Computer Science Comprehensive Examination
Computer Architecture
[60 points]

Please do all of your work on these sheets. Do not do your work in a blue book.

Problem 1: Short Answer [2 points each, 18 points total] (Keep your answer to one line or less)

A. Compared to an 8K direct-mapped cache, what type of misses will an 8K two-way set associative cache will have fewer of? Circle all that apply.

(a) compulsory
(b) conflict
(c) capacity

B. Which instruction set will give better code density: (a) a RISC instruction set with fixed-format 32-bit instructions, or (b) a CISC instruction set with variable-length instructions?

(b) CISC

C. Which mean should be used to combine normalized execution times? Circle all that apply.

(a) Arithmetic mean
(b) Geometric mean
(c) Harmonic mean

D. Adding a cache memory to a system changes which of the following memory parameters? Circle all that apply and denote the direction of change with an up arrow or a down arrow.

(a) Memory latency *(decreases)*
(b) Memory bandwidth *(usually increases)*
(c) Memory address space
(d) Memory reliability

E. With very accurate (>95%) branch prediction, adding an additional decode stage to a pipeline causes very little (<5%) increase in CPI. (True or False?)

True

F. In the steady state, what will be the prediction accuracy of a two-bit branch predictor on the sequence TTNTTNTTN (T=taken, N=not taken)? 6/9 = 67%

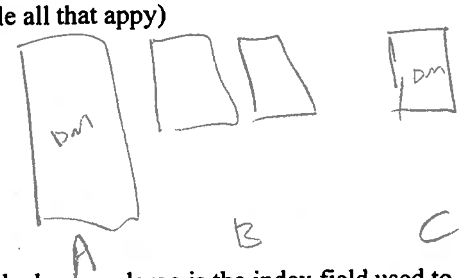
The 2-bit predictor always predicts T and thus is right 2/3 of the time.

G. An instruction for protected subsystem entry must change what two things atomically? (5 words or less).

Control flow and privilege level

H. Suppose that cache A, an 8K-byte direct-mapped cache, cache B, an 8K-byte two-way set associative cache, and cache C, a 4K-byte direct-mapped cache are all referenced with an identical address sequence. Which of the following statements are true: (circle all that appy)

(a) A will contain a superset of the data in B
(b) B will contain a superset of the data in A
(c) A will contain a superset of the data in C
(d) B will contain a superset of the data in C



I. In an 8K-byte two-way set-associative cache with 32-byte blocks, how large is the index field used to address the cache array? (write down the number of bits)

2^{13} bytes, 2^5 blocks, 2 ways => 2^7 blocks per way, so 7-bits are required

Problem 2: Cache Architecture [4 points each, 16 points total]

Consider a computer system with a 32-bit virtual addresses space, byte addressing, and a 4K-byte page size. Each virtual address is divided into a 12-bit offset field that identifies a byte within a page and a 20-bit page field that identifies a page in virtual memory. An address translation unit translates the 20-bit page number into a 20-bit physical page frame number.

A. With this machine organization how large a direct-mapped cache can be realized as a virtually-indexed, physically-tagged cache without any homonym or synonym problems?

4K Bytes. Any larger an the index field of the cache would need to include translated bits from the VA

B. How large a four-way set-associative cache can be realized as a virtually-indexed, physically-tagged cache without any homonym or synonym problems?

16K Bytes. Each of the four "ways" can be up to 4K Bytes in size.

C. Suppose I build a direct-mapped cache ~~eight~~ 16 times the size of your answer to part (A). What can go wrong? Give a short example.

Two synonyms may map to different locations in the cache even though they refer to the same word of main memory.
Suppose virtual addresses 00000000_{16} and 00010000_{16} both map to physical address 00000000. On cache hits, however they will refer to different sets.

D. Can the problem you identified in part (C) be fixed by restricting the mapping of pages to page frames? (Yes or No?). If your answer is "Yes", explain how.

Yes, by restricting each page to be mapped to a page frame so that bits 12-13 of the physical address match bits 12-13 of the virtual address.

0000|1000

0000|0000

12

15

Problem 3: Instruction Issue [16 points]

Consider the following instruction sequence:

```
1       LD      X,R1
2       LD      Y,R2
3       ADD     R1,R2,R3
4       ADD     R3,1,R8
5       LD      Z,R4
6       ADD     R2,R4,R5
7       MUL     R5,3,R6
8       ADD     R6,1,R7
```

You may assume that all operations have two-cycle latency. That is, the result of an operation is available two cycles after that operation enters the first execution stage of the machine. Also assume that there is full bypassing, and that all loads hit in the cache. Hint: you need only consider the execution and memory stages of the pipeline to answer this question.

A. [3 points] How many cycles does this sequence take to execute on a single-issue in-order machine? Measure time from the cycle that the first instruction issues to the cycle in which the result of the last instruction is available for use. Show your work.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	X	X												
2		X	X											
3				X	X									
4						X	X							
5							X	X						
6									X	X				
7											X	X		
8													X	X

14 cycles

B. [3 points] How many cycles does this sequence take to execute on an in-order machine with multiple issue (assume an issue width as wide as you need)? Show your work

	1	2	3	4	5	6	7	8	9	10	11	12
1	X	X										
2	X	X										
3			X	X								
4					X	X						
5					X	X						
6							X	X				
7									X	X		
8											X	X

12 cycles

C. [4 points] How many cycles does this sequence take to execute on a single-issue out-of-order machine? Again, show your work.

	1	2	3	4	5	6	7	8	9	10
1	X	X								
2		X	X							
3				X	X					
4						X	X			
5			X	X						
6					X	X				
7							X	X		
8									X	X

10 Cycles

D. [3 points] How many cycles does this machine take to execute on a multiple-issue out-of-order machine? Show your work.

	1	2	3	4	5	6	7	8
1	X	X						
2	X	X						
3			X	X				
4					X	X		
5	X	X						
6			X	X				
7					X	X		
8							X	X

8 Cycles

E. [3 points] Can you statically reorder the code to give the in-order machines the same performance as the out-of-order machines? If so, show the new ordering.

Yes! One possible ordering is 1,2,5,3,6,4,7,8 or:

```
1      LD      X,R1
2      LD      Y,R2
5      LD      Z,R4
3      ADD     R1,R2,R3
6      ADD     R2,R4,R5
4      ADD     R3,1,R8
7      MUL     R5,3,R6
8      ADD     R6,1,R7
```

Problem 4: Pipeline Microarchitecture and Performance [10 points]

Suppose you have a CPU with a 5 stage pipeline containing the following stages:

F: instruction fetch
R: register read (and instruction decode)
A: execute ALU operations (including determination of branch address and condition)
M: memory load and store
W: write back to register file

On this architecture, conditional branches account for 20% of all instructions, and all other instructions complete in a single cycle. That is, there are no pipeline stalls except for branches.

A. [2 points] With no speculation or prediction, what is the latency (in cycles) of a conditional branch operation on this CPU?

3 cycles

B. [3 points] What is the average CPI (cycles per instruction) for this CPU with no speculation or prediction of branches?

$CPI = (0.8)(1) + (0.2)(3) = 1.4$

C. [2 points] Suppose conditional branches are taken 60% of the time. What is the average CPI for this CPU if we speculatively predict all conditional branches as **not** taken?

$CPI = 0.8(1) + 0.2(0.4(1) + 0.6(3)) = 1.24$

D. [3 Points] Suppose we predict all conditional branches as taken and correctly predict the branch target address in the F stage 50% of the time. What is the average CPI for this case?

$CPI = 0.8(1)+0.2(0.4(3) + 0.6(0.5(1)+0.5(3))) = 0.8+ 0.48 = 1.28$