CPSC 3300 – Exam 2 Sample Questions

1. Matching. Write the correct term from the list into each blank. (2 pts. each)

control signal	structural hazard	control hazard	speculative execution
out-of-order execution	load-use data hazard	forwarding	IPC (instruction-per-cycle)

(a) _____ when hardware cannot support the combination of instructions we want to execute in the same clock cycle

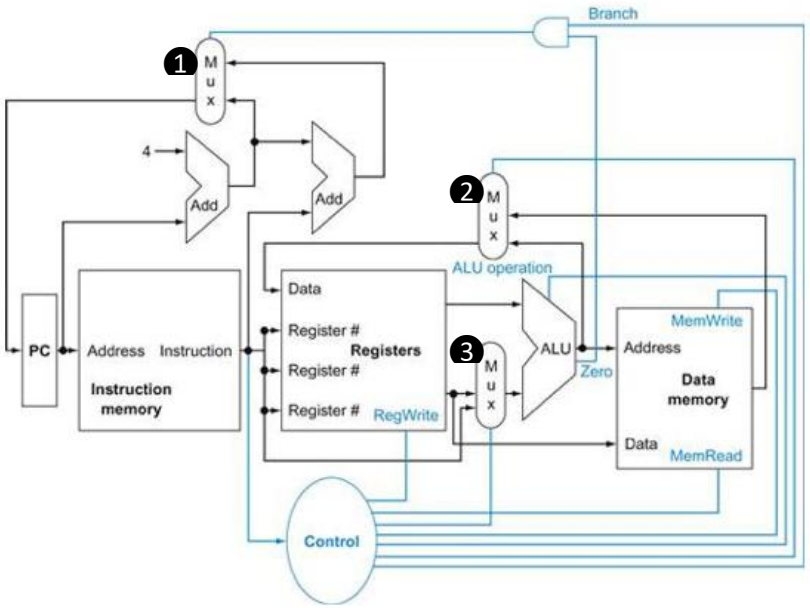
(b) _____ value used for selecting a mux input or selecting the operation of a functional unit

(c) _____ providing a data value to any unit where it is needed after the data value has been produced but before it is available in the register file

(d) _____ allowing an instruction that is control dependent on a branch to execute after the branch direction is predicted and before the branch is resolved

(e) _____ the measure of performance of (advanced) processor pipeline

(f) _____ allowing instructions behind a stalled instruction to proceed to execution



2. Consider the MIPS "and" instruction as implemented on the datapath above (Figure 4.2 from textbook):

and R1, R2, R3 // Reg[1] <- Reg[2] & Reg[3]

Circle the correct value 0 or 1 for the control signals (a-d) and circle whether each of the three muxes (e-g) selects its upper input, lower input, or don't care. For the ALU operation (h) circle one of the function names. (The Zero condition signal will be assumed to be 0.) (8 pts.)

(a) Branch = 0 1

(b) MemRead = 0 1

(e) Mux1 (upper left; output to PC) = upper, lower, don't care

(f) Mux2 (upper middle; output to Data port of Regs) = upper, lower, don't care

(c) MemWrite = 0 1 (g) Mux3 (lower middle; output to bottom leg of ALU) = upper, lower, don't care

(d) RegWrite = 0 1 (h) ALU operation = and, or, add, subtract, set-on-less-than, nor

3. Identify the five stages of the simple pipeline we studied, and explain what each stage does when processing the and instruction from question 2 above. (9 pts.)

and R1, R2, R3 // Reg[1] <-- Reg[2] & Reg[3]

4. Associate each term or statement below with a type of dependency. Circle one or more of RAW, WAR, or WAW. (Destination registers are listed first for add and subtract instructions.) (3 pts. each)

(a) RAW / WAR / WAW true data dependency

(b) RAW / WAR / WAW false data dependency (name dependency)

(c) RAW / WAR / WAW add R3,R1,R2 followed by sub R1,R3,R4

(d) RAW / WAR / WAW add R3,R1,R2 followed by sub R5,R3,R4

(e) RAW / WAR / WAW type of dependency that can cause a load-use penalty

5. Provide short answers for the following questions. (2 pts. each)

(a) Does internal forwarding always eliminate all stall cycles due to data hazards in a pipeline? You can justify/explain your answer by using a simple assembly or pseudo code.

(b) What does the Branch Target Buffer (BTB) store, and how is this information used?

(c) Given a simple 5-stage MIPS pipeline with single issue, what is the ideal IPC and for which scenario?

(d) What hazard does it resolve by using an instruction memory and a data memory in the datapath?

(e) When the MIPS pipeline uses extra hardware to compare two register values in the ID stage instead of using the ALU in the EXE stage, what hazard does it address and what does improve?

(f) Consider a two-bit history for branch prediction. It records the state of the last branch as taken (T) or untaken (U) and predicts the next branch. Assume the two-bit is initialized to Weakly Not Taken (01). Determine the prediction on the following branch trace.

Actual :	T T T T U T T T T U
Prediction:	_ _ _ _ _ _ _ _ _ _

6. Draw the dependency diagram for the following MIPS code. Destination registers are listed first except for the sw (store word) instructions; sw writes into memory rather than a register. (8 pts.)

```
i1:   loop:   lw $1, 0($6)
i2:           lw $2, 4($6)
i3:           beq $1, $2, 2
i4:           addi $6, $6, $4
i5:           bne  $6, zero, loop
```

7. The multi-cycle and pipelined datapaths that we have discussed in class have generally been broken down into 5 steps:

1. Hardware to support an instruction fetch
2. Hardware to support an instruction decode (i.e., a register file read)
3. Hardware to support instruction execution (i.e., the ALU)
4. Hardware to support a memory load or store
5. Hardware to support the write back of the ALU operation back to the register file

Assume that each of the above steps takes the amount of time specified in that table below

Fetch	Decode	Execute	Memory	Write back
305 ps	275 ps	280 ps	305 ps	250 ps

(a) Given the times for the datapath stages listed above, what would the clock period be for the entire datapath? [4pts]

(b) Assuming that N instructions are executed, and all N instructions are add instructions, what is the speedup of a pipelined implementation when compared to a multi-cycle implementation without the pipeline? [4pts]

8. Give the pipeline cycle diagram for the code segment given in question 6 above for the 5-stage pipeline with forwarding. Use arrows to mark the necessary forwarding. (10 pts.)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
loop: lw $1, 0($6)															
lw $2, 4($6)															
beq $1, $2, 2															
addi $6, $6, $4															
bne $6, zero, loop															

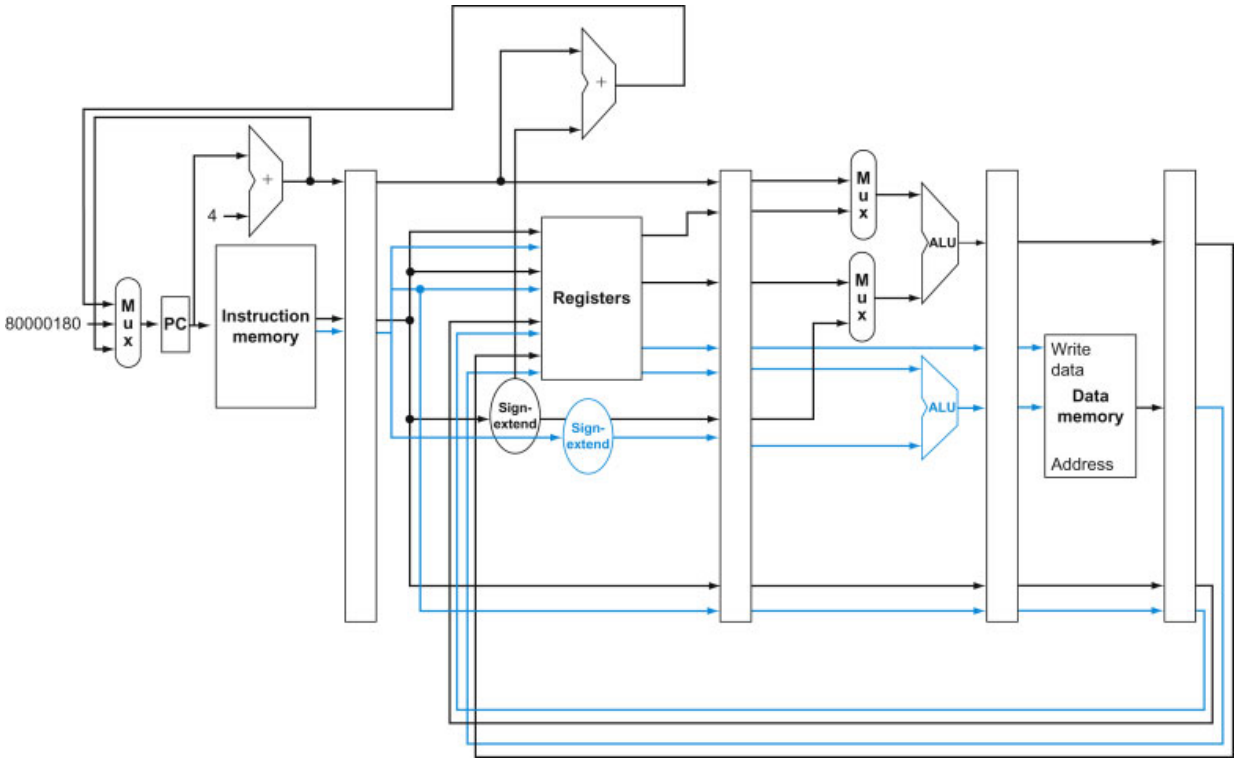
9. The branch CPI penalty is calculated as extra CPI = (branch freq.)*(misprediction freq.)*(mispredict penalty)

(a) Which one of the three terms in the penalty equation will techniques like loop unrolling and predication reduce? (3 pts.)

(b) Which one of the three terms in the CPI formula does dynamic branch prediction attempt to reduce? (3 pts.)

10. 2-issue processors. You are given the following code for a loop:

```
loop:       lw    t0, 0(s1)
            addu  t0, t0, s2
            sw    t0, 0(s1)
            addi  s1, s1, -4
            bne   s1, zero, loop
```



a. [4pt]If the loop exits after executing only two iterations. Draw a pipeline diagram for the given MIPS code on a 2-issue processor shown above. Assume the processor has perfect branch prediction.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

b. [4pts]Rearrange the code to achieve better performance on a 2-issue statically scheduled processor from Figure above. Write the sequence below.

c. [4pts]Repeat step a for the code from b.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14