16.482 / 16.561: Computer Architecture and Design

Fall 2014

Final Exam
December 4, 2014

Name: ____ **ID #:** _________________________

For this exam, you may use a calculator and two 8.5" x 11" double-sided page of notes. All other electronic devices (e.g., cellular phones, laptops) are prohibited. If you have a cellular phone, please turn it off prior to the start of the exam to avoid distracting other students.

The exam contains 6 questions for a total of 100 points. Please answer the questions in the spaces provided. If you need additional space, use the back of the page on which the question is written and clearly indicate that you have done so.

You will be provided with a single page containing additional copies of information used in Question 1, so that you can more easily refer to this material while answering these questions. You do not have to submit this page when you turn in your exam.

You will have three hours to complete this exam.

Q1: Multiple issue and multithreading	/ 18
Q2: Cache basics	/ 23
Q3: Virtual memory	/ 11
Q4: Cache optimizations	/ 19
Q5: RAID	/ 14
Q6: Multiprocessors	/ 15
TOTAL SCORE	/ 100

1. (18 points) ***<u>Multiple issue and multithreading</u>***

This problem deals with the 3 threads below (which are also shown on your extra handout):

<u>Thread 1:</u>

```
ADD.D F0, F2, F4
L.D   F6, 0(R1)
SUB.D F8, F0, F8
ADD.D F10, F0, F10
MUL.D F4, F8, F6
DADDUI R1, R1, #16
BNE   R1, R2, loop
```

<u>Thread 2:</u>

```
MUL.D F2, F6, F6
S.D   F2, 8(R1)
SUB.D F6, F6, F0
DADDUI R1, R1, #-1
BNEZ  R1, labelA
MUL.D F0, F4, F2
ADD.D F0, F6, F0
J     labelB
```

<u>Thread 3:</u>

```
L.D   F0, 0(R1)
ADD.D F4, F6, F8
ADD.D F10, F0, F4
L.D   F2, 8(R1)
S.D   F10, 16(R1)
L.D   F4, 24(R1)
SUB.D F6, F4, F2
JR    R31
```

In both parts of the problem, the processor executes instructions in order, and you should assume all branches are not taken. Each instruction has the latency given below:

- L.D/S.D: 3 cycles (1 EX, 2 MEM)
- MUL.D: 4 cycles
- ADD.D/SUB.D: 2 cycles
- All other operations: 1 cycle

a. (6 points) Will these threads run faster on a processor that uses fine-grained or coarse-grained multithreading? Assume the coarse-grained processor would switch threads on stalls greater than 3 cycles (i.e., 4 or more cycles). Explain your answer <u>without calculating the total time required for each case.</u> (Hint: look at the number and length of stalls in each thread.)

1 (continued)

b. (12 points) Using the same 3 threads from part (a), determine how long they take to execute using simultaneous multithreading on a processor with the following characteristics:

- Five functional units: 2 ALUs, 2 memory ports (load/store), 1 branch
    - Note: The ALUs can handle MUL.D and DADDUI operations
- Thread 1 is the preferred thread, followed by Thread 2 and Thread 3.

Your solution should use the table below, which contains columns to show each cycle and the functional units being used during that cycle. **Clearly indicate which thread contains each instruction when completing the table, but you do not have to write the full instruction—writing the opcode (i.e. L.D, ADD.D) is sufficient.**

Remember, your extra handout contains a copy of the threads and latencies.

Cycle	ALU1	ALU2	Mem1	Mem2	Branch
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

2. (23 points) ***Cache basics***

a. (4 points) Identify and explain the two types of locality, giving an example of each.

b. (3 points) Explain how LRU replacement works and why it is effective.

2 (continued)

c. (16 points) You are given a system which has a 16-byte, write-back cache with 4-byte blocks. The cache is direct-mapped. The system uses 8-bit addresses, and the cache is initially empty.

Assume the initial memory state shown below for the first 32 bytes:

Address		Address		Address		Address	
0	27	8	19	16	22	24	13
1	3	9	78	17	5	25	24
2	20	10	9	18	15	26	21
3	11	11	12	19	13	27	7
4	5	12	1	20	49	28	18
5	12	13	0	21	77	29	8
6	14	14	63	22	15	30	55
7	2	15	98	23	44	31	99

For each access in the sequence listed below, fill in the cache state, indicate what register (if any) changes, and indicate if any memory blocks are written back and if so, what addresses and values are written. The cache state should carry over from one access to the next.

Access	Modified register	Cache state							Modified mem. block
		V	D	Tag	Data				
`lb $t0,8($zero)`									
`sb $t0,9($zero)`									
`lb $t1,15($zero)`									
`sb $t1,24($zero)`									

3. (11 points) ***Virtual memory***

a. (3 points) Explain the purpose of a translation lookaside buffer (TLB).

b. (8 points) Fill in the table at the bottom of the page to show the **final** state of the given page table after the following sequence of accesses. Assume main memory has 8 frames, numbered 0-7, and frame 5 is the only frame that is free before this sequence executes.

ACCESS SEQUENCE

- Read page 3
- Write page 5
- Write page 1
- Read page 0

INITIAL PAGE TABLE STATE:

Virtual page #	Valid bit	Reference bit	Dirty bit	Frame #
0	1	1	1	3
1	0	0	0	--
2	1	0	1	4
3	0	0	0	--
4	1	1	0	0
5	1	1	0	1

FINAL PAGE TABLE STATE:

Virtual page #	Valid bit	Reference bit	Dirty bit	Frame #
0				
1				
2				
3				
4				
5				

4. (19 points) ***<u>Cache optimizations</u>***

a. (9 points) Briefly explain three of the following four cache optimizations:

i. Trace caches

ii. Way prediction

iii. Early restart/critical word first

iv. Hardware prefetching

4 (continued)

b. (10 points) Assume we have a system containing 32 blocks of memory, numbered 0-31. The processor's L1 data cache in this system contains four banks in which the blocks are sequentially interleaved. This system runs a program in which it must access the following ten blocks of data, but may do so in any order:

   1, 2, 10, 12, 15, 16, 24, 27, 28, 30

Assume all ten accesses are misses, each of which takes 20 cycles to handle, and also assume that only one new access can be started each cycle.

Determine the minimum number of cycles required to access all ten blocks, as well as a sequence of accesses that will take that amount of time. (Note: while there is only one correct value for the minimum number of cycles, there are multiple ways to access all ten blocks in that amount of time.) **For full credit, show all work.**

5. (14 points) ***RAID***
Both parts of this problem involve a six-disk RAID array containing a total of 18 sectors; the exact sector configuration depends on the RAID level used. In all cases, fifteen of the eighteen sectors (S0-S14) hold data, while the remaining three sectors (P0-P2) hold parity information.

a. (6 points) Determine how many of the six disks would be used in each of the following operations in this array, and briefly explain why:

i. Large read

ii. Large write

iii. Small read

iv. Small write

5 (continued)

b. (8 points) Given the six-disk array described on the previous page (sectors S0-S14 hold data; P0-P2 hold parity information), also assume the following for this part of the problem:

- The array is configured with RAID 5.
- Large reads/writes take 200 ms, small reads take 50 ms, and small writes take 100 ms.
- Requests are queued in such a manner that up to three consecutive operations may proceed simultaneously if they do not share any disk within the array. Multiple accesses in the same stripe are allowed.
- If two disks, D_x and D_y, are in use, and the access to D_x finishes before the access to D_y, a new operation may start immediately assuming it does not involve D_y.

Determine the time required for this array to complete the following sequence of accesses. For full credit, show all work, including the organization of the array:

1. read S6
2. write S12
3. read S8
4. write S2
5. write S5
6. write S9
7. write S11
8. read S14

The next page contains additional space to solve this problem if necessary.

<u>ADDITIONAL SPACE FOR QUESTION 5b SOLUTION</u>

6. (15 points) ***<u>Coherence protocols</u>***

a. (3 points) In a snooping coherence protocol, what is a relevant transaction?

b. (4 points) Explain the differences between the Fetch and Fetch/Invalidate messages used in a directory coherence protocol and describe a situation in which each message would be used.

c. (3 points) Explain the difference between true and false sharing misses.

6 (continued)

d. (5 points) Say we have a dual-processor system with two nodes, P0 and P1, which share a block at address *A*. The system uses a write-invalidate, directory coherence protocol. Initially, the directory entry for the block reads:

	P0	P1	Dirty
A	1	0	1

If P1 now attempts to read block *A*, what messages are sent between the nodes and directory to ensure P1 gets the most up-to-date block copy and the directory holds the appropriate state? You may want to draw a diagram to support your answer.