# 7 Pipelining [70 points]

Code Listing 1 contains a piece of assembly code. Table 1 presents the execution timeline of this code.

```
1     MOVI R1, X          # R1 <- X
2     MOVI R2, Y          # R2 <- Y
3  L1:
4     MUL  R4, R1, R1     # R4 <- R1 × R1
5     MUL  R1, R1, R2     # R1 <- R1 × R2
6     ADD  R4, R5, R6     # R4 <- R5 + R6
7     ADD  R5, R2, R4     # R5 <- R2 + R4
8     SUBI R3, R1, 2048   # R3 <- R1 - 2048, set condition flags
9     JNZ  L1             # Jump to L1 if zero flag is NOT set
10    MUL  R1, R1, R2     # R1 <- R1 × R2
```

Code Listing 1: Assembly Program

| | | Cycles | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Instructions | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 1 | MOVI R1, X | F | D | E1 | E2 | E3 | M | W | | | | | | | | | |
| 2 | MOVI R2, Y | | F | D | E1 | E2 | E3 | M | W | | | | | | | | |
| 3 | MUL R4, R1, R1 | | | F | D | - | E1 | E2 | E3 | M | W | | | | | | |
| 4 | MUL R1, R1, R2 | | | | F | - | D | E1 | E2 | E3 | M | W | | | | | |
| 5 | ADD R4, R5, R6 | | | | | | F | D | E1 | E2 | E3 | M | W | | | | |
| 6 | ADD R5, R2, R4 | | | | | | | F | D | - | - | E1 | E2 | E3 | M | W | |
| 7 | SUBI R3, R1, 2048 | | | | | | | | F | - | - | D | E1 | E2 | E3 | M | W |
| 8 | JNZ L1 | | | | | | | | | | | F | D | - | - | E1 | ... |
| 9 | ... | ... | | | | | | | | | | | | | | | |

Table 1: Execution timeline (F:Fetch, D:Decode, E:Execute, M:Memory, W:WriteBack)

Use this information to reverse engineer the architecture of this microprocessor to answer the following questions. Answer the questions as precisely as possible. If the provided information is not sufficient to answer a question, answer "Unknown" and explain your reasoning clearly.

(a) [15 points] List the data forwarding paths between pipeline stages.

The result of E3 stage is forwarded to E1 stage (e.g., R1's value at clock cycle 6 and R4's value at clock cycle 11). The result of M stage is forwarded to E1 stage (e.g., R1's value at clock cycle 7.)
The result of E3 stage is forwarded to the condition registers (e.g., SUBI and JNZ at clock cycle 15).
There is no other information for any other data forwarding. Therefore, other data forwardings are unknown.

(b) [10 points] Does this machine use hardware interlocking or software interlocking? Explain.

Hardware interlocking. It detects data dependencies and stalls the pipeline accordingly without needing any software-induced NOPs.

For the rest of this question, assume the following:

- X = 4, Y = 2 in Code Listing 1.
- Branch predictor always predicts correctly.
- The machine uses hardware interlocking.

At a given clock cycle $T$,

- the value stored in R1 is 1024.
- the processor fetches the dynamic instruction $N$ which is `MUL R4, R1, R1`

(c) [15 points] Calculate the value of $T$ (the clock cycle of the given snapshot). Show your work.

T = 82.

**Explanation.**
The instruction `MUL R4, R1, R1` is fetched for the first time at the clock cycle 3. After the first iteration of the loop, the instruction is fetched for the second time at the clock cycle 12.

The instruction `JNZ L1` stalls at the Decode stage and delays `MUL R4, R1, R1`. Due to this delay, there are 10 cycles in between the Nth and (N+1)th times the instruction is fetched, after the first iteration of the loop.

If R1 = 1024, this instruction is fetched and executed 8 times so far.

Since in cycle $T$ the first instruction in the loop (`MUL R4, R1, R1`) is being fetched, no cycles of the 9th iteration have executed so far.

Then, $T = 12 + 7 \times 10 = 82$

(d) [15 points] Calculate the value of $N$ (the total number of dynamic instructions fetched by the clock cycle $T$). Show your work.

N = 51.

**Explanation.**
Loop iterates for 8 times before the processor reaches to clock cycle $T$.

There are two instructions before the loop starts.

Then, $N = 2 + 8 \times 6 + 1 = 51$ (assuming that the instruction indices start from 1).

(e) [15 points] Calculate the total execution time of the assembly code in Code Listing 1 until the completion in terms of the number of clock cycles. Show your work.

100 cycles.

**Explanation.**
Until the end of the second iteration, the loop takes 19 cycles as shown above.
The steady-state throughput of an iteration after the first iteration is 6 instructions in 10 cycles.

Loop will iterate until R1 becomes 2048, which means 9 iterations in total.

There is only one instruction after the loop, which takes 1 cycle to complete.

Then, $T = 19 + 8 \times 10 + 1 = 100$