Organizzazione dei sistemi di calcolo

Prof. Ing. Donato Impedovo

Calcolatori Elettronici

Distinzioni:

Potenza di calcolo e capacità di memorizzazione

Ambiente e scopo per cui sono utilizzati.

In generale si possono distinguere:

Personal Computer (PC): usati come elaboratori di testo, Internet, banche dati, strumenti da ufficio, etc.

Workstation: usati per il calcolo e la programmazione, per la grafica avanzata e la ricerca.

MainFrame: grandi aziende, banche, gestione di complesse reti di computer e di apparecchiature, applicazioni gestionali

Network computer: computer collegati in rete condividendo dati (dischi) e altre risorse. I “terminali” sono postazioni prive di capacità di elaborazione, dotate solo di monitor e tastiera e collegate ad un computer centrale di cui sfruttano la CPU e la memoria.

Componenti principali di un Computer

Architettura dei Calcolatori e Dispositivi I/O

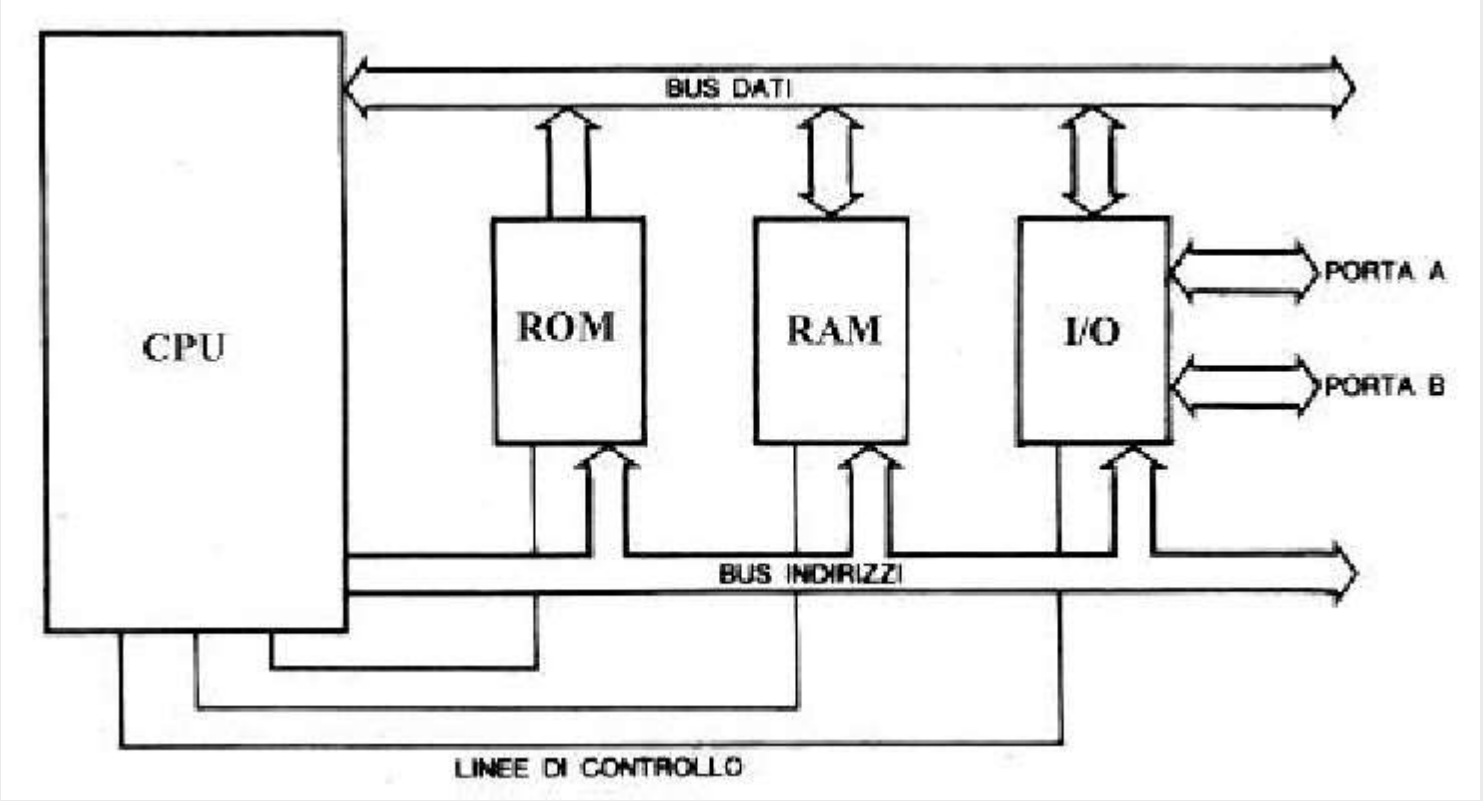
- Architettura di Von-Neuman (estesa)
  - CPU
  - ROM
  - RAM
  - Dispositivi di I/O:
    - Monitor
    - Hard Disk
    - Drive CD/DVD
    - Stampanti
  - BUS

- Architettura o schema di progettazione di calcolatori elettronici che prende nome dal matematico John Von Neuman.
- Schematizzazione molto sintetica, ma molto potente: i moderni computer (Personal Computer - PC) sono progettati secondo tale architettura.

Hardware: parte fisica di un personal computer: circuiti elettrici ed elettronici, cavi, supporti, schede, monitor, tastiera, dischi, stampanti, etc…

–*hard* (duro), *ware* (manufatto, oggetto)

Macchina di Von Neuman (estesa)



Scheda Madre

Raccoglie la circuiteria elettronica di interfaccia fra le componenti principali e fra queste e i bus di espansione e le interfacce verso l'esterno. È' responsabile della trasmissione e temporizzazione corretta dei segnali.

Principali componenti *montati* sulla scheda madre (tramite slot):

- Processore
- RAM
- ROM (BIOS)
- scheda video, scheda audio

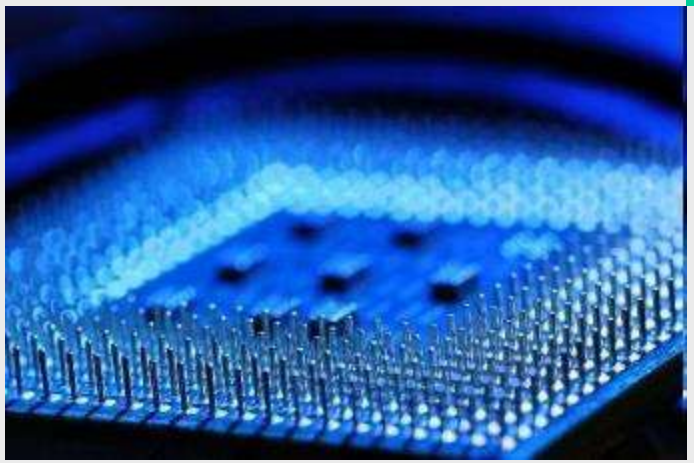
Collegati tramite cavi alla scheda madre:

- l'hard disk
- lettore di DVD
- Alimentatore
- slot lettura penne USB, etc.

Central Processing Units (CPU): Processore

La CPU esegue le istruzioni di un programma (che deve essere presente in memoria RAM, ROM o CACHE).

Durante l'esecuzione del programma, la CPU legge o scrive dati in memoria; il risultato dell'esecuzione dipende dal dato su cui opera e dallo stato interno della CPU stessa, che tiene traccia delle passate operazioni.



Elementi fondamentali:

Registri: locazioni di memoria interne alla CPU, molto veloci, a cui è possibile accedere molto più rapidamente che alla memoria centrale. Il valore complessivo di tutti i registri della CPU costituisce lo stato in cui essa si trova in un determinato istante. (Intel: circa 20 registri)

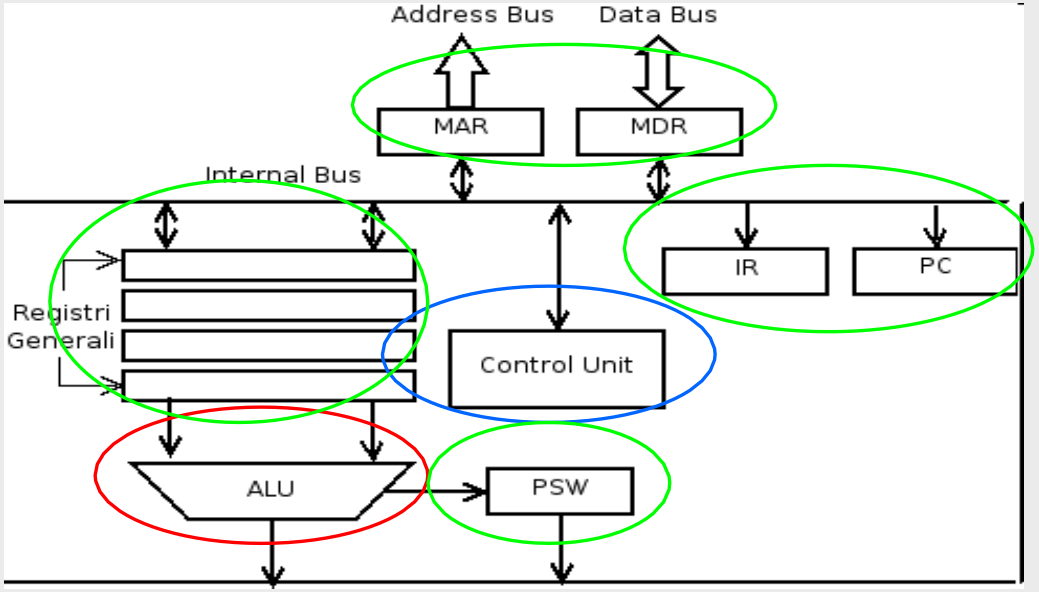
ALU (Arithmetic Logic Unit): esegue le operazioni logiche e aritmetiche.

Unità di controllo CU (Control Unit): "legge" dalla memoria le istruzioni, le decodifica, se occorre legge i dati per l'istruzione, "esegue" l'istruzione e "memorizza" il risultato se c'è, scrivendolo in memoria o in un registro della CPU.

Central Processing Units (CPU): Processore

Componenti principali:

- registri,
- l'unità logica/aritmetica (ALU)
- l'unità di controllo



Svolge semplici azioni in maniera sequenziale.

Lunghezza parola: 32 vs. 64 bit

Prestazioni di un processore:

- Velocità del clock [GHz].
- Istruzioni al secondo MIPS
  es. Pentium 3.8GHz fa circa 1000MIPS (1GIPS)

I registri del processore

- Visibili all'utente
  - Disponibili per tutti i programmi
  - Permettono di minimizzare i riferimenti alla memoria principale
  - Tipicamente disponibili sono quelli di:
    - dati
      - general purpose: *utilizzati per memorizzare temporaneamente dati e/o informazioni di controllo di vario genere*
      - Dedicati: *dedicati sempre ed esclusivamente ad uno scopo*
    - indirizzi (es: index register, segment pointer, stack pointer)
    - codici di condizione (flag - *parzialmente visibili*)

- Di stato e controllo
  - Memorizzano l'esito delle operazioni del processore
  - Solo ad alcuni programmi è possibile accedere tramite istruzioni eseguite in modalità di controllo o di sistema operativo

I registri del processore

- Scambio dati con la memoria:
  - **MAR** (Memory Address Register): indirizzo di riferimento alla memoria
  - **MDR** (Memory Data Register): dati provenienti/da inviare alla memoria

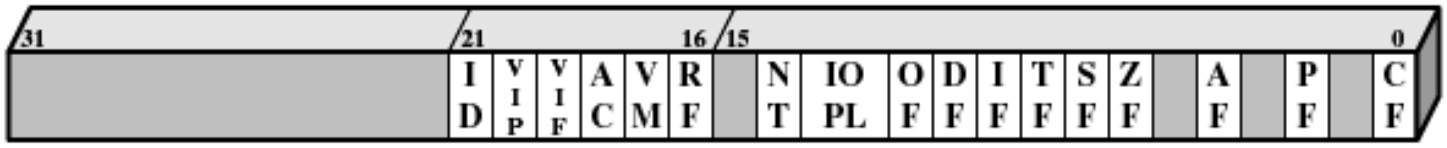
- Scambio dati con i moduli di I/O:
  - **I/O AR** (Input Output Address Register): specifica il dispositivo di I/O
  - **I/O BR** (Input Output Buffer Register): contiene i dati da scambiare con il dispositivo

- Esecuzione delle istruzioni:
  - **PC** (Program Counter): indirizzo della successiva istruzione da eseguire
  - **IR** (Instruction Register): istruzione corrente

- Controllo dell'esecuzione:
  - **PSW** (Program Status Word)
    - informazioni di stato (abilitazione/disabilitazione di interrupt, bit selezione SU o utente)
    - NB: alcune info di controllo sono specifiche per un SO (*designed for…*)

I registri del processore

Registro i-flag del Pentium:

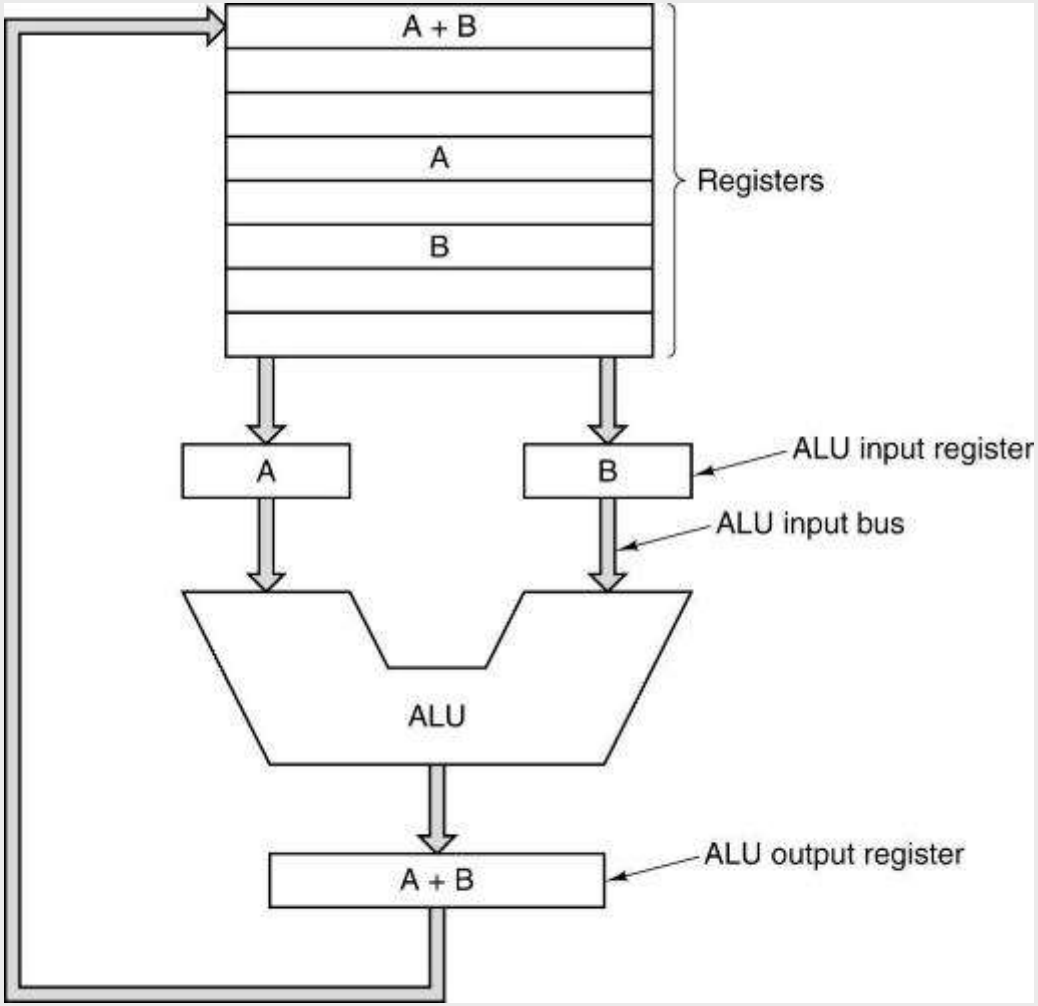


ID = Identification flag
VIP = Virtual interrupt pending
VIF = Virtual interrupt flag
AC = Alignment check
VM = Virtual 8086 mode
RF = Resume flag
NT = Nested task flag
IOPL = I/O privilege level
OF = Overflow flag

DF = Direction flag
IF = Interrupt enable flag
TF = Trap flag
SF = Sign flag
ZF = Zero flag
AF = Auxiliary carry flag
PF = Parity flag
CF = Carry flag

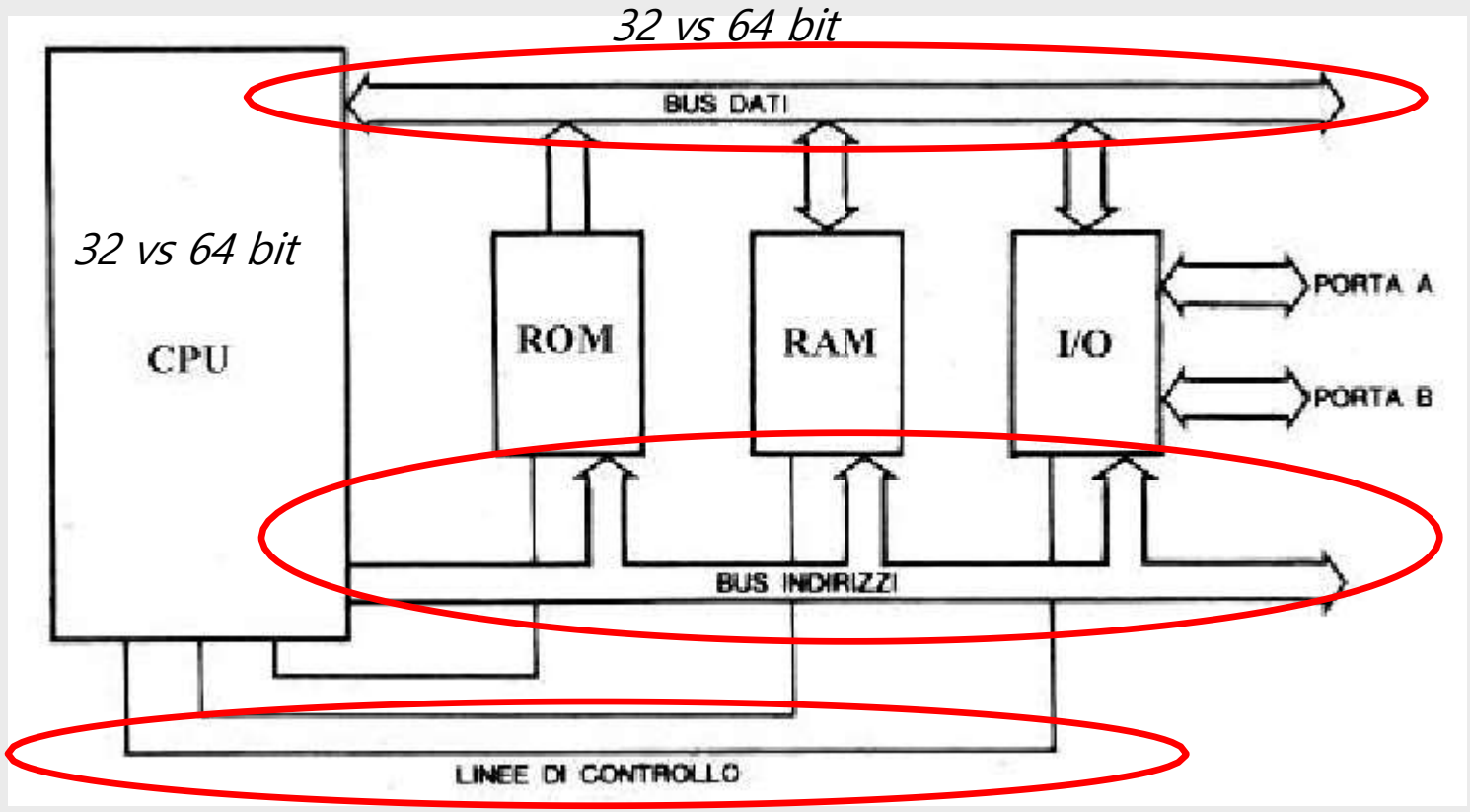
Figure 3.12 Pentium II EFLAGS Register

Data path in una macchina di Von Neumann (dati nei registri)



Il BUS

BUS: piste su circuito stampato utilizzate per trasportare i segnali elettrici che rappresentano i bit



Il BUS

BUS DATI: canale di comunicazione attraverso il quale “viaggiano” i dati .

usufruibile da tutti i componenti del sistema (scrittura/lettura)

bidirezionale.

Caratterizzato dall’ampiezza in termini di bit.

Un bus a 32(64) bit può “trasportare” 32(64) bit alla volta.

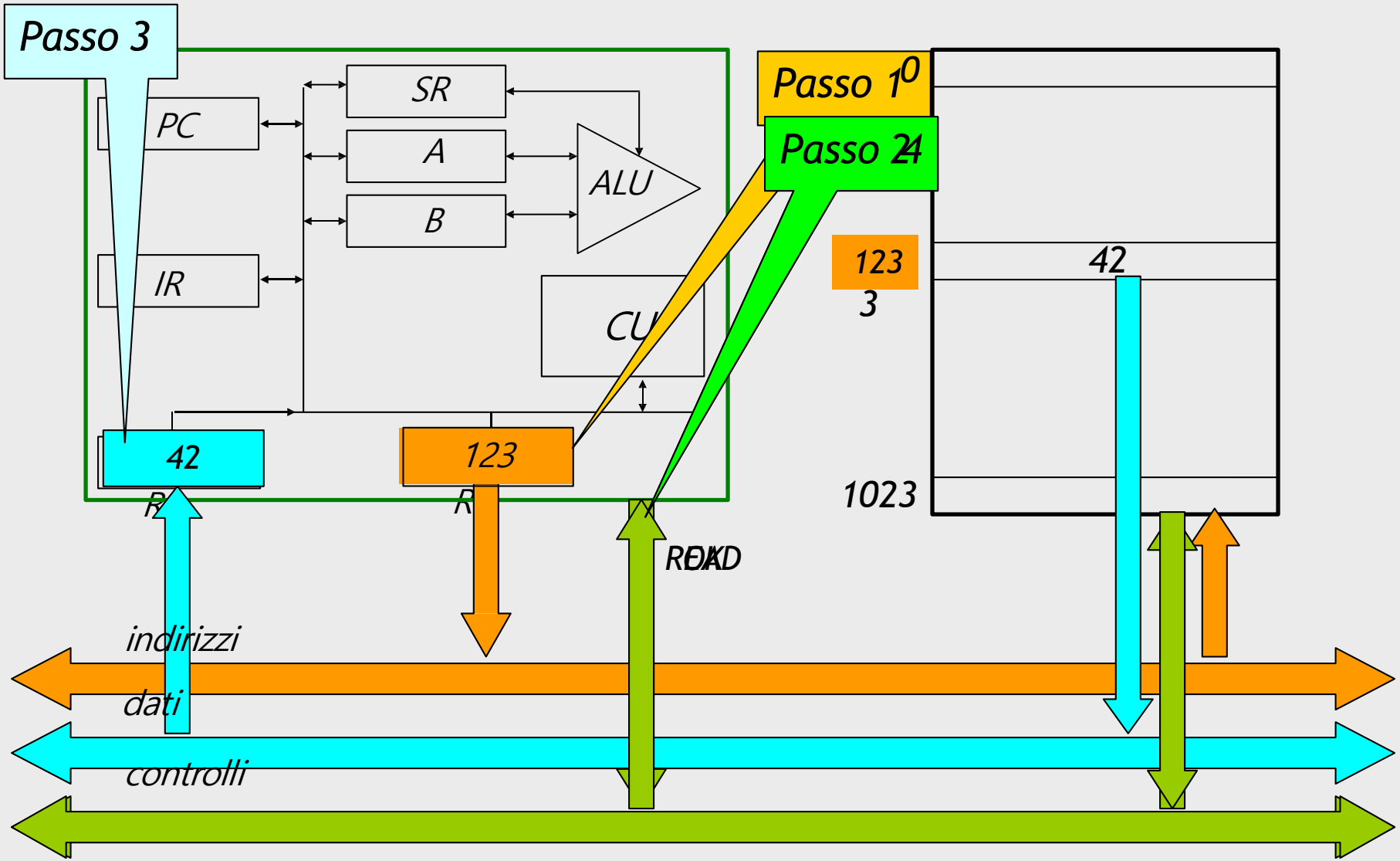
BUS INDIRIZZI: canale attraverso il quale la CPU specifica in quale indirizzo (celle di memoria RAM o periferiche di I/O) andare a scrivere/leggere dati

fruibile in scrittura solo dalla CPU ed in lettura dagli altri componenti

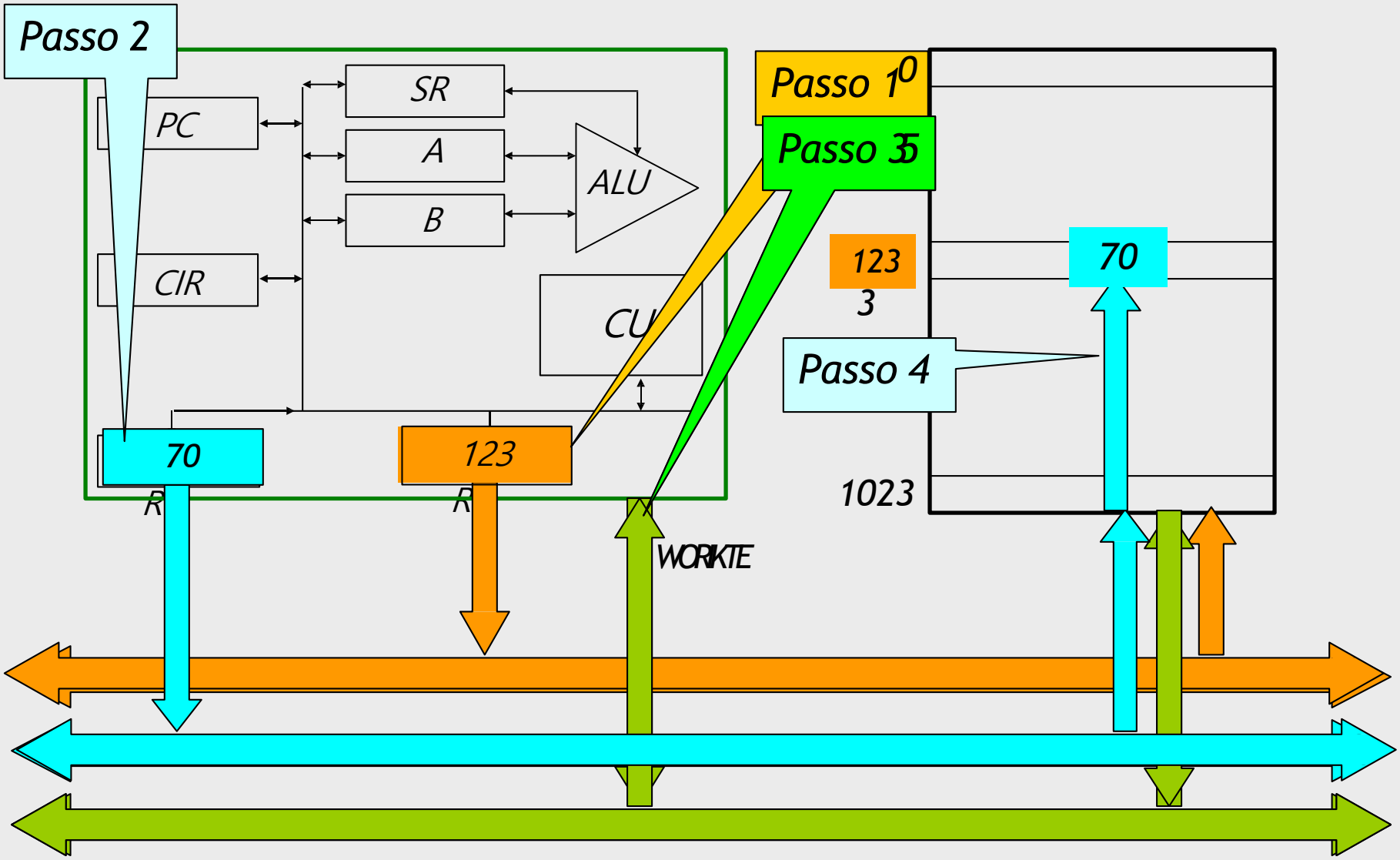
Monodirezionale

BUS CONTROLLI: insieme di collegamenti tramite i quali è possibile coordinare le attività del sistema; tramite esso, la CPU può decidere quale componente deve scrivere sul bus dati in un determinato momento, quale deve leggere l'indirizzo sul bus indirizzi, quali celle di memoria devono scrivere e quali invece leggere, etc.

La sequenza di lettura



La sequenza di scrittura



Central Processing Units (CPU): Processore

I processori possono implementare al loro interno più unità di esecuzione per eseguire più operazioni contemporaneamente. Questo approccio incrementa le prestazioni delle CPU ma ne complica l'esecuzione: per poter eseguire in modo efficiente più operazioni in parallelo la CPU deve poter organizzare le istruzioni.

La possibilità di disporre di più CPU permette al sistema operativo di far eseguire in parallelo più programmi aumentando notevolmente le prestazioni (*multi threading*)

- Es. processori multicore: CPU **dual core**: due processori “indipendenti”. Aumento della la potenza di calcolo senza aumentare la frequenza di clock. Minore calore, minore energia assorbita.
- Intel core i5 4.8GT/s (trasferimenti al secondo), i7(segmento superiore, ma più calore), AMD Athlon (NB: confrontabilità tra marche diverse sulla base dei GHz….)

Istruzioni

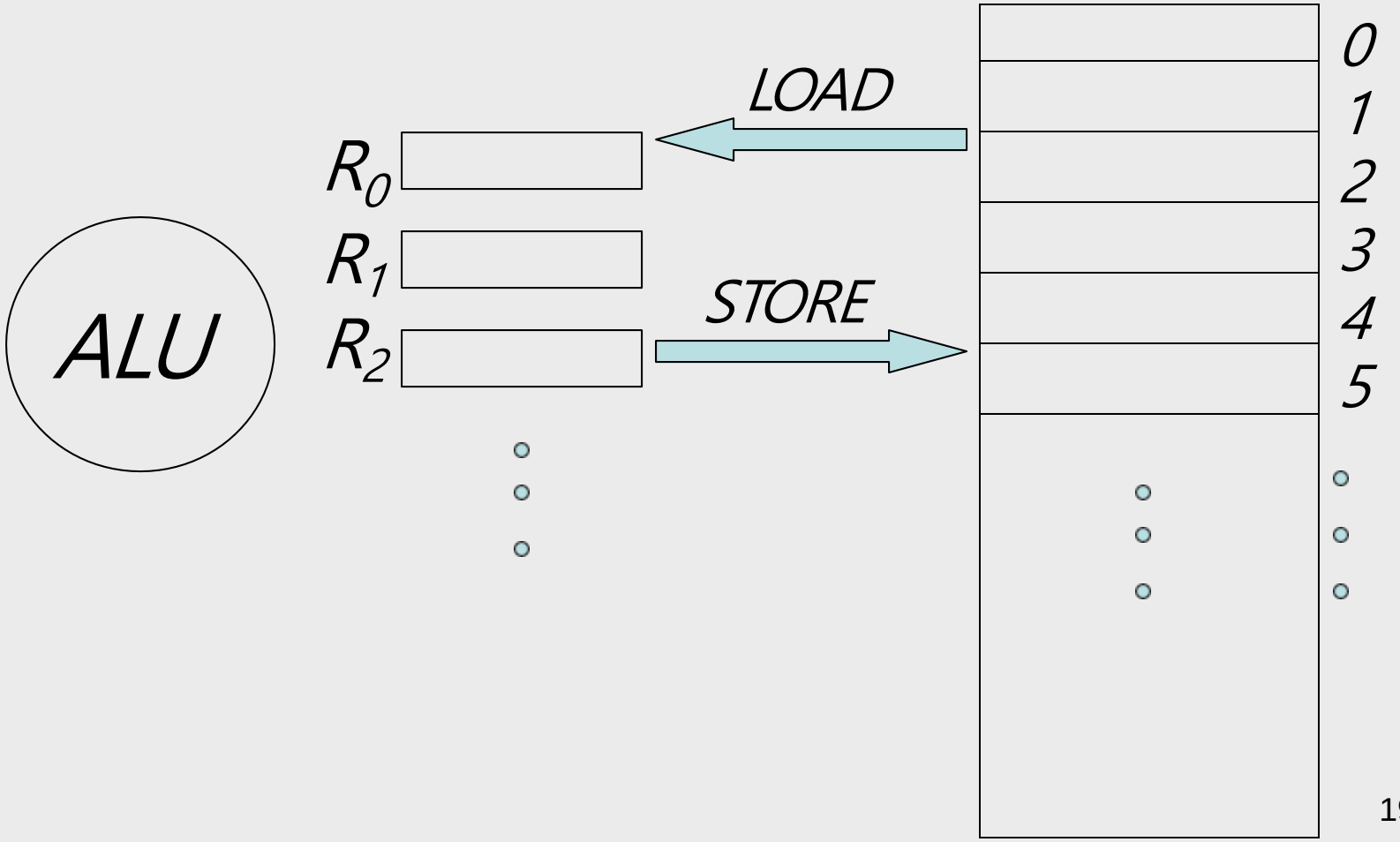
- Un programma è una sequenza di istruzioni
- Un istruzione è un comando, eventualmente opera su dati

Tipi di istruzioni:

- Trasferimento
  - processore – memoria (e viceversa)
  - processore - modulo di I/O (e viceversa)
- Elaborazione di dati (operazioni logico-aritmetiche)
- Controllo (modifica della sequenza di esecuzione)
- Una istruzione può contenere una combinazione delle precedenti possibilità
- Il set di istruzioni dipende dal processore

Formato delle Istruzioni
Alcuni esempi

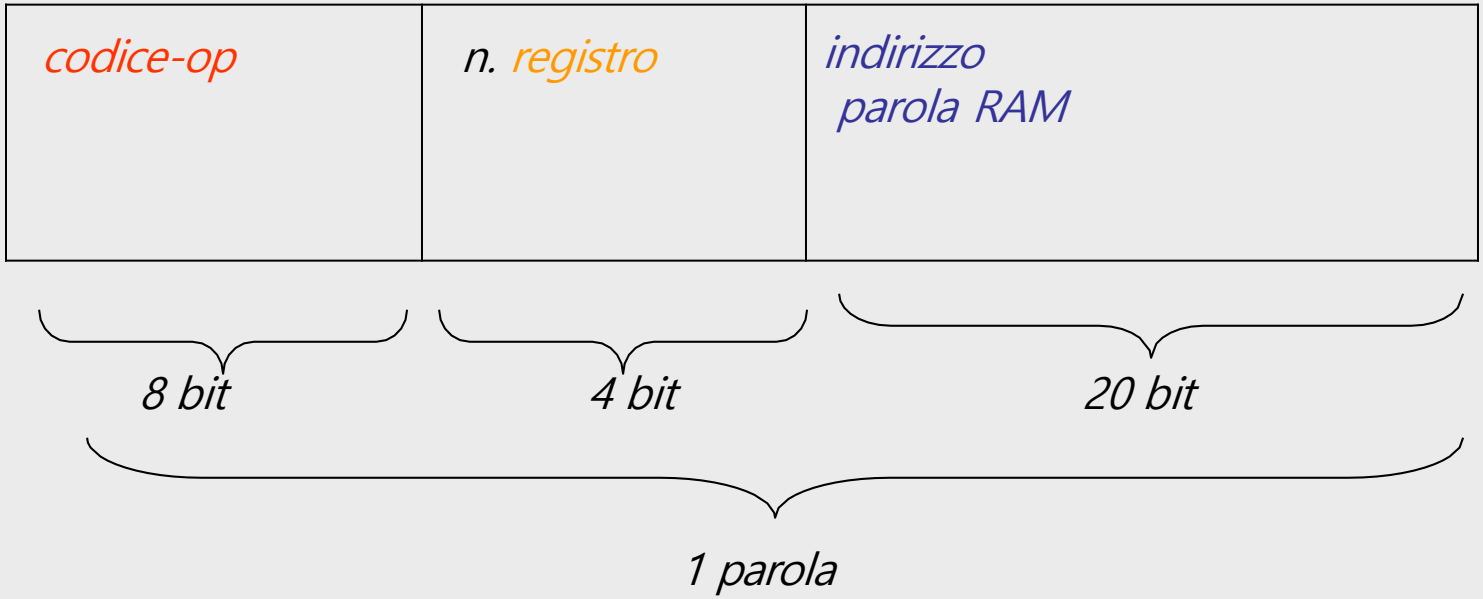
- Trasferimento REGISTRI <-> RAM



Formato delle Istruzioni
Alcuni esempi

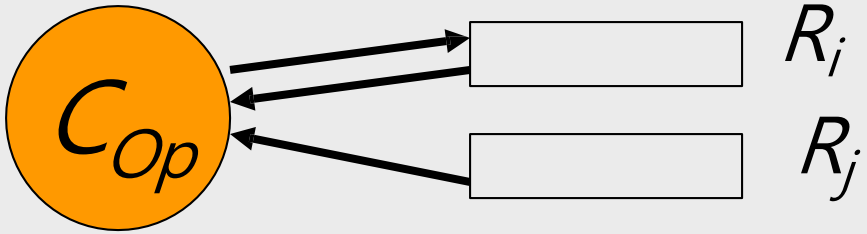
- Trasferimento REGISTRI <-> RAM

in binario!



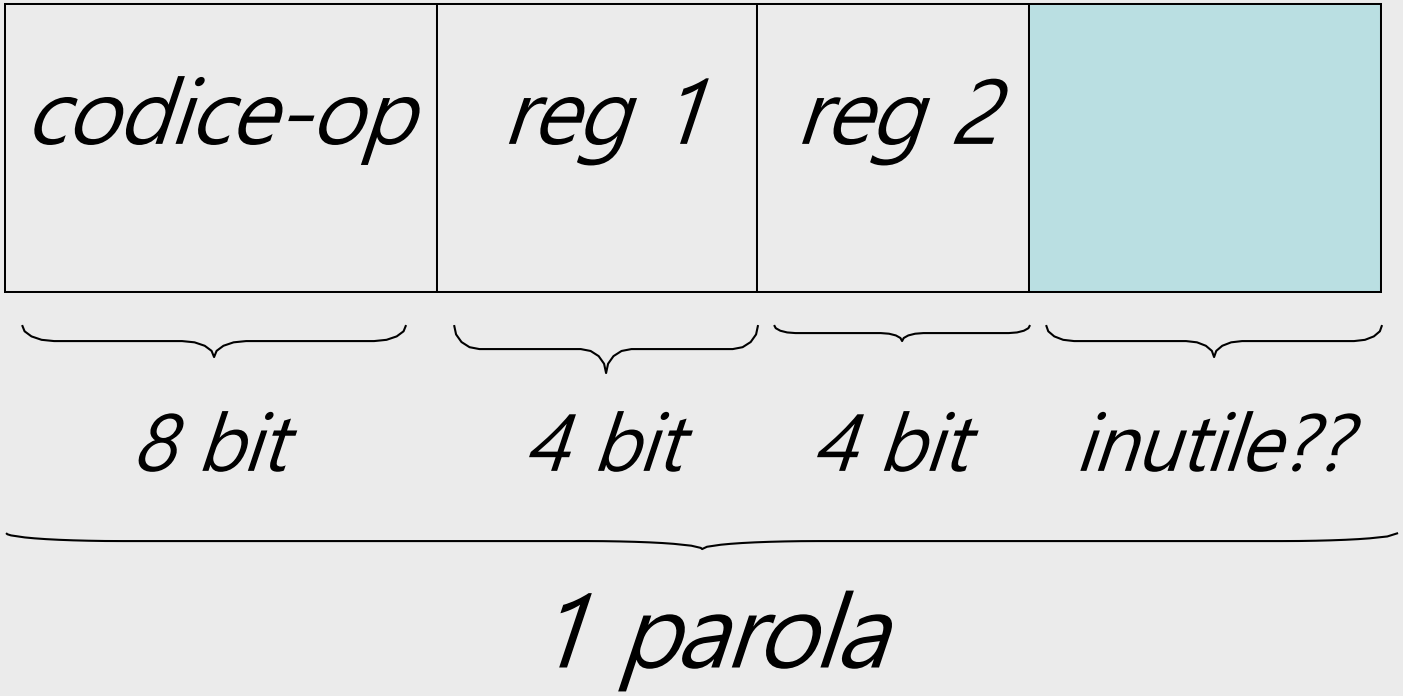
Formato delle Istruzioni
Alcuni esempi

- Operazioni aritmetiche: eseguono somma, differenza, moltiplicazione e divisione usando i registri del processore come operandi



ADD	**00000010**	**SUB**	**00000100**
MULT	**00000110**	**DIV**	**00001000**
MOD	**00001010**		

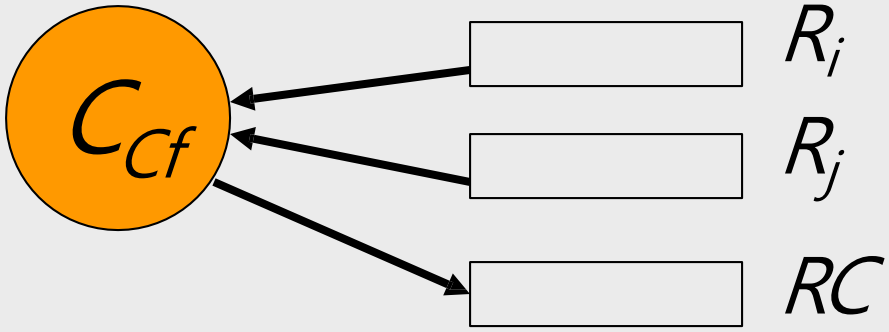
- Operazioni aritmetiche: eseguono somma, differenza, moltiplicazione e divisione usando i registri del processore come operandi



Formato delle Istruzioni
Alcuni esempi

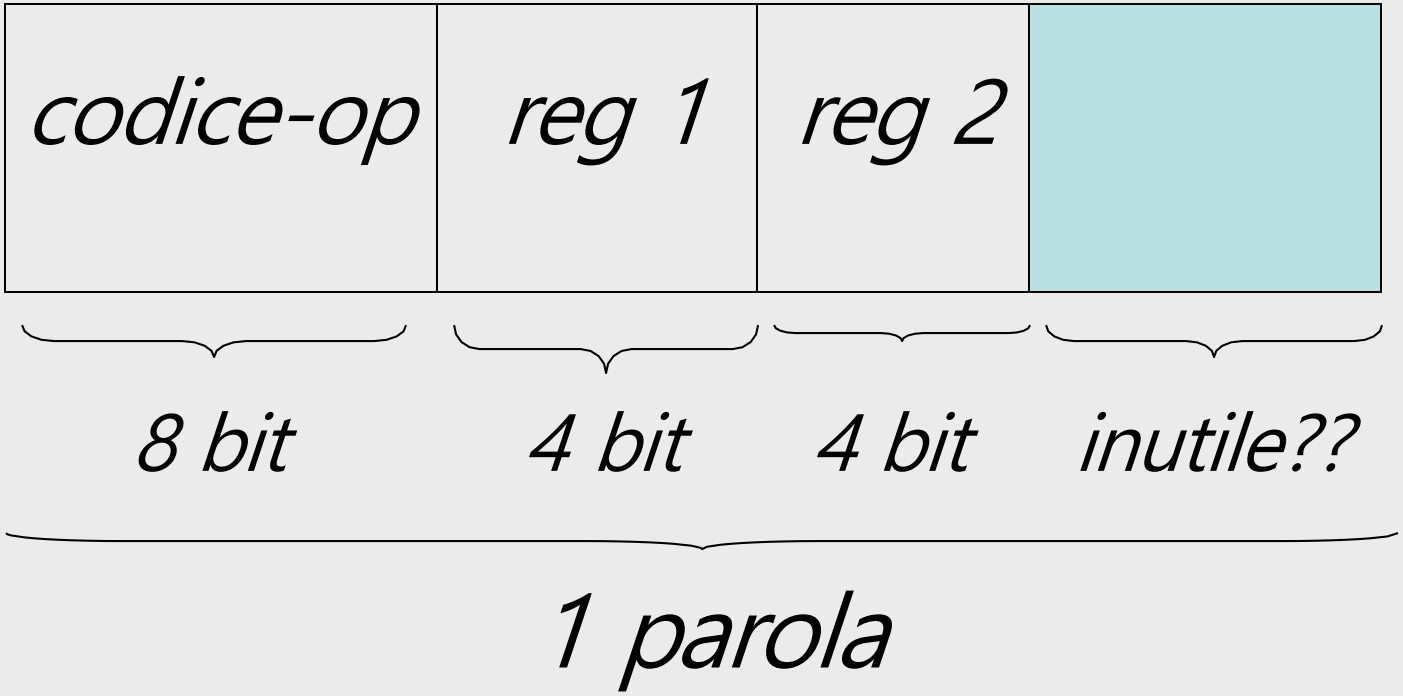
Confronto: paragona il contenuto di 2 registri R_i ed R_j e:

- se $R_i < R_j$ mette -1 nel registro RC
- se $R_i = R_j$ mette 0 in RC
- *se $R_i > R_j$ mette 1 in RC*



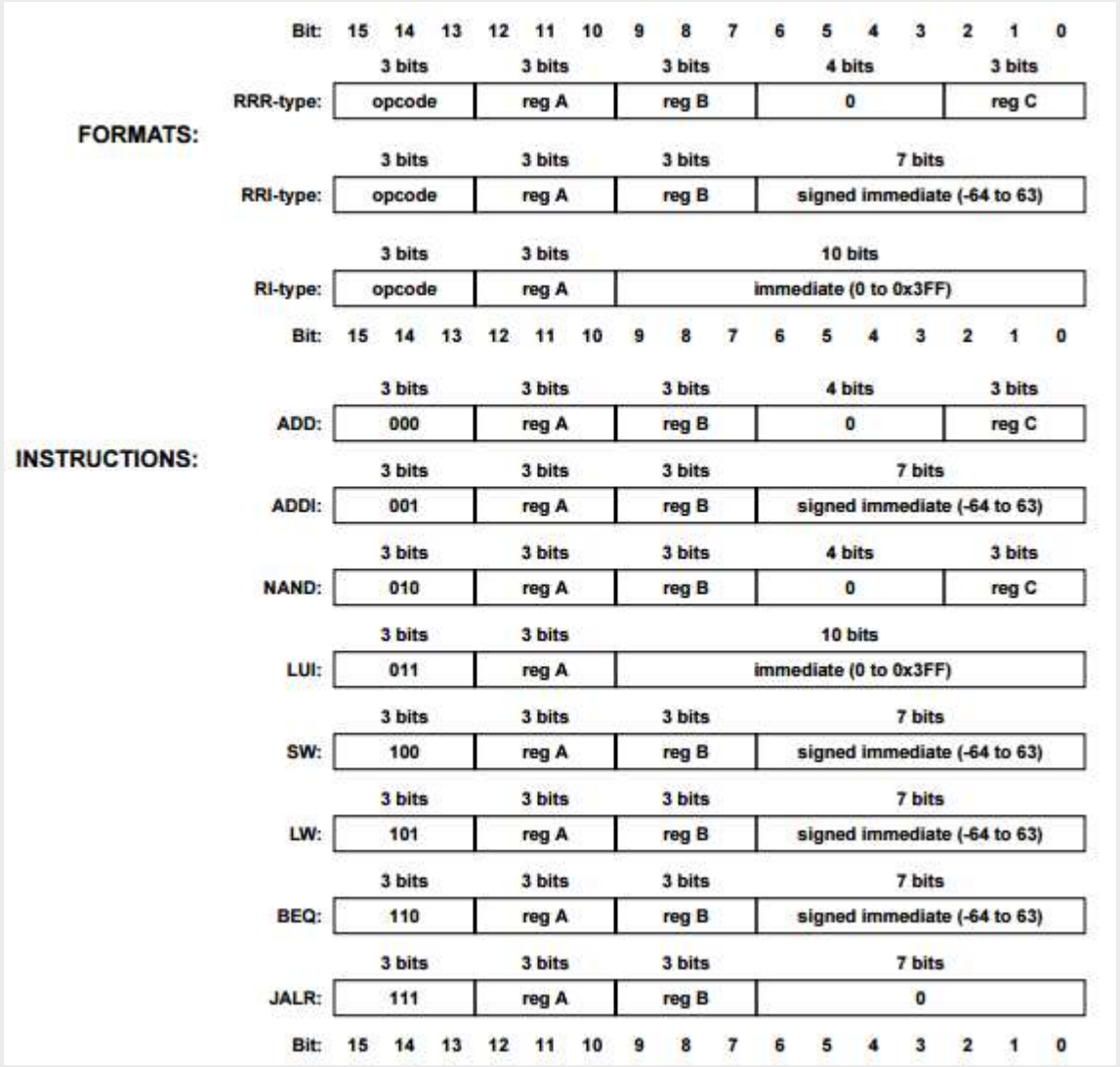
Codici: **`COMP 00100000`**

Formato delle Istruzioni
Alcuni esempi



Formato delle Istruzioni
Alcuni esempi

RISC «didattico»



Formato delle Istruzioni
Alcuni esempi

RISC CMPE 414/CMSC 691V

GENERAL:

5 bit opcode
3 bit operand fields (Rd, Rs1, Rs2) to specify one of the 8 general purpose registers.
Immediates are restricted in length to the number of bit positions remaining in a 16 bit instruction word.
ALL instructions are 16 bits in length.
Shift immediates can be restricted to 2 bits.

INSTRUCTIONS:

Rd - destination
Rs1 - source 1
Rs2 - source 2
- 5/8/11-bit immediate

Formato delle Istruzioni
Alcuni esempi

RISC CMPE 414/CMSC 691V

Table 1: OPCODES

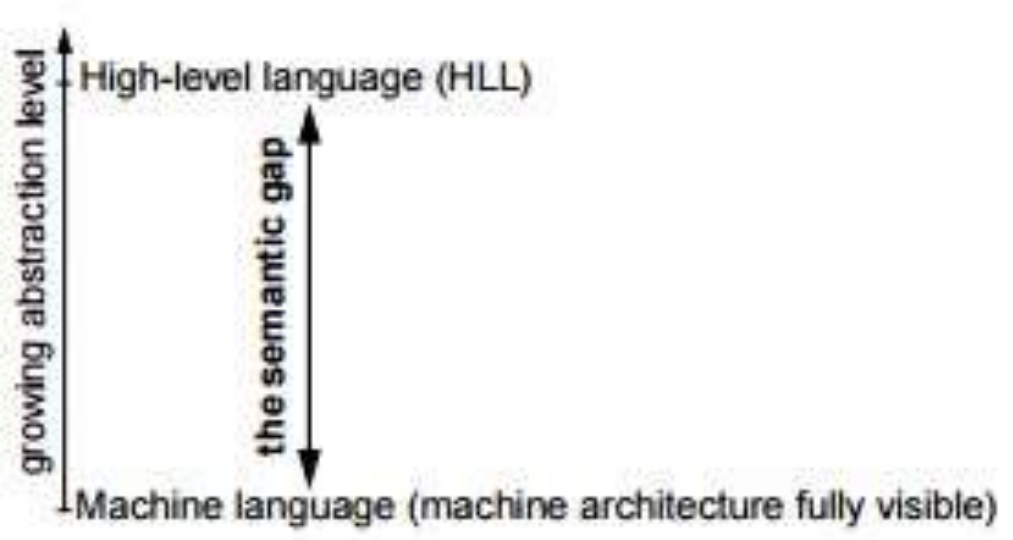
Name	Opcode	Format	Notes
SEQ (SetIfEq)	00000	SEQ Rd Rs1 Rs2	Set register Rd to 1 if the values in Rs1 and Rs2 are equal, else set to 0.
ADD	00001	ADD Rd Rs1 Rs2	Add registers Rs1 and Rs2 and store the result in Rd.
SGT (SetIfGtr)	00010	SGT Rd Rs1 Rs2	Set register Rd to 1 if Rs1 > Rs2, else set to 0.
ADDI	00011	ADDI Rd Rs #	Store in register Rd the value Rs + the 5-bit **sign-extended** immediate.
SUB	00100	SUB Rd Rs1 Rs2	Compute Rs1 - Rs2 and store the result in Rd.
SLT (SetIfLes)	00101	SLT Rd Rs1 Rs2	Set register Rd to 1 if Rs1 < Rs2, else set to 0.
Unused	00110		
OR	00111	OR Rd Rs1 Rs2	Store in register Rd the bitwise OR of Rs1 and Rs2.
ORI	01000	ORI Rd Rs #	Store in register Rd the bitwise OR of Rs and the 5-bit **zero-extended** immediate.
AND	01001	AND Rd Rs1 Rs2	Store in register Rd the bitwise AND of Rs1 and Rs2.
ANDI	01010	ANDI Rd Rs #	Store in register Rd the bitwise AND of Rs and the 5-bit **zero-extended** immediate.
XOR	01011	XOR Rd Rs1 Rs2	Store in register Rd the bitwise XOR of Rs1 and Rs2.
XNOR	01100	XNOR Rd Rs1 Rs2	Store in register Rd the bitwise XNOR of Rs1 and Rs2.
NOT	01101	NOT Rd Rs	Store in register Rd the bitwise complement Rs.
SRA (ShftRgtArith)	01110	SRA Rd Rs #	Store in register Rd the **sign-extended** value of Rs shifted to the right by the 2-bit immediate. (An immediate of 0 does not shift the operand).

Formato delle Istruzioni
Alcuni esempi

RISC CMPE 414/CMSC 691V

Name	Opcode	Format	Notes
SRL (ShftRghtLogic)	01111	SRL Rd Rs #	Store in register Rd the **zero-extended** value of Rs shifted to the right by the 2-bit immediate.
SLL (ShftLeftLogic)	10000	RLL Rd Rs #	Store in register Rd the **zero-extended** value of Rs shifted to the left by the 2-bit immediate.
SW (StoreWord)	10001	SW 0 Rs Raddr	Store to memory at address Raddr the value in Rs. (You can assume that address is an even number and the compiler inserts 3 zero bits for Rd).
MUL (Multiply)	10010	MUL Rd Rs1 Rs2	Multiply the lower 8 bits of Rs1 and Rs2 and store the result in Rd.
Unused	10011		
LW (LoadWord)	10100	LR Rd 0 Raddr	Load the word value at memory address Raddr into Rd. (You can assume that address is an even number and the compiler inserts 3 zero bits for Rs).
LBI (LoadBytImmed)	10101	LBI Rd #	Store the 8-bit **sign-extended** immediate into register Rd after sign extending it.
LBIU (LoadBytIUnsign)	10110	LBIU Rd #	Write the 8-bit **zero-extended** immediate into register Rd.
LHI (LoadHighImmed)	10111	LHI Rd #	Write the 8-bit immediate into the upper 8 bits of register Rd and optionally clear the low order 8 bits.
BEQZ (BrIfEqZero)	11000	BEQZ Rs #	Set PC to PC + 8-bit **sign-extended** immediate if the value in register Rs is zero.
BNEZ (BrIfNotZero)	11001	BNEZ Rs #	Set PC to PC + 8-bit **sign-extended** immediate if the value in register Rs is non-zero.
BC (BrIfCarrySet)	11010	BC #	Set PC to PC + 11-bit **sign-extended** immediate if the carry out is set.
BO (BrIfOvrflowSet)	11011	BO #	Set PC to PC + 11-bit **sign-extended** immediate if the overflowis set.
NOP (NoOperation)	11100	NOP	Do nothing.
J (Jump)	11101	J #	Set the PC to PC + 11-bit **sign-extended** immediate.
JR (JumpToRegister)	11110	JR 0 Rs	Set the PC to the value in register Rs.
JALR (JumpAndLink)	11111	JALR Rd #	Rd=PC, PC=PC+#. Save the current value of PC (which points to the NEXT instruction) to register Rd and set PC to PC + 8-bit **sign-extended** immediate.

GAP semantico…



Problema: Come facciamo a compilare un linguaggio di alto livello affinchè possa essere eseguito in maniera efficiente sul calcolatore?

Due possibili risposte:

1. **CISC**: avere una architettura molto complessa che includa un numero elevato di istruzioni e di modi di indirizzamento e che includa anche istruzioni molto vicine a quelle presenti nel linguaggio di alto livello *(…ricordate la storia dell'interprete della prima lezione??)*
2. **RISC** : semplificare il set delle istruzioni e adeguarlo alle necessità dell'utente

Formato delle Istruzioni
Alcuni esempi CISC e RISC

CISC Architectures:

VAX 11/780
Nr. of instructions: 303
Instruction format: not fixed
Addressing modes: 22
Number of general purpose registers: 16

Pentium
Nr. of instructions: 235
Instruction format: not fixed
Addressing modes: 11
Number of general purpose registers: 8

RISC Architectures:

Sun SPARC
Nr. of instructions: 52
Instruction format: fixed
Addressing modes: 2
Number of general purpose registers: up to 520

PowerPC
Nr. of instructions: 206
Instruction format: not fixed (but small differences)
Addressing modes: 2
Number of general purpose registers: 32

Esecuzione di una istruzione

1. Preleva la successiva istruzione dalla memoria e la porta nell'IR
2. Modifica il valore del PC per farlo puntare alla successiva istruzione
3. Determina il tipo di istruzione appena prelevata
4. Se l'istruzione usa una parola in memoria, determina dove si trova
5. Se necessario, preleva la parola dalla memoria per portarla in un registro della CPU
6. Esegue l'istruzione
7. Torna al punto 1 per iniziare l'esecuzione dell'istruzione successiva.

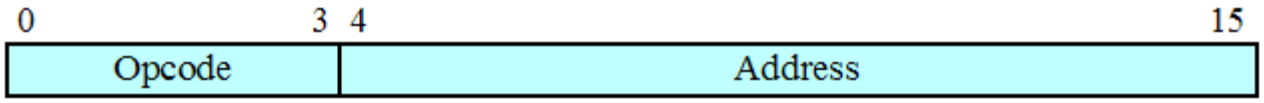
Esecuzione di una istruzione

L'eseguzione di una istruzione avviene attraverso le seguenti fasi:

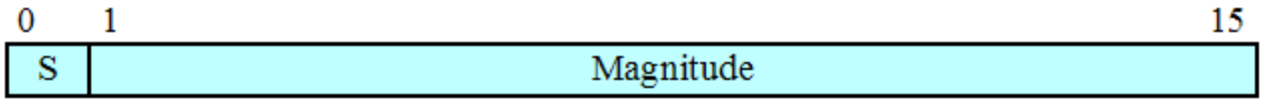
1. **INSTRUCTION FETCH** (Acquisizione dell'istruzione): il processore preleva l'istruzione dalla memoria, specificato dal registro PC (Programm counter), e la carica nell'IR. Viene incrementato il valore del PC.
2. **INSTUCTION DECODE** (Decodifica dell'istruzione): dall'istruzione prelevata viene determinata quale operazione debba essere eseguita e come ottenere gli operandi mediante la conoscenza dei codici operativi (operazione svolta dalla Control Unit)
3. **INSTRUCTION EXECUTE** (Esecuzione dell'istruzione): viene eseguita la computazione indicata
4. **MEMORY ACCESS:** nel caso in cui l'istruzione richieda un accesso alla memoria, questa fase sostituisce o segue la precedente.
5. **WRITE BACK:** scrittura del risultato in memoria

Istruzioni

Consideriamo una macchina ipotetica



(a) Instruction format



(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

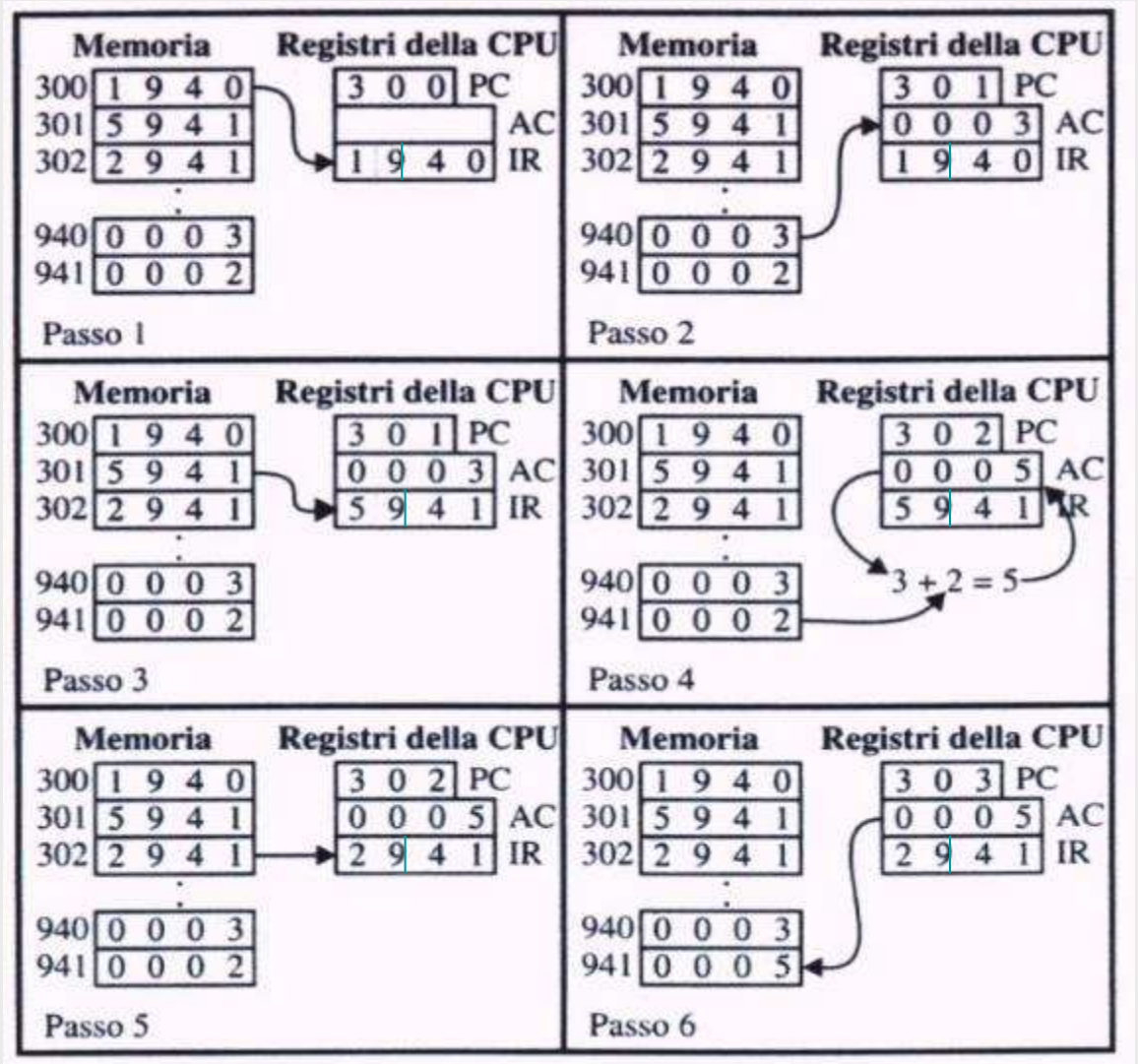
(c) Internal CPU registers

0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory

(d) Partial list of opcodes

Esempio di Esecuzione IF-DEC_EX

IF DEC_EX



Ciclo IF-DEC_EX

Legenda:

PC Program Counter

AC Accumulator

IR Instruction Register

Codici operazioni:

1 Carica in AC

2 Salva in ...

5 Somma ad AC

Esempio di Esecuzione IF-DEC_EX

Registri CPU

PC Program Counter

AC Accumulator

IR Instruction Register

Formato istruzione:

XYYY

X: codice operativo

YYY: riferimento a memoria

Codici operazioni:

X=1: Carica da MEM in AC

X=2: Salva AC in MEM

…

X=6 Sottrai ad AC contenuto MEM

Ciclo IF-DEC_EX

MEM

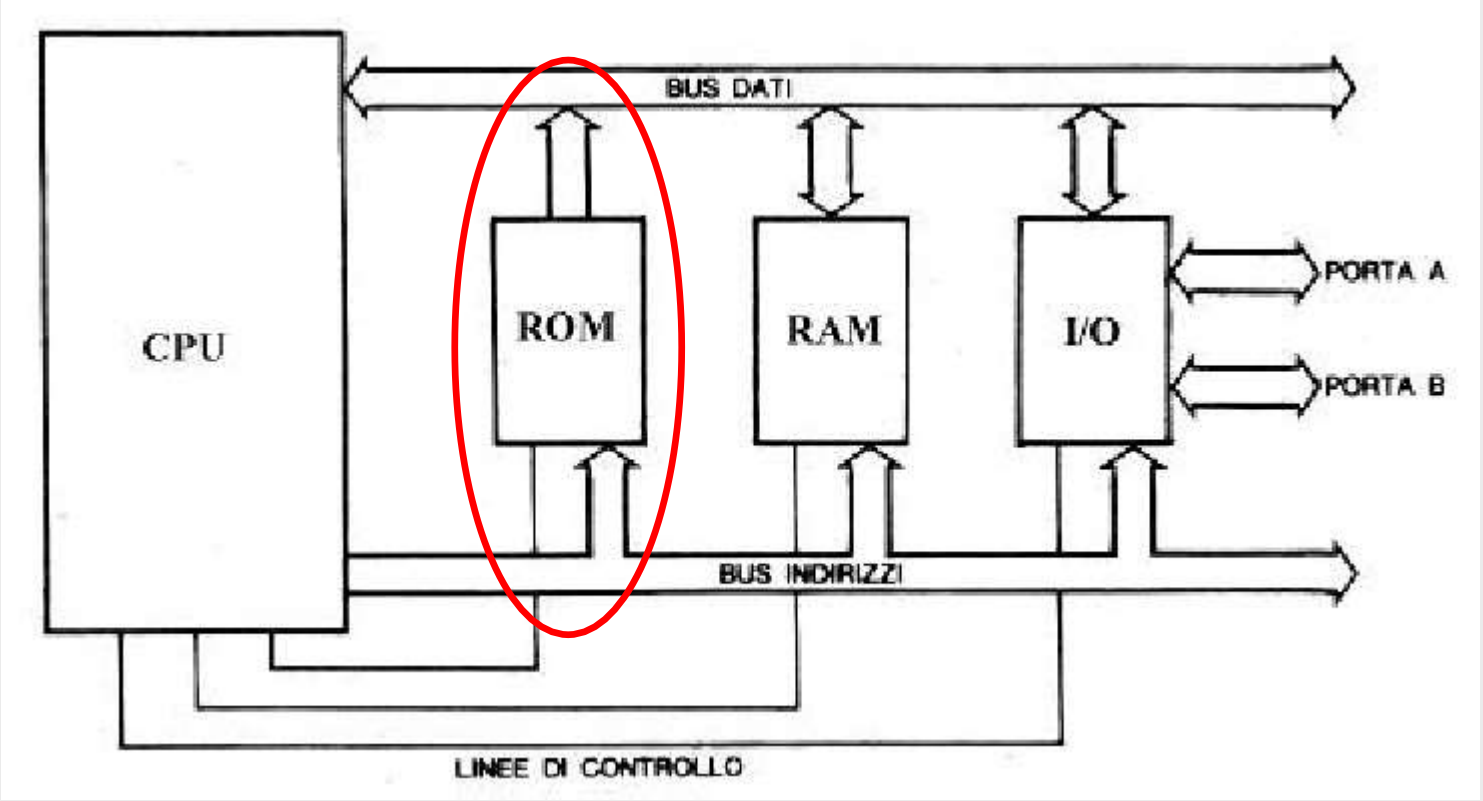
212	1940
213	6941
214	2942
	…
940	0037
941	0002
942	0701

CPU

PC	212
IR	
AC	0021

Si mostri il contenuto dei registri della CPU e della memoria nelle fasi di FETCH e DECODE-EXECUTE con riferimento all'esecuzione sequenziale delle istruzioni contenute in memoria nelle celle 212, 213, 214

ROM: Read Only Memory



ROM (Read Only Memory)

Memoria permanente di sola lettura, scritta in fase di fabbricazione dal costruttore. Non più modificabile.

Contiene:

- le informazioni “di base” (la cui modifica comprometterebbe l’uso della macchina)
- le istruzioni del programma di avviamento (fase di bootstrap) che si attiva all'accensione della macchina.

BIOS (Basic Input/Output System): sequenza di istruzioni di avvio eseguita automaticamente ad ogni accensione del computer. Si compone di 3 fasi:

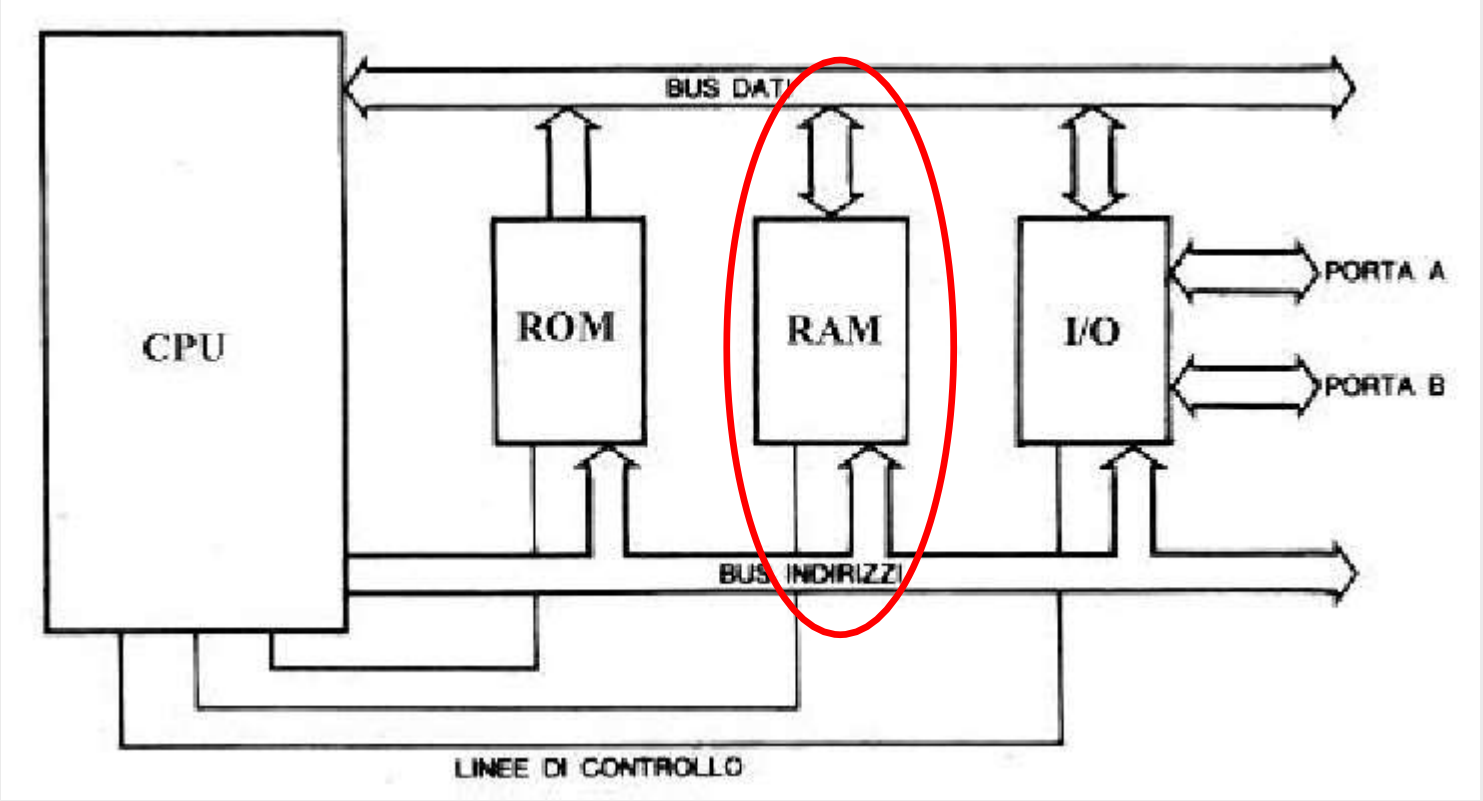
Attivazione dell'hardware installato e test di funzionamento del sistema (verifica dell'hardware)

Verifica della presenza del sistema operativo e suo caricamento

Avvio del primo processo

EPROM, (Electric-Programmable ROM) realizzate in modo da consentire sia la cancellazione che la riscrittura del contenuto. Cancellazione per mezzo di raggi ultravioletti..

RAM: Random Access Memory



RAM: Random Access Memory

Memoria nella quale vengono conservate le istruzioni del programma attualmente in esecuzione e i suoi dati. Durante l'esecuzione i programmi e i dati devono trovarsi almeno parzialmente nella memoria centrale.

Memoria volatile

I documenti che si costituiscono utilizzando un qualsiasi programma vengono posti all'interno della RAM e sono trasferiti sul disco (Hard Disk) solo quando l'utente ne richiede il salvataggio. Se manca la corrente, con lo svuotamento della RAM tutto il lavoro svolto dopo l'ultimo salvataggio viene perduto.

La capacità della RAM si misura in GigaByte (GB), ovvero miliardi di byte.

Dimensioni tipiche: 2 – 16 Gbyte

Frequenza di lavoro: più alta è la frequenza, più velocemente la memoria sarà accessibile. MT/s = milioni di trasferimenti al secondo

RAM: Random Access Memory

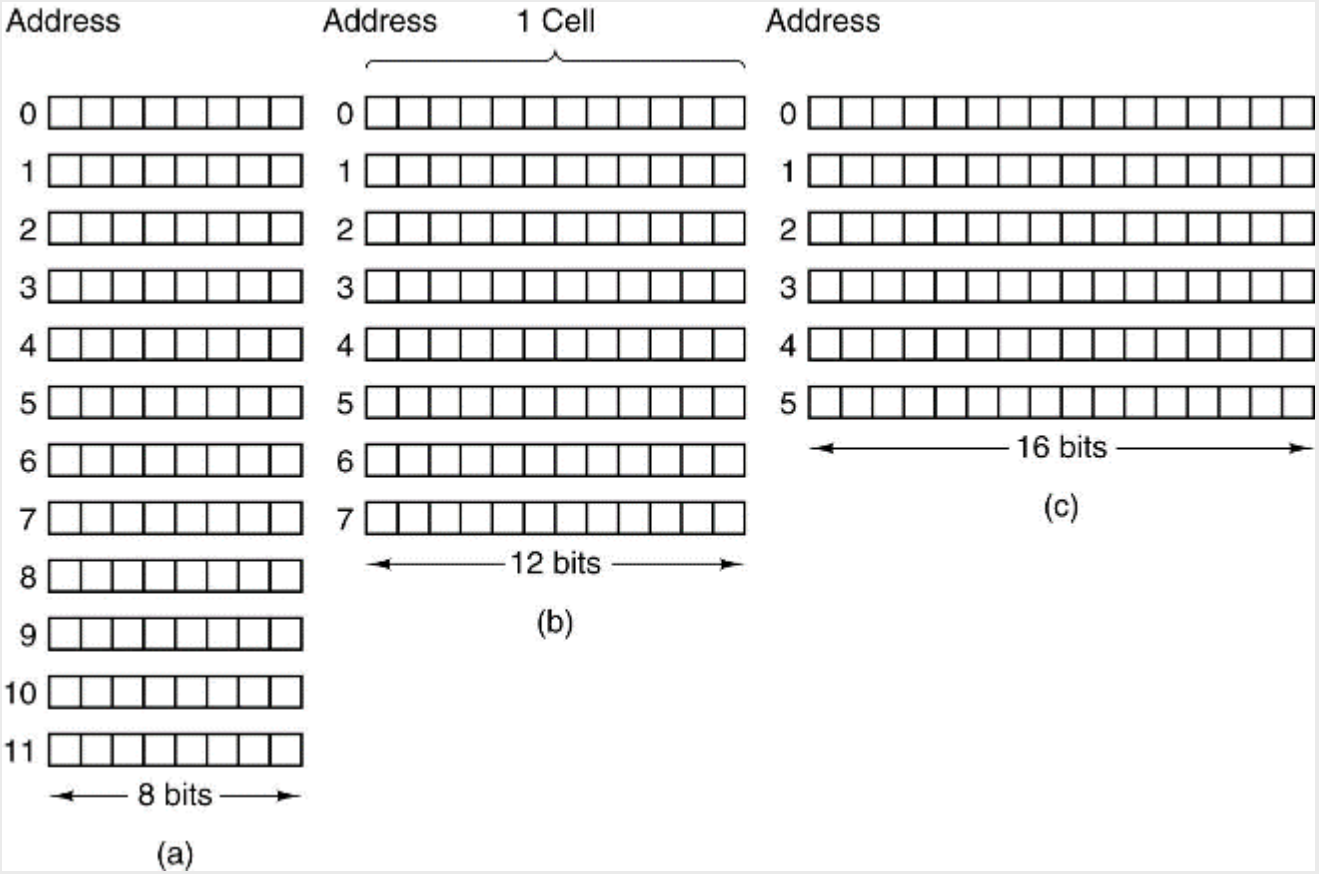
*Le memorie sono costituite da un certo numero di **celle** (o **locazioni**) ciascuna della quali può memorizzare informazioni.*

*Ciascuna cella ha un numero, chiamato **indirizzo**, attraverso il quale un qualsiasi programma può riferirsi ad essa. Se una memoria ha n celle, i suoi indirizzi variano da 0 a n-1.*

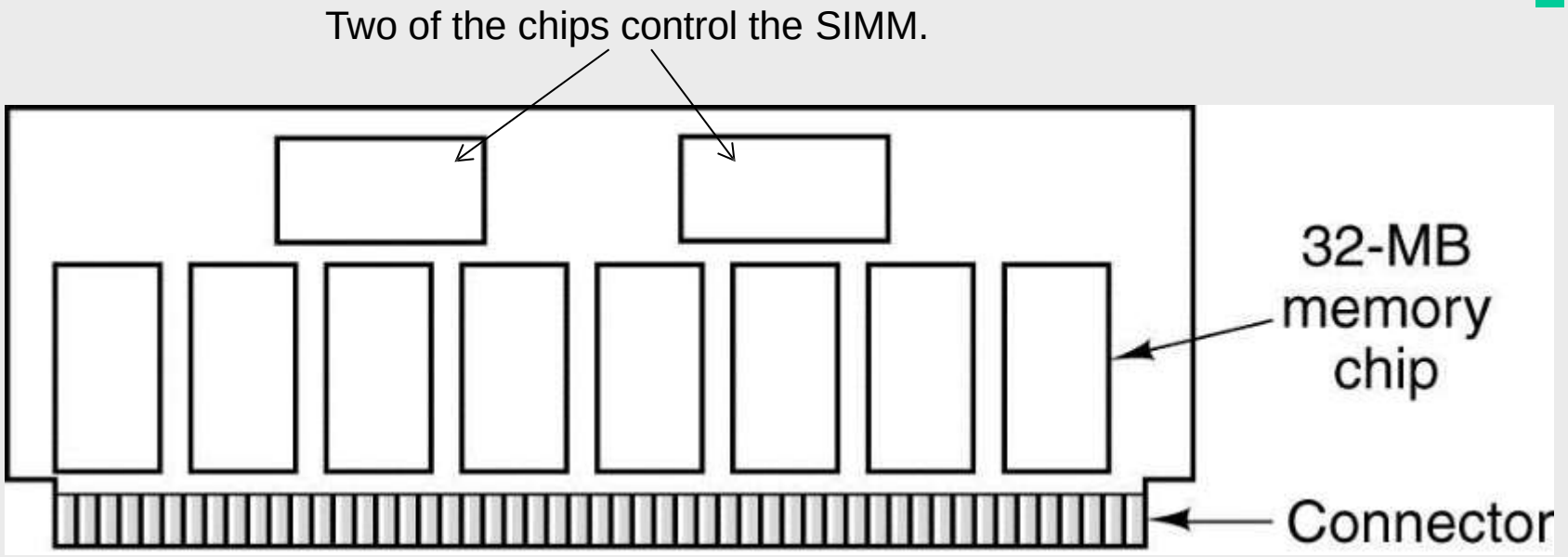
Tutte le celle contengono lo stesso numero di bit. Se una cella contiene k bit allora potrà assumere al massimo 2^k valori.

RAM: Random Access Memory

Indirizzi di memoria e dimensione della “word”: tre modi di organizzare una memoria a 96 bit.



Memory packaging and types



SIMM: Single Inline Memory Module - (tipical: 72 pin -- 32 bit/cycle)

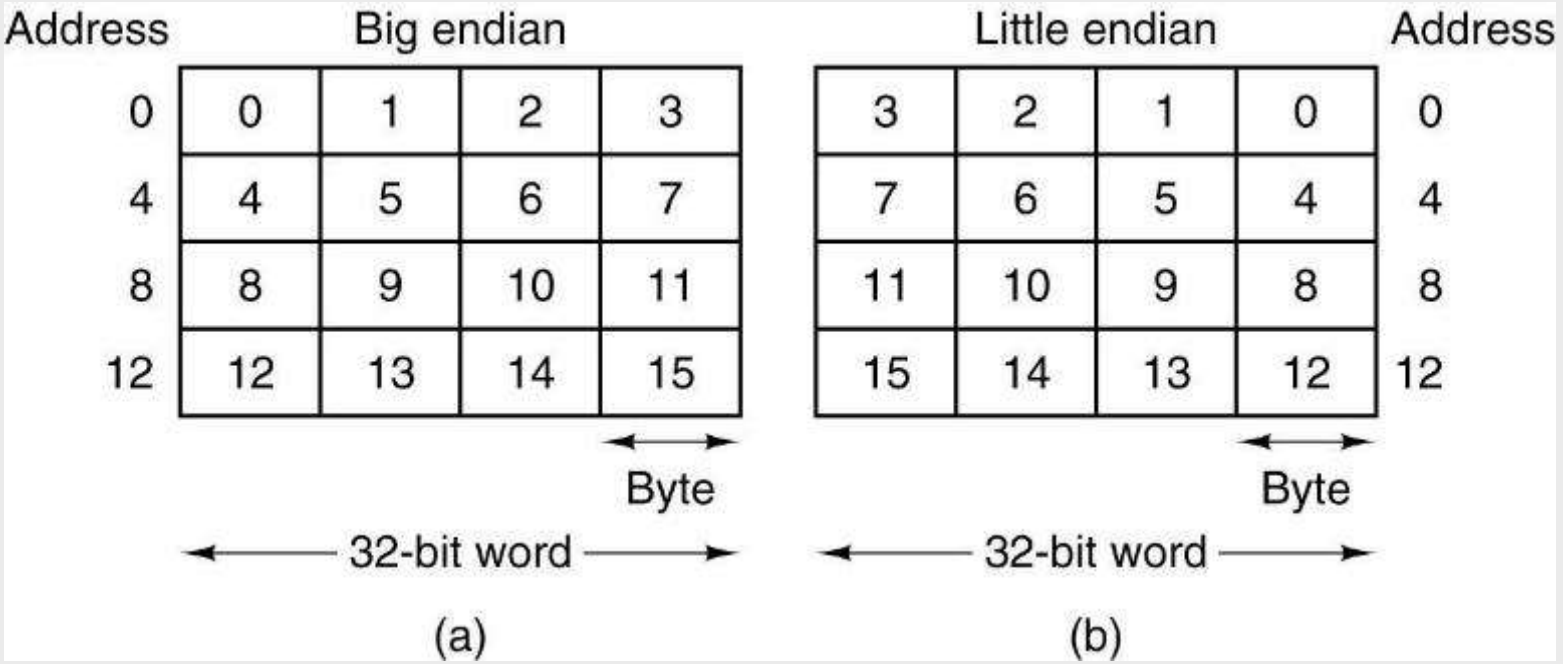
DIMM: Dual Inline Memory Module - (tipical: 84 pin -- 64 bit/cycle)

In un PC se ogni SIMM /DIMM è composta da 8 moduli da 32MB ciascuno si ha:
32MB x 8 Chip = 256 MB (4 SIMM → 1GB)

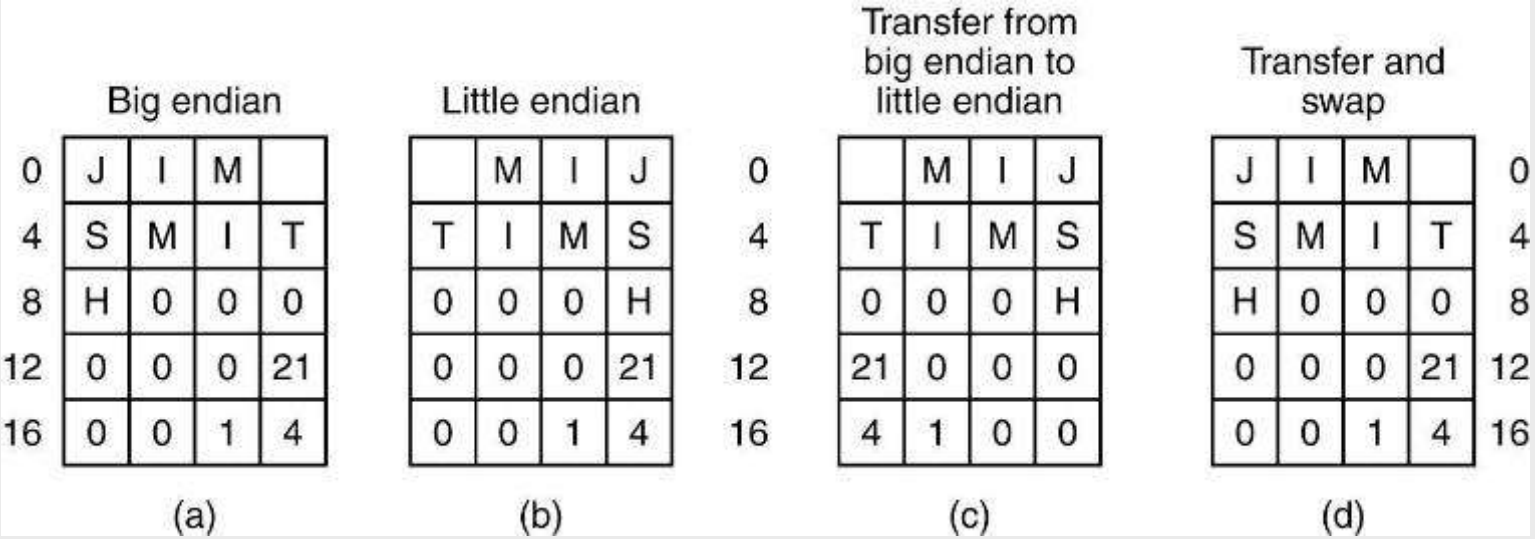
Ordinamento dei Byte

All'interno di una parola di memoria i byte possono essere numerati :

- da sinistra a destra (Big endian – dal byte più significativo al meno significativo)
- da destra a sinistra (Little endian – dal byte meno significativo al più significativo)



Trasferimento dei Byte



(a) Macchina “big endian”.

(b) Macchina “little endian”.

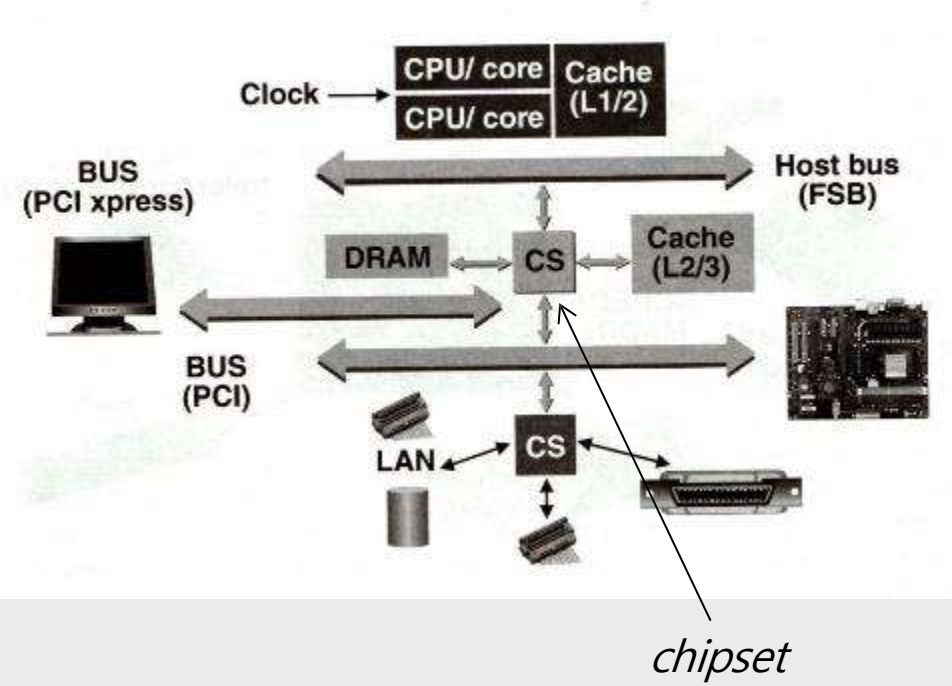
(c) RIsultati del trasferimento dati da “big endian” a “little endian”.

(d) Byte-swapping di (c).

RAM :Random Access Memory

Front Side Bus (FSB): trasporta i dati tra la CPU e i dispositivi veloci, tra cui la RAM, L3$, la memoria della scheda video.
La banda (o throughput) del FSB viene determinata moltiplicando:

- i byte delle word (parole) del processore
- la frequenza di clock (cicli al secondo) del BUS
- il numero di data transfer del BUS ad ogni ciclo.



Es: un sistema con:

- processore a 32 bit (4 byte);
- FSB a 100 MHz;
- 4 trasferimenti a ciclo;

ha una banda di 4(byte) x 100(FSB) x 4(tc) = 1600 megabyte al secondo (MB/s).

Memoria	Clock	Frequenza I/O (FSB)	Velocità trasferimento dati
DDR3 800	100 MHz	400 MHz	800 MT/s
DDR3 1600	200 MHz	800 MHz	1600 MT/s (=1.6GT/s)

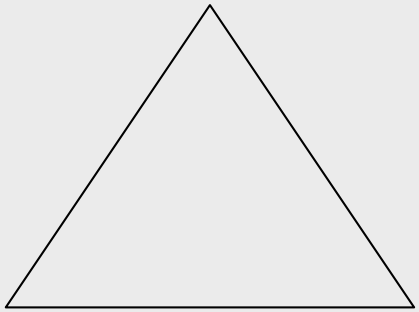
Intel core i5 4.8GT/s (trasferimenti al secondo)!!!!!!!!!

Necessità di una memoria più veloce!

La Memoria

- Parametri:
  - Dimensione (grande)
  - Velocità (elevata)
  - Costo (piccolo)

  Parametri sono in contrasto tra loro!



TRADE-OFF

- Vincoli tecnologici:
  - Tempo di accesso più breve, maggior costo per bit
  - Maggiore capacità, maggiore tempo di accesso
  - Maggiore capacità, minore costo per bit

Gerarchie delle Memorie

Soluzione alla inconciliabilità dei tre parametri

- Uso organizzato e integrato di diversi dispositivi di memoria con diverse caratteristiche

- Consente di:
  - diminuire il costo totale per bit
  - aumentare la capacità del sistema
  - contenere i tempi medi di accesso
  - diminuire la frequenza di accesso ai dispositivi più lenti

Scendendo nella gerarchia:

- Diminuisce il costo per bit
- Aumenta la capacità
- Aumenta il tempo di accesso
- Diminuisce la frequenza di accesso del processore

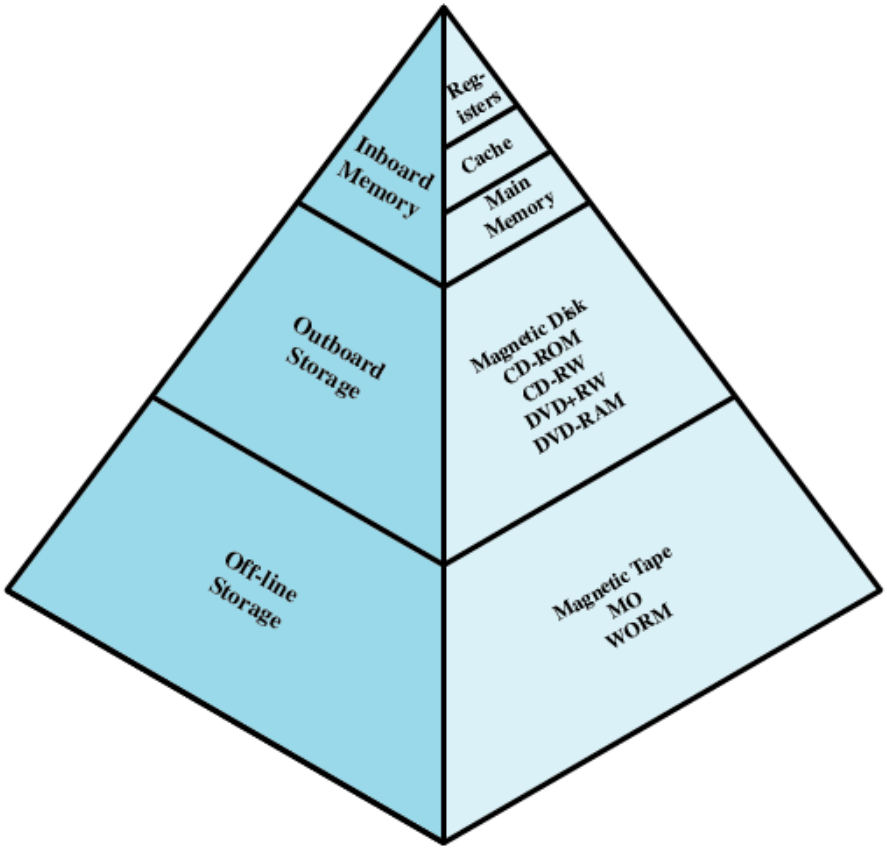


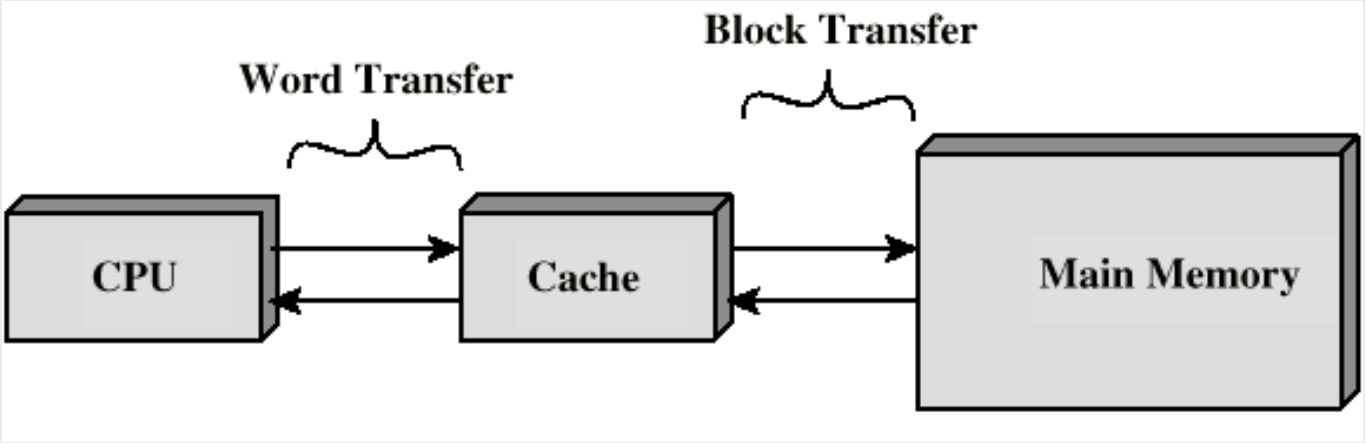
Figure 1.14 The Memory Hierarchy

Gerarchie delle Memorie

Memoria	Standard	Clock	Frequenza I/O	Velocità trasferimento dati	Banda per canale	Banda dual channel
DDR3 800	PC3-6400	100 MHz	400 MHz	800 MT/s	6,4 GB/s	12,8 GB/s
DDR3 1066	PC3-8500	133 MHz	533 MHz	1066 MT/s	8,5 GB/s	17,0 GB/s
DDR3 1333	PC3-10600	166 MHz	667 MHz	1333 MT/s	10,6 GB/s	21,2 GB/s
DDR3 1600	PC3-12800	200 MHz	800 MHz	1600 MT/s	12,8 GB/s	25,6 GB/s

Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

Memoria CACHE - $



- Contiene i dati di utilizzo più ricorrente in modo che il processore non li debba cercare nelle aree della memoria centrale.
- Memoria ad accesso molto veloce
- Serve a compensare la differenza di velocità di accesso e di trasferimento dei dati tra la CPU e la memoria RAM

Osservazione:
Il buon funzionamento della cache deriva dalla capacità del sistema nel mantenere in essa le informazioni che saranno necessarie.

Memoria CACHE - $

Principio di Località (spaziale): se in un certo istante viene referenziato un indirizzo di memoria è altamente probabile che in istanti immediatamente successivi possano venire referenziati indirizzi vicini.

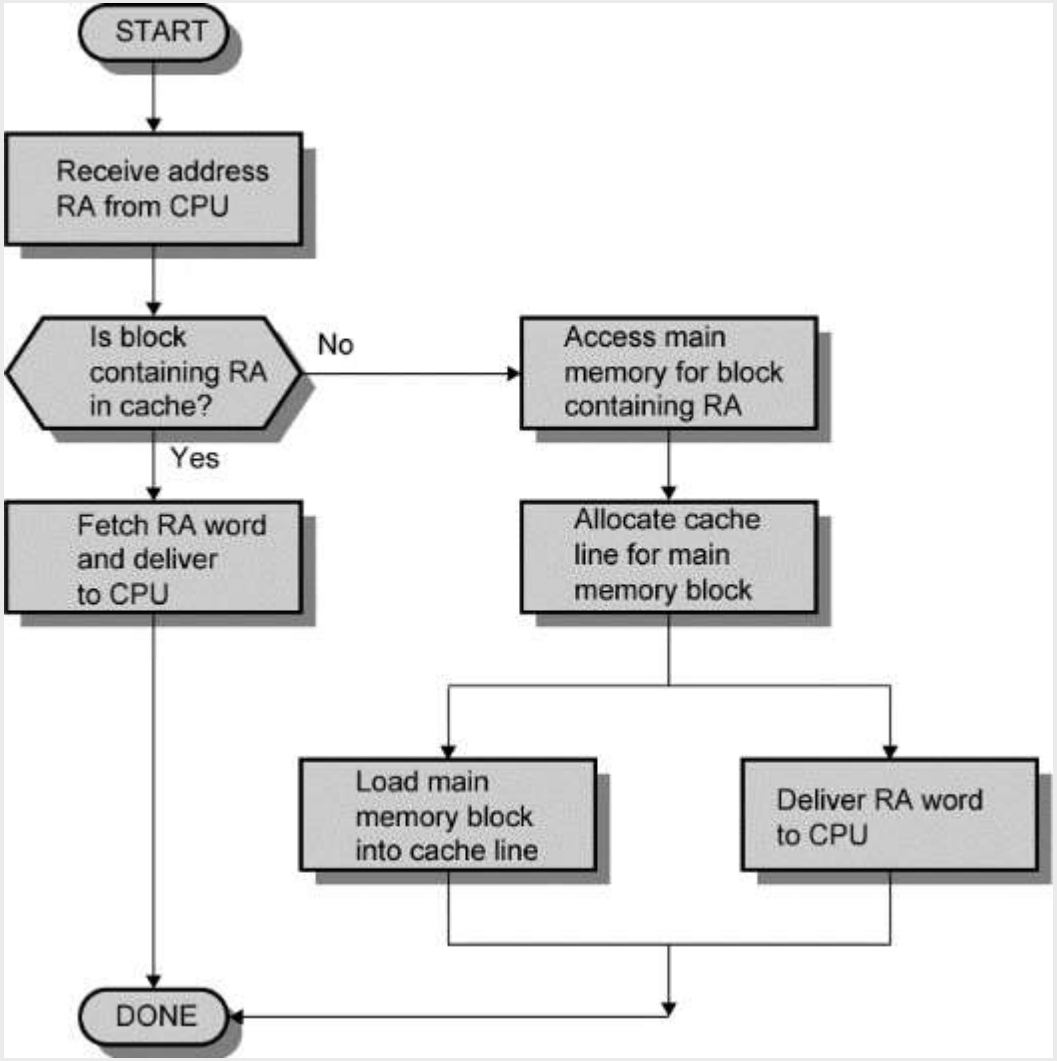
Principio di Località (temporale): se in un certo istante viene referenziato un indirizzo di memoria è altamente probabile che in istanti immediatamente successivi lo stesso indirizzo possa essere nuovamente referenziato.

I riferimenti alla memoria fatti in un breve intervallo di tempo tendono ad utilizzare solo una piccola parte della memoria totale

Motivazioni per la CACHE - $

- In tutti i cicli di istruzione il processore accede alla memoria principale almeno una volta (prelievo dell'istruzione IF)
- La velocità del processore è limitata dalla velocità della memoria
- La velocità della memoria centrale (per motivi tecnici e/o economici) è minore di quella del processore
- Sfruttando il principio di località si inserisce una memoria piccola e veloce (la cache) fra processore e memoria centrale
- Obiettivo
    - fornire una memoria con velocità prossima a quella del processore
    - disporre di una quantità di spazio sufficiente per non rallentare il processore
    - contenere i costi delle memorie del sistema
- Contenuto:
    - una copia di una porzione della memoria centrale

Lettura da CACHE - $



Memoria CACHE - $

Se durante un piccolo intervallo di tempo una parola è letta o scritta k volte, il calcolatore dovrà effettuare:

- 1 riferimento alla memoria (più lenta) (solo la prima volta) (+ 1 accesso, non riuscito, alla cache)
- (k-1) riferimenti alla cache (più veloce)

Posto

c=tempo di accesso alla cache

m=tempo di accesso alla memoria

Si ottiene $t_{memory}=km$; $t_{cache}=(c+m)+(k-1)c$.

Il vantaggio nell'uso della cache è quantificabile in:

$$\frac{t_{cache}}{t_{memory}} = \frac{(c+m)+(k-1)c}{km} = \frac{c+m}{km} + \frac{(k-1)}{k}\frac{c}{m}$$

Memoria CACHE - $

Nel caso più generale, posto:

h=*hit ratio* (frequenza di successi nell'accesso alla cache)

((1-h)=*miss ratio*)

Si ottiene che il tempo medio di accesso è pari a:

$$=h\cdot c+(1-h)\cdot(c+m)=$$

$$=hc+c-hc+(1-h)m=$$

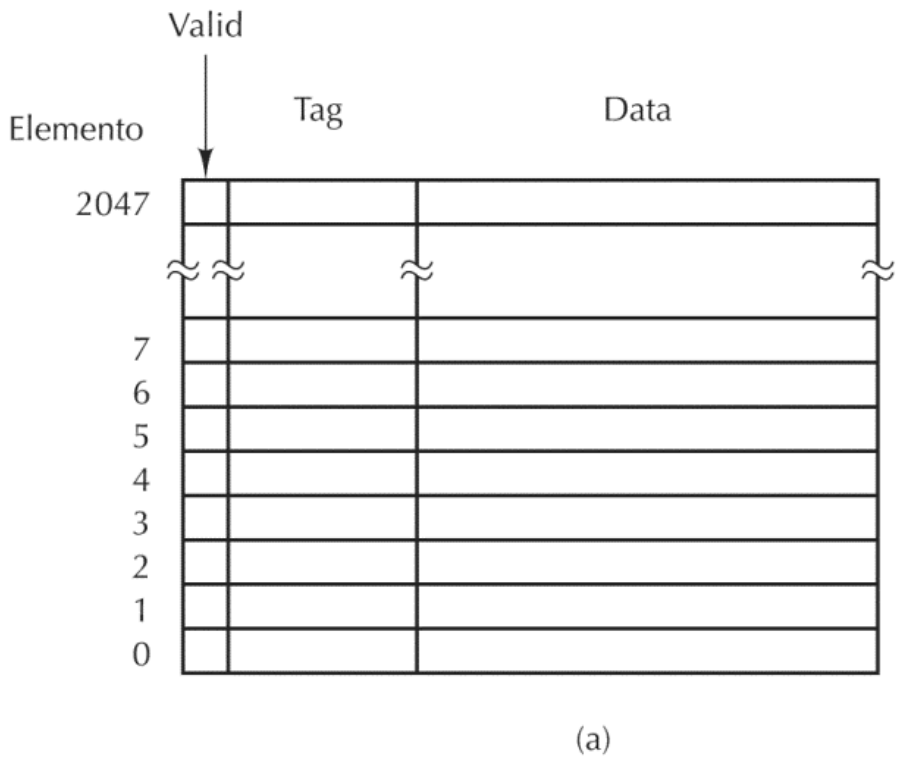
$$=c+(1-h)m$$

Gestione della CACHE - $

- Posizionamento di un blocco in cache:
  - Indirizzamento diretto
  - Fully associative
  - Set associative
- Algoritmo di rimpiazzo:
  - sceglie quale blocco rimpiazzare
  - sarebbe preferibile rimpiazzare il blocco meno necessario nel prossimo futuro (Politica di Belady o ottima)
  - Politica LRU (Last Recently Used)
- Politica di scrittura:
  - qualora i contenuti del blocco sono stati modificati bisogna riscriverlo in memoria centrale prima di rimpiazzarlo
  - sono possibili diverse politiche:
    - scrivere ogni volta che il blocco è aggiornato (write through)
      - » politica molto onerosa, ma mantiene la memoria costantemente aggiornata
    - scrivere quando il blocco è rimpiazzato (write back)
      - » ottimizza le scritture in memoria, ma può lasciare la memoria centrale non aggiornata per lunghi periodi

Posizionamento di un blocco in cache

- **Indirizzamento diretto**: un blocco può essere messo in un solo punto della cache. L'indirizzo del blocco è ottenuto mediante l'operazione (indirizzo di blocco) % (numero di blocchi nella cache).



Valid: indica se il dato è valido

Tag: fa riferimento alla linea di memoria da cui arrivano i dati

Data: contiene il dato di memoria

Posizionamento di un blocco in cache

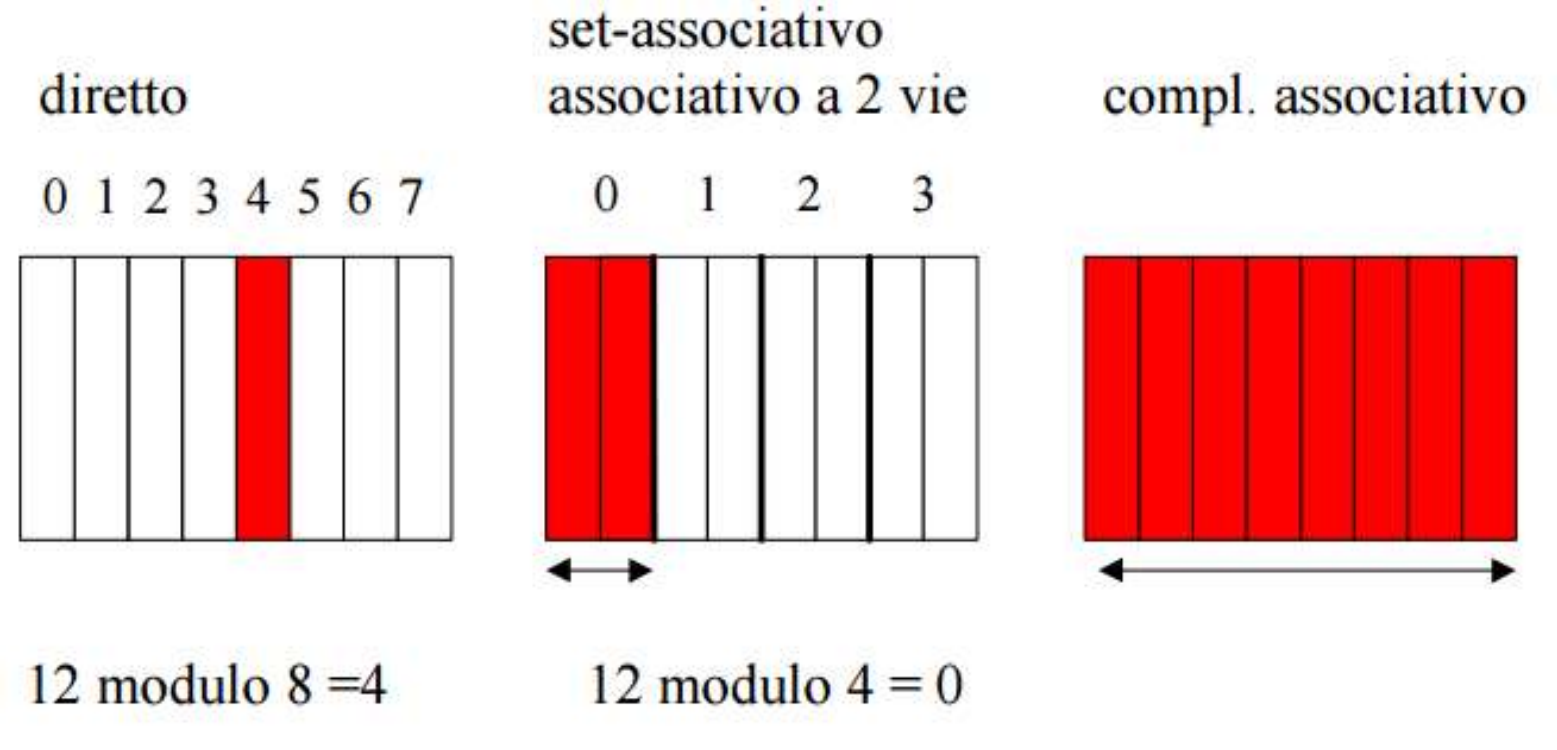
- **Fully Associative (completamente associativa):** un blocco può essere posto ovunque nella cache.
- PRO: molto facile da realizzare
- CONTRO: se i blocchi possono essere messi ovunque, avrò difficoltà nell'andare a ritrovarli o dovrò introdurre ulteriori meccanismi che mi ridurrebbero la facilità di implementazione

Posizionamento di un blocco in cache

- **Set Associative:** un blocco può essere posto in un insieme ridotto di posizioni nella cache
  - Un insieme (set) è un gruppo di due o più blocchi della cache
  - Un blocco viene prima messo in corrispondenza di un insieme e poi può essere messo in qualsiasi posizione dell'insieme
  - L'insieme viene scelto con la regola del modulo: (indirizzo blocco) modulo (numero di insiemi della cache)
  - Se ci sono n blocchi in un insieme, il posizionamento viene definito come *set-associativo a n vie*

Posizionamento di un blocco in Cache: un esempio

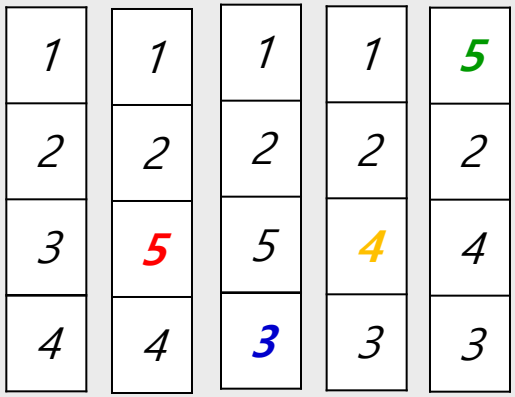
Esempio. Consideriamo il blocco di indirizzo 12 ed i tre tipi di indirizzamento in una memoria che può contenere 8 blocchi.



Algoritmi di replacement: Least Recently Used (LRU)

- Si sostituiscono i blocchi meno recentemente utilizzati
  - Ipotesi: il blocco meno utilizzato recentemente è quello che ha la minore probabilità di essere referenziato in futuro. Si approssima il futuro prossimo al passato recente

- Reference string: 1, 2, 3, 4, 1, 2, **5**, 1, 2, **3**, **4**, **5**



8 miss

Algoritmi di replacement: Random

- I blocchi candidati per la sostituzione sono scelti in modo casuale
  - Facilità di realizzazione rispetto a LRU
  - È possibile verificare empiricamente che al crescere della dimensione della cache lo scarto di performance in termini di miss rate è minimo o nullo

Associatività	**A 2 vie**		**A 4 vie**		**A 8 vie**	
Dimensione	LRU	Casuale	LRU	Casuale	LRU	Casuale
16KB	5,18%	5,69%	4,67%	5,29%	4,39%	4,96%
64KB	1,88%	2,01%	1,54%	1,66%	1,39%	1,53%
256KB	1,15%	1,17%	1,13%	1,13%	1,12%	1,12%

Fonte: Hennessy-Patterson, i risultati si riferiscono ad una macchina VAX con cache a blocchi di 16bit. I dati osservati si riferiscono sia a programmi utente che OS

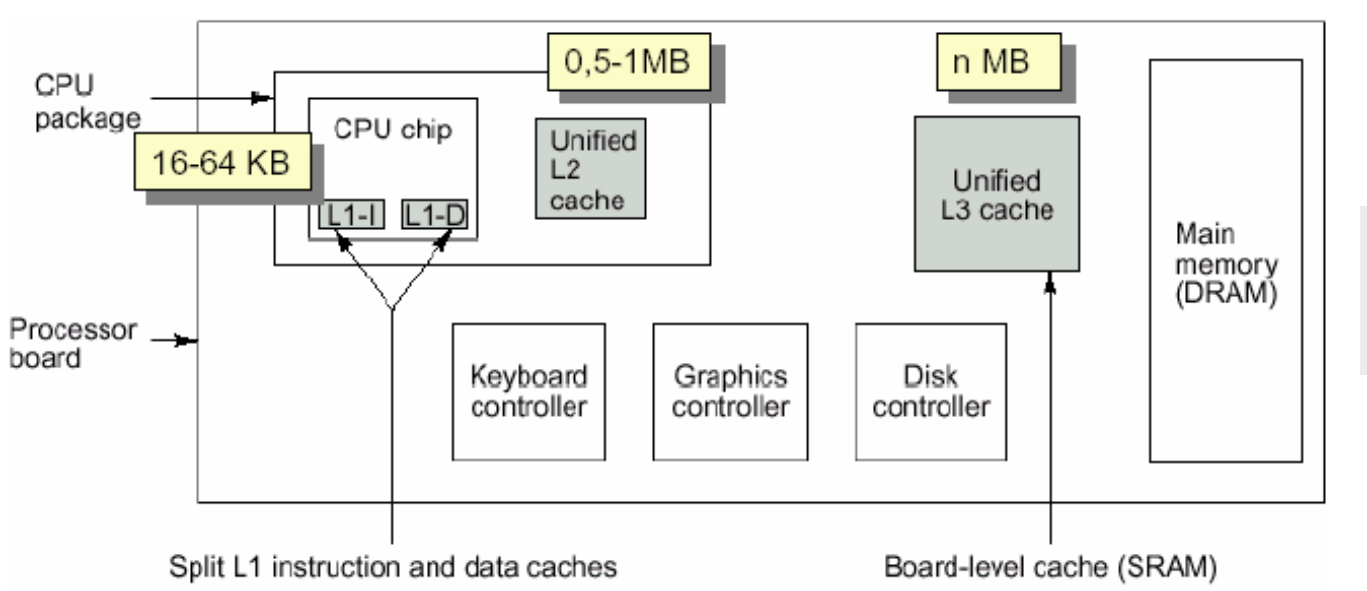
WRITE POLICY

PROBLEMA: scrittura da CPU in $, quando aggiornare la RAM?

POLITICHE:

- WRITE THROUGH
  - All writes go to main memory as well as cache
  - Multiple CPUs can monitor main memory traffic to keep local (to CPU) cache up to date
  - Lots of traffic
  - Slows down writes
- WRITE BACK
  - Updates initially made in cache only
  - Update bit for cache slot is set when update occurs
  - If block is to be replaced, write to main memory only if update bit is set
  - Other caches get out of sync
  - I/O must access main memory through cache
  - N.B. 15% of memory references are writes

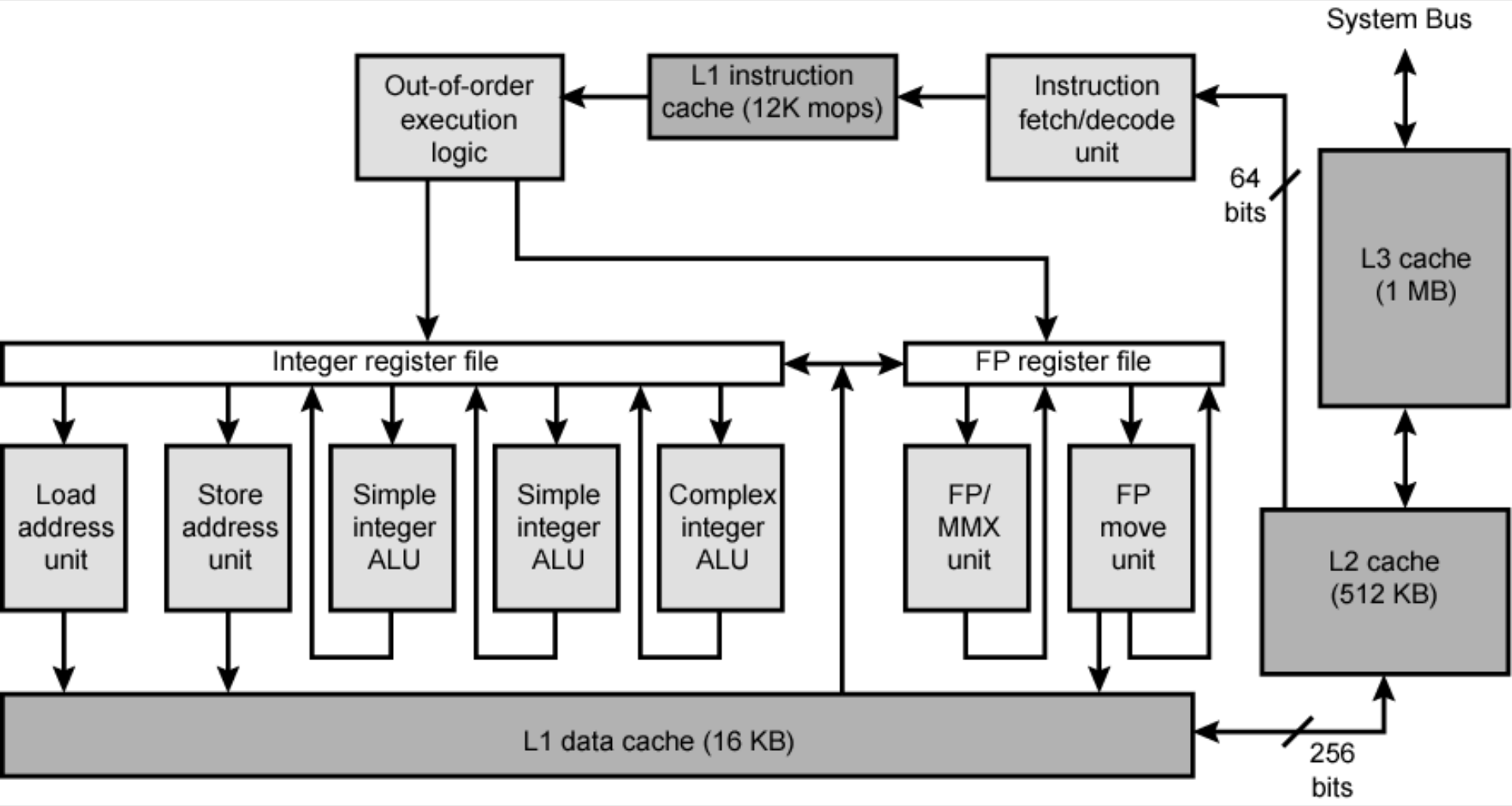
Memoria CACHE - $



L1	2 x 64 KB
L2	1-2 MB
L3	4- MB

Dapprima la CPU cerca il dato nella cache di livello 1; se avviene un **hit**, il processore procede ad alta velocità. Se si verifica un **miss**, allora viene controllata la $ di livello 2 e così via, fino ad accedere alla memoria principale.

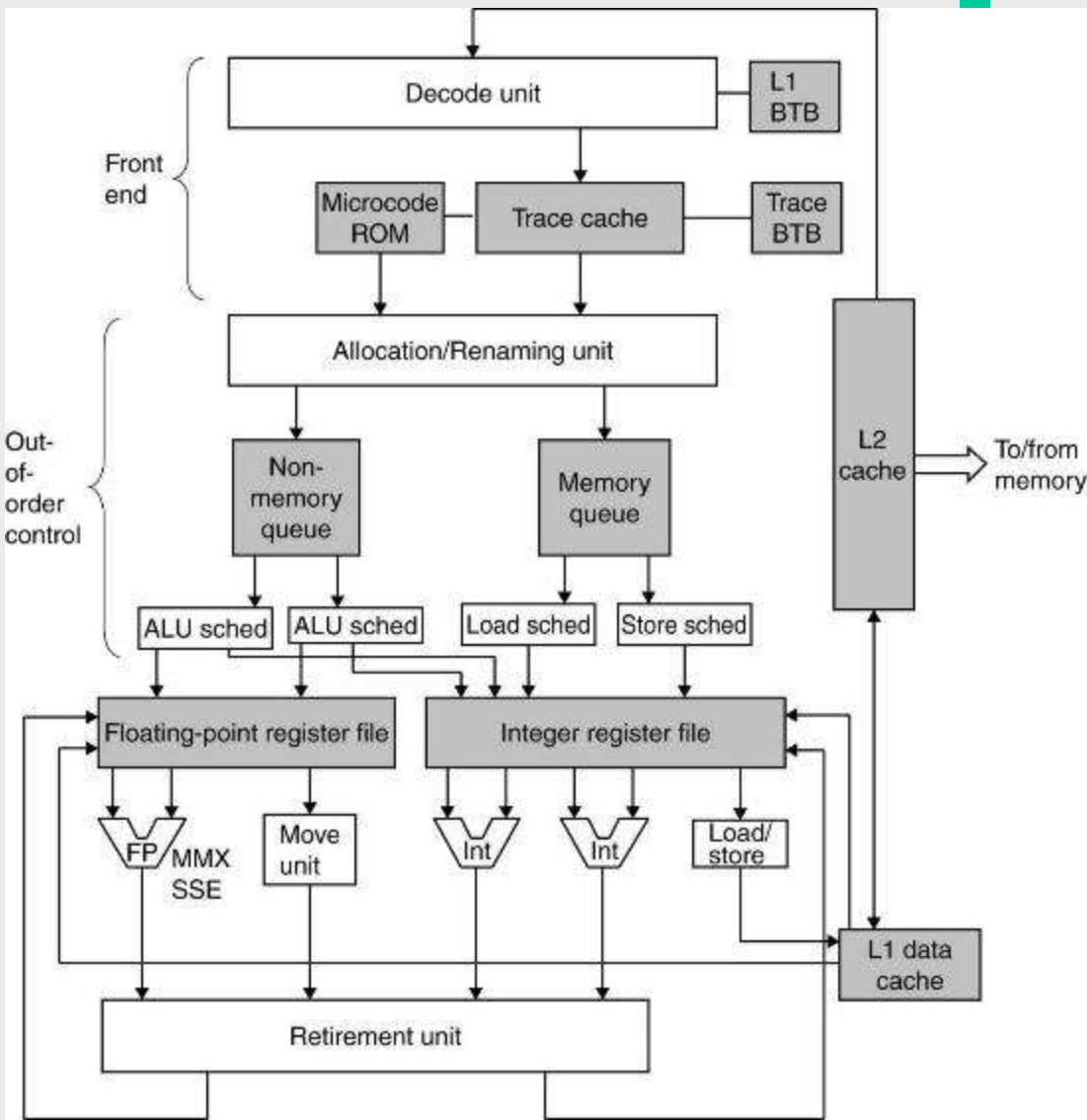
Memoria CACHE - $
Pentium 4



Memoria CACHE - $
Intel Core i7

Ogni core contiene:

- *L2, L1 e la logica per accedere a L3 condivisa*
- *$ di pre-fetch che cercano di caricare istruzioni prima che siano referenziate*
- *Front-end:*
  - *Prelievo, decodifica in formato RISC e salvataggio in $ delle istruzioni*
  - *Le istruzioni prelevate da $ L1 vengono passate ai decodificatori che determinano le operazioni da svolgere nella pipeline*
- *Unità di controllo fuori sequenza:*
  - *Tiene traccia delle micro-operazioni, se le risorse sono disponibili viene inserita in una delle due code altrimenti ritardata*
  - *Il fuori sequenza è dovuto alla pipeline*



Interruzioni

- Consentono di interrompere l'elaborazione normale del processore
- La funzione principale è migliorare l'efficienza della elaborazione (dispositivi periferici molto più lenti del processore)

  Es. stampa su stampante
- Permettono al modulo di I/O, al termine del comando di I/O, di segnalare l'evento al processore

Tipologie di interruzioni:

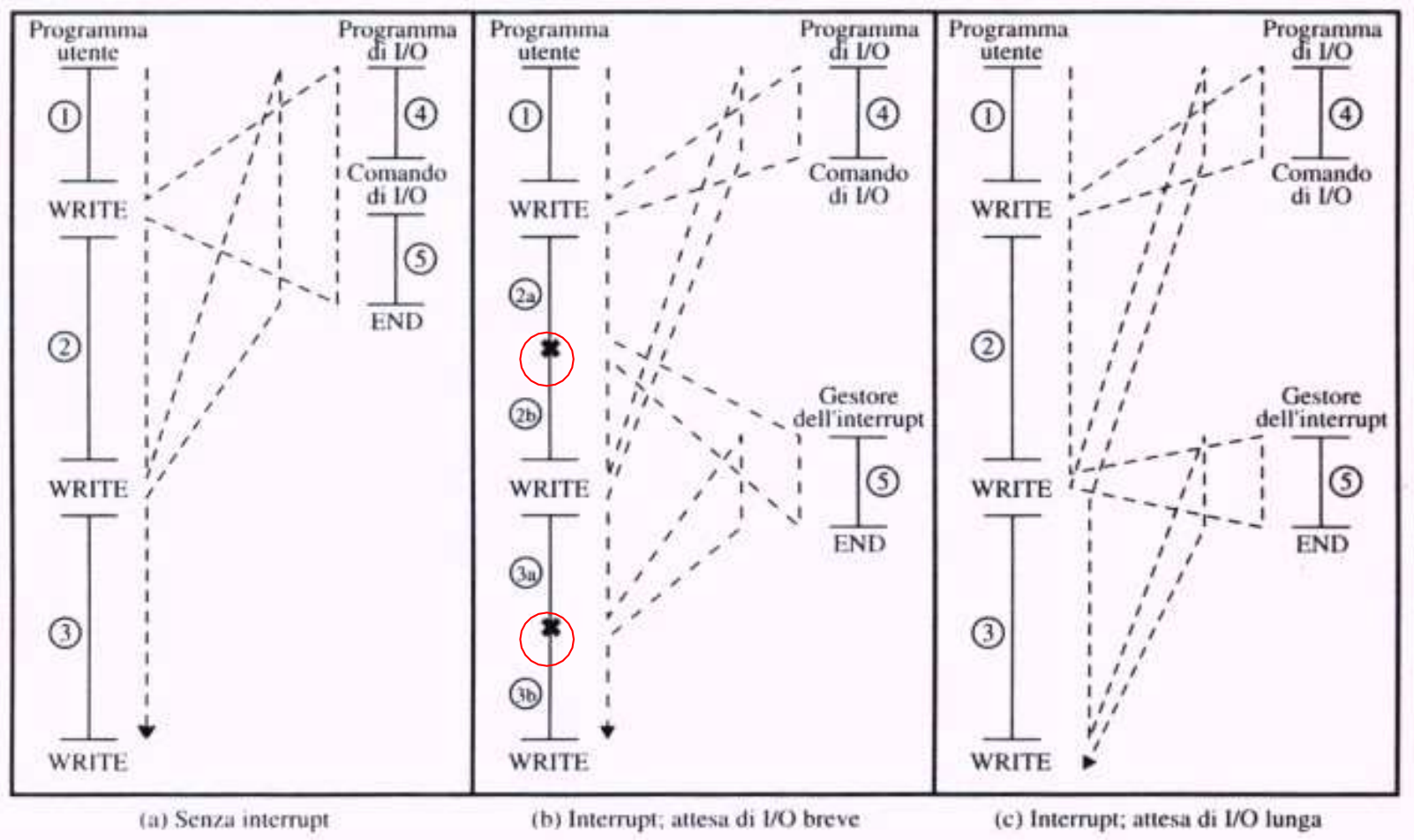
- Interrupt: generate da un dispositivo hardware
- Trap: generate da un programma in esecuzione

Linee controllo

Classi di interruzione:

- Programma
  - errore di esecuzione (overflow, divisione per zero, violazione spazio di memoria)
- Timer
  - operazioni pianificate (time slot)
- I/O
  - operazioni di I/O
- Errore hardware
  - problemi fisici (caduta di tensione)

Flusso di controllo con e senza interrupt



WRITE – chiamata di sistema
4 – Preparazione alle op. di I/O, es. copiatura dati da inviare alla stampante
5 – Completamento op. I/O, es. settaggio flag che indica l'esito della operazione

Gestione dell'interrupt

Il programma utente non contiene alcun comando speciale per la gestione della operazione di I/O

Il programma di gestione dell'interrupt è parte del SO:
-Determina il tipo di interruzione
-Chiama il programma che gestisce tale evento

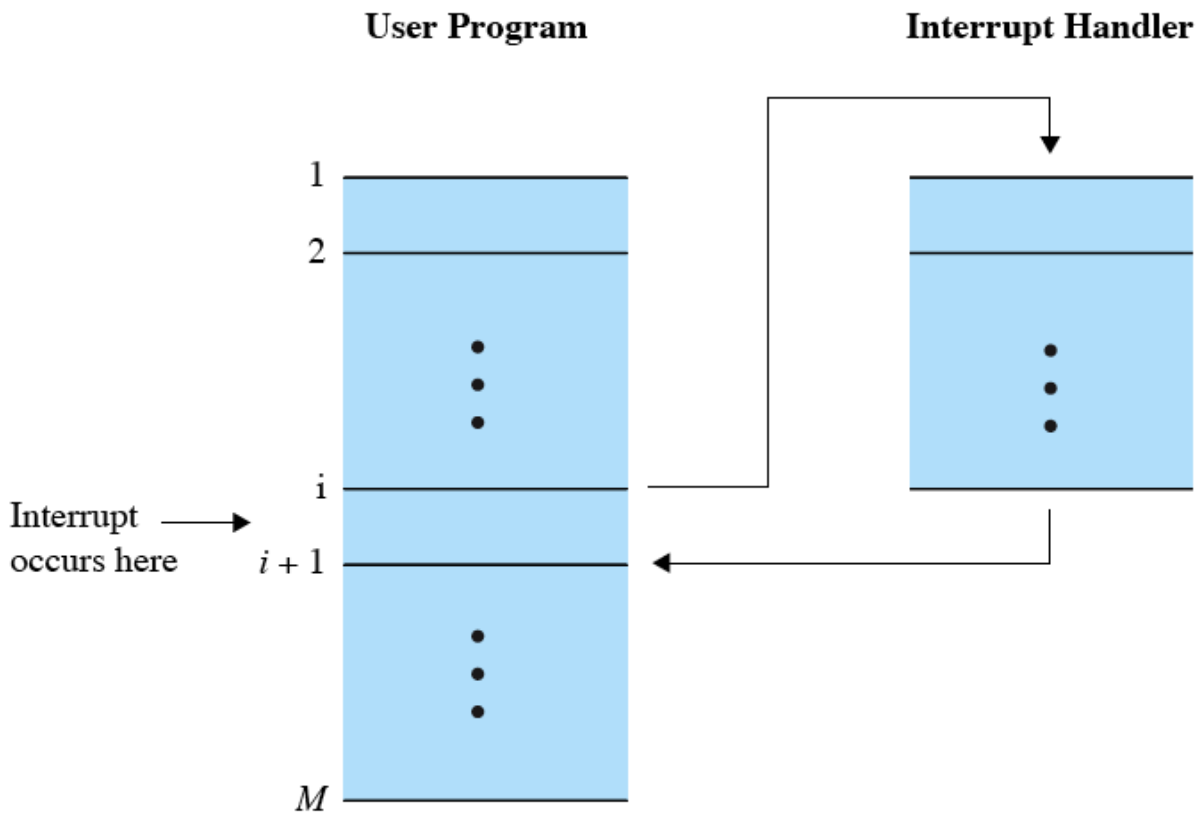
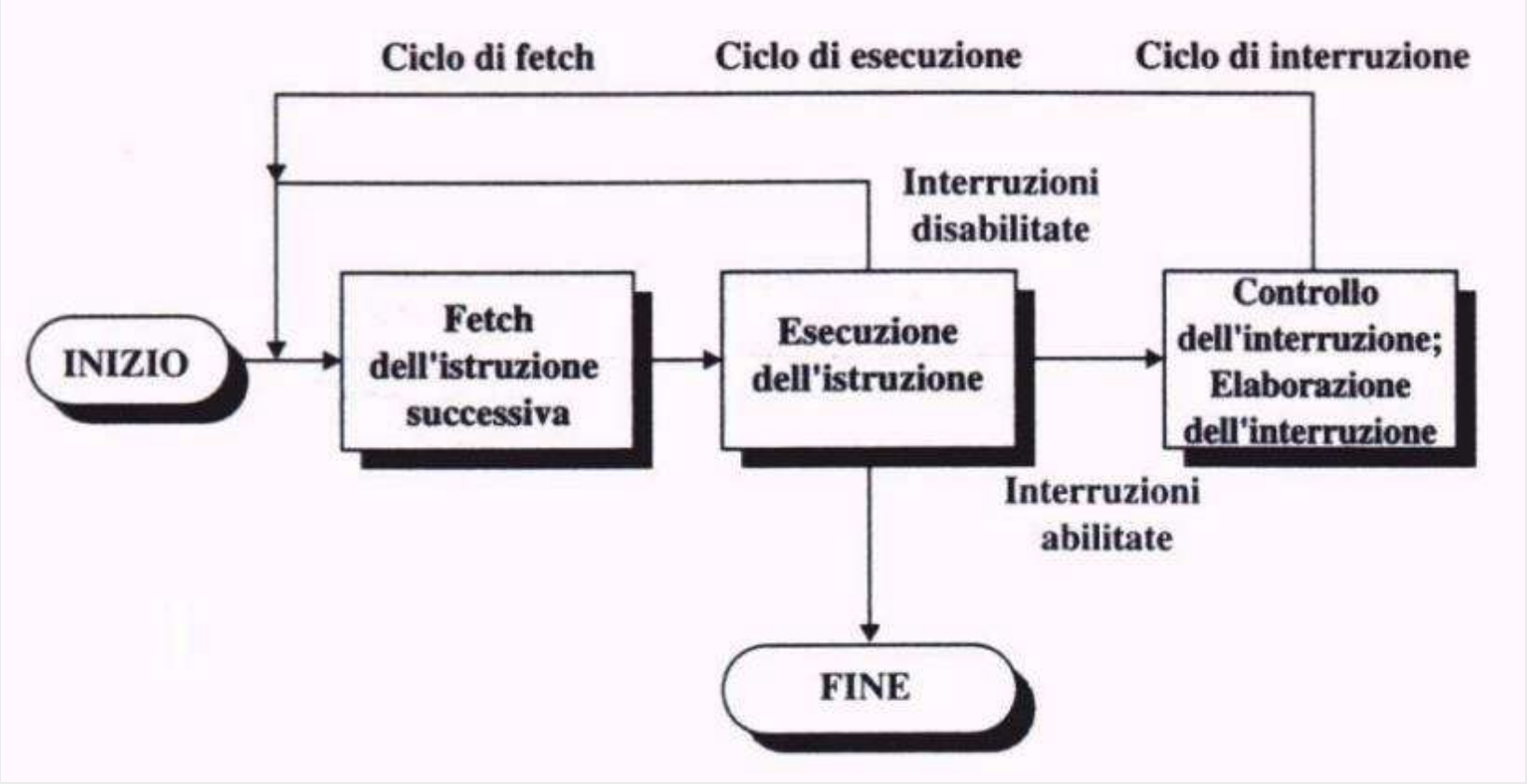


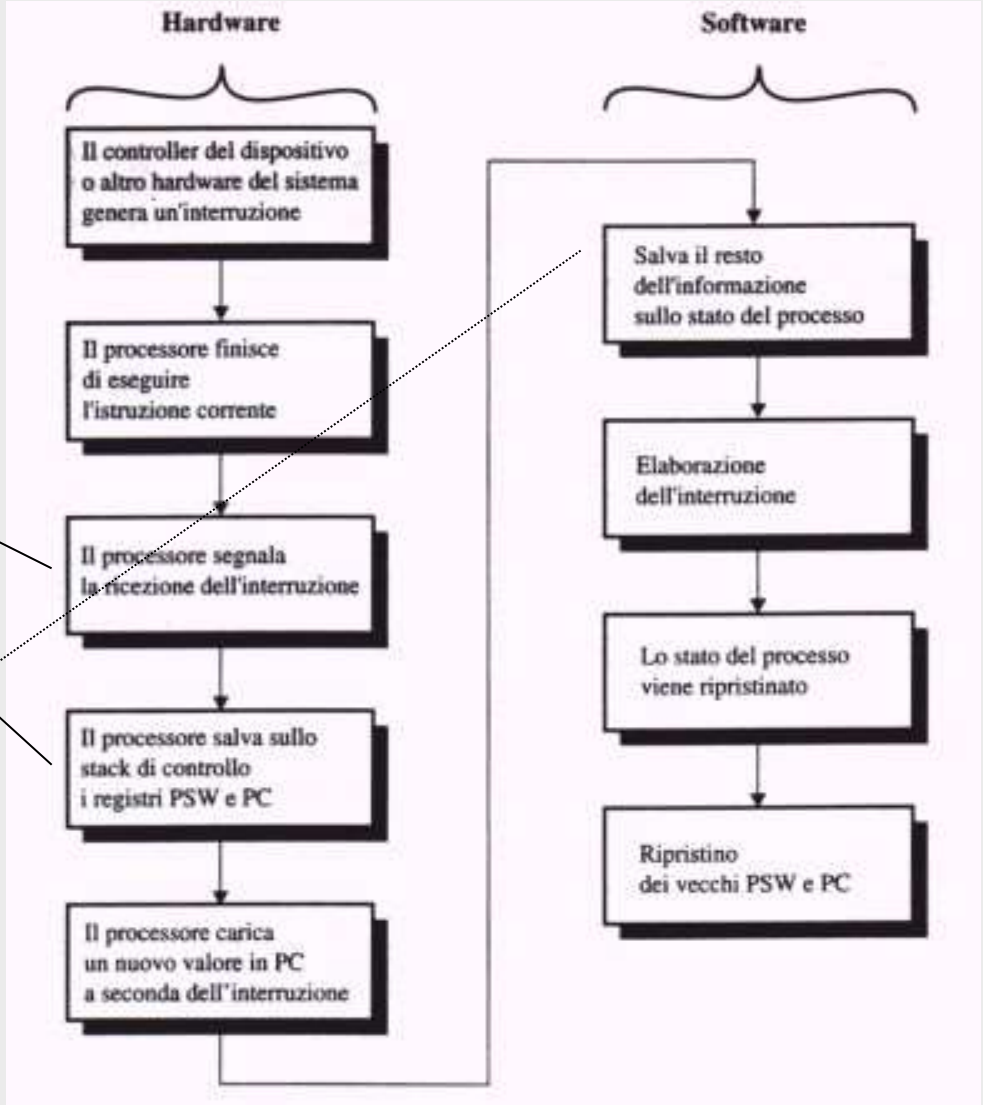
Figure 1.6 Transfer of Control via Interrupts

Ciclo di esecuzione con interruzione



NB: in questo schema il sistema controlla se ci sono interruzioni solo al termine della fase di exe

Elaborazione delle interruzioni



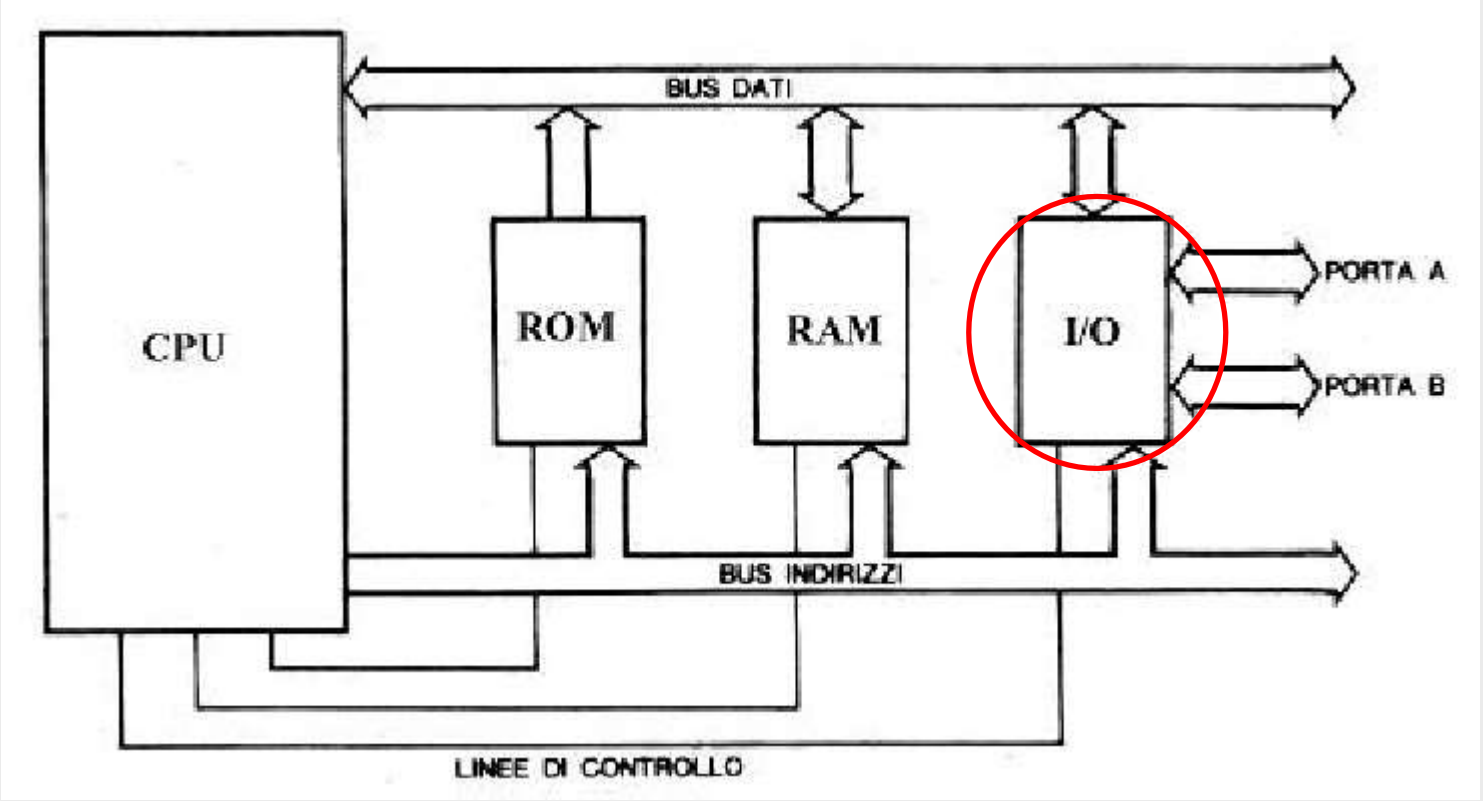
Il processore invia un ack al dispositivo

Dati salvati sullo stack:
-PC
-PSW
-Registri che potrebbero essere modificati dal programma di gestione dell'interrupt

Il programma che gestisce l'interrupt è parte del SO:

- Determina la natura dell'interruzione
- Chiama il modulo che la gestisce

Macchina di Von Neuman: Dispositivi di I/O



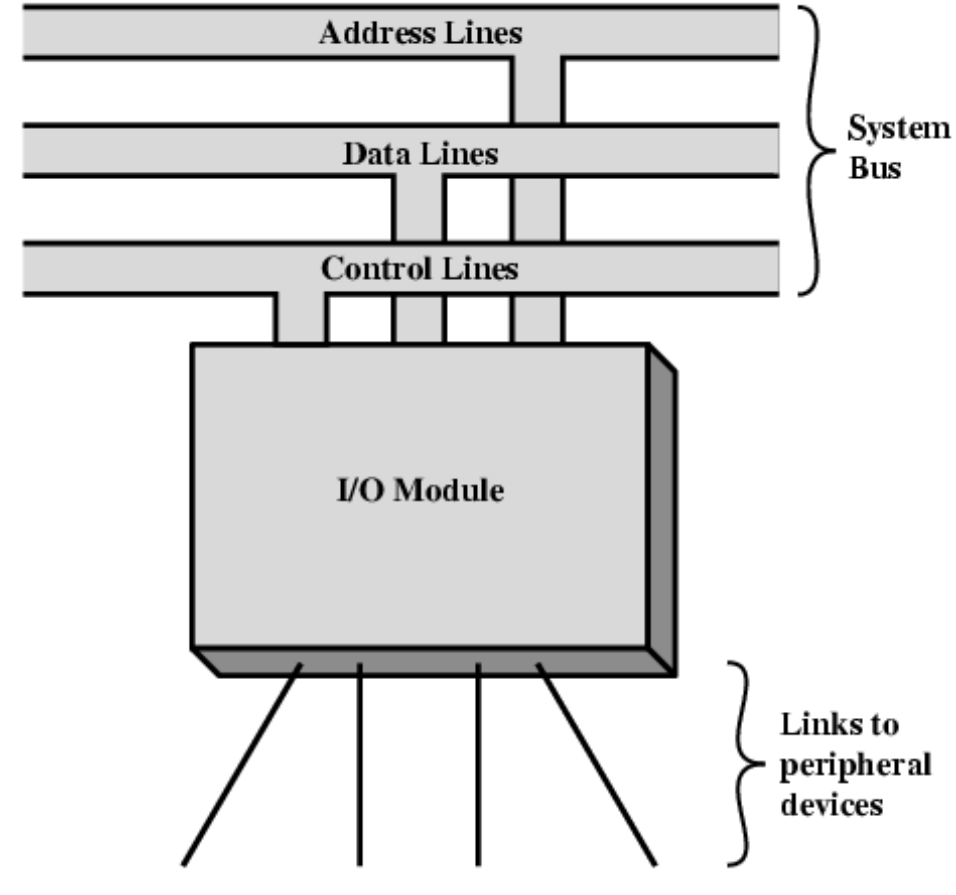
Dispositivi di I/O

Principali dispositivi di input:
- tastiera
- mouse

Principali dispositivi di output:
- monitor
- stampante
- casse acustiche

Collegamenti tramite:
1. porte di comunicazione (tastiera e mouse),
2. schede montate sulla motherboard.
   Monitor - scheda video, Casse - scheda audio.
   NB: queste due schede possono anche essere integrate nella scheda madre.

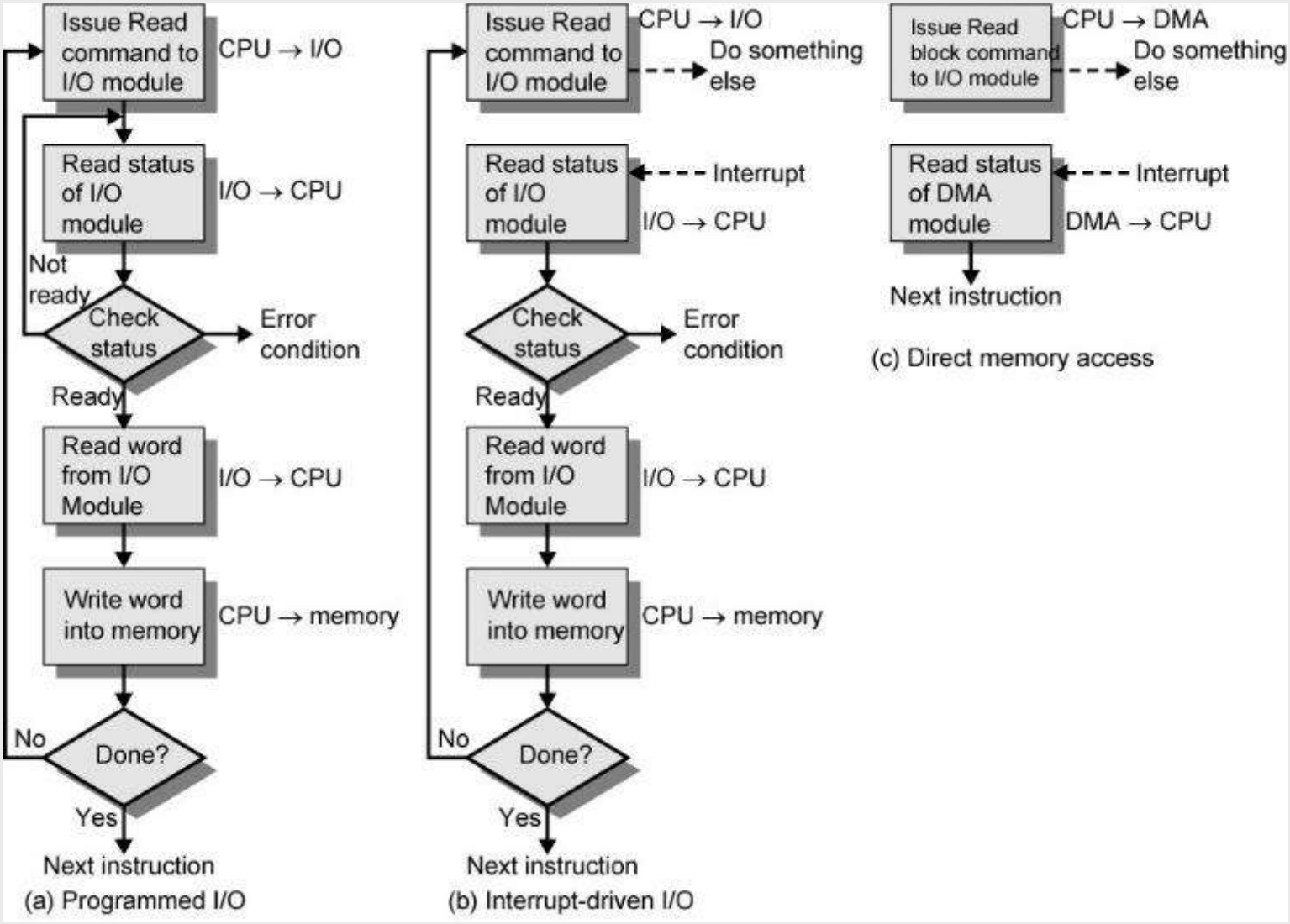
Modello di un generico modulo di I/O



Tecniche di I/O

- Programmato
- Interrupt driven
- Direct Memory Access (DMA)

Tecniche di I/O



I/O Programmato

- CPU ha controllo diretto sul dispositivo di I/O
  - Sensing status
  - Read/write commands
  - Transferring data
- CPU attende che il modulo di I/O completi l'operazione
- Wastes CPU time

I/O Interrupt driven

- Overcomes CPU waiting
- No repeated CPU checking of device
- I/O module interrupts when ready

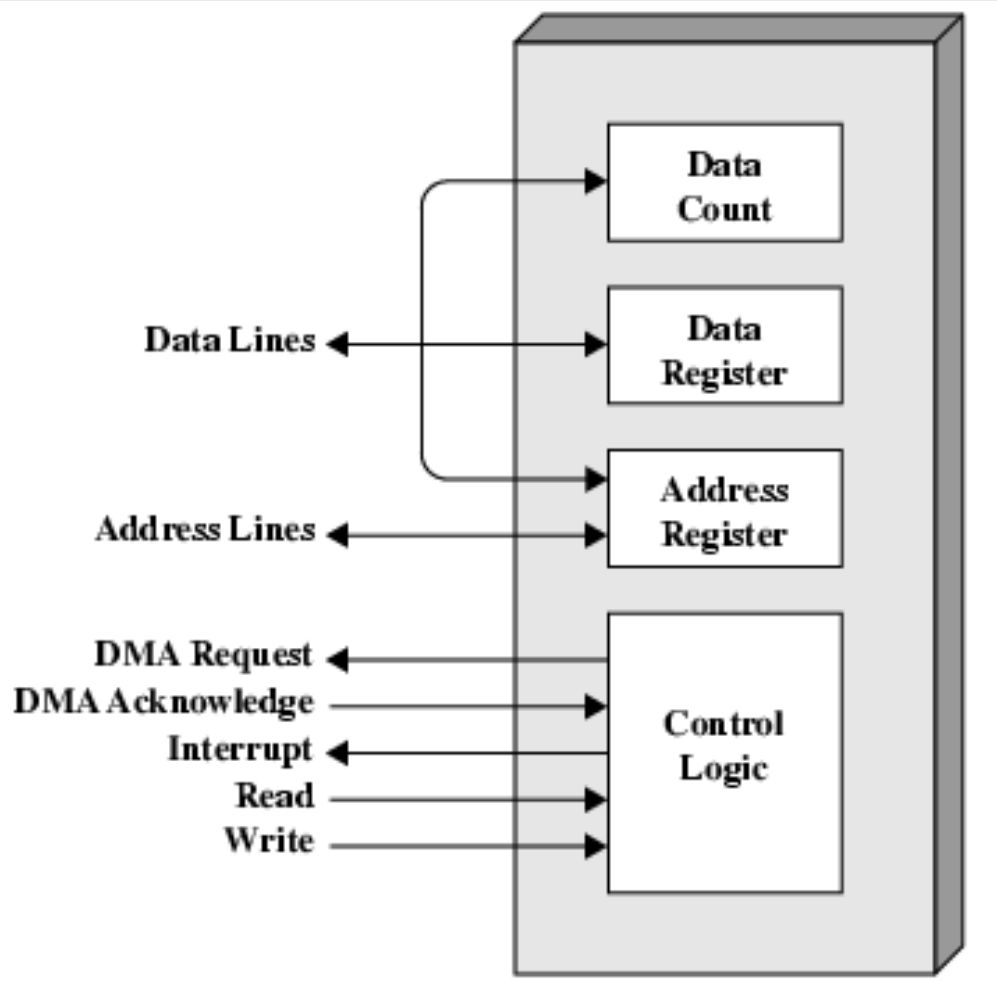
DMA

- I/O Interrupt driven e programmato richiedono l'intervento attivo della CPU
- DMA is the answer

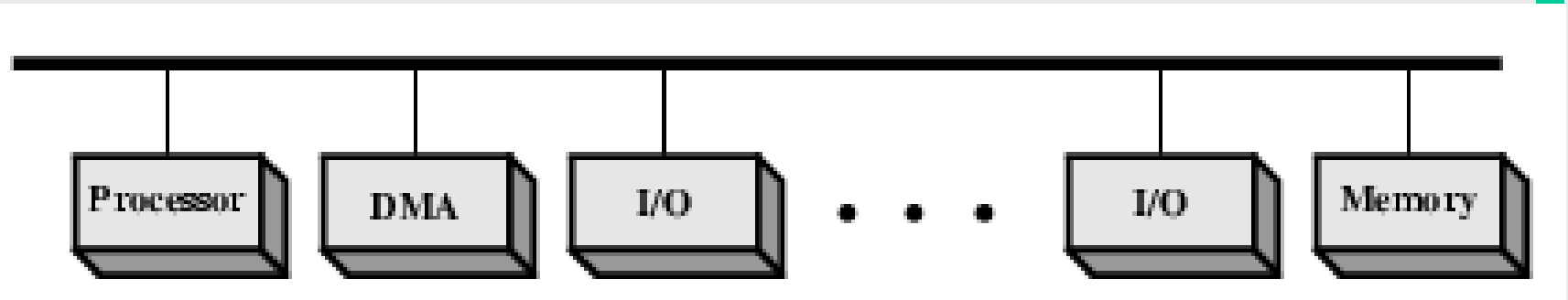
- Il DMA è un modulo aggiuntivo (hardware) montato sul bus

- OPERAZIONI SVOLTE DAL DMA
  - CPU comunica al controller di DMA l'oiperazione da svolgere
    - Read/Write
    - Device address
    - Starting address of memory block for data
    - Amount of data to be transferred
  - CPU svolge altri lavori
  - Il controller DMA si occupa del trasferimento
  - Il controller DMA invia un interrupt quando ha concluso il trasferimento

DMA Module Diagram

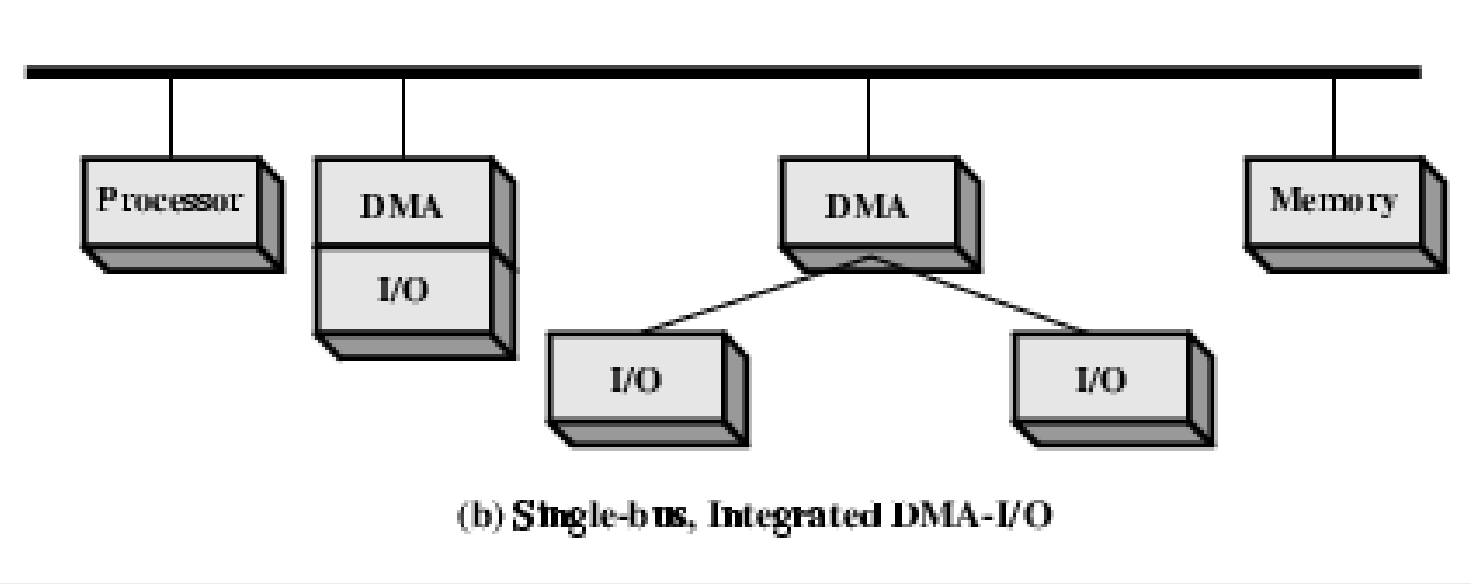


Configurazioni DMA



- Single Bus,
- Ogni trasgerimento utilizza il bus due volte:
  - I/O to DMA then DMA to memory

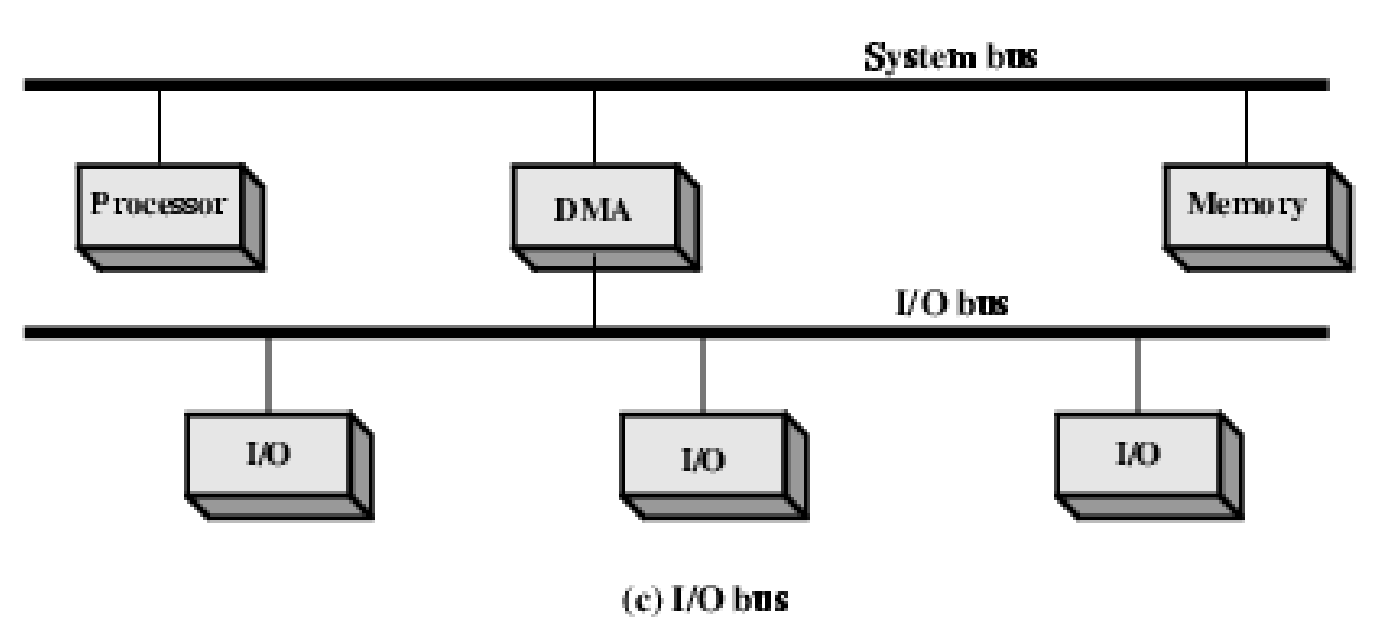
Configurazioni DMA



(b) Single-bus, Integrated DMA-I/O

- Single Bus,
- Controller may support >1 device
- Ogni trasferimento utilizza il bus una volta
  - DMA to memory

Configurazioni DMA



(c) I/O bus

- Bus dedicato all'I/O
- Ogni trasferimento impegna il bus una sola volta
  - DMA to memory

HARD DISK

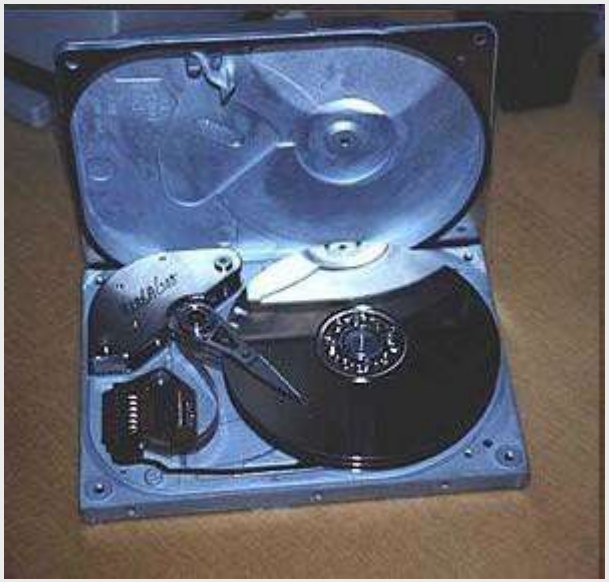
Memoria permanente.

Costituito da uno o più dischi in rapida rotazione, realizzati in alluminio o vetro, rivestiti di materiale ferromagnetico e da due testine per ogni disco (una per lato).

Dimensioni: 2.5 (Laptop), 3.5 Pollici (Desktop)

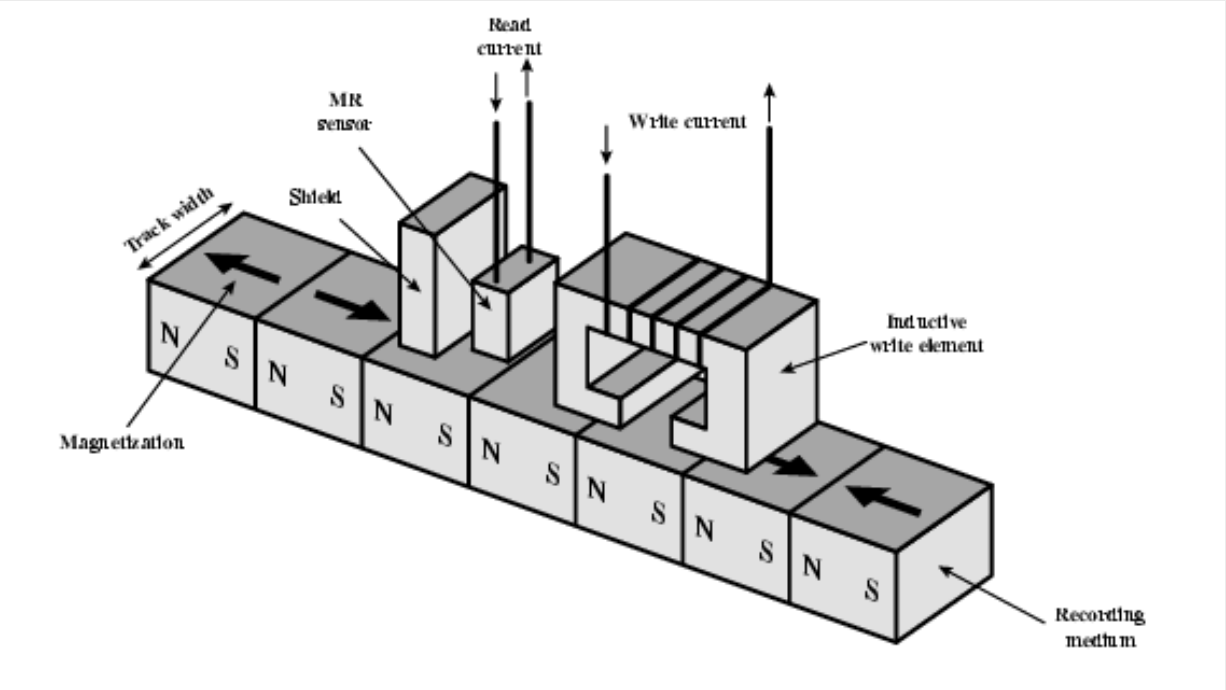
Prestazioni:

- Capacità (GB)
- Velocità di rotazione (rpm): 5.200, 5.400, 7.200, 10.000 e 15.000



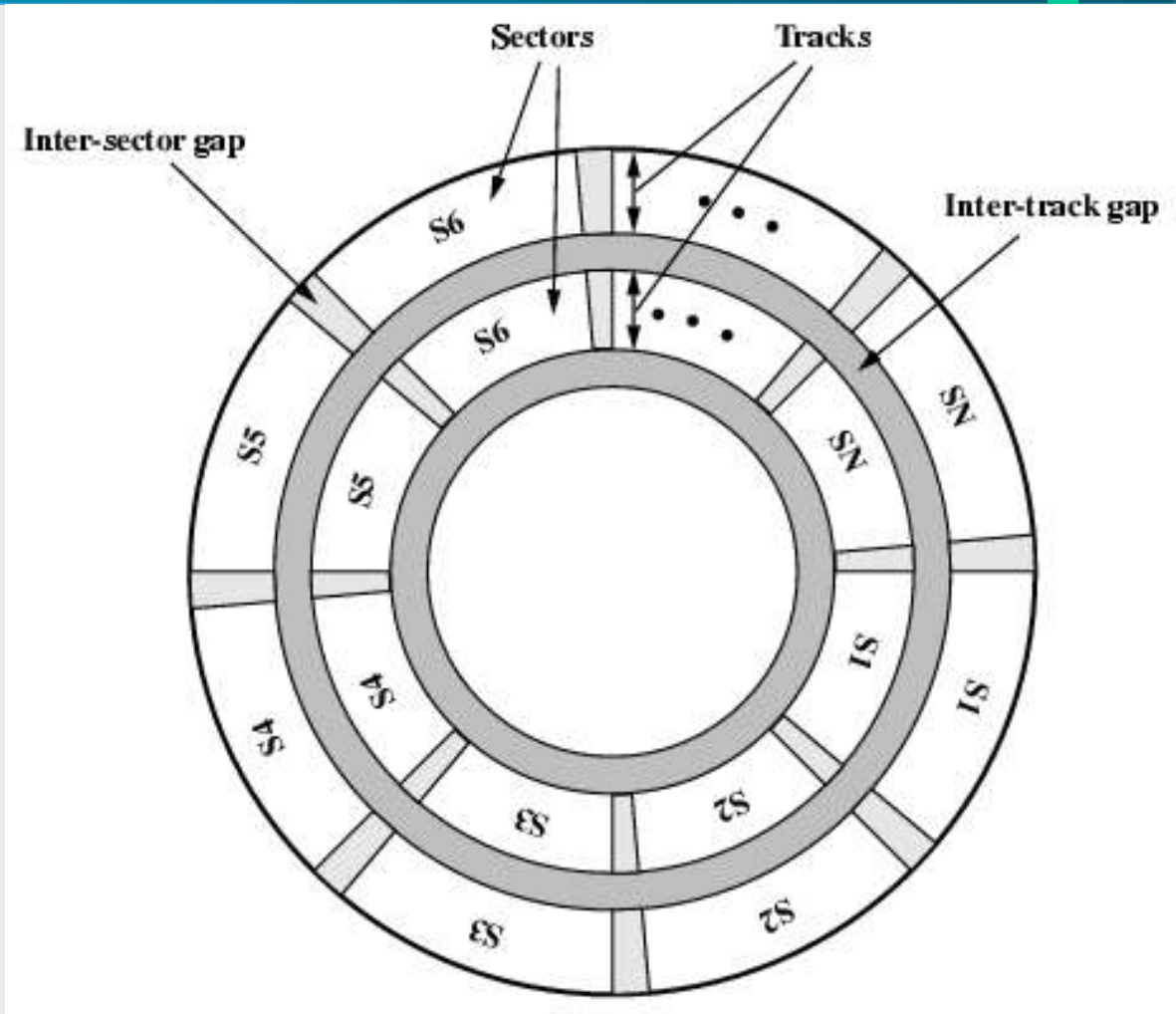
Meccanismi di scrittura e lettura

- Durante le operazioni di lettura e scrittura la testina è fissa e I piatti ruotano
- Scrittura
  - La corrente attraverso i fili produce un campo magnetico
  - L'informazione magnetica viene memorizzata sulla superficie sottostante
- Lettura
  - Rilevazione del campo magnetico: il campo magnetico in movimento produce nel filo una corrente elettrica



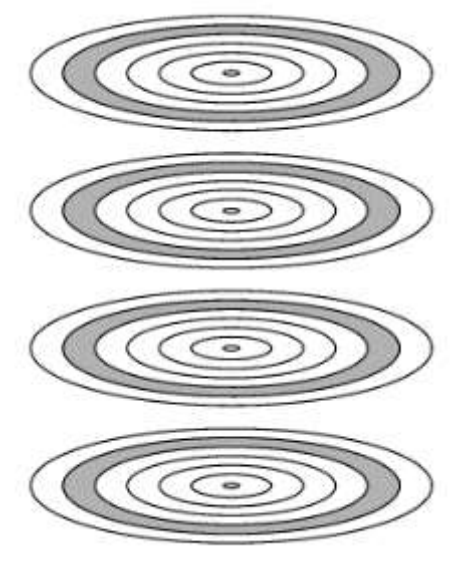
Organizzazione dei dati

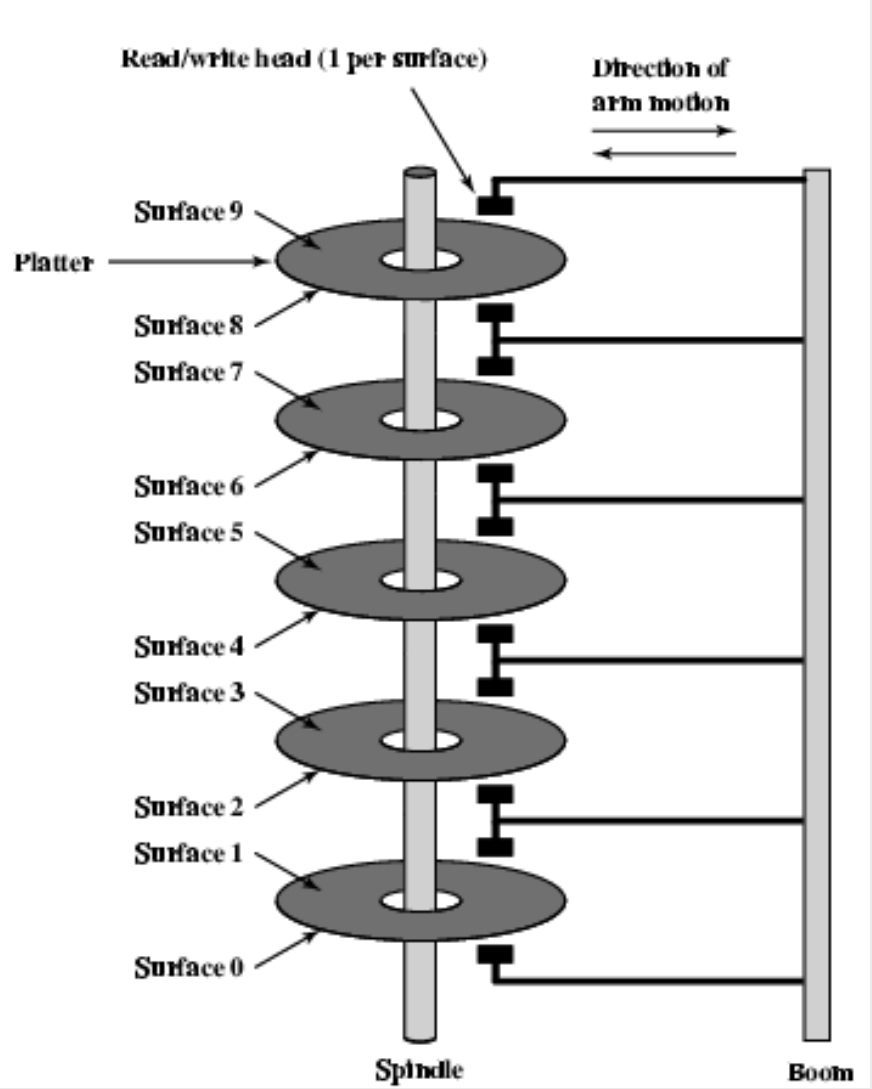
- Anelli concentrici o tracce
  - Gaps tra le traccie
- Le tracce sono divise in settori
- L'unità minima scrivibile/leggibile è il settore



HARD DISK – piatti multipli

- Una testina per lato (=due testine per piatto)
- Le testine sono tutte allineate
- Le tracce allineate su più piatti individuano un cilindro
- I dati sono distribuiti tra i cilindri:
  - Riduzione del movimento delle testine
  - Aumento della velocità (transfer rate)



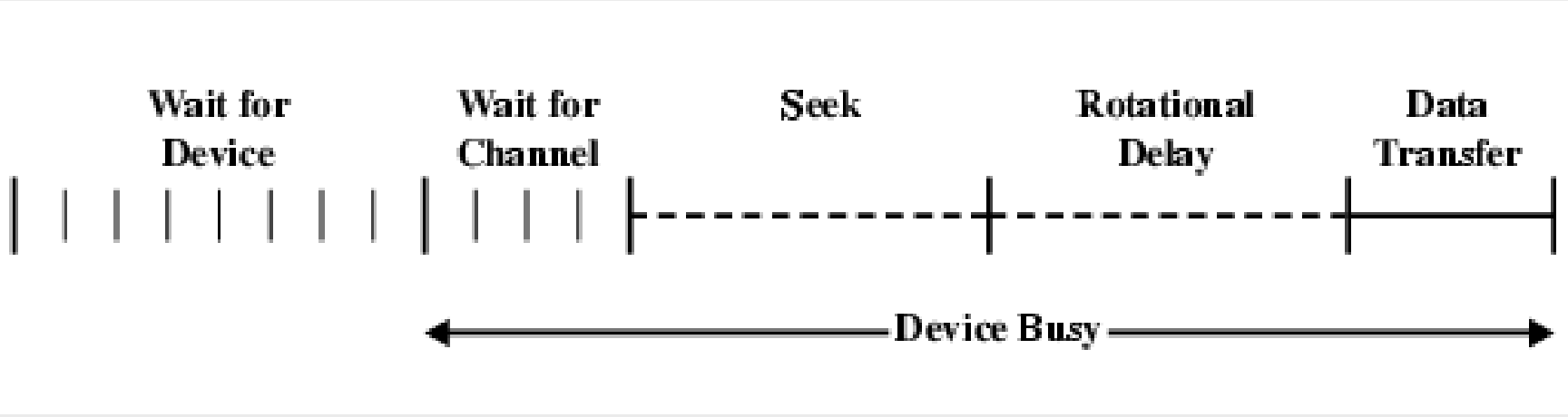


Velocità di accesso all' HARD DISK

Seek time: tempo necessario alla testina per posizionarsi sulla traccia di interesse.
Rotational delay: tempo necessario affinchè la testina si posizioni sul settore di interesse.

Transfer time: tempo necessario al trasferimento dei dati

Access time: tempo che intercorre tra la richiesta di accesso ai dati e l'istante in cui i dati sono disponibili.



Domande:

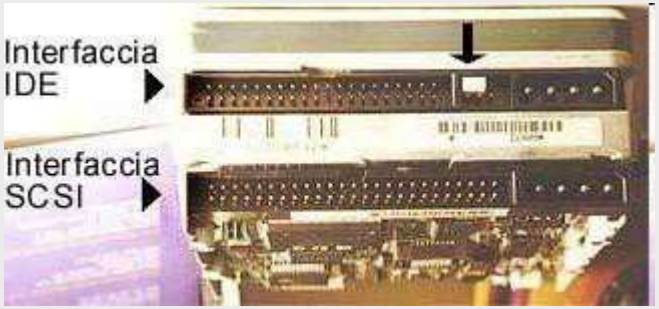
1. Su quale parametro influisce la disposizione dei dati secondo cilindri?
2. Su quale parametro influisce la velocità di rotazione del disco?

HARD DISK

Si differenziano sulla base dell'interfaccia di trasferimento dei dati (vale anche per tutti i dispositivi periferici quali CD, DVD, etc.):

- EIDE (Enhanced Integerated Drive Electronics) o PATA (Parallel Advanced Technology Attachment):
  - drive (apparato hardware) per la connessione di dispositivi
  - integrati nella scheda madre
  - generalmente collocati sui personal computer di medie/basse prestazioni
  - Velocità di trasferimento: 16.6 MB/s – 33MB/s (ATA-3 o ULTRA ATA)
  - le operazioni di trasferimento dati impegnando la CPU
  - non può operare sui due canali contemporaneamente, esegue una operazione di I/O alla volta.
  - *Sostituito da SATA (Serial ATA) velocità fino a 250MB/s (potenzialità: 500MB/s)*

- SCSI (Small Computer Systems Interface):
  - Sistema definito da interfaccia e controller
  - Velocità di trasferimento elevata (fino a 640 MB/s)
  - Il trasferimento dei dati viene gestito dal controller (integrato nel dispositivo)



RAID – Redundant Array of Independent Disks

Scopo: migliorare prestazioni ed affidabilità di un sistema di memorizzazione di massa utilizzando una batteria di dischi piuttosto che un unico disco (Petterson et. Al. 1988).

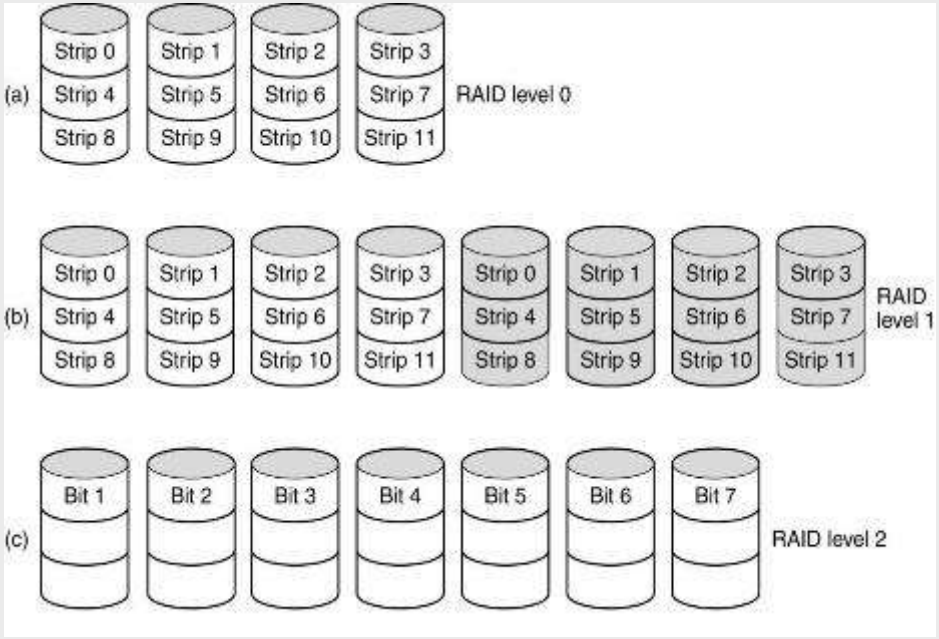
Al sistema operativo un disco RAID appare come un unico disco, potenzialmente con prestazioni e affidabilità maggiori rispetto ai dischi precedenti.

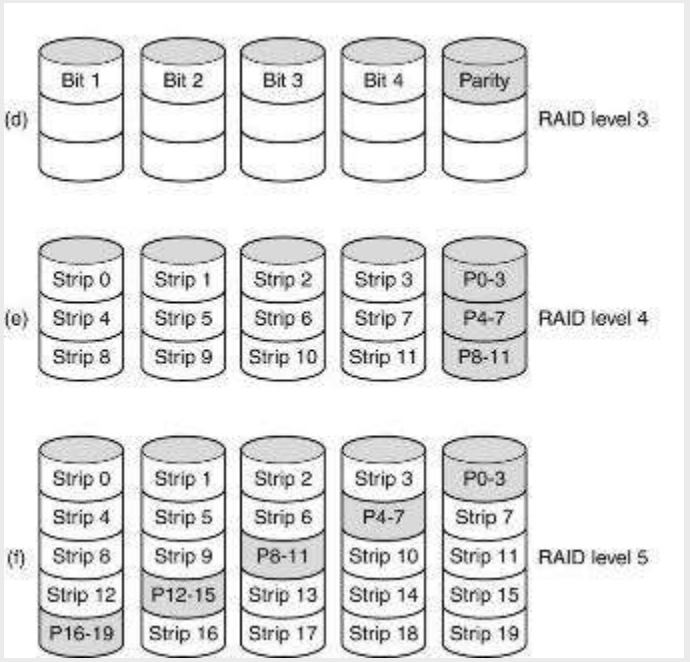
Implementazione:

- tecnologia SCSI che consente l'utilizzo simultaneo di più dischi con buone prestazioni
- un disco RAID consiste in un controllore RAID SCSI più un insieme di dischi SCSI.

I RAID hanno la proprietà di distribuire i dati sulle diverse unità consentendo elaborazioni parallele.

RAID – Redundant Array of Independent Disks

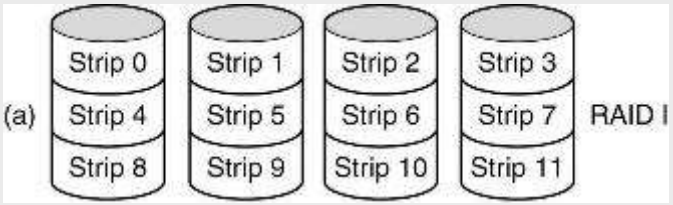




RAID levels 0 through 5.

Backup and parity disks are shown shaded.

RAID 0



In questo caso si usa una tecnica di striping : il disco simulato RAID è visto come se ognuno dei suoi k settori fosse diviso in strip (“strisce”), con (se k sono i dischi del RAID) :

Strip 0: settori da 0 a k-1 ;

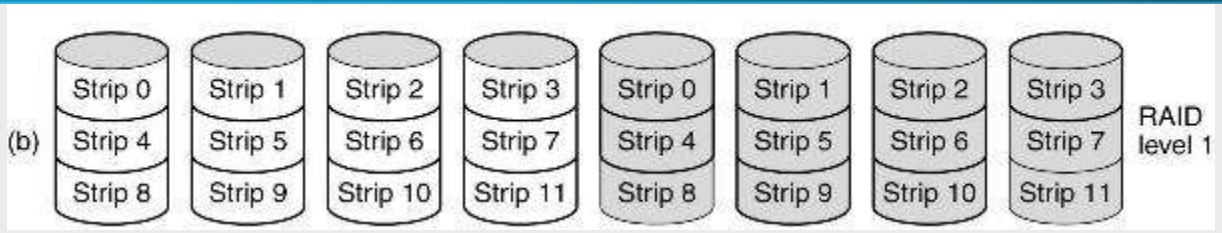
Strip 1: settori da k a 2k-1;

…

Per la lettura/scrittura dati di 4 settori successivi il controllore RAID spezzerà questo comando in 4 comandi separati (uno per ciascun disco) e li farà eseguire in parallelo).

RAID 0 lavora meglio quando le richieste sono di grandi dimensioni. RAID 0 non è ridondante (per questo non è considerato un vero e proprio “RAID”).

RAID 1



In questo caso si usa una tecnica di striping+mirroring. Come nel RAID 0 ogni strip viene mappata su due diversi dischi.

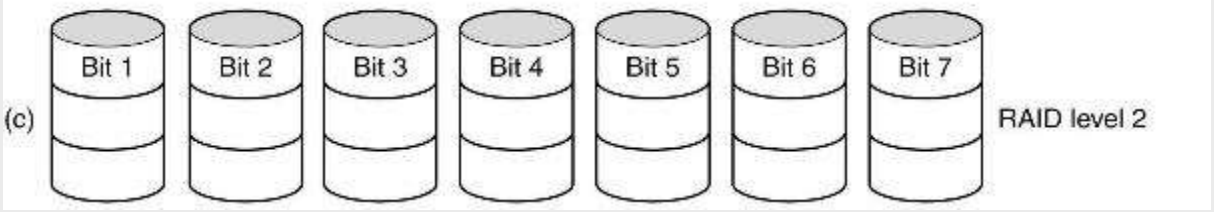
Vantaggi:

- *Una richiesta di lettura può essere eseguita dal disco più "scarico"*
- *Una richiesta di scrittura deve essere eseguita su due dischi (ma potenzialmente in parallelo)*
- *Il malfunzionamento di un disco è facilmente recuperabile.*

Svantaggi:

- *Costi elevati*

RAID 2



Usa una tecnica di accesso parallelo. Tutti i dischi del RAID partecipano all'esecuzione delle richieste di I/O in maniera sincrona (tutte le testine di tutti i dischi assumono posizioni analoghe in ogni momento).

RAID 2 adotta lo striping dei dati, ma questa volta le strisce sono di dimensione di mezzo byte, un byte o una parola.

Lavorando con mezzo byte (+ tre bit di controllo dati) si ottengono i sette bit memorizzati tutti su dischi diversi (come mostrato in figura).

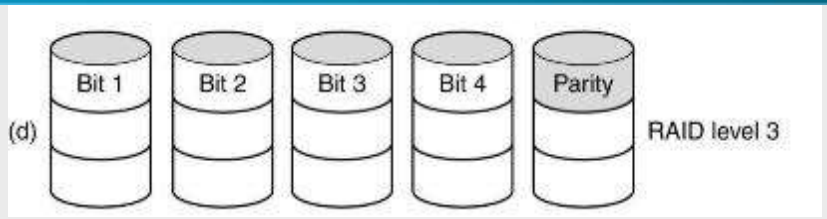
Vantaggi:

- *Lettura/scrittura in parallelo (di ciascun mezzo byte + tre bit di controllo dati)*

Svantaggi:

- *Dispendioso (meno che RAID 1)*
- *Rotazione sincronizzata dei dischi*
- *Molto lavoro al controllore che deve controllare rapidamente i bit di controllo*

RAID 3



Usa una tecnica di accesso parallelo. E' una versione semplificata del RAID 2. Il bit di parità viene calcolato per ogni parola di dati e poi scritto su un apposito disco.

Dato che le parole di dati sono distribuite su più unità, anche in questo caso i dischi devono essere sincronizzati.

Nel caso di una rottura di un disco, attraverso il bit di parità si riesce a recuperare i dati.

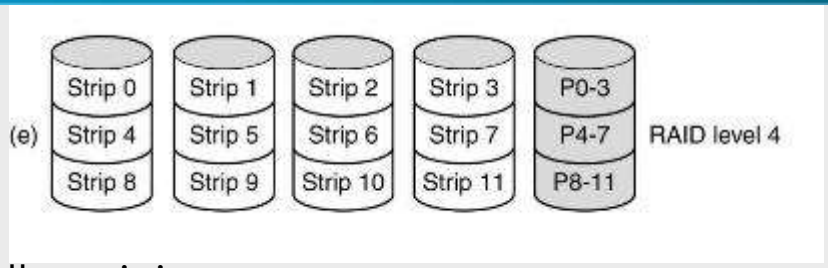
Vantaggi:

- *Lettura/scrittura in parallelo (di ciascun mezzo byte + 1 bit di parità)*

Svantaggi:

- *Rotazione sincronizzata dei dischi*

RAID 4



Usa una tecnica basata sullo striping.

Il RAID 4 è come il RAID 0 con una parità calcolata strip-per-strip, che viene scritta su un disco aggiuntivo.

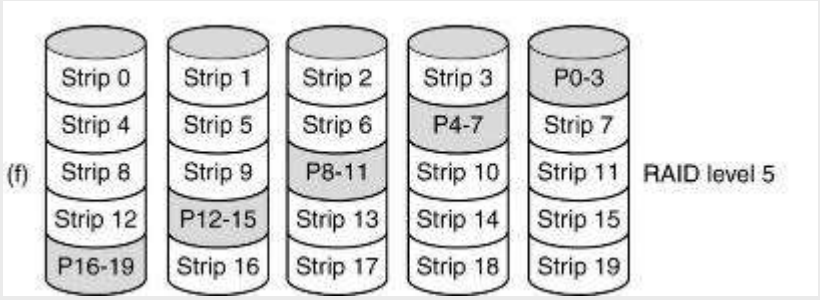
Vantaggi:

- *Protegge dalla rottura di un disco (l'informazione è recuperabile)*

Svantaggi:

- *Ha prestazioni scarse se si cambiano piccole moli di dati. Infatti per ogni, pur piccolo, cambiamento è necessario ricalcolare tutto lo strip di parità.*
- *Il disco per la parità diventa un "collo di bottiglia" per il sistema*

RAID 5



Usa una tecnica basata sullo striping.

Il RAID 5 è come il RAID 4 con gli strip di parità distribuiti su tutti i dischi, in modalità "round-robin" una parità calcolata strip-per-strip, che viene scritta su un disco aggiuntivo.

Vantaggi:

- *Protegge dalla rottura di un disco (l'informazione è recuperabile)*

Svantaggi:

- *Ha prestazioni scarse se si cambiano piccole moli di dati. Infatti per ogni, pur piccolo, cambiamento è necessario ricalcolare tutto lo strip di parità.*

Gestione dell' HARD DISK

Deframmentazione (XP: pannello di controllo ->strumenti di amministrazione ->gestione computer
7: start -> tutti i programmi -> accessori -> utilità sistema)

Formattazione: dividere la capacità del disco in una serie di blocchi di uguali dimensioni e fornire una struttura in cui verranno scritte le informazioni.

FISICA o di BASSO LIVELLO: il disco inizialmente non è suddiviso in tracce e settori. L'operazione che tramite induzione magnetica suddivide il disco in tracce e settori prende il nome di formattazione fisica

LOGICA o di ALTO LIVELLO: Lo spazio di memoria dell'HD viene organizzato logicamente per poter essere accessibile da uno specifico sistema operativo (FAT32, iNODE..)

Partizionamento: suddivisione di un'unità fisica in più unità logiche. Le singole unità logiche vengono viste dal sistema operativo come unità separate e possono essere formattate e gestite in modo indipendente.

Cosa significa cancellare un file??? :D (drive rescue e eraser)

SSD – Memorie a stato solido

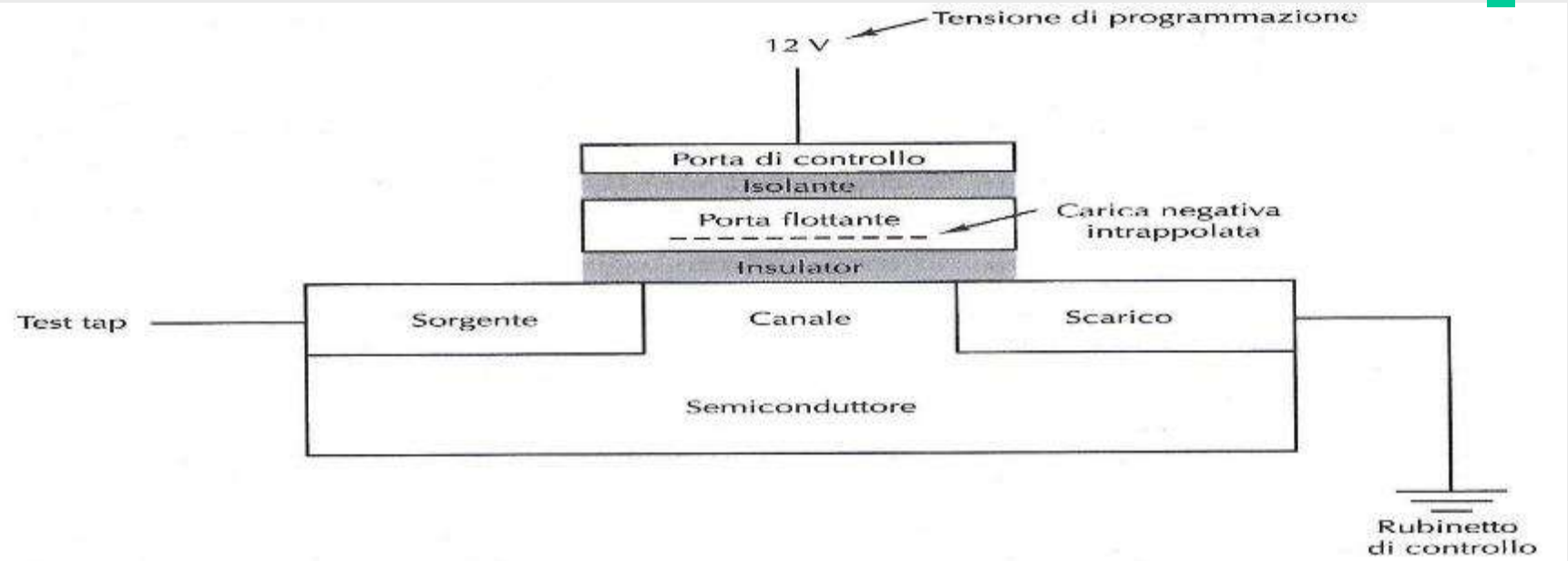


Figura 2.24 Una cella di memoria flash.

Basate su memorie flash.

PRO: Non hanno bisogno di rotazione: lettura/scrittura immediata.

CONTRO: Costo, possono essere scritte soltanto 100.000 volte

Osservati problemi di affidabilità.

Scheda Video & Scheda Audio

Scheda video e scheda audio sono componenti essenziali di un PC multimediale.

- **Scheda video** (Video Graphic Adapter o VGA): consente di visualizzare sullo schermo del monitor le informazioni elaborate dalla CPU.
  - Dispongono di un processore e di una memoria RAM (dimensioni) perché, oltre a raccogliere le informazioni ricevute dalla CPU, le elaborano prima di inviarle al monitor.
  - Utilizzano la CPU e condividono la RAM del Computer.

- **Scheda audio:** ha il compito di sintetizzare i suoni da inviare alle casse acustiche (riproduzione) o di registrare i suoni (campionamento e quantizzazione) acquisiti da una fonte esterna (microfono, lettore CD, ecc...).
  - Dispone di memoria dedicata e di un proprio processore
  - Integrata nella scheda madre.

Monitor LCD a matrice attiva (TFT)

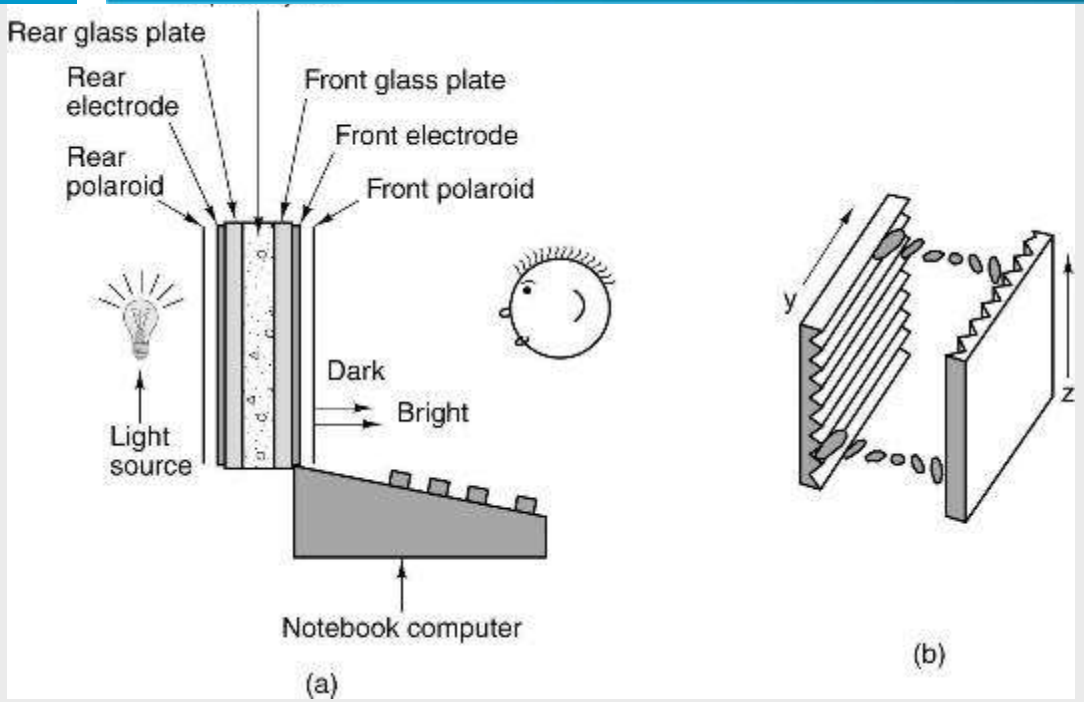
Principali Parametri:

- Contrasto: rapporto fra la luminosità del bianco e la luminosità del nero

  Monitor a LED, miglioramento del contrasto sulla singola immagine, agendo dinamicamente sulle varie porzioni di retroilluminazione.

  Forti contrasti sono tuttavia necessari solo per l'uso in piena luce dello schermo.
- Luminosità
- Linearità dei grigi: imperfezione delle tonalità
- Angolo di visuale: angolo massimo sotto cui si può guardare lo schermo mantenendo una luminosità ed un contrasto "accettabili": il grado di "accettabilità" può essere liberamente stabilito dai produttori..diversi produttori, diversi angoli
- Tempo di risposta: tempo necessario ai "cristalli liquidi" per passare da uno stato "tutto chiuso" (nero) ad uno "tutto aperto" (bianco), per poi tornare al "tutto chiuso". Da 60 fino a 120 Hz un monitor per PC.

Monitor LCD a matrice attiva (TFT)



La tecnologia più comune alla base degli schermi piatti è quella dello schermo a cristalli liquidi, LCD (Liquid Crystal Display).

Uno schermo LCD consiste in due lastre di vetro tra le quali è posto un cristallo liquido.

Alle lastre sono attaccati degli elettrodi trasparenti utilizzati per creare campi elettrici all'interno del cristallo liquido.

Inoltre sono presenti sulle due lastre dei filtri polarizzati.

I cristalli liquidi sono molecole organiche viscose che si muovono come un liquido, ma che hanno una struttura spaziale simile a quella di un cristallo.

Quando tutte le molecole sono orientate nella stessa direzione, le proprietà ottiche del cristallo dipendono dalla polarizzazione della luce incidente.

E' possibile modificare l'allineamento dei cristalli e quindi le proprietà ottiche. In particolare si può controllare l'intensità della luce uscente. Questo principio è utilizzato per la costruzione degli schermi.

Categorie di Calcolatori con Tassonomia di Flynn

Singolo Processore

- Single Instruction Single Data (SISD)
  - Nessun parallelismo (Von Neumann)

Processori Paralleli

- Single Instruction Multiple Data (SIMD)
  - ALU vettoriali

- Multiple Instruction Single Data (MISD)
  - Una stessa sequenza di dati è trasmessa a un set di processori
  - Ogni processore esegue una differente sequenza di istruzioni sugli stessi dati
  - Mai implementata

- Multiple Instruction Multiple Data (MIMD)
  - Un set di processori esegue simultaneamente diverse sequenze di istruzioni su set diversi di dati

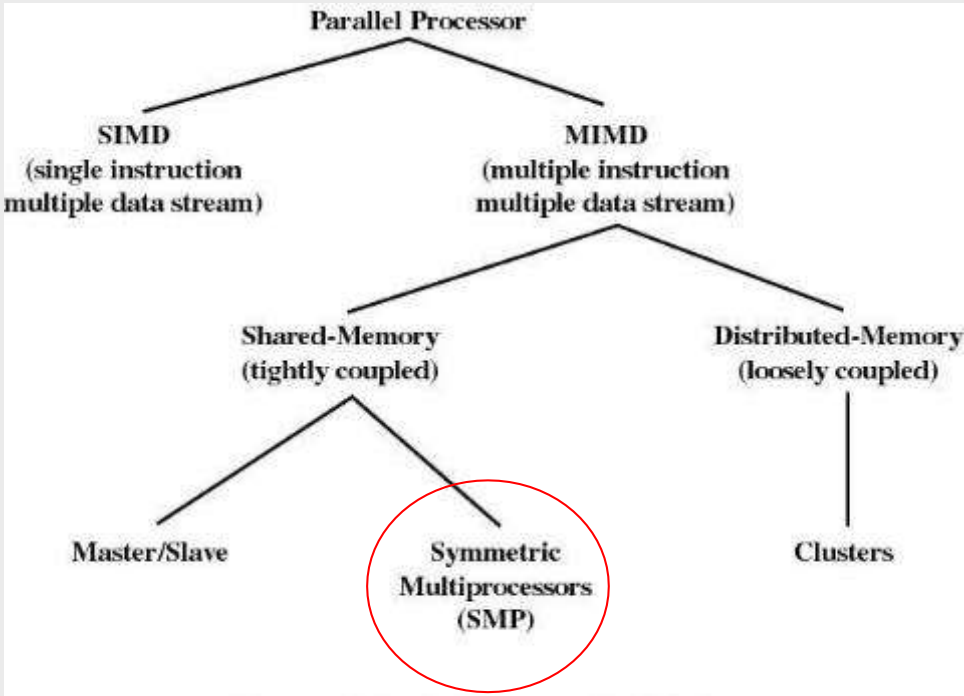


Figure 4.7 Parallel Processor Architectures

Parallelismo

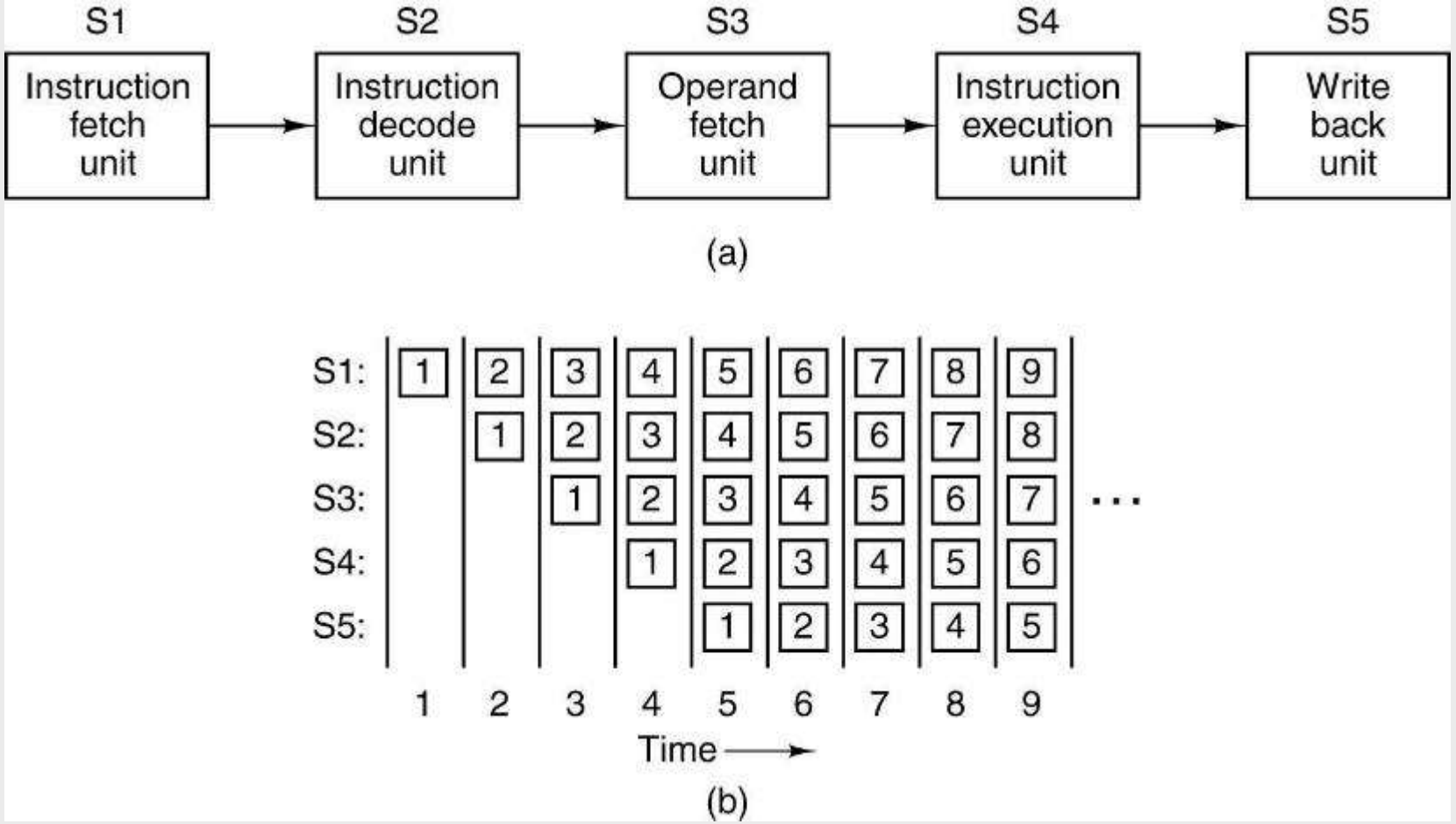
Parallelismo: capacità di eseguire più azioni nello stesso istante.

Esistono due tipologie di parallelismo:

- **a livello di processore**: più processori lavorano congiuntamente sullo stesso problema
- **a livello di istruzione**: il parallelismo viene sfruttato all'interno delle singole istruzioni per fare in modo che l'elaboratore esegue più istruzioni contemporaneamente.

*Uno dei concetti applicati è quello di **pipeline**.*
Il meccanismo di pipeline divide il ciclo di esecuzione in più parti e ciascuna parte è affidata ad una componente hardware.

Parallelismo - Pipeline

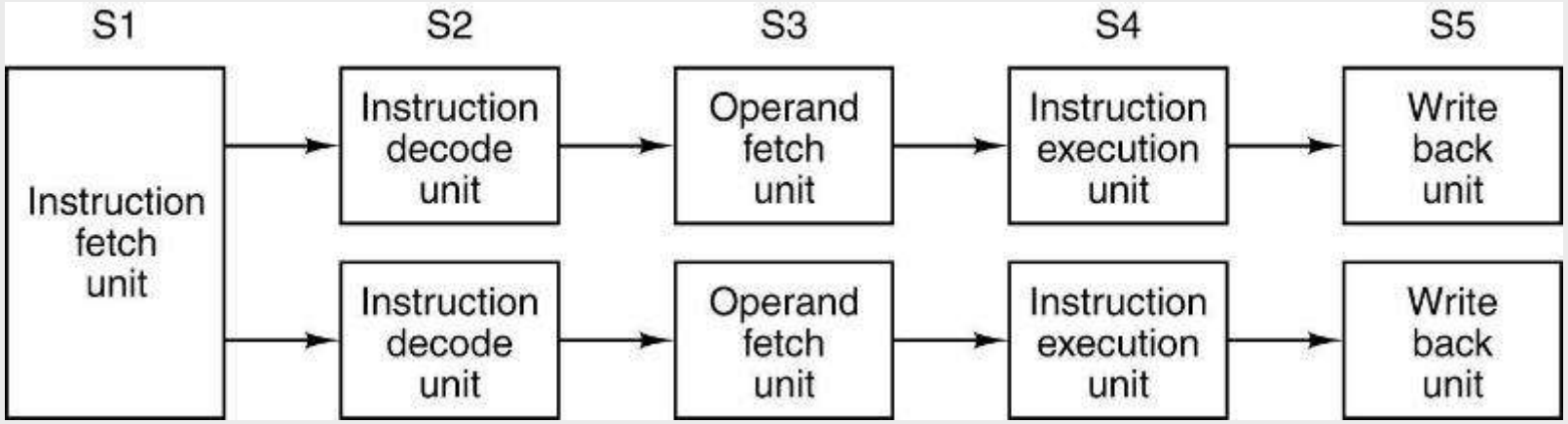


- Pipeline a cinque fasi
- Avanzamento degli stadi in funzione del tempo. Sono rappresentanti nove cicli

Parallelismo - Pipeline

L'Intel ha dotato i propri processori, a partire dal 486, di una pipeline, e a partire dal Pentium, di due pipeline a cinque stadi.

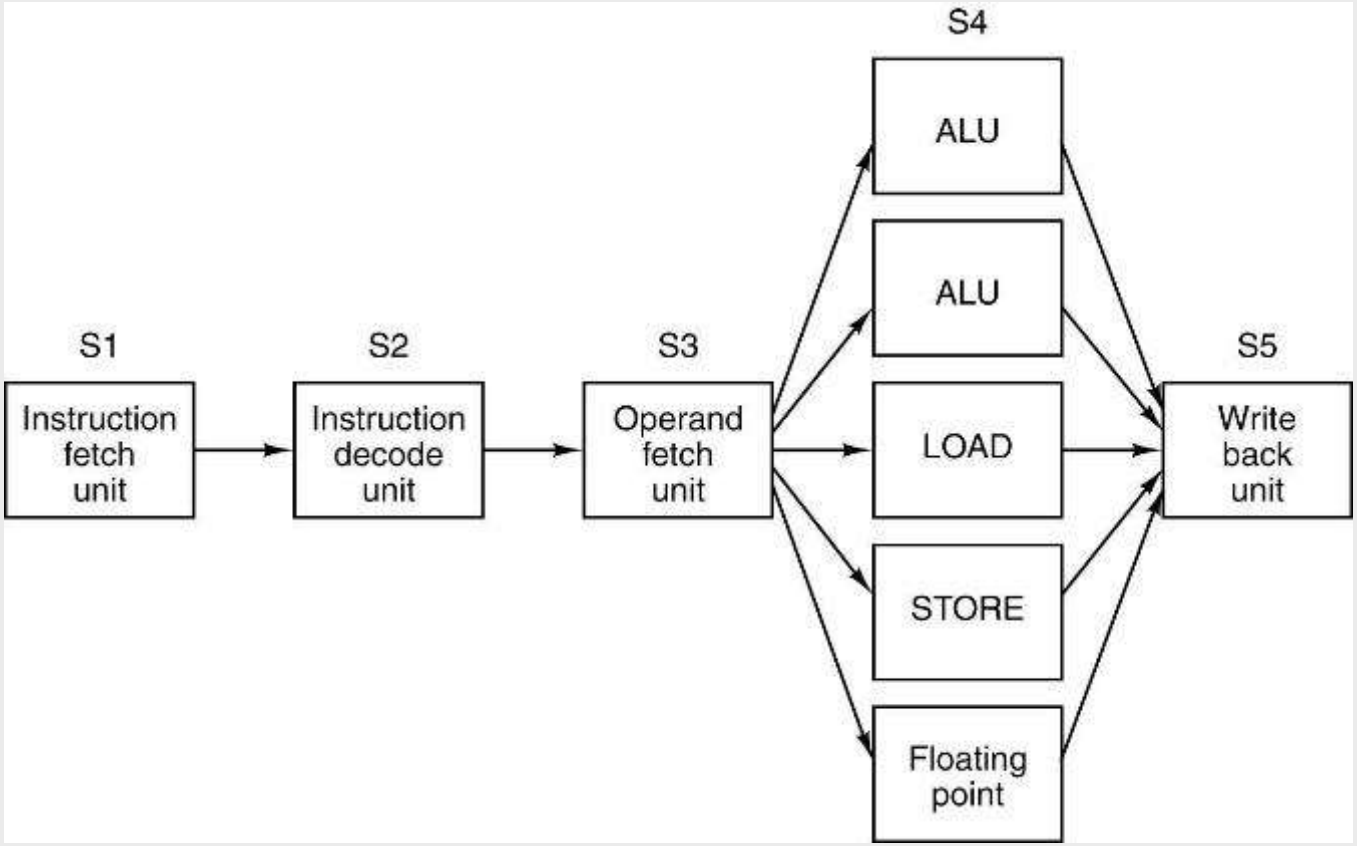
- La pipeline principale detta pipeline u esegue una qualsiasi istruzione Pentium.
- La seconda pipeline, detta pipeline v, esegue solo semplici istruzioni su interi e una su numeri a virgola mobile.



Doppia pipeline a cinque stadi con unità di fetch dell'istruzione in comune.

Parallelismo - Pipeline

Processore superscalare con 5 unità funzionali: architettura conuna sola pipeline alla quale vengono associate più unità funzionali.



Parallelismo - Pipeline

Sia

- k - numero di stadi della pipeline
- n - numero di istruzioni da eseguire
- t - tempo di uno stadio (tempo massimo).

(si considerino inoltre trascurabili i tempi di commutazione da uno stadio al successivo)

In una macchina sequenziale senza parallelismo il tempo per eseguire le istruzioni è

$$T_{convenzionale} = nkt$$

In una macchina dotata di pipe (senza salti):

$$T_{pipe} = [k + (n-1)] \cdot t$$

Il fattore di velocizzazione (speed-up) della pipeline rispetto ad una architettura tradizionale è:

$$\frac{T_{convenzionale}}{T_{pipe}} = \frac{nkt}{[k + (n-1)]t} = \frac{nk}{[k + (n-1)]}$$

Parallelismo – Pipeline
Trattamento dei salti

Le istruzioni di salto (condizionato) sono uno dei principali problemi che limitano le prestazione di una pipeline.

Le possibili soluzioni sono:

- Flussi multipli (multiple stream)
- Prelievo anticipato della destinazione (prefetch branch target)
- Buffer circolare (loop buffer)
- Predizione di salto (branch prediction)
- Salto ritardato (delayed branch)

Parallelismo – Pipeline
Trattamento dei salti

Flussi multipli (multiple stream)

- Si replicano le parti iniziali della pipeline per consentirle di prelevare entrambe le istruzioni (coinvolte nel salto condizionato), facendo uso dei due flussi.
- Problemi:
  - Generazione di ritardi per la contesa tra i due flussi
  - In ciascuno dei due flussi si possono presentare altre istruzioni di salto condizionato…

Prelievo anticipato della destinazione (prefetch branch target)

- Quando viene riconosciuto un salto condizionato viene anche prelevata anticipatamente (prefetching) la sua destinazione.
- L'indirizzo è mantenuto fino a quando si dovrà eseguire l'istruzione di salto di modo che, se occorre compiere il salto, l'indirizzo è già stato prelevato.

Parallelismo – Pipeline
Trattamento dei salti

Buffer circolare (loop buffer)

Piccola e veloce memoria che contiene le ultime n istruzioni prelevate con il fetch. Se occorre effettuare un salto l'hardware prima controlla se la destinazione si trova nel buffer. In caso affermativo la successiva istruzione viene prelevata dal buffer.

Vantaggi:

- Anticipando il fetch è possibile avere già nella memoria circolare alcune istruzioni evitando di dover far riferimento in memoria alcune volte;
- Nei cicli, se le istruzioni dell'intero ciclo riescono a stare nel buffer circolare, si ha un massimo vantaggio.

Parallelismo – Pipeline
Trattamento dei salti

Predizione di salto (branch prediction)

- Previsione di saltare sempre (approccio statico)
- Previsione di non saltare mai (approccio statico)
- Previsione in base al codice operativo (approccio statico)

- Bit taken/not taken (approccio dinamico)
- Tabella della storia dei salti (approccio dinamico)

Parallelismo – Pipeline
Trattamento dei salti

Predizione di salto (branch prediction)
Bit taken/not taken

Ad ogni istruzione di salto condizionato in cache viene associato un bit (taken/not taken) che memorizza il comportamento recente di quella istruzione.

Questa informazione viene utilizzata per anticipare il comportamento dell’istruzione di salto.

Parallelismo – Pipeline
Trattamento dei salti

Predizione di salto (branch prediction)
Tabella con la storia dei salti.

Viene usata una tabella con la storia dei salti.

Ciascuna riga contiene:

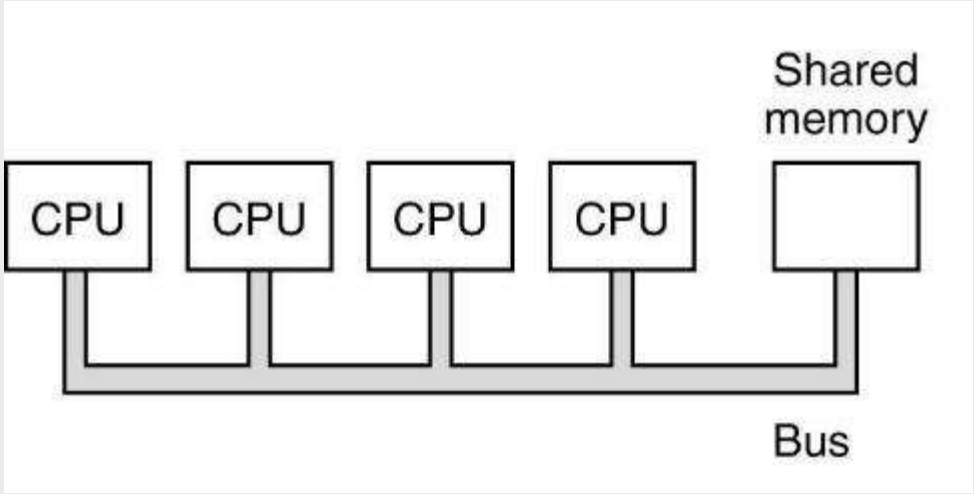
1) l'indirizzo dell'istruzione di salto
2) un certo numero di bit di storia
3) informazioni circa l'istruzione destinazione (di solito il suo indirizzo)

Parallelismo – Pipeline
Trattamento dei salti

Salto ritardato (delayed branch)

Ridisporre automaticamente le istruzioni all'interno di un programma in maniera tale che le istruzioni di salto si verifichino in ritardo rispetto al momento in cui effettivamente desiderate.

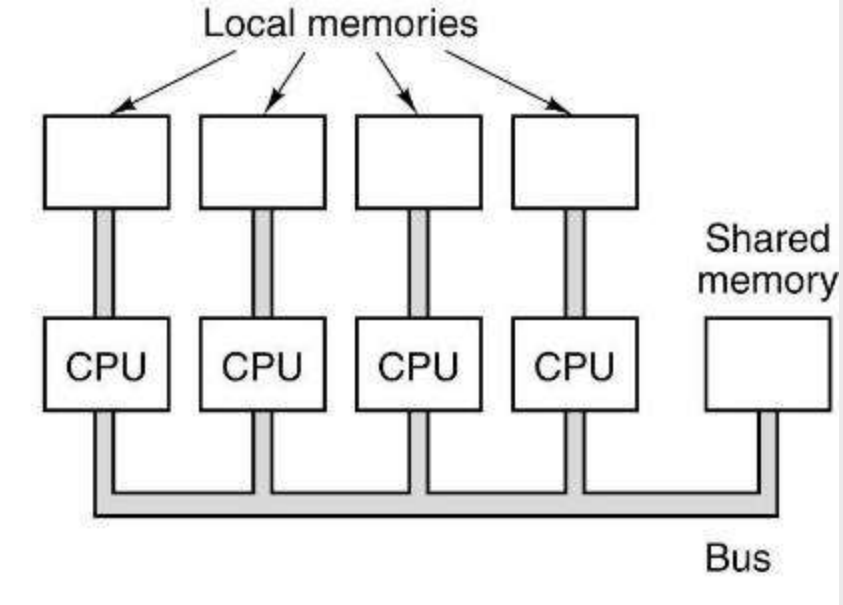
Multiprocessori



Multiprocessore: Più processori, una RAM
E' necessario che le CPU siano sincronizzate per evitare conflitti nelle operazioni sulla memoria condivisa.
CPU fortemente accoppiate.

Ad esempio:
Se nella memoria è contenuta una immagine da elaborare è possibile affidare una sezione della immagine ad ogni CPU che eseguirebbe il medesimo programma delle altre sulla sezione di immagine ad essa affidata, riducendo il tempo di elaborazione dell'immagine

MultiComputer



Più CPU dotate di una memoria privata nel quale vengono contenuti dati che non sono condivisi ma utili all'elaborazione.

Il minor scambio sul bus rende più veloce l'esecuzione.

L'area condivisa è utilizzata per contenere, ad esempio, il codice del programma da eseguire.

Le **CPU sono debolmente connesse,** la comunicazione fra di loro avviene attraverso l'uso di messaggi che viaggiano sul bus che le collega.

Symmetric Multi Processing (SMP)

- Un calcolatore con molti processori
- I processori condividono le stesse risorse
- Tutti i processori possono effettuare le stesse funzioni
- Ogni processore esegue una stessa copia del SO
- Vantaggi:
  - *parallelismo*, i thread possono essere schedulati su tutti i processori
  - *disponibilità*: simmetria dei processori, se uno fallisce, gli altri possono continuare a lavorare
  - crescita incrementale: aggiungere nuovi processori
  - Scalabilità
  - NB: vantaggi potenziali e non garantiti
- Presenza di più processori trasparente all'utente

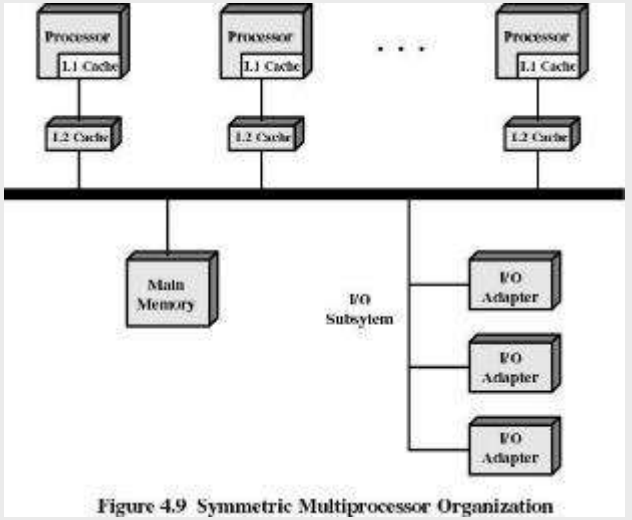
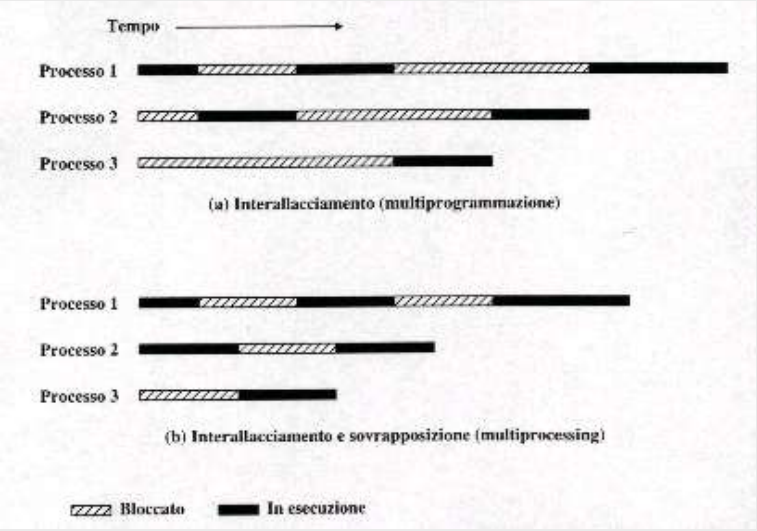


Figure 4.9 Symmetric Multiprocessor Organization



Vantaggio numerico di un Multiprocessore

Sia P un processo e supponiamo che sia costituito da una frazione seriale f_s (realizzabile in un tempo T_s) e da una frazione f_p realizzabile in parallelo (realizzabile in un tempo T_p) dove ovviamente $f_s+f_p=1$.

Sia T_1 e T_n rispettivamente il tempo di esecuzione del processo su 1 ed n processori:

$T_1=T_s+T_p$ (può essere considerato normalizzato a 1)

$T_n=T_s+T_p/n$

Lo speed-up S(n) con n processori è pari a T_1/T_n (Legge di Andahl)

$$S(n)=\frac{T_1}{T_n}=\frac{1}{f_s+\frac{f_p}{n}}=\frac{1}{f_s+\frac{1-f_s}{n}}=\frac{n}{nf_s+(1-f_s)}$$

NB: fatevi qualche esempio numerico per capire quando «serve» un multiprocessore!!!!

DRIVE e Dispositivi Mobili

- Lettori/Masterizzatori CD, DVD
- Supporti in grado di immagazzinare dati fino a 700 MB (CD) e fino a 4.7 GB (DVD).

Blu Ray (54 GB), DVD+R/+RW

	CD impresso (da matrice)	CD-R	CD-RW	DVD impresso (da matrice)	DVD-R	DVD+R	DVD-RW	DVD+RW	DVD+R DL	DVD-RAM	BD-R
Lettore CD audio	Lettura	Lettura[1]	Lettura[2]	No	No	No	No	No	No	No	No
Lettore CD-ROM	Lettura	Lettura[1]	Lettura[2]	No	No	No	No	No	No	No	No
Masterizzatore CD-R	Lettura	Scrittura	Lettura	No	No	No	No	No	No	No	No
Masterizzatore CD-RW	Lettura	Scrittura	Scrittura	No	No	No	No	No	No	No	No
Lettore DVD-ROM	Lettura	Lettura[3]	Lettura[3]	Lettura	Lettura[4]	Lettura[4]	Lettura[4]	Lettura[4]	Lettura[5]	No	No
Masterizzatore DVD-R	Lettura	Scrittura	Scrittura	Lettura	Scrittura	Lettura	Lettura[6]	Lettura	Lettura[5]	No	No
Masterizzatore DVD-RW	Lettura	Scrittura	Scrittura	Lettura	Scrittura	Lettura[6]	Scrittura[7]	Lettura	Lettura[5]	No	No
Masterizzatore DVD+R	Lettura	Scrittura	Scrittura	Lettura	Lettura	Scrittura	Lettura	Lettura	Lettura[5]	No	No
Masterizzatore DVD+RW	Lettura	Scrittura	Scrittura	Lettura	Lettura	Scrittura	Lettura	Scrittura	Lettura[5]	No	No
Masterizzatore DVD±RW	Lettura	Scrittura	Scrittura	Lettura	Scrittura	Scrittura	Scrittura	Scrittura	Lettura[5]	No	No
Masterizzatore DVD±RW/DVD+R DL	Lettura	Scrittura	Scrittura	Lettura	Scrittura[8]	Scrittura	Scrittura[8]	Scrittura	Scrittura	No	No
Masterizzatore DVD Super Multi	Lettura	Scrittura	Scrittura	Lettura	Scrittura	Scrittura	Scrittura	Scrittura	Scrittura	Scrittura	No
Masterizzatore Blu-Ray Super Multi	Lettura	Scrittura	Scrittura	Lettura	Scrittura	Scrittura	Scrittura	Scrittura	Scrittura	Scrittura	Scrittura

Stampanti – Ink Jet

Carrello si muove per tutta la larghezza del foglio, su di esso sono fissate le testine di stampa, che proiettano sul foglio micro-gocce di colore:

- Meccanismo termico: resistore in corrispondenza di ogni ugello attraverso il quale vengono fatti passare impulsi di corrente; ad ogni impulso si ha l'espulsione della goccia di inchiostro; (Hewlett-Packard, Canon, Lexmark)

- Meccanismo piezoelettrico: sotto ogni ugello è posizionato un canalino circondato da un cristallo piezoelettrico; un impulso elettrico provoca la deformazione del cristallo e conseguentemente la repentina strozzatura del canalino e l'eiezione dell'inchiostro; (Epson)

Stampanti – Ink Jet

Testine di stampa integrate nelle cartucce

esaurimento inchiostro -> sostituzione testina:

costo cartuccia vs efficienza testine

Essiccamento dell'inchiostro nelle testine

Risoluzione: dpi (dot per inches)

Velocità:15 ppm b/n, 10 ppm colori

Quadricromia (ciano/magenta/giallo/nero)

Esacromia (ciano/magenta/giallo/nero/arancione/verde)

Costo medio stampa: 10 cent di Euro/pagina

Vita della stampante in pagine stampate

Stampanti – Laser

Processo di stampa:

1. Il raggio laser infrarosso viene
   a) modulato secondo la sequenza di pixel che deve essere impressa sul foglio,
   b) deflesso da uno specchio rotante su un tamburo fotosensibile elettrizzato che si scarica dove colpito dalla luce.
2. L'elettricità statica attira una fine polvere (toner) di materiali sintetici e pigmenti che viene trasferito sulla carta.
3. Il foglio passa sotto un rullo riscaldato ad elevata temperatura, che fonde il toner facendolo aderire alla carta.

Stampa a colori: quattro toner (nero, ciano, magenta e giallo) trasferiti da un unico tamburo oppure da quattro distinti. (I toner possono essere integrati in uno solo)

Risoluzione: 4800x1200 dpi(dot per inches)

Velocità:70 ppm b/n, 40 ppm colori

Costo medio stampa: 4 cent di Euro/pagina

Sistemi Embedded

Dispositivo incapsulato all'interno del sistema da controllare **progettato per una determinata applicazione** supportato da una piattaforma hardware su misura.
E' essenzialmente un sistema a microprocessore come un PC ma:

- dedicato a svolgere un particolare compito
- ha risorse e dispositivi strettamente necessari
- hardware dedicato
- frequenze di CPU inferiori
- basso consumo
- poca memoria (anche di massa)

Caratteristiche:

- sistemi privi di interazione umana
- capaci di resistere ad eventi dannosi, a vibrazioni e shock
- ripartire in modo autonomo
- possibilmente di dimensioni ridotte
- raggiungibile tramite un qualche tipo di connessione
- deve avere risorse necessarie per l'esecuzione dei processi indispensabili al suo funzionamento
- spesso con vincoli temporali (sistema real-time)