**Student Name:** Sitare Arslantürk

**Student ID:** 57677

# COMP 303 - Comp Arch: HW4

*Due Date: Dec 29th, 2019 3.00 pm*

**Instructor: Didem Unat, TAs: Najeeb Ahmad**

**Notes:** You may discuss the problems with your peers but the submitted work must be your own work. No late assignment will be accepted. Submit a SOFT copy of your assignment to the blackboard *AND* submit a HARD copy of it to the COMP 303 mailboxes or bring it to the FINAL EXAM. This assignment is worth 4% of your total grade.

## Problem 1

(10 pts) Assume a multiprocessor environment that implements a sequentially consistent memory model. We have two processors $P1$ and $P2$.

```
1 =====P1=====                    =====P2======
2 a1: st 0x1, (0x1000)            b1: st 0x3, (0x1000)
3 a2: ld $r1, (0x1000)            b2: ld $r2, (0x1000)
4 a3: st 0x2, (0x1000)            b3: st 0x4, (0x1000)
5 a4: ld $r3, (0x1000)            b4: ld $r4, (0x1000)
```

Assume that the value at address $0x1000$ is initialized to zero.
(a) What are the possible values for $r2 after both processors complete their execution. List all possible values.

0x1, 0x2, 0x3

(b) After both P1 and P2 have finished executing, we realized that ($r1, $r2, $r3, $r4) = (1, 3, 2, 4). List all the different instruction interleavings that produce this result.

a1, a2 – b1, b2 – b3, b4 – a3, a4
b1, b2 – a1, a2 – a3, a4 – b3, b4
a1, a2 – a3, a4 – b1, b2 – b3, b4
a1, a2 – b1, b2 – a3, a4 – b3, b4
b1, b2 – a1, a2 – b3, b4 – a3, a4
b1, b2 – b3, b4 – a1, a2 – a3, a4

## Problem 2

(12 points) Consider a memory system with the following parameters:
- Translation Lookaside Buffer has 128 entires and it is 2-way set associative.
- Page size is 8 KB.
- The tag bits for TLB is 31 bits.
- Physical memory is 32GB.

a) What is the virtual address size (in bits)?

$128 \rightarrow 2^7 / 2 = 2^6$
$8\ KB \rightarrow 2^{13}$
$Tag = 31\ bits \rightarrow 2^{31}$
$31 + 13 + 6 = 50$

b) What is the physical address size (in bits)?

$$\text{physical memory} \rightarrow \frac{32\ GB}{8\ KB} \rightarrow \frac{2^5 \cdot 2^{30}}{2^3 \cdot 2^{10}} = 2^{22}$$

$22 + 13 = 35\ bits$

c) How many physical pages are there?

$2^{35} / 2^{13} = 2^{22}$

d) What is the size of virtual memory in TB?

$2^{50}\ bytes = 2^{10}\ TB = 1024\ TB$

e) If the hit rate is 99% and hit time is 1 cycle to TLB, what is the average memory access time to the TBL if its miss penalty is 20 cycles when there is no page fault?

$AMAT = 1 + 0.01 \times 20 = 1.2$

f) Assume that 0.02% of the time, there is a page fault, and the disk latency is 10,000 cycles. Then, what is the average access time for TLB?

$AMAT = 1 + 0,01 \times (20 + .0.0002 \cdot 10,000) = 1.22$

## Problem 3

(10 points - 2 pts each)

```
1   for(i=0; i< N; i++){
2      for(j=0; j< N; j++){
3          A[i] = A[i] * B[j];
4      }
5    }
```

Assume that this loop nest is executed on a machine with one word cache blocks and a cache that always replaces the least recently used (LRU) block when space is needed. Assume A and B are very large arrays that they cannot fit into the cache. Give concise answers to the following questions and use only the space provided under the question.

a) Explain the cache behaviour for $A$.

N elements used for the first time sequentially, there will be N misses.

b) Explain the cache behaviour for $B$.

Each access in B will result in a miss, it is sequentially accessed so everytime it is accessed, there is a miss.

c) Explain whether the nested loop would benefit from a multi-word cache.

Yes, because this is for loop where there is one step incremental which means a multi-word cache will allow access to the upcoming elements because they are in blocks. We benefit from the spatial locality.

d) Explain whether the nested loop would benefit from a fully-set associative cache.

No, it is redundant to replace B every time we access it.

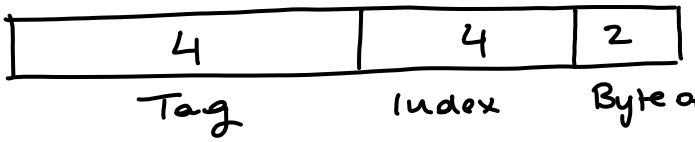
e)Explain whether the nested loop would benefit from random replacement.

Yes, random is always preferred because it is easier compared to LRU. in implementation.

## Problem 4

$2^4 \rightarrow$ index $\quad 2^2 \rightarrow$ offset

(10 points) Assuming a directed-mapped cache with 16 one-word blocks (4 byte blocks).
a) Find the number of tag bits, index bits, and byte offset bits in a 10-bit address.




10_4_2 = 4 ↓ tag

b) Here is **a series of byte** address references in base ten: 4, 16, 32, 4, 6, 68, 144, 4. For the cache described above, label each reference in the list as a hit or a miss. Show the final content of the cache.

index = 4
tag = 4.

| | Valid | tag | data |
|---|---|---|---|
| | | | |
| 0001 | Y | 0001 | M(4-7) |
| | | | |
| 0100 | Y | 0010 | M(144-147) |
| | | | |
| 1000 | Y | 0000 | M(32-35) |
| | | | |

| Byte Address | Hit or Miss |
|---|---|
| 4 | Miss |
| 16 | Miss |
| 32 | Miss |
| 4 | Hit |
| 6 | Hit |
| 68 | Miss |
| 144 | Miss |
| 4 | Miss |

000100
010000
100000
000100
000110
1000100
10010000
000100

0001
0100
1000