Microistruzioni e Unità di controllo microprogrammata

Aniello Minutolo

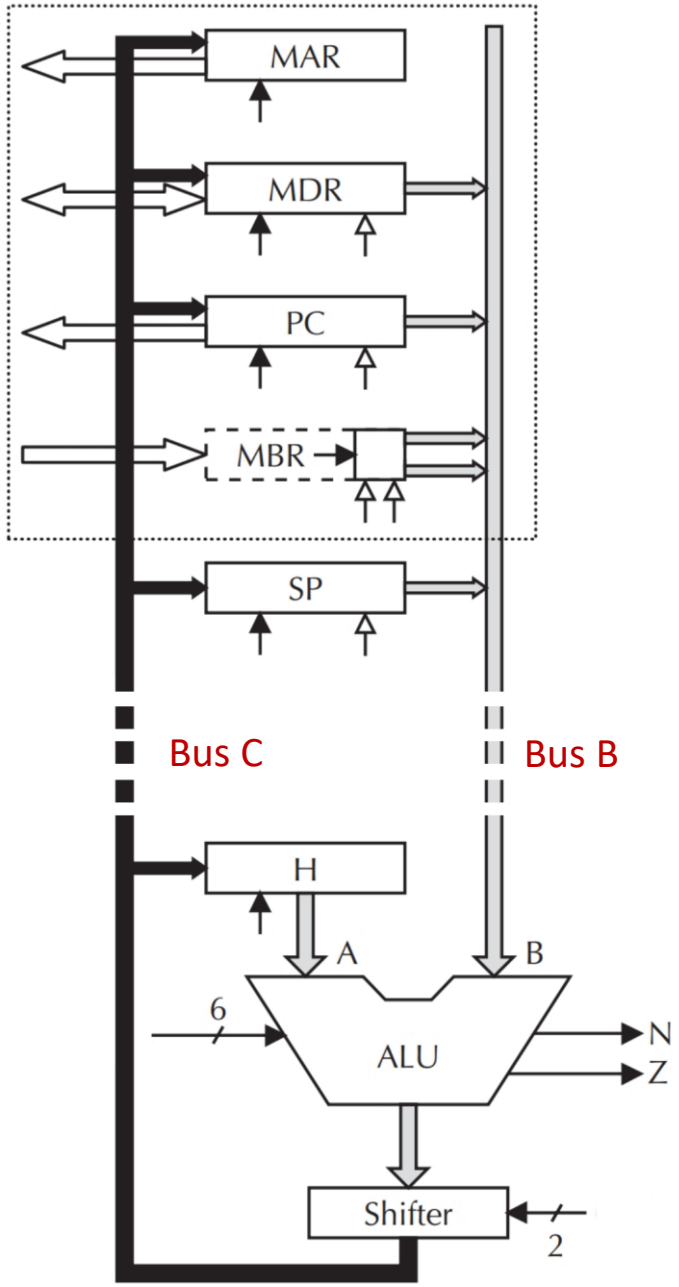
Microistruzioni

Sommario

1. Microistruzioni
2. Unità di controllo microprogrammata: Mic-1
3. Funzionamento di Mic-1

Formato delle microistruzioni

In un processore con architettura CISC dotato di questo data path, **ogni microistruzione richiede 36 bit** per includere tutti i segnali di controllo e le informazioni necessarie per la microistruzione successiva.



Segnali di controllo

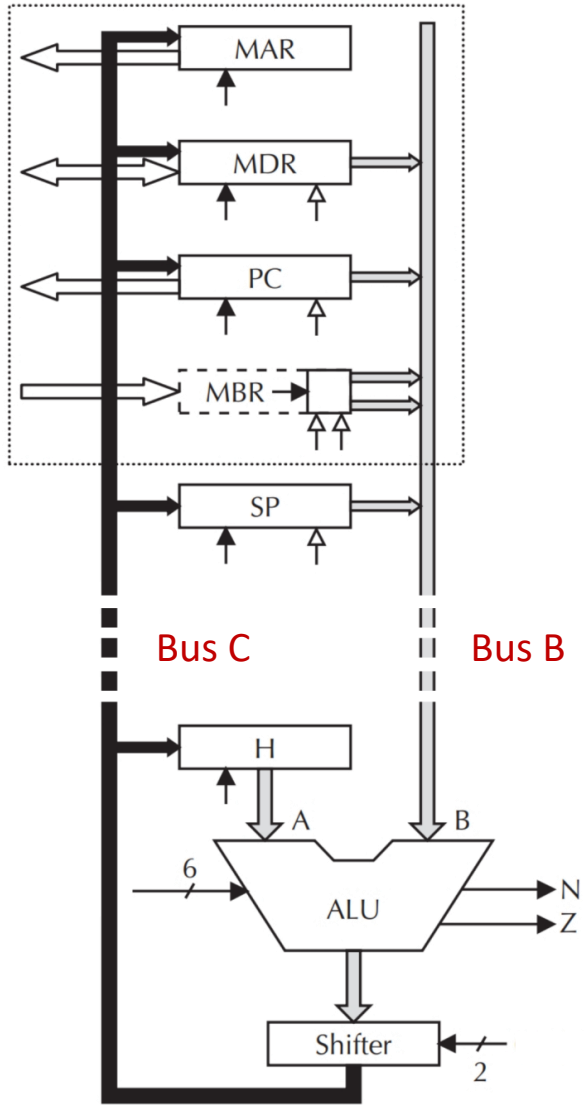
Abilita la scrittura sul bus B

Scrive il bus C nel registro

Segnali di controllo del percorso dati

I **segnali di controllo** del percorso dati sono **29** e si dividono in cinque gruppi funzionali

- 9 segnali per controllare l'abilitazione dei registri per la scrittura dei dati dal bus C.
- 9 segnali per controllare l'abilitazione dei registri sul bus B per l'input della ALU.
- 8 segnali per controllare le funzioni della ALU e dello shifter.
- 2 segnali (non mostrati) per indicare alla memoria di leggere (scrivere) attraverso MAR (MDR).
- 1 segnale (non mostrato) per indicare il prelievo dalla memoria attraverso PC o MBR.

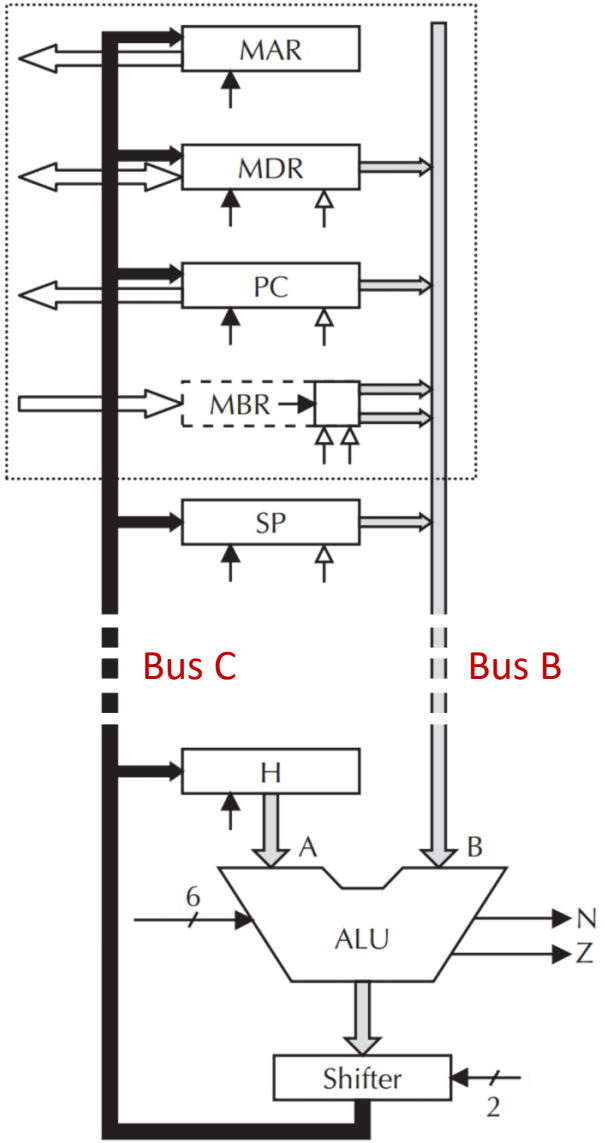


Formato delle microistruzioni

I valori di questi **29 segnali** specificano le operazioni da eseguire durante un ciclo del percorso dati, che consiste in:

- portare i valori dei registri sul **bus B;**
- propagare i segnali attraverso la **ALU** e lo **shifter;**
- guidarli sul **bus C;**
- scrivere i risultati nel registro o nei registri appropriati.

In alcuni casi può essere utile scrivere l'output presente nel bus C in più di un registro, mentre in nessun caso ha senso abilitare nello stesso momento più di un registro sul bus B



Formato delle microistruzioni

Con pochi circuiti aggiuntivi è possibile ridurre il numero di bit necessari per la selezione delle sorgenti che alimentino il bus B

- ci sono soltanto 9 possibili registri di input che possono guidare il bus B (considerando separatamente le versioni con e senza segno di MBR);
- possiamo quindi codificare in 4 bit l'informazione del bus B e utilizzare un decodificatore per generare 16 segnali di controllo, 7 dei quali non vengono utilizzati.

Formato delle microistruzioni

I valori di questi 29 bit controllano il percorso dati soltanto per un ciclo

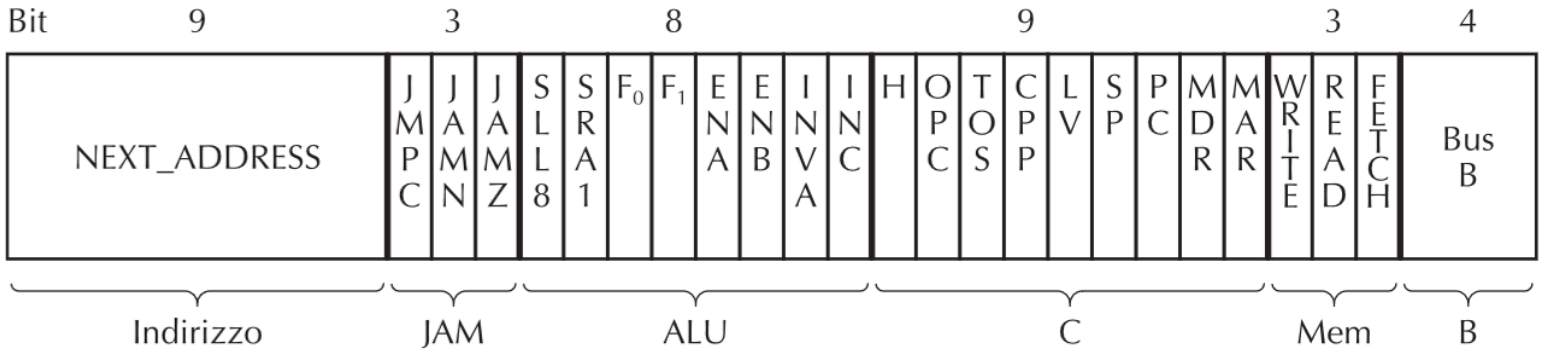
- la seconda parte del controllo consiste nel determinare che cosa deve essere effettuato durante il ciclo successivo.

Il formato delle microistruzioni della nostra macchina di esempio, che chiameremo **Mic-1**, richiede quindi, oltre ai 29 bit di controllo, due campi aggiuntivi: **NEXT_ADDRESS** e **JAM.**

Formato delle microistruzioni

La Figura mostra un possibile formato, diviso in sei gruppi (elencati sotto l'istruzione) e contenente i 36 segnali seguenti

- **Addr** contiene l'indirizzo di una potenziale successiva microistruzione;
- **JAM** determina come viene selezionata la successiva microistruzione;
- **ALU** seleziona le funzioni della ALU e dello shifter;
- **C** seleziona quali registri sono scritti dal bus C;
- **Mem** seleziona la funzione della memoria;
- **B** seleziona la sorgente del bus B.



Unità di controllo microprogrammata: Mic-1

Selezione delle microistruzioni

Il **sequenzializzatore** ha la responsabilità di far avanzare passo passo la sequenza di operazioni necessarie per eseguire una singola istruzione ISA, e decide quale dei segnali di controllo abilitare durante ciascun ciclo.

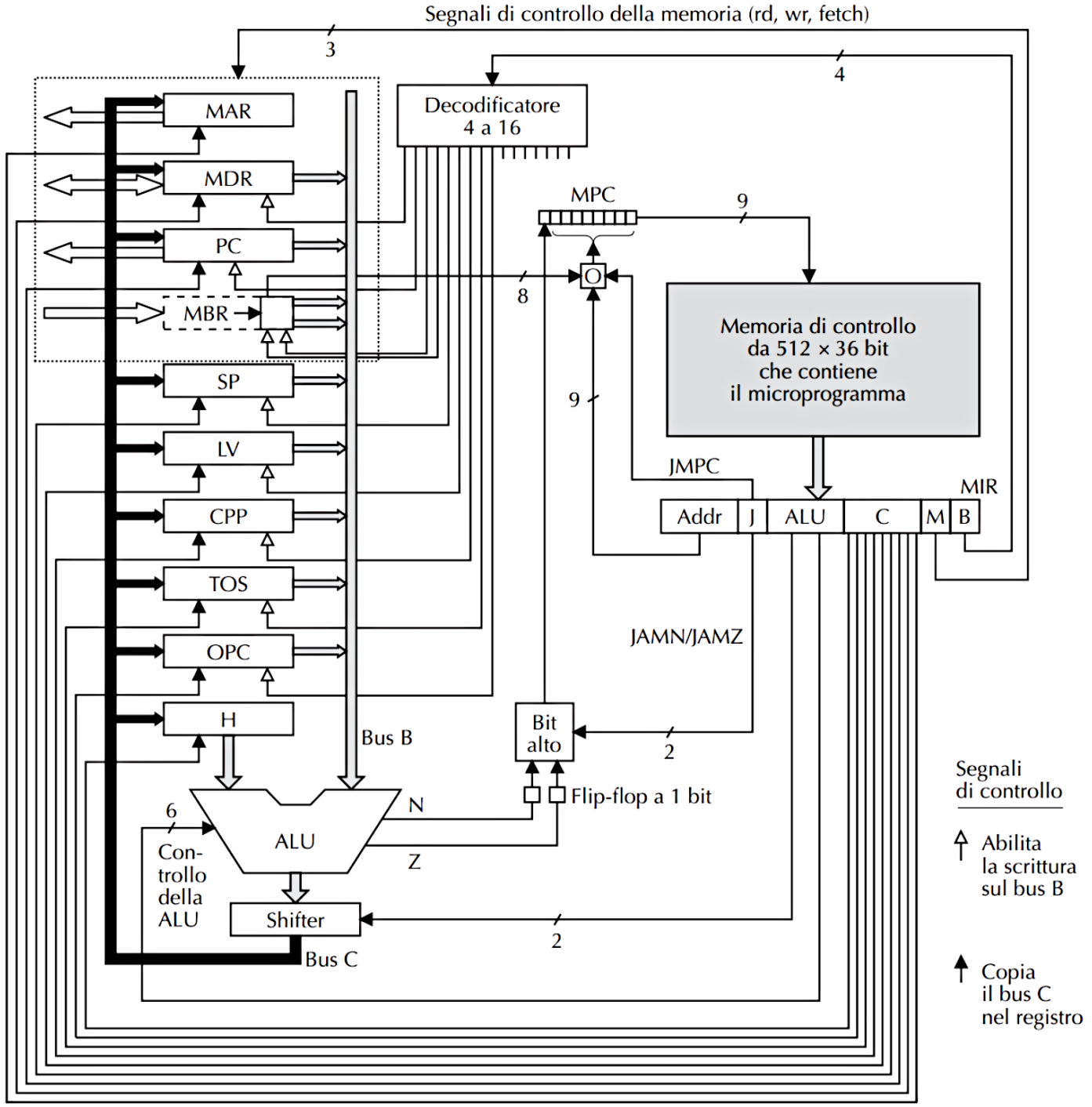
Durante ogni ciclo il **sequenzializzatore** deve produrre due tipi d'informazione:

1. lo stato di ogni segnale di controllo del sistema;
2. l'indirizzo della microistruzione da eseguire subito dopo.

Diagramma a blocchi dettagliato

Il diagramma a blocchi dettagliato del nostro esempio di microarchitettura di **Mic-1** è composto da **due parti principali**

- **percorso dati**, sulla sinistra;
- **sezione di controllo**, sulla destra.

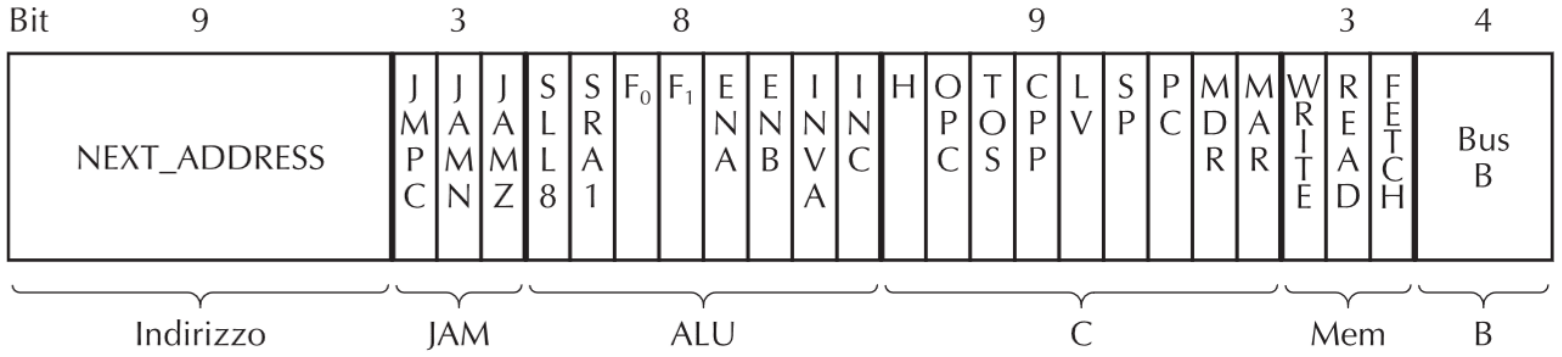


La memoria di controllo

La **memoria di controllo** è una memoria ROM interna alla CPU e non accessibile dall'esterno che contiene le microistruzioni che compongono i microprogrammi che codificano le istruzioni ISA.

La **memoria di controllo** del nostro esempio contiene 512 parole di 36 bit ciascuna (lunghezza microistruzione):

- possiamo quindi memorizzare un insieme di microprogrammi la cui lunghezza totale non supera le 512 microistruzioni.



La memoria di controllo

La memorizzazione dei microprogrammi nella **memoria di controllo** differisce in modo significativo da quella dei programmi assembler (istruzioni ISA) nella memoria principale.

Le **istruzioni ISA** sono in genere eseguite nello stesso ordine in cui sono memorizzate, tranne per i salti condizionati o incondizionati:

- per questo motivo, il registro **PC** viene normalmente incrementato di un'unità per indicare la prossima istruzione da eseguire.

I **microprogrammi** invece richiedono maggiore flessibilità, poiché le sequenze di microistruzioni devono essere il più brevi possibile:

- per questo, ogni microistruzione specifica esplicitamente la successiva, che può trovarsi in qualsiasi posizione della memoria di controllo.

La memoria di controllo

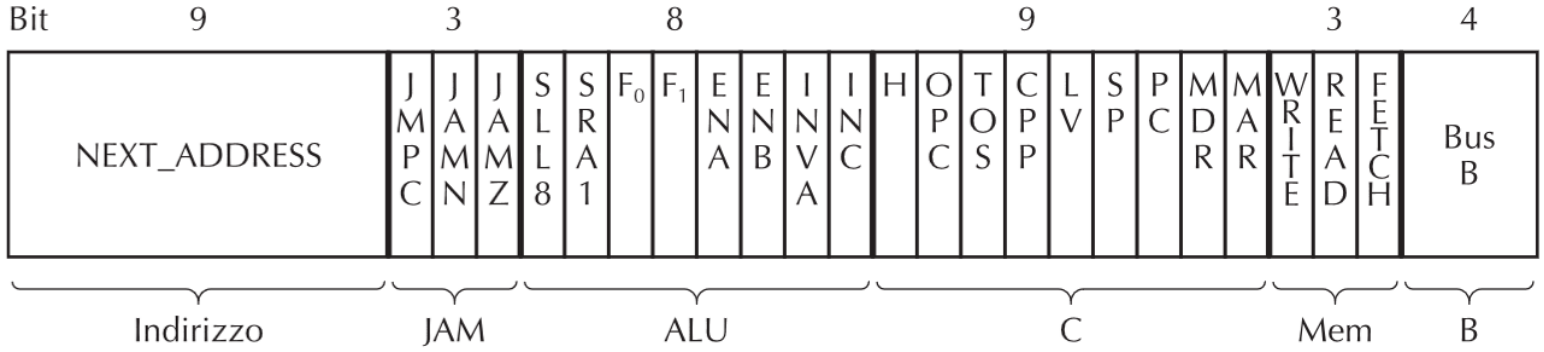
Il control store è collegato al resto della CPU attraverso due registri:

- il registro **MPC** (MicroProgram Counter) indica l'indirizzo della prossima microistruzione da eseguire. Ha una dimensione di **9 bit**, permettendo l'indirizzamento di **512 parole;**
- il registro **MIR** (MicroInstruction Register) contiene la microistruzione corrente, i cui bit controllano i segnali che attivano il *data path*. La sua lunghezza è di **36 bit.**

La memoria di controllo

I segnali del registro MIR sono organizzati in gruppi:

- **Addr** e **JAM** controllano la selezione della microistruzione successiva, mentre **ALU** seleziona le funzioni della ALU e dello shifter.
- **C** indica in quali registri viene caricato l'output della ALU, mentre **Mem** controlla le operazioni della memoria.
- **B** utilizza un decodificatore da 4 a 16 bit per selezionare il registro da abilitare sul bus B, anche se solo 9 segnali sono effettivamente utilizzati.

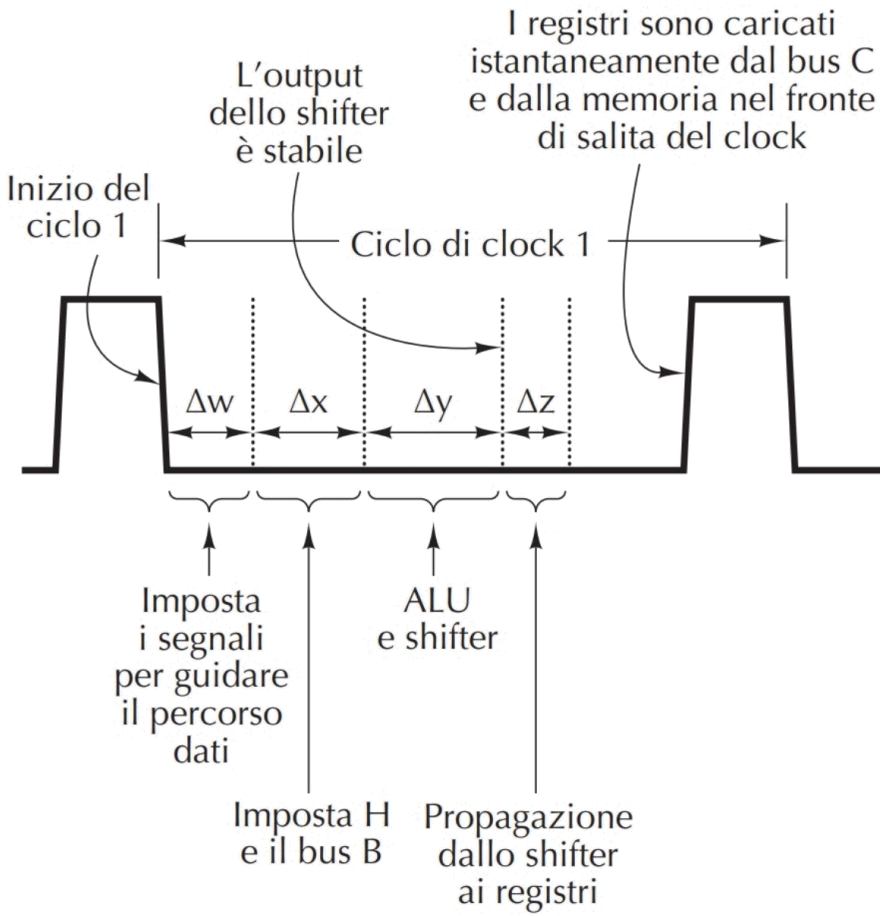


Funzionamento di Mic-1

Primo sottociclo del data path

All'avvio di un ciclo di clock (sul fronte di discesa), la parola contenuta nella memoria di controllo e puntata da **MPC** viene trasferita in **MIR**

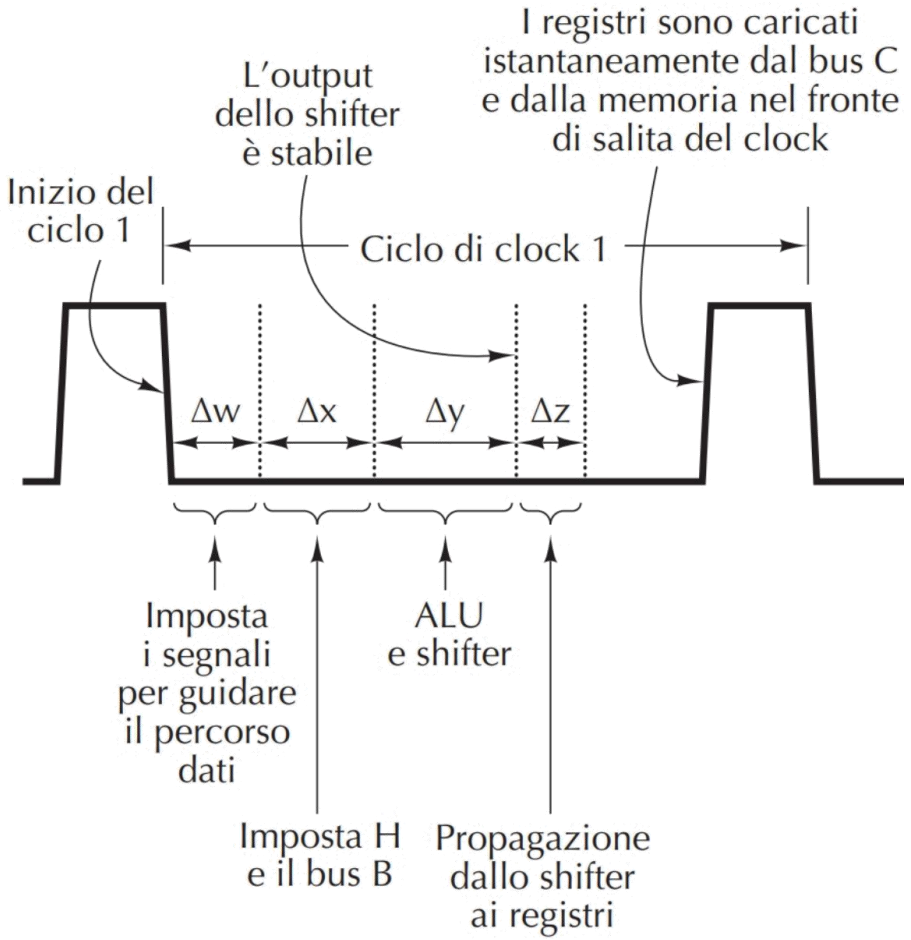
- l'operazione di caricamento deve completarsi entro Δw
- MIR viene caricato quindi durante il primo sottociclo del data path



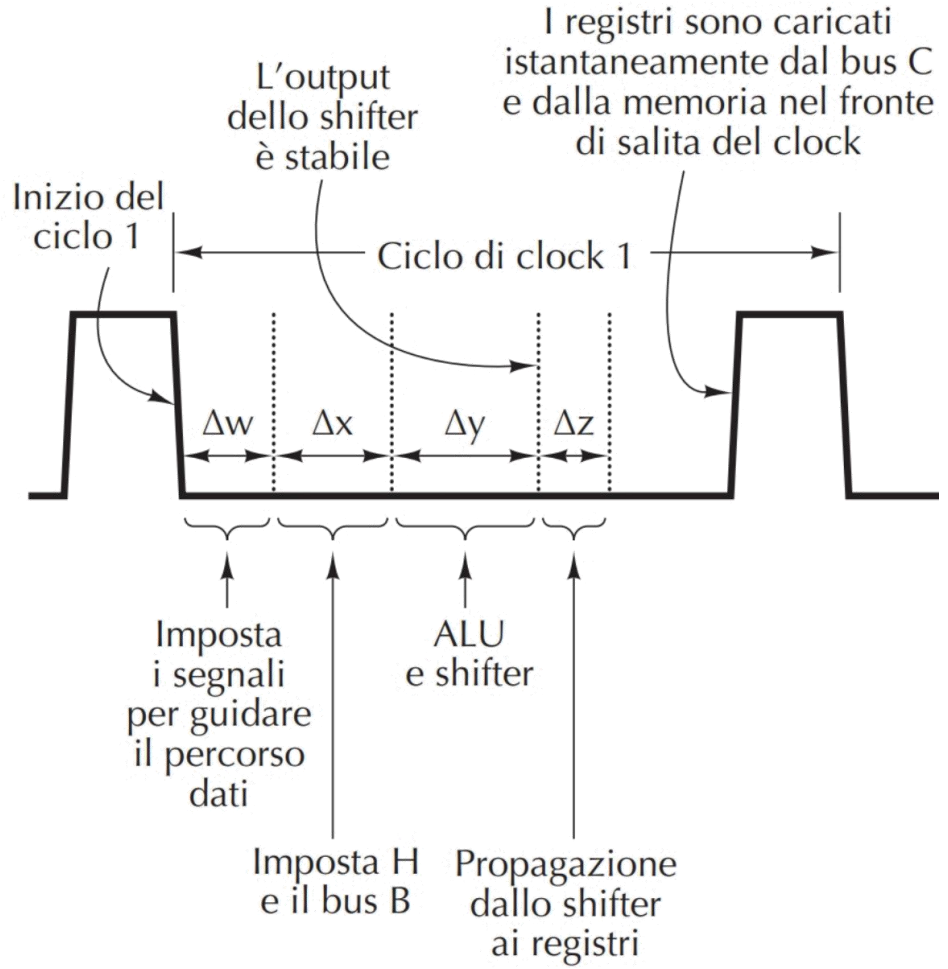
Secondo sottociclo del data path

Dopo Δw, i vari segnali si propagano all'interno del percorso dati

- H e uno dei registri (attraverso il bus **B**) vengono inviati alla **ALU** che sa quale operazione deve eseguire;
- seguono poi varie azioni, fino ad arrivare al secondo sottociclo;
- entro Δw + Δx dall'inizio del ciclo, gli input della ALU diventano stabili.



Terzo sottociclo del data path



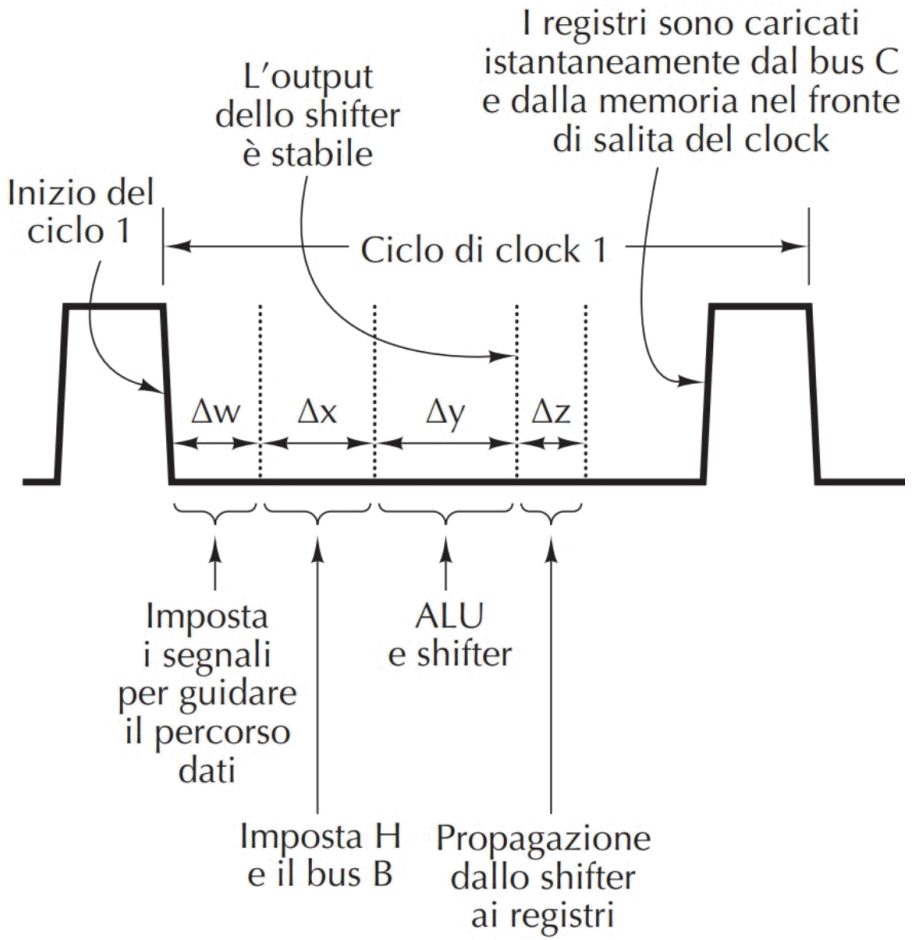
La ALU calcola la funzione richiesta e, dopo Δy, tutto nel circuito si stabilizza, compresi gli output della **ALU**, di **N**, di **Z** e dello **shifter**

- l'output della ALU non viene memorizzato, ma semplicemente inserito nello shifter;
- le attività della ALU e dello shifter si svolgono durante il sottociclo 3.

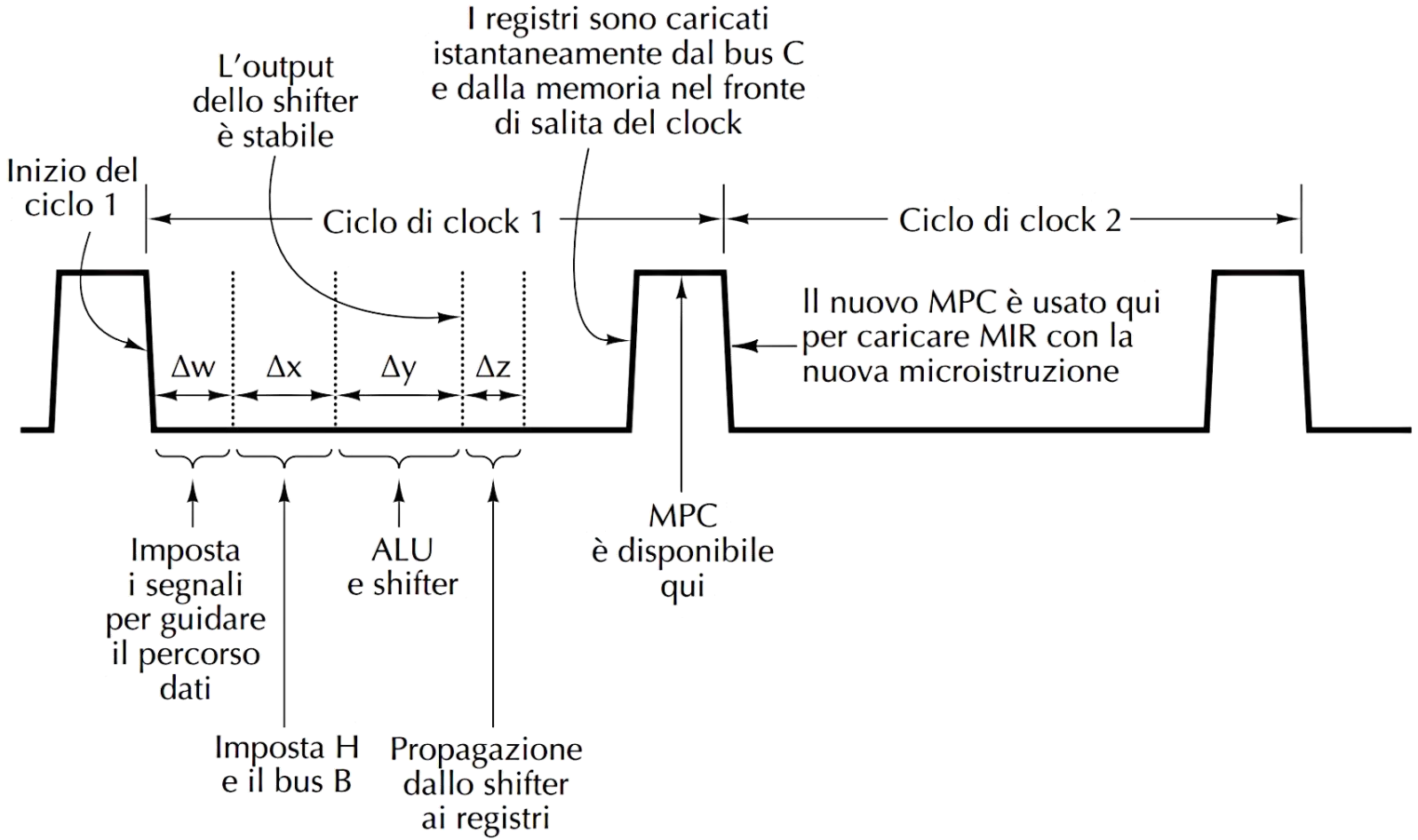
Quarto sottociclo del data path

Dopo un ulteriore intervallo di tempo Δz, l'output dello shifter raggiunge i registri attraverso il bus C

- I registri possono successivamente essere caricati sul fronte di salita del clock;
- nel sottociclo 4 vengono caricati i registri e i flip-flop N e Z.



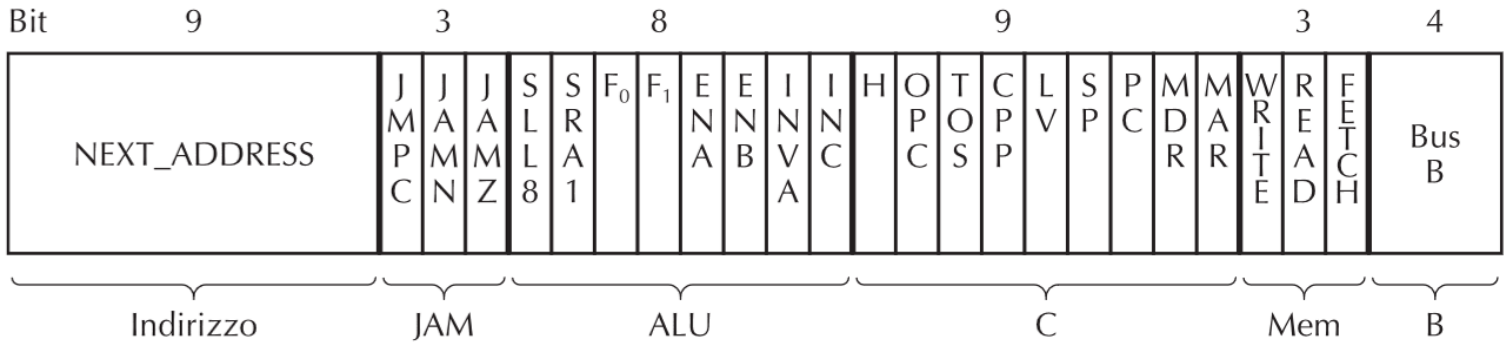
Quarto sottociclo del data path



Il sottociclo termina leggermente dopo il fronte di salita del clock, quando tutti i risultati sono stati salvati, e MPC è stato caricato tutto il ciclo continua ripetutamente finché la macchina resta accesa.

Calcolo dell'indirizzo successivo

- Il microprogramma, oltre a guidare il percorso dati, deve parallelamente determinare quale sarà la microistruzione successiva dato che non è necessario che esse vengano eseguite nello stesso ordine in cui appaiono nella memoria di controllo.
- Il calcolo dell'indirizzo della microistruzione successiva comincia dopo che MIR è stato caricato ed è diventato stabile, copiando i 9 bit del campo NEXT_ADDRESS vengono copiati all'interno di MPC.



Calcolo dell'indirizzo successivo

Mentre si svolge questa copia viene ispezionato il campo JAM

- se JAM == 000, allora non viene eseguita alcuna azione aggiuntiva, e l’indirizzo della prossima microistruzione diviene semplicemente quello indicato da NEXT_ADDRESS;
- se invece uno o più bit di JAM valgono 1, allora è necessario compiere delle azioni per calcolare l’indirizzo della prossima microistruzione.

Calcolo dell'indirizzo successivo

Quando JAMN e/o JAMZ sono pari a 1 abilitano, rispettivamente, l'utilizzo dei flip-flop N e Z per impostare il bit più significativo di MPC.

In dettaglio, la funzione booleana che calcola il bit più significativo di MPC è quindi seguente:

- F = (JAMZ AND Z) OR (JAMN AND N) OR NEXT_ADDRESS[8]

Quindi, MPC può assumere soltanto uno di questi due valori.

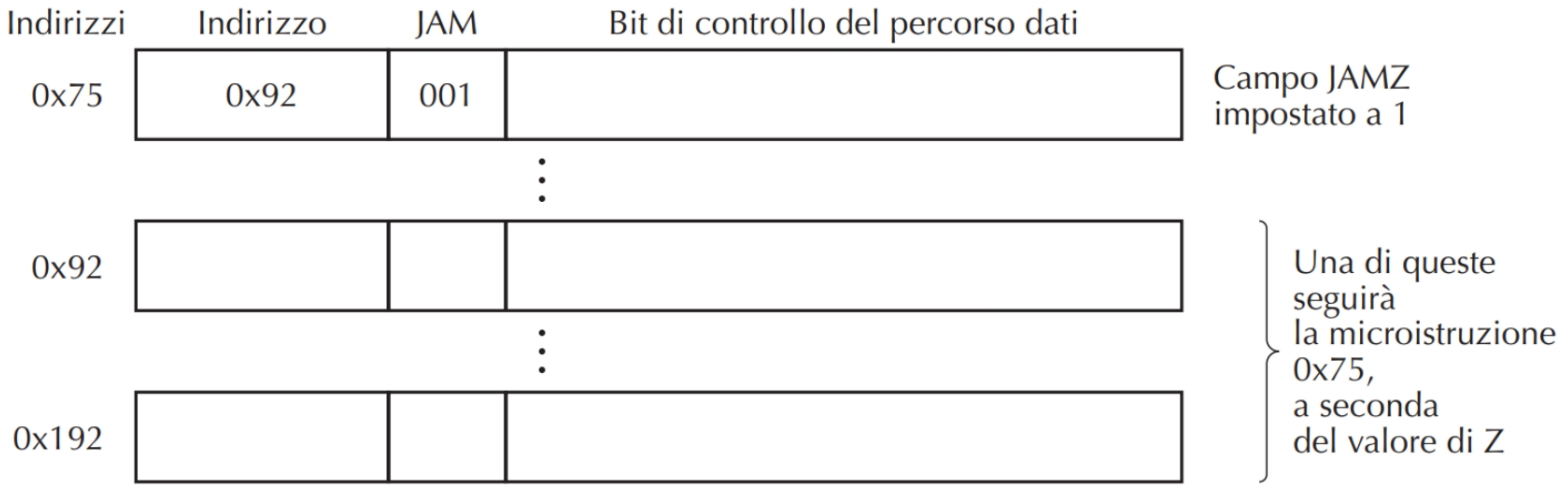
1. NEXT_ADDRESS
2. NEXT_ADDRESS, in cui il bit più significativo è posto a 1.

Osservazione

La ragion per cui sono necessari i flip-flop N e Z è che, dopo il fronte di salita del clock (mentre il clock è ancora alto), il bus B non viene più alimentato e quindi gli output della ALU non possono più essere considerati corretti:

- salvando in N e Z i flag di stato della ALU è possibile rendere stabili questi valori, per poterli utilizzare per calcolare correttamente MPC, indipendentemente dalle operazioni che la ALU compie nel frattempo.

Calcolo dell'indirizzo successive: esempio



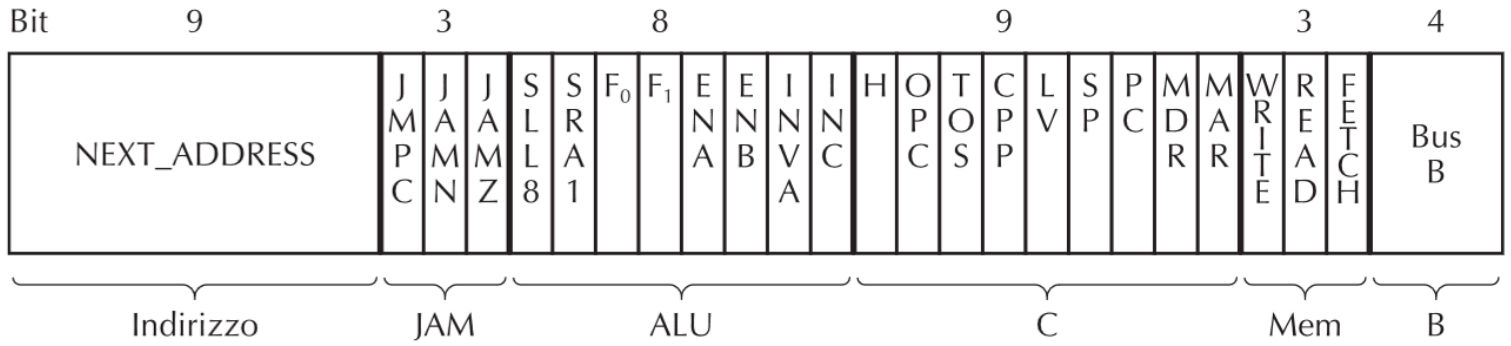
Esempio

- la microistruzione corrente, che si trova nella locazione 0x75, ha il campo NEXT_ADDRESS = 0x92 e JAMZ impostato a 1;
- di conseguenza l'indirizzo della microistruzione successiva dipende dal bit Z memorizzato durante la precedente operazione della ALU;
- se il bit Z vale 0, la microistruzione successiva comincerà a partire dall'indirizzo 0x92; altrimenti, l'indirizzo sarà 0x192.

Uso di JMPC per diramazioni

Il terzo bit del campo JAM è **JMPC**. Se **JMPC** è abilitato, gli 8 bit di MBR vengono combinati in OR con gli 8 bit meno significativi di NEXT_ADDRESS per determinare l'indirizzo successivo

- questa modalità è utile per implementare diramazioni efficienti basate sui codici operativi memorizzati in MBR;
- JMPC permette di saltare a un indirizzo specifico in base al valore di MBR, facilitando l'esecuzione di microistruzioni associate a codici operativi.



Bibliografia

Libro di testo

- Andrew S. Tanenbaum e Todd Austin. Architettura dei calcolatori. Un approccio strutturale. 6/ED. Anno 2013. Pearson Italia spa. (Disponibile nella sezione “Biblioteca”)

Fonte argomenti e immagini

- Capitolo 4, Paragrafi da 4.1.2 a 4.1.3, pp. 257-264