Computer Engineering 110: Computer Architecture

Midterm Examination 2

Fall 2016

Name: _____Solution_____

Q1	12	
Q2	12	
Q3	18	
Q4	18	
Total	60	

This exam is closed book and closed notes. Personal calculators (four-function calculators only) *are* allowed. Show your work on the attached sheets (front and back) and insert your answer in the space(s) provided. **Please provide details on how you reach a result.** Ask for extra paper sheets if necessary.

You have 70 minutes to complete the exam. This exam is worth 60 points. This exam counts for 15% of your course grade.

Grading instructions for Q1-Q3: first check the total number of cycles, and then look into the details of the pipeline diagram

1. If the pipeline diagram does not show the right number of cycles in total, then take away 2 points (no penalty, if the pipeline diagram shows the correct number of cycles but the student made a stupid mistake by writing down a wrong number in his/her answer. Simply comment on the mistake.)

2. For every instruction that has misplaced pipeline stages, take away 1 points (one instruction 1 points, no matter how many misplaced stages)

3. For every missing or incomplete instruction (in case the student doesn't complete the pipeline diagram), take away 2 points

4. For every wrong bypassing arrow (missing or extra), take away 1 points. Arrows can be placed either in front of, between, or following bypassing stages.

(12 points) 1. Pipelining/bypassing.
Consider the code sequence shown below running on the baseline 5-stage instruction pipeline discussed in class. Each stage takes 1 cycle without hazards. How many cycles will the program take assuming MX, WX, and WM data bypassing is available? Note that the pipeline stalls on data dependencies that cannot be bypassed and also stalls if a structural hazard would occur. Please fill in the provided pipeline diagram (with D in the stall cycles and arrows showing where bypassing is taking place).

```
lw   r2, 40(r6)
add r4, r3, r2
sw   r4, 80(r7)
addi r6, r6, 4
addi r7, r7, 4
addi r1, r1, -1
```

Insn	1	2	3	4	5	6	7	8	9	10	11	12
lw r2, 40(r6)	F	D	X	M	W							
add r4, r3, r2		F	D	D	X	M	W					
sw r4, 80(r7)				F	D	X	M	W				
addi r6, r6, 4					F	D	X	M	W			
addi r7, r7, 4						F	D	X	M	W		
addi r1, r1, -1							F	D	X	M	W	

(12 points) 2. Pipelining/branch.

Now consider the following code, how many cycles will one iteration through the loop take? Please fill in the provided pipeline diagram (with D in the stall cycles and arrows showing where bypassing is taking place).

```
loop:    lw   r2, 40(r6)
         mult r4, r3, r2
         sw   r4, 80(r7)
         addi r6, r6, 4
         addi r7, r7, 4
         addi r1, r1, -1
         bnez r1, loop
```

Assumptions:

- The pipeline stalls on true data dependencies that cannot be bypassed (i.e., all kinds of data bypassing are available).
- The pipeline also stalls if a structural hazard would occur.
- `mult` has 4 pipelined execution stages P1, P2, P3, P4, which takes 4 cycles in total; `add` has a 1-cycle eXecution stage. `mult` and `add` use different ALUs.
- `mult` does not have M stage; all other instructions have M stage.
- bnez comparison is done in the Decode stage.

Insn	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
`lw r2, 40(r6)`	F	D	X	M	W																	
`mult r4, r3, r2`		F	D	D	P1	P2	P3	P4	W													
`sw r4, 80(r7)`				F	D	D	D	X	M	W												
`addi r6, r6, 4`							F	D	X	M	W											
`addi r7, r7, 4`								F	D	X	M	W										
`addi r1, r1, -1`									F	D	X	M	W									
`bnez r1, loop`										F	D	D	X	M	W							

(18 points) 3. Out-of-order execution.

With the code and assumptions in Question 2, how many cycles will five iterations take in a pipeline with out-of-order execution capability? Please fill in the provided pipeline diagram for one iteration only (with D in the stall cycles and arrows showing where bypassing is taking place).

Insn	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
lw r2, 40(r6)	F	D	X	M	W																											
mult r4, r3, r2		F	D	/	P1	P2	P3	P4	W																							
sw r4, 80(r7)			F	D	/	/	/	X	M	W																						
addi r6, r6, 4				F	D	X	M	//	//	//	W																					
addi r7, r7, 4					F	D	X	M	//	//	//	W																				
addi r1, r1, -1						F	D	/	X	M	//	//	W																			
bnez r1, loop							F	D	D	D	X	M	//	W																		
lw r2, 40(r6)											F	D	X	M	//	W																
mult r4, r3, r2												F	D	/	P1	P2	P3	P4	W													
sw r4, 80(r7)													F	D	/	/	/	X	M	W												
addi r6, r6, 4														F	D	X	M	//	//	//	W											
addi r7, r7, 4															F	D	X	M	//	//	//	W										
addi r1, r1, -1																F	D	/	X	M	//	//	W									
bnez r1, loop																	F	D	D	D	X	M	//	W								
lw r2, 40(r6)																					F	D	X	M	//	W						

Assuming no branch prediction available: One iteration 14 cycles; five iterations 5 * 10 cycles +4 = 54 cycles (/: wait in reservation station; //: wait in reorder buffer)

Insn	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
lw r2, 40(r6)	F	D	X	M	W																											
mult r4, r3, r2		F	D	/	P1	P2	P3	P4	W																							
sw r4, 80(r7)			F	D	/	/	/	X	M	W																						
addi r6, r6, 4				F	D	X	M	//	//	//	W																					
addi r7, r7, 4					F	D	X	M	//	//	//	W																				
addi r1, r1, -1						F	D	/	X	M	//	//	W																			
bnez r1, loop							F	D	D	D	X	M	//	W																		
lw r2, 40(r6)										F	D	X	M	//	W																	
mult r4, r3, r2											F	D	/	P1	P2	P3	P4	W														
sw r4, 80(r7)												F	D	/	/	/	X	M	W													
addi r6, r6, 4													F	D	X	M	//	//	//	W												
addi r7, r7, 4														F	D	X	M	//	//	//	W											
addi r1, r1, -1															F	D	/	X	M	//	//	W										
bnez r1, loop																F	D	D	D	X	M	//	W									
lw r2, 40(r6)																			F	D	X	M	//	W								

Assuming branch prediction 100% accuracy: One iteration 14 cycles; five iterations 5 * 9 cycles + 5 = 50 cycles (/: wait in reservation station; //: wait in reorder buffer)

Either answer is correct. The solution shows two iterations. But students only need to show one iteration.

(18 points) 4. Branch prediction.

For the following questions, assume a branch outcome of

T T N N T T N N T T N N …

T = taken, N = not-taken

1) Fill in the table below to determine the prediction accuracy of a 1-bit prediction scheme where the initial prediction state is not taken? Add a * (as shown below) to indicate when the Prediction State incorrectly predicts the branch outcome.

Prediction State	N*	T	T*	N	N*	T	T*	N	N*	T
Branch Outcome	T	T	N	N	T	T	N	N	T	T

What is the prediction accuracy in percentage? 50%

2) Fill in the table below to determine the prediction accuracy of a 2-bit, saturating counter prediction scheme where the initial prediction state is strongly not taken (N)? Add a * to indicate when the Prediction State incorrectly predicts the branch outcome.

Prediction State	N*	n*	t*	n	N*	n*	t*	n	N*	n*
Branch Outcome	T	T	N	N	T	T	N	N	T	T

What is the prediction accuracy in percentage? 25% in steady state, or 20% for the 10 string branch example above. Either answer is correct.

Grading instructions:

1. Each percentage worth 1 point
2. Each prediction state worth 1 point. However, provide them with 1 free point when more than 2 states are wrong. For example, if 2 prediction states are wrong, charge 2 points; if 3 prediction states are wrong, charge 2 points; if 5 prediction states are wrong, only charge 4 points;