Homework - 3

CS2323 - Computer Architecture

1. Given: 6 staged pipeline processer (p)
Delays - 130, 125, 110, 132, 135, 145 ps.
Pipeline register delay - 12 ps.

a) # instruction to be executed - 1000 (n)

~~I.C.T~~

cycles required (with pipeline and no stall cycles) = $n + p - 1$

$= 1000 + 6 - 1$

$= 1005$

Clock cycle = ~~Delay of the longest~~ Longest delay + pipeline register delay (if any)

Clock cycle = $(145 + 12)$ ps

$= 157$ ps

∴ Total execution time = 1005×157

$= 157785$ ps

b) Now 20% of the cycles incur a 1 cycle stall

∴ 200 extra cycles

Additional time = 200 x 157
= 31400 ps

Total time = (157785 + 31400) ps
= 189185 ps

2) Given: P1 is a 5 stage ideal processor with 800ps latency
P2 is similar to P1 but includes multi instruction increasing the latency of ALU by 250ps but reduces instruction count by 10%

a) For P1, clock cycle = time for each stage (as it's equally divided)

Min clock cycle = $\frac{800}{5}$ = 160 ps

For P2, clock cycle = longest stage delay (i.e the ALU stage) = 160 + 250
= 410 ps

b) Lets say P1 has to execute x instructions.

∴ P2 executes 0.9x instructions

$T_{P1} = I.C_{P1} \times (\text{Clock cycle})_{P1}$

$T_{P2} = I.C_{P2} \times (\text{Clock cycle})_{P2}$

$T_{P1} = (x + 5 - 1) \times 160 \text{ ps}$

As it a pipelined processor
Actual I.C = n + p - 1

$T_{P1} = (x + 4)160$
$= (160x + 640) \text{ ps}$

$T_{P2} = (0.9x + 5 - 1) \times 410$
$= (0.9x + 4) \times 410$
$= (369x + 1640) \text{ ps}$

x cannot be non negative and for any $x \geq 0$, $T_{P2} < T_{P1}$

∴ P1 executes a given code in smaller time.

c) ~~I~~ $n_{P1} = 5000$ (Given)

$n_{P2} = 5000 \times 0.9$
$= 4500$

$I.C_{P1} = n_{P1} + p - 1$
$= 5000 + 5 - 1 \quad (p = 5)$
$= 5004$

$I.C_{P2} = 4504$

$T_{P1} = I.C_{P1} \times \text{Clock cycle}_{P1}$

$= (5004 \times 160)\ ps$
$= 800640\ ps$

$T_{P2} = I.C_{P2} \times \text{Clock cycle}_{P2}$
$= (4504 \times 410)\ ps$
$= 1846640\ ps$

3. Original code -

```
1 add  x14  x12  x11
2 add  x15  x14  x12
3 ld   x13  8(x13)
4 ld   x12  0(x14)
5 and  x13  x15  x13
6 ld   x11  4(x13)
7 sd   x13  0(x15)
```

a)
```
add  x14  x12  x11      # R instruction
nop                      requires 2 nop's
nop                      to counter
add  x15  x14  x12       data hazard
ld   x13  8(x13)
ld   x12  0(x14)         # Only 1 nop
nop                        as ld x12 0(x14)
and  x13  x15  xx13      acts as an nop.
nop
nop                      # Same as above.
ld   x11  4(x13)
sd   x13  0(x15)
```

i) 2 x14 data accesses made successively in line 1 and 2
∴ 2 nops for writeback of line 1 to be completed before decode of line 2

ii) x13 is accessed in line 3 & 5
line 4 acts as a nop and ∴ one more nop is required to avoid data hazard

iii) x13 accessed successively in line 5 & 6. Handled as in (i)

b) No "nops required as pipeline registers allow for the forwarding of the required data to the appropriate stage.