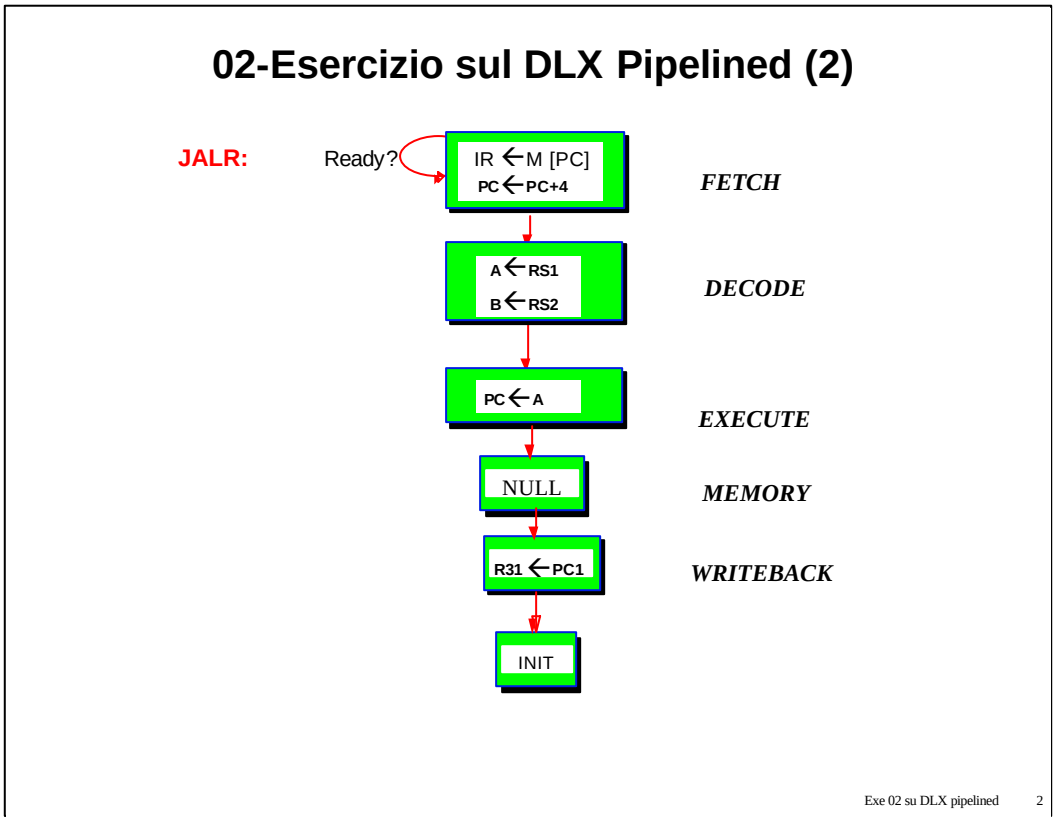
02-Esercizio sul DLX Pipelined (1)

Si spieghi il funzionamento e si indichi una possibile codifica binaria della seguente istruzione DLX:

JALR R13 (jump and link indiretto)

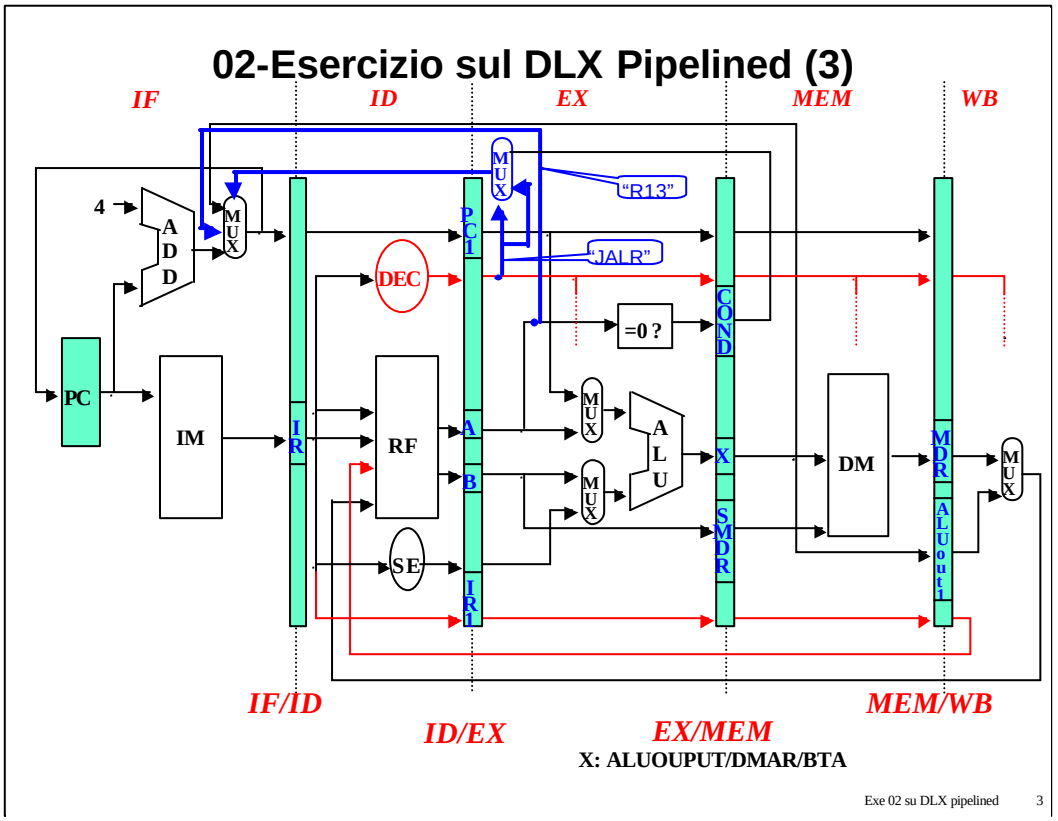
•1) Si mostrino poi le operazioni che devono essere eseguite in ogni stadio durante l'esecuzione dell'istruzione JALR nell'ipotesi di ridurre il più possibile con l'uso di FU eventuali alee.
•2) Si disegni schematicamente il datapath del DLX in pipeline indicando chiaramente quali sono le modifiche necessarie per supportare l'esecuzione dell'istruzione JALR rispetto alla struttura del datapath in pipeline visto a lezione introducendo i forwarding opportuni.
•3) Si verifichi infine se la realizzazione sviluppata nel punto 2 impone alla pipeline di stallare per una situazione di alea di controllo e, se sì, per quanti periodi di clock.



1) Dopo la fase di Decode, cioè in Execute, sono disponibili sia l'indirizzo del salto letto nel registro R13 (in A), che il "vecchio" program counter già incrementato di 4 (in PC1). Perciò in Execute si può già completare il salto scrivendo il nuovo indirizzo in PC. Il link al PC precedente il salto sarà scritto in R31 durante il WB.

Per limitare gli stalli risolutivi di eventuali alee di controllo è bene anticipare il più possibile PC←A.

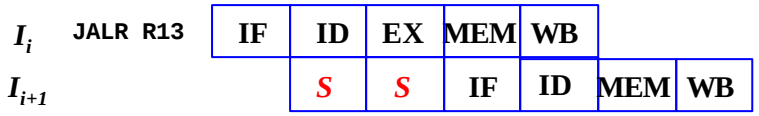
Viceversa per R31 ←PC1 si può aspettare il WB facendo passare il dato per il percorso già esistente nel datapath attraverso la ALU, X e ALUOUT1. E' possibile attendere il WB senza problemi perchè ci saranno almeno un paio di istruzioni da eseguire prima di caricare nuovamente PC←R31 per tornare al chiamante con J R31.



2) Con la linea blu di spessore maggiorato sono indicate le modifiche da fare al datapath per anticipare la scrittura dell'ind. del salto sul PC. Il MUX aggiunto serve per gestire la commutazione del mux che manda il dato all'ingresso del PC.

Solo nel periodo di clock successivo all'aggiornamento di PC sarà possibile eseguire il fetch della prima istruzione che si trova nell'indirizzo a cui si è saltato (anche nella normale dinamica del PC esso è incrementato in IF, cioè con un ciclo di anticipo rispetto a quello in cui è compiuto il fetch).

02-Esercizio sul DLX Pipelined (4)



Exe 02 su DLX pipelined 4

3) Sì, la pipeline deve stallare per un'alea di controllo finché JALR non abbia aggiornato il PC.

Infatti mentre JALR è in Decode ed Execute la pipeline non può accettare in ingresso altre istruzioni.

Quando poi JALR va in Memory, può contemporaneamente essere eseguito il Fetch dell'istruzione all'indirizzo del salto; quindi complessivamente JALR introduce un'alea di controllo che si risolve introducendo due stalli.

Il problema per l'aggiornamento di R31 sul RF si porrebbe solo nel caso in cui l'istr *i+1* fosse già l'istruzione di ritorno dalla subroutine (in ASM del DLX essa è "J R31") perché la sua decodifica sarebbe eseguita mentre è aggiornato R31. Tale alea si può risolvere pensando di avere un Register File con registri in grado di essere scritti e letti contemporaneamente e dai quali si può già leggere il valore nuovo che si sta scrivendo nonostante la contemporaneità delle due azioni.