Architettura degli Elaboratori

Lezione 15 – Elementi architetturali di base

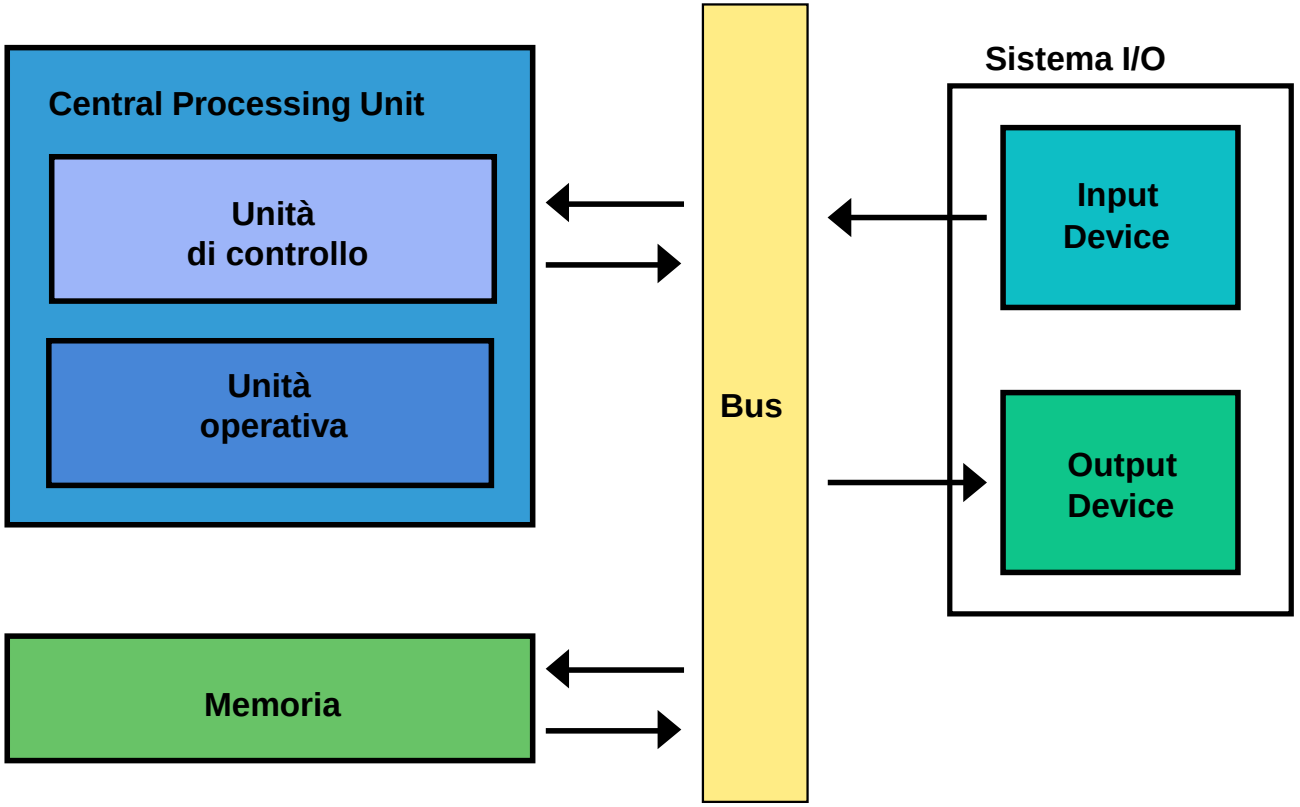
Giuseppe Cota
Dipartimento di Scienze Matematiche Fisiche e Informatiche
Università degli Studi di Parma

Indice

- Architettura di von Neumann
- Codifica istruzioni
- CPU
- Esecuzione delle istruzioni

Architettura di von Neumann

Architettura di von Neumann

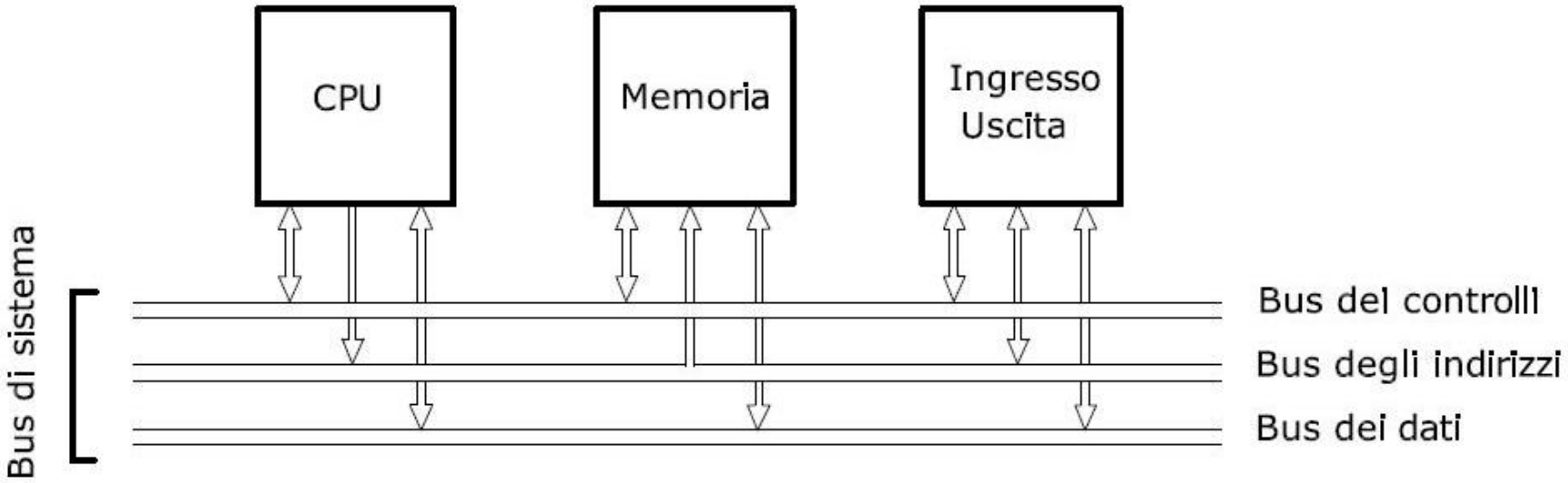


Architettura di von Neumann

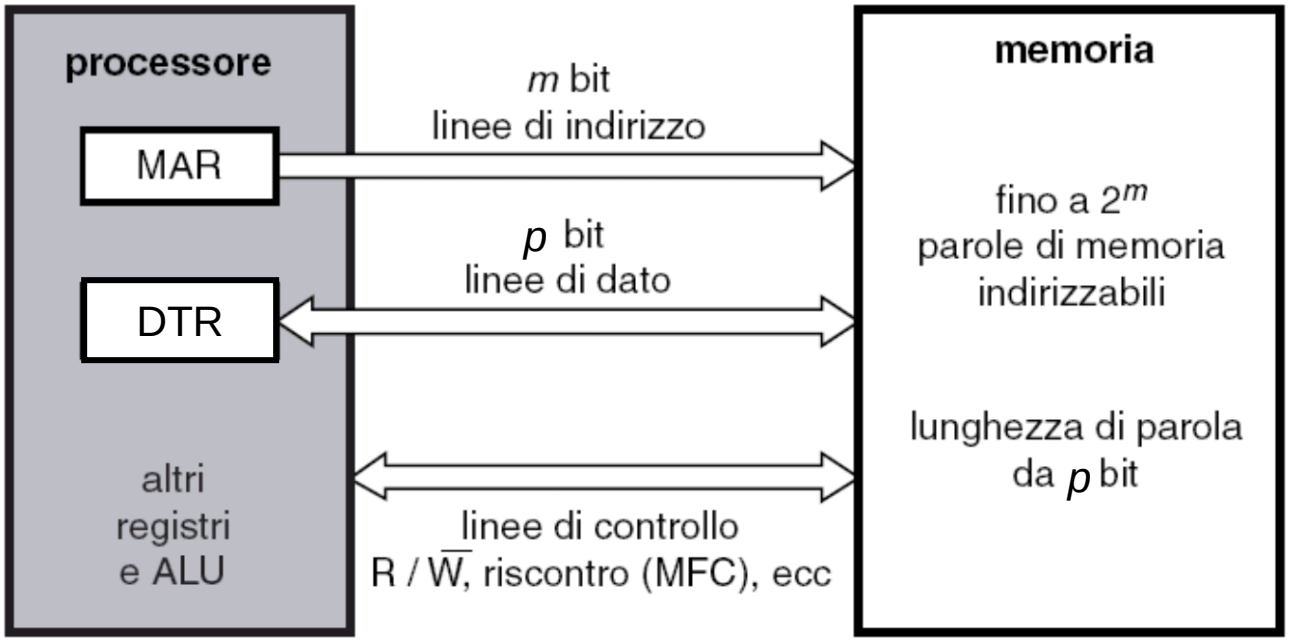
- **Unità di elaborazione e controllo (CPU*, Central Processing Unit):*** componente responsabile dell'interpretazione delle istruzioni.
  - Altri termini: *processore*, *elaboratore*
  - Solitamente suddivisa in due parti:
    - **Unità operativa:** circuiti deputati alla manipolazione delle informazioni.
    - **Unità di controllo:** responsabile del coordinamento di tutte le attività che si svolgono nella macchina. Comanda l'unità operativa, la memoria e sistema di I/O.
- **Memoria: c**ontenitore di istruzioni e dati codificati in forma binaria.
  - Le istruzioni vengono lette in memoria dalla CPU. La CPU *interpreta* le istruzioni e le *esegue*.
- **Sistema di ingresso o uscita (*I/O*, Input/Output):**
  - Insieme dei dispositivi che servono:
    - per comunicare con il mondo esterno e l'utente: ad es. video, tastiere, …
    - per aggiungere funzionalità al sistema stesso.
  - Le *interfacce* fanno da ponte tra i dispositivi periferici e il calcolatore stesso.
- **Sistema di interconnessione (bus):** circuiti che realizzano i collegamenti tra le varie parti.

Organizzazione

- Sistema a bus
  - Organizzazione tradizionale dei sistemi a microprocessore
  - Ormai superata nei PC
  - Utile schematizzazione



Memoria

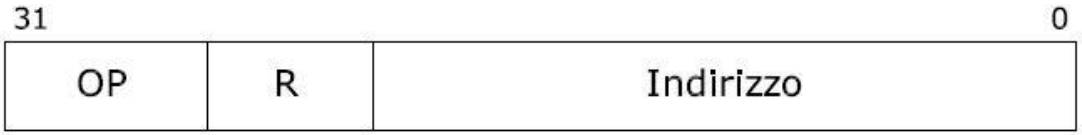


- Una memoria composta da *celle*
- Ad ogni cella corrisponde un *indirizzo*
- Il complesso delle possibili posizioni in memoria è lo *spazio degli indirizzi.*
- **Per leggere** è necessario fornire l'indirizzo (A_0 .. A_{m-1}) e il comando di lettura
- **Per scrivere** sono necessari indirizzo, dato (D_0 .. D_{p-1}) e il comando di scrittura

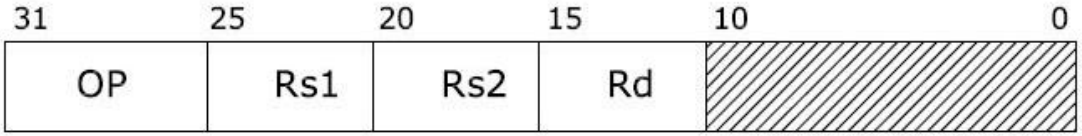
Codifica istruzioni

Codifica istruzioni

- Le operazioni effettuate dalla CPU consistono nel trasferimento dati da/a memoria (o dispositivi I/O) e alla loro manipolazione
- Le *istruzioni* sono comandi che la CPU può interpretare e associarle ad operazioni
- Supponiamo di avere istruzioni a 32 bit
  - Suddivise in campi aventi significati ben definiti,
  - I campi possono essere di lunghezza variabile e interpretati a seconda del codice dell'istruzione (OP code)
  - Esempio



Istruzione Load LD



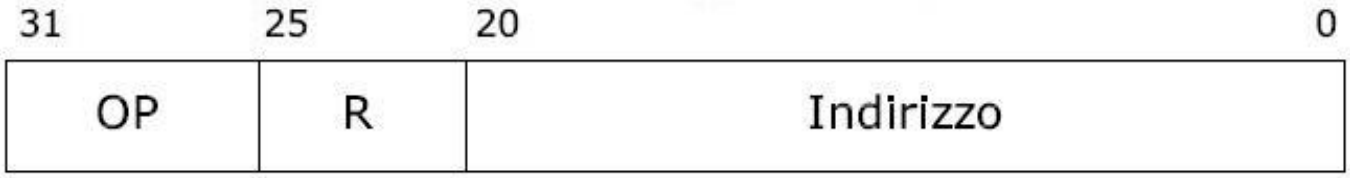
Istruzione Load ADD

Esempio istruzione Load LD

- Istruzione Load (a 32 bit): carica il contenuto di una cella di memoria in un registro della CPU
- Codice operazione: su 6 bit
  - OP per l'istruzione di load 010011
- Identificatore registro: su 5 bit
- Indirizzo su 21 bit
- **Le istruzioni possono essere rappresentate in forma simbolica, ossia in linguaggio assemblativo (assembly language)**

Esempio `LD R1,[<Address>]`

Traduzione: prendi il valore contenuto nella cella con indirizzo <Address> e mettilo dentro il registro R1



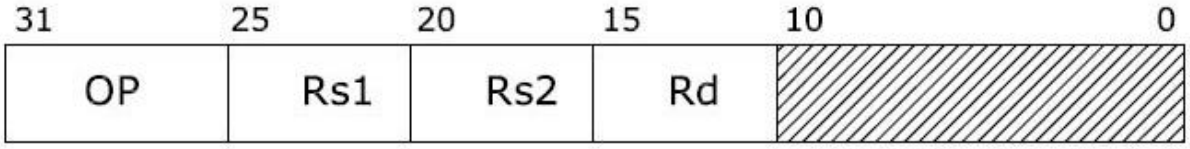
010011 00001 01011011011010011010 1

Esempio istruzione somma ADD

- Istruzione Add (a 32 bit): somma il contenuto di due registri della CPU e deposita il risultato in un terzo registro
- Codice operazione: 6 bit
  - ADD = 000101
- Identificatore registri: 5 bit
- 11 bit inutilizzati a 0

Esempio `ADD R1,R2,R3`

Traduzione: somma il valore contenuto nel registro R2 con il valore contenuto nel registro R3; il risultato mettilo dentro il registro R1.



000101 00010 00011 00001 00000000000

Esempio di sequenze di istruzioni

Lo statement C

$$a = b + c$$

si traduce in assembly come:

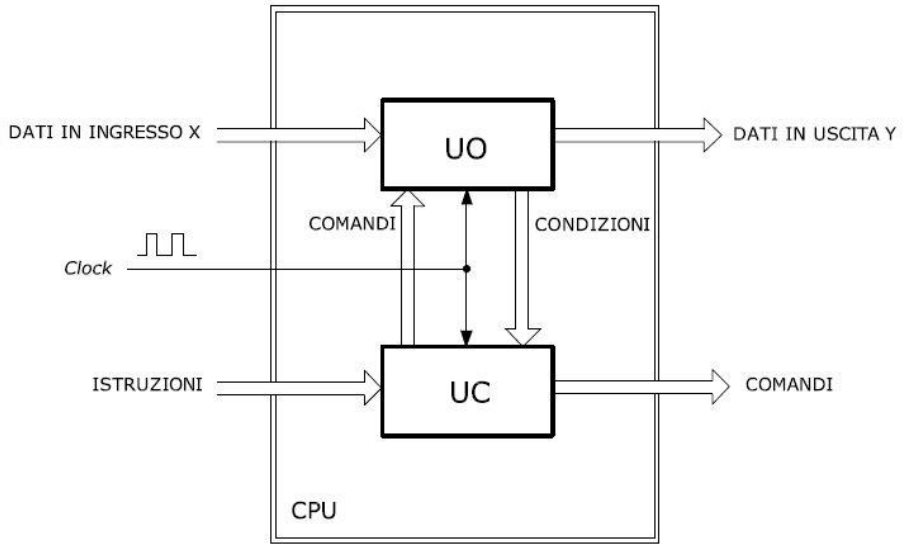
```
LD   R2, B          ;B indirizzo a cui è allocata la parola b
LD   R3, C          ;C indirizzo a cui è allocata la parola c
ADD R1, R2, R3
ST   A, R1          ;A indirizzo a cui è allocata la parola a
```

Sequenzializzazione delle istruzioni

- Quando si scrive un programma le istruzioni vengono eseguite secondo l'ordine testuale.
- L'esecuzione ordinata delle istruzioni è permessa grazie ad un **contatore di programma (Program Counter, PC)**.
- Il PC è un registro della CPU che punta alla prossima istruzione da eseguire.
  - Contiene l'indirizzo della prossima istruzione da eseguire.
  - Ogni volta che un'istruzione a 32 bit (ossia 4 byte) viene eseguita, il PC viene incrementato di 4.
- In realtà i programmi non sono lineari, ma contengono dei cicli (while, for) o dei percorsi alternativi (if, else).
  - Istruzioni di *salto condizionato JZ*: se è verificata una condizione, si salta alla posizione codificata nell'istruzione.
  - Istruzioni di *salto incondizionato JMP*: si salta alla posizione codificata nell'istruzione.

CPU

La CPU: unità operativa e unità di controllo



- **L'unità di controllo (UC)** è responsabile dell'esecuzione delle istruzioni, comandando l'unità operativa a svolgere le operazioni corrispondenti ai codici di istruzione.
  - UC interpreta l'istruzione e la traduce in segnali di controllo per UO (o altri dispositivi).
- **L'unità operativa (UO)** svolge la funzione di manipolazione dell'informazione trasformando i dati in ingresso X in dati in uscita Y.
  - Componente essenziale: ALU
  - Le condizioni di stato dell'UO in seguito ad un'operazione vengono inviate all'UC.
- Il funzionamento della CPU è scandito dal clock.

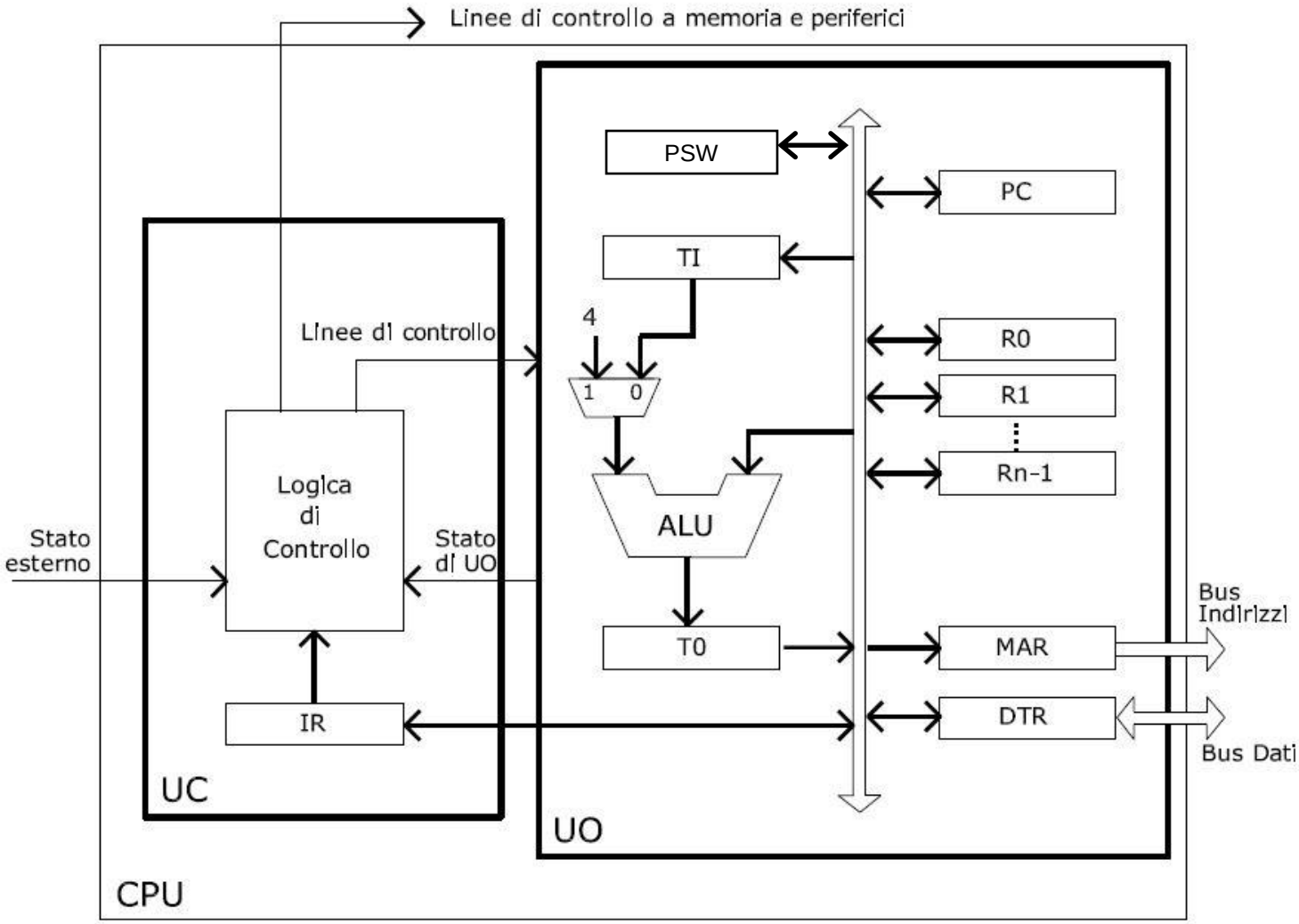
CPU: macrofasi del processo di elaborazione di una istruzione

1. Fase di fetch (lettura da memoria e decodifica):
   a) UC comanda la lettura del contenuto della parola il cui indirizzo I è nel PC
   b) Decodifica l'istruzione letta
2. Esecuzione dell'istruzione:
   a) L'ALU (*Arithmetic and Logic Unit)* svolge le operazioni numeriche e logiche.
   b) Fine esecuzione: il PC viene incrementato di 4 (I+4) in modo da puntare all'istruzione successiva a quella eseguita.

Componenti principali della CPU

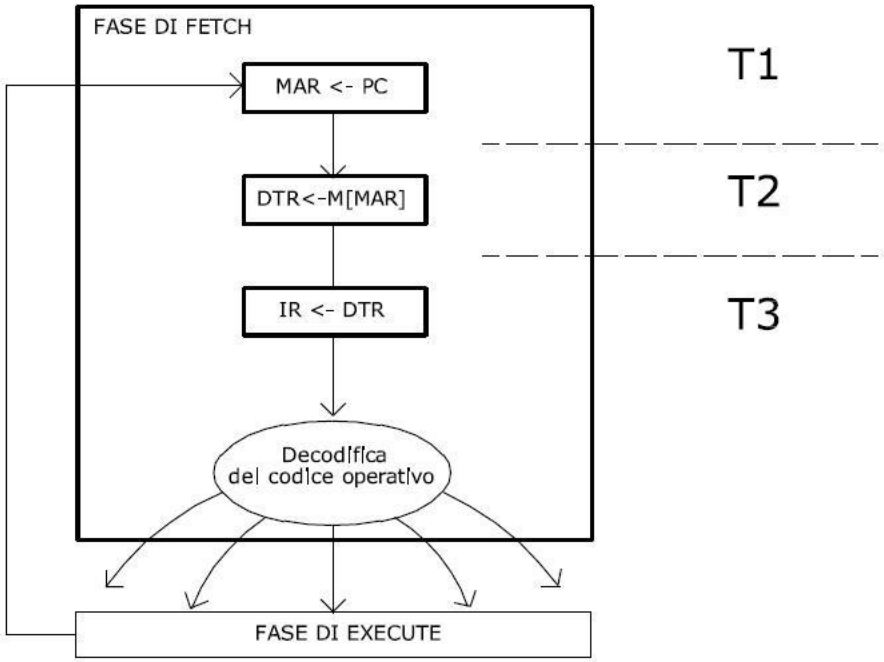
- **PC** *(Program Counter)*: registro che *punta* alla prossima istruzione da eseguire, ossia contiene l'indirizzo della prossima istruzione da eseguire.
- **IR** *(Instruction Register):* contiene l'istruzione da eseguire.
  - Fa parte dell'UC, viene caricato in fase di fetch, l'UC lo decodifica (interpreta) generando una sequenza sincronizzata di segnali di controllo inviati ai diversi componenti del sistema (UO, memoria, dispositivi I/O)
- **ALU** (*Arithmetic Logic Unit)*: componente che svolge le operazioni aritmetico-logiche.
- **R1, R2, ... , Rn** (*General Purpose Registers)*: impiegati per contenere i dati su cui l'ALU esegue le proprie operazioni
- **MAR** *(Memory Address Register)* contiene l'indirizzo della locazione di memoria da leggere o scrivere.
- **DTR** *(Data Transfer Register)* registro contenente l'informazione che deve essere scambiata fra la memoria e la CPU.
- **PSW** *(Processor Status Word)*: registro contenente i bit di stato della CPU.
  - Ad esempio: bit di overflow, carry, divisione per zero, interrupt, ecc.

Organizzazione CPU



Esecuzione delle istruzioni

Fase di fetch



- La fase di fetch inizia con il prelievo dell'istruzione e si conclude con la sua decodifica.
- Passi eseguiti ad ogni ciclo di clock:
  1. Il contenuto del PC viene trasferito in MAR.
  2. L'uscita di MAR viene presentata sul bus degli indirizzi e viene asserito il comando di lettura della memoria; il contenuto della cella viene presentato sul bus dei dati e caricato in DTR.
  3. Il contenuto di DTR viene copiato in IR in modo da essere decodificato dall'UC.

Fase di esecuzione

- L'unità di controllo genera i segnali che fanno svolgere le azioni richieste in base al contenuto di IR e di eventuali condizioni di stato.
- Esempio esecuzione istruzione `ADD R1,R2,R3`:
  1. Il contenuto di R2 viene trasferito nel registro temporaneo TI che costituisce un operando dell'ALU
  2. Il contenuto di R3 viene presentato, via bus interno, all'altro ingresso dell'ALU; viene asserito il comando ADD all'ALU in modo da fare la somma. Il risultato viene memorizzato nel registro temporaneo TO.
  3. Il contenuto di TO viene trasferito in R1

Domande?

Riferimenti principali

- Capitolo 2 di ***Calcolatori elettronici. Architettura e Organizzazione,*** Giacomo Bucci. McGraw-Hill Education, 2017.