Lab Assignment help:

Write a C++ program to add two given integer numbers using user defined function.

```
#include <iostream>
using namespace std;

int add (int x, int y); // function declaration
int main() // main code
{
    int a=5, b=4, c;
    c = add(a, b); // function call
    cout<<"Result:"<<c;
    return 0;
}

int add (int x, int y) // function definition
{
    return (x + y);
}
```

5 4

5 + 4 = 9

OUTPUT.

```
Result : 9
```

BL →

ARMv7

```
→ 100   MOV  R1, #5
→ 104   MOV  R2, #4
→ 108   BL addition
→ 112   B Exit
→ addition:
  →116  ADD R9, R1, R2 ✓
   120  BX LR
Exit:
```

→ 100 101 102 103

Register

R1	5.
R2	4
R9	9.
LR.	108

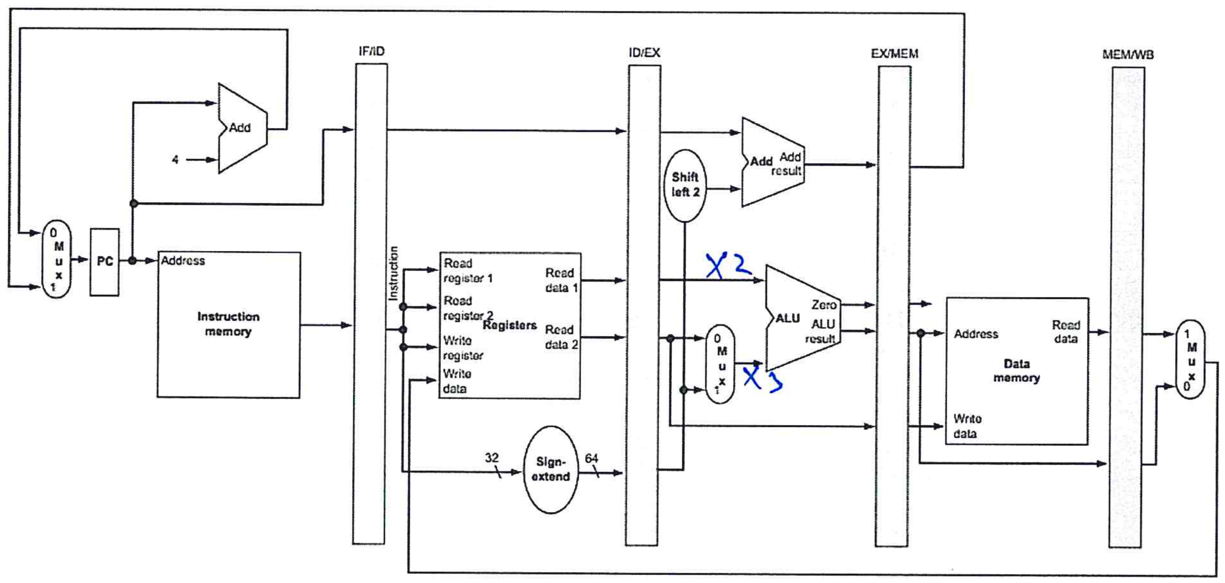
① → LDUR X1, [X4, #100] ✓

Pipelined Datapath

② LDUR X2, [X4, #100] ✓

Pipelining is an implementation technique in which multiple instructions are overlapped in execution.

ADD X1, X2, X3.



Instruction Fetch (IF)

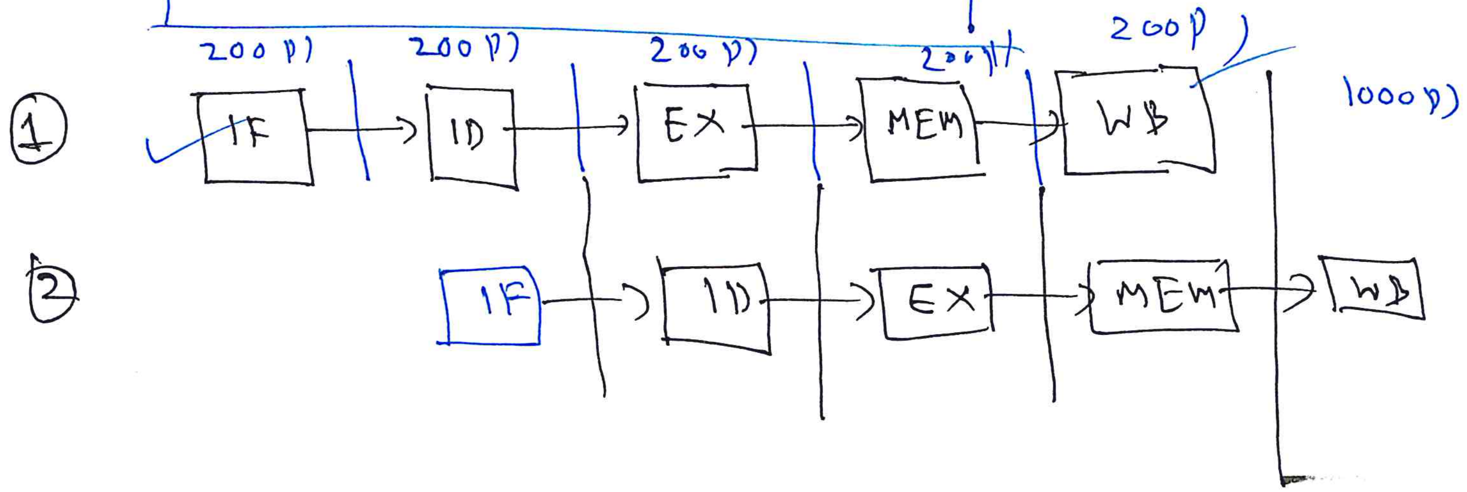
Instruction Decode (ID)

Execution (EX)

Memory access (MEM)

Write back (WB)

2000ps

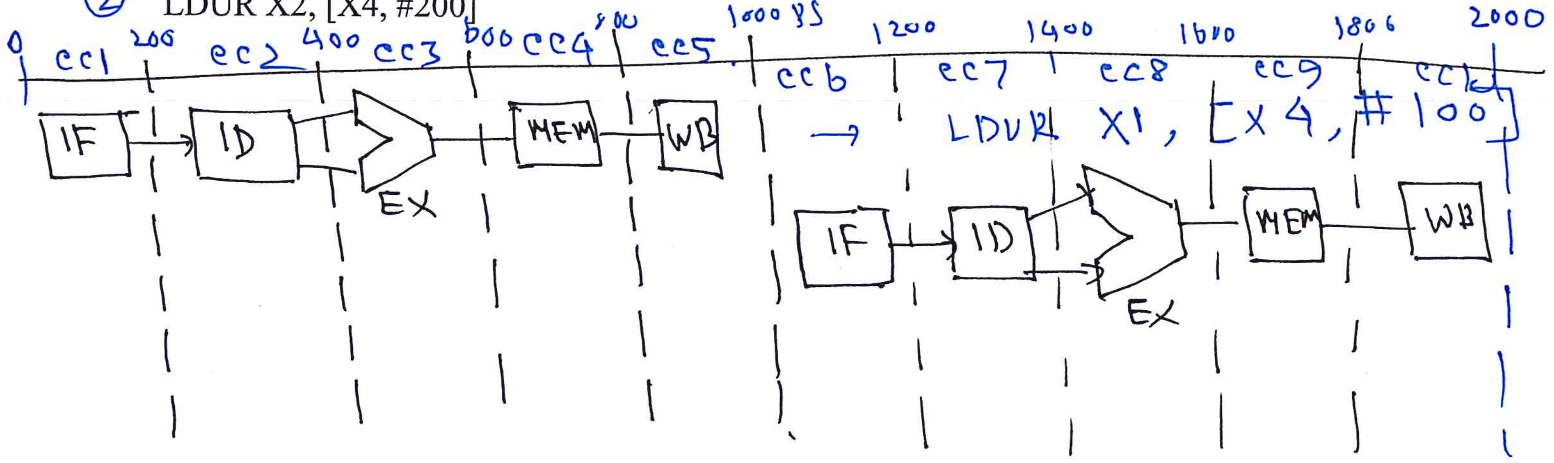


Non- pipelined VS pipelined

Example: Draw the multicycle Non-pipelined diagram for the following instructions. Calculate total execution time. Consider each stage takes one clock cycle which is 200ps.

① LDUR X1, [X4, #100]

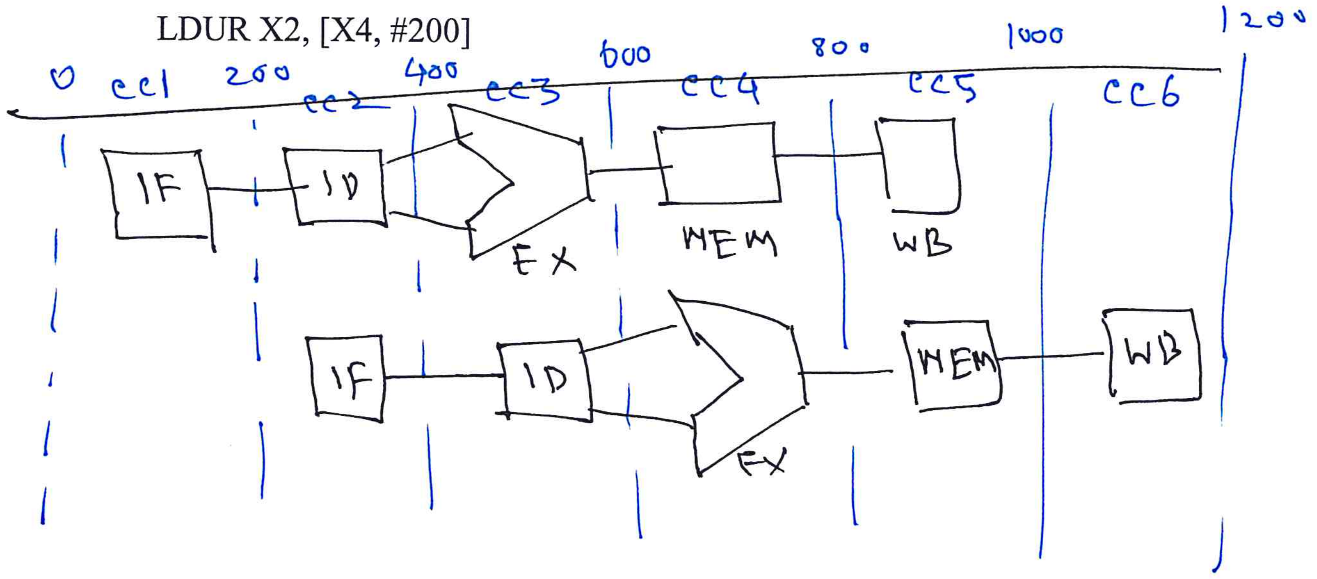
② LDUR X2, [X4, #200]



Example: Draw the multicycle pipelined diagram for the following instructions. Calculate total execution time. Consider each stage takes one clock cycle which is 200ps.

LDUR X1, [X4, #100]

LDUR X2, [X4, #200]

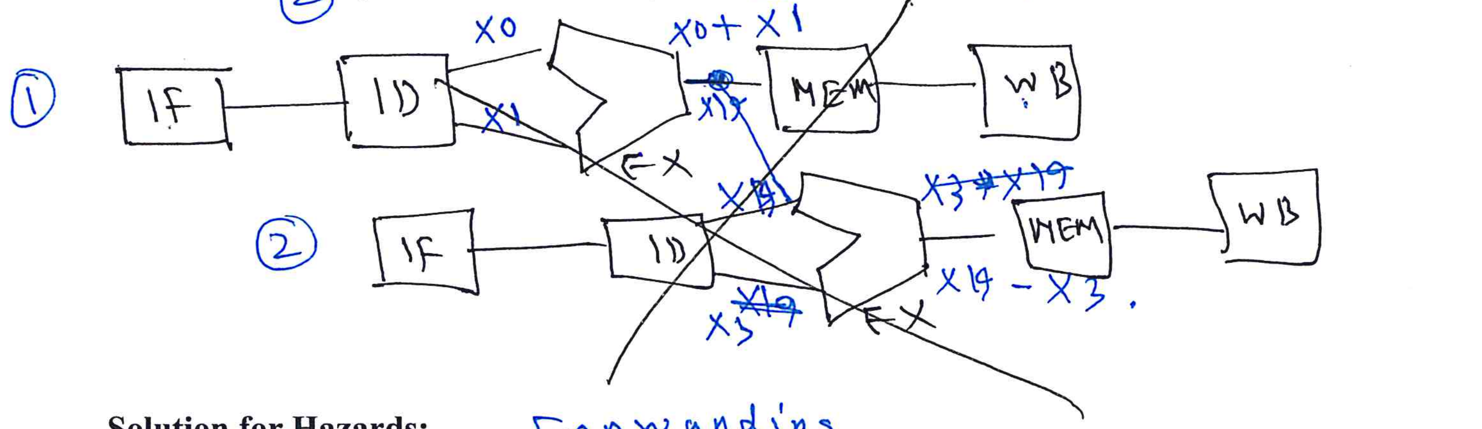


Pipeline Hazards: There are situations in pipelining when the next instruction cannot execute in the following clock cycle. These events are called hazards, and there are three different types.

Data Hazards: When a planned instruction cannot execute in the proper clock cycle because data that are needed to execute the instruction are not yet available. ✓

100 → ① ADD X19, X0, X1 // X19 = X0 + X0

104 ② SUB X2, X19, X3 // X2 = X19 − X3



Solution for Hazards: Forwarding

- A method of resolving a data hazard by retrieving the missing data element from internal buffers rather than waiting for it to arrive from programmer visible registers or memory.
- Forwarding paths are valid only if the destination stage is later in time than the source stage.