1 L'ARCHITETTURA DELLA CPU

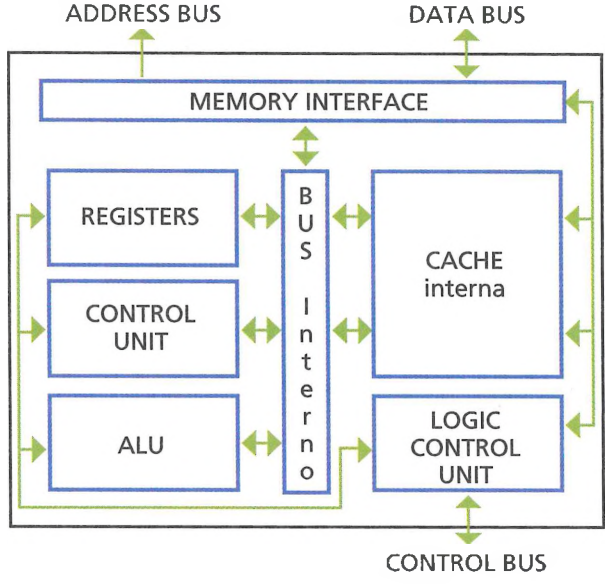
1.1 La CPU

FIGURA 1 Chip dei microprocessori Intel di decima generazione

L'elemento fondamentale di un sistema di elaborazione, come proposto da von Neumann, è la **CPU** (Central Processing Unit). Questa viene realizzata fisicamente attraverso un **microprocessore** (FIGURA 1) che può essere **multicore**, cioè costituito da più CPU uguali in un unico componente integrato.

Nel corso dei decenni l'evoluzione delle CPU è stata enorme in termini di capacità di elaborazione e velocità, ma la struttura interna, cioè l'architettura, ha mantenuto le caratteristiche e le funzionalità evidenziate nella FIGURA 2.

FIGURA 2 Schema a blocchi della CPU



Si possono individuare i seguenti blocchi:

- **CU** (Control Unit): coordina e gestisce le operazioni interne dei vari blocchi in base ai segnali ricevuti dall'esterno e alle istruzioni da eseguire;
- **ALU** (Arithmetic Logic Unit): esegue tutte le operazioni logico-matematiche necessarie richieste dall'unità di controllo;
- **REGISTERS**: piccole aree di memoria (le dimensioni massime dipendono da quanti bit può elaborare simultaneamente l'unità di controllo) molto veloci che conservano i dati da elaborare e le informazioni relative alle operazioni da eseguire durante l'esecuzione delle istruzioni;
- **CACHE interna**: area di memoria nella quale sono inserite le istruzioni successive a quella in corso di esecuzione, velocizzando così le operazioni; è organizzata su più livelli (L1, L2, ecc.) in base alla velocità di accesso e alla frequenza d'uso;
- **LCU** (Logic Control Unit): insieme di circuiti che trasformano gli impulsi elettrici provenienti dall'esterno in segnali utili per l'unità di controllo e trasformano in impulsi elettrici i comandi provenienti dall'unità di controllo;
- **MI** (Memory Interface): insieme di circuiti che si occupa di fornire ai bus esterni di comunicazione (dati e indirizzi) gli impulsi necessari per le comunicazioni con

la memoria e con le periferiche e di trasformare gli impulsi ricevuti dall'esterno in segnali utili per l'unità di controllo;
- **BUS interno** (o **CPU bus**): insieme di collegamenti elettrici che consente di trasferire dati e indirizzi tra i vari blocchi del microprocessore.

1.2 I bus

Come abbiamo visto nella Lezione 3 dell'Unità 1, la CPU dialoga con le memorie e con le periferiche attraverso tre tipi di bus esterni che costituiscono il **System bus**:
- **Control bus**: connessioni sui cui sono trasferiti i comandi (*read*, *write* ecc.) della CPU ai dispositivi periferici e le richieste dei dispositivi periferici alla CPU;
- **Address bus**: connessioni su cui sono trasferiti gli indirizzi;
- **Data bus**: connessioni su cui sono trasferiti i dati.

Il **Control bus** è costituito da un insieme di segnali logici. Alcuni di questi segnali escono dalla CPU e quindi sono comandi che la CPU invia agli altri dispositivi mentre altri entrano nella CPU e quindi sono informazioni che gli altri dispositivi inviano alla CPU. Il compito fondamentale del Control Bus è quello di sincronizzare i trasferimenti che avvengono su tutti i bus esterni in base al clock del sistema.
L'**Address bus** è l'insieme di linee su cui la CPU scrive l'indirizzo della cella di memoria a cui vuole accedere (in lettura o in scrittura).
Il **Data bus** è, invece, l'insieme di linee su cui la CPU o la memoria scrivono il dato che deve essere trasferito:
- da CPU a memoria: la CPU scrive sul bus e la memoria legge dal bus;
- da memoria a CPU: la memoria scrive sul bus e la CPU legge dal bus.

In molti microprocessori è presente un unico bus temporizzato per il trasferimento di dati e indirizzi (**Address/Data bus**).
La dimensione dell'Address/Data bus spesso coincide con la dimensione dei registri e con la dimensione massima delle istruzioni macchina e dei tipi di dati previsti. Quando questo si verifica, la circuiteria integrata è meno complessa.
Un'architettura descritta come "a 64 bit" ha generalmente i registri del processore da 64 bit e gestisce dati di questa dimensione, sia internamente sia esternamente, attraverso bus anch'essi a 64 bit.

Una CPU può essere a 64 bit internamente (registri e bus interno), ma il suo Address/Data bus può avere dimensioni differenti, maggiori o minori.
Il numero di bit in questo caso esprime la quantità di dati che il processore è in grado di elaborare contemporaneamente (per esempio, 64 bit per volta), ma offre indicazioni anche sulla velocità di elaborazione dei dati e la capacità massima della RAM utilizzabile.
A seconda del contesto, con il termine architettura si intende l'**architettura hardware** (principalmente il microprocessore con i suoi registri e bus) oppure l'**architettura software** (principalmente il Sistema Operativo).
Una macchina con architettura effettivamente a 64 bit deve avere microprocessore, bus e Sistema Operativo a 64 bit.

Un computer può comunque funzionare con queste combinazioni:
- processore a 32 bit e Sistema Operativo a 32 bit;
- processore a 64 bit e Sistema Operativo a 32 bit;
- processore a 64 bit e Sistema Operativo a 64 bit.

L'ultima combinazione è potenzialmente la più performante.

#prendinota

Maggiori sono i bit su cui si basa l'architettura, maggiori sono le possibili istruzioni codificabili e maggiori risultano le celle di memoria indirizzabili. Con 64 bit si possono indirizzare 2^{64} (=18 446 744 073 709 551 616) celle diverse (memoria virtuale).

Anche i programmi (software applicativi) devono essere compatibili con il Sistema Operativo: i programmi a 32 bit sono adatti ai Sistemi Operativi a 32 bit, i programmi a 64 bit sono adatti ai Sistemi Operativi a 64 bit.
La maggior parte dei programmi a 32 bit funziona anche in Sistemi Operativi a 64 bit. Si possono avere problemi di incompatibilità se si tratta di antivirus, utility di sistema o programmi che utilizzano driver incorporati.

I REGISTRI

Un **registro** è una piccola e veloce memoria interna alla CPU.
Anche i registri hanno attraversato un'evoluzione negli anni e ne sono aumentati il numero e la dimensione, crescendo con le potenze del 2: inizialmente registri a 8 bit, poi 16, 32, fino allo standard attuale a 64 bit.
I registri fondamentali, presenti in tutte le CPU (pur con nomi differenti a seconda dell'azienda produttrice di microprocessori), sono rappresentati nella FIGURA 3.

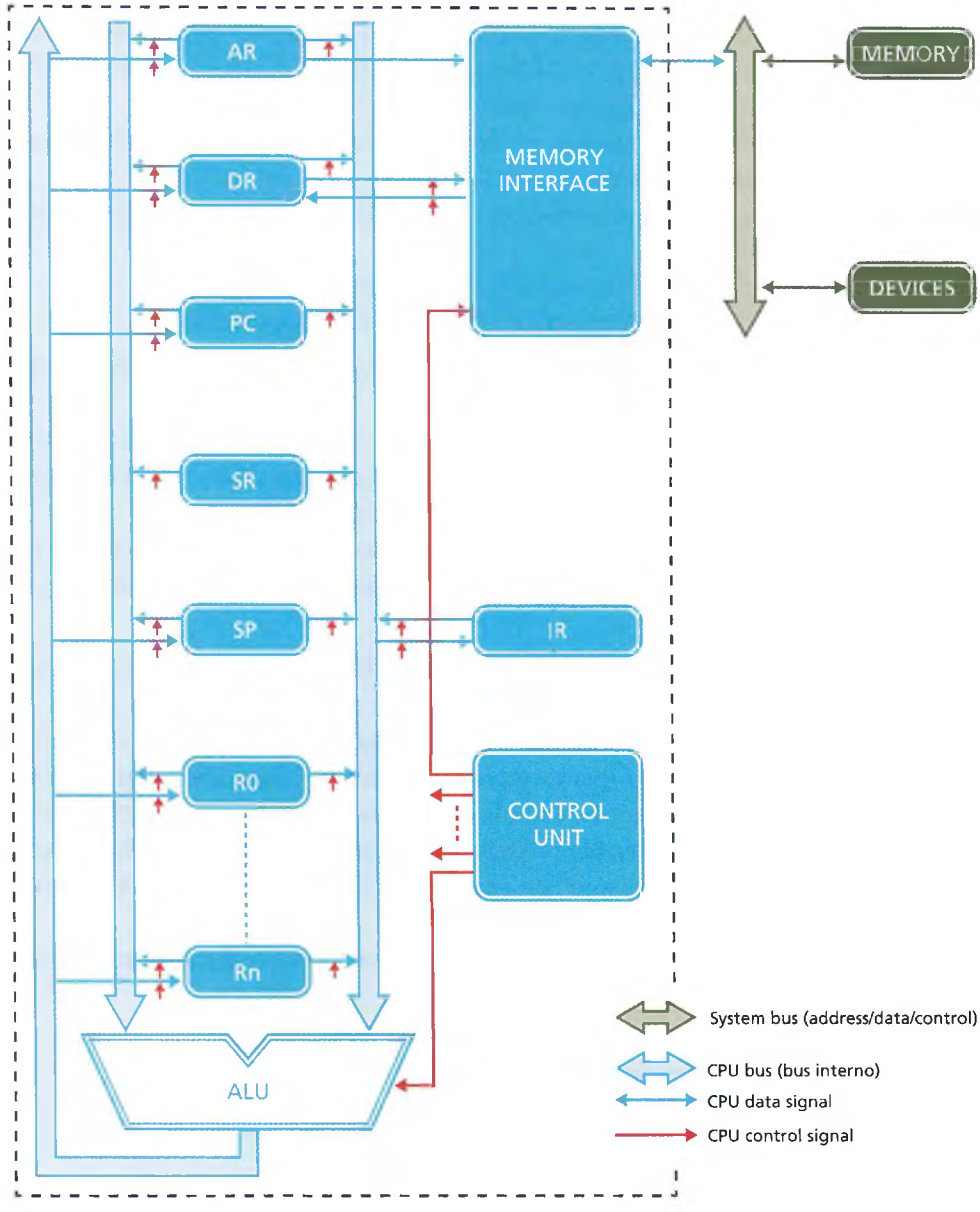


FIGURA 3 Architettura di una generica CPU con registri fondamentali

Le linee rosse rappresentano i bus su cui viaggiano i segnali di controllo, mentre le linee blu rappresentano i bus su cui viaggiano dati e indirizzi.

Vediamo lo scopo di ciascun registro.

- **AR** (Address Register): memorizza gli indirizzi per gli accessi in memoria (dunque solo dalla CPU verso la memoria).
- **DR** (Data Register): memorizza i dati provenienti:
  – dalla memoria diretti alla CPU;
  – dalla CPU diretti alla memoria.
- **IR** (Instruction Register): memorizza il codice operativo (**opcode**) dell'istruzione da eseguire. Sulla base dell'opcode, l'unità di controllo capisce quale operazione deve essere eseguita e ne comanda l'esecuzione.
- **PC** (Program Counter): memorizza l'indirizzo iniziale della prossima istruzione da eseguire.
- **SR** (Status Register): memorizza, tramite una serie di *flag*, lo stato del processore successivo all'esecuzione dell'ultima operazione. I suoi bit assumono valore 0 o 1 in base al risultato delle operazioni svolte dal processore. Approfondiremo i vari flag del registro di stato nella Lezione 6, dedicata al linguaggio assembly dei processori Intel x86.
- **SP** (Stack Pointer): memorizza l'indirizzo top dello stack. Lo **stack** è un'area della memoria in cui i dati sono letti/scritti in modalità **Last-In-First-Out** (LIFO). Tipicamente si usa per salvare l'indirizzo di ritorno dalle subroutine (funzioni o procedure) chiamate dal **main program**. L'ultima subroutine chiamata è la prima a essere finita e per tornare al main program serve l'indirizzo di ritorno, salvato in cima alla pila (stack) e puntato da SP. Affronteremo le subroutine e lo stack nella Lezione 6 di questa unità e nella Lezione 9 della prossima unità.
- **R0, ..., Rn: registri di lavoro** (o **registri generali**) che memorizzano i risultati temporanei in ingresso e in uscita dall'ALU.

Il set di registri della CPU può comprendere anche registri per contenere indirizzi iniziali (base register), contatori (count register), indici (index register), puntatori (pointer register) e altro ancora.

FISSA LE CONOSCENZE

- Quali sono i blocchi fondamentali che costituiscono la CPU?
- Che cosa si intende per architettura hardware?
- Che cosa si intende per architettura software?
- Qual è la funzione del registro AR (*Address Register*)?
- Qual è la funzione del registro PC (*Program Counter*)?
- A che cosa servono i registri di lavoro?

2 IL CICLO MACCHINA

La CPU ha il compito di eseguire una sequenza di istruzioni (un programma). Prima preleva l'istruzione insieme a eventuali dati dalla memoria esterna (la RAM) e carica tutto negli opportuni registri. Poi l'ALU elabora l'istruzione e mette a disposizione il risultato in un registro dedicato o in memoria.
Tutto questo rappresenta un **ciclo macchina** (FIGURA 4) o **fetch-execute cycle**.

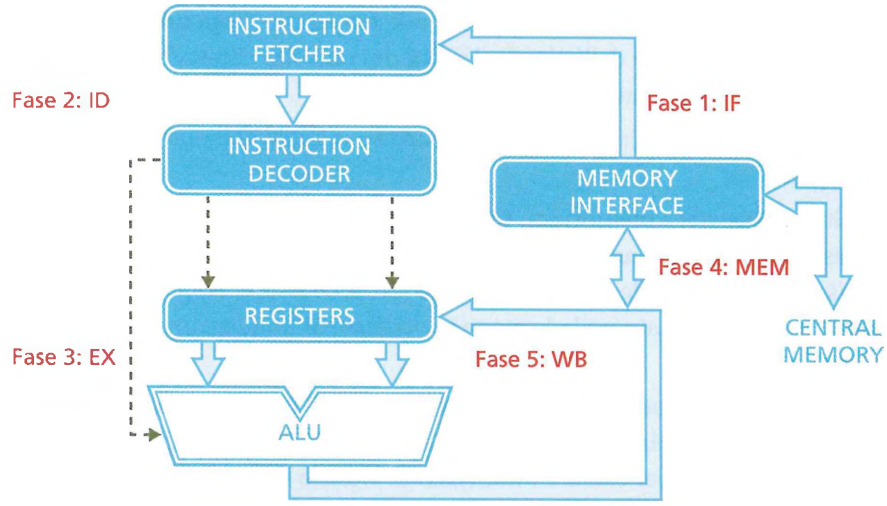


FIGURA 4 Schema a blocchi del ciclo macchina

Le fasi durante le quali la CPU compie un intero ciclo sono cinque e si ripetono in sequenza fino al termine del programma:

1. **IF** (Instruction Fetch): lettura dell'istruzione da memoria;
2. **ID** (Instruction Decode): decodifica istruzione e lettura dati dai registri;
3. **EX** (Execution): esecuzione dell'istruzione;
4. **MEM** (Memory): scrittura del risultato in memoria (solo per certe istruzioni);
5. **WB** (Write Back): scrittura del risultato nel registro opportuno e aggiornamento dello stato.

Gli attuali microprocessori superano la suddivisione delle istruzioni in cinque fasi. È possibile frazionare le fasi del ciclo in sequenze di **micro-operazioni**, ognuna di complessità comparabile alle altre, che sono ripetute identicamente su flussi continui di dati.
Ottimizzando i microprocessori, ogni micro-operazione è eseguita in un ciclo di clock.

Un ciclo macchina è così costituito da una sequenza di fasi, ognuna a sua volta costituita da micro-operazioni eseguite dalla CPU. La durata di una micro-operazione prende il nome di **cycle time** ed è comunemente indicata con t_{CPU}.

esempio

Frequenza di clock = 2 GHz
t_{CPU} = 0,5 ns (nanosecondi).

Il valore $\mathbf{1/t_{CPU}}$ è la **frequenza di clock** della CPU, misurata in gigahertz (GHz).
Nella FIGURA 5 sono mostrate tre istruzioni eseguite in sequenza, in cui ogni intervallo dell'asse del tempo rappresenta un t_{CPU}, che per semplicità supponiamo corrispondere alla durata di una fase (anche se sarebbe più realistico che corrispondesse alla durata di una delle micro-operazioni in cui è suddivisa ogni fase).
Se ogni fase ha un t_{CPU} di 0,5 ns, occorreranno 7,5 ns (3 istruzioni × 5 fasi × 0,5 ns) per eseguirle.

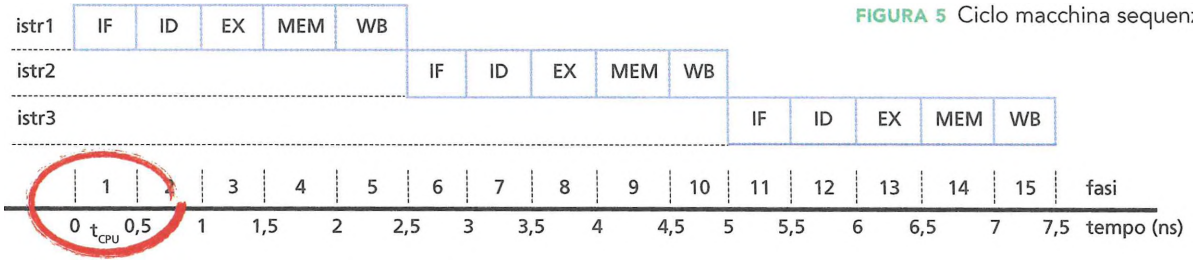


FIGURA 5 Ciclo macchina sequenziale

Il flusso di esecuzione dei cicli macchina coinvolge tutti i **registri** che abbiamo visto nella precedente lezione.
Il flowchart nella FIGURA 6 a pagina seguente mostra i passi principali del ciclo e i registri coinvolti (i blocchi tratteggiati indicano azioni non obbligatoriamente realizzate a ogni ciclo).

FASE 1: IF

- L'indirizzo della prima istruzione viene caricato nel Program Counter e poi da questo all'Address Register.
- L'istruzione è prelevata (fetch) dalla memoria e caricata nell'Instruction Register (anche eventuali dati vengono caricati nel Data Register).
- Il Program Counter viene incrementato al fine di puntare alla successiva istruzione da eseguire. La posizione dell'istruzione successiva è (salvo nelle istruzioni di salto) implicita; il programma è eseguito in sequenza e quindi l'indirizzo dell'istruzione successiva corrisponde all'indirizzo della prima istruzione successiva all'istruzione corrente.

FASE 2: ID

L'opcode dell'istruzione è decodificato e in base a esso si caricano nei registri di lavoro (R0, ..., Rn) gli operandi necessari all'istruzione e si attivano le microcircuiterie necessarie a svolgere l'operazione richiesta.

FASE 3: EX

Le elaborazioni previste dall'opcode dell'istruzione sono eseguite nell'ALU, sfruttando i registri di lavoro (R0, ..., Rn) per eventuali risultati parziali.

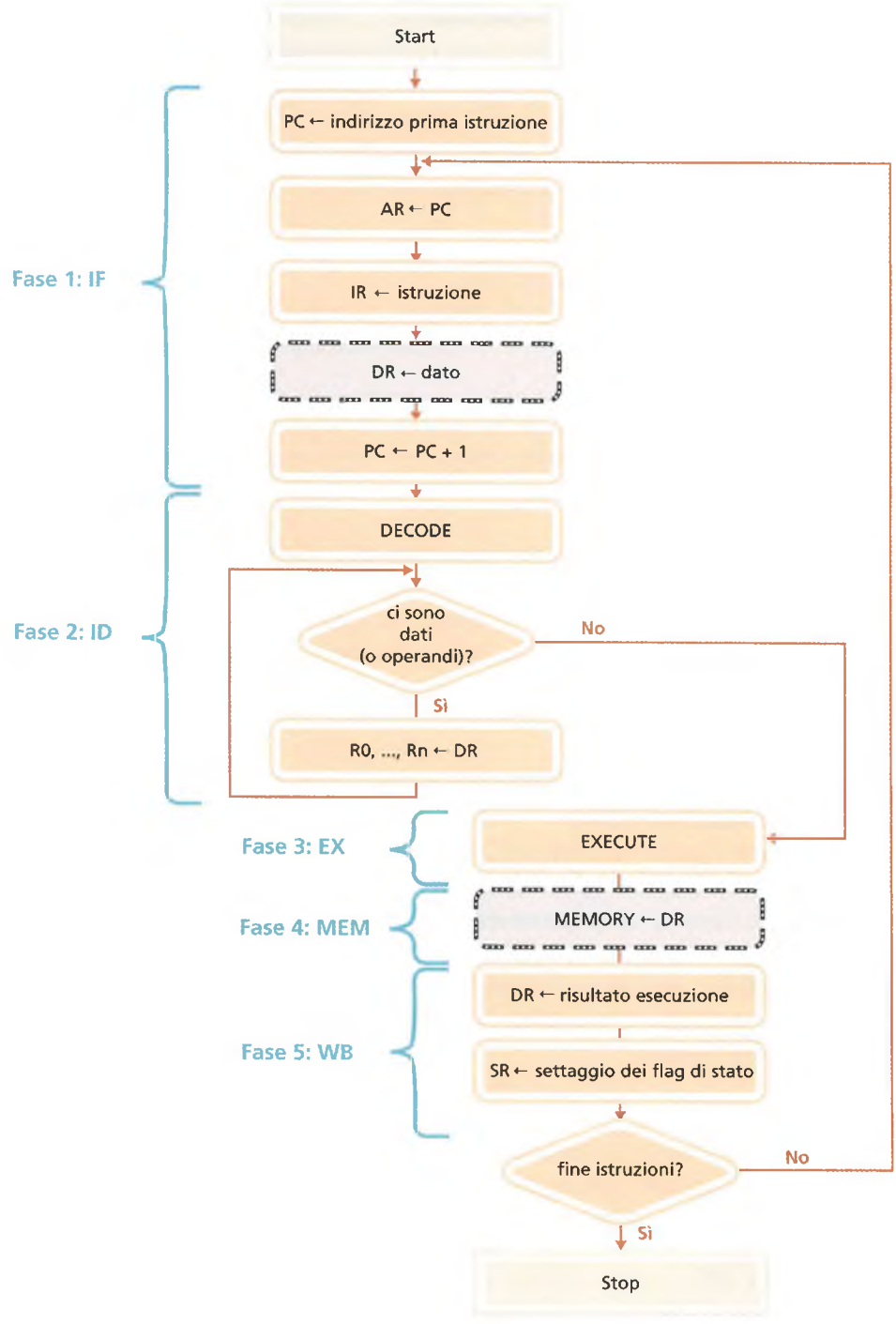
FASE 4: MEM

Consiste nell'eventuale scrittura del risultato di un'operazione in memoria (per esempio in caso di assegnamento di valori a variabili del programma) o verso le periferiche.

FASE 5: WB

Il risultato dell'esecuzione dell'istruzione viene scritto nel Data Register e lo stato del processore a seguito dell'esecuzione viene settato nello Status Register.

FIGURA 6 Flowchart del ciclo macchina



FISSA LE CONOSCENZE

- Quali sono le cinque fasi di un ciclo macchina?
- Che cos'è il cycle time (indicato con t_{CPU})?
- Che cos'è la frequenza di clock della CPU e con quale unità di misura è misurata?
- Descrivi nel dettaglio che cosa succede nella fase IF (*Instruction Fetch*).

3 LA TECNICA PIPELINING

3.1 La pipeline

Il **pipelining** è una tecnica che consente di elaborare in parallelo più istruzioni.
Come abbiamo visto nella Lezione precedente, l'esecuzione delle istruzioni segue il ciclo macchina costituito da una sequenza di fasi, a loro volta costituite da diverse micro-operazioni, eseguite dalla CPU ognuna in un ciclo di clock.

Con la tecnica del pipelining, le micro-operazioni sono raggruppate in **unità funzionali** indipendenti le une dalle altre. Le unità funzionali sono dei blocchi (microcircuiti) specializzati nell'eseguire velocemente determinate micro-operazioni. Questo implica che nella stessa CPU si possa elaborare **contemporaneamente** più micro-operazioni appartenenti a unità funzionali diverse. Ogni sequenza di elaborazioni indipendente forma una "conduttura", detta **pipeline**.

In pratica più unità eseguono le loro operazioni (eventualmente identiche) in **parallelo** su diversi dati.

Supponiamo per semplicità di essere nella situazione ottimale in cui un'unità funzionale sia in grado di elaborare in modo indipendente un'intera fase del ciclo macchina di un'istruzione.
Abbiamo quindi 5 unità funzionali indipendenti (si può arrivare quindi a 5 pipeline) e supponiamo di dover elaborare un programma di 8 istruzioni.
Nella FIGURA 7 si vede come dall'intervallo di tempo 5 all'intervallo 8, le 5 unità funzionali lavorino contemporaneamente realizzando delle pipeline a pieno regime di funzionamento.

#prendinota

Un parrucchiere per signora può organizzare il suo salone con 5 dipendenti, ognuno dei quali specializzato in un servizio diverso: tinta, shampoo, taglio, piega e manicure.
Se ha molte clienti da servire (così come una CPU ha molte istruzioni da eseguire), può servirle meglio e in minor tempo avendo dipendenti specializzati che possono lavorare in parallelo ognuno su un servizio diverso.

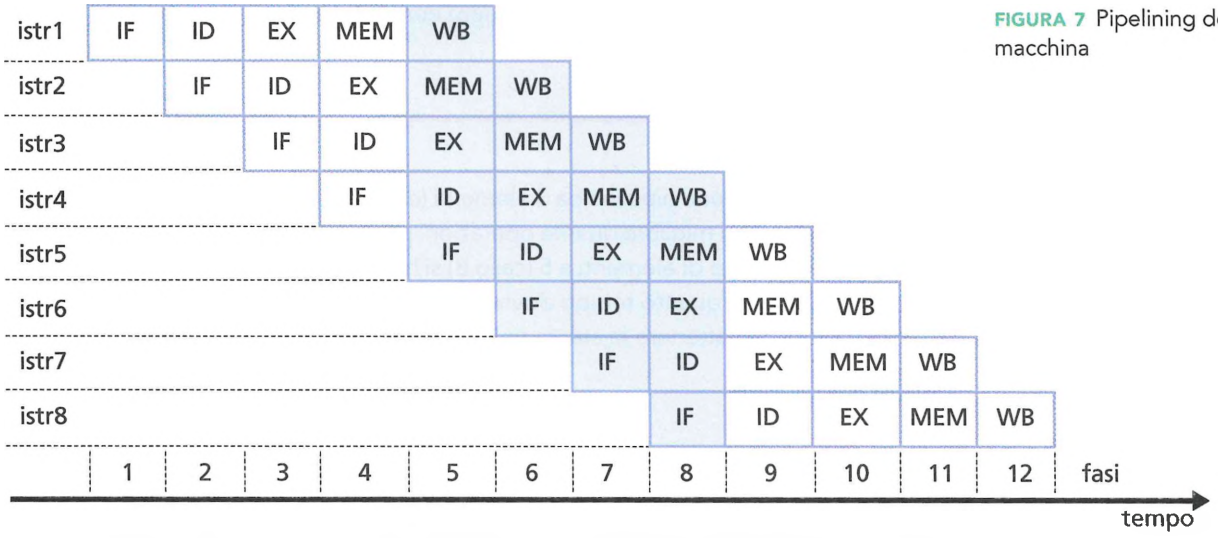


FIGURA 7 Pipelining del ciclo macchina

Quando la prima istruzione è nella fase 5, la seconda si trova già nella fase 4 e può immediatamente passare alla fase 5. Come si può osservare dalla Figura 7, 8 istruzioni sono eseguite in 12 fasi complessive. Non utilizzando la tecnica delle pipeline, occorre invece aspettare 40 fasi (5 fasi × 8 istruzioni). Nei sistemi di elaborazione privi di pipelining, ogni fase può iniziare solo dopo che la precedente è terminata.