


L'architettura di riferimento

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it

Università degli Studi di Milano


Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

La CPU.

Il bus.

L'organizzazione della memoria.

Il ciclo di esecuzione di un'istruzione.

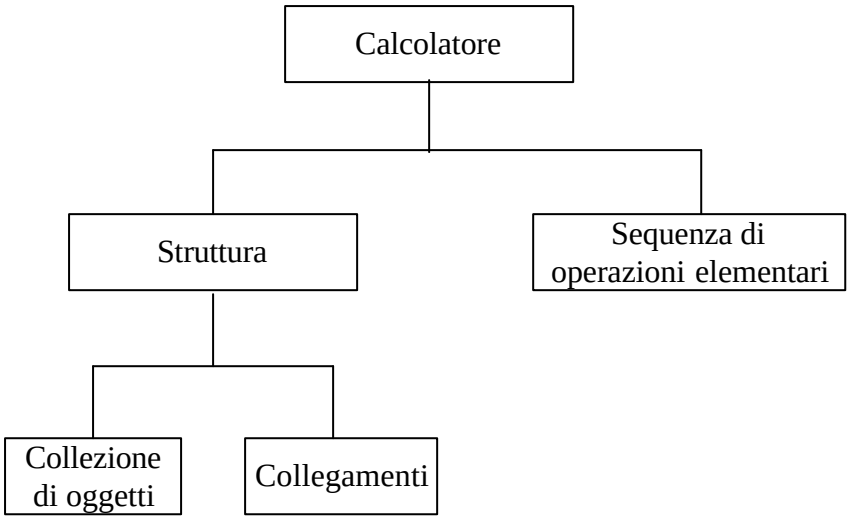
Obbiettivo di un'architettura

Elabora in modo adeguato un input per produrre l'output.

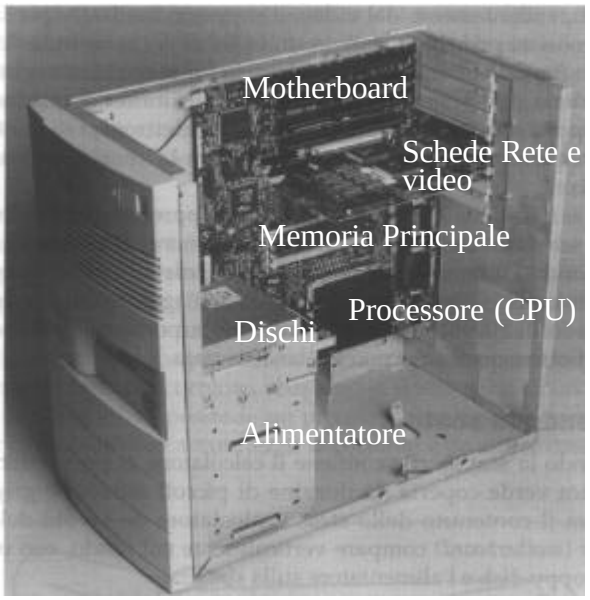
- Le unità di *ingresso* (tastiera del terminale video, mouse o altri dispositivi grafici di ingresso, ecc.) permettono al calcolatore di acquisire informazioni dall'ambiente esterno.
- L'architettura di elaborazione.

- Le unità di *uscita* (monitor grafico del terminale video, stampanti, ecc.) consentono al calcolatore di comunicare i risultati ottenuti dall'elaborazione all'ambiente esterno.

Descrizione di un elaboratore



Cosa c'è dentro un elaboratore?

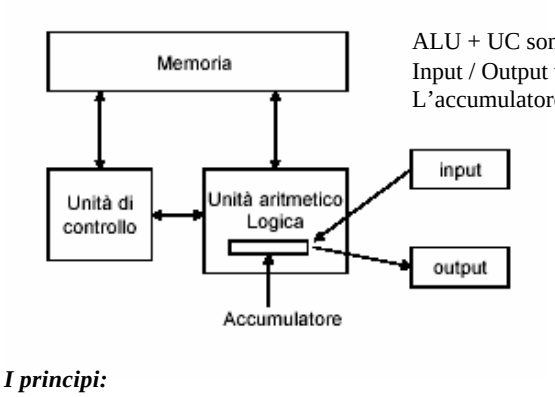


Architettura di riferimento degli elaboratori

- Elementi principali di un elaboratore:
  - Unità centrale di elaborazione (*Central Processing Unit - CPU*)
  - Memoria di lavoro o memoria principale *(Main Memory - MM)*

- Sulla motherboard:menti principali di un calcolatore:
  - Bus di sistema (dati, indirizzi, controllo)
  - Interfacce per i dispositivi di *Input/Output - I/O*: il terminale, la memoria di massa (di solito dischi magnetici), le stampanti, …

Architettura di Von Neumann



ALU + UC sono incorporate nella CPU.
Input / Output vengono in realtà trasferiti via bus.
L'accumulatore è un possibile tipo di architettura.

I principi:

- I dati e le istruzioni sono memorizzate in una memoria read/write.
- Il contenuto della memoria può essere recuperato in base alla sua posizione, e non è funzione del tipo di dato.
- L'esecuzione procede sequenzialmente da un'istruzione alla seguente.

Alcuni tipi di architetture

Accumulator (1 register = 1 indirizzo di memoria).

1 address	add A	acc ← acc + mem[A]
1+x address	addx A	acc ← acc + mem[A + x]

Stack (posso operare solo sui dati in cima allo stack):

0 address	add	tos ← tos + next

General Purpose Register (tanti diversi indirizzi di memoria quanti sono i registri, indirizzamento indiretto):

2 address	add A B	EA(A) ← EA(A) + EA(B)
3 address	add A B C	EA(A) ← EA(B) + EA(C)

Load/Store (posso operare solamente sui dati contenuti nei registri. Devo prima caricarli dalla memoria).

3 address	add Ra Rb Rc	Ra ← Rb + Rc
	load Ra Rb	Ra ← mem[Rb]
	store Ra Rb	mem[Rb] ← Ra

Architetture LOAD/STORE

- Il numero dei registri ad uso generale (ad esempio 32 registri da 32 bit ciascuno) non è sufficientemente grande da consentire di memorizzare tutte le variabili di un programma ⇒ ad ogni variabile viene assegnata una locazione di memoria nella quale trasferire il contenuto del registro quando questo deve essere utilizzato per contenere un'altra variabile.

- *Architetture LOAD/STORE:* gli operandi dell'ALU possono provenire soltanto dai registri ad uso generale contenuti nella CPU e **non** possono provenire dalla memoria. Sono necessarie apposite istruzioni di:
  - *caricamento (LOAD)* dei dati da memoria ai registri;
  - *memorizzazione (STORE)* dei dati dai registri alla memoria.

Architetture di tipo RISC (*Reduced Instruction Set Computer*)

- Ispirate al principio di eseguire soltanto istruzioni semplici: le operazioni complesse vengono scomposte in una serie di istruzioni più semplici da eseguire in un ciclo base ridotto, con l'obiettivo di migliorare le prestazioni ottenibili dalle *CPU CISC*.

- Caratterizzate da istruzioni molto semplificate.

- Gli operandi dell'*ALU* possono provenire dai registri ma *non* dalla memoria. Per il trasferimento dei dati da memoria ai registri e viceversa si utilizzano delle apposite operazioni di caricamento (*load*) e di memorizzazione (*store)* ⇒ *architetture load/store.*

CPU di tipo RISC
(*Reduced Instruction Set Computer*)

- *CPU* relativamente semplice ⇒ si riducono i tempi di esecuzione delle singole istruzioni, che sono però meno potenti delle istruzioni *CISC*.
- Dimensione *fissa* delle istruzioni ⇒ più semplice la gestione della fase di prelievo *(fetch)* e della codifica delle istruzioni da eseguire.

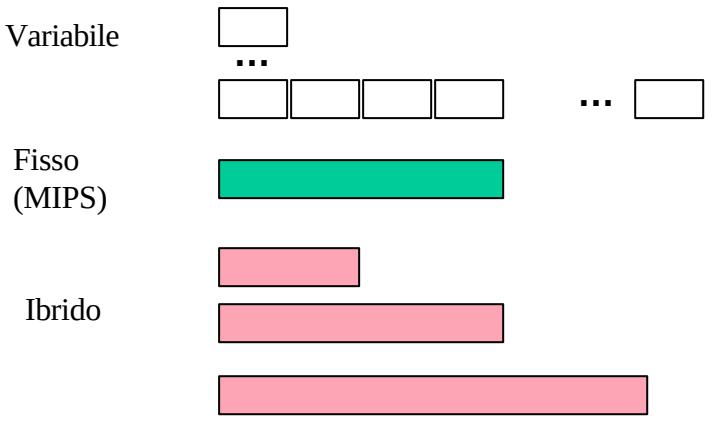
CPU di tipo CISC
(*Complex Instruction Set Computer*)

- Caratterizzate da elevata complessità delle istruzioni eseguibili ed elevato numero di istruzioni che costituiscono l'insieme delle istruzioni.
- Numerose modalità di indirizzamento per gli **operandi** dell'*ALU* che possono provenire da registri oppure da memoria, nel qual caso l'indirizzamento può essere diretto, indiretto, con registro base, ecc.

CPU di tipo CISC (*Complex Instruction Set Computer*)

- Dimensione *variabile* delle istruzioni a seconda della modalità di indirizzamento di ogni operando ⇒ complessità di gestione della fase di prelievo o *fetch* in quanto a priori non è nota la lunghezza dell'istruzione da caricare.
- Elevata complessità della *CPU* stessa (cioè dell'hardware relativo) in termini degli elementi che la compongono con la conseguenza di rallentare i tempi di esecuzione delle operazioni. Elevata profondità dell'albero delle porte logiche, utilizzato per la decodifica.

I diversi formati di istruzioni



Il formato fisso consente di massimizzare la velocità, il formato ibrido consente di minimizzare la lunghezza del codice.

Utilizzo architettura Intel 80x86: le 10 istruzioni più frequenti

° Rank	instruction	Integer Average Percent total executed
1	load	22%
2	conditional branch	20%
3	compare	16%
4	store	12%
5	add	8%
6	and	6%
7	sub	5%
8	move register-register	4%
9	call	1%
10	return	1%
	Total	96%

° Simple instructions dominate instruction frequency => RISC

Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

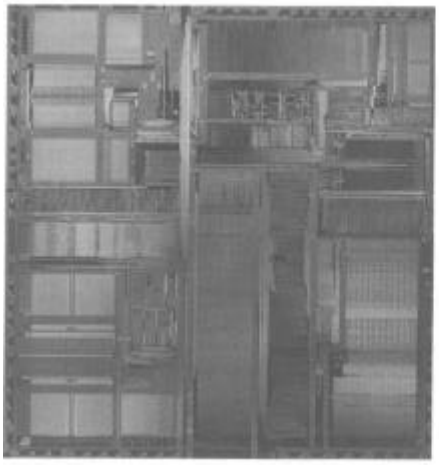
La CPU.

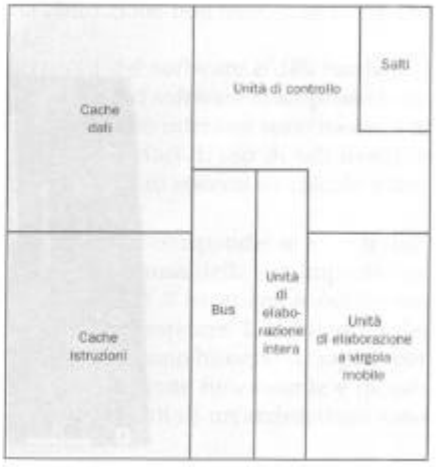
Il bus.

L'organizzazione della memoria.

Il ciclo di esecuzione di un'istruzione.

Componenti di un processore Pentium

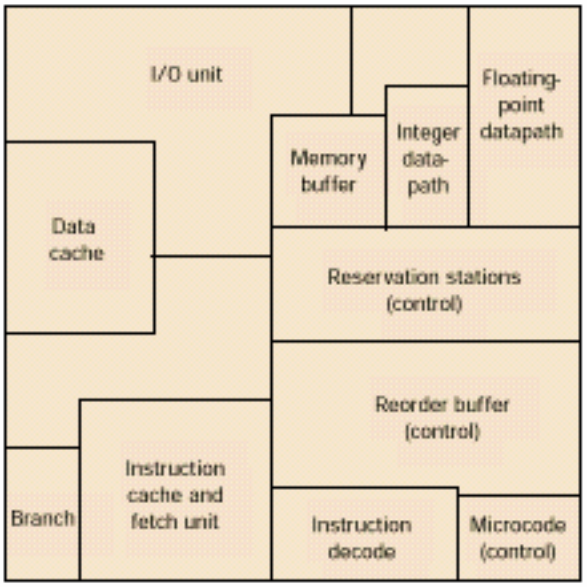




- 3,3 Milioni di transistor su 91mm^2

Il processore Pentium Pro

- 5,5 Milioni di transistor su 306mm^2 (2cm x 1.5cm) con cache esterna da 31 milioni.



Unità centrale di elaborazione *(Central Processing Unit - CPU)*

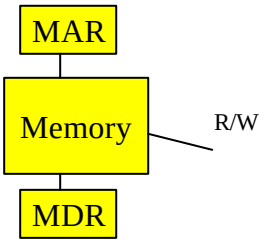
- La *CPU* provvede ad eseguire le istruzioni che costituiscono i diversi programmi elaborati dal calcolatore.
- Eseguire un'istruzione vuol dire operare delle scelte, eseguire dei calcoli a seconda dell'istruzione e dei dati a disposizione.

Elementi principali della CPU

- Banco di registri *(Register File)* ad accesso rapido, in cui memorizzare i dati di utilizzo più frequente. Il tempo di accesso ai registri è circa 10 volte più veloce del tempo di accesso alla memoria principale. Tra questi registri si distinguono il *MAR* e *MDR.*
- Registro *Program counter (PC).* Contiene l'indirizzo dell'istruzione corrente da aggiornare durante l'evoluzione del programma, in modo da prelevare dalla memoria la corretta sequenza di istruzione;
- Registro *Instruction Register (IR)*. Contiene l'istruzione in corso di esecuzione.
- Unità per l'esecuzione delle operazioni aritmetico-logiche (*Arithmetic Logic Unit - ALU*). I dati forniti all'*ALU* possono provenire da registri oppure direttamente dalla memoria, a seconda delle modalità di indirizzamento previste;
- Unità aggiuntive per elaborazioni particolari come unità aritmetiche per dati in virgola mobile *(Floating Point Unit – FPU)*, sommatori ausiliari, ecc.;
- ***Unità di controllo***. Controlla il flusso e determina le operazioni di ciascun blocco.

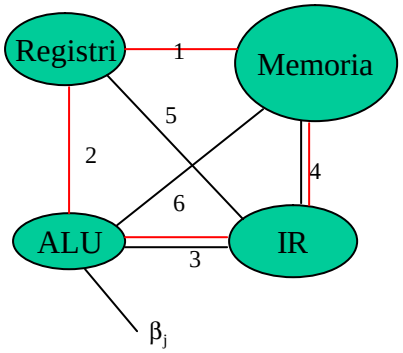
Interfaccia processore-memoria

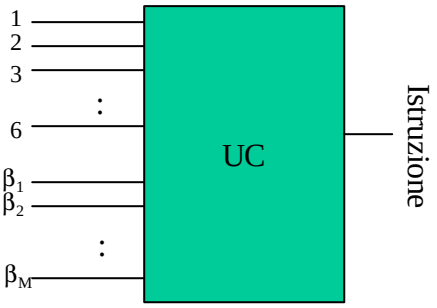
- **MAR – Memory Address Register:** registro degli indirizzi della memoria per memorizzare l'indirizzo della posizione della memoria principale in cui o da cui i dati o istruzioni devono essere trasferiti.
- **MDR – Memory Data Register:** registro dei dati della memoria per memorizzare i dati che devono essere scritti in memoria o letti dalla memoria e le istruzioni lette dalla memoria nella locazione indicata dall'indirizzo specificato



L'unità di controllo

- Unità di controllo coordina i flussi di informazione (è il "cervello" della CPU):
- 1) abilitando le vie di comunicazione opportune a seconda dell'istruzione in corso di esecuzione.
- 2) selezionando l'operazione opportuna delle ALU.





Collegamenti bidirezionali tra i dispositivi: n (n-1) → non praticabile, non sono neppure necessari!

Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

La CPU.

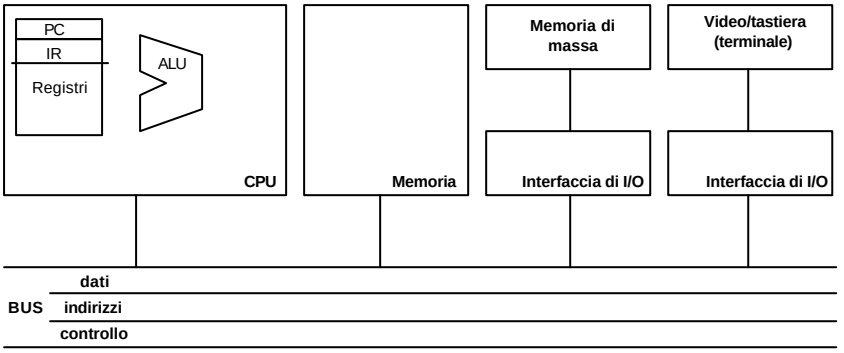
Il bus.

L'organizzazione della memoria.

Il ciclo di esecuzione di un'istruzione.

Collegamento tra i vari componenti mediante bus

Connessione *a nodo comune (bus).*



Numero di collegamenti: 2n.

Tutte le unità del calcolatore sono connesse al bus.

Bus di sistema

- Permette la comunicazione tra le diverse unità del calcolatore ed è generalmente composto da tre parti logiche:
  - *Bus dati*, comprende le linee per trasferire dati e istruzioni da/verso I dispositivi (la memoria). In generale, la dimensione del bus dati è tale da garantire il trasferimento contemporaneo di una o più parole di memoria;
  - *Bus indirizzi*, su cui la *CPU* provvede a trasmettere l'indirizzo da cui prelevare il dato nel caso di lettura dalla memoria, oppure in cui depositarlo nel caso di scrittura nella memoria (esempio la cella di memoria).
  - *Bus di controllo*, dove transitano le informazioni ausiliarie per la corretta definizione delle operazioni da compiere (per esempio l'indicazione che si vuole effettuare una *lettura* piuttosto che una *scrittura*) e per la sincronizzazione tra *CPU* e memoria.

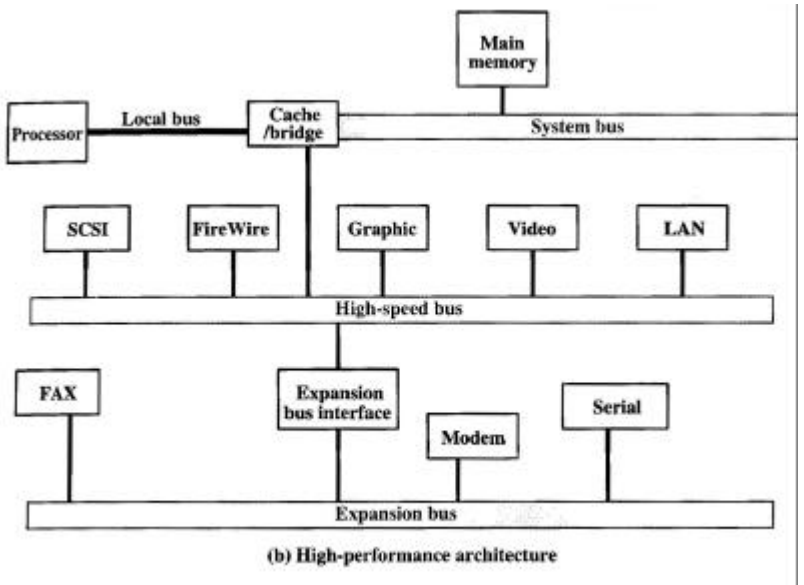
Esempio di utilizzo del bus

- Operazione di lettura dalla memoria. La *CPU* fornisce l'indirizzo della parola desiderata sul bus indirizzi, quindi viene richiesta l'operazione di lettura attivando il bus di controllo. Quando la memoria ha completato la lettura della parola richiesta, il dato viene trasferito sul bus dati e la *CPU* può prelevarlo ed utilizzarlo nelle sue elaborazioni.
- La struttura del bus può essere realizzata secondo diverse topologie di interconnessione.
- Il bus può esser utilizzato per un solo trasferimento alla volta $\Rightarrow$ in ogni istante soltanto due unità *(Master e Slave)* possono usare il bus.
- Nelle architetture di riferimento l'unità Master è solitamente la CPU. Esistono alcune schede di I/O particolarmente performanti che possono diventare anch'esse bus-master.
- Le linee di controllo del bus vengono utilizzate per inviare più richieste contemporanee di utilizzo del bus che vengono gestite dalla logica di *arbitraggio* del bus.

Bus di sistema & buffer

- Principali vantaggi della struttura a bus singolo: elevata flessibilità e bassi costi.
- I dispositivi collegati al bus variano in termini di velocità dell'esecuzione delle operazioni ⇒ necessario un meccanismo di sincronizzazione per garantire il trasferimento efficiente delle informazioni sul bus.
- Tipicamente all'interno delle unità che utilizzano il bus sono presenti dei registri di buffer per mantenere l'informazione durante i trasferimenti e non vincolarsi alla velocità del dispositivo più lento connesso al bus.
- Problema? Banda. Un video PAL 720 x 576 x 4 byte x 30 frame /s produce quasi 50Mbyte/s..

Architettura di bus



(b) High-performance architecture

Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

La CPU.

Il bus.

L'organizzazione della memoria.

Il ciclo di esecuzione di un'istruzione.

Indirizzi nella memoria principale

- La memoria è organizzata in *parole* o *word* composte da *n-bit* che possono essere caricate e memorizzate con una singola operazione di lettura/scrittura della memoria. (*n* è chiamata *lunghezza di parola*, tipicamente da *16* a *64* bit).
- Ogni parola di memoria è associata ad un indirizzo composto da *k-bit.*
- I 2^k indirizzi (corrispondenti a 2^k parole) costituiscono lo *spazio di indirizzamento* del calcolatore. Ad esempio un indirizzo composto da *32-bit* genera uno spazio di indirizzamento di 2^{32} o *4G* parole = *16Gbyte*.
- Compito principale consiste nel contenere i moduli attivi del sistema operativo ed i processi in esecuzione (completi di istruzioni e dati).

Memoria Principale

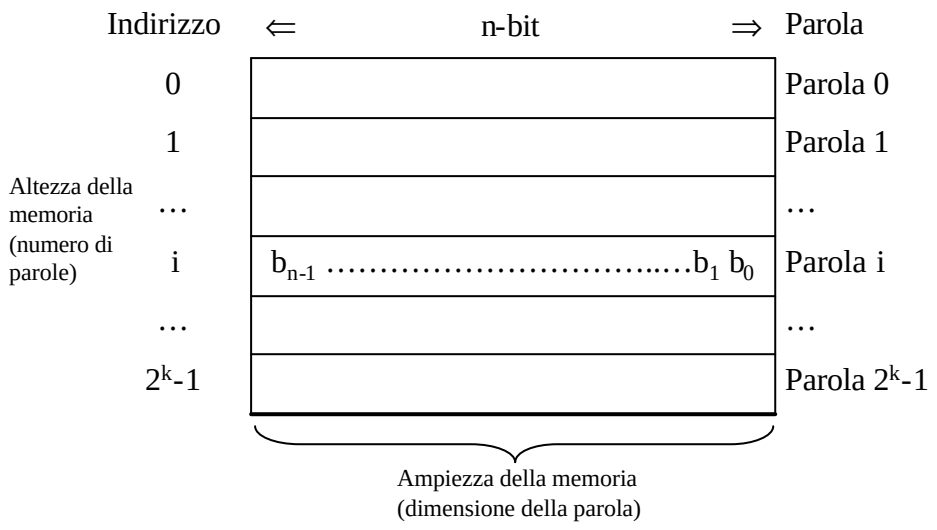
- In genere, la dimensione della parola di memoria coincide con la dimensione dei registri contenuti nella *CPU*, in modo da poter caricare una parola di memoria in un registro della *CPU*. Se anche il bus dati è largo come la parola di memoria

  ⇒ l'operazione di *load/store* avviene in un singolo ciclo.

- Le memorie in cui ogni locazione può essere raggiunta in un breve e prefissato intervallo di tempo misurato a partire dall'istante in cui si specifica l'indirizzo desiderato, vengono chiamate memorie ad accesso casuale *(Random Access Memory – RAM)*

- Nelle RAM il *tempo di accesso alla memoria* (tempo necessario per accedere ad una parola di memoria) è *fisso* e *indipendente* dalla posizione della parola alla quale si vuole accedere.

Indirizzi nella memoria principale



Capacità della memoria = #_parole x dim_parola

Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

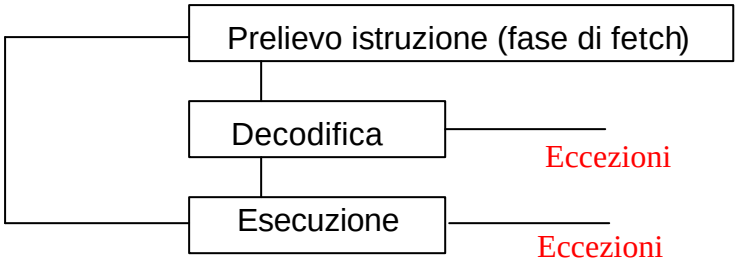
La CPU.

Il bus.

L'organizzazione della memoria.

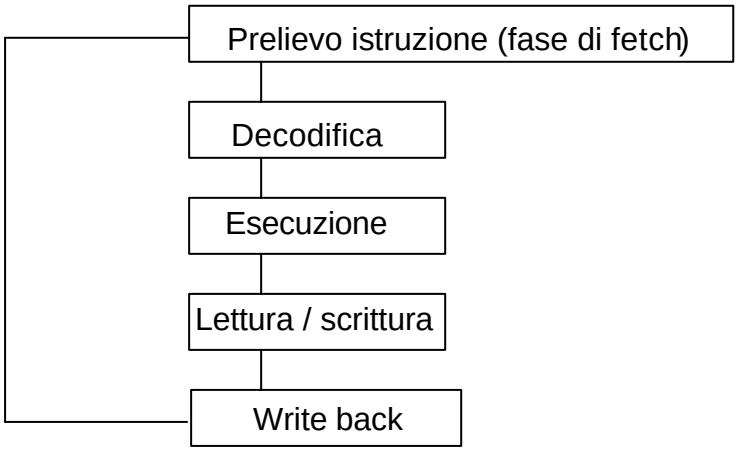
Il ciclo di esecuzione di un'istruzione.

Ciclo di esecuzione di un'istruzione



Il normale flusso di esecuzione di un programma può essere interrotto da un segnale di interruzione esterno *(INTERRUPT)* per una richiesta di intervento che un dispositivo di I/O ad esempio invia al processore.

Ciclo di esecuzione di un'istruzione MIPS



Lettura dell'istruzione (fetch)

- Istruzioni e dati risiedono nella memoria principale, dove sono stati caricati attraverso un'unità di ingresso.
- L'esecuzione di un programma inizia quando il registro PC punta alla prima istruzione del programma.
- Il segnale di controllo per la lettura (READ) viene inviato alla memoria.
- Trascorso il tempo necessario all'accesso in memoria, la parola indirizzata (in questo caso la prima istruzione del programma) viene letta dalla memoria e trasferita nel registro IR.

Decodifica dell'istruzione

- L'istruzione contenuta nel registro IR viene decodificata per essere eseguita. Alla fase di decodifica corrisponde la predisposizione della CPU (apertura delle vie dicomunicazione appropriate) all'esecuzione dell'istruzione.
- In questa fase vengono anche recuperati gli operandi. Nelle architetture MIPS gli operandi possono essere solamente nel Register File (oppure contenuti nell'istruzione stessa).

Esecuzione dell'istruzione

Viene selezionato il circuito / i circuiti combinatori appropriati per l'esecuzione delle operazioni previste dall'istruzione e determinate in fase di decodifica.

Tra le operazioni previste, c'è anche la formazione dell'indirizzo di memoria da cui leggere o su cui scrivere un dato.

Lettura / Scrittura in memoria

In questa fase il dato presente in un registro, viene scritto in memoria oppure viene letto dalla memoria un dato e trasferito ad un registro.

Questa fase non è richiesta da tutte le istruzioni!

Nel caso particolare di Architetture LOAD/STORE, quali MIPS, le istruzioni di caricamente dalla memoria sono separate da quelle aritmetico/logiche. Se effettuo una Letura / Scrittura, **non** eseguo operazioni aritmetico logiche sui dati.

Sistema di memoria “sganciato” dalla coppia register-file + CPU.

Scrittura in register file (write-back)

- Il risultato dell’operazione può essere memorizzato nei registri ad uso generale oppure in memoria.
- Mentre viene eseguita un’istruzione, il contenuto del PC viene incrementato in modo da puntare all’istruzione successiva.
- Non appena è terminato il ciclo di esecuzione dell’istruzione corrente (termina la fase di Write Back), si preleva l’istruzione successiva dalla memoria.

Esecuzione dell'istruzione

Viene selezionato il circuito / i circuiti combinatori appropriati per l'esecuzione delle operazioni previste dall'istruzione e determinate in fase di decodifica.

Eesempio: add $s3, $s2, $s1

Fase di fetch:	Caricamento dell'istruzione.
Decodifica:	Preparazione della CPU a svolgere una somma.
Lettura dei dati:	Indirizzamento adeguato del Register File.
Esecuzione:	Esecuzione della somma.

Recupero degli operandi

- Se l'istruzione deve essere svolta dall'unità aritmetico-logica è necessario recuperare gli operandi richiesti, che possono risiedere nei registri di uso generale oppure in memoria.
- Architetture a registri:
  - Se un operando risiede in memoria, deve essere prelevato caricando l'indirizzo dell'operando nel registro MAR e attivando un ciclo di READ della memoria.
  - L'operando letto dalla memoria viene posto nel registro MDR per essere trasferito alla ALU, che esegue l'operazione. Nelle architetture MIPS, l'operando viene trasferito nel Register file nella fase di Scrittura.
- Architetture LOAD/STORE:
  - Le istruzioni di caricamente dalla memoria sono separate da quelle aritmetico/logiche.

Esempio ciclo di esecuzione

Somma: add $s3, $s2, $s1

Fase di fetch:	Caricamento dell'istruzione.
Decodifica:	Preparazione della CPU a svolgere una somma. Determinazione dei segnali di controllo. Lettura degli operandi (che sono contenuti nei registri $s2, $s1).
Esecuzione:	Esecuzione della somma.
R / W:	Nulla
Write-back	Scrittura del registro $s3.

Sommario

L'Architettura di Von Neuman, architetture CISC e RISC.

La CPU.

Il bus.

L'organizzazione della memoria.

Il ciclo di esecuzione di un'istruzione.