CS 5513: Computer Architecture
Final
Fall 2015
Total Marks: 100
<u>Time: 2 hour 30 min</u>

Name:
Banner Id:
Alias:

1. [10 pt] Indicate **True** or **False** for the following statements:
   (a) Multilevel caches do not affect miss penalty.
   (b) Fully associative cache does not require index calculation.
   (c) An infinite cache can have only compulsory misses.
   (d) A cache miss is handled by the hardware but a page fault is handled by the operating system.
   (e) Maximum speed up that can be achieved by enhancing 50% of a program execution is 5.
   (f) Structural hazard cannot be eliminated by duplicating resources.
   (g) The ideal CPI for a single issue pipeline is 1.
   (h) A branch delay slot instruction is executed no matter what the outcome of the branch is.
   (i) Performance is proportional to execution time.
   (j) A superscalar processor issues more than one instruction in every clock cycle.

2. [20 pt] CPU Performance and Amdahl's Law:

   (a) [10 pt] Suppose you can apply two enhancements - enh_1 and enh_2. They can be applied to 40% and 25% of a program respectively. If we apply the enhancements one at a time, the overall speed ups of the program are 1.50 and 1.10 respectively.

      i. [3 pt] What is the speed up (i.e., S_1) of enh_1?

      ii. [3 pt] What is the speed up (i.e., S_2) of enh_2?

      iii. [4 pt] What is the overall speed up of the program if we apply both enhancements together?

(b) [2+3+3+2 pt] Suppose we have made the following measurements:

- Frequency of FP operations = 25%
- Average CPI of FP operations = 4.0
- Average CPI of other instructions = 1.33
- Frequency of FPSQR= 2%
- CPI of FPSQR = 20

Assume that the two design alternatives are (a) to decrease the CPI of FPSQR to 2, or (b) to decrease the average CPI of all FP operations to 2.5.

i. What is the CPI of the original design?
ii. What is the CPI of design alternative (a)?
iii. What is the CPI of design alternative (b)?
iv. Which design alternative is better? What is its speed up?

3. [20 pt] Pipeline:
   Assume a five stage MIPS pipeline. Registers are written in first half of a clock cycle but read in the second half. We have separate instruction and data memory. A branch instruction is resolved in the EX stage. Branch is handled by pipeline *freeze* policy. The pipeline implements forwarding policy. Consider this pipeline for question (a) - (e).

   (a) [2 pt] Give an example where a load is immediately followed by an arithmetic instruction that depends on the load. Show the pipeline diagram to calculate the number of stall cycles.

   (b) [2 pt] If 30% of all loads are immediately followed by dependent arithmetic instructions, what is the average number of stall cycles for a load instruction?

   (c) [2 pt] What is the average number of stall cycles for a branch instruction?

   (d) [2 pt] Assume that except for load and branch instructions, the pipeline does not stall for any other instructions. If a program has 35% load instructions, 20% branch instructions, what is the CPI?

(e) [6 pt] Fill up the pipeline diagram.

Ins	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD R2, 0(R4)																				
ADD R1, R2, R3																				
ADD.D R2, R1, R2																				
SD R2, 4(R4)																				
BEQZ R2, L1																				
L1: ADD R1, R2, R4																				

(f) [6 pt] The assumption of the pipeline is the same as in (e) except that there is *no* forwarding. Fill up the pipeline diagram.

Ins	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD R2, 0(R4)																				
ADD R1, R2, R3																				
ADD.D R2, R1, R2																				
SD R2, 4(R4)																				
BEQZ R2, L1																				
L1: ADD R1, R2, R4																				

4. [16 pt] Tomasulo's Algorithm:
   We have the following assumptions.

   - Assume you have 2 Integer, 1 FP Add/Sub, and 1 FP Divider/Multiplier Functional Unit. Integer FU takes 1 cycle, FP Add/Sub takes 6 cycles and FP Divider/Multiplier takes 12 cycles.
   - This is a single issue speculative machine.
   - There is just **one CDB**.
   - Whenever there is a conflict for a resource, assume that the oldest (by program order) of the conflicting instruction gets access, while others are stalled.
   - Assume that the branch is correctly predicted to be not taken.
   - Loads and stores use the integer functional unit to perform effective address calculation during the EX stage. Memory access takes **1 cycle**. Data memory can serve only one access at a time.
   - Except for loads, stores and branches, other instructions free up both the functional unit and reservation station after CDB write stage i.e., if the instruction is at CDB write stage at cycle x, the same functional unit and reservation station can be used by other instruction at cycle x+1. Loads free up the functional unit after the address calculation and the reservation station after the memory read stage. Stores and branches free up both the functional unit and the reservation station after the execution stage.
   - Assume that the number of reservation stations is 3 for Integer, 1 for FP Divider/Multiplier and 2 for FP Add/Sub. Load, store and branch instructions use integer reservation stations.

   Complete Table 1 using Tomasulo's algorithm with the above specifications.

Instruction	Functional Unit	Reservation Station	Issue	Execution	Memory Read	CDB Write	Commit
LD F0, 0(R1)							
ADD.D F2, F0, F6							
SD F0, 8(R1)							
BEQZ F2, **L1**							
ADD.D F4, F6, F6							
DIV.D F6, F4, F2							
SD F4, 16(R1)							
DADD.I R1, R1, #-32							

Table 1: Instruction sequence

5. [19 pt] Memory hierarchy:
A processor has a 4KByte direct mapped L1 cache with 16 Byte cache lines. The cache implements write-back and no write allocate policy. A memory address is 16 bits long. Assume that the cache is initially empty.

   (a) [3 pt] How many bits are used for block offset, index and tag calculation?

   (b) [12 pt] Fill up the following table.

Addr R/W	Binary Addr	Tag	Index	Hit/ Miss	Memory Reference To L2
A101 R	1010 0001 0000 0001				
C123 R	1100 0001 0010 0011				
B105 W	1011 0001 0000 0101				
A107 R	1010 0001 0000 0111				
C122 W	1100 0001 0010 0010				
D20E W	1101 0010 0000 1110				
D12E W	1101 0001 0010 1110				
D120 R	1101 0001 0010 0000				

Table 2: Workout Table

(c) [4 pt] Assume that the system has 16 bit virtual address, 14 bit physical address and page size is 4 KByte. How many entries are there in the page table? A page table is given in Table 3. Translate the virtual addresses in Table 4 into physical addresses.

Virtual Page	Physical Page	Valid
0	0	0
1	2	0
2	2	1
3	3	0
4	2	0
5	0	0
6	1	0
7	0	0
8	1	0
9	2	0
A	3	1
B	2	0
C	0	1
D	1	1
E	1	0
F	3	0

Table 3: Partial page table

Virtual Address	Physical Address
A101	
C122	
D20E	
D102	

Table 4: Address to be translated

6. [15 pt] Short Questions:

(a) [3 pt] What is a precise exception?

(b) [3 pt] Explain pseudo associativity?

(c) [3 pt] How do you determine whether a miss is a conflict miss or a capacity miss?

(d) [3 pt] What is the difference between statically scheduled and dynamically scheduled processor?

(e) [3 pt] How does write buffer work?