CS & IT ENGINEERING

COMPUTER ORGANIZATION AND ARCHITECTURE

Pipeline Processing

Lecture No.- 04

By- Vishvadeep Gothi sir

Recap of Previous Lecture

Topic Pipeline Hazards

Topic Data Hazard Classification

Topics to be Covered

Topic CPI in Pipeline

Topic Classical RISC Pipeline

Topic : CPI Calculation in Pipeline

Without hazard :-

no. of cycles needed to execute n $inst^{ns}$ = $k + n - 1$

1 $inst^{n}$ = $\frac{k+n-1}{n}$ ← CPI

1 $inst^{n}$ in ideal Case = 1 ⇐ (CPI)

Avg. $inst^{n}$ execution time = CPI * tp

with hazard :-

Assume no. of stalls due to hazards = x

$$\text{CPI with hazard} = \frac{k+n-1+x}{n}$$

$$\text{CPI in ideal conditions with hazard} = \frac{n+x}{n} = 1 + \frac{x}{n}$$

[NAT]

#Q. Consider a 5-stage pipeline which is executing a program of 1000 instructions.

Among all instructions 200 instructions cause 2 stall cycles each.

1. Calculate CPI of pipeline?
2. If pipeline cycle time is 3ns then what is average instruction execution time?
3. Calculate CPI of pipeline in ideal conditions?
4. If pipeline cycle time is 3ns then what is average instruction execution time in ideal conditions?

$k = 5$
$n = 1000$

total stalls $(x) = 200 * 2 = 400$

1. $CPI = \frac{5 + 1000 - 1 + 400}{1000} = 1.404$

2. $A.I.E.T. = 1.404 * 3\ ns = 4.212\ ns$

3. $CPI_{ideal} = 1 + \frac{400}{1000} = 1.4$

4. $A.I.E.T._{ideal} = 1.4 * 3\ ns = 4.2\ ns$

from prev. quest^{n} $\frac{200}{1000} = 0.2$ or 20% inst^{ns} cause stalls

$$CPI_{ideal} = 1 + \frac{200 * 2}{1000} = 1 + \underline{0.2} * \underline{2} = 1.4$$

$\underline{2}$ → no. of stalls per inst^{ns}

$\underline{0.2}$ ↑ % of inst^{ns} cause stalls

[NAT]

#Q. Consider a 5-stage instruction pipeline, where all stages take equal delay. When an application is executing on this 5-stage pipeline, consider 20% of the instructions incur 3 pipeline stall cycles is.

1. Average CPI for pipeline? $1 + 0.2 * 3 = 1.6$
2. Average instruction execution time for pipeline? $1.6 * t_p$
3. The speedup of pipeline achieved with respect to non-pipeline?

(use ideal case because total no. of inst^{ns} (n) not given)

$$S_{ideal} = \frac{t_n}{t_p}$$

without hazard

$$S_{ideal\ with\ hazard} = \frac{t_n}{CPI * t_p}$$

assume each stage takes x delay.

t_n = sum of all seg. delays

$= 5x$

$t_p = max(x, x, x, x, x)$

$= x$

Speed up = $\frac{5x}{1.6 * x} = 3.125$

[NAT]

#Q. Consider a 5-stage instruction pipeline, where stages take delays 5ns, 4ns, 6ns, 4ns and 5ns respectively. The pipeline is used to execute a program in which 25% instructions cause 4 stalls due to hazard. The average instruction execution time in the pipeline is __12__ ns?

$CPI = 1 + 0.24 * 4 = 2$

$t_p = \max(5,4,6,4,5) = 6ns$

avg. inst^n execut^n time $= 2 * 6ns$
$= 12\ ns$

$t_n = 5+4+6+4+5 = 24\ ns$

Speed of pipeline $= \frac{24ns}{12ns} = 2$

[NAT]

#Q. A CPU has a five-stage pipeline and runs at 1 GHz frequency. Instruction fetch happens in the first stage of the pipeline. A conditional branch instruction computes the target address and evaluates the condition in the third stage of the pipeline. The processor stops fetching new instructions following a conditional branch until the branch outcome is known. A program executes 10^9 instructions out of which 20% are conditional branches. If each instruction takes one cycle to complete on average, the total execution time of the program is 1.4 seconds?

$k = 5$

clock freq. $= 1\,GHz$

cycle time $(t_p) = \frac{1}{1\,GHz} = 1ns$

stalls due to each branch inst^n $= 3 - 1 = 2$

$CPI_{avg} = 1 + 0.2 * 2 = 1.4$

Avg. $inst^n$ $execut^n$ time $= 1.4 * 1ns = 1.4ns$

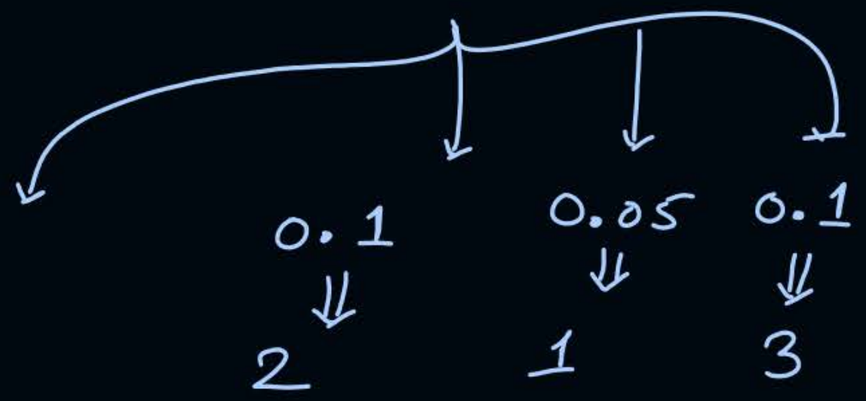
Prog. execution time $= 10^9 * 1.4ns$

$= 1.4$ seconds

Ques) among all inst^ns 10% cause 2 stalls each due to branch hazard

5% —"— 1 stall each —"— data —"—

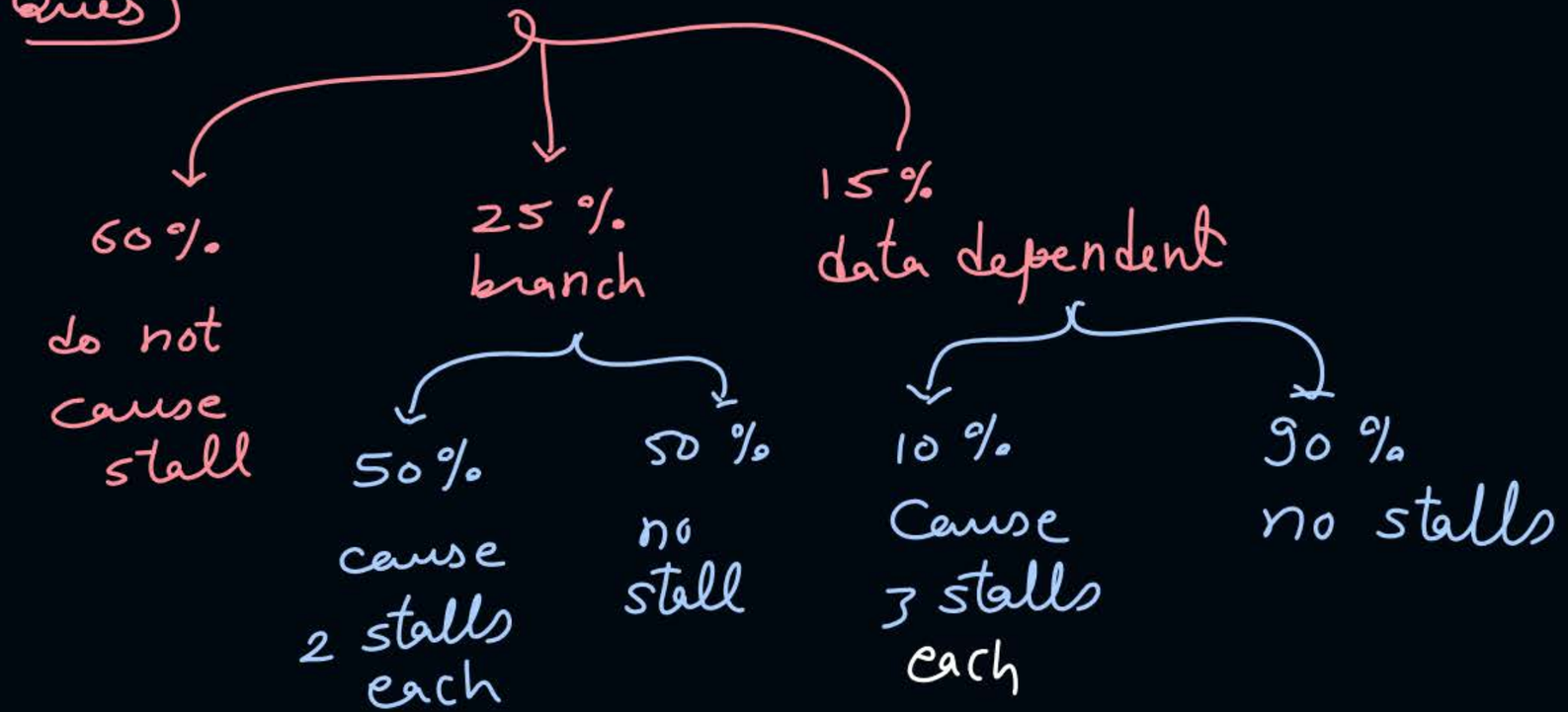
10% cause 3 —"— —"— structural —"—



$$CPI = 1 + 0.1*2 + 0.05*1 + 0.1*3$$

$$= 1.55$$

Ques)



$$CPI = 1 + 0.25 * 0.5 * 2 + 0.15 * 0.1 * 3$$

$$= 1.295$$

[NAT]

GATE - PYQ 2020

#Q. Consider a non-pipelined processor operating at 2.5 GHz. It takes 5 clock cycles to complete an instruction. You are going to make a 5- stage pipeline out of this processor. Overheads associated with pipelining force you to operate the pipelined processor at 2 GHz. In a given program, assume that 30% are memory instructions, 60% are ALU instructions and the rest are branch instructions. 5% of the memory instructions cause stalls of 50 clock cycles each due to cache misses and 50% of the branch instructions cause stalls of 2 cycles each. Assume that there are no stalls associated with the execution of ALU instructions. For this program, the speedup achieved by the pipelined processor over the non-pipelined processor (round off to 2 decimal places) is 2.16 .

non-pipeline :-

clock = 2.5 GHz

cycle time = $\frac{1}{2.5\ GHz}$ = 0.4 ns

t_n = 5 * 0.4 ns

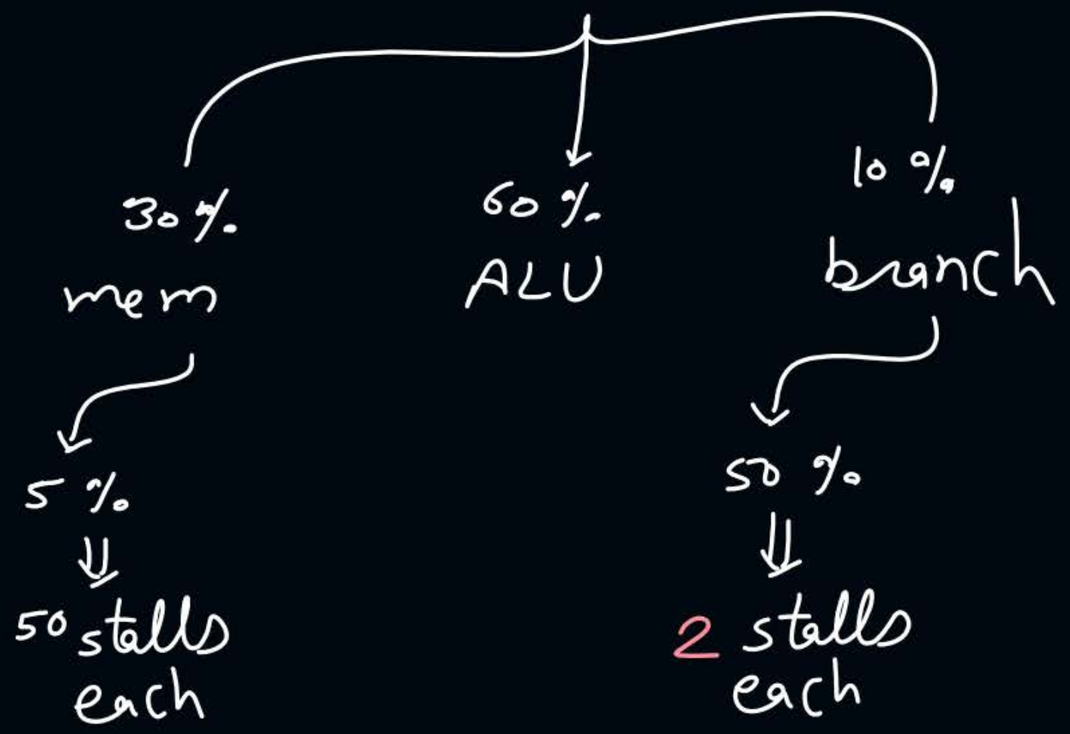
= 2 ns

pipeline :-

k = 5

clock = 2 GHz

cycle time (t_p) = $\frac{1}{2}$ = 0.5 ns



$$CPI = 1 + 0.3 * 0.05 * 50 + 0.1 * 0.5 * 2$$

$$= 1.85$$

$$\text{Speed up} = \frac{t_n}{CPI * t_p} = \frac{2ns}{1.85 * 0.5ns}$$

$$= 2.16$$

[NAT]

S1 S2 S3 S4 S5

— — — — S1

#Q. An instruction pipeline has five stages where each stage takes 2 nanoseconds, and all instructions use all five stages. Branch instructions are not overlapped, i.e., the instruction after the branch is not fetched till the branch instruction is completed. Under ideal conditions.

1. Calculate the average instruction execution time assuming that 20% of all instruction executed are branch instructions. Ignore the fact that some branch instructions may be conditional. 3.6 ns

2. If a branch instruction is a conditional branch instruction, the branch need not be taken. If the branch is not taken, the following instructions can be overlapped. When 80% of all branch instructions are conditional branch instructions, and 50% of the conditional branch instructions are such that the branch is taken, calculate the average instruction execution time? 2.96 ns

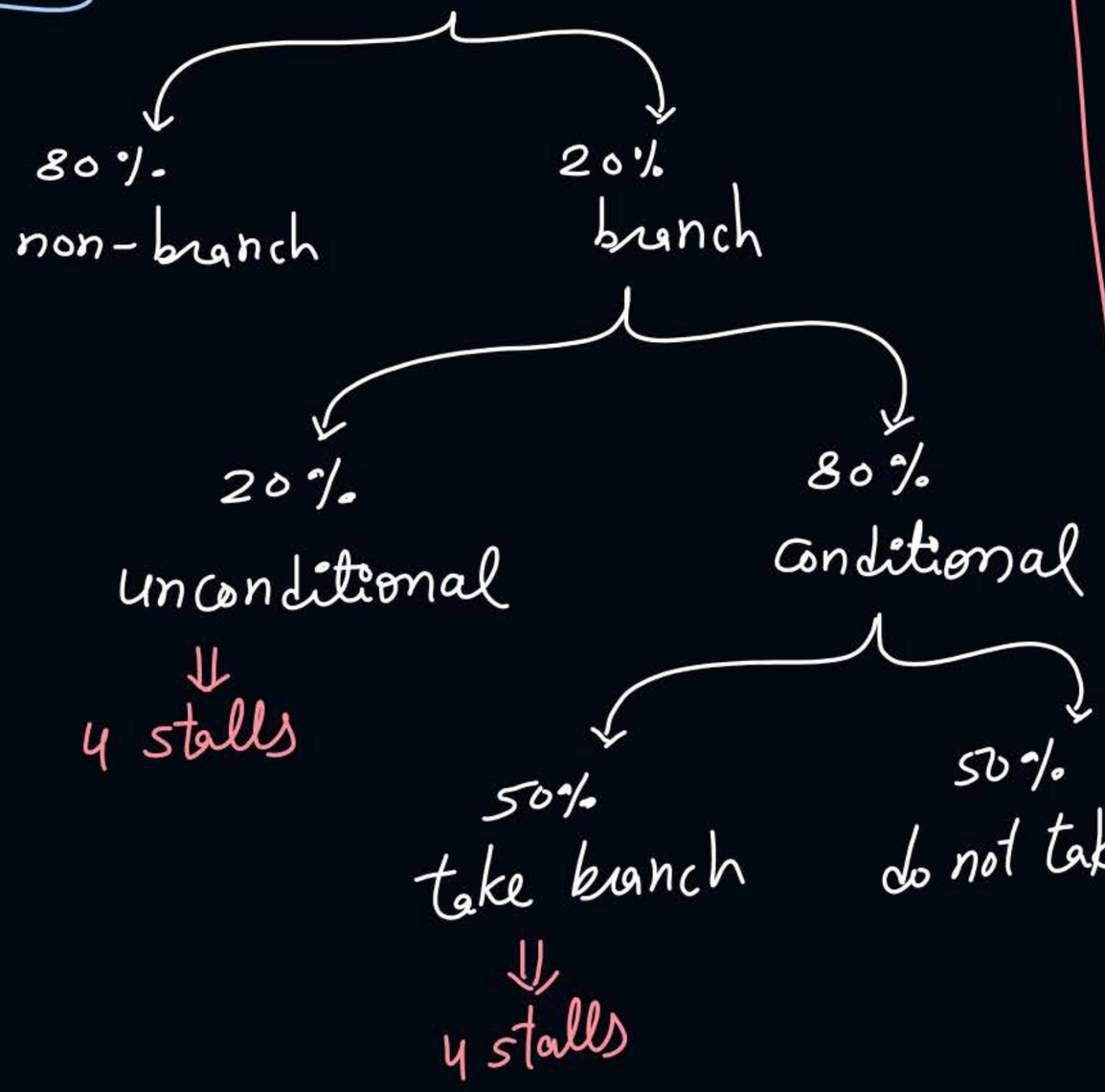
$t_p = 2ns$

stalls due to each branch $inst^n = 5 - 1 = 4$

1. $CPI = 1 + 0.2 * 4 = 1.8$

$A.I.E.T. = 1.8 * 2 = 3.6\ ns$

2.



$$CPI = 1 + 0.2 * 0.2 * 4$$

$$+ 0.2 * 0.8 * 0.5 * 4$$

$$= 1.48$$

$$A.I.E.T. = 1.48 * 2\,ns$$

$$= 2.96\,ns$$

[NAT]

#Q. A processor X_1 operating at 2 GHz has a standard 5-stage RISC instruction pipeline having a base CPI (cycles per instruction) of one without any pipeline hazards. For a given program P that has 30% branch instructions, control hazards incur 2 cycles stall for every branch. A new version of the processor X_2 operating at same clock frequency has an additional branch predictor unit (BPU) that completely eliminates stalls for correctly predicted branches. There is neither any savings nor any additional stalls for wrong predictions. There are no structural hazards and data hazards for X_1 and X_2. If the BPU has a prediction accuracy of 80%, the speed up (rounded off to two decimal places) obtained by X_2 over X_1 in executing P is

X_1 :-

1 cycle time $= \frac{1}{2} = 0.5$ ns

CPI $= 1 + 0.3 * 2 = 1.6$

A.I.E.T. $= 1.6 * 0.5 = 0.8$ ns

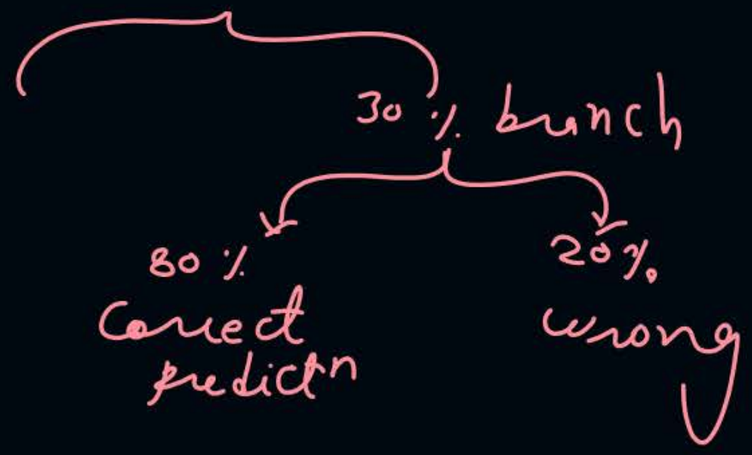
X_2 :-

cycle time $= \frac{1}{2} = 0.5$ ns

CPI for X_2 = 1 + 0.3 * 0.2 * 2

= 1.12

A.I.E.T. = 1.12 * 0.5 = 0.56 ns



$$\text{Speed up} = \frac{\text{Avg. instn execut}^{n} \text{ time in } X1}{\text{A. I. E. T. in } X_2}$$

$$= \frac{0.8}{0.56} = 1.428 = 1.43 \text{ Ans.}$$

Topic : Classic RISC Pipeline

1. IF — Instⁿ fetch
2. ID — Instⁿ decode
3. EX — Execution in ALU
4. MEM — memory Access
5. WB — write back

Topic : Classic RISC Pipeline for Computation

type instns (ALU operatns)

1. IF — Instn fetch & PC increment
2. ID — Instn decode & operand fetch from register → register read
3. EX — ALU operation
4. MEM — Memory Access ⇒ not used
5. WB — write back result to register → Register write

Topic : Classic RISC Pipeline for Load

Register ← Mem

1. IF — Instⁿ fetch & PC increment
2. ID — Decode
3. EX — Nothing
4. MEM — Memory Read
5. WB — write in Reg.

Topic : Classic RISC Pipeline for Store

Mem. ← Reg.

1. IF — Instⁿ fetch & PC increment
2. ID — Instⁿ decode and register read
3. EX — Nothing
4. MEM — Memory write
5. WB — Nothing

Topic : Classic RISC Pipeline for Branch

1. IF — Instⁿ fetch & PC increment
2. ID — Instⁿ decode | Target address calculation | Condition evaluation | PC updation by Target
3. EX
4. MEM
5. WB

} nothing

stalls due to each branch instⁿ = 2 − 1 = 1

2nd phase (ID) evaluates Conditⁿ & provides branch result.

[NAT] Ans = 1.88 GATE – 2021

#Q. Consider a pipelined processor with 5 stages, Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory Access (MEM), and Write Back (WB). Each stage of the pipeline, except the EX-stage, takes one cycle. Assume that the ID stage merely decodes the instruction and the register read is performed in the EX-stage. The EX-stage takes one cycle for ADD instruction and two cycles for MUL instruction. Ignore pipeline register latencies.

Consider the following sequence of 8 instructions:

ADD, MUL, ADD, MUL, ADD, MUL, ADD, MUL

Assume that every MUL instruction is data-dependent on the ADD instruction just before it and every ADD instruction (except the first ADD) is data-dependent on the MUL instruction just before it. The Speedup is defined as follows:

$$Speed\ up = \frac{Execution\ time\ without\ operand\ forwarding}{Execution\ time\ with\ operand\ forwarding} = \frac{30}{16} = 1.875 = 1.88$$

The Speedup achieved in executing the given instruction sequence on the pipelined processor (rounded to 2 decimal places) is 1.88.

IF — 1

ID — 1

EX — 1

MEM — 1

WB — 1

ADD	MUL
1	2

⇓

1 stall each

with operand forwarding :-

no. of cycles w/o hazard = 5+8-1 = 12

stalls due to structural hazard = 4*1 = 4

16

n = 8 instns

k = 5

stalls for each data dependency = WB phase no − OF phase no = 5−3 = 2

without operand forwarding :-

no. of cycles without hazard = 5+8-1 = 12

stalls due to structural hazard = 4*1 = 4

—"— data —"— = 7*2 = 14

Total = 30

2 mins Summary

Topic CPI in Pipeline

Topic Classical RISC Pipeline

Happy Learning

THANK - YOU