COMPUTER ARCHITECTURE

HOMEWORK - 3.

1.a) Clock cycle time of a non-pipelined processor
300ps + 450ps + 200ps + 400ps + 200ps = 1550 ps.
Clock Cycle time of a pipelined processor : Clock cycle time of the longest block (Instruction Decode) = 450ps.

b) For a load word instruction: All the stages of the processor is used. Latency of a non-pipelined processor = 1550 ps.
Pipelining doesn't improve latency of a processor just the overall throughput.
latency in a pipelined processor = 450 x 5 stages = 2250.

c) We could split the largest stage into 2 parts with latency = 225ps each.
The new clock cycle time would be ~~the~~ 400ps (MEM) Stage

d) The load word & store word instructions use the data memory which accounts for 25% of CPU utilization.

e) The ALU & load word could write back to the write-register in the registers unit, which accounts for 75% of CPU utilization.

f) Single cyt Non-pipelined processors take one cycle to execute each instruction.

Pipelining increases instruction throughput.
A lw would complete in 5 cycles
A sw would complete in 4 cycles
An ALU would comple in 4 cycles
A Branch would complete in 3 cycles

Speed up of the pipeline would be
(.6x4 + .15x3 + .15x5 + .1x4) = 4.

2. a) Data Dependencies in the second & third instruction. R_1 is used before it is written back to the register. R_2 is used before it is computed in the third instruction.
No Control Hazards
* Structural hazard is writing to reg in inst 1 & reading in 3
b) The hazards present are Data Hazards.
or r1, r2, r3
nop.
nop.
.
or r_2, r_1, r_4
nop.
nop.
.
or r_1, r_1, r_2.

c) If there is full forwarding, no nop instructions are required.

d) Total execution time without forwarding
300 × 11 cycles = 3300 ps.
Total execution time with full-forwarding
350 × 7 cycles = 2450.
Speedup achieved = 11 cycles / 7 cycles = 1.34

e) No nops required for AW-AW forwarding also

f) Total execution time with AW-AW forwarding = 340 ps × 7
= 2380
Speedup compared to a no forwarding pipeline = 3300/2380
= 1.386

3.a) The first instruction that consumes the result after it is produced will require 2 stalls & the second one will require one stall.
Total cycles per instruction = 1 + .05×2 + .15×2 + .05×1 + .10×1
.10×2 = 1 + 0.75 + .15
= 1.9.

No. of cycles per instruction lost due to data hazards = .9.

fraction of cycles lost due to data hazards = 0.9/1.9
= 0.47.

b) If there is full forwarding all EX/MEM stage results can be forwarded without stalls. The WB stage results require stalls.

CPI with stalls = 1+ .15×1 = 1.15

clocks lost due to hazards = 0.15

fraction of clocks lost = 0.15/1.15 = 0.013

c) for EX/MEM no of stall cycles =

.15 + .05 + .2 + .15 = 0.55

for MEM/WB no of stall cycles =

0.05 + 0.15 + 0.1 + 0.15 = 0.45

Since MEM/WB results in lesser number of clock cycles it is better.

d) No forwarding

CPI = 1.9

Clock cycle time = 170 ps.

Time taken to execute instruction = 1.9 × 170

= 323 ps.

full forwarding

CPI = 1.15

Clock cycle time = 170 ps

Time taken to execute instruction = 195.5 ps.

Speed up achieved = 323/195.5 = 1.652 = 65.2% speedup

e) If all data hazards were eliminated.
CPI = 1.
Clock cycle time = 180 ps.
Additional speedup achieved = 191.5/180 = 1.086
= 8.6%.
This is 8.6% more than full forwarding.

f) Number of Stall cycles for Ex/MEM = 0.55
CPI = 1.55.
Clock cycle time = 170.
Latency = 170 × 1.55 → ①
for NEM/WB = 0.45
Clock cycle time = 170
Latency = 0.45 × 170. → ②

Seeing ① & ② we can see that NEM/WB results in a shorter time per instruction.

4.a) CPI due to mispredicted branches = percentage of branches × misprediction percentage × penalty
= 0.2 × 0.6 × 2 = 0.24 additional cycles required.

b) Additional CPI = 0.2 × 0.4 × 2 = 0.16 additional cycles.

c) Additional CPI = 0.2 x 0.15 x 2 = 0.06 additional cycles.

f) Let number of instructions executed in the program be x.

Correctly predicted $x \times 0.85$

Correctly predicted complex branches $= x \times 0.05$

Accuracy on complex branches $= (x \times 0.05)/(x \times 0.2)$

$= .25$

25% of instructions are predicted correctly.

d) Only half the branch are branch statements.

Additional CPI = $1 + 0.2 \times 0.15 \times 2 \times 0.5 = 1.03$

Speedup = $1.06/1.03 = 1.029$.

e) $(1 + 2(0.5)) \times (0.2)(0.5) = 0.052 = 1.052$ of ALU

Speed up = $1.06/1.052 = 1.007$

5. a) Accuracy of Always taken = $3/5 = 0.6$

Accuracy of Always not taken = $2/5 = 0.4$

b) Predictor result : 0 1 0 1 0

So it predicts : NT NT NT NT NT

Accuracy = $2/5 = 0.4$.

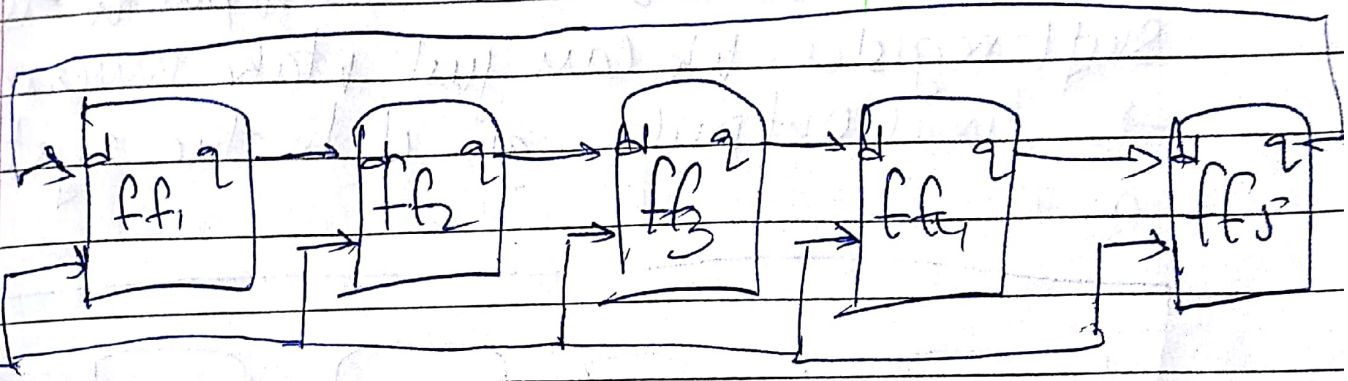
c) If this pattern is repeated forever.

predictor: 0 1 0 1 0 | 1 0 1 0 1 | 0 1 0 1 0 | 1 0 1 0 1 | ... steady state

steady state pred: T T T T T

Accuracy = 3/5 = 0.6

d) We can have a 5 bit shift register to store the correct branch. So, if the pattern is repeated forever. The correct branch will be taken always.



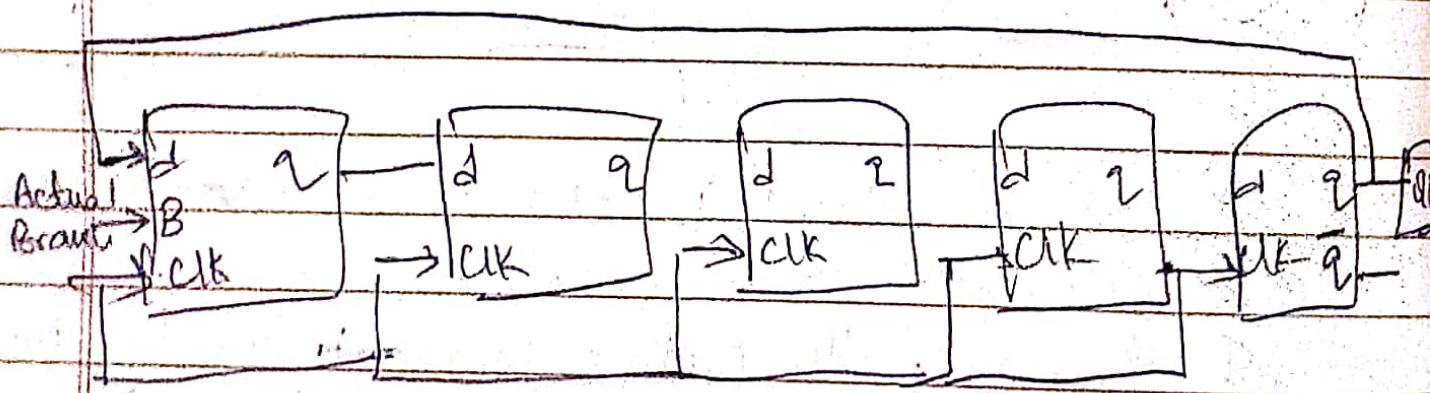
The pattern will just keep shifting through the shift register & produce the correct branch O/P. In this case, the pattern repeated would be 01001

e) If the exact opposite is given, the accuracy would be zero.

b) for the first 4 branches
number of predictions correct = 3.
Accuracy = 3/4.
→ ignore the first calculation 5b)

~~for a 2 bit predictor, if it is repeated forever.~~
~~accuracy = 3/5.~~

f) The same branch predictor can be implemented, but based on the input to the shift register. We can just take inverse of the final output or give it to the system as it is.



output logic drawn below