Lecture 18: Pipelining

- Today's topics:
 - Hazards and instruction scheduling
 - Branch prediction
 - Out-of-order execution

Problem 1







Problem 3





A 7 or 9 stage pipeline



lw \$1, 8(\$2)

add \$4, \$1, \$3



Without bypassing: 4 stalls IF:IF:DE:DE:RR:AL:DM:DM:RW IF: IF :DE:DE:DE:DE:DE :DE:RR:AL:RW

With bypassing: 2 stalls IF:IF:DE:DE:RR:AL:DM:DM:RW IF: IF :DE:DE:DE:DE:RR :AL:RW



- Simple techniques to handle control hazard stalls:
 - for every branch, introduce a stall cycle (note: every 6th instruction is a branch!)
 - assume the branch is not taken and start fetching the next instruction – if the branch is taken, need hardware to cancel the effect of the wrong-path instruction
 - fetch the next instruction (branch delay slot) and execute it anyway – if the instruction turns out to be on the correct path, useful work was done – if the instruction turns out to be on the wrong path, hopefully program state is not lost
 - make a smarter guess and fetch instructions from the expected target

Branch Delay Slots





Source: H&P textbook

Pipeline without Branch Predictor



Pipeline with Branch Predictor



Bimodal Predictor



- For each branch, maintain a 2-bit saturating counter: if the branch is taken: counter = min(3,counter+1) if the branch is not taken: counter = max(0,counter-1) ... sound familiar?
- If (counter >= 2), predict taken, else predict not taken
- The counter attempts to capture the common case for each branch

- Perfect pipelining with no hazards → an instruction completes every cycle (total cycles ~ num instructions)
 → speedup = increase in clock speed = num pipeline stages
- With hazards and stalls, some cycles (= stall time) go by during which no instruction completes, and then the stalled instruction completes
- Total cycles = number of instructions + stall cycles

Multicycle Instructions



^{© 2003} Elsevier Science (USA). All rights reserved.

- Multiple parallel pipelines each pipeline can have a different number of stages
- Instructions can now complete out of order must make sure that writes to a register happen in the correct order

An Out-of-Order Processor Implementation



Completion times	with in-order	with ooo	
ADD R1, R2, R3	5	5	
ADD R4, R1, R2	6	6	
LW R5, 8(R4)	7	7	
ADD R7, R6, R5	9	9	
ADD R8, R7, R5	10	10	
LW R9, 16(R4)	11	7	
ADD R10, R6, R9	13	9	
ADD R11, R10, R9	14	10	

• Final grades will be based on class rank after totaling scores on HW (30%), midterm (30%), and final (40%)

Class rank	Grade	Midterm s	score
30	А	94	
50	A-	90	Since overall performance in
65	B+	87	Since overall performance in
85	В	83	this midlern was pretty good
93	B-	80	
101	C+	77	(potentially B-)
115	С	73 ((potentially C+)
125	C-	69	(potentially C)
129	D+, D, D-	66	(potentially C-)
133	E	39	
No Show	EU		17