: 16, Cache block size: 8 words, LRU
  + Number of Blocks: 32, Cache block size: 8 words, LRU
* Group 3, Directed-mapping:
  + Number of Blocks: 16, Cache block size: 8 words, LRU
  + Number of Blocks: 8, Cache block size: 16 words, LRU
  + Number of Blocks: 4, Cache block size: 32 words, LRU
  + Number of Blocks: 2, Cache block size: 64 words, LRU
* Group 4, Fully associative:
  + Number of Blocks: 8, Cache block size: 4 words, LRU
  + Number of Blocks: 8, Cache block size: 4 words, Random
    - Reset/repeat 3 times: report all three and the average hit rates
  + Number of Blocks: 4, Cache block size: 8 words, LRU
  + Number of Blocks: 4, Cache block size: 8 words, Random
    - Reset/repeat 3 times: report all three and the average hit rates
* Group 5:
  + Direct-mapping, Number of Blocks: 16, Cache block size: 8 words, LRU
  + Fully associative, Number of Blocks: 16, Cache block size: 8 words, LRU
  + N-way set associative, Number of Blocks: 16, Cache block size: 8 words, LRU, Set size: 1
  + N-way set associative, Number of Blocks: 16, Cache block size: 8 words, LRU, Set size: 2
  + N-way set associative, Number of Blocks: 16, Cache block size: 8 words, LRU, Set size: 4
  + N-way set associative, Number of Blocks: 16, Cache block size: 8 words, LRU, Set size: 8

**Report**: Write a report based on your simulation results.

* For every group, summarize the **hit rate** in a table or a histogram. Make sure to include labels and captions to present your data clearly.
* For every group, explain and discuss the difference you observe for the different configurations in a group: What is the change (hit rate increasing/decreasing)? What is the reason of the change (or no-change)?

# Grading rubric:

* + Data collected and presented clearly 25/40
  + Discussion/explanation of the data 15/40
* We will be comparing the generated output file and the expected output file for grading.
* **No late work allowed after 24 hours.**
  + Late submission automatically uses one of your tokens. If you run out of tokens, late penalty will be applied.

# Extra credit for early submissions:

* 1% extra credit rewarded for every 24 hours your submission made before the due time.
* Up to 3% extra credit will be rewarded.
* Your latest submission will be used for grading and extra credit checking. You CANNOT choose which one counts.
* You will need to make two separate submissions for Part1 and Part2 respectively. The later one of those two will be used to determine whether your HW4 is early / normal/ late.