UNIVERSITA' DEGLI STUDI DI FOGGIA
DIPARTIMENTI
DI AREA MEDICA

CdLS in Odontoiatria e Protesi Dentarie

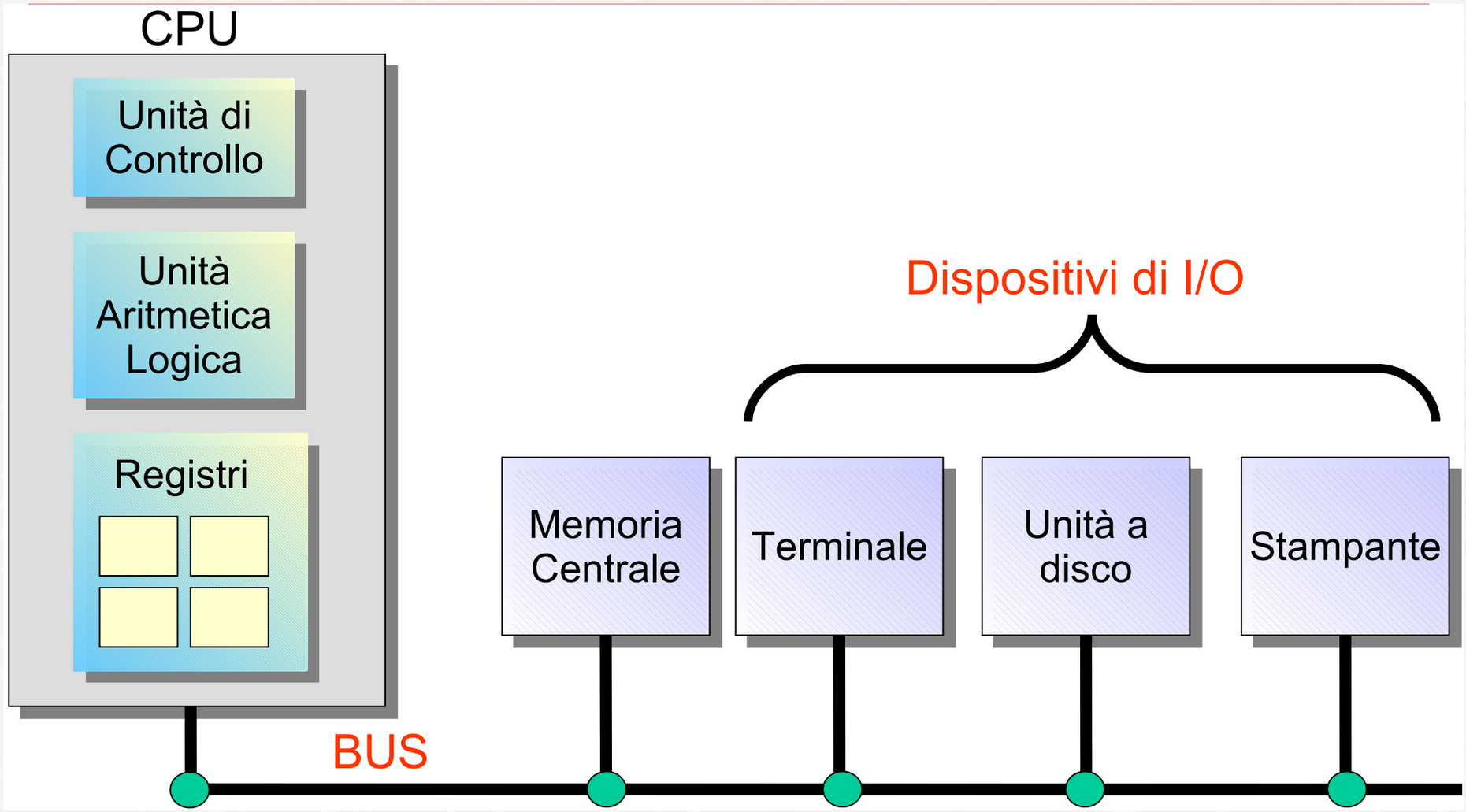
Corso di Informatica

Prof. Crescenzio Gallo
crescenzio.gallo@unifg.it

Il Processore (CPU)

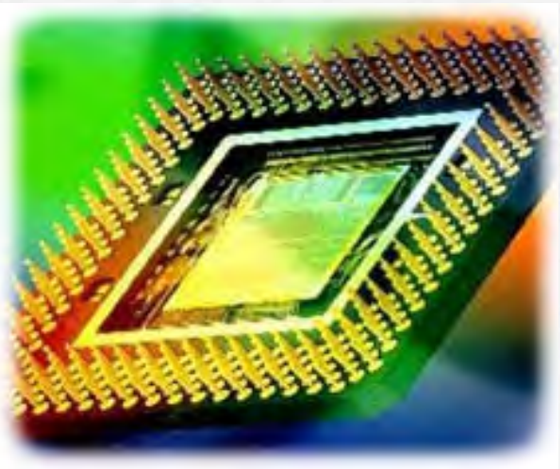
Architettura del processore

Organizzazione bus-oriented



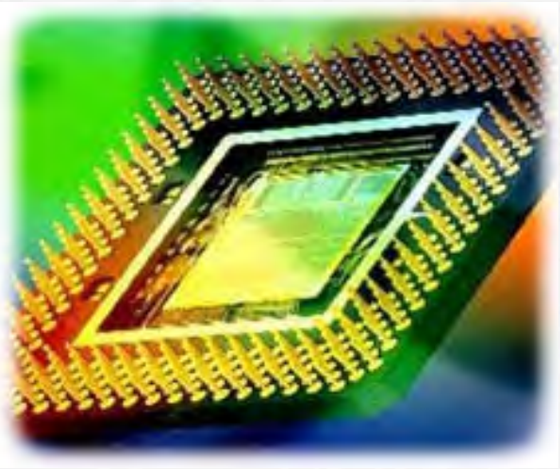
Elementi della CPU

- 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

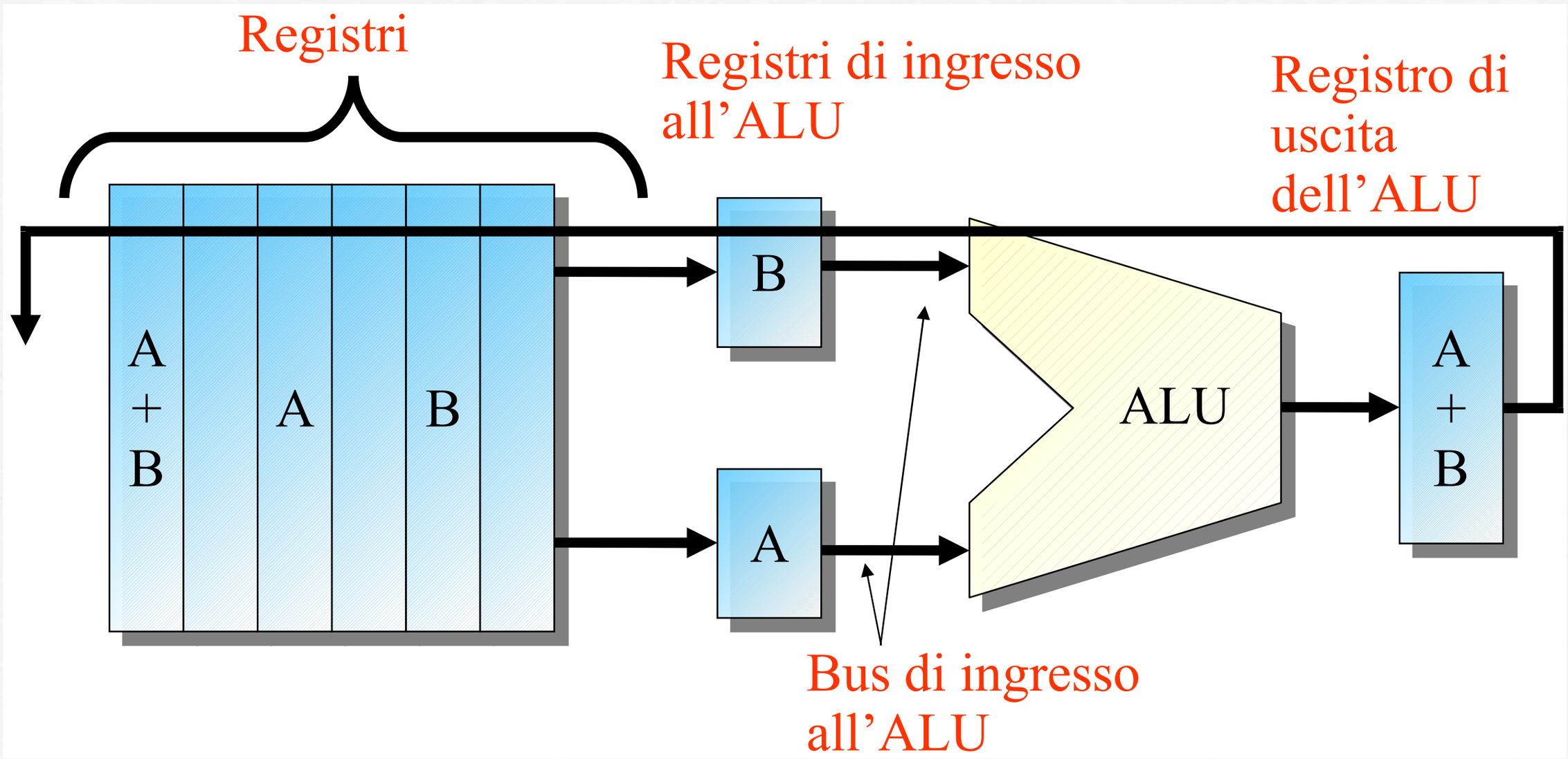


Tipologie di istruzioni macchina

- 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



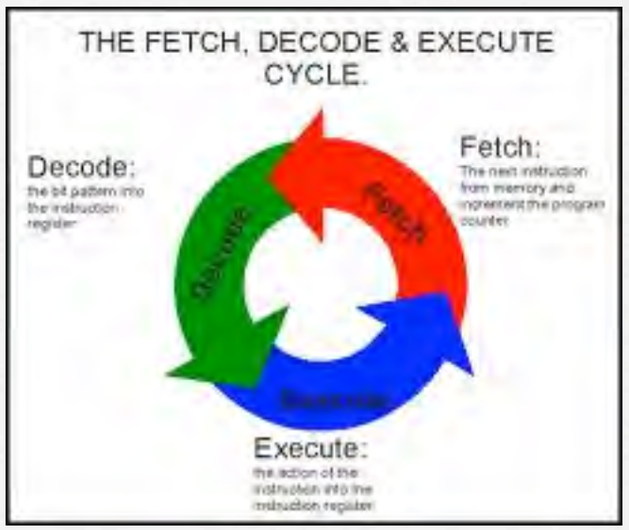
Struttura del "data path"



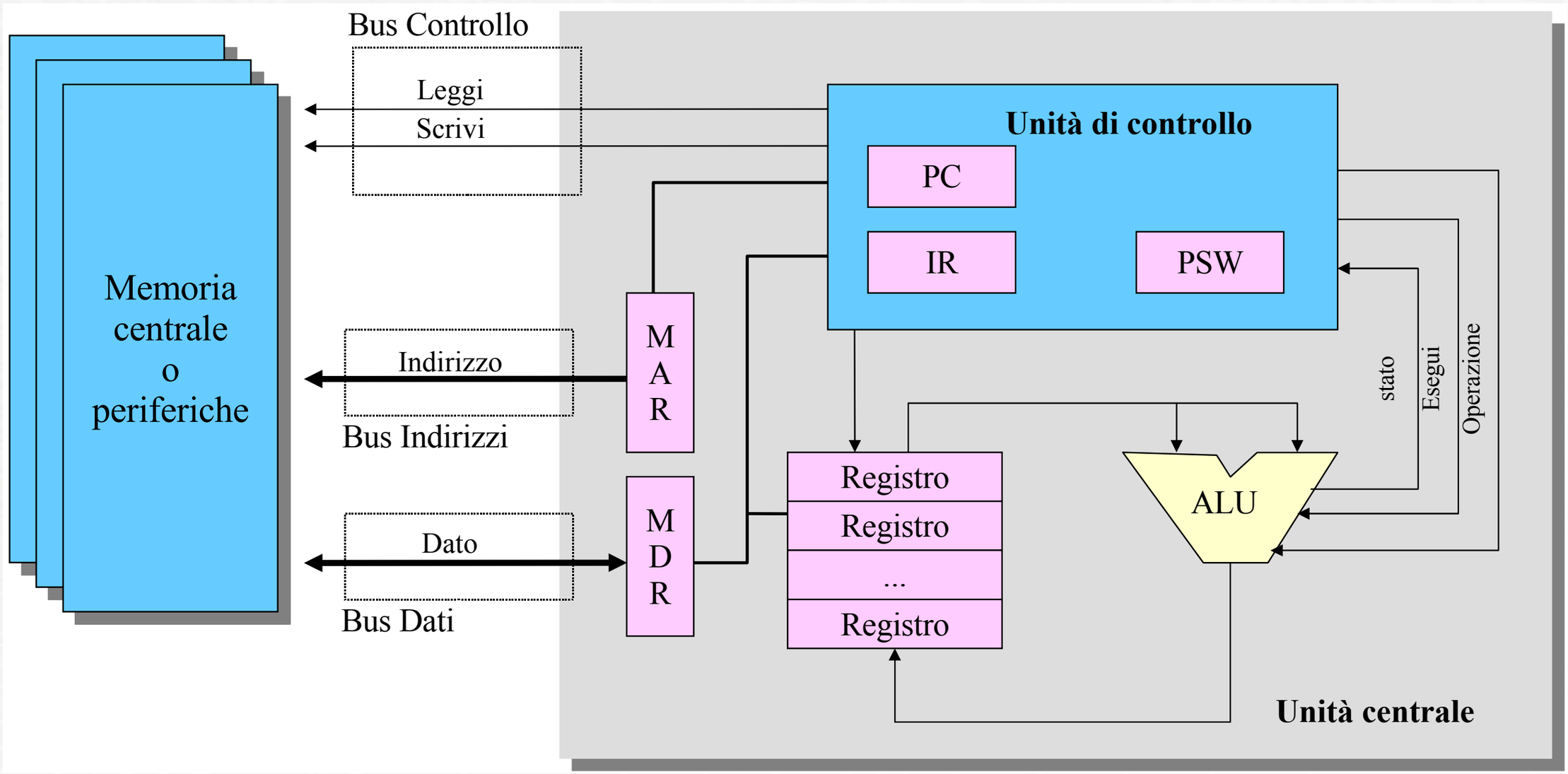
Esecuzione delle istruzioni

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

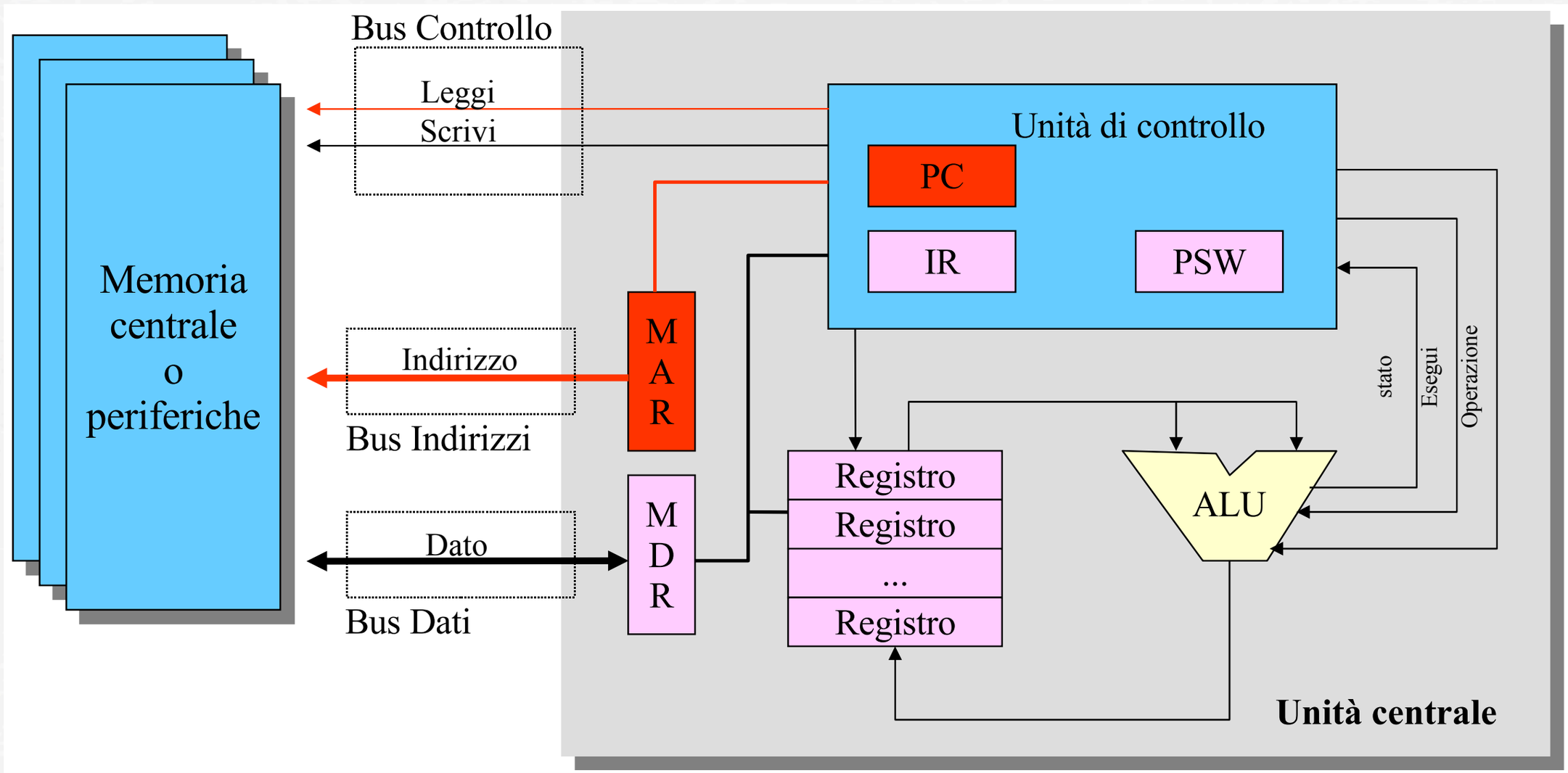


Struttura semplificata della CPU



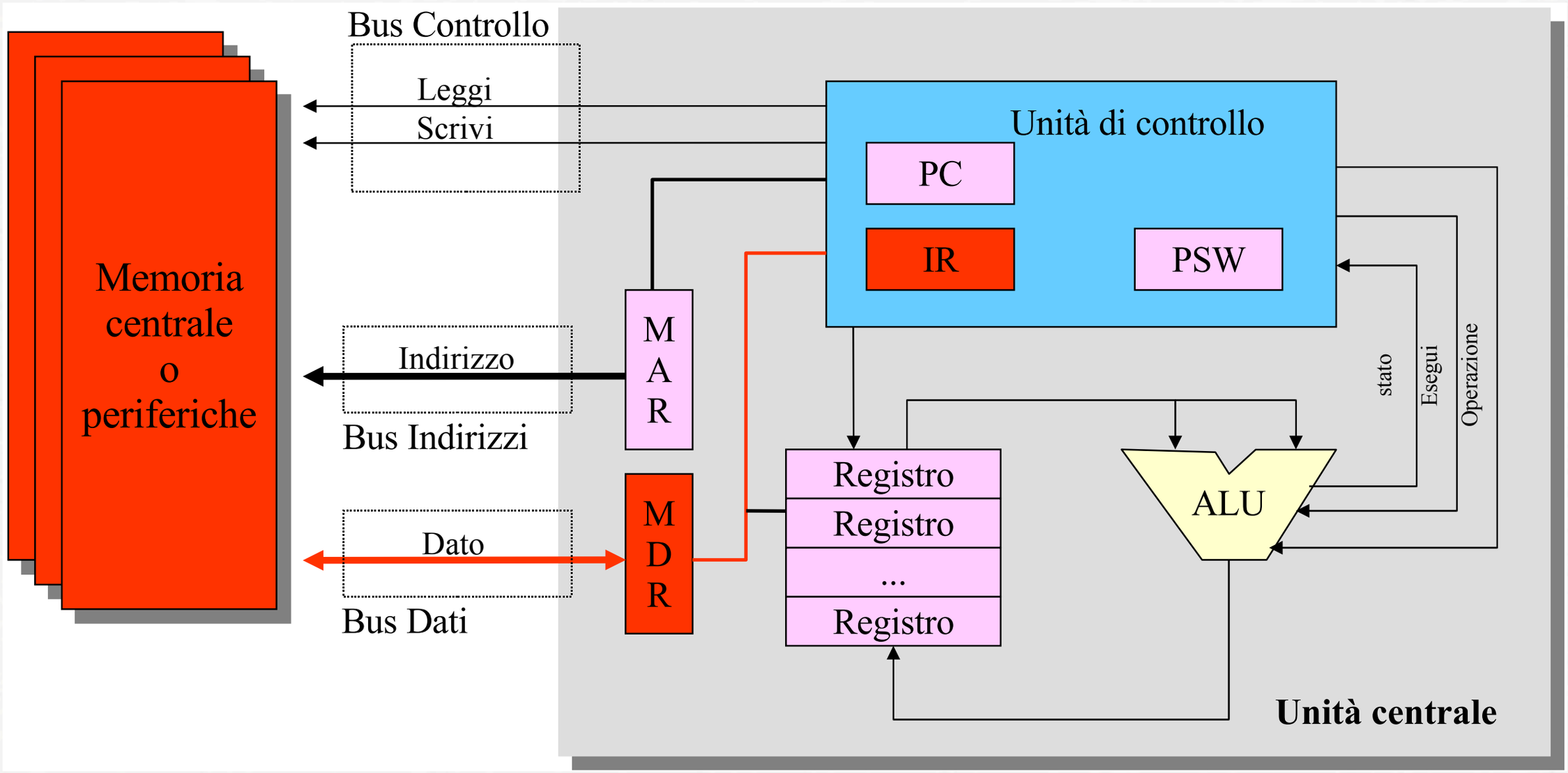
Esempio: lettura dalla memoria

Fase di Fetch (1 di 2)



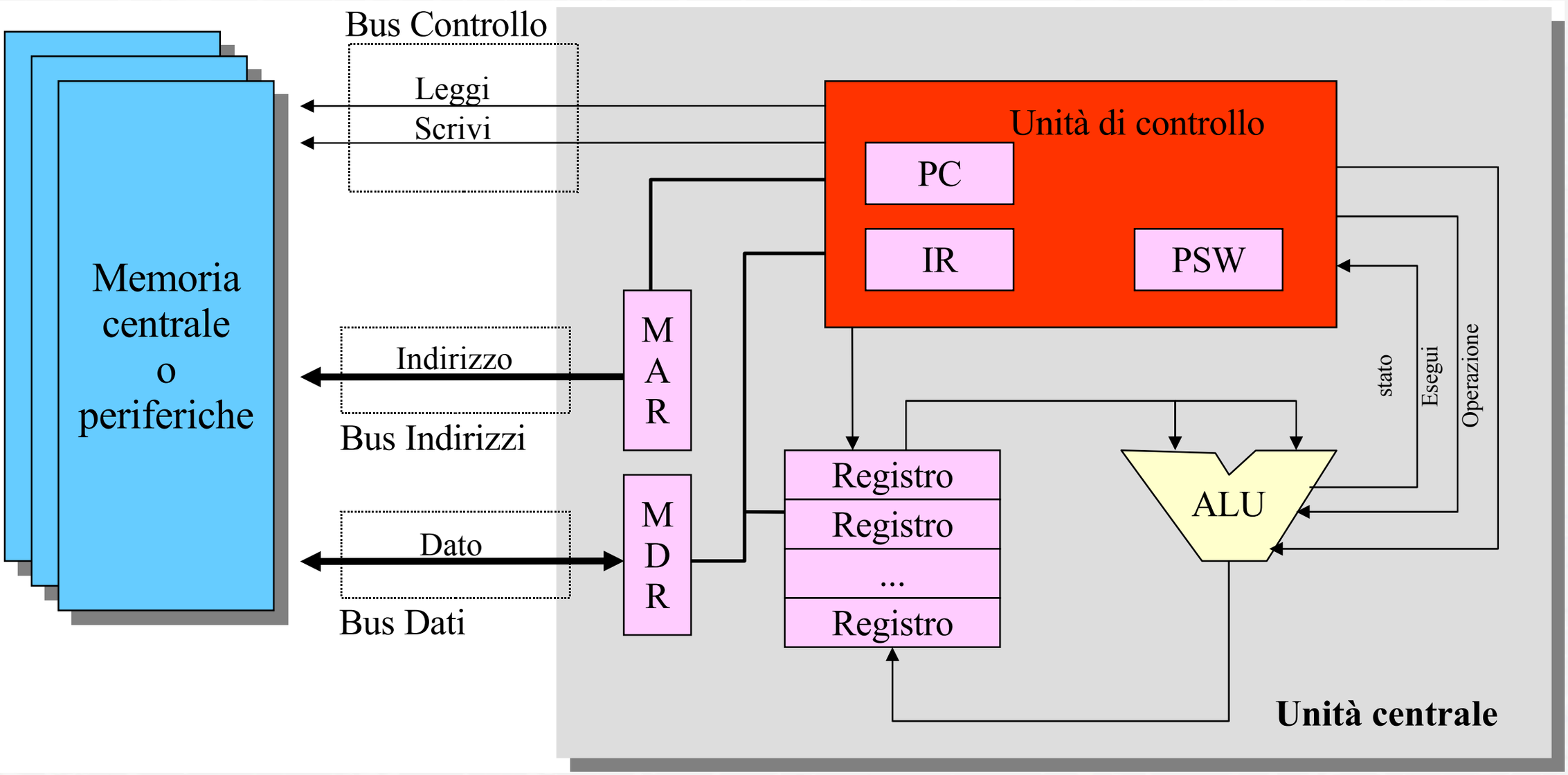
Esempio: lettura dalla memoria

Fase di Fetch (2 di 2)



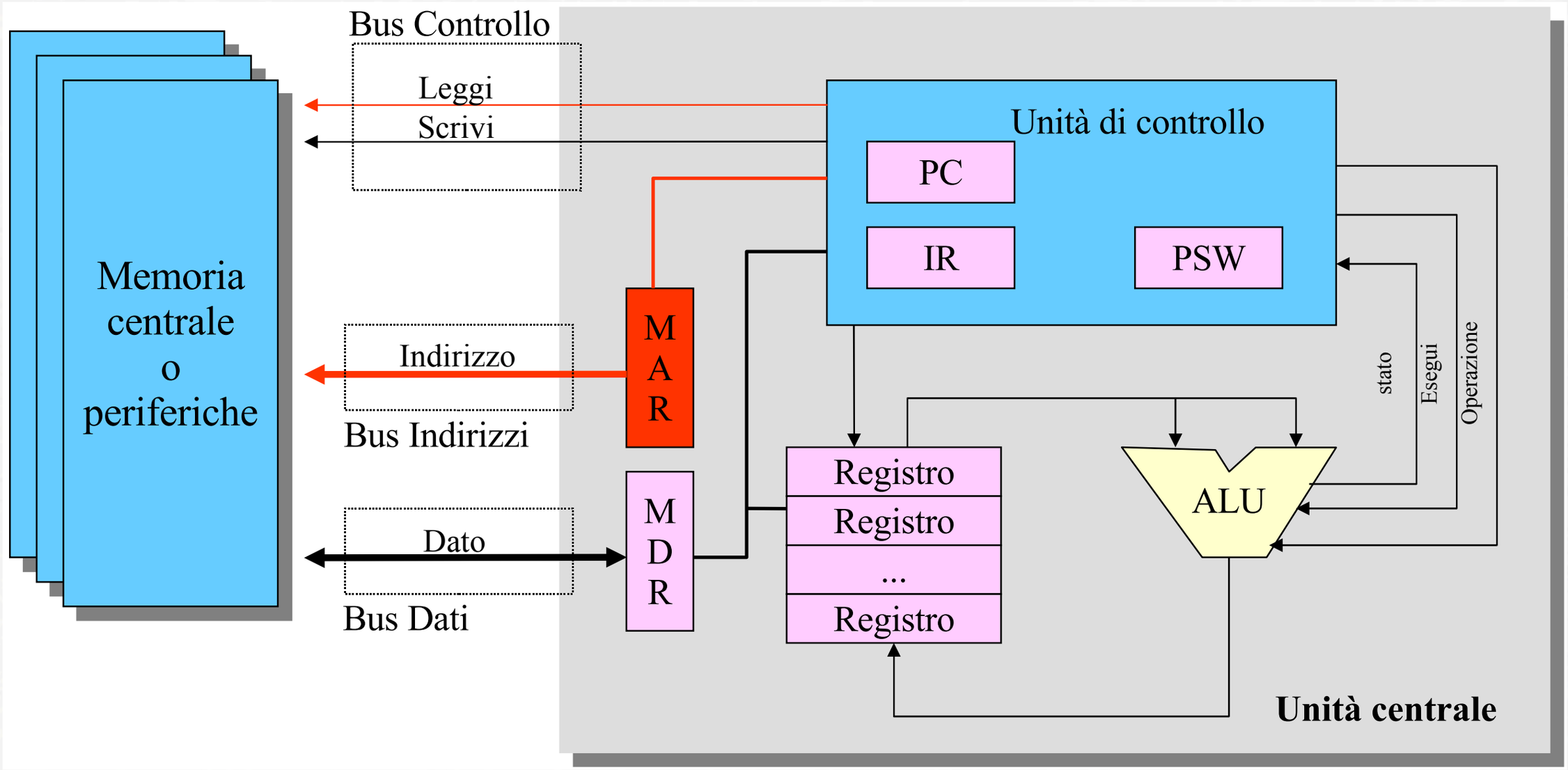
Esempio: lettura dalla memoria

Decodifica



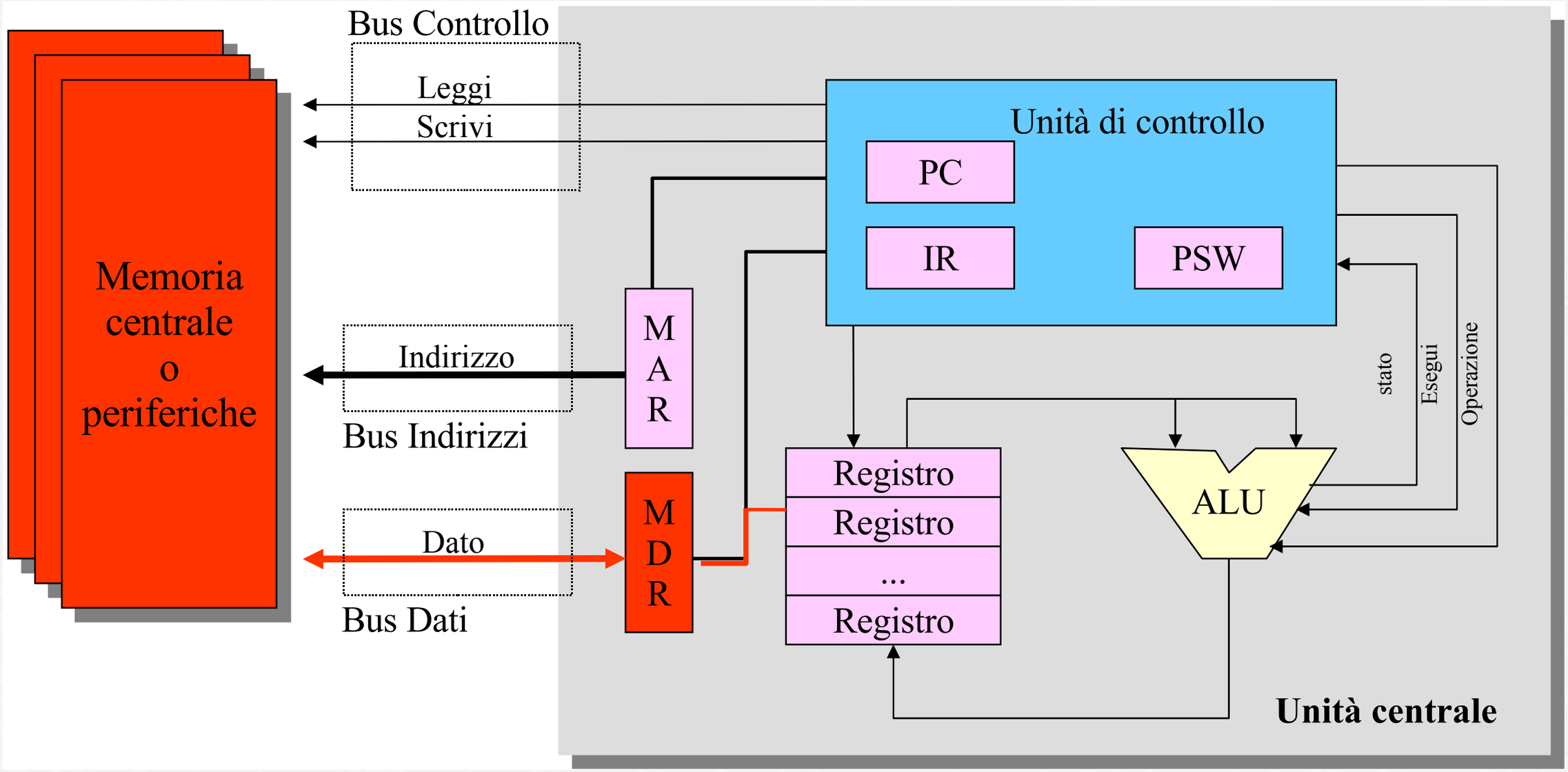
Esempio: lettura dalla memoria

Esecuzione (1 di 2)

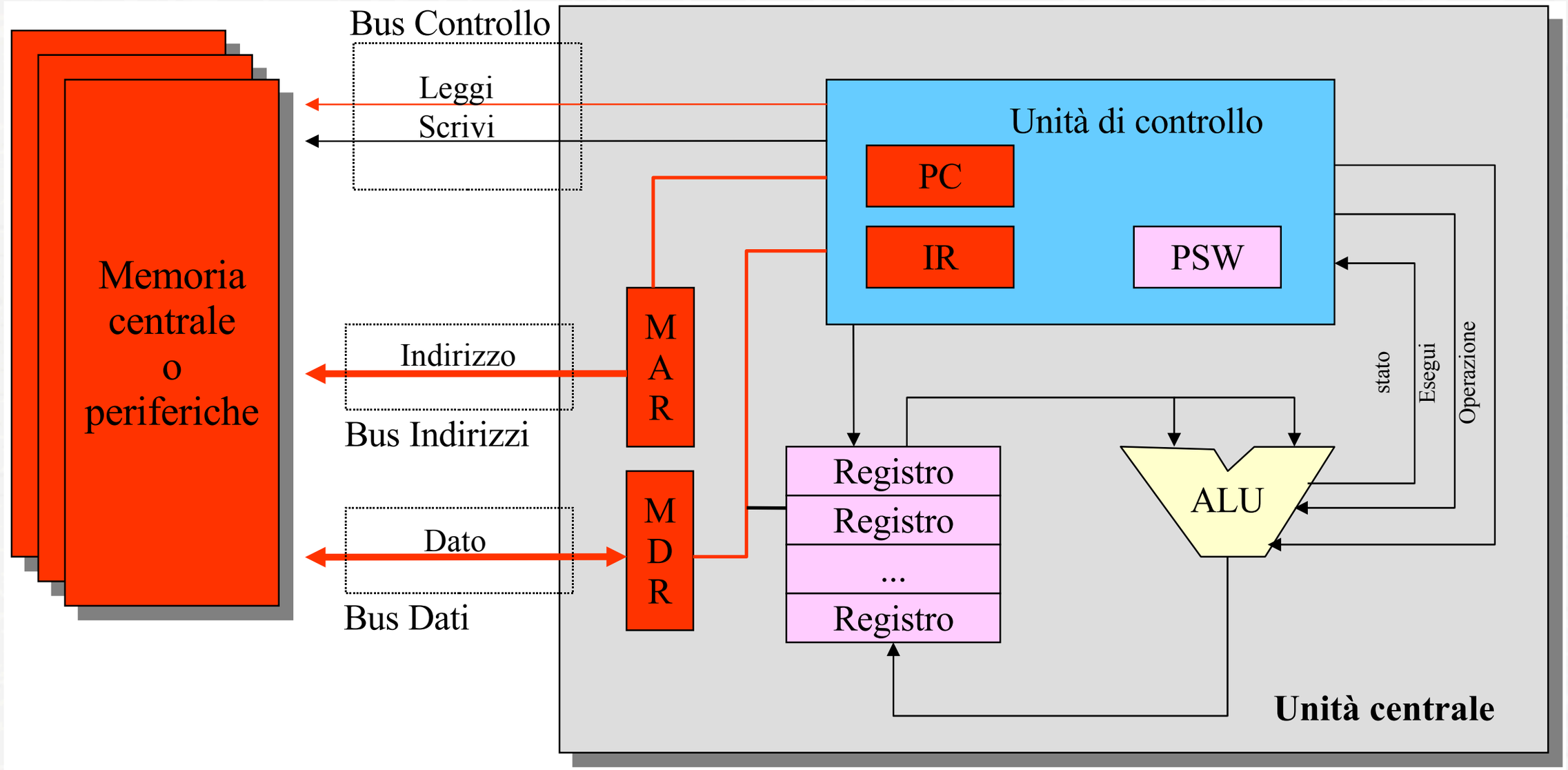


Esempio: lettura dalla memoria

Esecuzione (2 di 2)

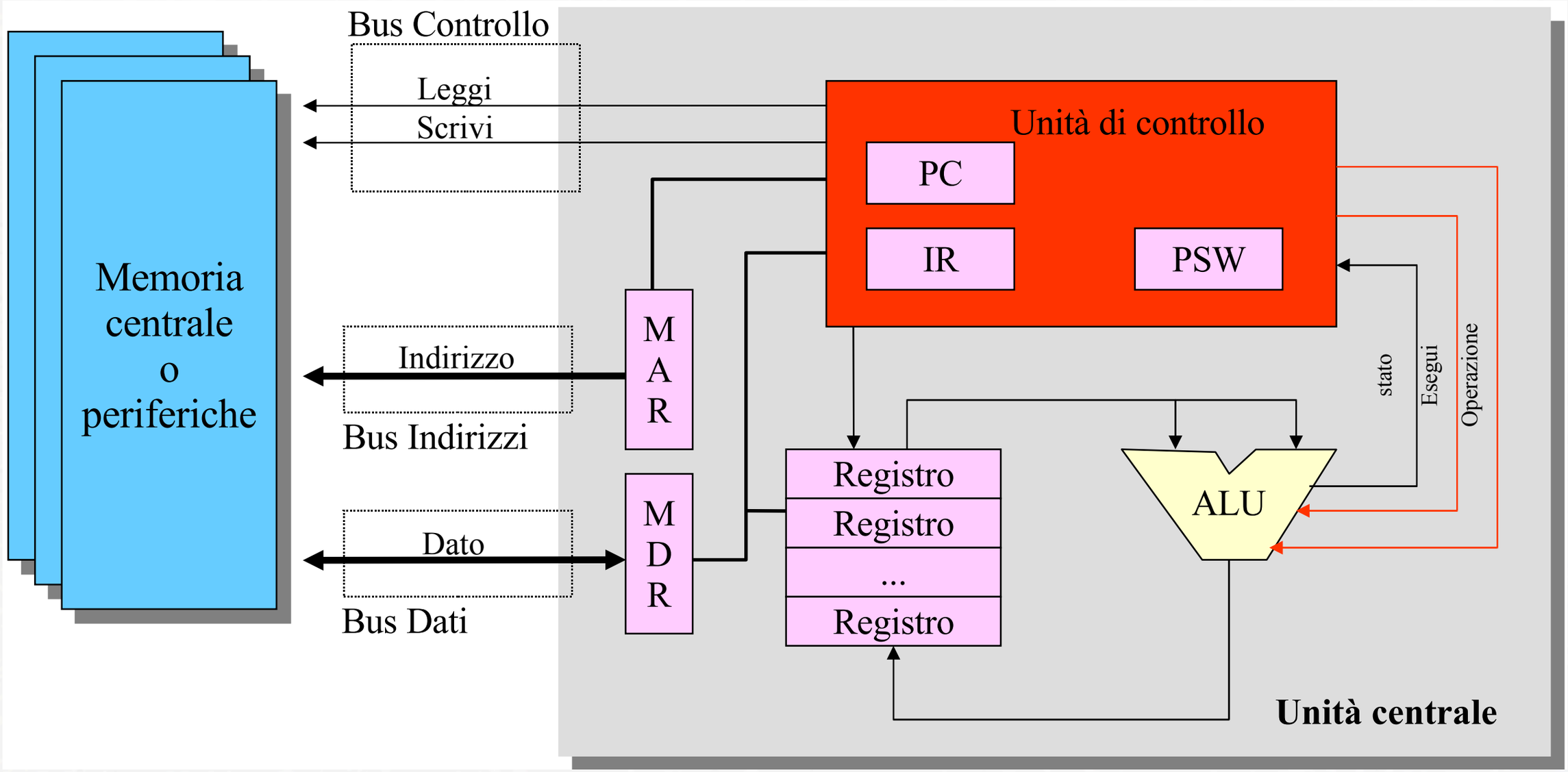


Esempio: somma tra due registri



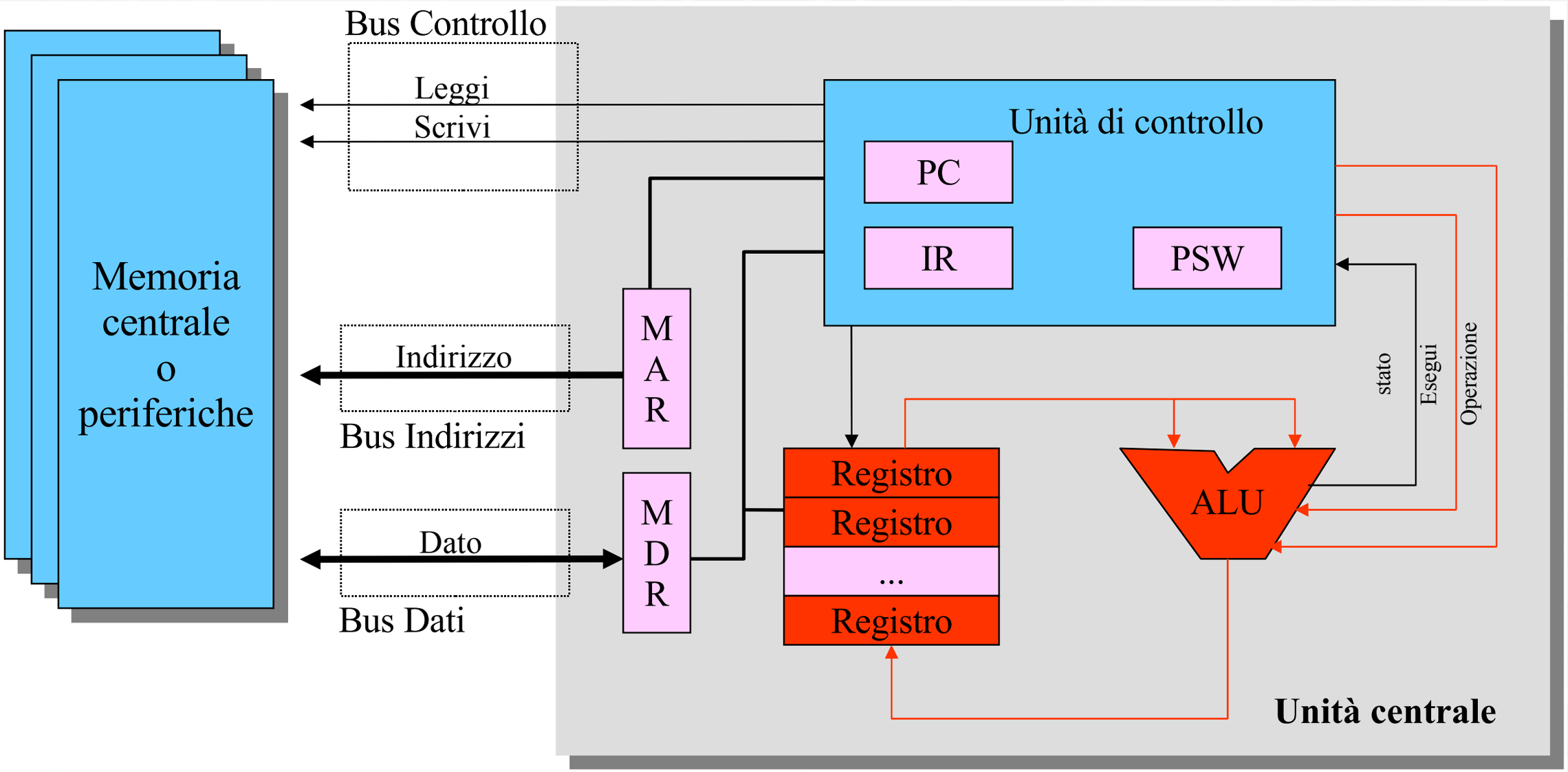
Esempio: somma tra due registri

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

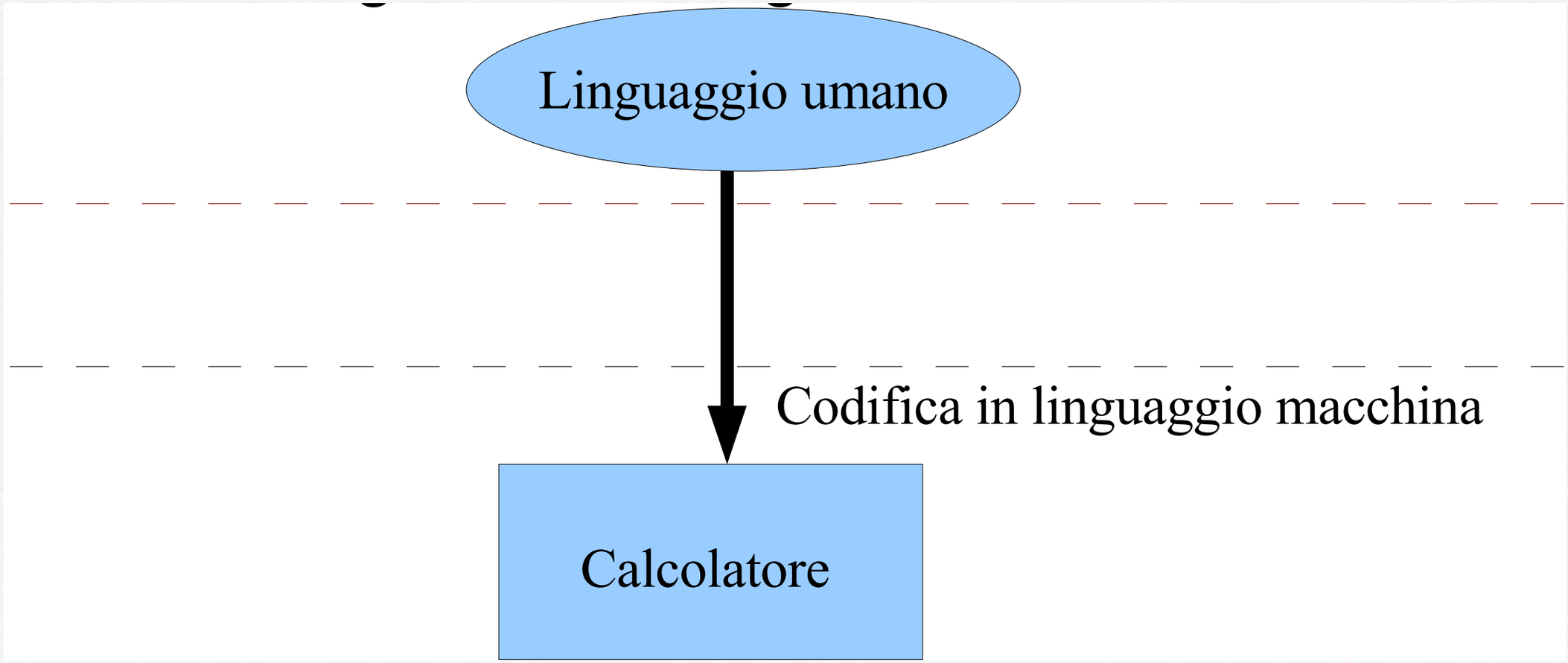
- Tradizionalmente la dimensione di MDR dà la misura del grado di parallelismo della macchina (8, 16, 32, 64 bit)

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