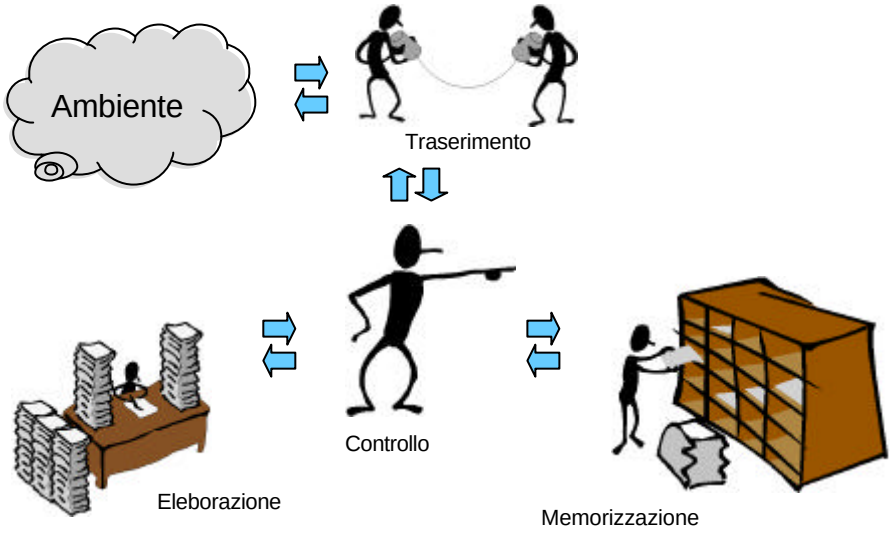
L'Architettura di un Calcolatore

Maurizio Palesi
Salvatore Serrano

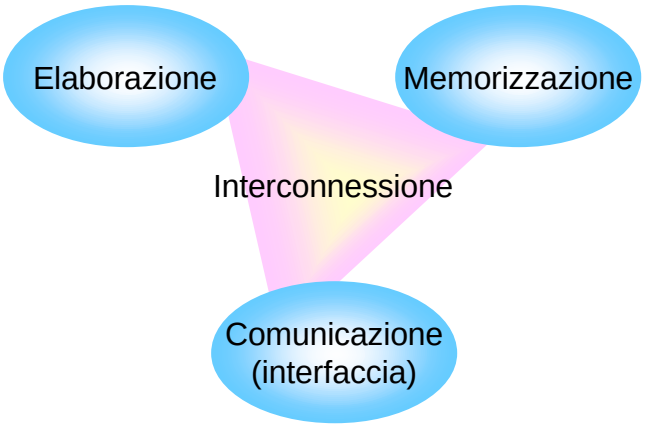
Sommario

- Architettura generale di un calcolatore
  - ➔Componenti fondamentali
  - ➔Il sistema di interconnessione
- L'unità centrale di elaborazione
  - ➔Componenti fondamentali
  - ➔Data path
  - ➔Il ciclo macchina
  - ➔Misura delle prestazioni
- La memoria
  - ➔Generalità e caratterizzazione
  - ➔Gerarchie di memorie e memoria cache

Vista funzionale di un Calcolatore

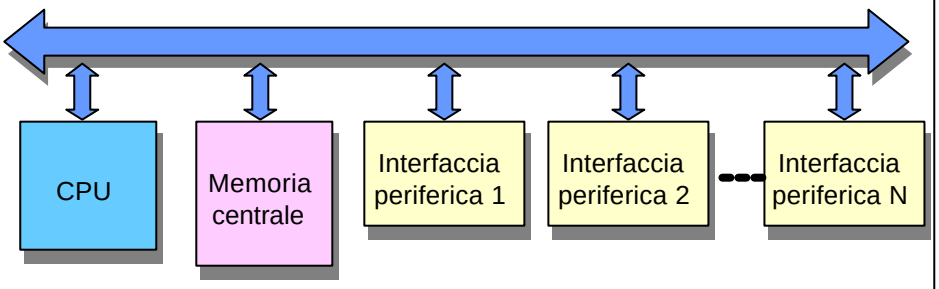


Il Calcolatore: Modello Concettuale

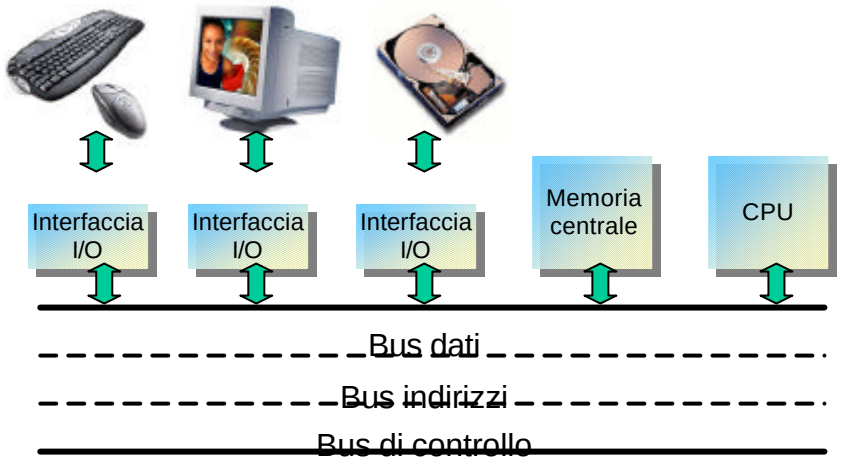


La Macchina di von Neumann

- 4 elementi funzionali fondamentali:
  - ➔Unità di elaborazione (CPU, *Central Processing Unit*)
  - ➔Memoria centrale
  - ➔Periferiche
  - ➔Bus di sistema



Lo Schema di Riferimento



Bus e Master-Slave

- Il bus è una linea a cui sono contemporaneamente connesse le unità del calcolatore e che consente il trasferimento di dati tra tali unità
  - ➔Problema: contesa su un mezzo condiviso!
  - ➔Soluzione: CPU = master, periferiche = slave

Bus e Master-Slave - Pregi

- Semplicità: 1 sola line di connessione ∀ # di dispositivi
- Estendibilità: nuovi dispositivi possono essere aggiunti tramite un'interfaccia al bus senza influenzare l'HW preesistente
- Standardizzabilità: definizione di normative che consentono a periferiche di costruttori diversi di interagire correttamente

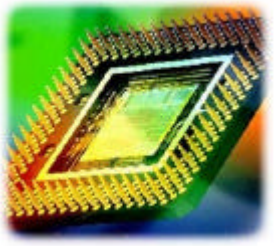
Bus e Master-Slave - Difetti

- Lentezza: l'uso in mutua esclusione del bus inibisce almeno parzialmente la parallelizzazione delle operazioni di trasferimento di dati tra dispositivi
- Limitata capacità: al crescere del numero di dispositivi la presenza di una sola linea comporta un limite alla capacità di trasferire dati
- Sovraccarico della CPU: l'unità centrale viene coinvolta in tutte le operazioni di trasferimento di dati

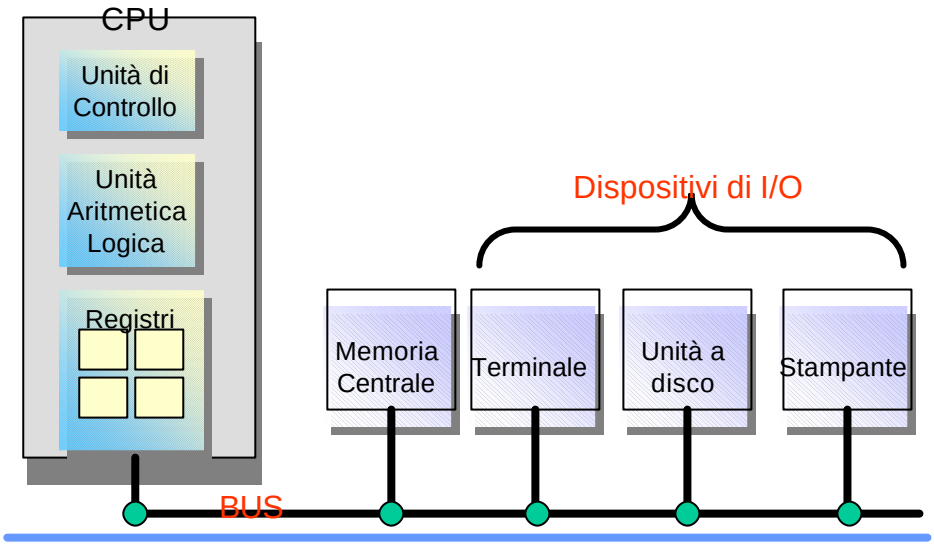
Tipi di Bus

- Bus dati: utilizzato per trasferire dati (es. fra memoria e CPU, fra CPU e interfacce di I/O)
- Bus indirizzi: che identifica la posizione delle celle di memoria un cui la CPU va a scrivere o leggere
- Bus di controllo: in cui transitano i segnali di controllo che consentono di selezionare le unità coinvolte in un trasferimento dati (sorgente e destinazione), di definire la direzione dello scambio (scrittura o lettura)

L'Unità Centrale di Elaborazione



Organizzazione Tipica (bus oriented)



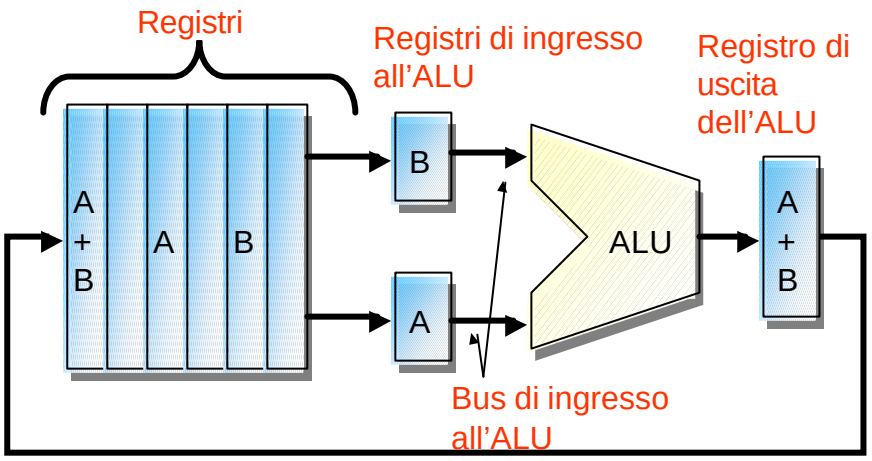
Tre Tipologie di Istruzioni

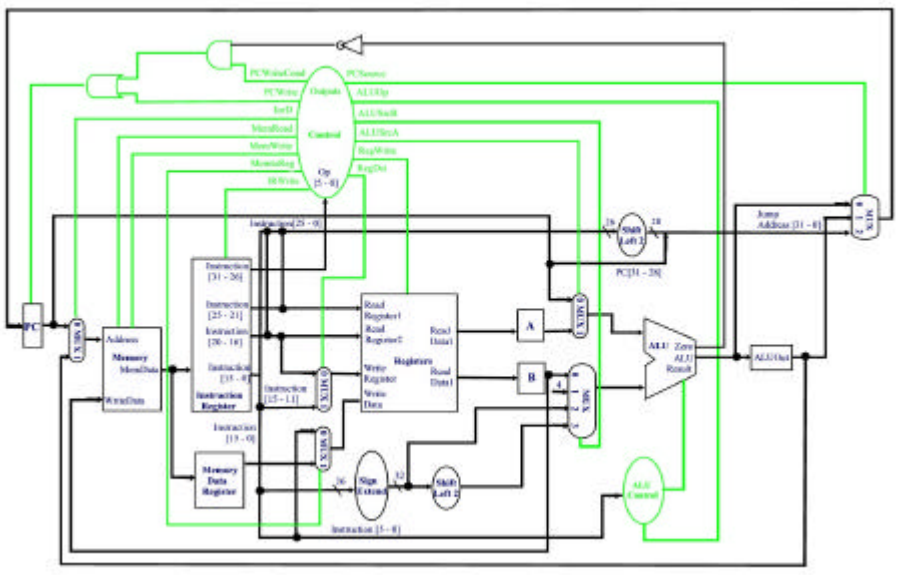
- Istruzioni Aritmetico Logiche (Elaborazione dati)
  - ➔Somma, sottrazione, divisione, …
  - ➔And, Or, Xor, …
  - ➔Maggiore, minore, uguale, maggiore uguale, …
- Controllo del flusso delle istruzioni
  - ➔Sequenza
  - ➔Selezione
  - ➔Ciclo a condizione iniziale, a condizione finale, …
- Trasferimento di informazione
  - ➔Trasferimento dati e istruzioni tra CPU e memoria
  - ➔Trasferimento dati e istruzioni tra CPU e dispositivi di I/O

Elementi di una CPU

- Unità di controllo
  - ➔Legge le istruzioni dalla memoria e ne determina il tipo
- Unità aritmetico-logica
  - ➔Esegue le operazioni necessarie per eseguire le istruzioni
- Registri
  - ➔Memoria ad alta velocità usata per risultati temporanei
  - ➔Determina il parallelismo della CPU
  - ➔Esistono registri generici e registri specifici
    - ✓Program Counter (PC)
    - ✓Instruction Register (IR)
    - ✓…

Struttura del “data path”





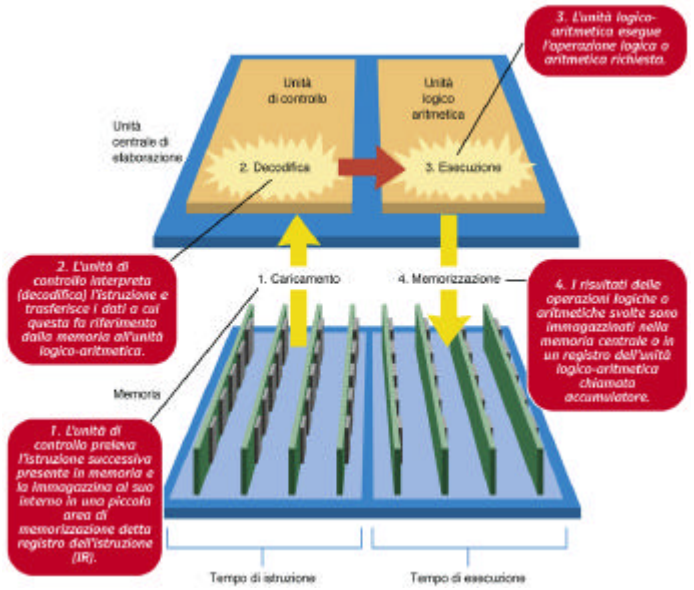
L'Esecutore

- Un calcolatore basato sull'architettura di Von Neumann esegue un programma sulla base dei seguenti principi
  - ➔Dati e istruzioni sono memorizzati in una memoria unica che permette sia la scrittura che la lettura
  - ➔I contenuti della memoria sono indirizzati in base alla loro posizione
  - ➔Le istruzioni vengono eseguite in modo sequenziale

Linguaggio Macchina e Assembly

- Linguaggio macchina
  - ➔Rudimentale
  - ➔Il concetto di tipo di dato è quasi assente
  - ➔Il numero di operandi è limitato
  - ➔Il numero di operazioni previste è ridotto

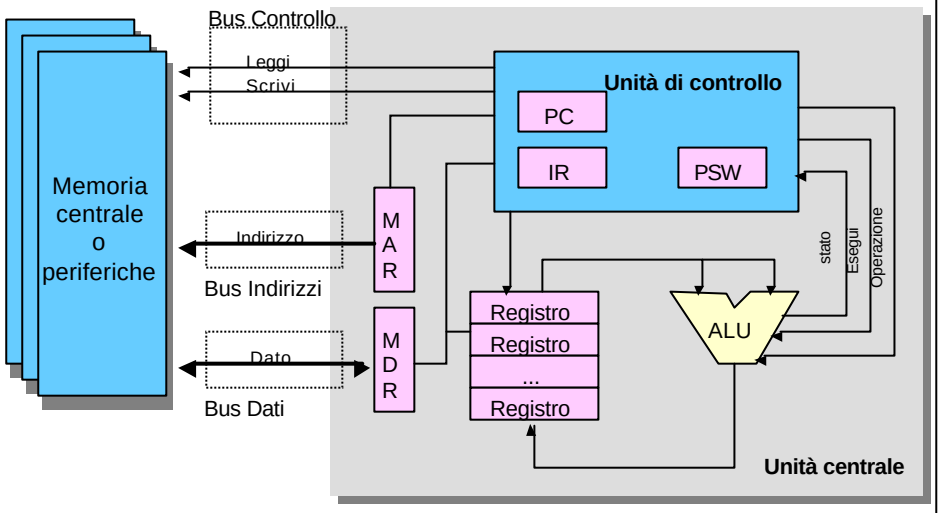
Struttura di una istruzione della CPU	codice operativo	op1	op2
Specifica analoga alla codifica in assembly	SOMMA	Reg1	Reg2
Codifica in un ipotetico linguaggio macchina	10000011	001	010



Esecuzione delle Istruzioni

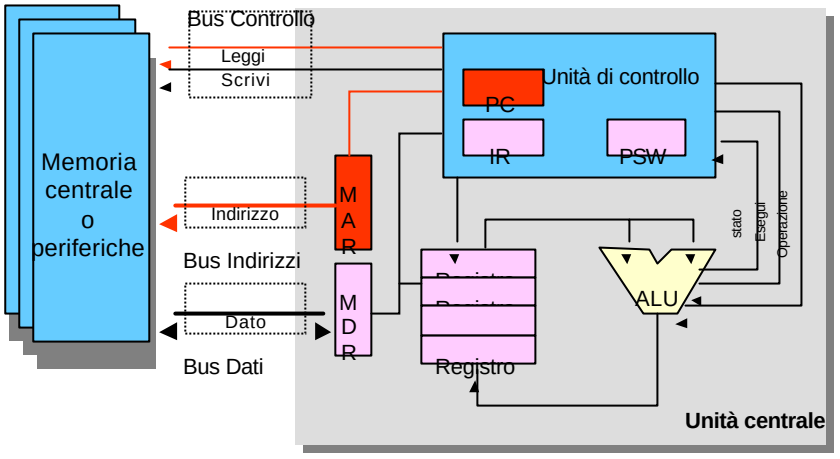
- Ciclo Fetch-Decode-Execute
  1. Prendi l'istruzione corrente dalla memoria e mettila nel registro istruzioni (**IR**) [Fetch]
  2. Incrementa il program counter (**PC**) in modo che contenga l'indirizzo dell'istruzione successiva
  3. Determina il tipo dell'istruzione corrente [Decodifica]
  4. Se l'istruzione usa una parola in memoria determina dove si trova
  5. Carica la parola, se necessario, in un registro della CPU
  6. Esegui l'istruzione [Execute]
  7. Torna al punto 1.

Struttura Semplificata di una CPU



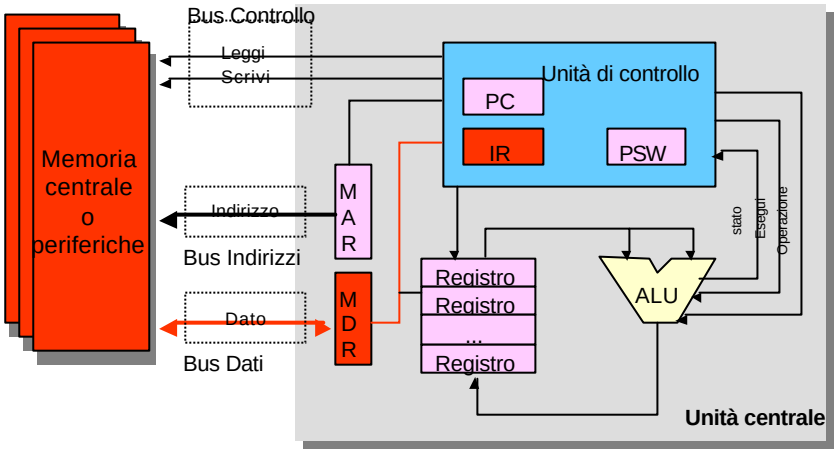
Esempio: Lettura dalla Memoria

- Fase di Fetch (1 di 2)



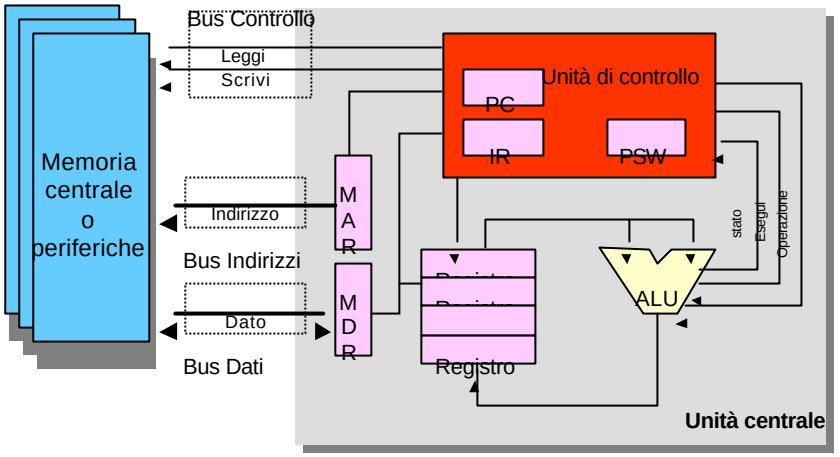
Esempio: Lettura dalla Memoria

- Fase di Fetch (2 di 2)



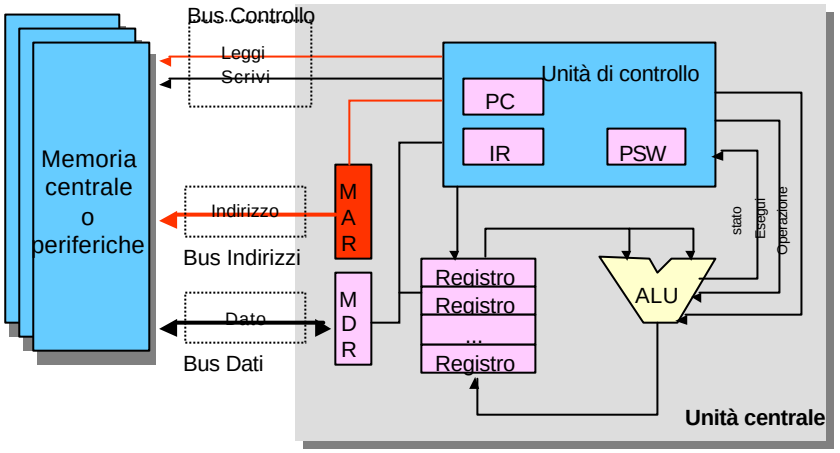
Esempio: Lettura dalla Memoria

- Decodifica



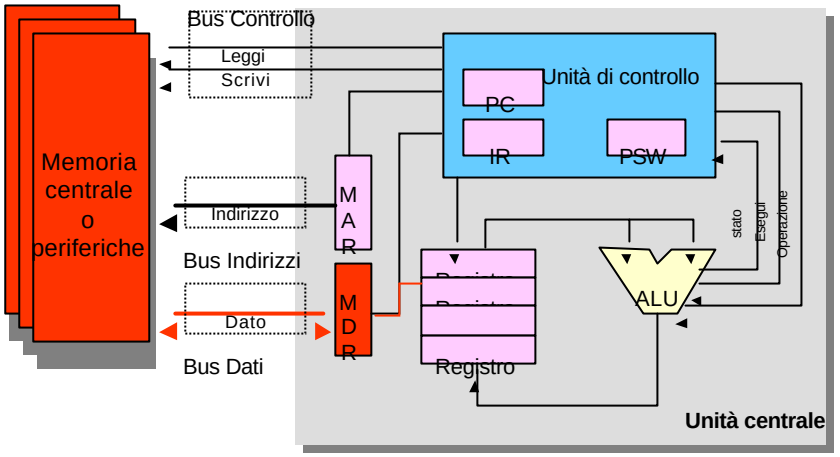
Esempio: Lettura dalla Memoria

- Esecuzione (1 di 2)



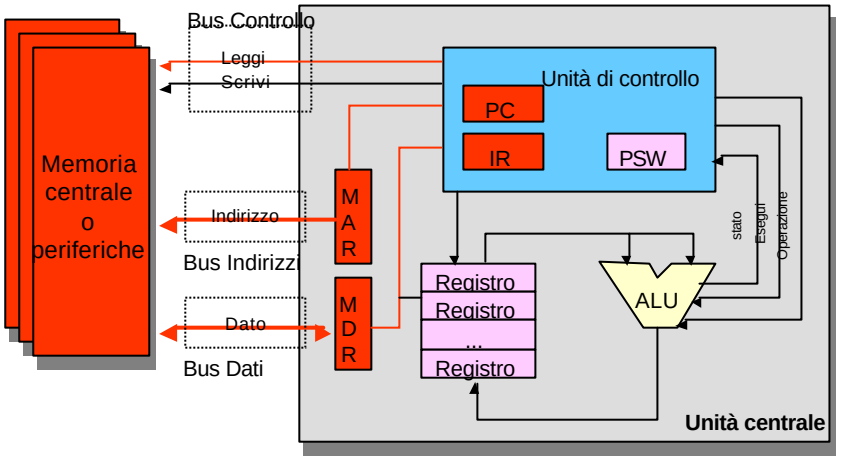
Esempio: Lettura dalla Memoria

- Esecuzione (2 di 2)



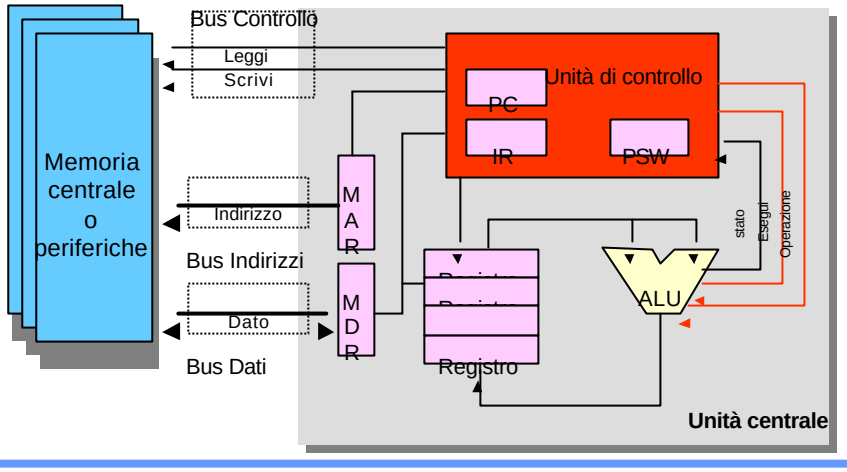
Esempio: Somma tra due registri

- Fetch (come prima)



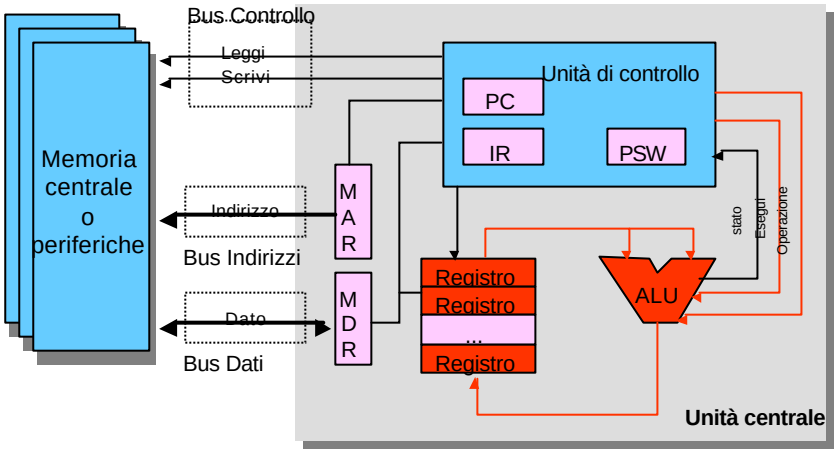
Esempio: Somma tra due registri

- Decodifica

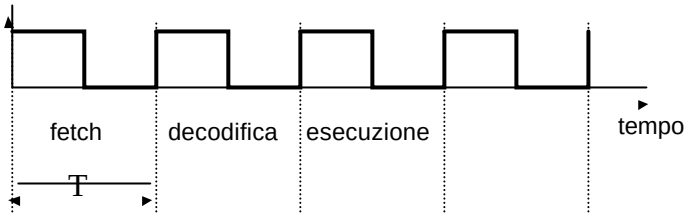


Esempio: Somma tra due registri

- Esecuzione



Evoluzione delle CPU



CPU	Anno	Frequenza (MHz)	Dimensione registri/bus dati	Numero di transistors
8086	1978	4.77-12	8/16	29K
80286	1982	8-16	16/16	134K
80386	1986	16-33	32/32	275K
80386 SX	1988	16-33	32/16	275K
80486	1989	33-50	32/32	1.2M
Pentium	1993	60-200	32/64	3.1M
Pentium II	1997	233-400	32/64	7.5M
Pentium III	1999	450-1133	32/64	24M
Pentium 4	2000	1600-2000	32/64	42M

Le Prestazioni di un Sistema Inf.co

- Due punti di vista
  - ➔Utente: interessato al tempo di risposta (o tempo di esecuzione o latenza)
  - ➔Il direttore di un centro di calcolo: interessato al throughput
- In generale
  - ➔il calcolatore che svolge la stessa quantità di lavoro nel minor tempo è più veloce

Come misurare il tempo?

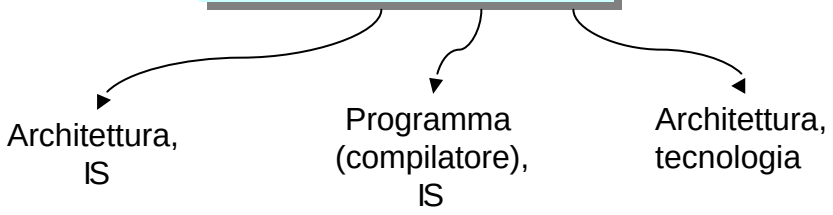
Elapsed time vs. CPU time

- Utente:
  - ➔Misura il tempo complessivo che trascorre tra l'inizio e la fine dell'esecuzione del compito. Tempo che comprende:
    - ✓Gli accessi ai dischi
    - ✓Gli accessi alla memoria
    - ✓Le attività di I/O
    - ✓Il sovraccarico di lavoro dovuto al S.O.
    - ✓L'esecuzione di altri processi contemporaneamente attivi
- Direttore del CdC:
  - ➔Misura il tempo effettivo di elaborazione dedicato dalla CPU allo svolgimento del compito (*CPU time*)

Tempo di CPU

- $T_{CPU} = n_{CK} \times T_{CK}$
  - ➔ n_{CK} numero di cicli di clock per eseguire il programma
  - ➔ T_{CK} periodo di clock
- Se con nI indichiamo il numero di istruzioni eseguite
  - ➔ $CPI = n_{CK} / n_I$

$$T_{CPU} = CPI \times n_I \times T_{CK}$$



CISC vs. RISC

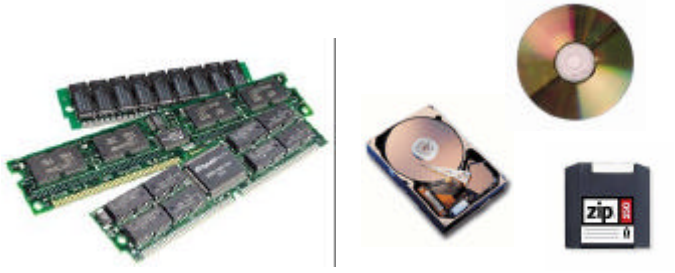
- Nella scelta del set di istruzioni (IS) sono contrapposte due diverse filosofie di progettazione
  - ➔ CISC (**C**omplex **I**nstruction **S**et **C**omputer)
  - ➔ RISC (**R**educed **I**nstruction **S**et **C**omputer)

Tipo di CPU	CISC	RISC
Cicli di clock per istruzione (CPI)	Alto	Basso
Numero di istruzioni per un processo (Ni)	Basso	Alto
Frequenza di clock (Fck)	Media	Alta

Altri Indici di Prestazioni

- MIPS: **M**ega **I**nstructions **P**er **S**econd
  - ➔ MIPS = $n_I / T_{CPU} \times 10^{-6}$
  - ➔ Facilmente comprensibile
  - ➔ Poco affidabili
    - ✓ Nel confronto di IS diversi
    - ✓ Fortemente dipendenti dal programma
    - ✓ Possono variare in maniera inversamente proporzionale alle prestazioni (es. presenza di un coprocessore matematico o acceleratore)
- MFLOPS: **M**ega **FL**oating point **O**perations **P**er **S**econd

La Memoria



La Memoria

- Supporto alla CPU
  - ➔Deve fornire alla CPU dati ed istruzioni il più rapidamente possibile
- Archivio
  - ➔Deve consentire di archiviare dati e programmi garantendone la conservazione e la reperibilità dopo elevati periodi di tempo
- Diverse esigenze
  - ➔Velocità: per il supporto alla CPU
  - ➔Non volatilità ed elevate capacità per l'archiviazione
- Diverse tecnologie
  - ➔Elettronica: ma costosa e volatile
  - ➔Magnetica e ottica: non volatile ed economica ma lenta

Criteri di Caratterizzazione

- Velocità
  - ➔Tempo di accesso (tempo che intercorre tra richiesta e risposta)
  - ➔Velocità di trasferimento (byte/sec che è posibile trasferire)
- Volatilità
  - ➔Cosa succede quando la memoria non è alimentata?
  - ➔Per quanto tempo i dati vi rimangono immagazzinati?
- Capacità
  - ➔Quanti byte può contenere?
- Costo per bit
- Modalità di accesso
  - ➔Diretta (o casuale): il tempo di accesso è indipendente dalla posizione
  - ➔Sequenziale: il tempo di accesso dipende dalla posizione
  - ➔Associativa: indicato il dato la memoria risponde con la posizione in cui quel dato si trova

La Memoria Centrale (R.A.M)

- Mantiene al proprio interno i dati e le istruzioni dei programmi in esecuzione
- Memoria ad accesso casuale
- Tecnologia elettronica
  - ➔Veloce ma volatile e costosa
- Due “eccezioni”
  - ➔R.O.M. Elettronica ma permanente e di sola lettura
  - ➔Flash: Elettronica ma permanente e riscrivibile

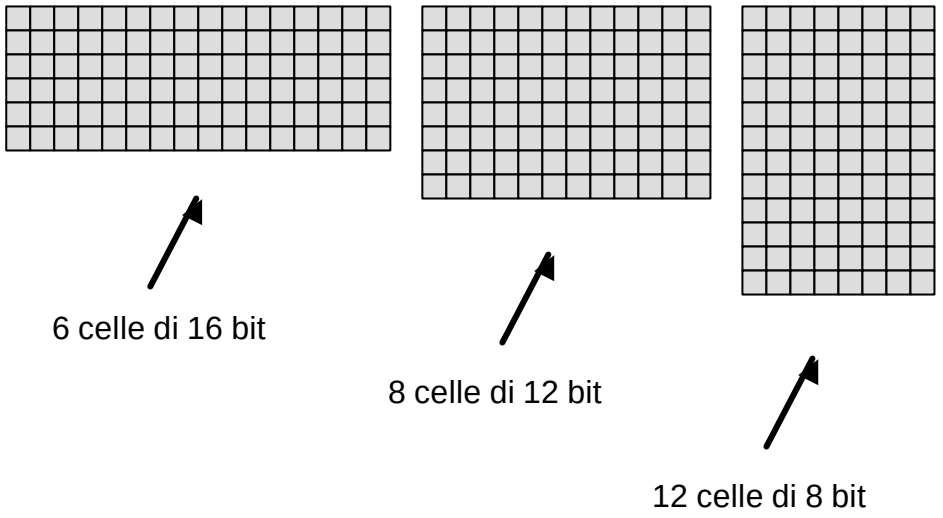
Indirizzi di Memoria

- I bit nelle memorie sono raggruppate in celle
  - ➔Tutte le celle sono formate dello stesso numero di bit
  - ➔Una cella di *k* bit può contenere qualsiasi delle 2^k combinazioni diverse di bit
- Ogni cella ha un indirizzo
  - ➔Serve come accesso all’informazione
  - ➔In una memoria con *N* celle gli indirizzi vanno da *0* a *N-1*
- La cella è l’unità indirizzabile più piccola

Organizzazione della Memoria

- Anche gli indirizzi di memoria sono rappresentati come numeri binari
  - ➔Un indirizzo di M bit permette di indirizzare 2^M celle
  - ➔Il numero di bit nell'indirizzo determina il numero di celle indirizzabili nella memoria ed è indipendente dal numero di bit per cella
- Una memoria può essere organizzata in diversi modi
  - ➔Es. Una memoria di 96 bit
    - ✓6 celle di 16 bit ($6 \times 16 = 96$)
    - ✓8 celle di 12 bit ($8 \times 12 = 96$)
    - ✓12 celle di 8 bit ($12 \times 8 = 96$)

Organizzazione della memoria



Memoria vs. CPU

- Le CPU sono sempre state più veloci delle memorie
  - ➔L'aumento di integrazione ha consentito di progettare CPU pipelined, superscaleri, molto veloci
  - ➔Nelle memoriè è aumentata più la capacità che la velocità
- L'accesso alla memoria passa attraverso il bus
  - ➔La frequenza di funzionamento del bus è molto più bassa di quella della CPU
  - ➔Gran parte delle istruzioni eseguite sono di accesso alla memoria
- E' possibile integrare memorie che sono molto veloci all'interno del chip ma sono piccole e costose

Gerarchie di Memorie - Introduzione

- Memorie di gran capacità, relativamente lente, economiche ed accessibili tramite il bus
  - ➔**MGL** ovvero Memoria Grossa e Lenta
  - ➔Dimensioni pari a circa 10 unità
- Memorie veloci integrate nello stesso chip della CPU ma costose
  - ➔**MPV** ovvero Memoria Piccola e Veloce
  - ➔Dimensioni pari a circa 1 unità
  - ➔Tempo di accesso pari a circa 1 unità
- **Obiettivo**: Realizzare una memoria grossa e veloce
  - ➔Dimensioni: pari a circa quelle della memoria grossa
  - ➔Prestazioni: pari a circa quelle della memoria veloce

Gerarchie di Memorie

- Memoria formata da una MPV e da una MGL
  - ➔La MPV contiene una copia di alcune celle della MGL
  - ➔Quando la CPU chiede una particolare cella di memoria la richiesta va ad entrambe le memorie
    - ✓Se il dato si trova nella MPV viene passato direttamente alla CPU
    - ✓Se il dato si trova nella MGL viene anche caricato nella MPV
- Ipotesi: Distribuzione uniforme delle richieste
  - ➔La frequenza con cui si trova il dato nella MPV (hit ratio) sarà in media 10%, in questi casi il tempo di accesso (hit time) sarà 1 unità
  - ➔La frequenza con cui è necessario accedere alla MGL (miss ratio) sarà in media 90%, in questi casi il tempo di accesso (miss penalty) sarà 10 unità
  - ➔Il tempo medio di accesso sarà: $0.1 \times 1 + 0.9 \times 10 = 9.1$ unità!

I Principi di Località

- Località spaziale
  - ➔Quando si accede ad un indirizzo A è molto probabile che gli accessi successivi richiedono celle vicine ad A
    - ✓Le istruzioni del codice vengono in genere lette da locazioni consecutive in memoria
    - ✓Gli accessi ad array o a strutture dati sono “vicini”
- Località temporale
  - ➔Quando si accede all’indirizzo A, è molto probabile negli accessi successivi si richieda di nuovo la cella A
    - ✓Cicli di istruzioni accedono ripetutamente alle stesse locazioni di memoria
    - ✓Istruzioni vicine tendono ad utilizzare le stesse variabili

Come si sfrutta la Località

- Diversi approcci a seconda del tipo di località
  - ➔ Località temporale: i dati prelevati dalla MGL vengono conservati nella MPV il più a lungo possibile
  - ➔ Località spaziale: quando si copia un dato dalla MGL alla MPV si copiano anche i dati vicini (cache line o blocco)
- La frequenza di successo (hit ratio) cresce fino a superare il 99%
  - ➔ l'hit ratio (h) dipende da due caratteristiche contrasstanti
    - ✓ La dimensione dei blocchi
      - – un blocco grande sfrutta meglio la località spaziale
    - ✓ Quanti sono i blocchi in memoria
      - – se c'è spazio per tanti blocchi un blocco può restare in memoria più a lungo e può sfruttare più a lungo la località temporale
  - ➔ C'è anche il problema del costo della cache!

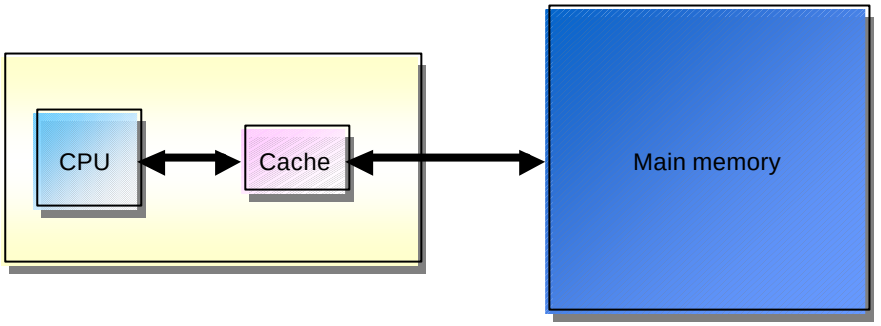
L'Effetto della Località

- Valutiamo l'effetto della località sull'esempio di prima
  - ➔ Tempo di accesso alla cache (hit time) 1 unità ($TA_c=1$)
  - ➔ Tempo di accesso alla memoria (miss penalty) 10 unità ($TA_M=10$)
  - ➔ Frequenza di successo (hit ratio, h=0.99)
  - ➔ Frequenza di fallimento (miss ratio, m=1-h=0.01)
  - ➔ Tempo di accesso medio pari a:

$$TA = h \times TA_C + m \times TA_M$$

$$TA = 0.99 \times 1 + 0.01 \times 10 = 1.09$$

L'Effetto della Località



- Il tempo medio di accesso sperimentato dalla CPU per accedere ad un dato in memoria **passa da 9.1 unità a 1.09 unità!**