Chapter 4: 4.8. 4.9 11912039 郑鑫毅

4.8 In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

Also, assume that instructions executed by the processor are broken down as follows:

alu	beq	lw	sw
45%	20%	20%	15%

4.8.1 [5] <§4.5> What is the clock cycle time in a pipelined and non-pipelined processor?

in pipelined processor:

clock cycle time = 350 ps

non-pipelined processor:

clock cycle time = (250+350+150+300+200) = 1250 ps

4.8.2 [10] <§4.5> What is the total latency of an LW instruction in a pipelined and non-pipelined processor?

pipelined 350 × 5 = 1750 ps

non-pipelined 1250 ps

4.8.3 [10] <§4.5> If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

- I would break ID stage into two stages each with 175 ps
- new clock cycle: 300 ps

4.8.4 [10] <§4.5> Assuming there are no stalls or hazards, what is the utilization of the data memory?

only lw sw take use of data memory

So the utilization is:

15% + 20% = 35%

4.8.5 [10] <§4.5> Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

lw and alu

20% + 45% = 65%

4.8.6 [30] <§4.5> Instead of a single-cycle organization, we can use a multi-cycle organization where each instruction takes multiple cycles but one instruction finishes before another is fetched. In this organization, an instruction only goes through stages it actually needs (e.g., ST only takes 4 cycles because it does not need the WB stage). Compare clock cycle times and execution times with single-cycle, multi-cycle, and pipelined organization.

	single	multi	pipelined
clock cycle time	1250 ps	250 ps	350 ps.

for execution time: (IC - instruction count)

Single cycle: IC · 1250 ps = 1250 IC

pipelined: IC · 350 + 4 × 350 = 350 (IC+4)

multi cycle: alu. .sw: 4 stage

lw: 5 stage beq: 3 stage.

t = 5. # of lw + 4. # of alu. sw + 3. # of beq

$= [5 \cdot IC \cdot 20\% + 4 \cdot IC \cdot 60\% + 3 \cdot IC \cdot 20\%] \cdot 350$

$= 1400 \cdot IC$

single ↔ pipeline $\frac{1120\,IC}{350(IC+4)} \approx 3.6$

multi ↔ pipeline $\frac{1400\,IC}{350(IC+4)} \approx 4$

4.9 In this exercise, we examine how data dependences affect execution in the basic 5-stage pipeline described in Section 4.5. Problems in this exercise refer to the following sequence of instructions:

```
1 or r1,r2,r3
2 or r2,r1,r4
3 or r1,r1,r2
```

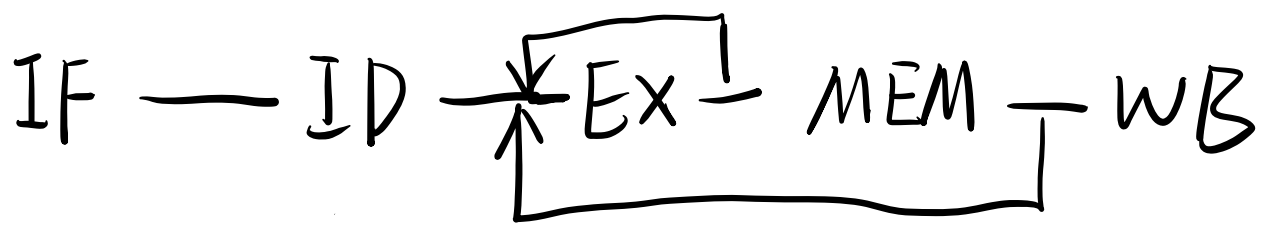
$r_1 = r_2$ or r_3

$r_2 = r_1$ or r_4

$r_1 = r_1$ or r_2

Also, assume the following cycle times for each of the options related to forwarding:

Without Forwarding	With Full Forwarding	With ALU-ALU Forwarding Only
250ps	300ps	290ps



4.9.1 [10] <§4.5> Indicate dependences and their type.

all are EXE - EXE dependency type.

1: r_1 in instr 2 depends on output of instru 1

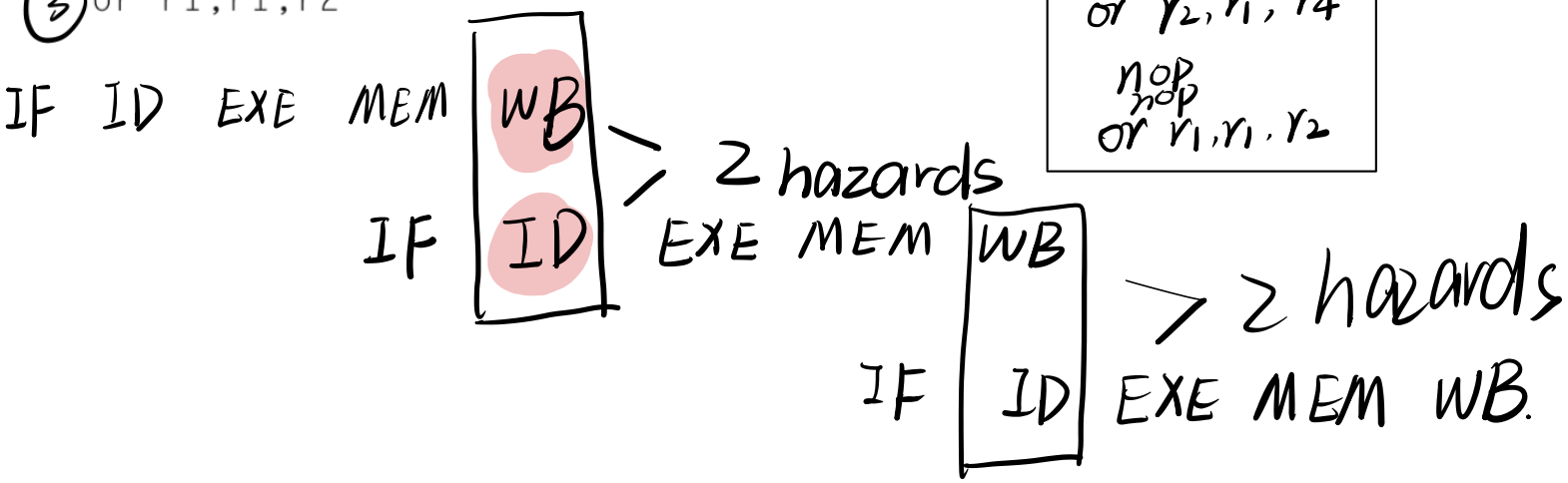
2: r_1 in instr 3 depends on output of instr 1

3: r_2 in instr 3 depends on output of instr 2.

4.9.2 [10] <§4.5> Assume there is no forwarding in this pipelined processor. Indicate hazards and add `nop` instructions to eliminate them.

```
1 or r1,r2,r3
2 or r2,r1,r4
3 or r1,r1,r2
```

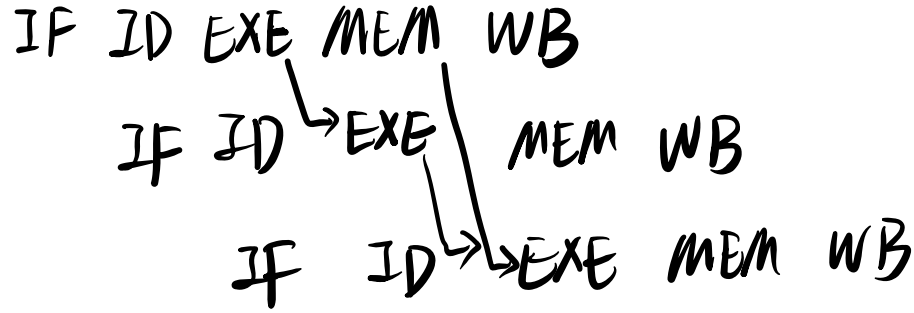
同T周期 既写又读、




```
or r1,r2,r3
nop
nop
or r2,r1,r4
nop
nop
or r1,r1,r2
```

4.9.3 [10] <§4.5> Assume there is full forwarding. Indicate hazards and add `NOP` instructions to eliminate them.

```
1 or r1,r2,r3
2 or r2,r1,r4
3 or r1,r1,r2
```



So there's no hazards and no need to add NOP

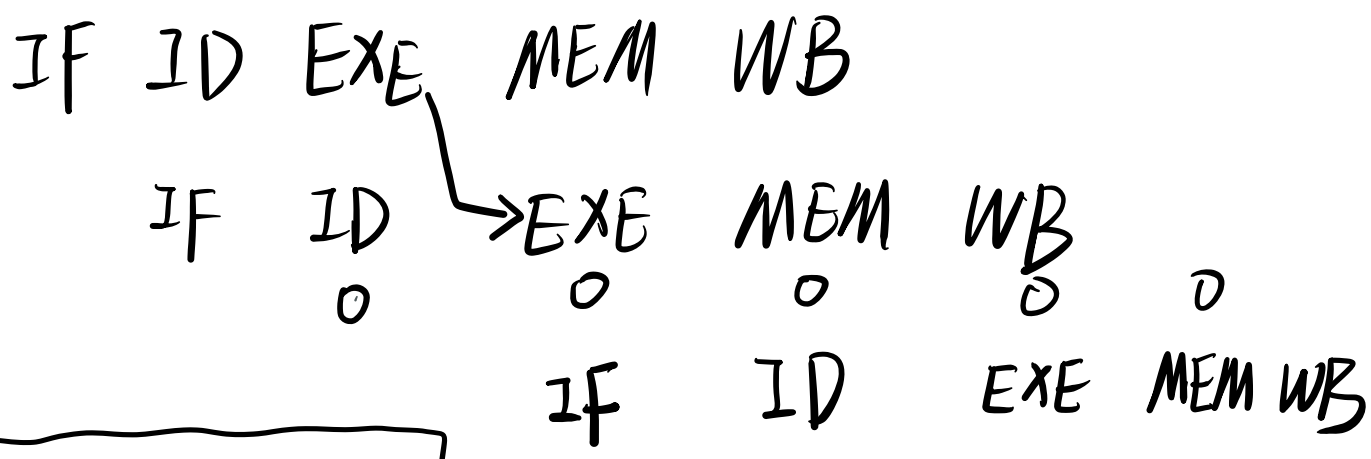
4.9.4 [10] <§4.5> What is the total execution time of this instruction sequence without forwarding and with full forwarding? What is the speedup achieved by adding full forwarding to a pipeline that had no forwarding?

without forwarding: $7 \times 250 + 4 \times 250 = 2750$ ps

with full forwarding: $3 \times 300 + 4 \times 300 = 2100$ ps

speed up $= \frac{2750}{2100} \approx 1.31$

4.9.5 [10] <§4.5> Add `nop` instructions to this code to eliminate hazards if there is ALU-ALU forwarding only (no forwarding from the MEM to the EX stage).




```
or r1, r2, r3
or r2, r1, r4
nop
or r1, r1, r2
```

4.9.6 [10] <§4.5> What is the total execution time of this instruction sequence with only ALU-ALU forwarding? What is the speedup over a no-forwarding pipeline?

$$8 \times 290 = 2320 \text{ ps}$$

$$\text{Speed up} = \frac{2750}{2320} \approx 1.19$$