HW4_2025

Sriraj Vasanta
CS24BTECH11066

October 2025

---

1. **Clock period**

$$\text{Clock period} = \max(\text{All delays}) + (\text{Pipeline register delay})$$
$$= 145 + 12 = 157\,\text{ps}$$

   (a) **Without stalls**

$$\text{Total cycles} = (\text{No. of stages}) + (\text{No. of instructions} - 1)$$
$$= 6 + 1000 - 1 = 1005$$
$$\text{Execution time} = 1005 \times 157 = \boxed{1{,}57{,}785\,\text{ps}}$$

   (b) **With stalls**

$$\text{Total cycles} = \text{Normal cycles} + \text{Stall cycles}$$
$$= 1005 + 200 \quad (20\% \text{ of } 1000)$$
$$= 1205$$
$$\text{Execution time} = 1205 \times 157 = \boxed{1{,}89{,}185\,\text{ps}}$$

2. (a) **Clock Period Calculation for P_1**

$$\text{Total latency of } P_1 = 800\,\text{ps}$$
$$\text{Clock period of } P_1 = \frac{\text{Total latency}}{\text{Number of stages}} = \frac{800}{5} = \boxed{160\,\text{ps}}$$

   Clock Period Calculation for P_2

$$\text{Latency of ALU of } P_2 = 160 + 250 = 410\,\text{ps}$$
$$\text{Latency of other stages of } P_2 = 160\,\text{ps (Same as } P_1)$$
$$\text{Clock period of } P_2 = \max(\text{Latency of all stages}) = \boxed{410\,\text{ps}}$$

   (b) **Execution Time Comparison**

   Let there be N instructions on P_1.

   Then, there are $\frac{9N}{10}$ instructions on P_2 (since there is a 10% decrease).

$$t_1 = (5 + N - 1) \times 160 = (N + 4) \times 160 \quad \text{(i)}$$
$$t_2 = \left(5 + \frac{9N}{10} - 1\right) \times 410 = \left(\frac{9N}{10} + 4\right) \times 410 \quad \text{(ii)}$$

$$t_2 = 369N + 1640$$
$$t_1 = 160N + 640$$

Since $t_2 > t_1$, we have:

$$\boxed{P_1 \text{ executes in smaller time}}$$

(c) From (i):

$$t_1 = (5000 + 4) \times 160 = \boxed{800{,}640\,\text{ps}}$$

From (ii):

$$t_2 = (4500 + 4) \times 410 = \boxed{1{,}846{,}640\,\text{ps}}$$

3. (a) **No data forwarding and no hazard detection:**

```
add x14, x12, x11
# 2 nops after a writeback instruction
nop
nop
add x15, x14, x12
ld x13, 8(x13)
ld x12, 0(x14)
# 1 nop will suffice because the above instruction is occupying one cycle
nop
and x13, x15, x13
# 2 nops after a writeback instruction
nop
nop
ld x11, 4(x13)
sd x13, 0(x15)
```

(b) **Data forwarding with no hazard detection:**

```
# No nops needed after writeback instructions, except maybe after ld instruction
# The 1 cycle gap is required because the value to be loaded is determined in the MEM stage, not the EX stage
add x14, x12, x11
add x15, x14, x12
ld x13, 8(x13)
ld x12, 0(x14)
# 1 cycle gap is required after a ld instruction with data forwarding, which is provided by the above instruction.
and x13, x15, x13
ld x11, 4(x13)
sd x13, 0(x15)
```

(No nops in this case)

4. (a) **No data forwarding and no hazard detection:**

```
add x14, x12, x11
# 2 nops after a writeback instruction
nop
nop
# 2 nops after a branch instruction, as the next instruction should be fetched after updating PC, which is done in the EX stage
L2: beq x15, x14, L1
    nop
    nop
    ld x13, 8(x13)
```

nop
nop
beq x12, x13, L2
nop
nop
L1: and x13, x15, x13
 # 2 nops after a writeback instruction
nop
nop
ld x11, 4(x13)
sd x13, 0(x15)

(b) **Data forwarding with no hazard detection:**
 # No nops needed after writeback instructions, except maybe after ld *instruction*
add x14, x12, x11
 # Control hazards are not fixed by data forwarding, so the number of nops after a branch instruction is same as before
L2: beq x15, x14, L1
nop
nop
ld x13, 8(x13)
 # There is no instruction in between the two instructions, unlike in 3b, so 1 nop should be added
nop
beq x12, x13, L2
nop
nop
L1: and x13, x15, x13
ld x11, 4(x13)
sd x13, 0(x15)