CONFLITTI DI SALTO CONDIZIONATO

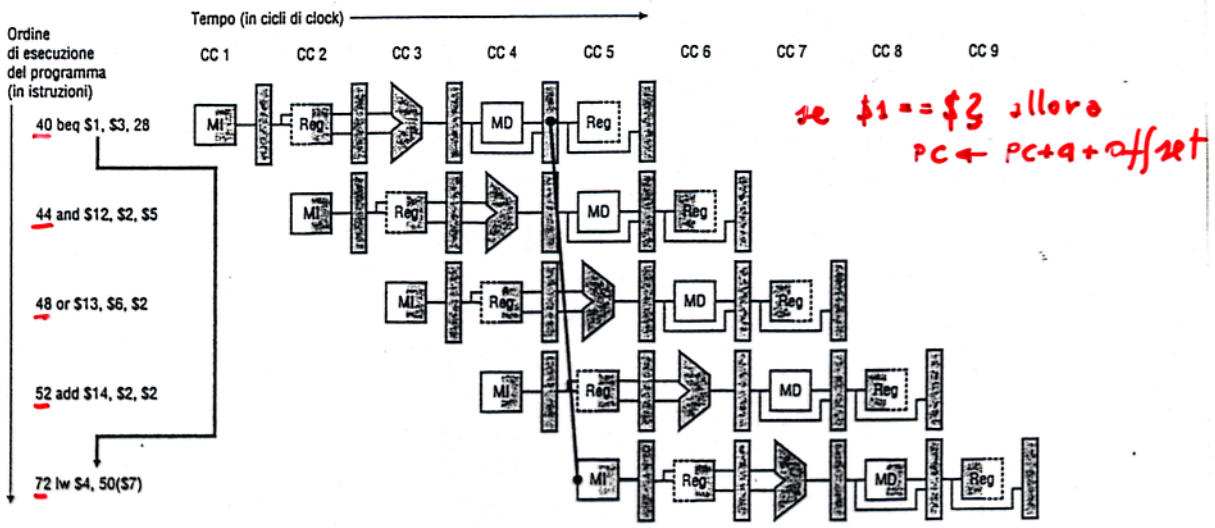


FIGURA 6.50 L'impatto della pipeline sull'istruzione di salto condizionato. I numeri alla sinistra dell'istruzione (40, 44, ...) sono gli indirizzi dell'istruzione. Poiché l'istruzione di salto condizionato decide se effettuare tale salto nello stadio MEM (ciclo di clock 4 per l'istruzione `beq`) le tre istruzioni sequenziali che seguono il salto condizionato verranno prelevate e inizierà la loro esecuzione. Senza intervento, le tre istruzioni successive verranno completate prima che l'istruzione `beq` effettui il salto all'istruzione `lw` all'indirizzo 72.

SOLUZIONE PESSIMISTICA: PORRE SEMPRE IN STATO DI STALL

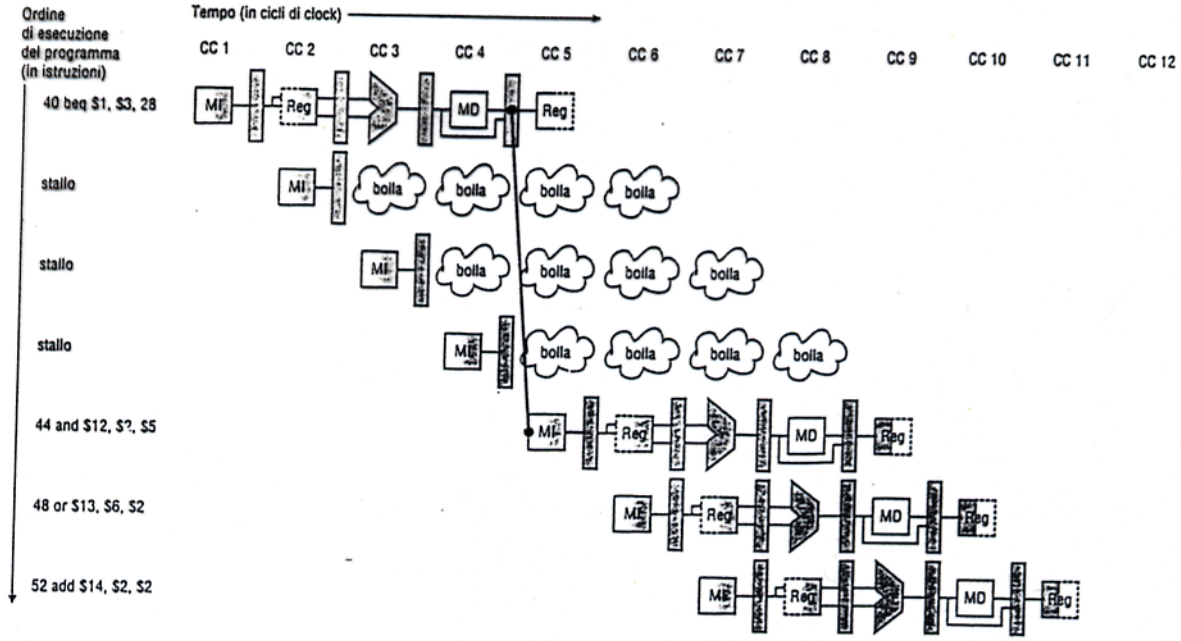
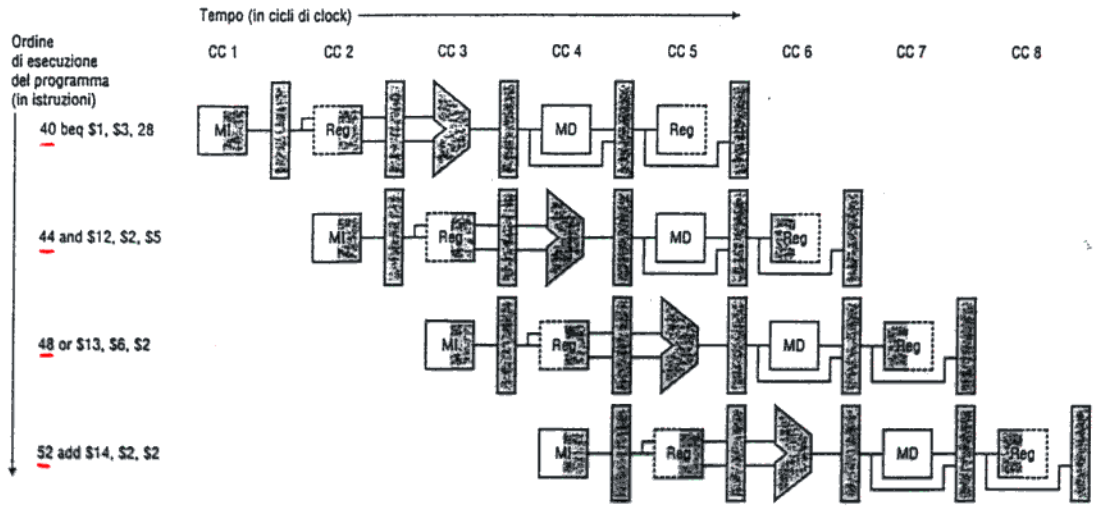


FIGURA 6.51 Un salto condizionato con stalli per risolvere il conflitto di controllo. La soluzione più semplice consiste nel porre in stallo tutte le istruzioni che seguono il salto condizionato fino a che la decisione sia chiara, e poi proseguire con l'esecuzione delle corrette istruzioni. In pratica, l'uso degli stalli incrementa il costo di un salto condizionato da uno a quattro cicli di clock.

SOLUZIONE OTTIMISTICA : IPOTIZZARE CHE IL SALTO NON VENGA ESEGUITO → NEL CASO DI SALTO ANNULLARE GLI EFFETTI



SEQUENZA SENZA SALTO

40 beq $1,$3,28 [MI] [Reg] [ALU] [MD] [Reg]

stadio in cui avviene la scelta tra { PC+4 ; PC+4+offset (i.e. branch AND zero = 1) }

↓ ANNULLARE EFFETTI ISTRUZIONI SUCC.

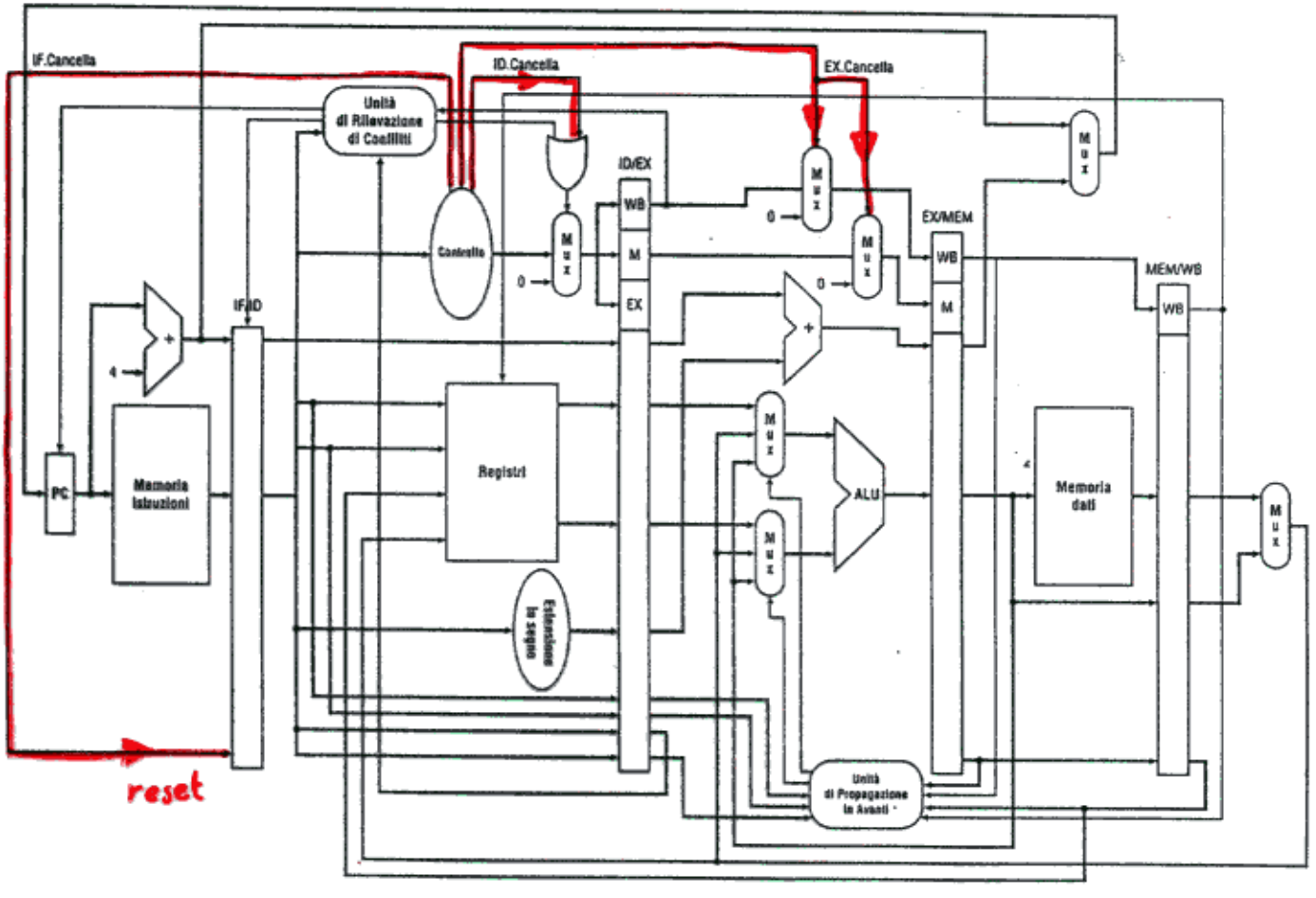
44 and $12,$2,$5 [MI] [Reg] [ALU] [bolla] [bolla]

48 or $13,$6,$2 [MI] [Reg] [bolla] [bolla] [bolla]

52 add $14,$2,$2 [MI] [bolla] [bolla] [bolla] [bolla]

72 lw $4, 50($7) [MI] [Reg] [ALU] [MD] [Reg]

SEQUENZA CON SALTO : BISOGNA INSERIRE 3 BOLLE

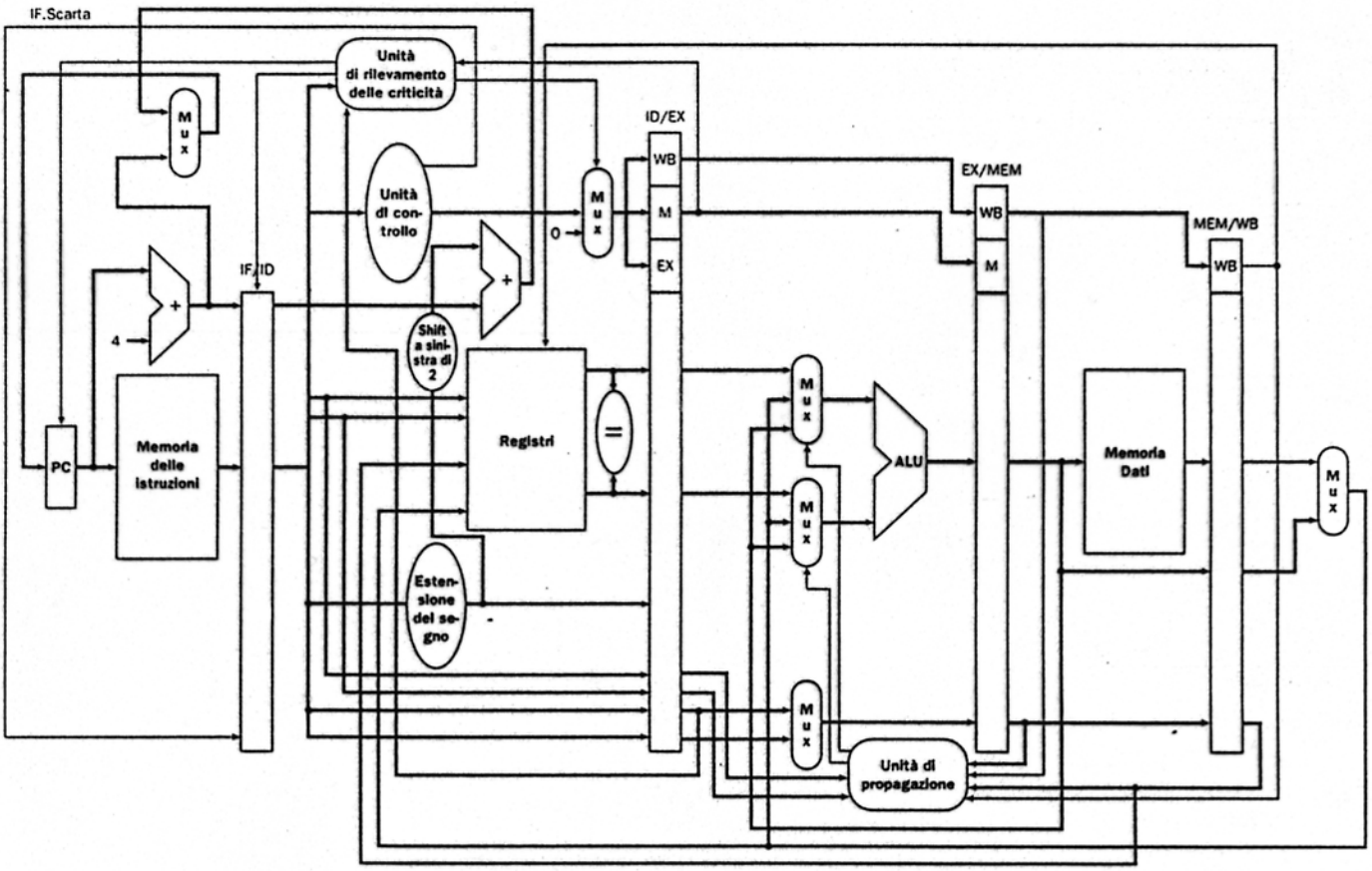


N.B. I FLIP/FLOP DI STATO DELL'ALU NON DEBBONO ESSERE MODIFICATI DALL'ISTRUZIONE SUCCESSIVA AL "beq"

OPPURE POSSONO ESSERE MODIFICATI MA DEBBONO ESSERE RIPORTATI ALLO STATO PRECEDENTE (RECOVERY)

RIDUZIONE NUMERO DI BOLLE

"decisione al 2° stadio"



gura 6.51 Unità di elaborazione per i salti, compreso l'hardware per scartare istruzioni successive al salto. L'ottimizzazione sposta la decision lativa al salto dal quarto stadio della pipeline al secondo; solo un'istruzione tra quelle che seguono il salto si trova nella pipeline in quel momento. La line controllo IF.Scarta trasforma l'istruzione caricata in una nop azzerando il registro IF/ID della pipeline. Benché la linea per lo scarto compaia qui in uscit ıll'unità di controllo, in realtà proviene dall'hardware che determina se il salto debba venire eseguito (etichettato con un segno di uguale a destra dei registı ello stadio ID).

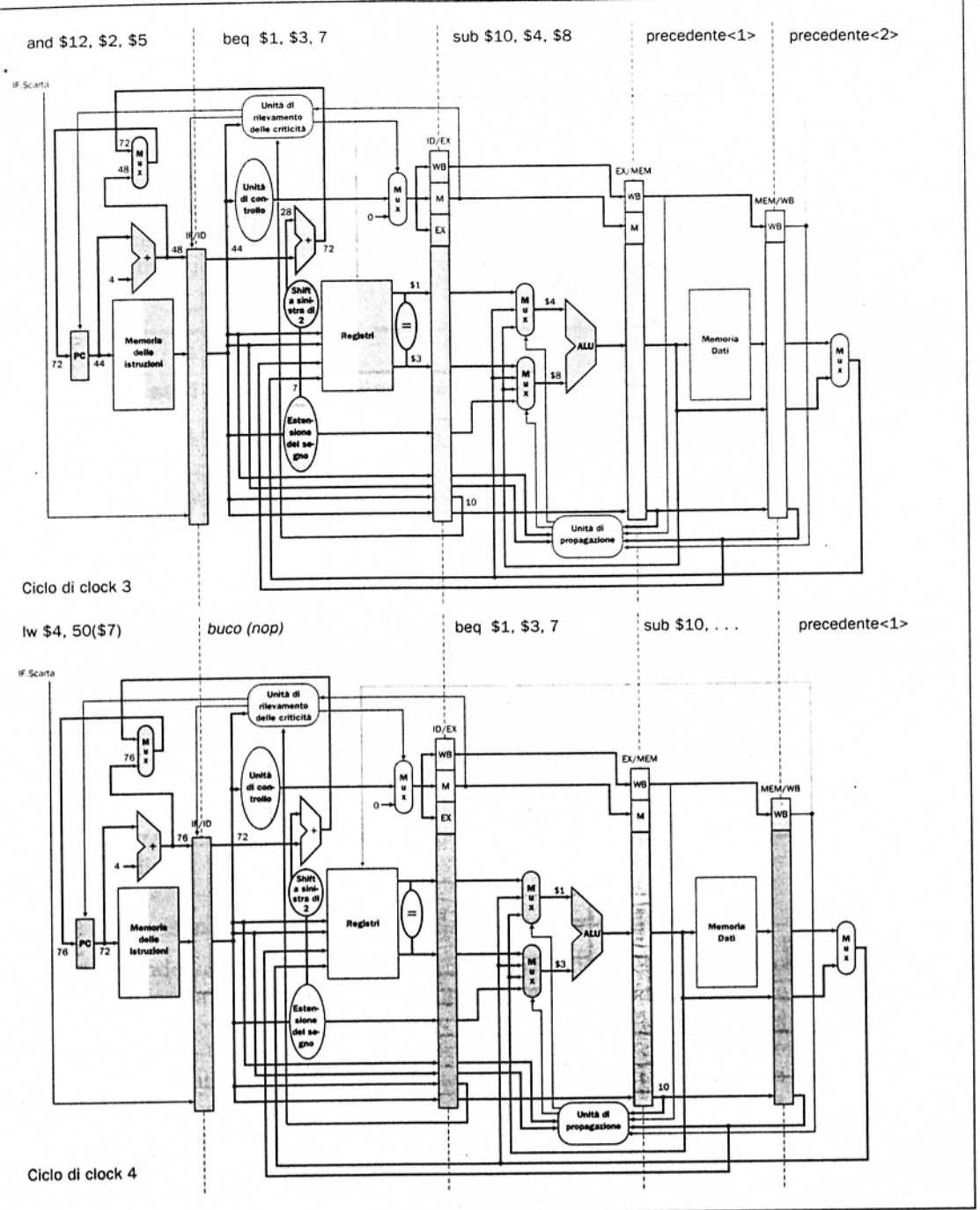


Figura 6.52 Lo stadio ID nel ciclo di clock 3 decide che il salto deve venire eseguito, quindi seleziona 72 come successivo indirizzo da caricare in PC e azzera l'istruzione caricata per il ciclo di clock successivo. Il ciclo di clock 4 mostra l'istruzione all'indirizzo 72 mentre viene caricata, nonché il buco corrispondente all'istruzione `nop` all'interno della pipeline, risultato dell'esecuzione del salto. Dal momento che `nop` corrisponde a `or $0, $0, 0`, si può discutere se lo stadio ID nel clock 4 debba venire evidenziato o meno.

GESTIONE ECCEZIONE DI OVERFLOW

ALL'ATTO DELLA RIVELAZIONE DI UN OVERFLOW:

1. MEMORIZZAZIONE DELL'INDIRIZZO DELL'ISTRUZIONE CHE HA CAUSATO L'OVERFLOW
2. PASSAGGIO DEL CONTROLLO ALLA PRIMA ISTRUZIONE DEL PROGRAMMA GESTORE DELL'ECCEZIONE
   i.e. $PC \leftarrow (4000\ 0040)_{HEX}$

bisogna annullare gli effetti delle istruzioni che nel frattempo sono entrate nella pipeline

inserzioni di bolle come nel caso di salto condizionato

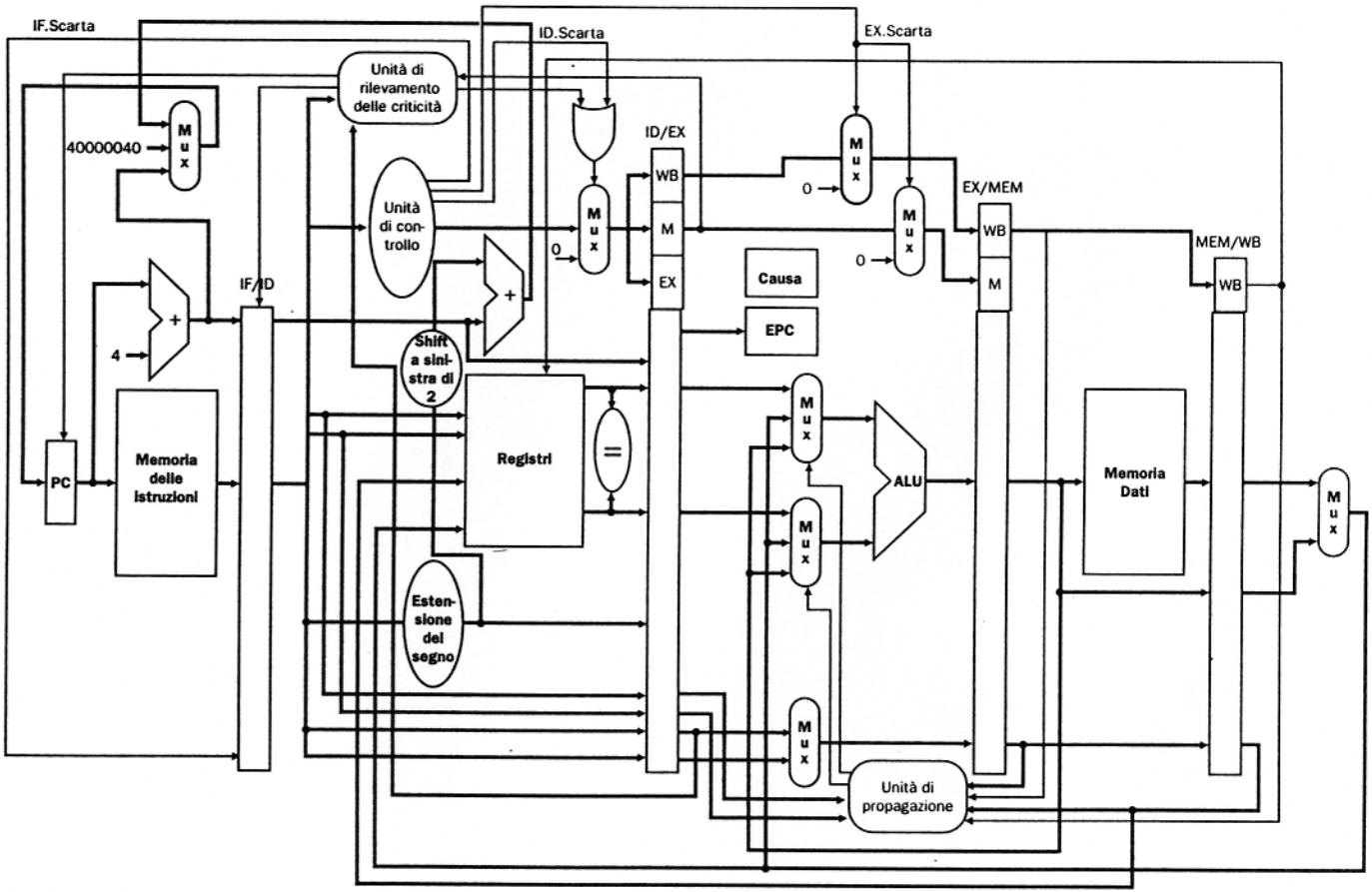


Figura 6.55 L'unità di elaborazione per la gestione delle eccezioni, con i relativi controlli. I cambiamenti rispetto alla figura 6.51 comprendono un nuovo ingresso, pari a 4000 0040_{esa}, collegato al multiplexer che fornisce il nuovo valore al PC, un registro Causa che mantiene traccia del motivo per cui si è verificata l'eccezione, ed un registro Exception PC che salva l'indirizzo dell'istruzione che ha provocato l'eccezione. L'ingresso 4000 0040_{esa} del multiplexer rappresenta l'indirizzo da cui iniziare il prelievo delle istruzioni nel caso si verifichi un'eccezione. Anche se qui non è mostrato, il segnale di overflow della ALU è uno degli ingressi dell'unità di controllo.

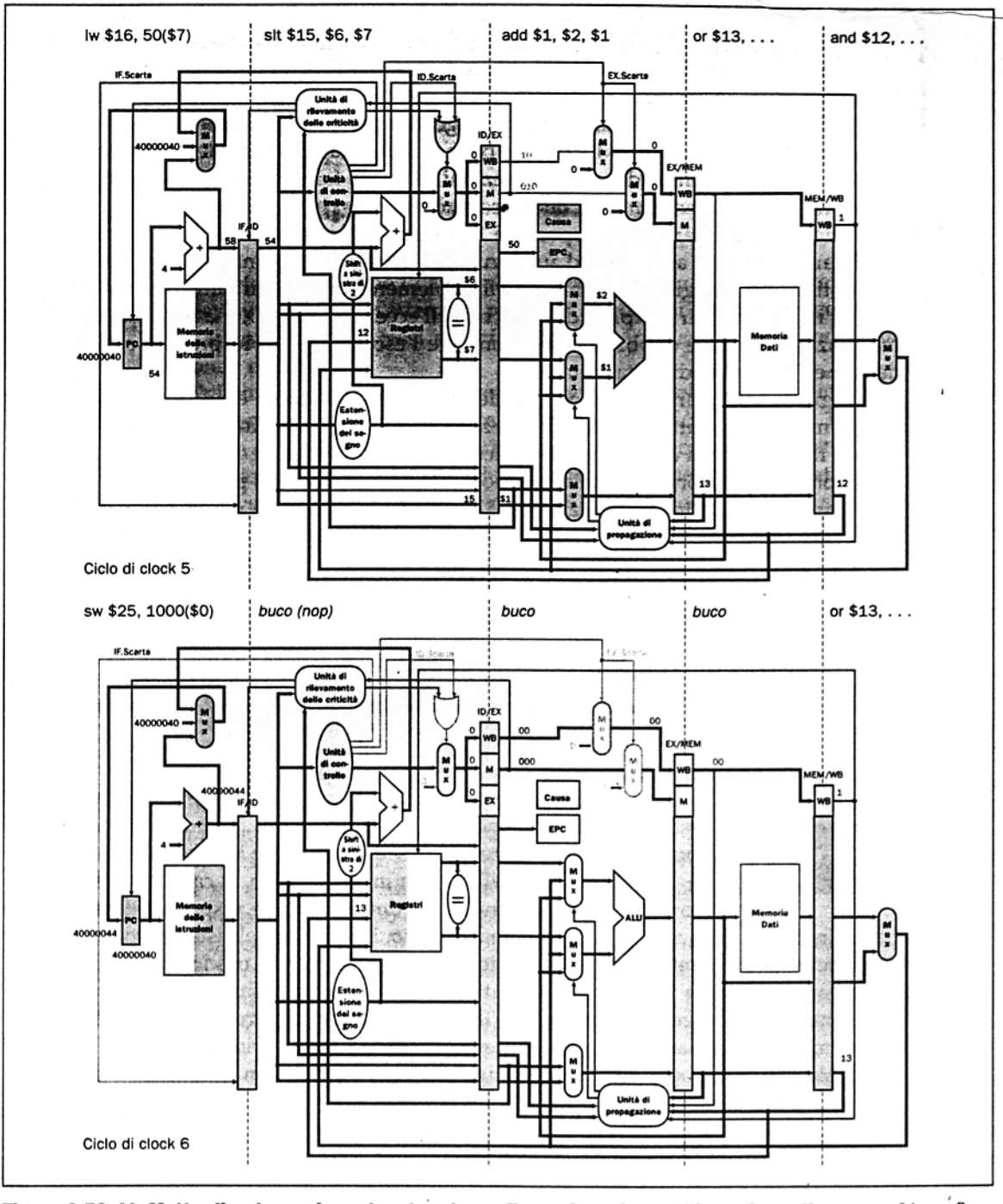


Figura 6.56 L'effetto di un'eccezione dovuta ad overflow aritmetico nell'istruzione di somma. L'overflow è riconosciuto durante lo stadio EX del ciclo di clock 5, provocando il salvataggio dell'indirizzo dell'istruzione che segue la somma nel registro EPC ($4C_{esa} + 4 = 50_{esa}$). L'overflow fa sì che tutti i segnali Scarta vengano attivati verso la fine di questo ciclo, azzerando i segnali di controllo dell'istruzione `add`. Il ciclo di clock 6 mostra le istruzioni convertite in buchi all'interno della pipeline ed il prelevamento della prima istruzione della procedura di gestione delle eccezioni (`sw $25, 1000($0)`) dalla locazione 4000 0040_{esa}. Si noti che le istruzioni `and` e `or`, precedenti alla `add`, vengono completate. Anche se qui non è mostrato, il segnale di overflow della ALU è uno degli ingressi dell'unità di controllo.

SCHEDULAZIONE DINAMICA DELLA PIPELINE

Obiettivo: quando si verifica uno stallo cercare di eseguire le istruzioni successive che non hanno dipendenze con le istruzioni che hanno causato lo stallo o sono in stallo

SCHEDULAZIONE DELLE ISTRUZIONI:
DA STATICA A DINAMICA

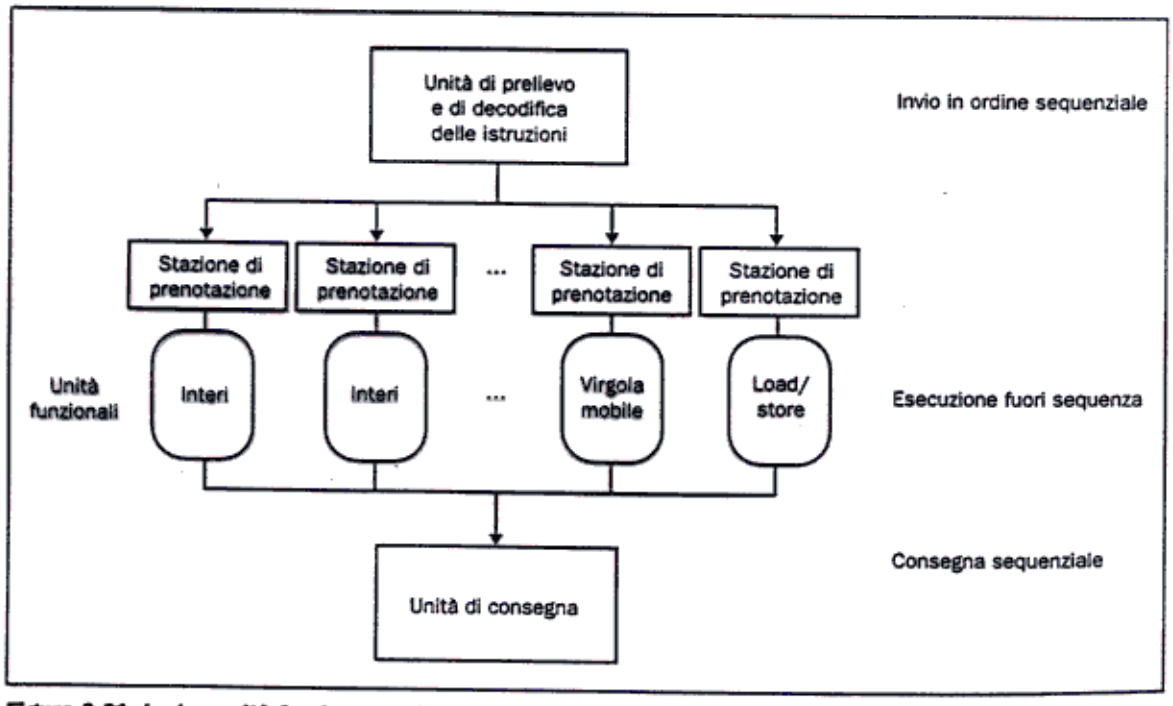


Figura 6.61 Le tre unità fondamentali di una pipeline con schedulazione dinamica.

Modello completamente in ordine: ordine di consegna uguale all'ordine di esecuzione del programma

Modello completamente fuori ordine: ordine di consegna diverso da quello di esecuzione del programma

- PROBLEMI DI CONSISTENZA -

Architettura del Power PC 604 e Pentium Pro

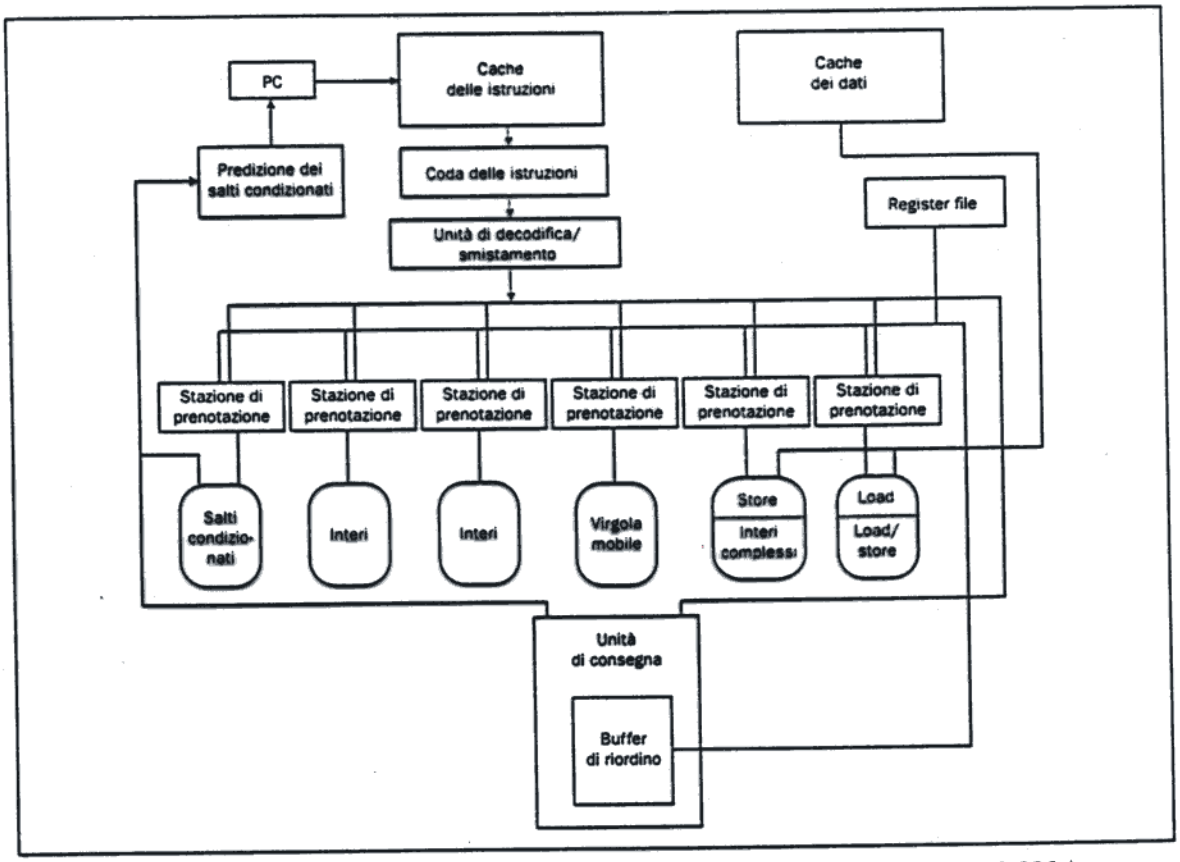


Figura 6.62 L'organizzazione generale della pipeline dell'Intel Pentium Pro e del PowerPC 604.

fetch: 4 istruzioni per Power PC
16 byte per Pentium-Pro

PRESTAZIONI E PROFONDITA' DEL PIPELINE

Aumentare la profondità del pipeline comporta sempre un incremento delle prestazioni?

NO perche':

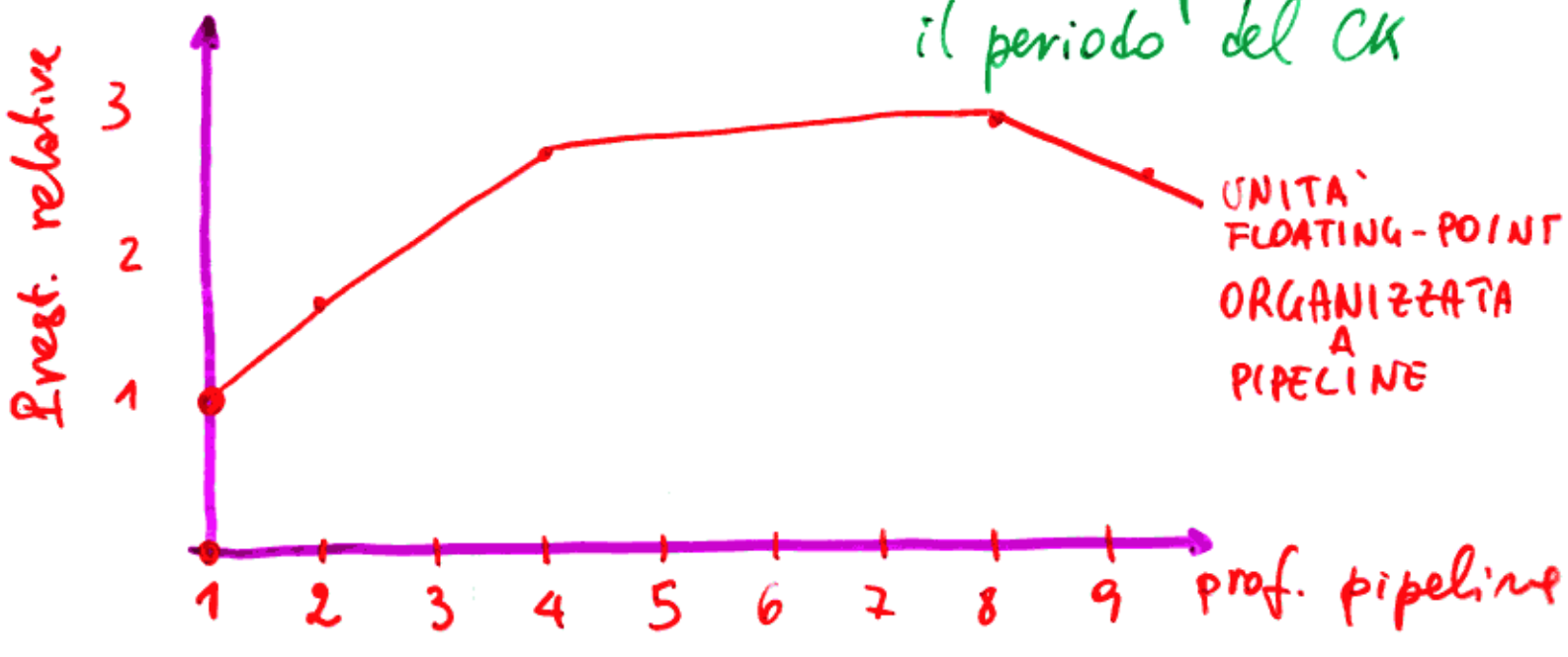
1) più stadi incrementano il numero di conflitti dati e quindi può incrementare il numero di stall

2) salti condizionati "più lenti"

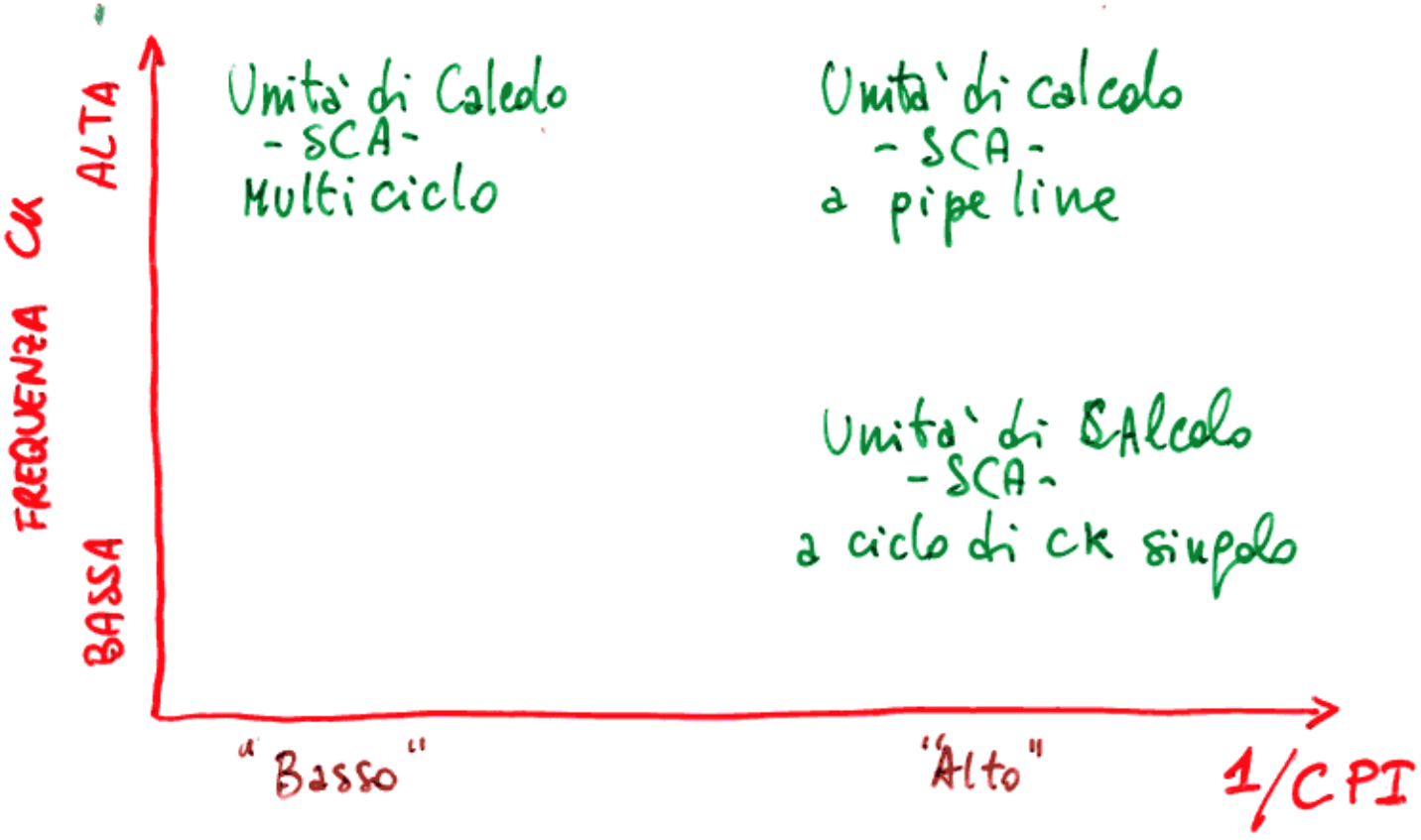
3) più stadi → dimensioni più grandi dell'architettura VLSI

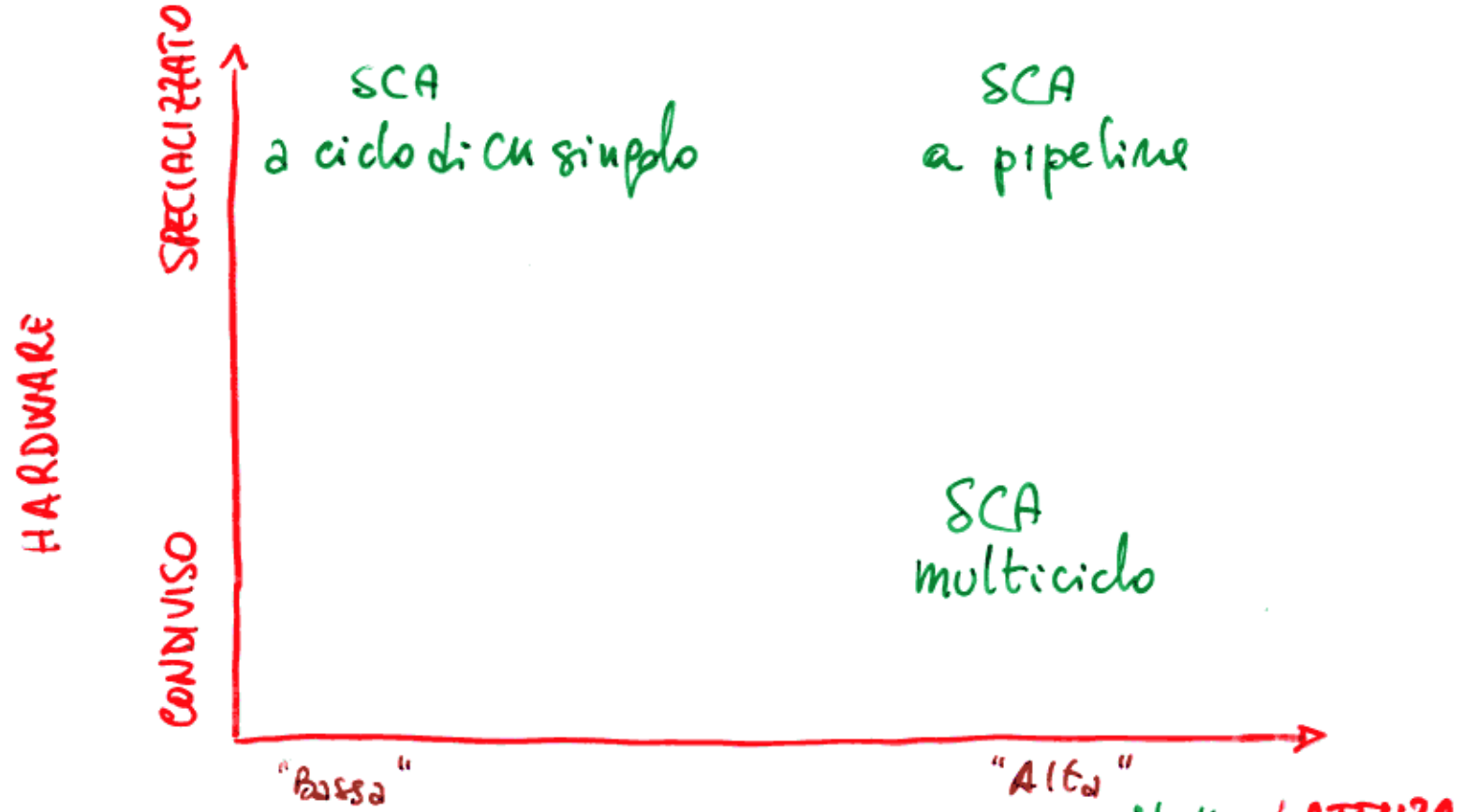
→ sfasamento del CK più grande

→ quindi non si può ridurre a piacimento il periodo del CK



PRESTAZIONI ED ORGANIZZAZIONE DELLA CPU-MEMORIA





n°ck = LATENZA
×istruz. ISTRUZ.

SVILUPPI RECENTI

SUPERSCALARE

esempio di superscalare a due vie

SUPERPIPELINED

esempio di superpipeline ad 8 stadi
p.e. MIPS R4000

SVANTAGGIO: COMPILATORE PIU' COMPLESSO

MIPS SUPERSCALARE A DUE VIE

Tipo di istruzione	Stadi della pipeline							
Istruzione della ALU o salto condizionato	IF	ID	EX	MEM	WB			
Istruzione load o store	IF	ID	EX	MEM	WB			
Istruzione della ALU o salto condizionato		IF	ID	EX	MEM	WB		
Istruzione load o store		IF	ID	EX	MEM	WB		
Istruzione della ALU o salto condizionato			IF	ID	EX	MEM	WB	
Istruzione load o store			IF	ID	EX	MEM	WB	
Istruzione della ALU o salto condizionato				IF	ID	EX	MEM	WB
Istruzione load o store				IF	ID	EX	MEM	WB

Figura 6.57 Funzionamento di una pipeline superscalare. Le istruzioni della ALU e quelle di trasferimento dati sono eseguite contemporaneamente.

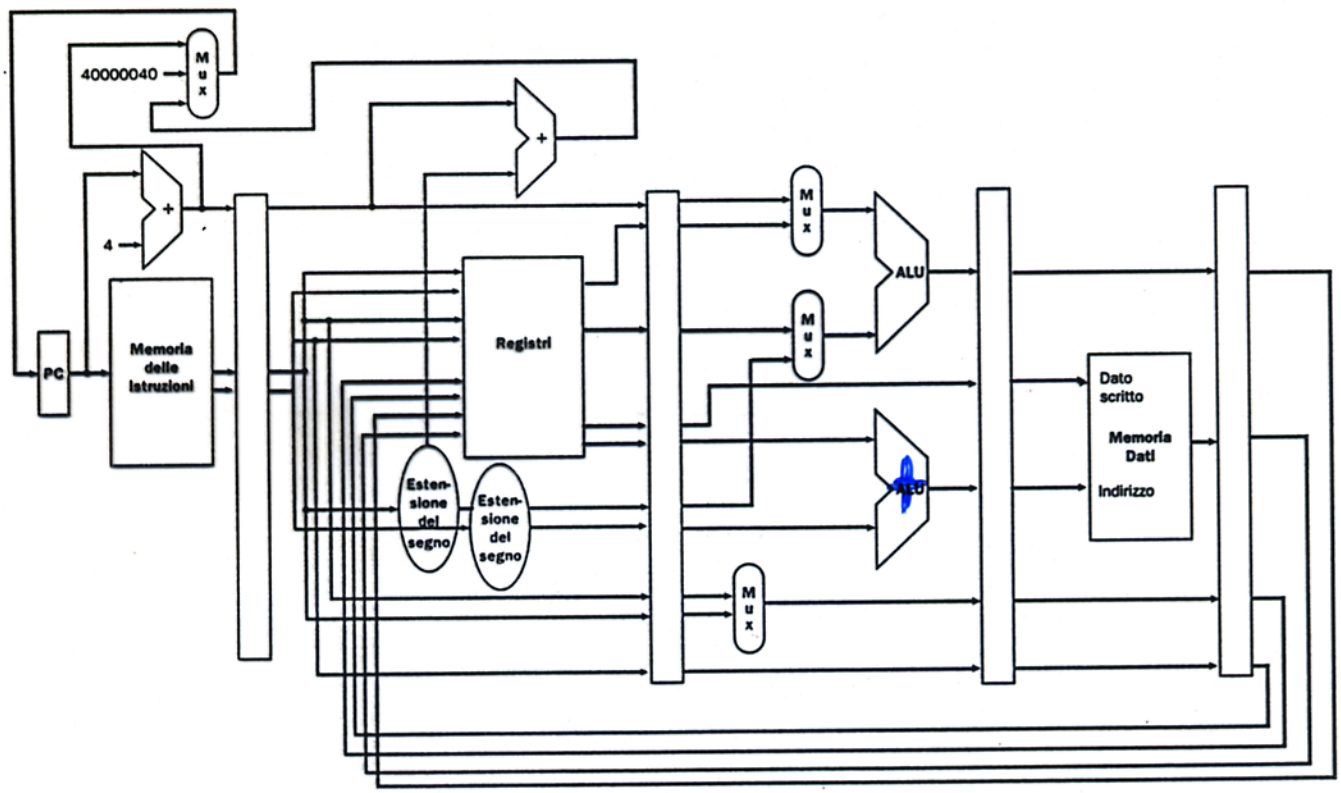


Figura 6.58 Unità di elaborazione superscalare. Sono evidenziate le aggiunte dovute all'implementazione superscalare: 32 bit addizionali dalla memoria delle istruzioni, due porte aggiuntive in lettura ed una in scrittura per il register file, ed un'altra ALU. Si assuma che la ALU in basso si occupi del calcolo degli indirizzi per il trasferimento dati, e che quella in alto si occupi di tutto il resto.

Vantaggi: possibilità di eseguire 2 istruzioni alla volta

Svantaggi: staticità della struttura che necessita di una sequenzializzazione particolare delle istruzioni