4.16 4.27 4.28 CS M151B HW 5

868 16. 1) Pipelined : 350ps

When pipelined, all stages take a single clock cycle so just choose the slowest operation.

Non-pipelined : 1250 ps

When non-pipelined, clock goes through all of the stages

250 + 350 + 150 + 300 + 200 = 1250

2) Pipelined : 1750ps

lw uses all stages of the pipeline : 350 × 5 = 1750ps

Non-pipelined : 1250ps

lw would just be a single-cycle instruction

3) It makes sense to split ID as this is the slowest operation, this will actually decrease the cycle time. The new cycle time would be the latency of the new slowest operation : 300ps.

4) Data memory is only used by lw, sw instructions.

Load : 20% + Store : 15% = 35%.

5) Write Register is only used by ALU, LW instructions.

ALU : 45% + Load 20% = 65%.

874 27. 1)

Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13
ADD	IF	ID	EX	MEM	WB									
LW		X	X	IF	ID	EX	MEM	WB						
LW					IF	ID	EX	MEM	WB					
OR						X	IF	ID	EX	MEM	WB			
SW								X	X	IF	ID	EX	MEM	WB

X : NO OP

s0 - r1
s1 - r2
s3 - r5
s4 - r2

2)
```
ADD $s3 , $s1, $s0
LW $s1, 0($s4)
NOP
LW $s2, 4($s3)
NOP
NOR
OR $s2 , $s3, $s2
NOP
NOP
SW $s2, 0($s3)
```

3) If there is forwarding but no hazard detection unit, the pipeline won't know what or when to pipeline. There may be exceptions or execution interrupts due to control flow mismatches. In the best case, everything just executes normally.

4)

Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13
ADD	IF	ID	EX	MEM	WB									
LW		IF	ID	EX	MEM	WB								
LW			IF	ID	EX	MEM	WB							
OR				IF	ID	EX	MEM	WB						
SW					IF	ID	EX	MEM	WB					

5) We need new input for EX hazards and MEM hazards. For inputs we check the same conditions for EX/MEM hazards that were covered in class. We output control lines that are selected 0 to stall/no-op.

6)

Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13
ADD	IF	ID	EX	MEM	WB									
LW		IF	ID	X	X	EX	MEM	WB						
LW			IF	ID	ID	ID	EX	MEM	WB					
OR				IF	IF	IF	ID	X	EX	MEM	WB			
SW							IF	IF	ID	EX	X	MEM	WB	

28.

R-type	Branch	JAL	LW	SW
40%	25%	5%	25%	5%

Always Taken	Always Not Taken	2-Bit	(Accuracies)
45%	55%	85%	

1) First three execution cycles: IF, ID, EX
Branch outcome is determined in EX
Therefore we lose 3 cycles to mispredicted branch
Branch accounts for 25% of instructions
- Always taken: Correct 45% of time, Incorrect 55% of time

Extra CPI = 3 × 0.55 × 0.25 = 0.4125

2) Extra CPI = 3 × 0.45 × 0.25 = 0.3375

3) Extra CPI = 3 × 0.15 × 0.25 = 0.1125

4) We lose 3 cycles to mispredicted branch
- 2 Bit: Correct 85% of time Incorrect 15% of time

Let's say both branch and ALU instruction have CPI = 1
CPI without conversion: 1 + [3 × 0.15 × 0.25] = 1.1125
CPI with conversion: 1 + [3 × 0.15 × 0.25 × 0.5] = 1.056
↖ halved # of branches now

$\frac{1.1125}{1.056}$ ~~= 1.054~~

1.054 speedup

5) CPI without conversion: 1.1125

CPI with conversion: $1 + [1+3 \times 0.15] \times 0.25 \times 0.5 = 1.181$

$\frac{1.1125}{1.181} = 0.94$ 0.94 speedup

6) 2-Bit: 85% Accuracy

Know that 80% of instructions predicted correctly

Need to find other 20%.

Assume we have N branches in a program

Correctly predicted branches: $N \times 0.85$

Correctly predicted non-loop-back instructions: $N \times 0.05$

Accuracy on non-loop-back branches:

$= \frac{\text{Correctly predicted non-loop-back instructions}}{\text{Remaining instructions to predict}} = \frac{N \times 0.05}{N \times 0.20}$

$0.25 = \boxed{25\% \text{ accuracy}}$