CMPE 110: Computer Architecture

Week 6
Midterm2 Review

Jishen Zhao (http://users.soe.ucsc.edu/~jzhao/)

Reminder

- Midterm
  - When: Oct. 31 (Monday) 1:20-2:25pm
  - Where:
    - DRC students: arrangement made by DRC
    - Took Midterm1 in overflow room: E2-215
    - Everybody else: classroom
  - What

Computer Engineering 110: Computer Architecture

Midterm Examination 2

Fall 2016

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.

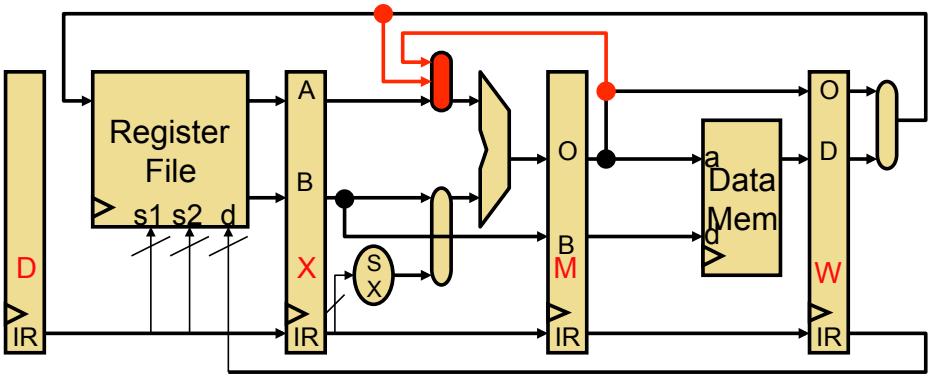
What midterm covers

- Data path and pipelining
  - How to draw pipeline diagram (table), when
    - We have to stall on every hazard
    - Bypassing (forwarding) is available
    - Branch prediction is available
    - Multi-cycle operations
  - What to add to the pipeline datapath to enable
    - Bypassing (forwarding)
  - Branch predictor: 1-bit branch predictor, 2-bit branch predictor
- Out-of-order execution
  - How to draw pipeline diagram, when
    - The pipeline has out-of-order execution capabilities

Pipelining

- 5-stage pipelined datapath: F, D, X, M, W
  - Base CPI = 1
- Multi-cycle operations:
  - X stage takes multiple cycles in a side datapath
  - **Multiply**: the multi-cycle X stage is typically pipelined
  - **Divide**: the multi-cycle X state is typically not pipelined
- Hazards
  - Without any hazard solution, need to stall
  - Data hazards
    - Solution: bypassing (forwarding)
  - Structural hazards
  - Control hazards
    - Solution: branch prediction

Bypassing



- MX bypassing
  - Connect the output of M stage pipeline register to the input of the ALU in X stage
- WX bypassing
  - Connect the out put of W stage pipeline register to the input of the ALU in X stage

Pipelining: Example

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

- 5-stage pipeline
  - `mult` has 4 pipelined cycles in execution stage but no M stage
  - `bnez comparison is done in D stage`
- How many cycles will one iteration through the loop take if
  - 1. In-order pipeline, stalls on all hazards?
  - 2. In-order pipeline with bypassing?
  - 3. In-order pipeline with bypassing, also with branch prediction at F stage and having 100% accuracy?
  - 4. Out-of-order execution with bypassing also branch prediction at F stage with 100% accuracy
- How about 10 iterations?

Branch prediction example

- The outcomes of branch execution for a given branch can be seen as a string of bits, 1 for taken and 0 for not taken. For example, the string for the "end of a loop" conditional branch looks like "1 1 1 . . . 1 1 1 0 1 1 1 . . . 1 1 1 0 1 1 1 . . . 1 1 1 0". Consider, instead, a "diamond" structure, that is code of the form "if condition then S1 else S2." What are the prediction accuracies of a 1-bit scheme and of a 2-bit branch prediction scheme when the conditional expression alternates between *true* and *false*? (Assuming the initial prediction is set to be not-taken.)

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	D	X1	X2	X3	X4	W												
sw r4, 80(r7)					F	D	D	D	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	D	X	M	W				
next iteration lw																F	D	X	M	W		

The next iteration starts at cycle 16.
So ten iterations would take
10 * 15 cycles + 3 cycles = 153 cycles

1.One (the first) iteration 18 cycles; ten iterations would have 180 cycles if every iteration starts after the W stage of branch
ten iterations would have 10 * 15 cycles = 3 cycles = 153 cycles if every iteration starts after the D stage of branch

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	X1	X2	X3	X4	W													
sw r4, 80(r7)				F	D	D	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								
next iteration lw												F	D	X	M	W						

2. One iteration 14 cycles; ten iterations 10*11 cycles + 3cycles = 113 cycles

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	X1	X2	X3	X4	W													
sw r4, 80(r7)				F	D	D	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								
next iteration lw											F	D	X	M	W							

3. One iteration takes 14 cycles; ten iterations take 10*10cycles + 4cycles = 104 cycles

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	/	X1	X2	X3	X4	W													
sw r4, 80(r7)			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									
next iteration lw								F	D	X	M	//	//	W								

4. One iteration 13 cycles; ten iterations 10* 7 cycles + 6 cycles = 76 cycles (/: wait in reservation station; //: wait in reorder

Example Question 2 solution:

When the conditional expression alternates between true and false, then the branch outcome bits are 101010... (or 01010...)

Accuracies of a 1-bit scheme – Assuming initial state is not-taken then the prediction will *always* be incorrect. After the first predicted-not-taken but taken branch, the 1-bit predictor is reset to taken. So it will mispredict again on the next branch, etc as illustrated below (* indicates a mispredicted branch) and continue to mispredict (100% of the time).

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

However, if the initial state is set as taken, then it will predict correctly on the first branch and then always predict incorrectly from that point on. (And, vica-versa for the alternating state of 0 1 0 1 0 ...)

Accuracies of a 2-bit scheme – Assuming the initial state is strongly not-taken (N as in class), then the predictor will be correct 50% of the time (all the not taken branches correctly predicted and all the taken branches incorrectly predicted) as illustrated below.

Prediction	N*	n	N*	n	N*	n	N*	n	N*
Branch Outcome	T	N	T	N	T	N	T	N	T