Istr.Store - funzionamento

mercoledì 1 marzo 2023 12:31

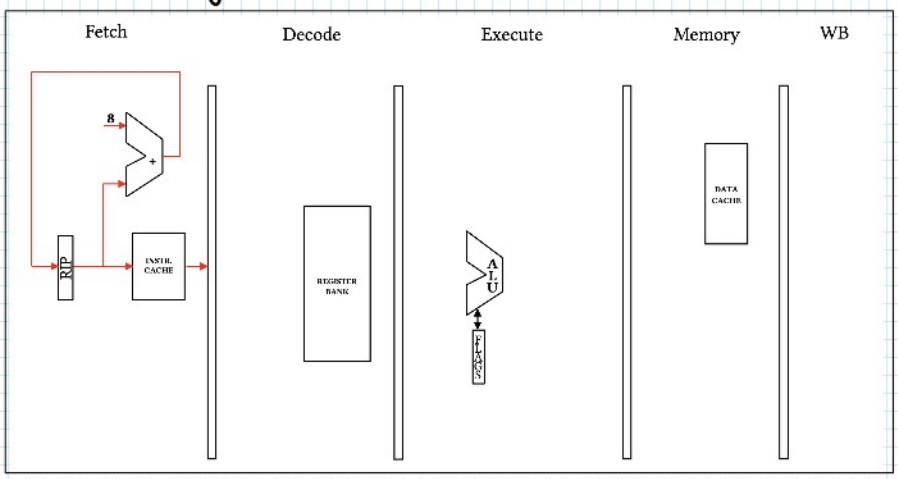
questa è la SCRITTURA IN MEMORIA, è l'opposto della load.

Istruzione Store: formato

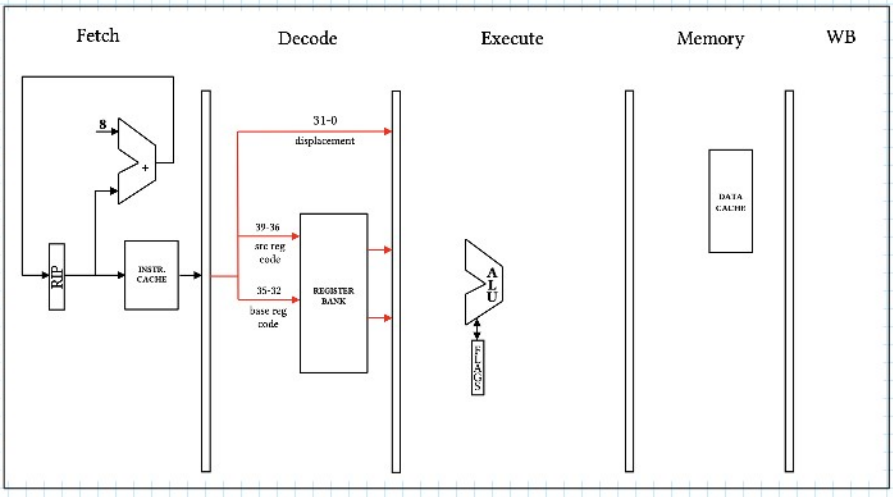
1. B_p = 1: utilizziamo il registro base
2. Displacement è una d.c.c.: non si possono usare costanti e non si possono effettuare operazioni in memoria
3. SS/DS = 11: solo operandi a 64 bit
4. DI = 10: è utilizzato uno spiazzamento
5. Mem = 01: il secondo operando è in memoria

IL REGISTRO SORGENTE è a tutti gli effetti IL REGISTRO SORGENTE; IL REGISTRO DESTINAZIONE diventa il REGISTRO di BASE dell'operando destinazione, ho uno SPIAZZAMENTO per calcolare BASE + SPIAZZAMENTO.

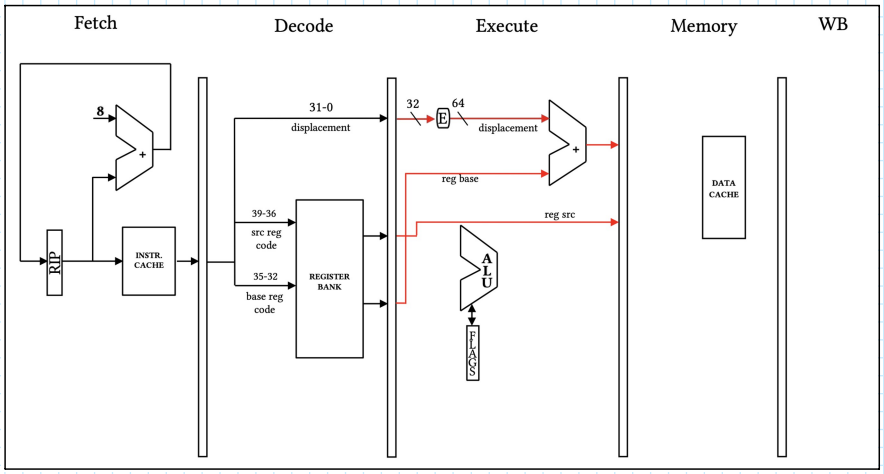
LA FASE di Fetch è sempre uguale!



NELLA fase di decode NON devo propagarmi IN avanti il codice del REGISTRO destinazione, perché lo USO PER calcolarmi l'INDIRIZZO. Mi è sufficiente leggere il contenuto del REGISTRO destinazione, perché SARÀ la BASE a cui vado ad applicare lo SPIAZZAMENTO, e IL REGISTRO SORGENTE NON C'È altro che IL VALORE che voglio andare a SCRIVERE IN memoria.

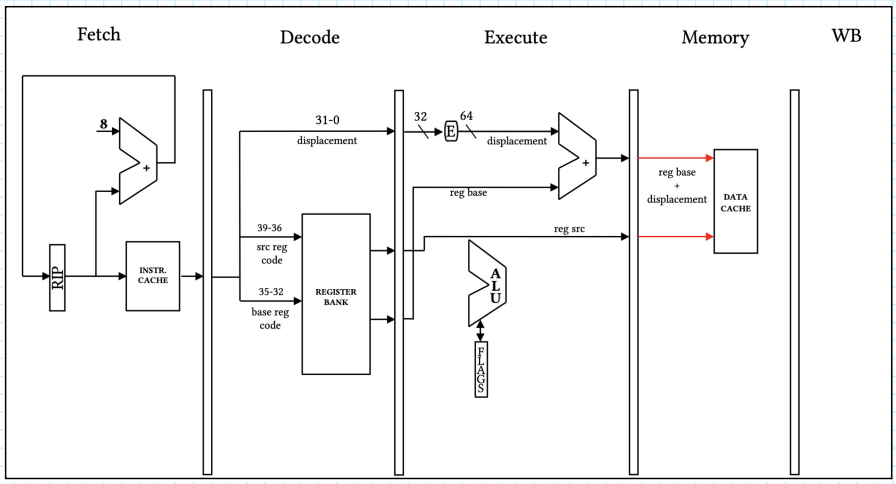


NELLA fase di EXECUTE la somma sarà ancora fra lo SPIAZZAMENTO e IL REGISTRO di BASE, che è IL REGISTRO di BASE dell'operando destinazione, sto SCRIVENDO IN MEMORIA.

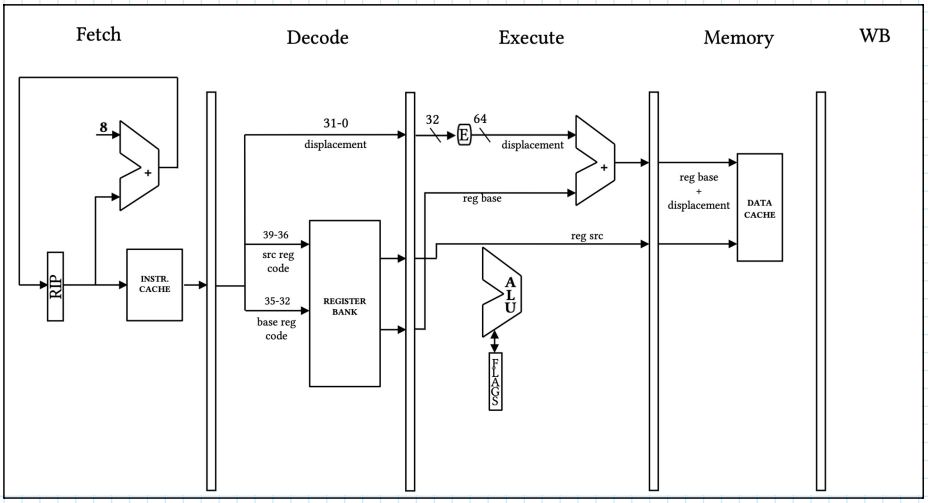


AVRÒ come operando destinazione : BASE+SPIAZZAMENTO.

A questo punto SCRIVO IN MEMORIA. passo alla data cache sia l'indirizzo (BASE +SPIAZZAMENTO) sia il valore che mi sono propagato del registro sorgente.



Nella fase di WRITE BACK non devo fare NULLA : L'ISTRUZIONE di STORE ha 4 FASI.



NON ho nessun valore da riscrivere dentro il Banco dei Registri !