CSE 141: Introduction to Computer Architecture Spring 2024

Midterm Examination

Instructor: Jishen Zhao **Due on:** May 3, 2024 @ 11:59PM *(48 points)*

Name: _____
PID: _____
Email: _____

Q1	4	
Q2	6	
Q3	8	
Q4	14	
Q5	16	
Total	**48**	

"While taking this examination, I have not witnessed any wrongdoing, nor have I personally violated any conditions of this course's integrity policy."

If you can honestly attest to the statement above, **write "I excel with Integrity" below and sign.**

If you do not write and sign, the instructor will contact you by e-mail to request you clarify/explain.

Signature:

Instructions:

- This exam is open book and open notes. Show your work and insert your answer in the space(s) provided. Please provide details on how you reach a result unless directed by the question as not to.
- The exam totals 48 points. It counts for 24% of your course grade. Please submit answers to the following questions as a PDF via Gradescope by May 3, 2024 at 11:59 PM. Policy of late submissions is the same homework assignments. Handwritten or typed responses are accepted.

Q1 (4 points): Single choice and short answer.

1A) What is the primary problem when dealing with branches (such as beq and bne in MIPS) in pipeline designs? (Single choice, no further explanation needed)

i. You need to fetch an instruction after the branch before knowing the branch outcome

ii. Calculating if the branch should be taken or not taken complicates the design of the Execute (or ALU) stage

iii. You can have a branch dependent on a value loaded from memory which delays completing the branch

iv. Trick question – branches pose no more problem than any non-branch instruction for a pipeline.

1B) A program P has an instruction count of 10 billion, an average CPI of 3, and runs on a processor with a clock rate of 2 GHZ. What is the execution time for program P? 15s (No further explanation needed)

Q2 (6 points): We have a program with 30 billion instructions that takes 45 seconds to run on a 2GHz machine. It is given that the program consists of 25% branch instructions, and the CPI of branch instructions is 4.

2A) What is the average CPI of the program?
$CPI = (45 * 2 * 10^9)/(30 * 10^9) = 3$

2B) Using a newly developed compiler, the recompiled program now uses 20 billion instructions. It is still composed of 25% branch instructions, but the CPI of the branch instructions has been reduced by a factor of 2 (CPI of the other types of instructions remains the same). What is the expected execution time speedup of the new program over the original program (on the same machine)?
$CPI_{other} = (3 * 30 * 10^9 – 4 * 30 * 10^9 * 25\%)/(30 * 10^9 * 75\%) = 8/3$
$New\ execution\ time = (20 * 25\% * 10^9 * 4/2 + 20 * 75\% * 10^9 * (8/3)) = 50 * 10^9 cycles$
$Original\ executiontime = 45 * 2 * 10^9 = 90 * 10^9 cycles$
$Speedup = 90/50 = 1.8$

Q3 (16 points): Given the following assembly code:

```
L3: addu R7, R4, R3
    lw R7, (R7)
    addu R8, R5, R3
    lw R8, (R8)
    mul R7, R7, R8
    addu R2, R2, R7
    addiu R3, R3, #4
    bne R3, R6, L3
```

Where registers are written as R(reg number), e.g., R1.

In a non-pipelined CPU with the following instruction categories and execution latencies:

Instruction	Latency (Cycles)
add (addu, addiu)	4
load (lw)	8
multiply (mul)	20
branch (bne)	10

3A) (4 points) What is the CPI of the loop for one iteration (the bne instruction included)? There are a total of 8 instructions in the loop. The percentage breakdown by type is:
add: 50%
load: 25%
multiply: 12.5%
branch: 12.5%
So the CPI is: 0.5*4 + 0.25*8 + 0.125*20 + 0.125*10 = 7.75

3B) (4 points) Which of the following optimizations would produce a bigger CPI improvement for one iteration (the bne instruction included)?

i. Implement prefetching to reduce the latency of loads from 8 cycles to 6 cycles.

ii. Implement branch prediction to reduce the latency of branch instructions from 10 cycles to 3 cycles.

CPI for prefetching: 0.5*4 + 0.25*6 + 0.125*20 + 0.125*10 = 7.25
CPI for branch prediction: 0.5*4 + 0.25*8 + 0.125*20 + 0.125*3 = 6.875
So, branch prediction results in more CPI improvement

3C) (8 points) 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).

Assumptions:

- The pipeline is an in-order pipeline, i.e., no out-of-order execution
- 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.
- bne comparison is done in the Decode stage.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L3: addu R7, R4, R3	F	D	X	M	W												
lw R7, (R7)		F	D	X	M	W											
addu R8, R5, R3			F	D	X	M	W										
lw R8, (R8)				F	D	X	M	W									
mul R7, R7, R8					F	D	D	P1	P2	P3	P4	W					
addu R2, R2, R7						F	F	D	D	D	D	X	M	W			
addiu R3, R3, #4								F	F	F	F	D	X	M	W		
bne R3, R6, L3												F	D	D	X	M	W

Q4 (10 points): Branch prediction

For the following questions, assume a branch outcome of

T T N N N T T N N N T T N N N . . .

T = taken, N = not-taken

4A) Fill in the table below to determine the prediction accuracy of a 1-bit prediction scheme, where the initial state is not taken. Use a * in the "Incorrect" row (as shown below) to indicate when the Prediction **incorrectly** predicts the branch outcome, while leaving the entry blank if it is a correct prediction.

Branch Outcome	T	T	N	N	N	T	T	N	N	N	T	T	N	N	N	...
State	N	T	T	N	N	N	T	T	N	N	N	T	T	N	N	...
Prediction	N	T	T	N	N	N	T	T	N	N	N	T	T	N	N	...
Incorrect	/		*			*		*			*		*			

What is the steady state prediction accuracy in percentage? 60%

4B) 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)? Use a * in the "Incorrect" row (as shown below) to indicate when the Prediction **incorrectly** predicts the branch outcome, while leaving the entry blank if it is a correct prediction.

Branch Outcome	T	T	N	N	N	T	T	N	N	N	T	T	N	N	N	...
State	N	n	t	n	N	N	n	r	n	N	N	n	t	n	N	...
Prediction	N	N	T	N	N	N	N	T	N	N	N	N	T	N	N	...
Incorrect	/	*	*			*	*	*			*	*	*			

What is the steady state prediction accuracy in percentage? 40%

Q5 (12 points): Pipelining

Assume the following program is running on the 5-stage in-order pipeline processor shown in class. All registers are initialized to 0. Assuming only WX and WD (register file internal forwarding) forwarding, branches are resolved in **Decode** stage, and branches are always predicted **not-taken**. How many cycles will it take to execute the program, if the branch outcome is **actually taken**? Draw a pipeline diagram (table) to show the details of your work. Use arrows to indicate forwarding.

```
        lw $r6 0($r10)
        lw $r7 0($r11)
        add $r2 $r6 $r7
        beq $r2 $r3 label
        sub $r6 $r8 $r4
        sw $r6 0($r10)
label:  lw $r1 0($r2)
        or $r4 $r2 $r1
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
lw $r6 0($r10)	F	D	X	M	W										
lw $r7 0($r11)		F	D	X	M	W									
add $r2 $r6 $r7			F	D	D	X	M	W							
beq $r2 $r3 label					F	D	D	D	X	M	W				
sub $r6 $r8 $r4								F	&	&	&	&			
sw $r6 0($r10)															
label: lw $r1 0($r2)									F	D	X	M	W		
or $r4 $r2 $r1										F	D	D	X	M	W