Computer Organization and Architecture

AFCOE

Neeraj Sharma
2K19/CO/244

Assignment :

Ans:1 $n = 500$ instructions

stages $m = 5$

Probability. instruction being branch $p = 0.3$

$$\text{speed up} = \frac{m*n}{n+m-1+p*m(n-1)}$$

$$= \frac{500 \times 5}{5+499-1+0.3(500*4)}$$

$$= 2.26$$

for $5n = \frac{2500}{504+2000p} \geq 4$

Hence $p \leq \frac{625-504}{2000}$

or $p \leq 0.0605$

∴ number of branch instructions $\leq 0.0605 \times 500$

≤ 32.5

$$5n = \frac{2500}{504+2000p} \geq 5$$

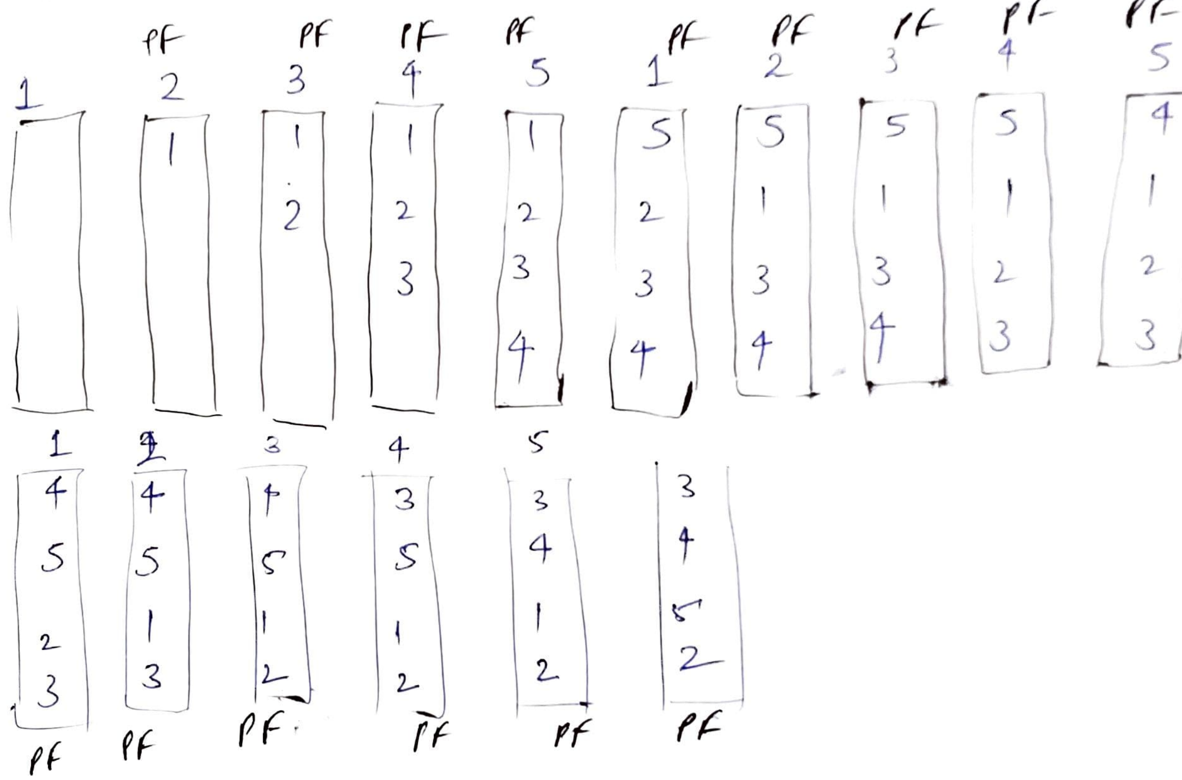
2K19/CO/244

$\Rightarrow p \leq -\frac{4}{2000}$

$\therefore$ speed up of 5 is impossible

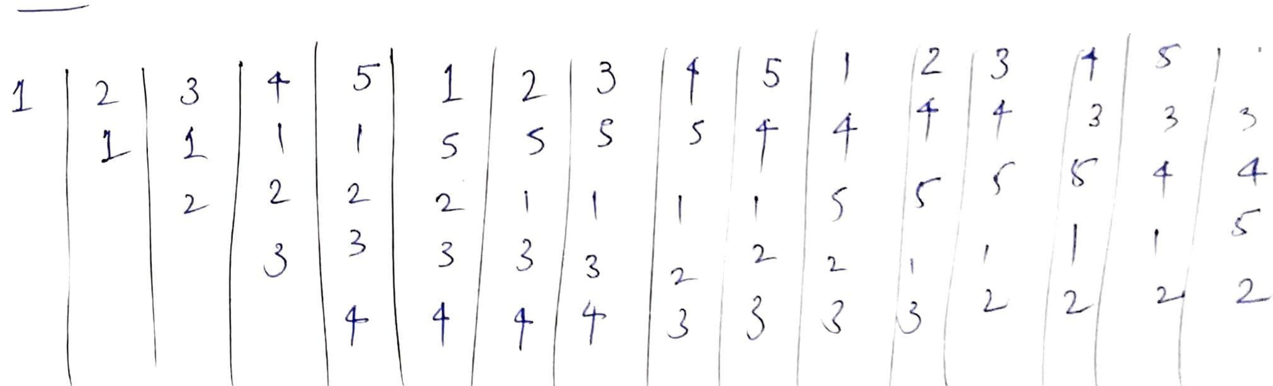
Ans. 2 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5

(a) FIFO:



Hit ratio = 0 %.

(b) LRU:

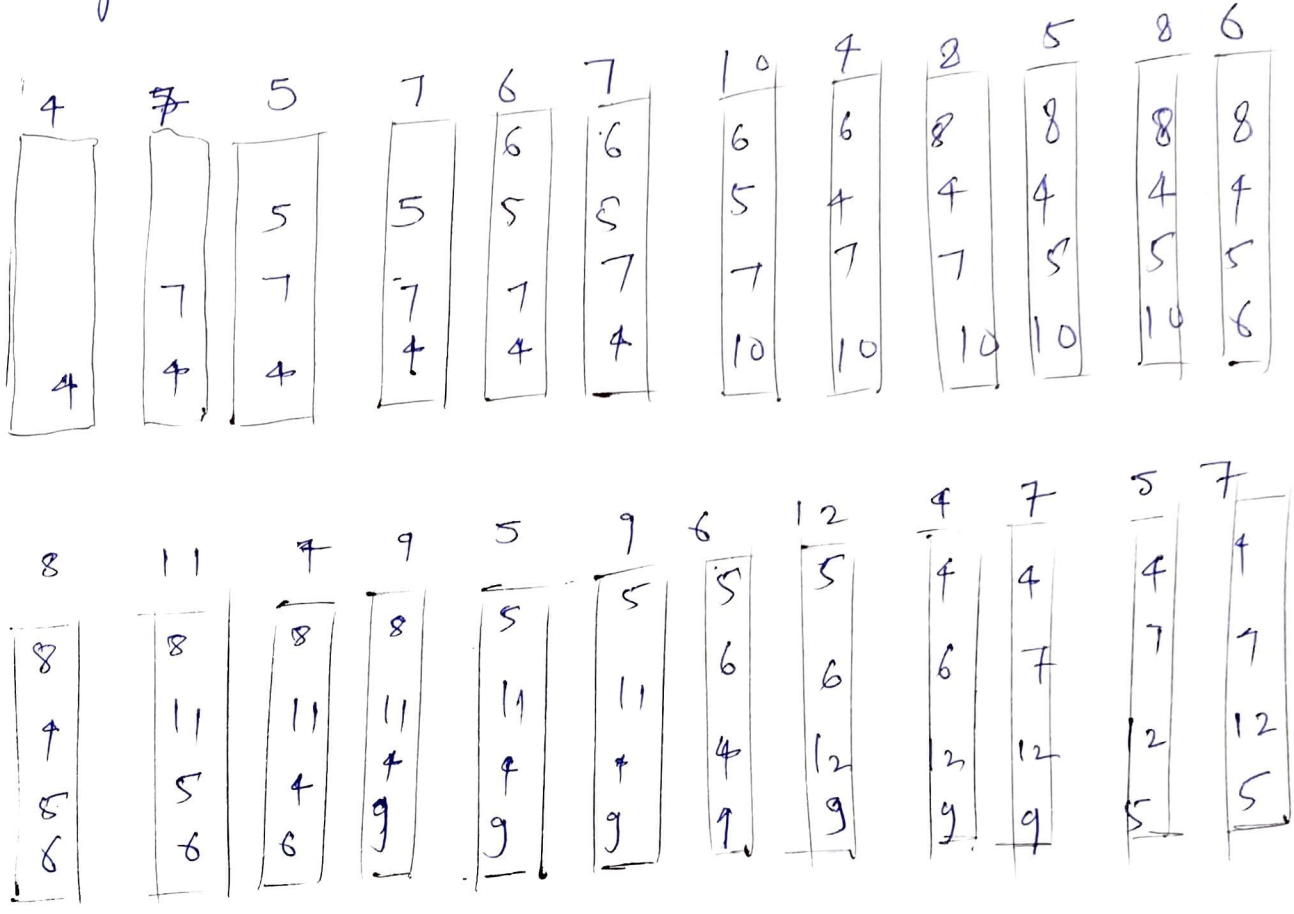


Hit ratio = 0 %.

Ans. 3

4, 7, 5, 7, 6, 7, 10, 4, 8, 5, 6, 8, 11, 4, 9, 5, 9, 6, 9, 12, 4, 7, 5, 7

Using LRU technique



Number of page faults = 17.

Ans. 4 ta = average time

t_{mm} → main memory access time = 100ns

t_c = cache access time = 20ns

n = interleaving factor.

n = interleaving factor

$t_a < 60$

$t_c + \frac{t_{mm}}{n} \leq 60\,ns$

$20 + \frac{100}{n} < 60$

$\Rightarrow n > \frac{100}{40}$

$\Rightarrow n > 2.5$

Since n is usually a power of 2

$\therefore n = 4$

Average access time of resulting system

$= 20 + \frac{100}{4} = 45\,ns$

Ans: 5

t_c = access time of cache memory = 20ns

t_{mm} → access time of semiconductor main memory = 100ns

t_{am} = access time of disk secondary memory = 1ms $= 10^6\,ns$.

Average access time

$$h_c t_c + (1-h_c)[h_m t_m + (1-h_m) t_a]$$

$$= (0.9 \times 20) * (0.1)[(0.95 \times 100) + (0.05 \times 10^6)]$$

$$= 18 + (0.1)(95 + 50000)$$

$$= 5027.5 \text{ ns}.$$

Ans: 6. (a)

	Horizontal Microinstructions		Vertical Microinstructions.
①	It supports longer control word	①	It supports shorter control word
②	It allows higher degree of parallelism.	②	It allows low degree of parallelism.
③	No additional hardware is required.	③	Additional hardwares like decoder are required
④	Faster than vertical microprogrammed control unit	④	Slower than horizontal microprogrammed control unit

(5) It is less flexible	(5) It is more flexible
(6) Energy bit in the control field attaches to a control line	(6) Code is used for each action and the decoder translates this code into individual control signals.
b) Hardwared Control Unit	Microprogrammed Control Unit
(1) It generates the control signals needed for the processor using logic circuits	(1) It generates the control signals with the help of micro-instructions stored in control memory.
(2) It is faster as the required control signals are generated with the help of hardware.	(2) It is slower than the other as micro instructions are used for generating signal here.

2K19/CO/244

3.	Different to modify as the control signals that need to be generated are hard wired	3. Easy to modify as the modification need to be done only at the instruction level
4.	It cannot handle complex instructions as the circuit design for it becomes complex	4. It can handle complex instructions

Ans: 7 Structural Hazards

These hazards arise due to resource conflict in the pipeline. A resource conflict is a situation where more than one instruction tries to access the same resource in the single cycle.

A resource can be a register, memory or ALU.

2K19/Co/244

Example	1	2	3	4
I_1	I_f (Mem)	ID	EX	Mem
		If	ID	EX
I_2			If	ID
I_3				If / Mem
I_4				

- Solution: Resource duplication can be a possible soln. We divide the memory into two individual modules used to store instruction and data seperately.

Control Hazards

These hazards occur during the transfer of control instructions such as BRANCH, CALL etc.

Due to this unwanted instructions are fed to pipeline.

Example:

Data Hazard:

These Hazards occurs when instructions that exhibit data dependence.

Let there be two instructions. I, J.

Read after write occurs.

when J tries to read before I has written

I : $R_2 \leftarrow R_1 + R_3$

J : $R_4 \leftarrow R_2 + R_3$.

write after read

occurs when J tries to write before I reads it

I : $R_2 \leftarrow R_1 + R_3$

J : $R_3 \leftarrow R_4 + R_5$

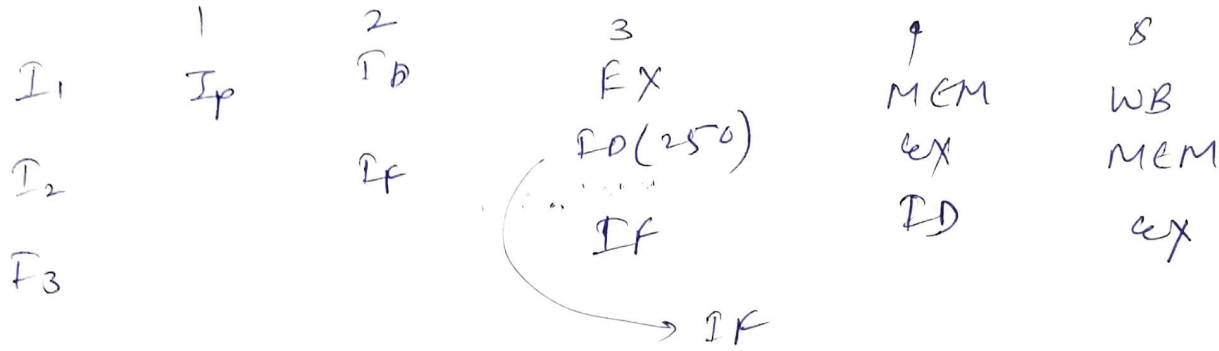
write after write

occurs when J tries to write output before I

I : $R_2 \leftarrow R_1 + R_3$

J : $R_2 \leftarrow R_4 + R_1$.

2K19/Co/249



∴ I_3 is partially executed here.

→ Flushing: It is the worst case scenario either we remove the computed data or we don't work at all

→ N.O.P (No-operation): Meaning Compiler won't execute any instructions

→ Code rearrangement: or delayed load.

Here with smart Compiler we can first execute some instructions which are independent from the current logic

→ Estimation.

Here we can use either 2 stage then/from policy change

Solution :

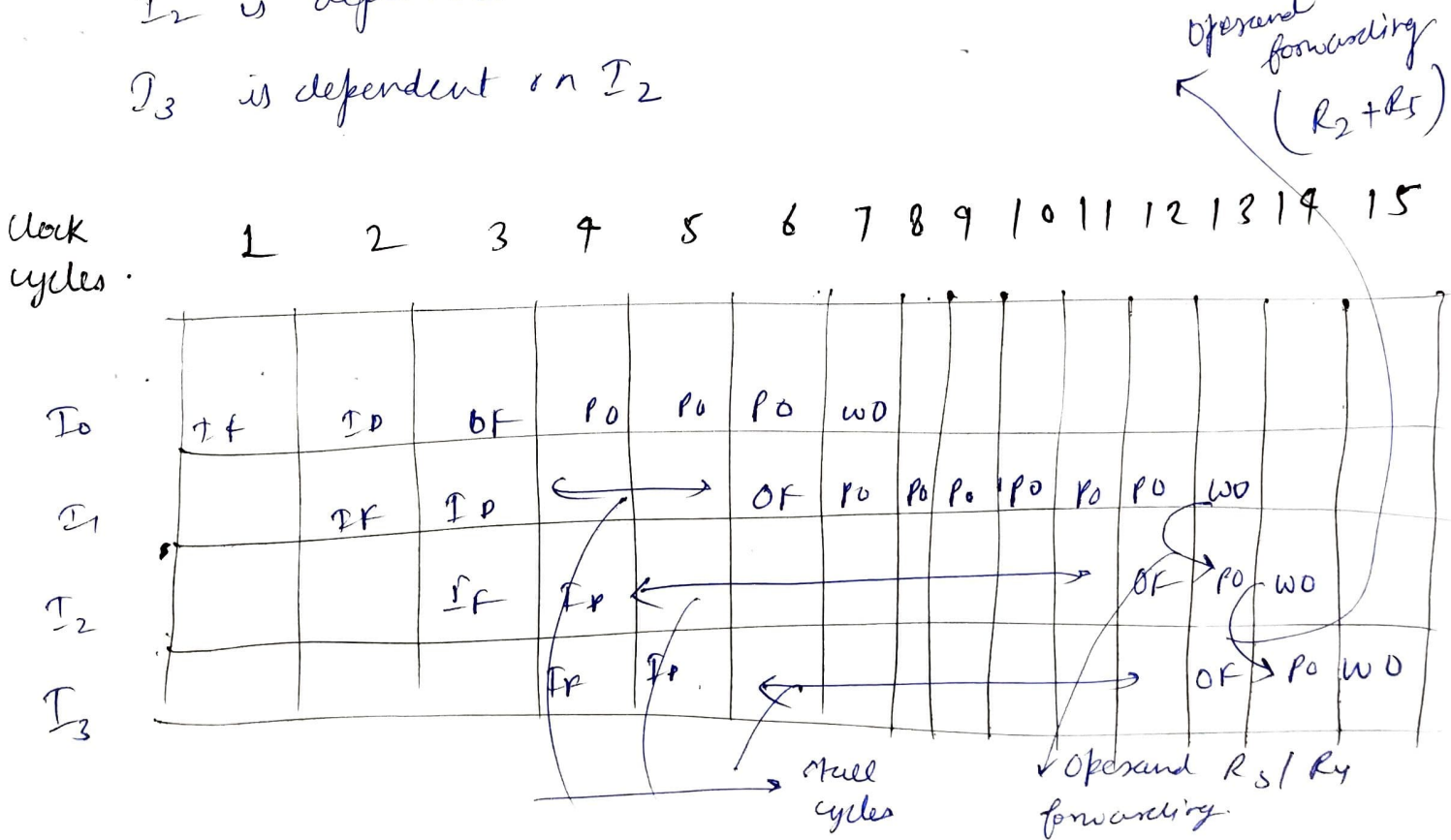
→ We can use code movement or code relocation

→ Also we can use operator forwarding using which we can directly access the result after execution instead of waiting that it gets stored in the memory

Ans: 8 Operand forwarding : In this technique, the value of operand is given to the concerned stage of dependent instruction before it is used.

I_2 is dependent on I_0 and I_1, and

I_3 is dependent on I_2



2K19/CO/244

Here I0 is a MUL operation which takes 3 clock cycles in PO stage and 1 cycle for rest.

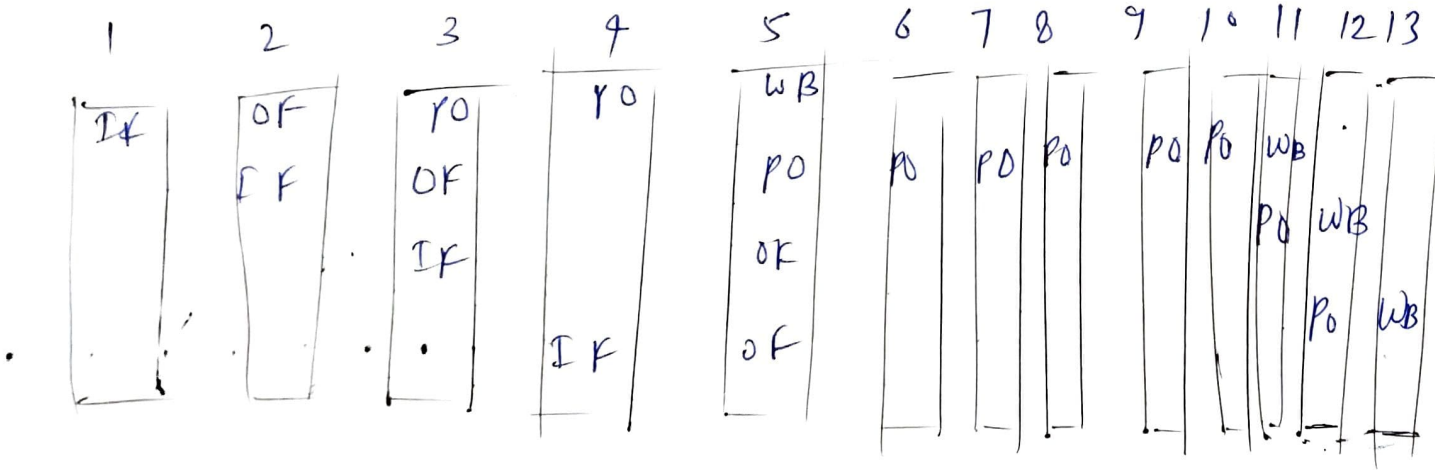
→ I1 is a DIV operation which takes 6 cycles in PO stage, and 1 cycle for rest

→ I2 is an ADD operation which takes cycle of CU for all stages but is a dependent operation.

→ I3 is again uses operand forwarding

Hence, total no. of cycle = 15 option (B).

Ans: 9



Total no of cycles = 13.

2K19/CO/244.

Ans:10, Here each stage takes 1 CPI on an average.

In 1st case 80% take 1 clock & 20% take clocks.

∴ total time

$$P = (0.8 \times 1 + 0.2 + 3) \times 2.2$$

$$= 3.08$$

$$q = (0.8 \times 1 + 0.2 \times 6) \times 1$$

$$= 2$$

$$\frac{P}{q} = 3.08/2 = 1.54$$