UNIVERSITA' DEGLI STUDI DI FOGGIA
DIPARTIMENTI
DI AREA MEDICA

CdL in Tecniche Sanitarie di Radiologia Medica

Sistemi di Elaborazione delle Informazioni

Prof. Crescenzio Gallo
crescenzio.gallo@unifg.it

Architettura dei calcolatori

Architettura di un calcolatore

Che cos'è un calcolatore? Come funziona un calcolatore?

- un calcolatore è un *sistema;*
- un sistema è un oggetto costituito da molte parti (*componenti*) che interagiscono, cooperando, al fine di ottenere un certo risultato.

Studiare l'architettura di un sistema vuol dire

- individuare ciascun componente del sistema;
- comprendere i principi generali di funzionamento di ciascun componente;
- comprendere come i vari componenti interagiscono tra di loro.

Hardware/Software

La prima decomposizione di un calcolatore è relativa alle seguenti macro-componenti:

Hardware

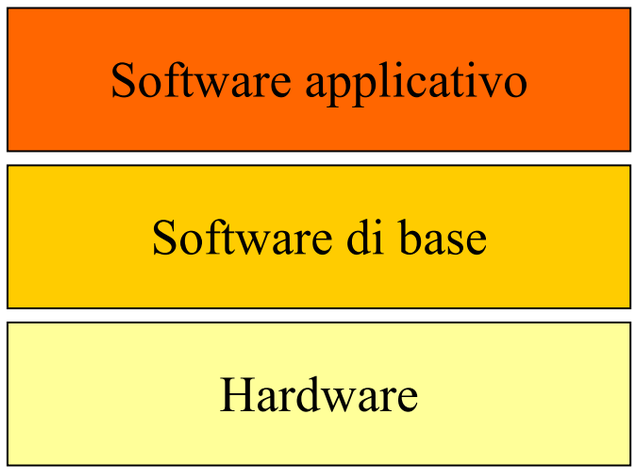
- La struttura fisica del calcolatore, costituita da componenti elettronici ed elettromeccanici

Software

- L'insieme dei programmi che consentono all'hardware di svolgere dei compiti utili.
- Il software comprende il software di base (tra cui il sistema operativo) e il software applicativo.

Organizzazione stratificata

- Hardware e software sono organizzati a livelli (o strati).
- Ciascun livello corrisponde a una macchina (reale o virtuale) in grado di eseguire un proprio insieme di operazioni.
- Ciascun livello fornisce un insieme di operazioni più semplici da utilizzare rispetto a quelle del livello sottostante.
- Ciascun livello è realizzato in termini dell'insieme di operazioni fornite dal livello immediatamente sottostante.



Unità centrale/Memoria

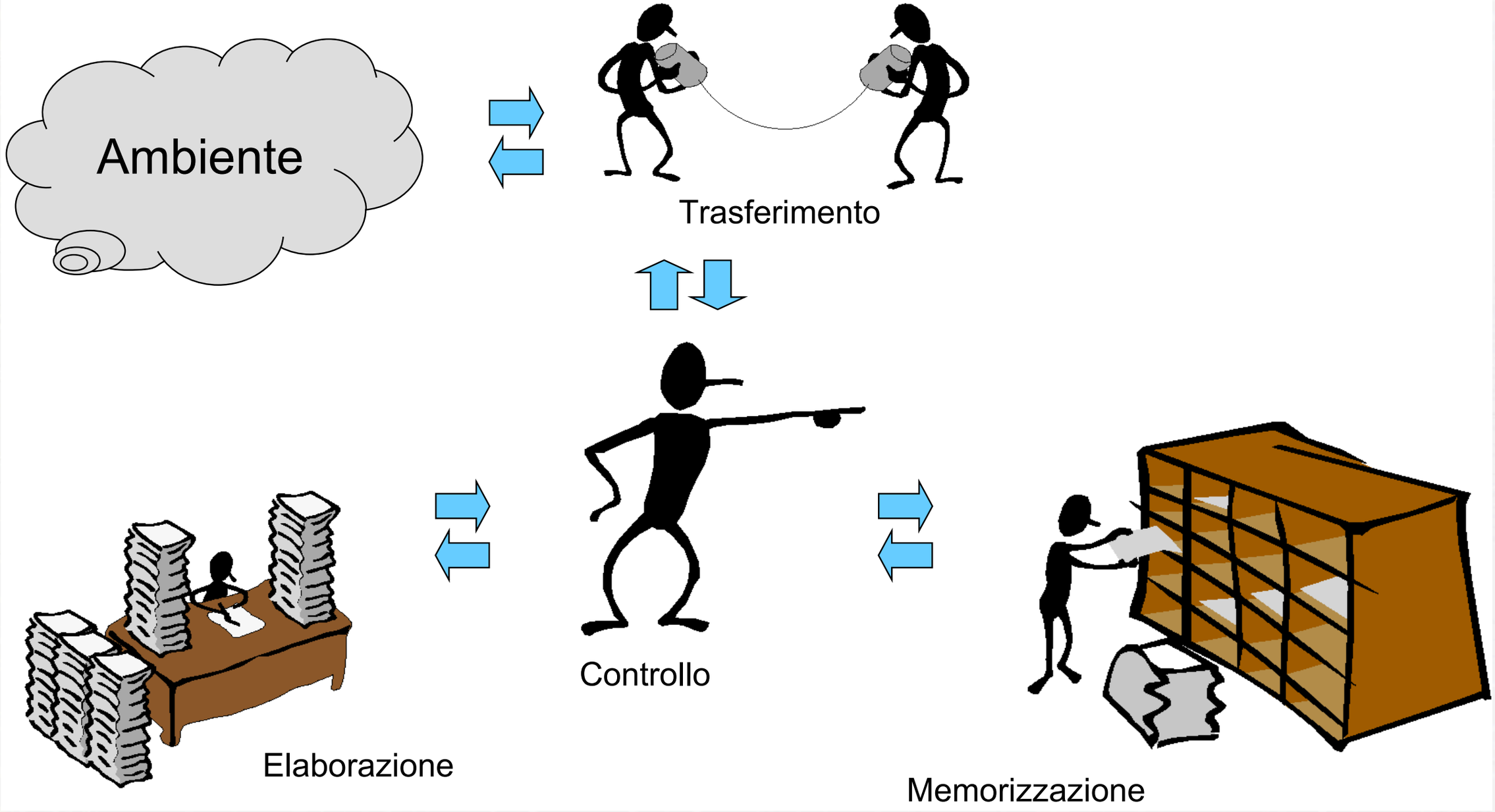
L'unità centrale di elaborazione (**CPU**)

- Componenti fondamentali
- Data path
- Il ciclo macchina

La **memoria**

- Generalità e caratterizzazione
- Gerarchie di memorie e memoria cache

Vista funzionale di un calcolatore

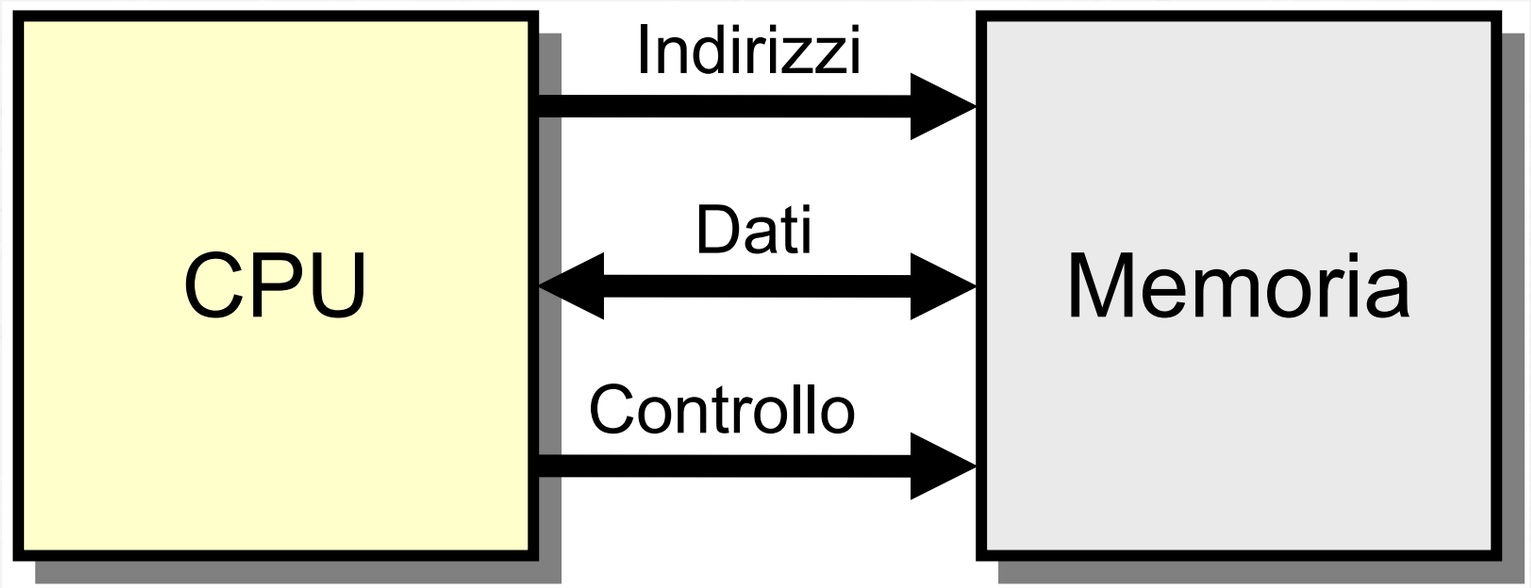


Caratteristiche fondamentali

- Capacità di eseguire sequenze di istruzioni memorizzate
- **Processore (CPU)** = Unità di Elaborazione (ALU) + Unità di Controllo (CU)
  - 1. Preleva le istruzioni dalla memoria
  - 2. Interpreta i codici di istruzione
  - 3. Effettua le azioni che questi prevedono
- **Programma** = Insieme organizzato di istruzioni

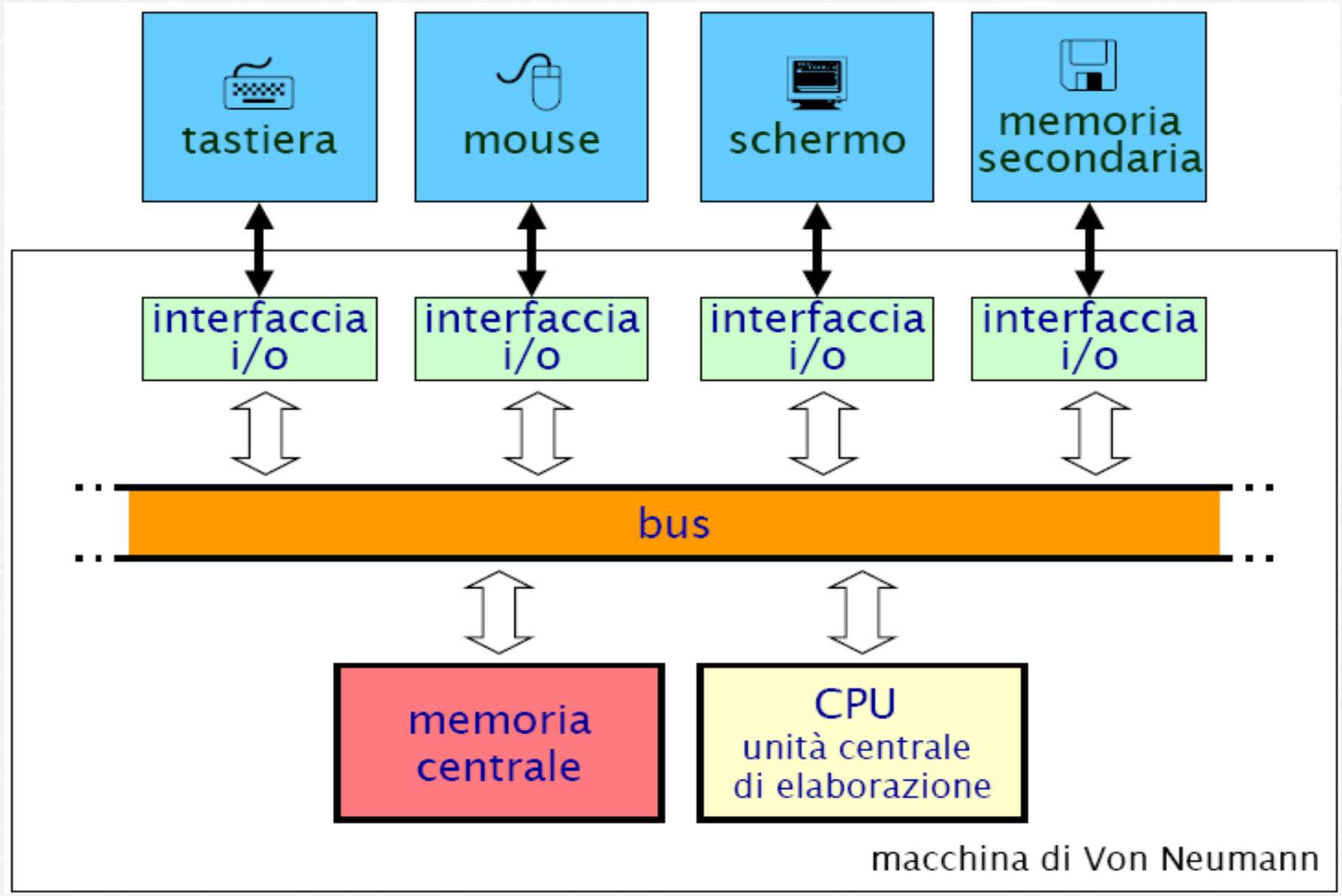
Architettura di von Neumann

Von Neumann fu il primo a proporre che il codice del programma potesse essere memorizzato nella stessa memoria dei dati, a differenza di quanto avveniva prima.



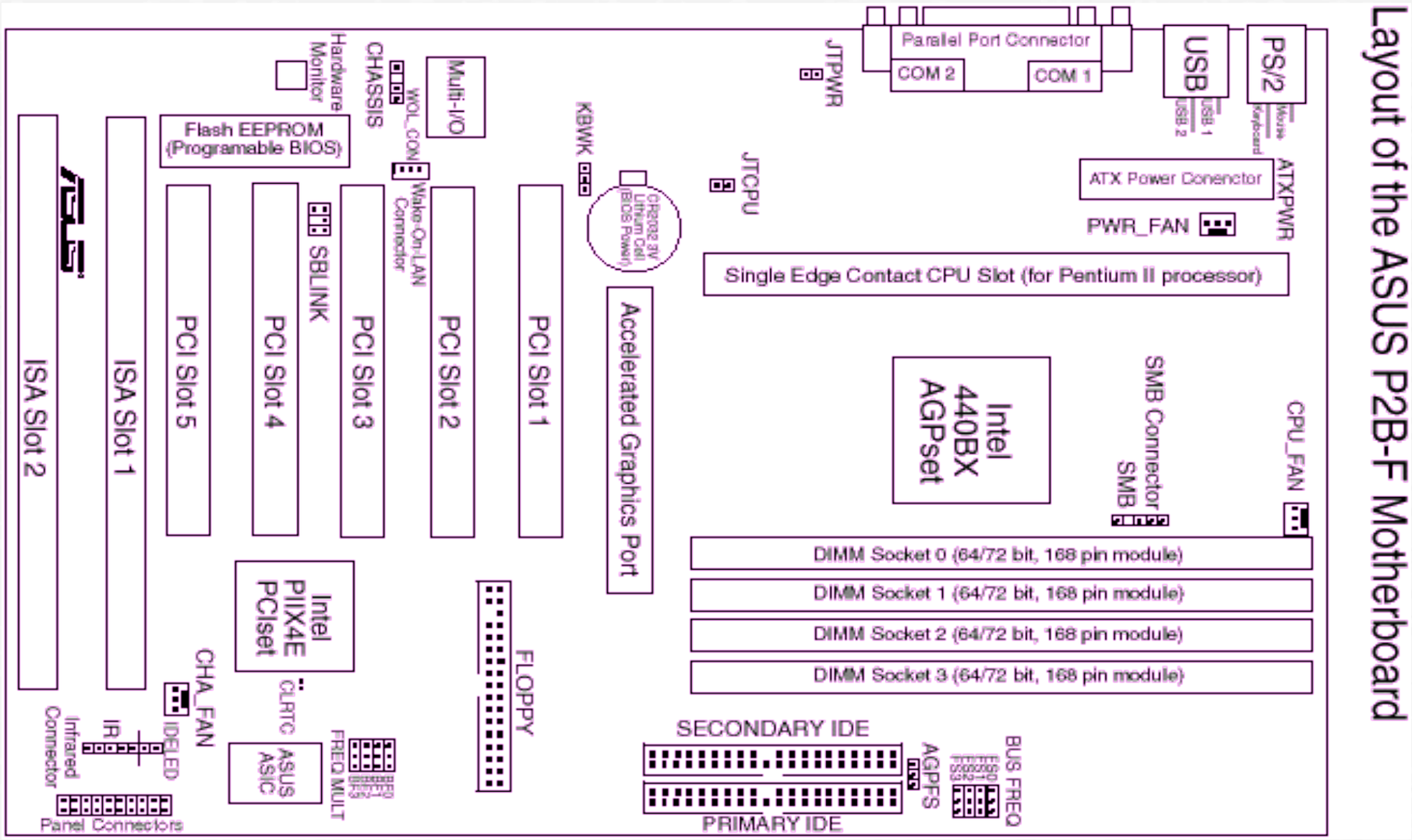
- Memoria indifferenziata per dati o istruzioni.
- Solo l'interpretazione da parte della CPU stabilisce se una data configurazione di bit è da riguardarsi come un dato o come un'istruzione.

Componenti di un calcolatore



Componenti di un calcolatore

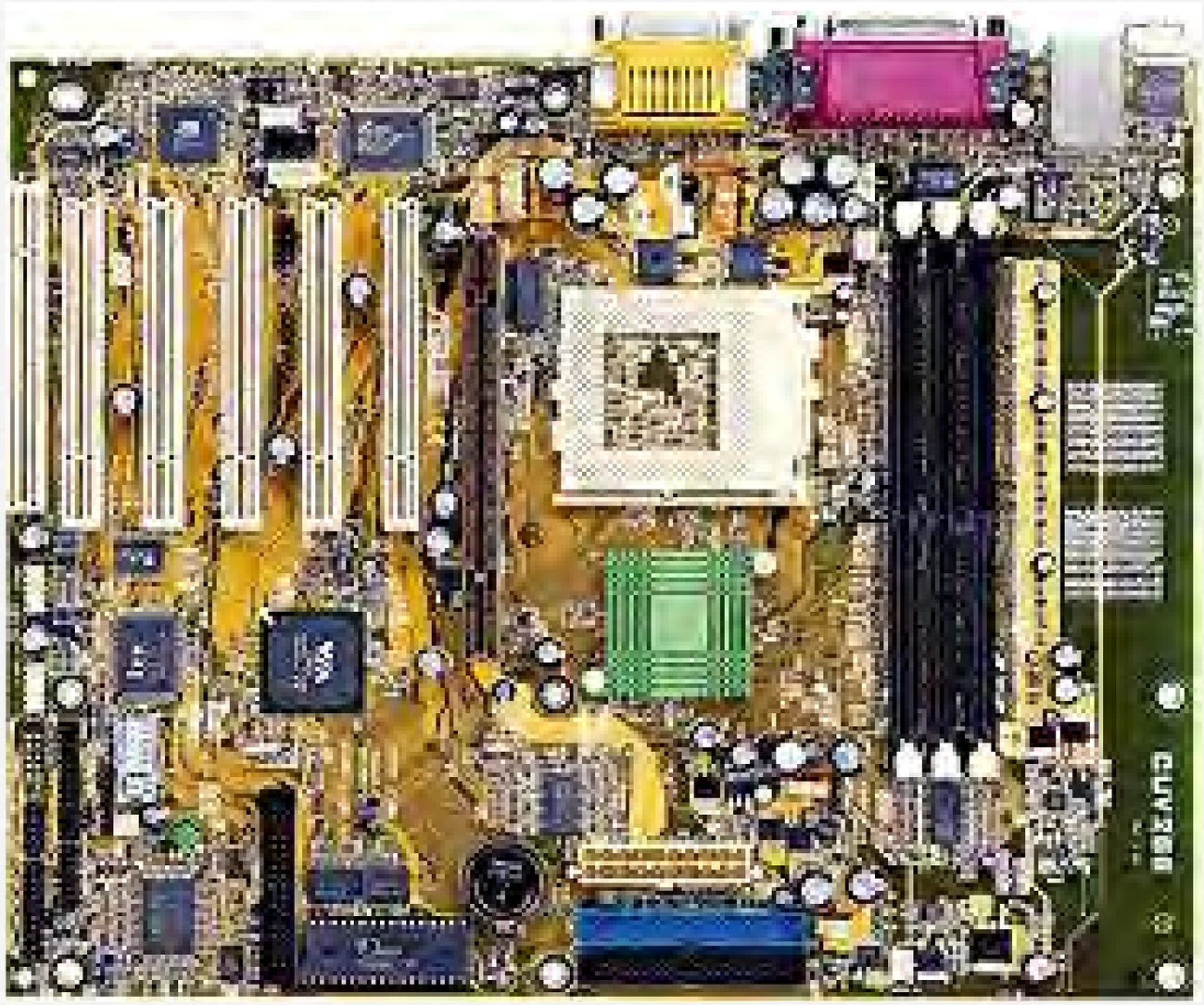
Scheda madre



Componenti di un calcolatore

Scheda madre

Le componenti interne funzionano a corrente continua: l'alimentatore (contenuto nel case o cabinet insieme alla scheda madre) fornisce l'alimentazione elettrica appropriata.



Tipicamente una scheda madre può essere alloggiata in un opportuno case, compatibile per forma e caratteristiche.

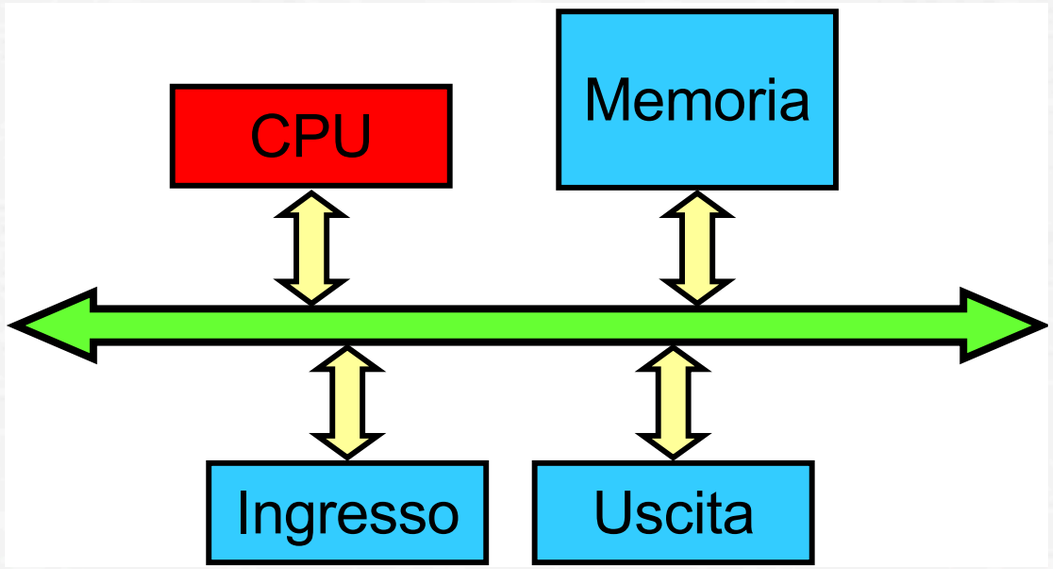
Il Bus

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*, memoria e periferiche = *slave*

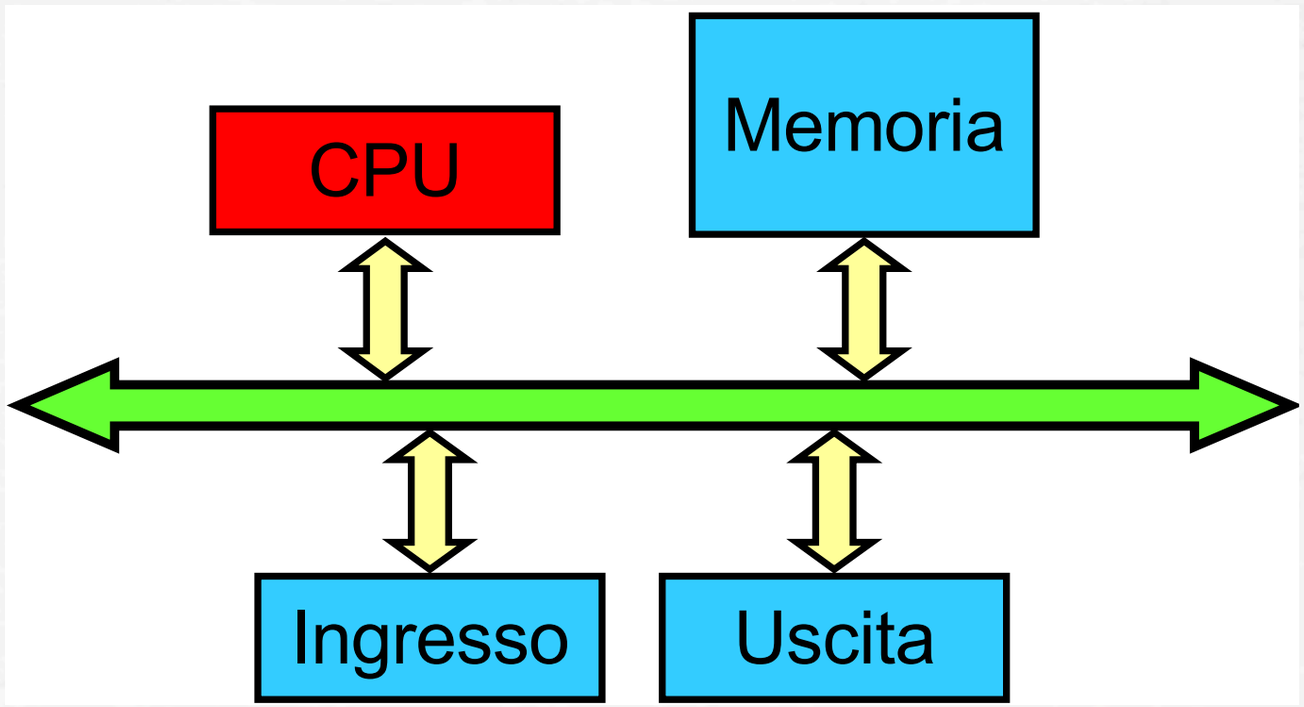
oppure

Canale DMA (accesso diretto delle periferiche alla memoria)



Bus Master-slave: pregi

Semplicità: una sola linea di connessione per tutti i 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.

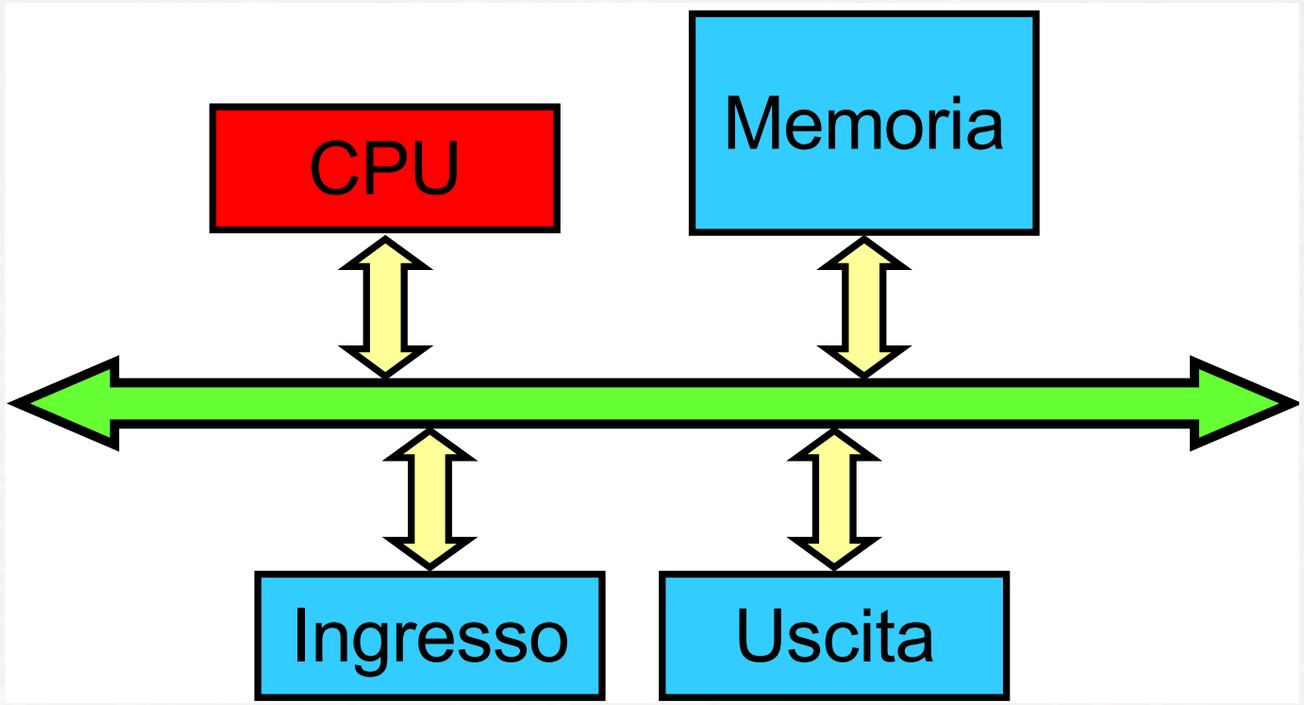


Bus Master-slave: difetti

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.



Bus di sistema: schema

Interfaccia I/O

Interfaccia I/O

Interfaccia I/O

Memoria centrale

CPU

Bus dati

Bus indirizzi

Bus di controllo

Tipi di Bus

Il bus di sistema si divide in tre bus minori.

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).

Tipi di Bus

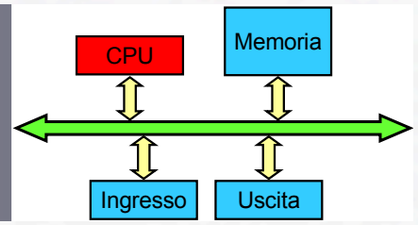
I primi PC “IBM compatibili” degli anni ’80 erano equipaggiati con il bus ISA a 8 bit, poi evoluto nel bus AT a 16 bit e nell’EISA a 32/16 bit.

L’IBM introdusse alla fine degli anni ’80 il bus proprietario MCA a 32 bit, retrocompatibile 16 bit.

Il bus AGP (Accelerated Graphics Port) è un bus locale (cioè connesso direttamente alla CPU) come il suo predecessore VESA. Fu introdotto nel 1997 per le connessioni video ad alta velocità, e sostituito nel 2004 dal bus PCI.

Standard attuale: **PCI** (Peripheral Component Interconnect) **Express** (bus seriale; sino a 2,5 GB/s) di terza generazione.

Bus



- Il bus è assimilabile ad un cavo che collega tra loro i vari componenti del computer. In realtà si tratta di un insieme di cavi su cui viaggiano segnali digitali a velocità particolarmente elevate.
- L'evoluzione tecnologica ha portato i bus a diversificarsi tra loro e a "specializzarsi". Troviamo cosi' i **bus di sistema** (connettono la CPU con la Memoria) oppure i **bus locali**, che connettono a più alta velocità differenti tipi di unità periferiche alla CPU.

A questo proposito può essere utile definire la differenza tra bus e interfaccia. Si tratta in entrambi i casi di connessioni tra diversi dispositivi, ma si parla di bus quando allo stesso cavo sono collegati più dispositivi (con un sistema a "cascata"). L'interfaccia invece può connettere solo un dispositivo.

Bus: architettura

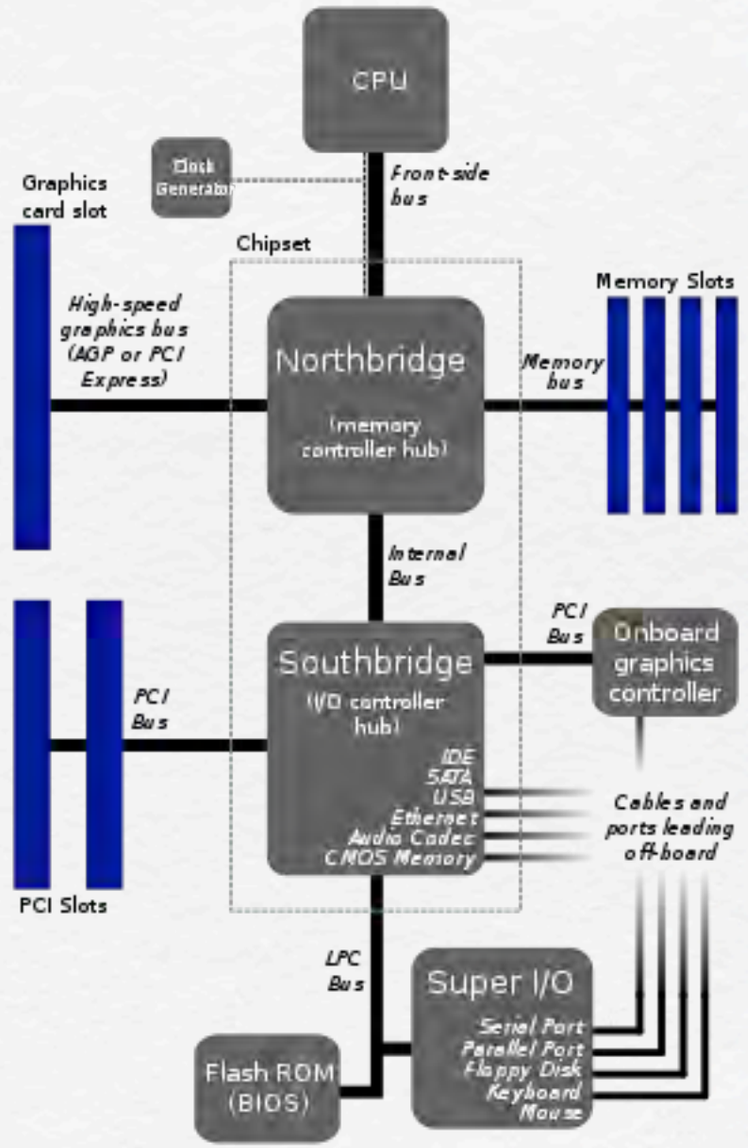
FSB (Front Side Bus) identifica quella parte di bus di sistema che mette in collegamento la CPU con la memoria e qualunque altra periferica installata nel computer (ad esempio modem interni, scheda video, scheda audio, etc.).

Il FSB collega la CPU al resto dell'hardware attraverso un chipset che, di solito, è diviso in *Northbridge* e *Southbridge*.

A questo, poi, si collegano tutti gli altri bus della scheda madre, come i bus PCI e AGP.

Questi bus secondari, di solito, hanno una velocità che dipende dal FSB e non sono necessariamente sincroni ad esso.

Infine, l'**LPC** (Low Pin Count) **bus** collega le periferiche a bassa velocità e legacy (tradizionali).



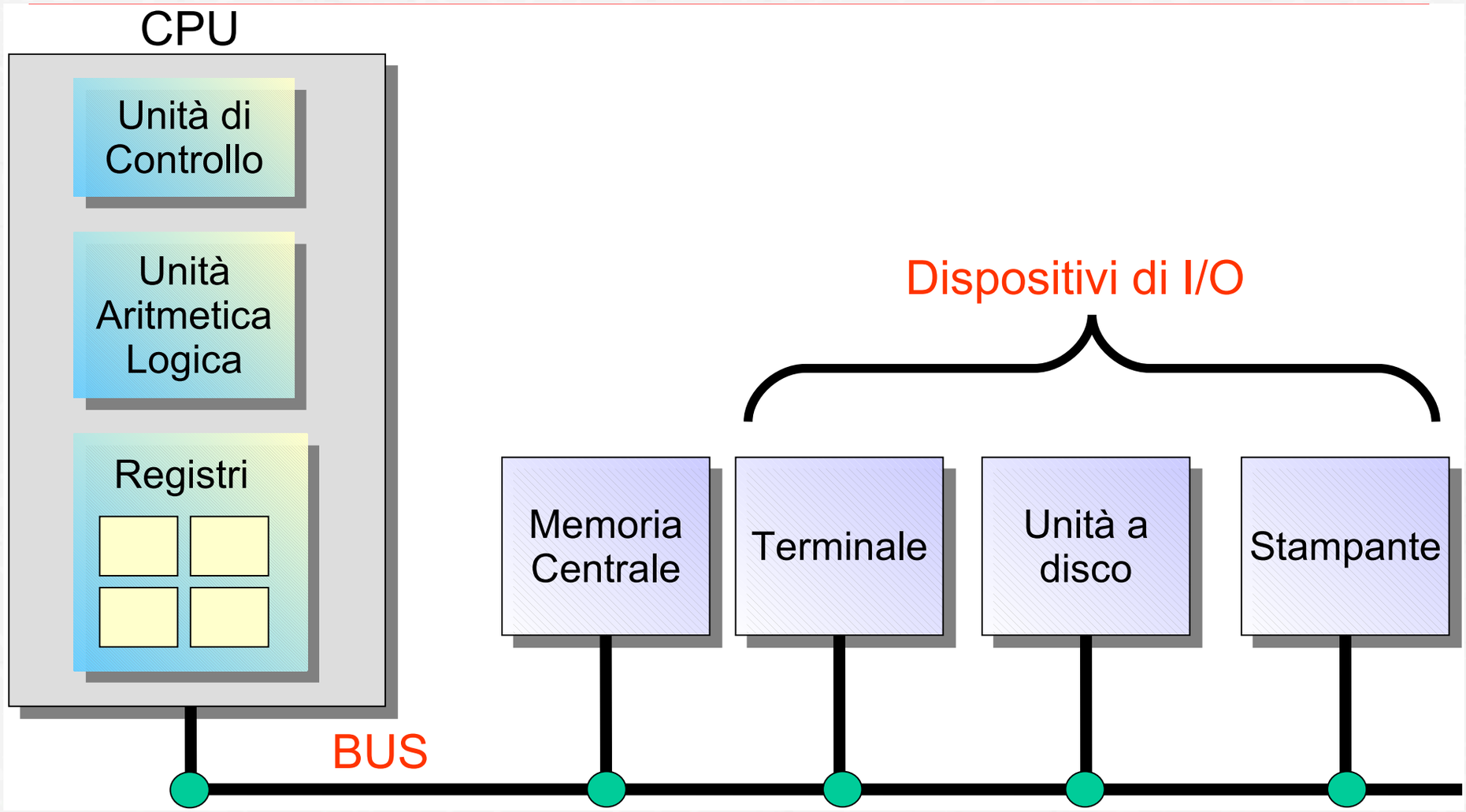
Bus: evoluzione

- I primi PC “IBM compatibili” degli anni ’80 erano equipaggiati con il bus **ISA** a 8 bit, poi evoluto nel bus **AT** a 16 bit e nell’**EISA** a 32/16 bit.
- L’IBM introdusse alla fine degli anni ’80 il bus proprietario **MCA** a 32 bit, retrocompatibile 16 bit.
- Il bus **AGP** (Accelerated Graphics Port) è un bus locale (cioè connesso direttamente alla CPU) come il suo predecessore **VESA**. Fu introdotto nel 1997 per le connessioni video ad alta velocità, e sostituito nel 2004 dal bus PCI.
- Standard attuale: **PCI** (Peripheral Component Interconnect) **Express** (bus seriale; sino a 8 Gb/s per canale nella versione 3.0).

Il Processore (CPU)

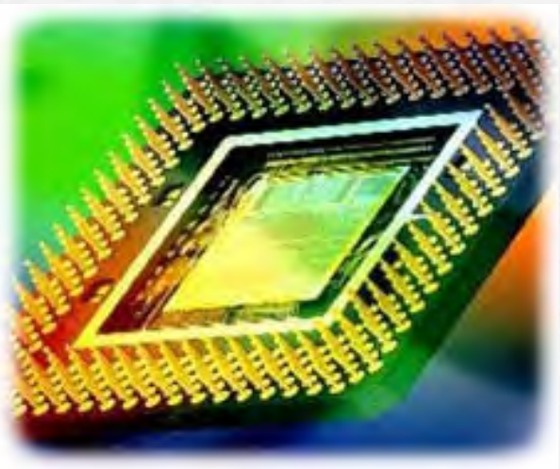
Architettura del processore

Organizzazione bus-oriented



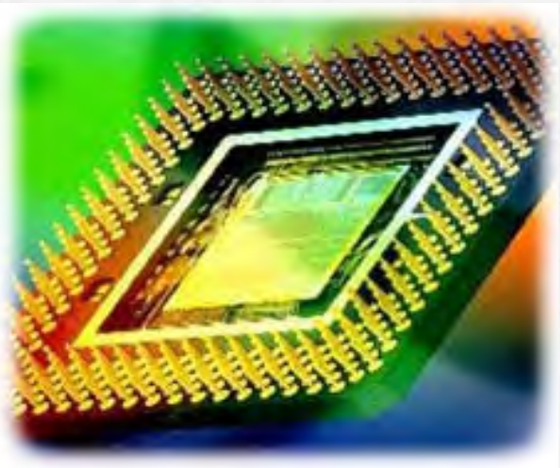
Elementi della CPU

- Unità di controllo
  - Legge le istruzioni dalla memoria e ne determina il tipo
- Unità aritmetico-logica (ALU)
  - Esegue le operazioni necessarie per eseguire le istruzioni
- Registri
  - Memorie ad alta velocità usate per risultati temporanei
  - Determinano il parallelismo (*pipelining*) della CPU
  - Esistono registri generici e registri specifici:
    - ✓Program Counter (PC), Instruction Register (IR), ...
- Centinaia di milioni di transistor nelle moderne CPU per PC

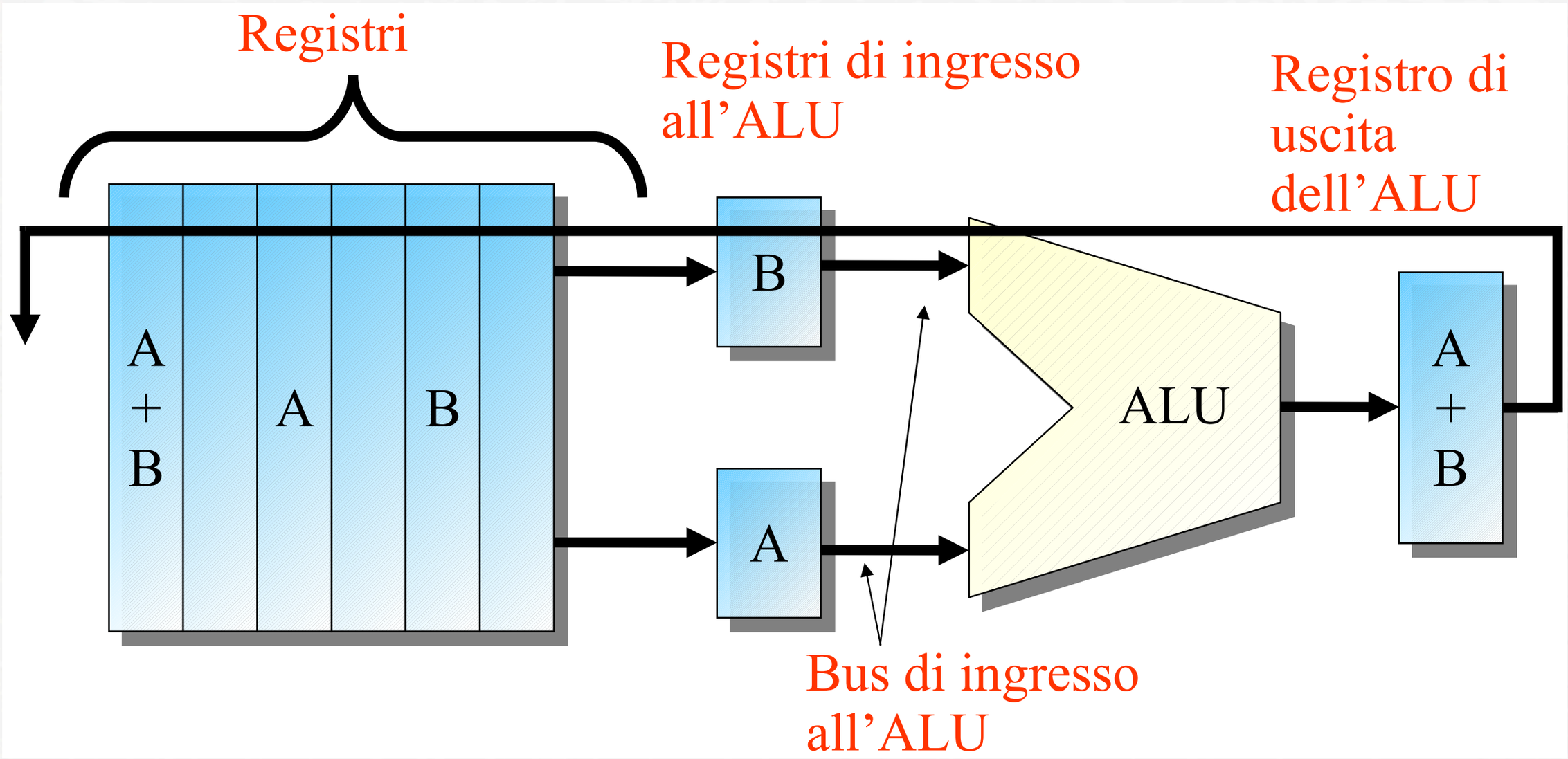


Tipologie di istruzioni macchina

- 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 informazioni
  - Trasferimento dati e istruzioni tra CPU e memoria
  - Trasferimento dati e istruzioni tra CPU e dispositivi di I/O



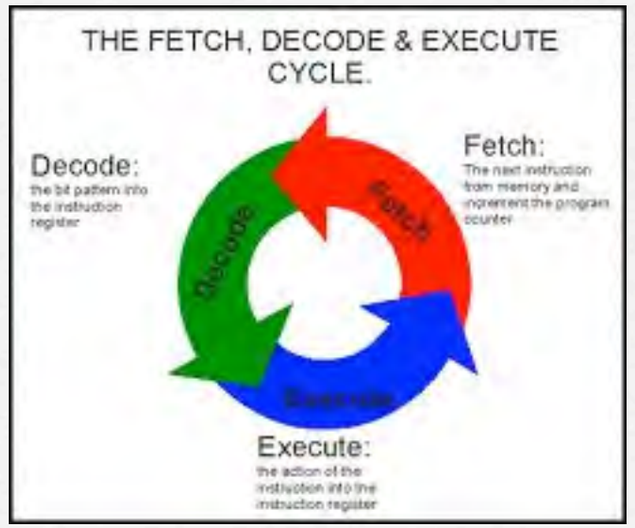
Struttura del "data path"



Esecuzione delle istruzioni

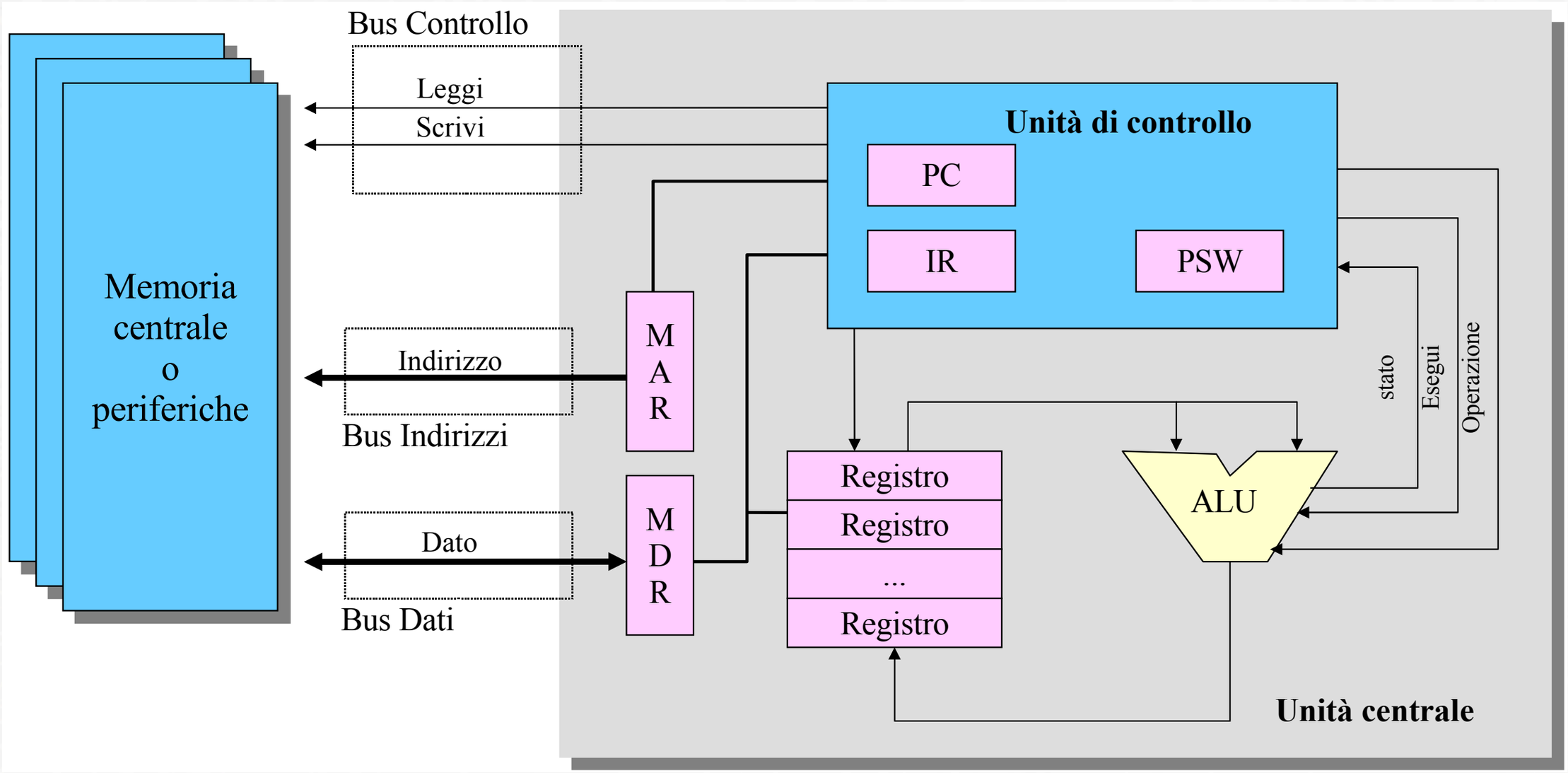
Ciclo **Fetch-Decode-Execute**

- Prendi l'istruzione corrente dalla memoria e mettila nel registro istruzioni (**IR**) [**Fetch**]
- Incrementa il program counter (**PC**) in modo che contenga l'indirizzo dell'istruzione successiva(*)
- Determina il tipo dell'istruzione corrente [**Decode**]
- Se l'istruzione usa una parola in memoria determina dove si trova
- Carica la parola, se necessario, in un registro della CPU
- Esegui l'istruzione [**Execute**]
- Riprendi dal punto iniziale



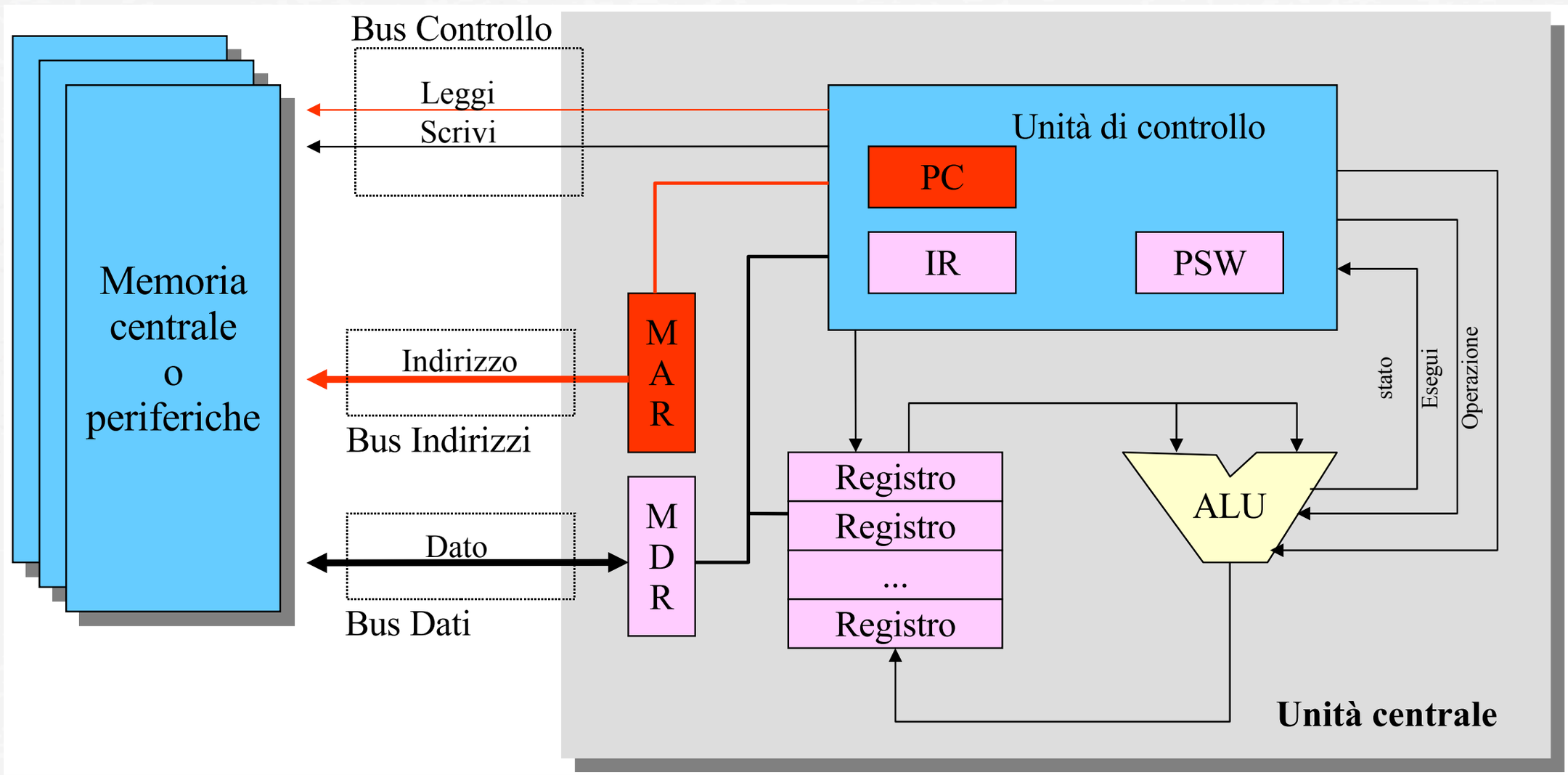
(*) *Va eseguita prima dell'Execute, altrimenti in caso di istruzione JMP l'indirizzo sarebbe alterato...*

Struttura semplificata della CPU



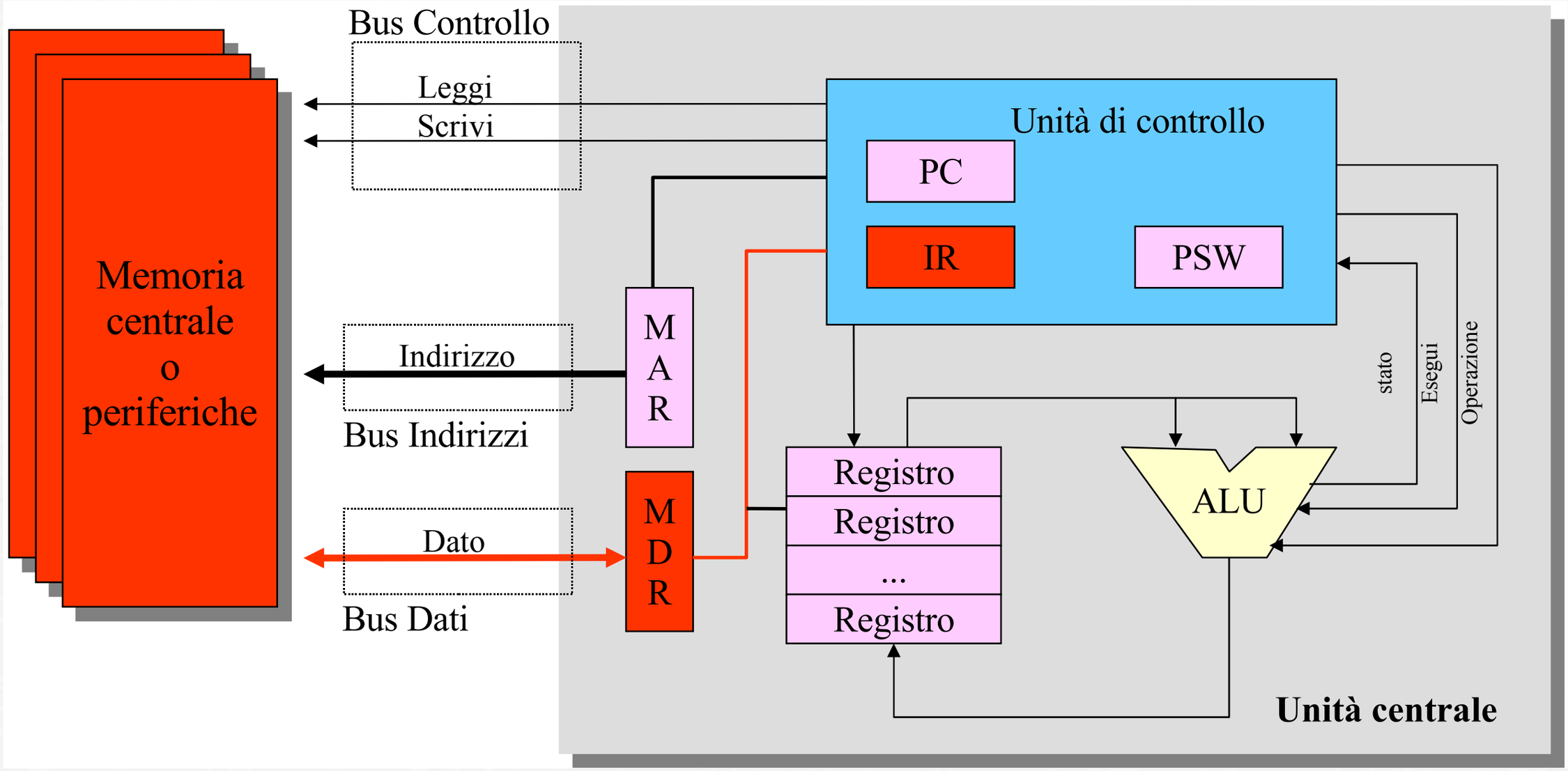
Esempio: lettura dalla memoria

Fase di Fetch (1 di 2)



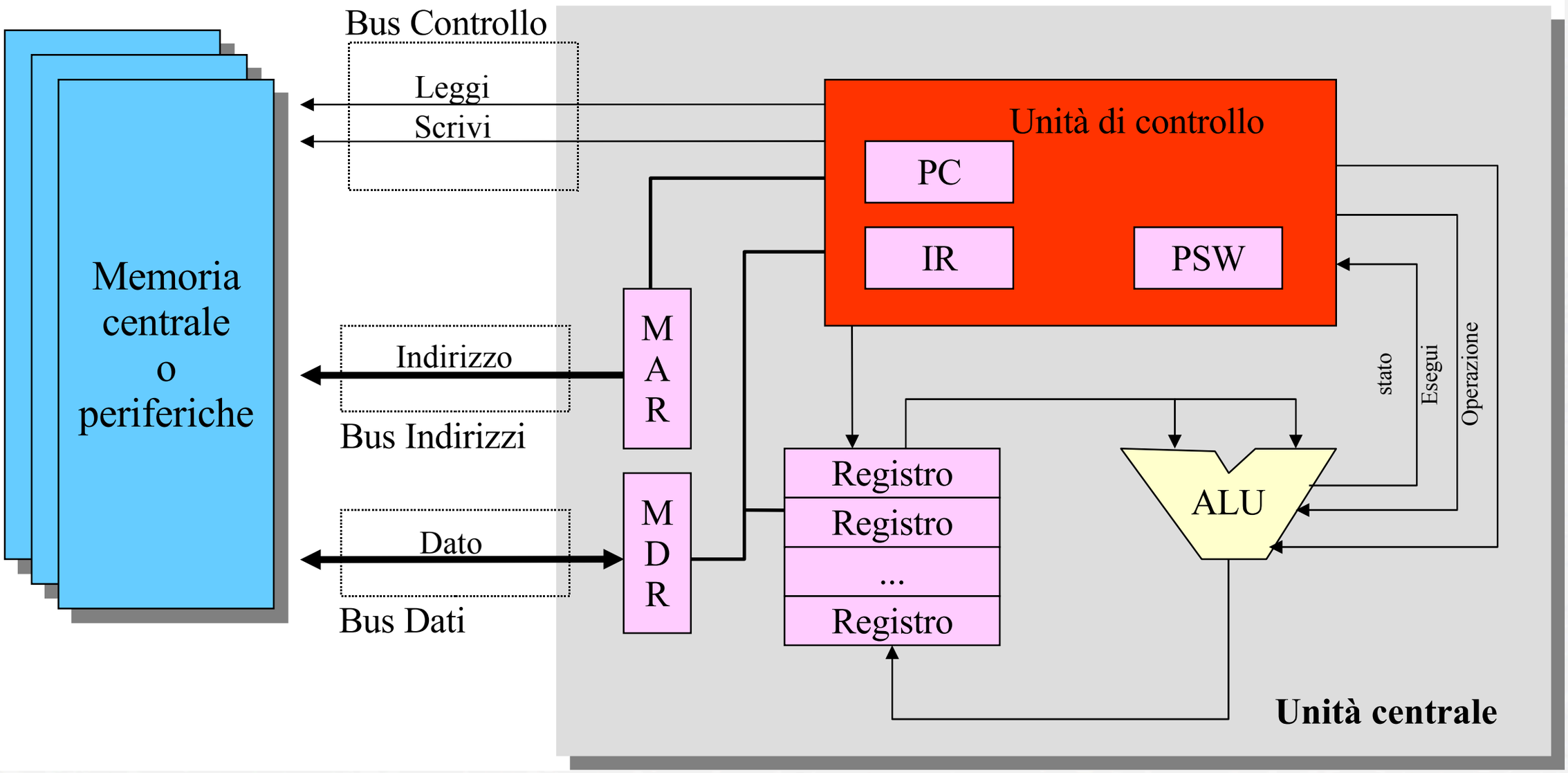
Esempio: lettura dalla memoria

Fase di Fetch (2 di 2)



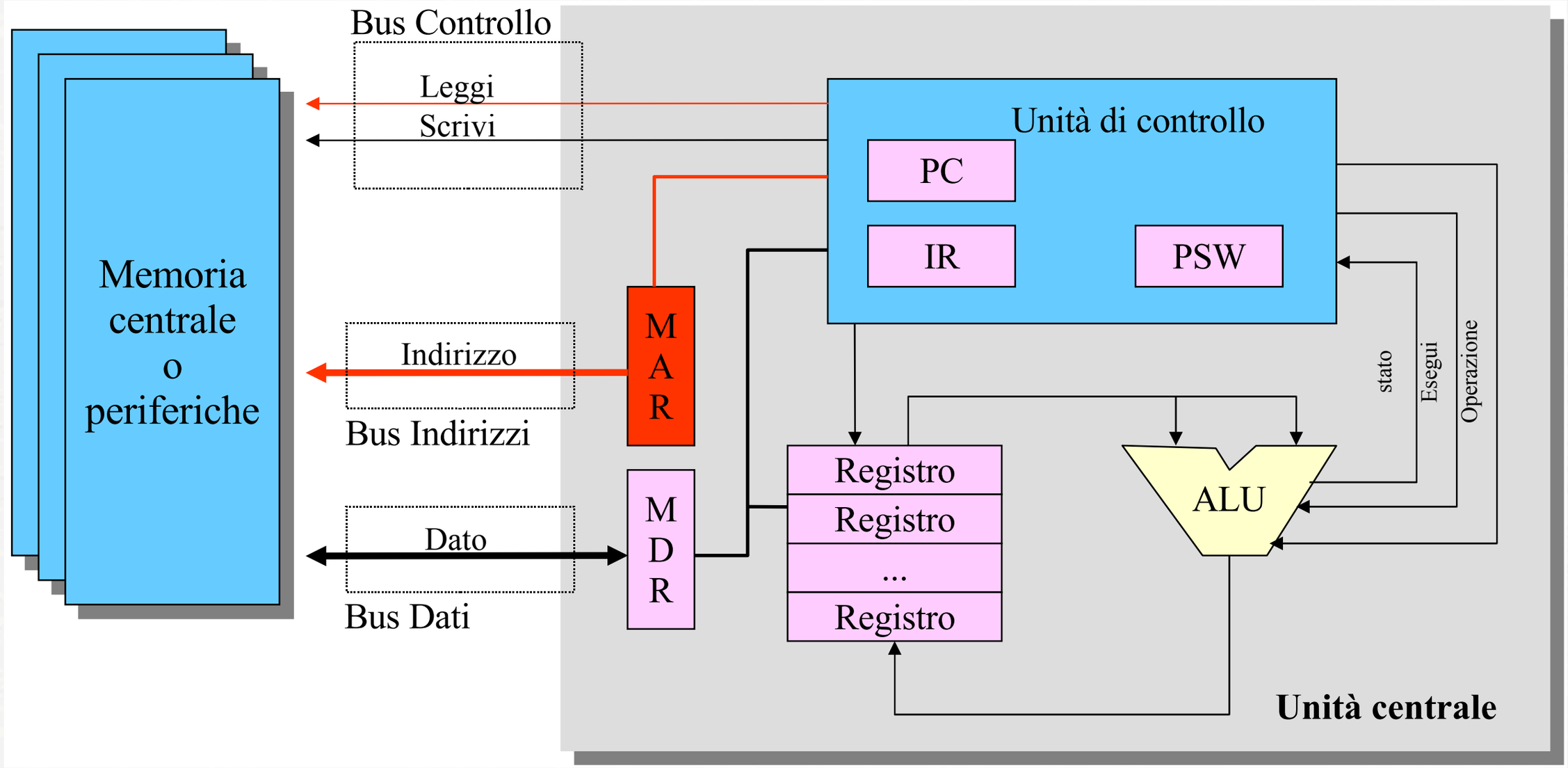
Esempio: lettura dalla memoria

Decodifica



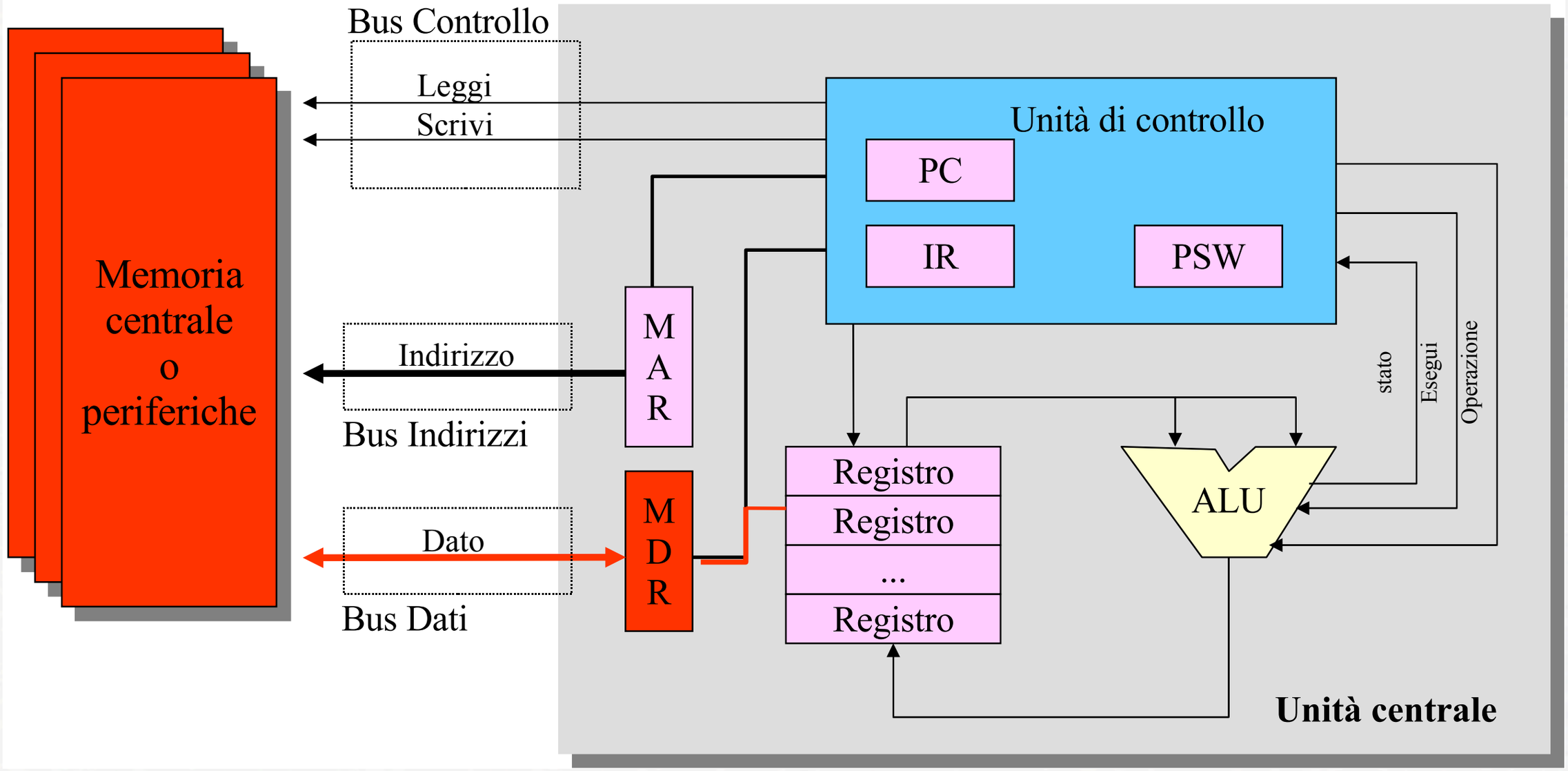
Esempio: lettura dalla memoria

Esecuzione (1 di 2)



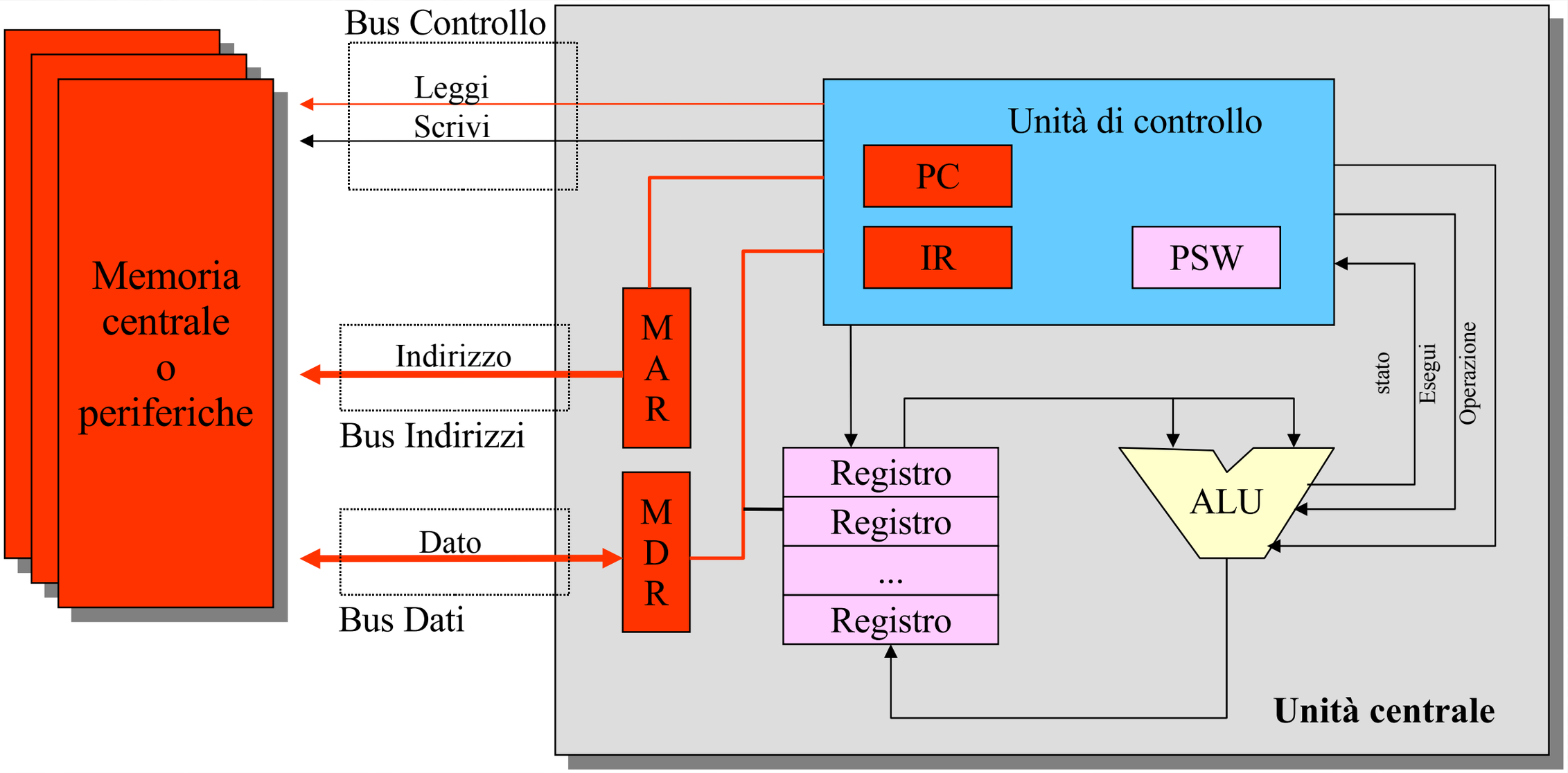
Esempio: lettura dalla memoria

Esecuzione (2 di 2)



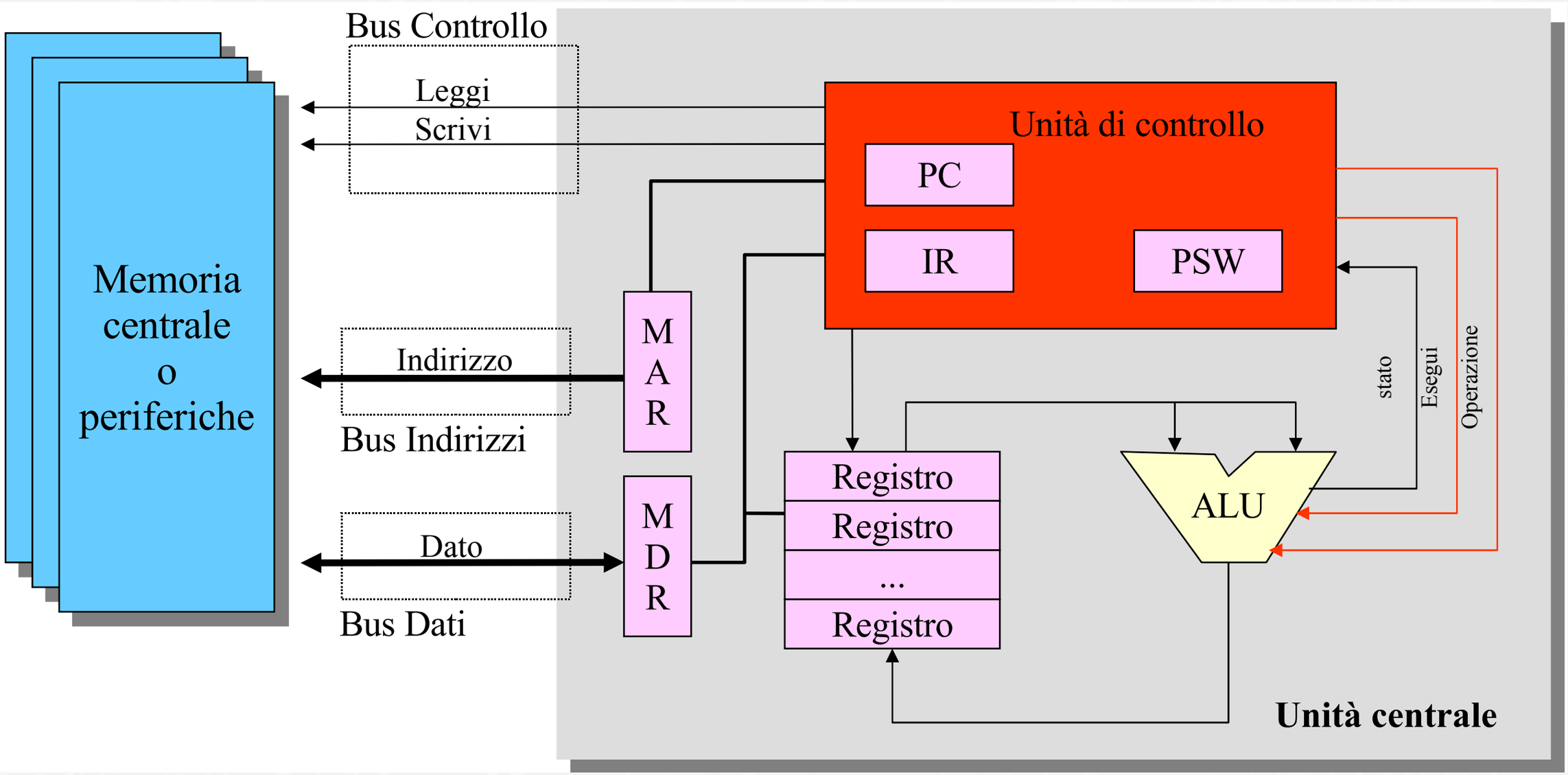
Esempio: somma tra due registri

Fetch (...)



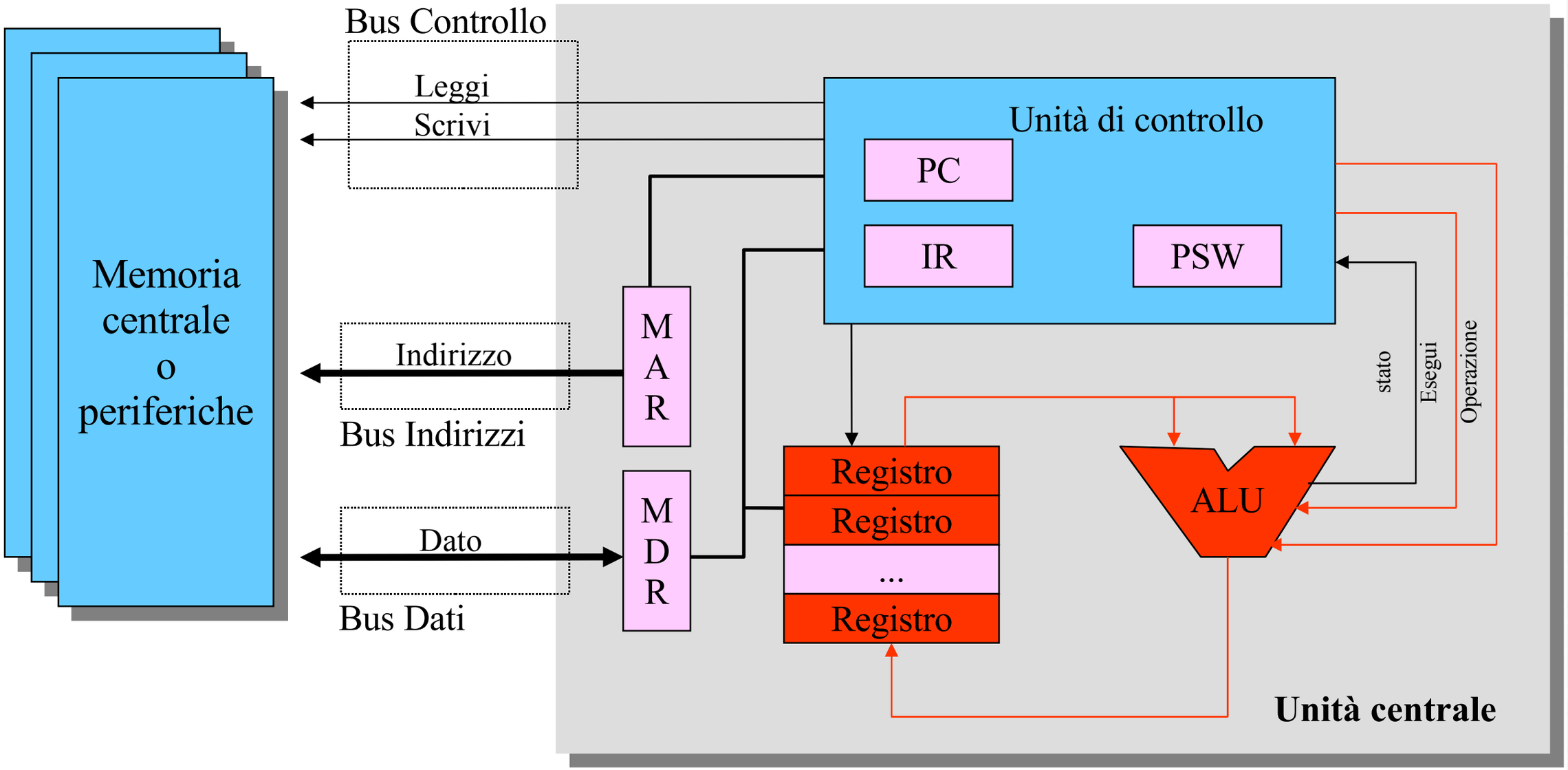
Esempio: somma tra due registri

Decodifica



Esempio: somma tra due registri

Esecuzione



Riepilogo registri CPU

IR: Usato per contenere l'istruzione in corso di esecuzione

- Caricato in fase di fetch
- Determina le azioni svolte durante la fase di esecuzione

PC: Tiene traccia dell'esecuzione del programma

- Contiene l'indirizzo di memoria in cui è memorizzata la prossima istruzione da eseguire

MAR: Contiene l'indirizzo della locazione di memoria da leggere o scrivere

- La dimensione di MAR determina l'ampiezza dello spazio di memoria fisica
- Dalla fine degli anni '80 vengono prodotti microprocessori con bus indirizzi a 32 bit

MDR: Registro attraverso il quale viene scambiata l'informazione tra la memoria e la CPU

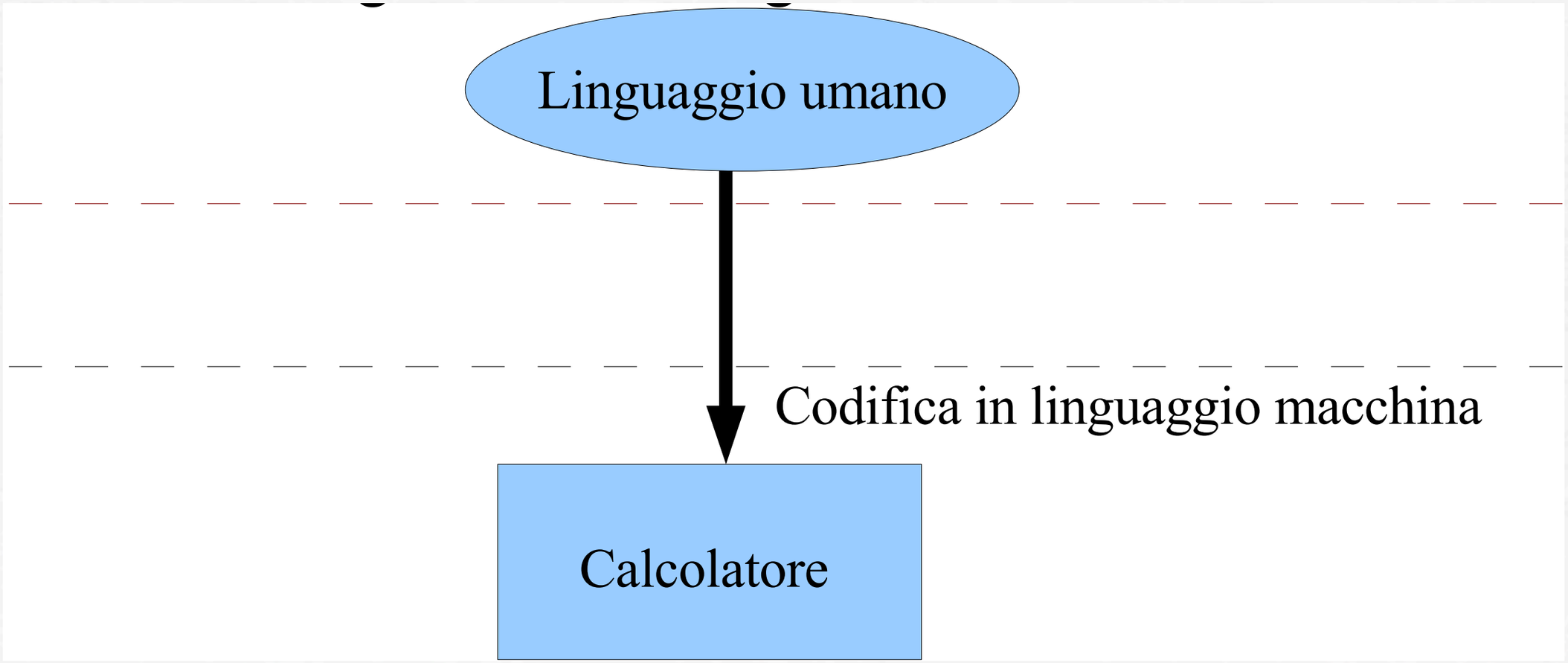
- Tradizionalmente la dimensione di MDR dà la misura del grado di parallelismo della macchina (8, 16, 32, 64 bit)

PSW: (Program Status Word) contiene informazioni sullo stato di esecuzione del programma

R0, R1, ..., Rn: Registri di uso generale

Distanza uomo-macchina

Quanto devono essere complesse le istruzioni che una CPU è in grado di eseguire?



Approccio CISC

Complex Instruction Set Computing

Un repertorio di istruzioni esteso è preferibile perché:

- ✓ Istruzioni potenti semplificano la programmazione
- ✓ Riduce il gap tra linguaggio di macchina e linguaggio di alto livello
- ✓ L'uso efficiente della memoria (all'epoca era costosa) era la preoccupazione principale:
  - ➡ Meglio avere codici compatti

Approccio RISC

Reduced Instruction Set Computing

Memorie più veloci ed economiche

➡ Posso anche mettere un numero maggiore di istruzioni, però più semplici.

Comportamento dei programmi

➡ L'80% delle istruzioni eseguite corrispondeva al solo 20% del repertorio

➡ Conviene investire nella riduzione dei tempi di esecuzione di quel 20%, anziché aggiungere raffinate istruzioni, quasi mai usate, ma responsabili dell'allungamento del tempo di ciclo di macchina

Conviene costruire processori molto veloci (alta frequenza di clock) con ridotto repertorio (set) di istruzioni macchina.

RISC: criteri di progettazione

Frequenza di clock

- Velocità con cui gli istanti di tempo si succedono all'interno della CPU.
  Si misura in Hz, che significa “volte al secondo”: 2Hz = 2 volte al secondo.
- Periodo di clock: intervallo di tempo tra un istante ed il successivo.
  E' l'inverso della frequenza di clock.
- Il periodo di clock deve essere sufficientemente “lungo” da consentire a tutti i segnali elettrici di arrivare.

Le istruzioni devono essere semplici

- Se l'introduzione di una operazione di macchina fa crescere del 10% il periodo di clock, allora essa deve produrre una riduzione di almeno un 10% del numero totale di cicli eseguiti

RISC: criteri di progettazione

Tutte le istruzioni occupano lo stesso spazio di memoria (una parola)

Ristretto numero di formati

- La codifica “ordinata” consente accorgimenti per velocizzare l'esecuzione (pipeline), difficilmente applicabili a repertori di istruzioni complesse

La semplificazione del repertorio tende a far aumentare la dimensione del codice

- Non è un problema, vista la tendenza alla riduzione dei costi e all'aumento della densità delle memorie
- Dal punto di vista della velocità i guadagni che si ottengono nel semplificare le istruzioni sono superiori all'effetto negativo del maggior numero di istruzioni per programma

La memoria principale

Organizzazione della memoria

La memoria principale è organizzata come un insieme di locazioni di uguale dimensione, ognuna delle quali è identificata tramite un numero progressivo ad essa associato, detto **indirizzo**, che rappresenta la posizione di quella locazione rispetto alla prima.

- Il *contenuto* delle locazioni non è immediatamente riconoscibile: non c'è distinzione esplicita tra istruzioni e dati e tra dati di tipo diverso.
- Una istruzione o un dato possono risiedere su più *locazioni consecutive*, se la dimensione della singola locazione di memoria non è sufficiente.
- Il parallelismo di accesso è definito dall'*ampiezza* della locazione.

0	**01101101**
1	**10010110**
2	**00111010**
3	**11111101**
.	.
.	.
.	.
1022	**00010001**
1023	**10101001**

Organizzazione della memoria

Operazioni sulla memoria principale

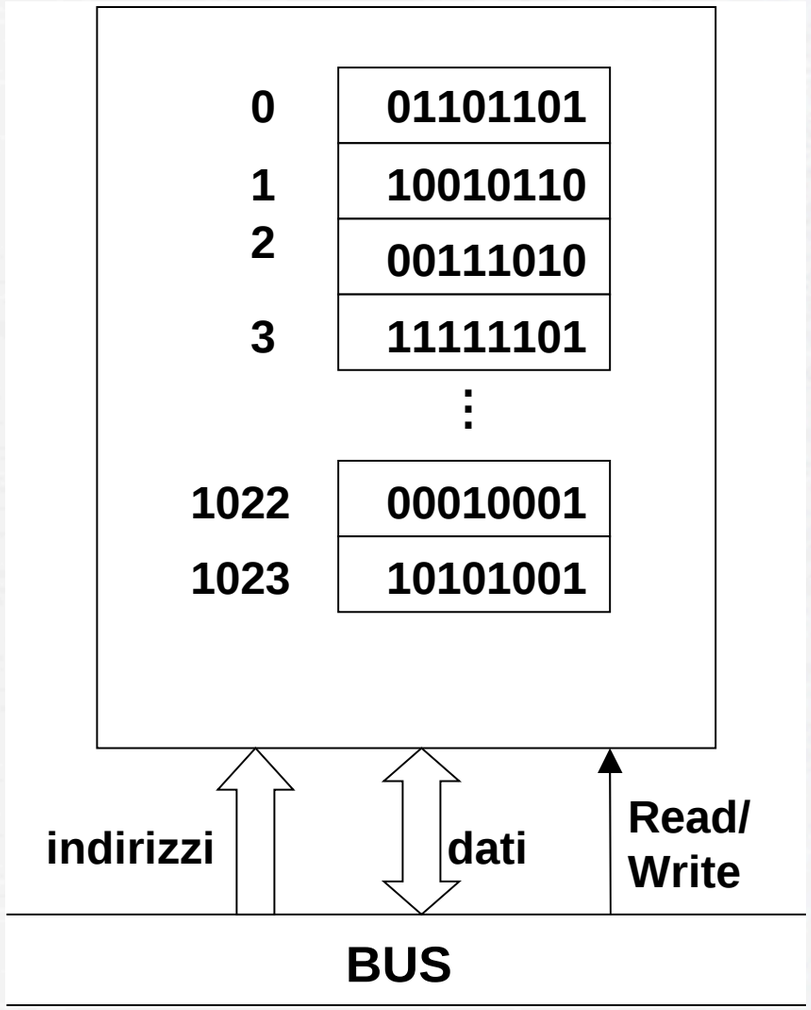
Il modulo di memoria principale è connesso al resto del sistema tramite il BUS.

In particolare, sono presenti tre gruppi di linee:

- linee **indirizzi**
- linee **dati**
- linee **read/write**

In ogni operazione è quindi necessario specificare:

- su quale locazione si intende compiere l'operazione → *indirizzo*
- che tipo di operazione si intende realizzare → *Read/Write*
- in caso di scrittura, quale sia il *valore* da memorizzare



Organizzazione della memoria

Codifica degli indirizzi

- Se la memoria contiene N registri (locazioni) abbiamo bisogno di N indirizzi diversi.
- Di quanti bit ho bisogno per rappresentare N indirizzi diversi? Risposta: $\lceil \mathbf{Log_2\ N} \rceil$

Esempio:

1024 registri	→	**10 bit** ($Log_2\ 1024 = 10$, $2^{10} = 1024$)
65536 registri	→	**16 bit** ($Log_2\ 65536 = 16$, $2^{16} = 65536$)
⋮		
4.294.967.296 registri (4 Giga)	→	**32 bit**

Parametri della memoria principale

Capacità

Fornisce una misura della quantità di informazione che è possibile memorizzare.

Questa dipende dall'ampiezza delle singole locazioni e dal numero di locazioni contenute.

La capacità delle memoria si misura in termini di byte (1 **Megabyte** = 2^{20} byte, 1 **Gigabyte** = 2^{30} byte, 1 **Terabyte** = 2^{40} byte).

Tempo di accesso

E' il tempo minimo che intercorre tra due operazioni (accessi) in memoria.

Dipende dalla tecnologia di realizzazione della memoria.

Si misura in termini di secondi (1 **nanosecondo** = 10^{-9} secondi).

Tipologie di memorie

Memorie RAM

RAM è l'acronimo di *Random Access Memory*.

Sta ad indicare che il tempo di accesso è costante per ogni locazione di memoria.

Hanno le seguenti caratteristiche:

- si possono realizzare operazioni sia di lettura che di scrittura;
- mantengono il loro contenuto finché è presente l'alimentazione (sono dette ***memorie volatili***).

Tipologie di memorie

Memorie ROM

ROM è l'acronimo di *Read Only Memory*.

Sta ad indicare che il suo contenuto è inserito una volta per sempre all'atto della costruzione e non può più essere modificato o cancellato (a meno di particolari procedimenti — PROM, EPROM, EEPROM).

Hanno le seguenti caratteristiche:

- sono ***permanenti*** (NON volatili);
- anche in questo caso il tempo di accesso è costante.

Le memorie RAM

Esistono due tipi di memoria RAM:

RAM dinamica o **DRAM** (Dynamic Random Access Memory)

- Alta densità di integrazione, economica, lenta, bassa potenza di alimentazione
- *Dynamic*: è necessario rigenerare i contenuti periodicamente (refresh)

RAM statica o **SRAM** (Static Random Access Memory)

- Bassa densità di integrazione, costosa, veloce, alta potenza di alimentazione
- *Static*: il contenuto viene mantenuto finché è presente l'alimentazione

Le memorie *RAM ECC* (a correzione di errore) si utilizzano principalmente sui server e sui grossi calcolatori.

Il packaging delle memorie

- Fino all'inizio degli anni '90 le memorie venivano realizzate su chip singoli.
- Oggi si monta un gruppo di chip, tipicamente 8 o 16, su una piccola scheda stampata.
- Si parla di:
  - **SIMM** (Single In line Memory Module): la fila di connettori si trova da un solo lato della scheda
  - **DIMM** (Dual In line Memory Module): i connettori si trovano su ambedue i lati della scheda

Le memorie ROM

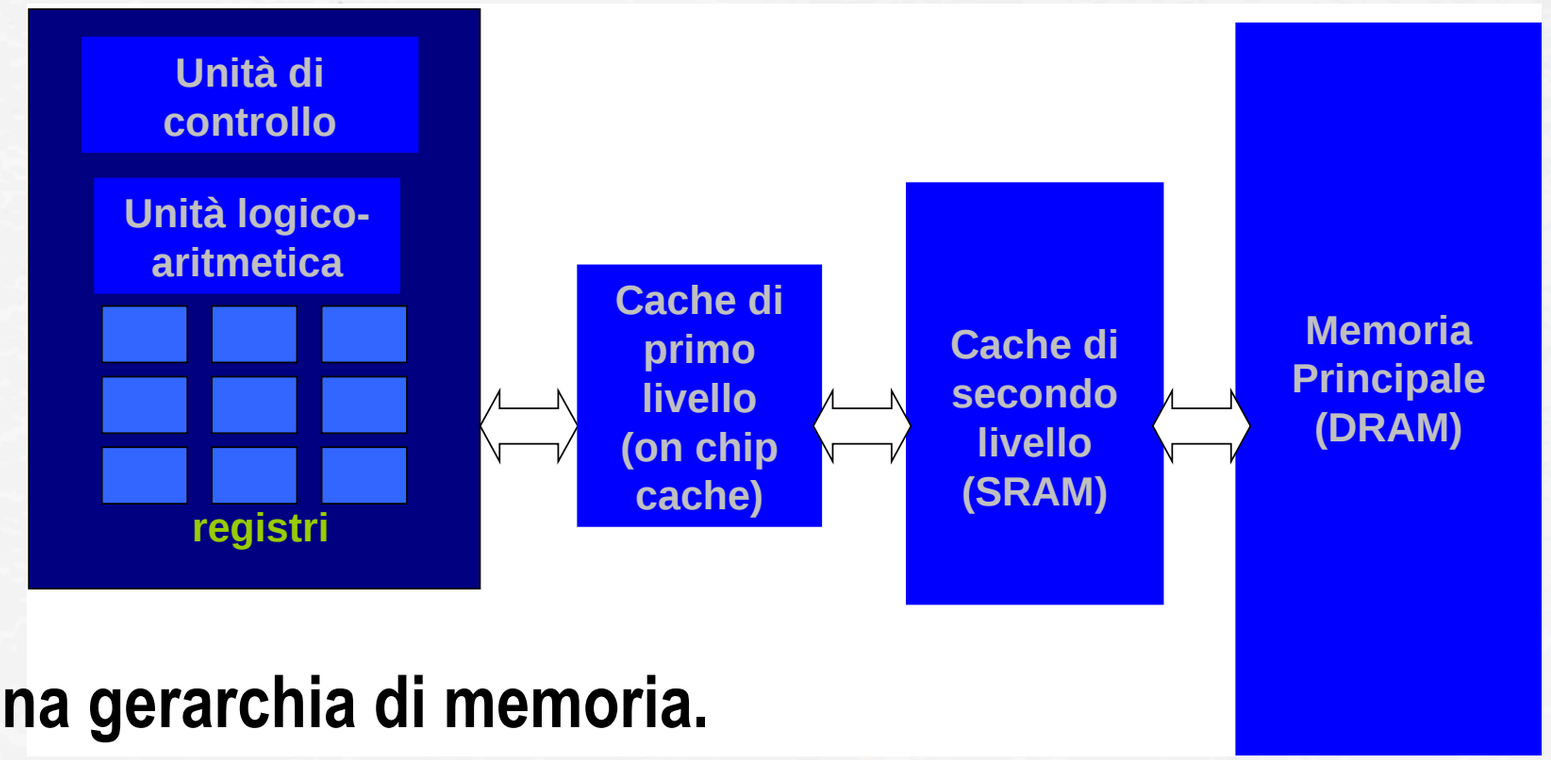
- Le memorie ROM vengono usate all'interno del calcolatore per memorizzare i programmi per l'avvio all'accensione (***bootstrap***), i quali devono rimanere memorizzati anche quando l'alimentazione viene a mancare. Questi sono, inoltre, programmi e dati che, una volta memorizzati, non devono essere più modificati.
- In generale questo tipo di memorie si usa per memorizzare il **firmware**, programmi e dati che sono memorizzati in maniera permanente su un qualunque dispositivo:
  - Cellulari
  - Lettore/masterizzatori DVD
  - Navigatori

- Allo stato attuale si usano si usano le ROM programmabili, le EEPROM (Electrically Erasable Programmable Read-Only Memory), dette anche memorie **FLASH**.

Organizzazione di un sistema di memoria

- Requisiti ideali di un sistema di memoria: **capacità** infinita, **velocità** infinita.
- Ma la situazione reale è:
  → le memorie capienti ed economiche (DRAM) sono lente
  → le memorie veloci (SRAM) sono costose e meno integrabili

Come realizzare un sistema di memoria che sia capiente, economico e veloce?

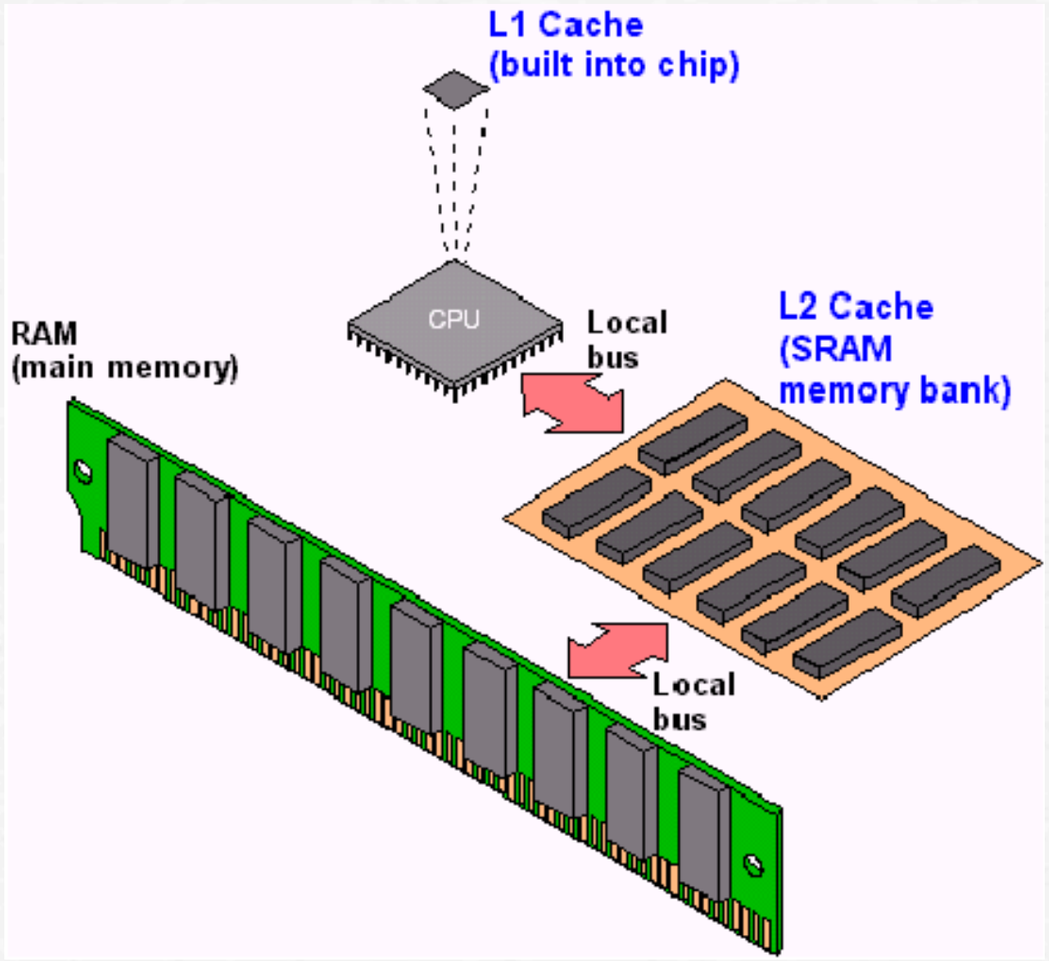


Un sistema basato su una gerarchia di memoria.

Realizzazione di un sistema di memoria

- Il sistema di memoria è composto da moduli di memoria con caratteristiche diverse e organizzati a livelli.
- Tra CPU e memoria principale viene posto un modulo di memoria intermedio (cache), ad accesso veloce, ma di capienza limitata.
- I dati memorizzati sono distribuiti sui vari moduli e possono essere trasferiti tra moduli adiacenti.

La distribuzione è realizzata in maniera da cercare di memorizzare i dati e le istruzioni richiesti più frequentemente nella cache, in modo che la CPU possa accedervi velocemente.



- Il tempo di propagazione del segnale (è un vincolo per il tempo di accesso) è minore.
- Una memoria grande ha bisogno di indirizzi grandi (maggiori tempi di decodifica).
- Il miglioramento delle prestazioni dovuto alla memoria cache si basa sul **principio di località del riferimento:**

  i dati usati più di recente saranno utilizzati ancora nel recente, cioè nel prossimo futuro.

Funzionamento della cache

L'**algoritmo** seguito per la cache è il seguente:

1. Il dato viene cercato prima nella cache.
2. Se è presente abbiamo finito (*cache hit*).
3. Se non è presente, si legge in RAM e si mette una copia nella cache (*cache miss*).
4. Se non c'è spazio di solito si sovrascrivono i dati utilizzati meno di recente (strategia **LRU** *Least Recently Used*).
5. Per le scritture generalmente si scrive la RAM e si aggiorna la copia, se c'è.