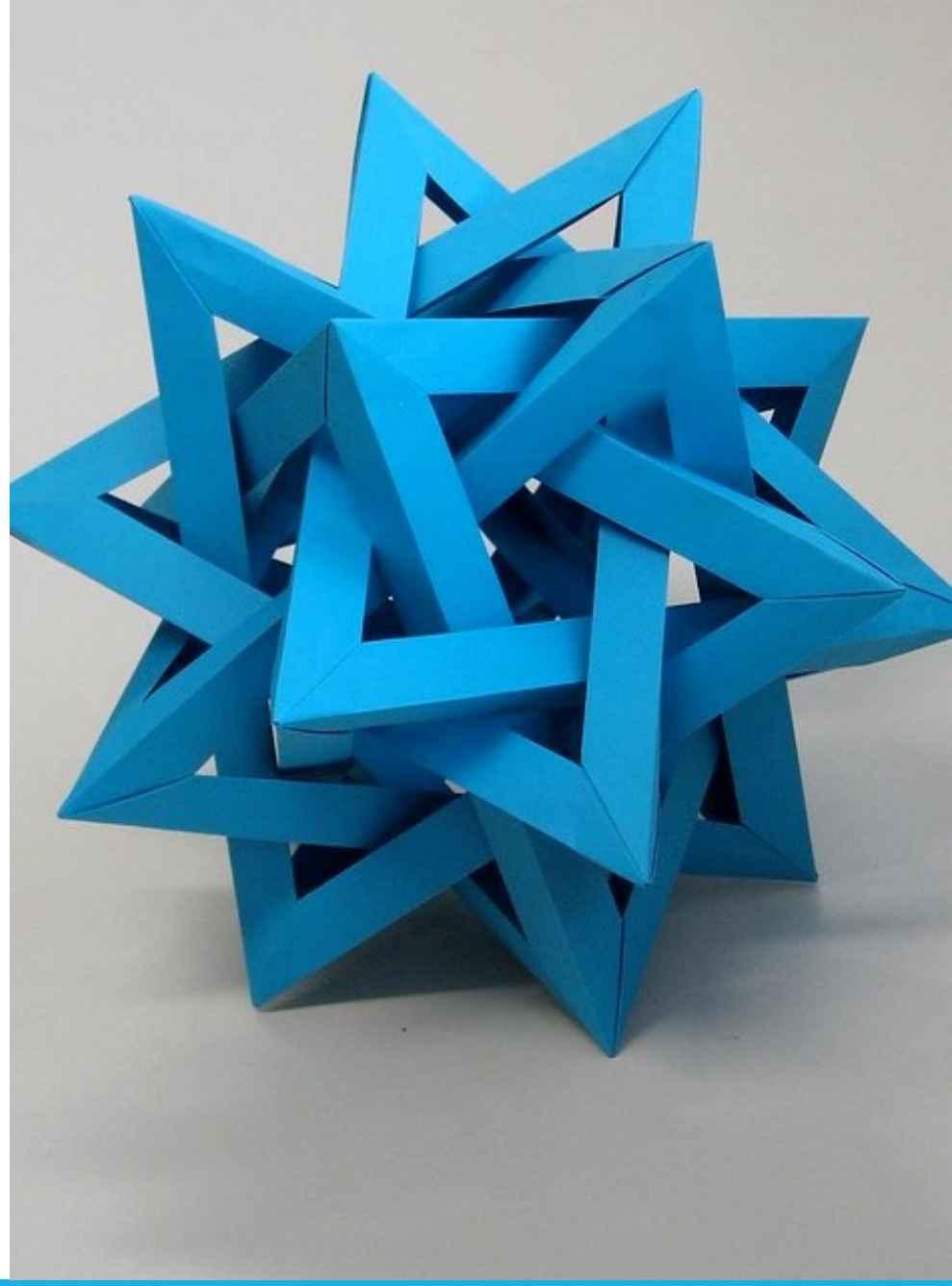
Politecnico di Torino
Dipartimento di Automatica e Informatica

Unità T2: Architettura degli elaboratori



Architettura degli elaboratori

Architettura di un elaboratore

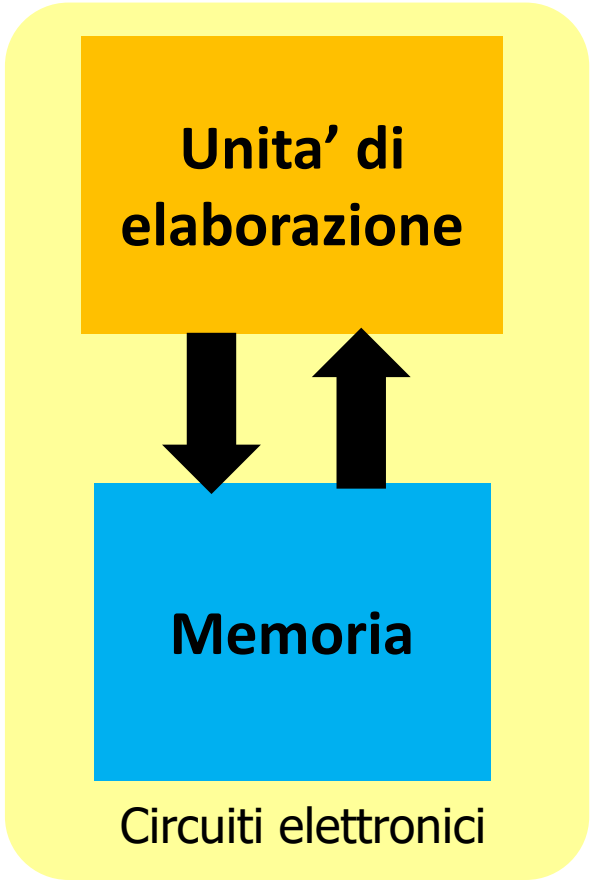
- Per comprendere il processo di programmazione, è necessario conoscere almeno per grandi linee gli elementi costitutivi di un computer.

- Faremo riferimento al tipico PC, anche se tutti i calcolatori hanno sostanzialmente la stessa struttura.

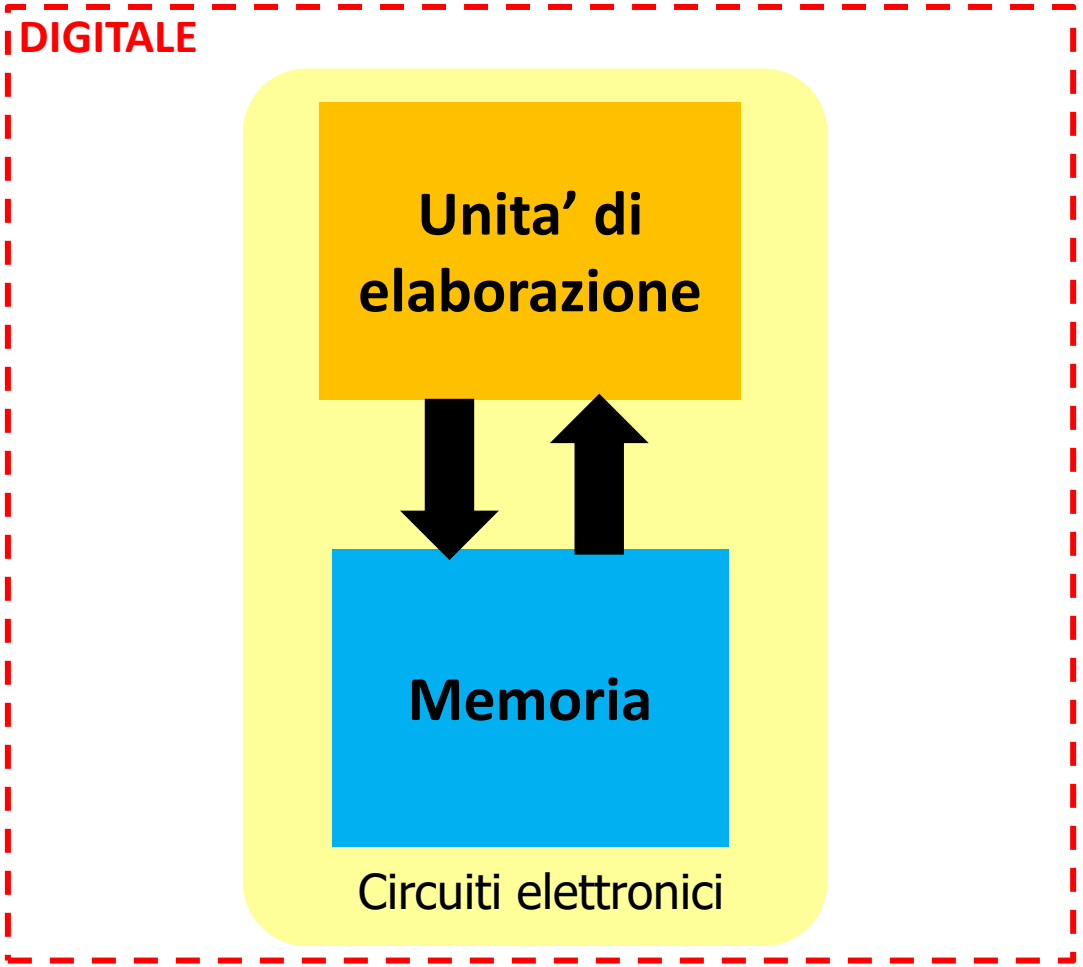
I blocchi fondamentali dell’elaboratore

Circuiti elettronici

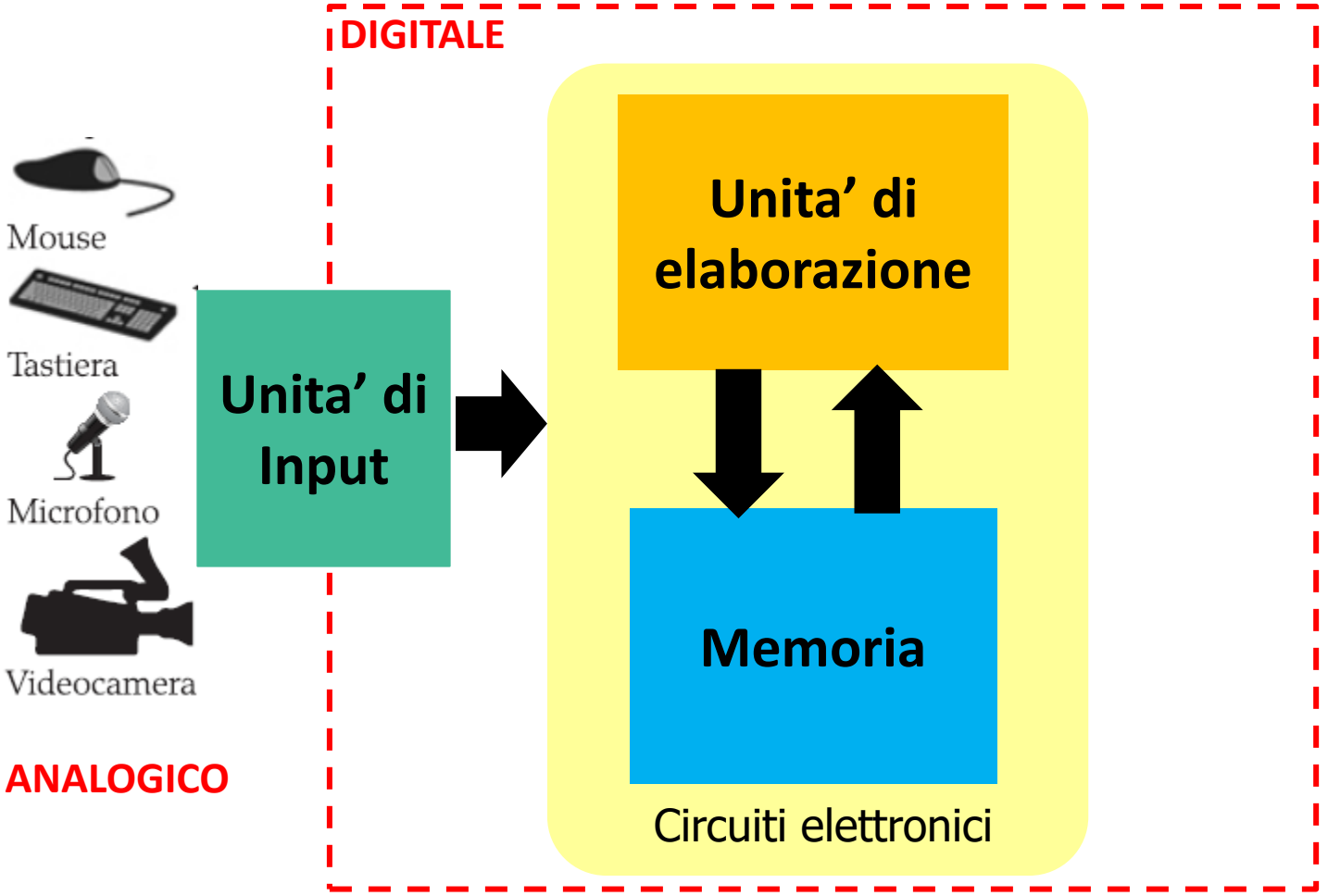
I blocchi fondamentali dell'elaboratore



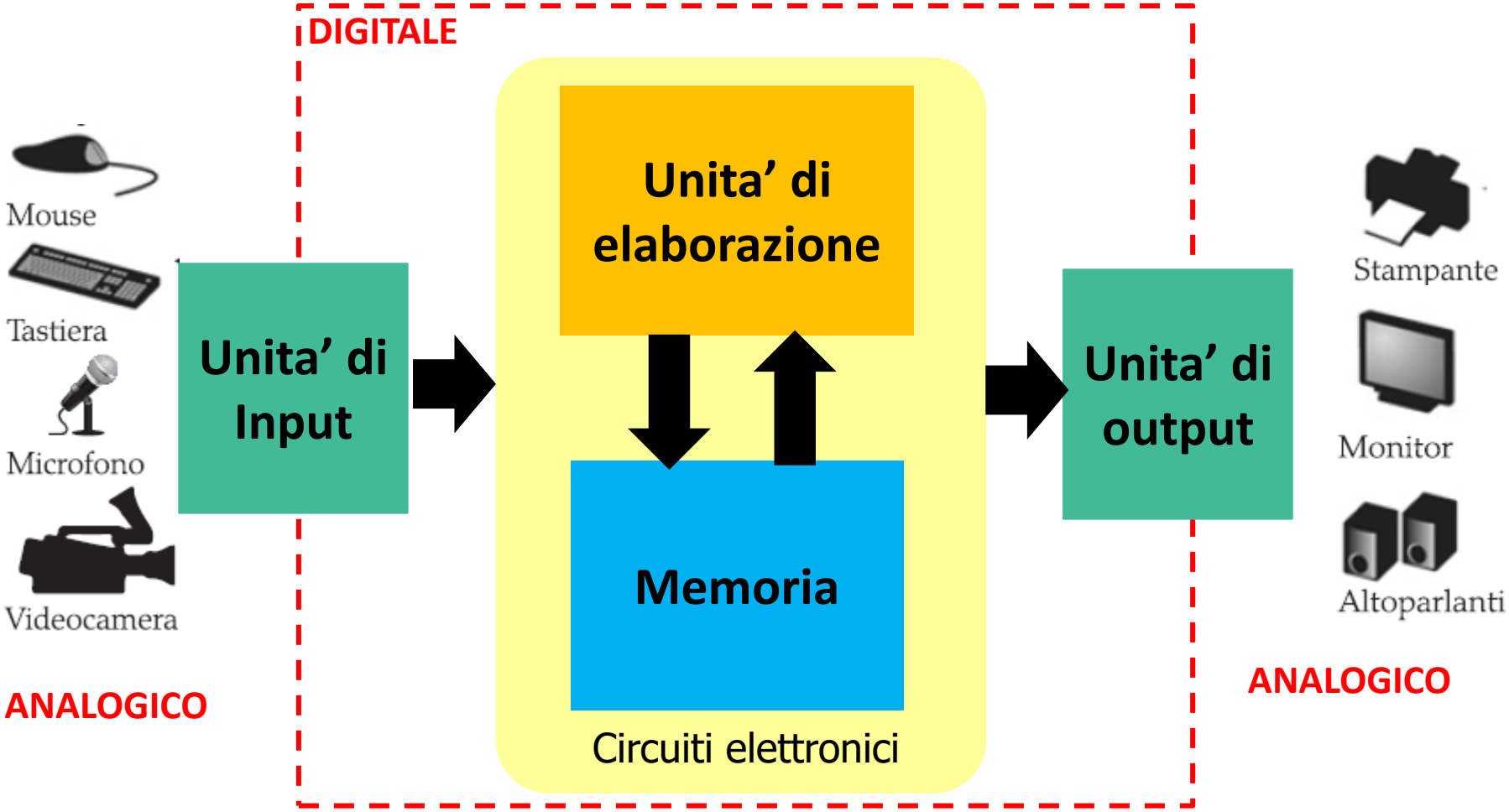
I blocchi fondamentali dell'elaboratore



I blocchi fondamentali dell'elaboratore



I blocchi fondamentali dell'elaboratore



Componenti fondamentali

- Unità di I/O
  - Interfaccia da/verso utente
  - Implicano un cambio di dominio fisico
    - Umano = analogico, asincrono, non elettrico
    - Calcolatore = digitale, sincrono, elettrico
    - Necessarie opportune conversioni

Componenti fondamentali

- Unità di I/O
  - Interfaccia da/verso utente
  - Implicano un cambio di dominio fisico
    - Umano = analogico, asincrono, non elettrico
    - Calcolatore = digitale, sincrono, elettrico
    - Necessarie opportune conversioni
- Unità di elaborazione
  - Contiene i circuiti per l'esecuzione delle 'istruzioni'
  - "Microprocessore"

Componenti fondamentali

- Unità di I/O
  - Interfaccia da/verso utente
  - Implicano un cambio di dominio fisico
    - Umano = analogico, asincrono, non elettrico
    - Calcolatore = digitale, sincrono, elettrico
    - Necessarie opportune conversioni
- Unità di elaborazione
  - Contiene i circuiti per l'esecuzione delle 'istruzioni'
  - "microprocessore"
- Memoria
  - Memorizza in modo permanente dati e programmi
  - Necessaria per l'elaborazione per motivi di efficienza

Componenti fondamentali

- Unità di I/O
  - Interfaccia da/verso utente
  - Implicano un cambio di dominio fisico
    - Umano = analogico, asincrono, non elettrico
    - Calcolatore = digitale, sincrono, elettrico
    - Necessarie opportune conversioni
- Unità di elaborazione
  - Contiene i circuiti per l'esecuzione delle 'istruzioni'
  - "microprocessore"
- Memoria
  - Memorizza in modo permanente dati e programmi
  - Necessaria per l'elaborazione per motivi di efficienza

CIRCUITI (dentro la "scatola")

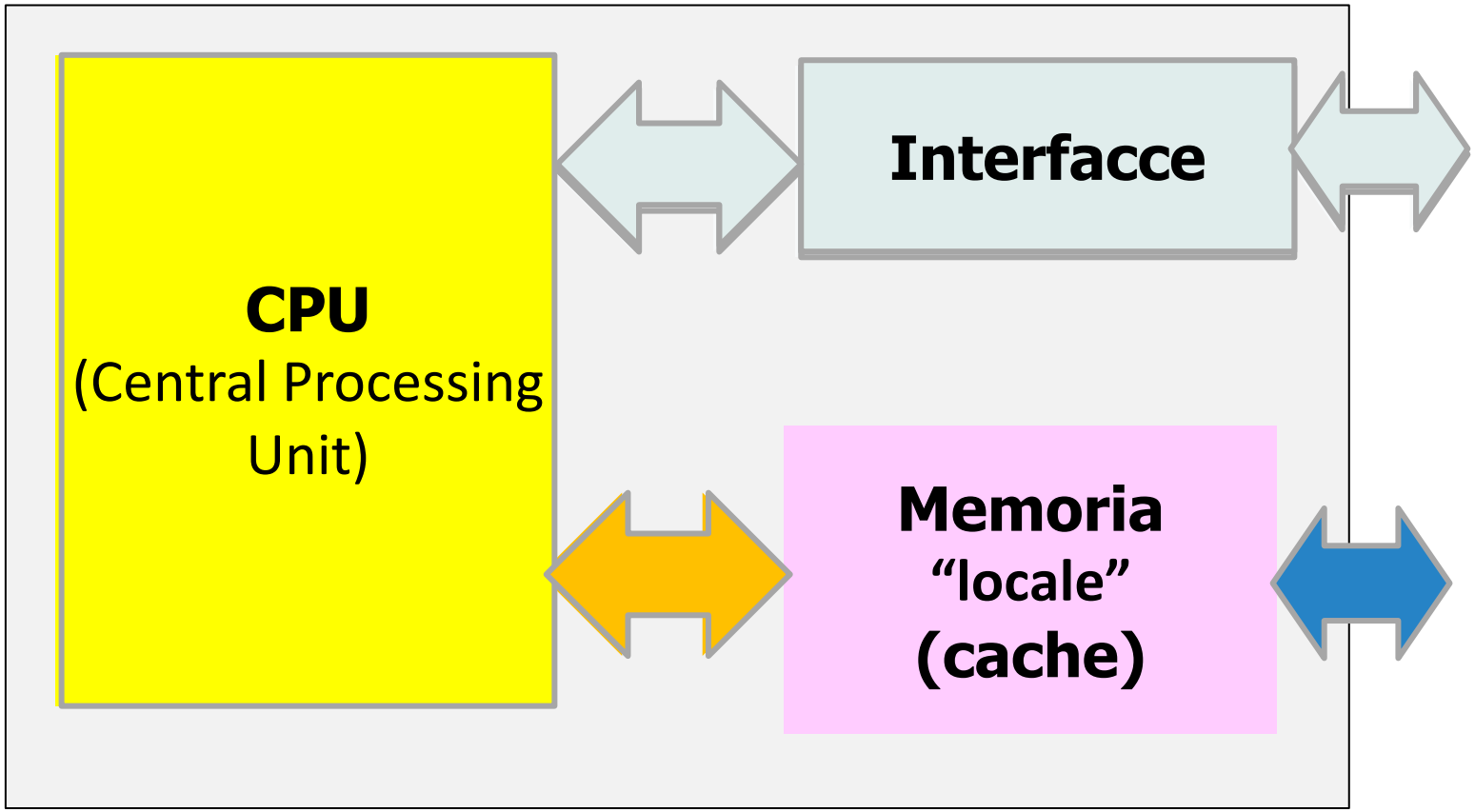
Il microprocessore

Microprocessore

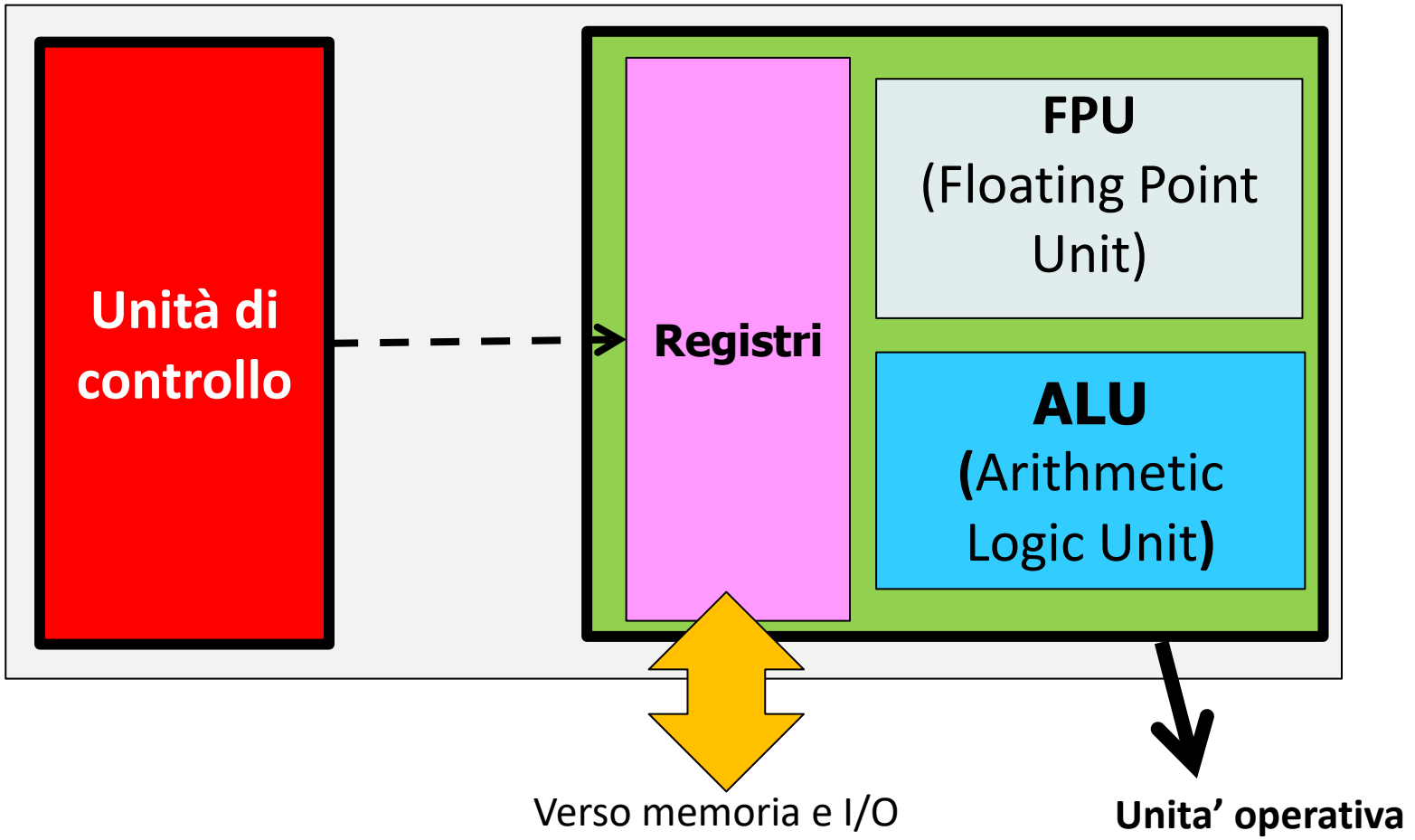
- Il microprocessore è il circuito che fisicamente esegue TUTTE le istruzioni

- Contiene quindi:
  - Tutti i circuiti per eseguire le operazioni di base su numeri interi, reali e operazioni logiche
  - Opportuni circuiti per il "coordinamento" dell'esecuzione delle istruzioni (per es. il loro sequenziamento, controllo degli errori)
  - Interfacce per spostare dati da/verso la memoria
  - Interfacce per spostare dati da/verso unità di I/O
- Ha (in linea di principio) limitate capacità di memorizzare dati e/o istruzioni
  - Lo stretto necessario per eseguire le operazioni
  - Ma per motivi di efficienza (v. dopo) una parte della memoria è "ospitata" nel microprocessore

Struttura del microprocessore

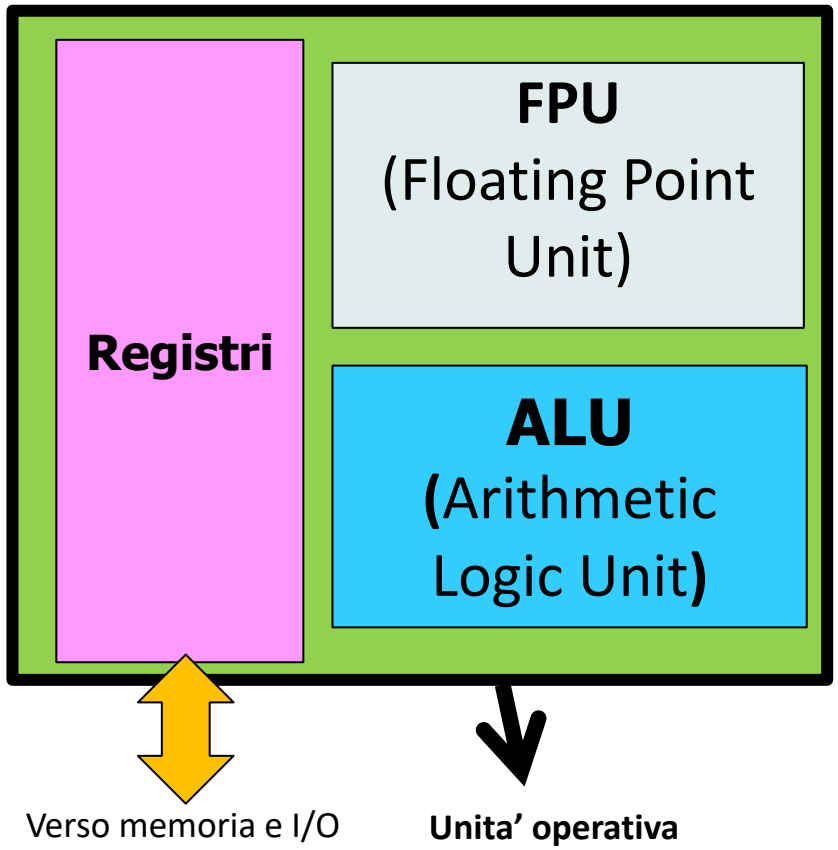


La CPU


Unità di controllo
Registri
FPU
(Floating Point Unit)
ALU
(Arithmetic Logic Unit)
Verso memoria e I/O
Unita' operativa

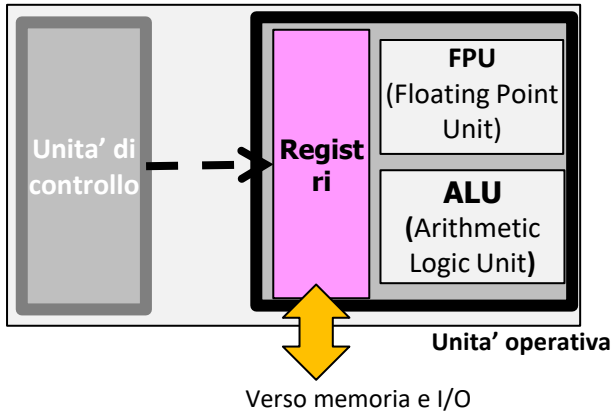
Unità operativa

- Svolge tutte le elaborazioni richieste (aritmetiche, logiche, grafiche?, ...).
- È composta di:
  - **ALU** (Arithmetic Logic Unit)
  - **Registri** istruzioni e dati
  - **FPU** (Floating Point Unit)
  - Registro dei **Flag**

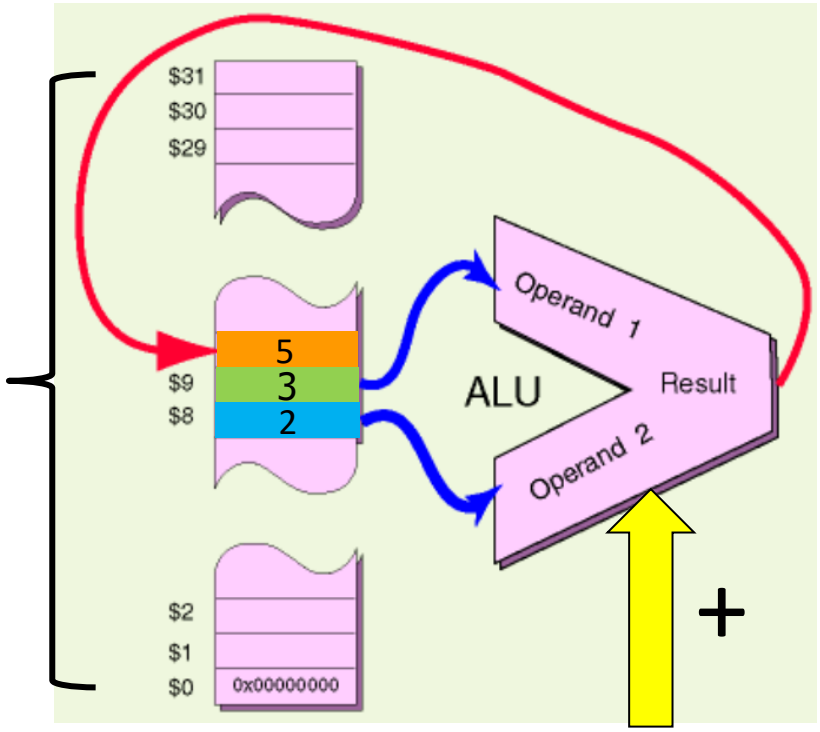


Registri

- Elementi di memoria locale usati per conservare temporaneamente dei dati (es. risultati parziali) o istruzioni
  Ogni trasferimento da processore a memoria e viceversa avviene tra registri e memoria
- Numero limitato (8...128)



32 registri



Registro dei Flag

- Registro i cui bit segnalano:
  - Determinati stati dell'insieme delle unità di calcolo
  - Alcune informazioni sul risultato dell'ultima operazione eseguita
- Utilizzati per implementare alcune operazioni condizionali

- Alcuni flag significativi
  - Zero: Segnala se il risultato dell'operazione è o no zero.
  - Segno: indica il segno del risultato dell'operazione precedente
  - Overflow: indica se il risultato dell'operazione precedente eccede i limiti della rappresentazione

ALU e FPU

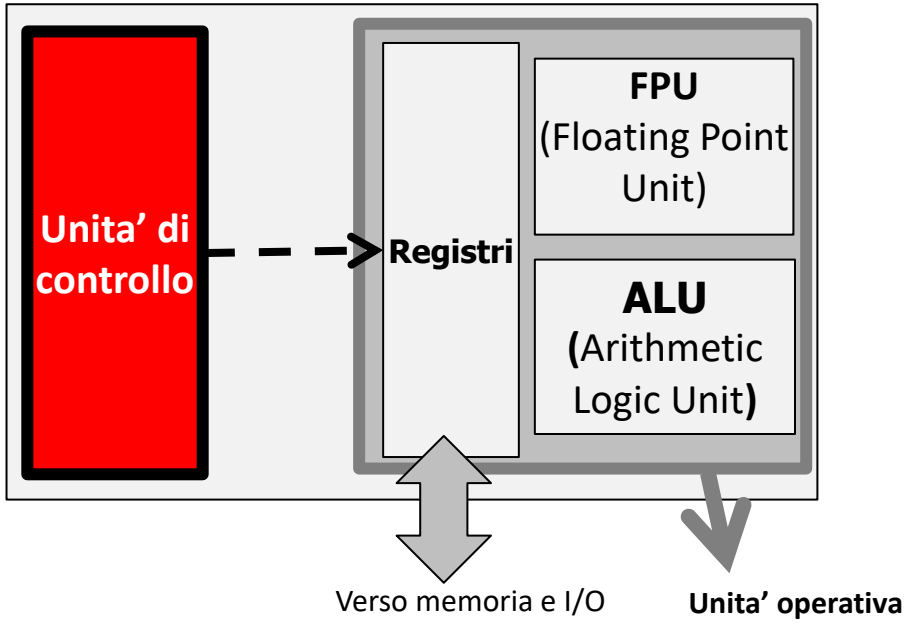
- **ALU**
  - Svolge tutti i calcoli (aritmetici, logici, confronti) su numeri interi
- **FPU**:
  - Svolge I calcoli su numeri reali

- Notevole differenza nel tempo di esecuzione!
  - Il rapporto dipende dallo specifico processore, un'operazione FPU è tipicamente più lenta di 5-50 volte più lenta di un'operazione ALU.

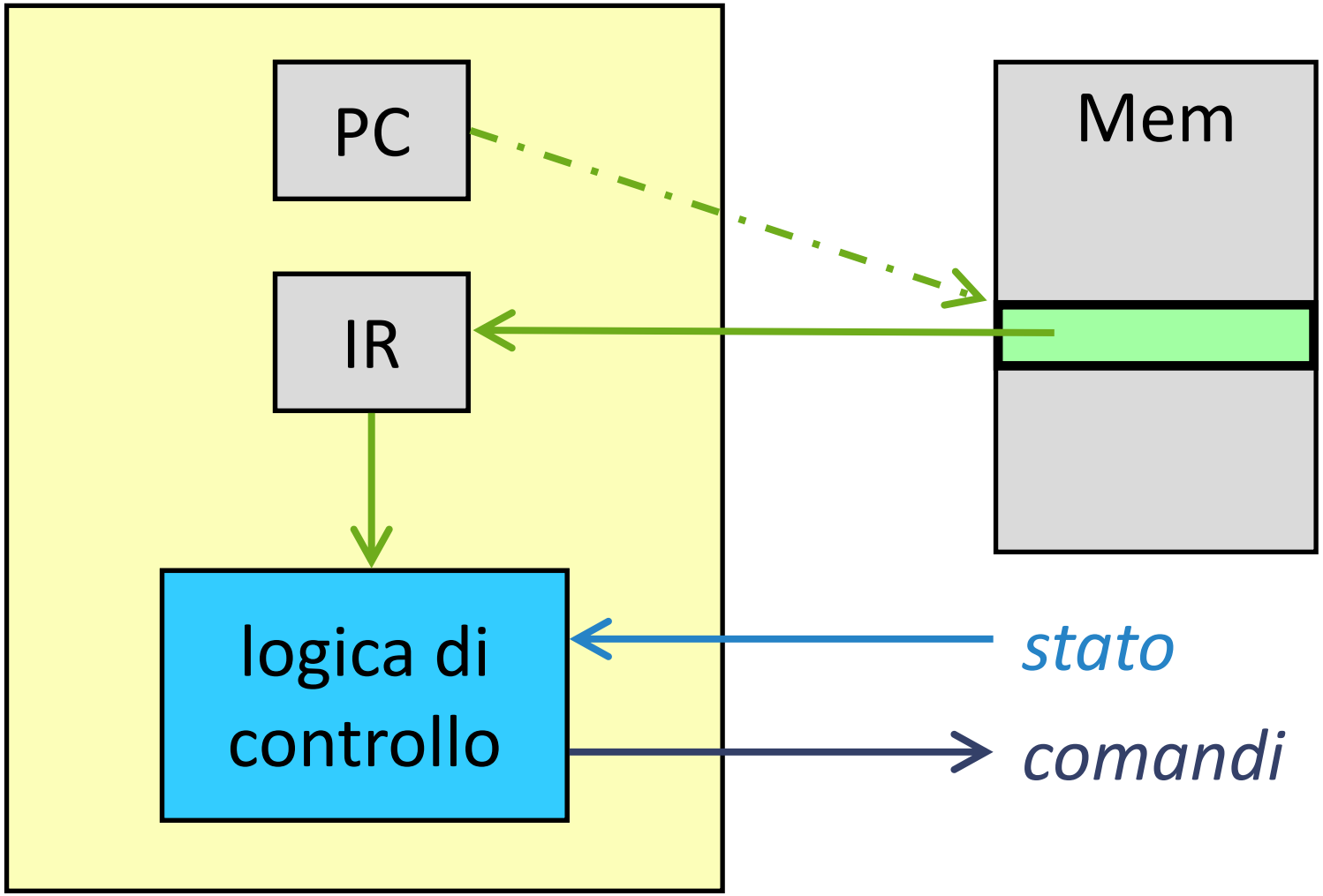
Unità di controllo

- È il cuore dell'elaboratore:

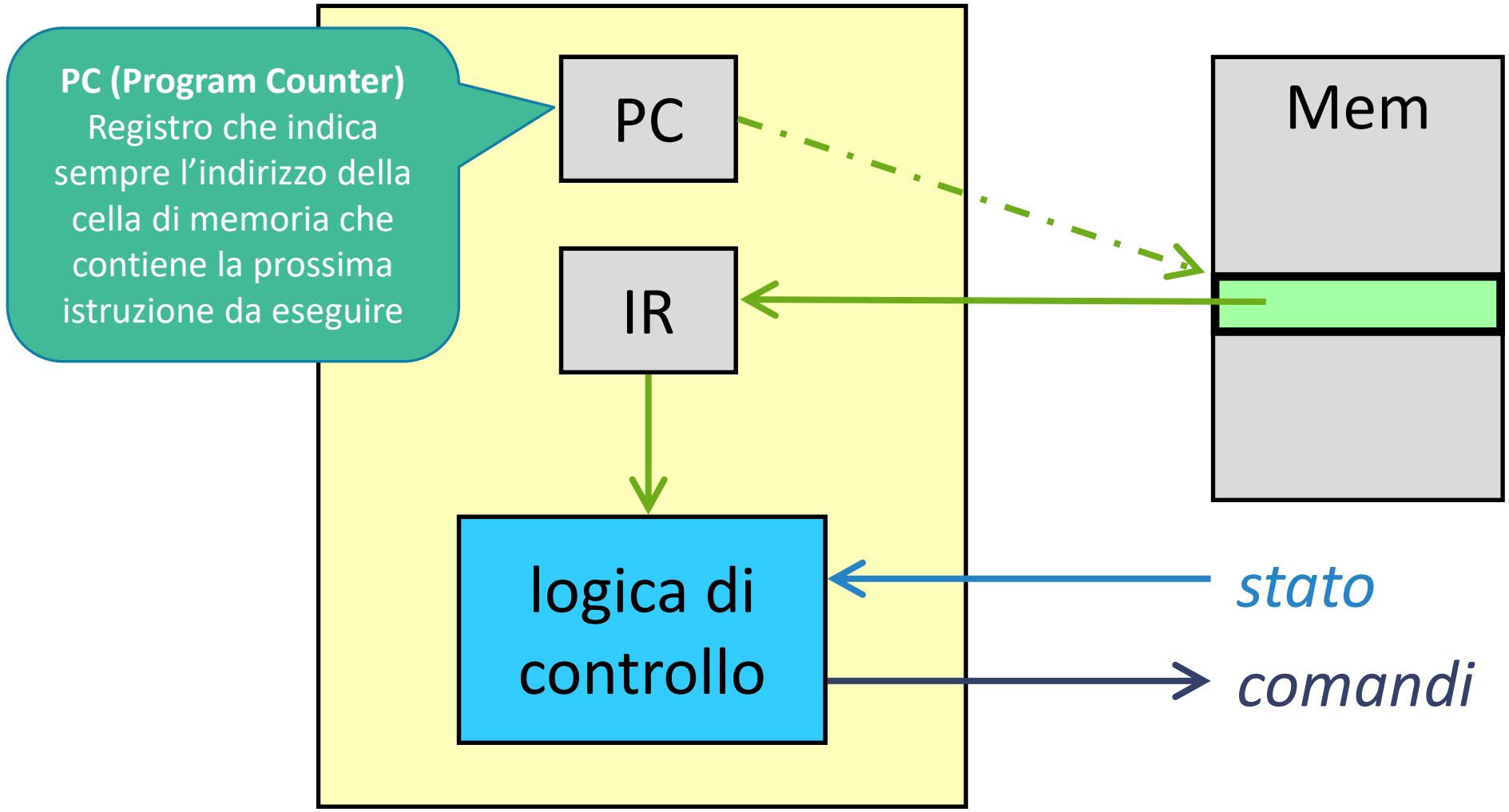
- In base alle **istruzioni** contenute nel programma che esegue ed allo **stato** di tutte le unità **decide** l'operazione da eseguire ed emette gli **ordini** relativi



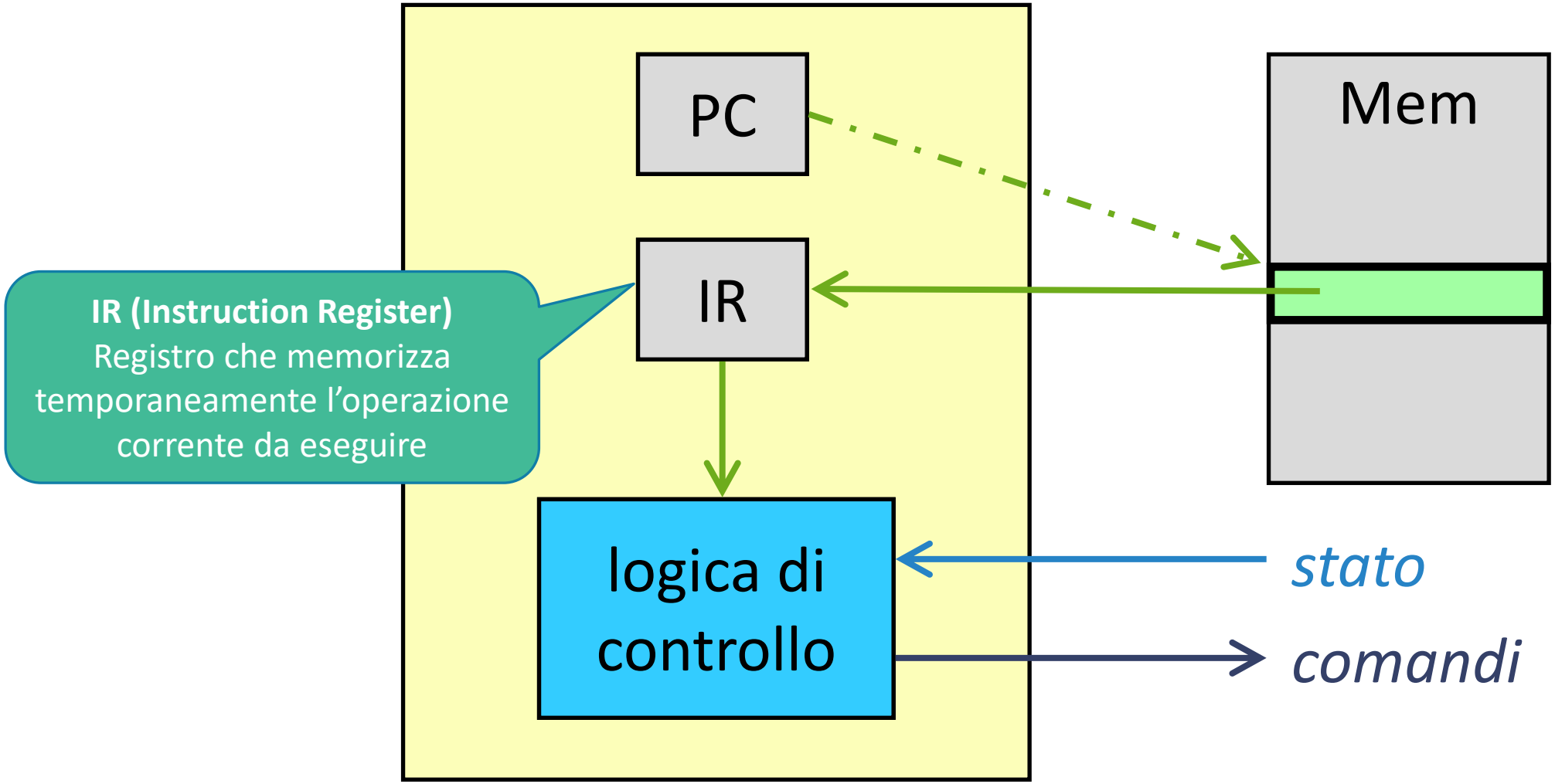
Unità di controllo: schema funzionale



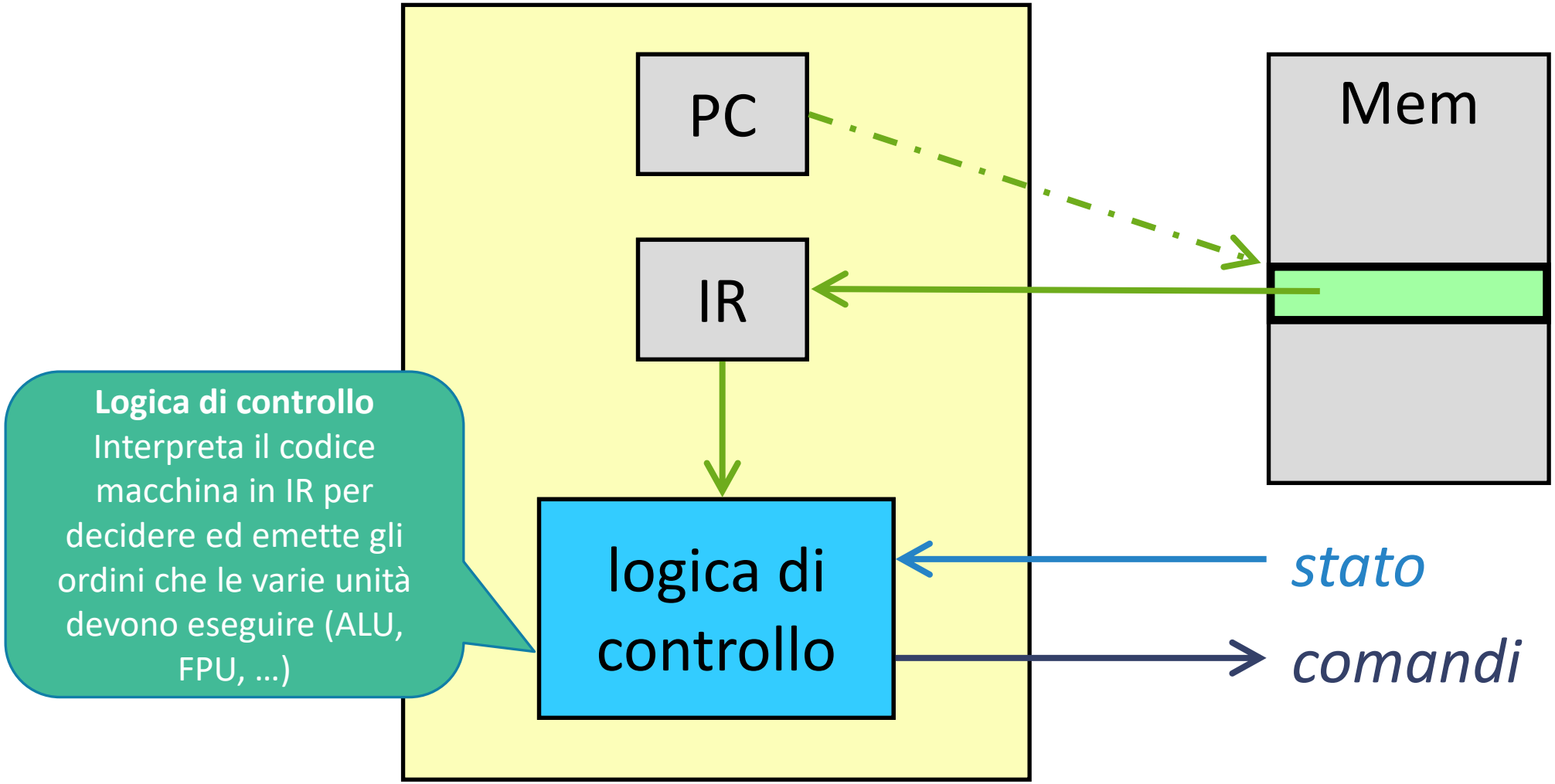
Unità di controllo: schema funzionale



Unità di controllo: schema funzionale



Unità di controllo: schema funzionale



Componenti dell'UC

- **PC** (Program Counter)
  registro che indica sempre l'indirizzo della cella di memoria che contiene la prossima istruzione da eseguire

- **IR** (Instruction Register)
  registro che memorizza temporaneamente l'operazione corrente da eseguire

- Logica di controllo
  interpreta il codice macchina in IR per decidere ed emette gli ordini che le varie unità devono eseguire

Esecuzione di un'istruzione

- **Tre fasi:**

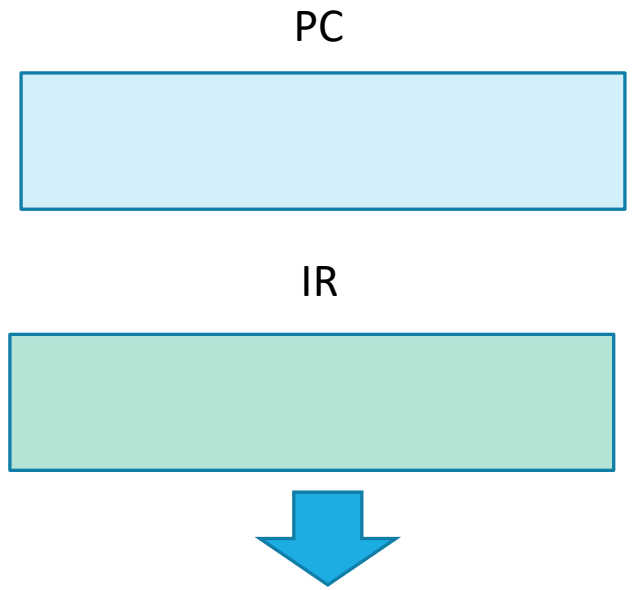
FETCH **IR ← M [PC]**
PC ← PC + 1

DECODE ordini ← **decode(IR)**

EXECUTE attiva i blocchi interessati dall'istruzione

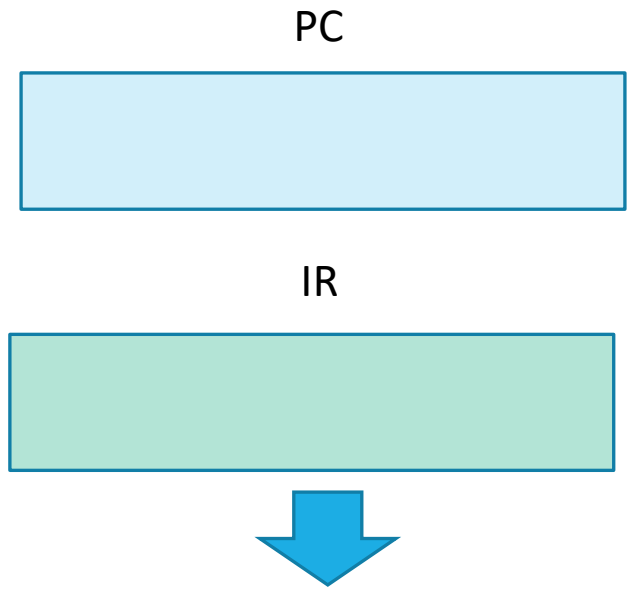
- **IR ← M [PC]:** preleva dalla memoria l'istruzione nella posizione indicata da PC
- **PC ← PC + 1:** incrementa il valore di PC (al passo successivo conterrà la prossima istruzione da eseguire)

Esempio



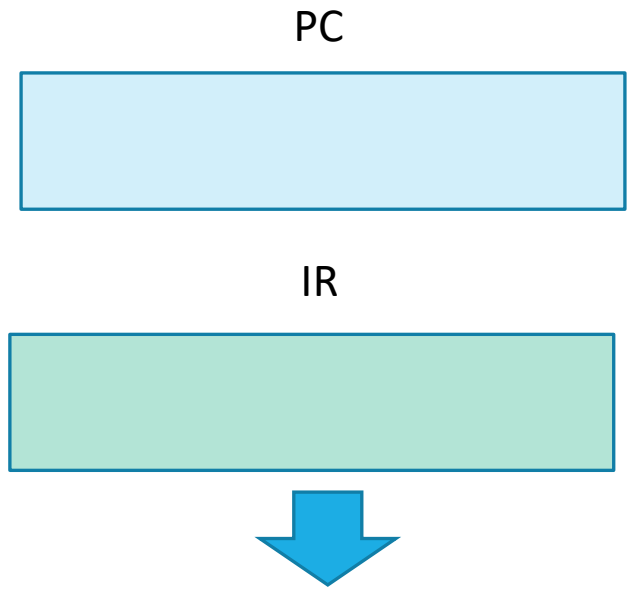
999	…	
1000	**0000……1101**	x=2
1001	1010……1001	y=x
1002	1110……0001	print(x)
1003	…	
	…	

Esempio



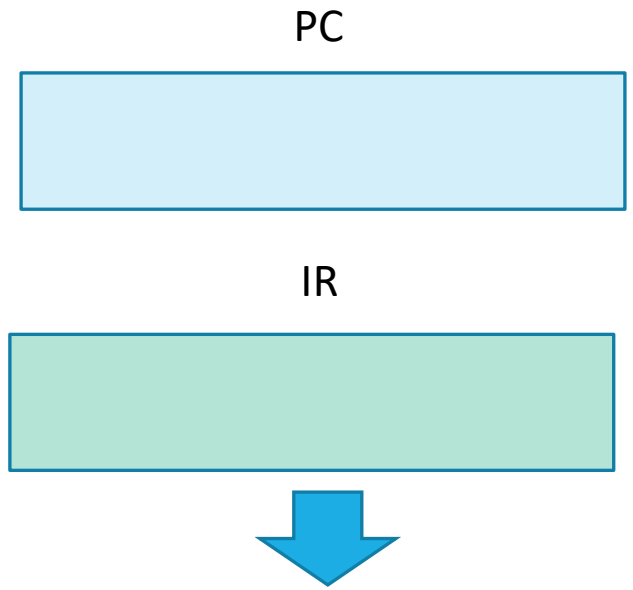
999	…	
1000	0000……1101	x=2
1001	**1010……1001**	y=x
1002	1110……0001	print(x)
1003	…	
	…	

Esempio



999	…	
1000	0000……1101	x=2
1001	1010……1001	y=x
1002	**1110……0001**	print(x)
1003	…	
	…	

Esempio



999	…	
1000	0000……1101	x=2
1001	1010……1001	y=x
1002	1110……0001	print(x)
1003	…	
	…	

Un collegamento ai flowchart…

- Conoscendo ora i blocchi fondamentali, siamo in grado di valutare quali operazioni sono da considerarsi “elementari” (e sono usabili dentro i blocchi di un DDF)

Categoria	Operazioni specifiche
Operazioni aritmetiche (ALU)	+, - , * , /, resto tra numeri interi
Operazioni aritmetiche (FPU)	+, - , * , /, resto tra numeri reali
Operazioni logiche (ALU)	Operazioni su quantita’ logiche (unione, intersezione, negazione)
Confronti (ALU)	<,>,=, <=, >=, !=
CPU + Memoria	Trasferimento dati dalla/verso memoria
CPU + unita’ di I/O (+ Memoria)	Lettura/scrittura da/su dispositivo

Un breve riassunto…

- Sistema di elaborazione =
  Unità di I/O + Unità centrale (CPU) + Memoria

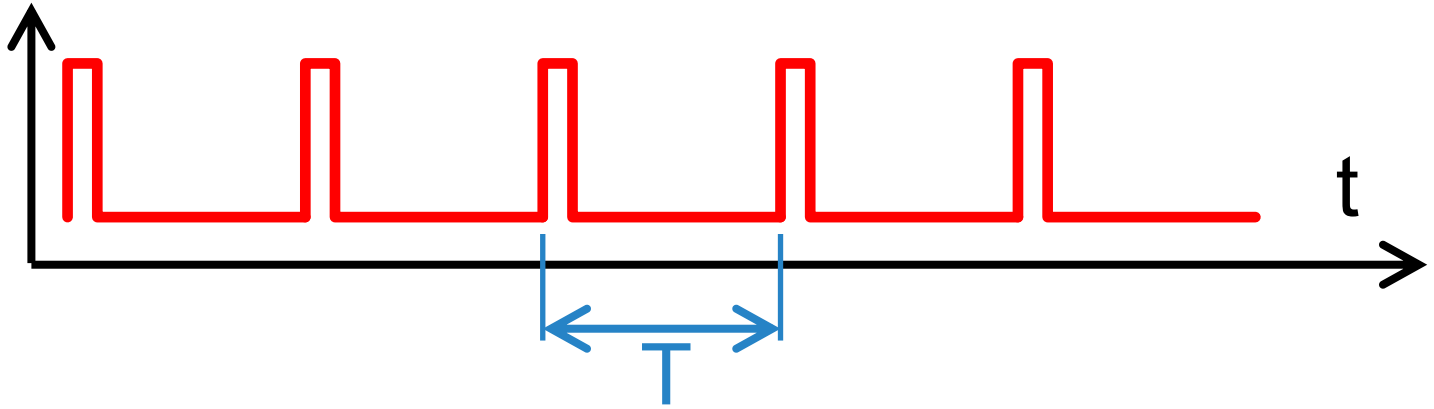
- CPU = Unità operativa + Unità di controllo

- Unità operativa:
  - Svolge i calcoli (contiene i circuiti che li eseguono)
  - Contiene alcuni registri
    (“parcheggi” per i dati da e verso memoria e I/O)

- Unità di controllo:
  - Governa l’esecuzione delle istruzioni (che legge dalla memoria)
  - Procede secondo tre fasi principali

Il clock

- Ogni elaboratore contiene un elemento di temporizzazione (detto clock) che genera un riferimento temporale comune per tutti gli elementi costituenti il sistema di elaborazione.
- T = periodo di clock
  - unità di misura = s
- f = frequenza di clock (= 1/T)
  - unità di misura = s–1 = Hz (cicli/s)



Tempistica delle istruzioni

- Un ciclo-macchina è l'intervallo di tempo in cui viene svolta una operazione elementare ed è un multiplo intero del periodo del clock
- L'esecuzione di un'istruzione richiede un numero intero di cicli macchina, variabile a seconda del tipo di istruzione
  - Esempio

Tipo	n. Cicli
ALU	1
FPU	15
MEM	5
I/O	100

- La prestazione complessiva del programma dipenderà dal mix di istruzioni!!!

Un esempio di calcolo

- **Calcolatore con un tempo di ciclo di 1 ns (1 GHz clock)**
- **Programma che contiene 10^6 operazioni**
  - In base alla distribuzione delle istruzioni la durata media è di **4 cicli/istruzione** (4 ns/istruzione)
- Tempo totale di esecuzione:
  10^6 istruzioni x 3 ns/istruzione = 3ms

Un esempio di calcolo (2)

- Calcolatore con un tempo di ciclo di 1 ns (1 GHz clock)
- Un programma ordina una serie di n valori e impiega un tempo quadratico (proporzionale a n^2) ad ordinarli
  - In base alla distribuzione delle istruzioni la durata media e' di **2 cicli/istruzione** (2 ns/istruzione)
- **Quanto ci vuole ad ordinare 10^6 valori?**
  - Ordinare 10^6 valori = 10^{12} operazioni
  - Tempo totale di esecuzione:
    10^{12} istruzioni x 2 ns/istruzione = 2000 s (~35 minuti!!!)

Importanza dell'efficienza dei programmi

- Numero di istanze eseguite in un dato tempo per algoritmi con diversi complessita' di esecuzione (in relazione al numero di dati coinvolti)

Complessita'	Computer con prestazioni P	Computer con prestazion 1OOO*P
n	N_1	1000 N_1
n logn	N_2	387 N_2
n^2	N_3	31.6 N_3
n^5	N_4	3.98 N_4
2^n	N_5	N_5 +9.97

La memoria

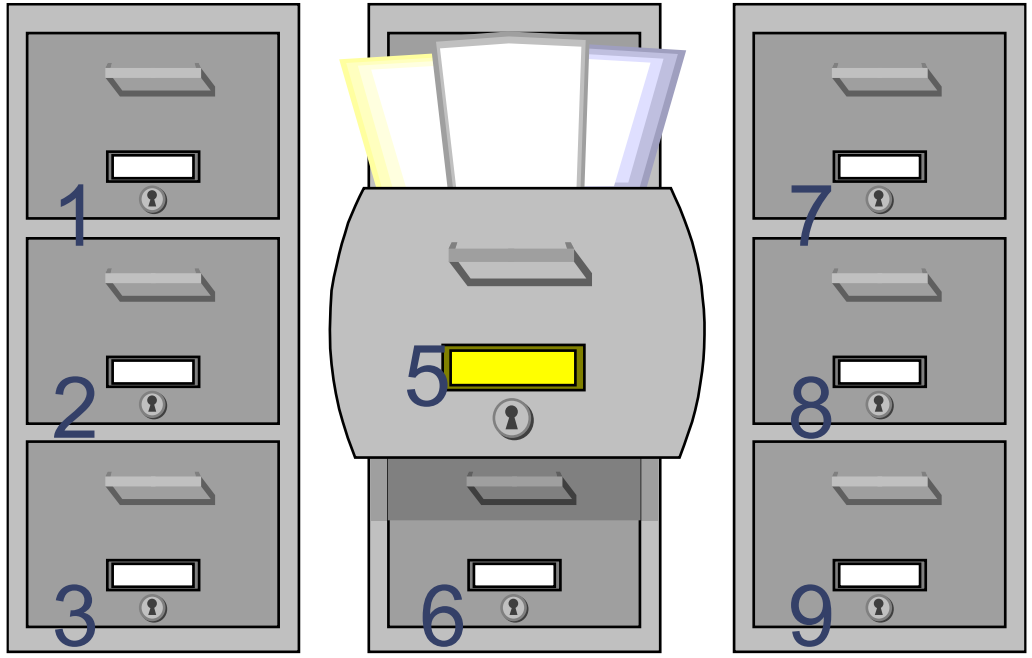
Memoria

- Memorizza i dati e le istruzioni necessarie all'elaboratore per operare
- Perché è necessaria???
  - Per motivi intrinseci di memorizzazione (memoria di massa)
  - Per motivi di efficienza

- Caratteristiche:
  - indirizzamento
  - parallelismo
  - accesso (sequenziale o casuale)
  - Struttura gerarchica

Indirizzamento

- La memoria è organizzata in celle (mimima unità accessibile direttamente). Ad ogni cella di memoria è associato un indirizzo (numerico) per identificarla univocamente.

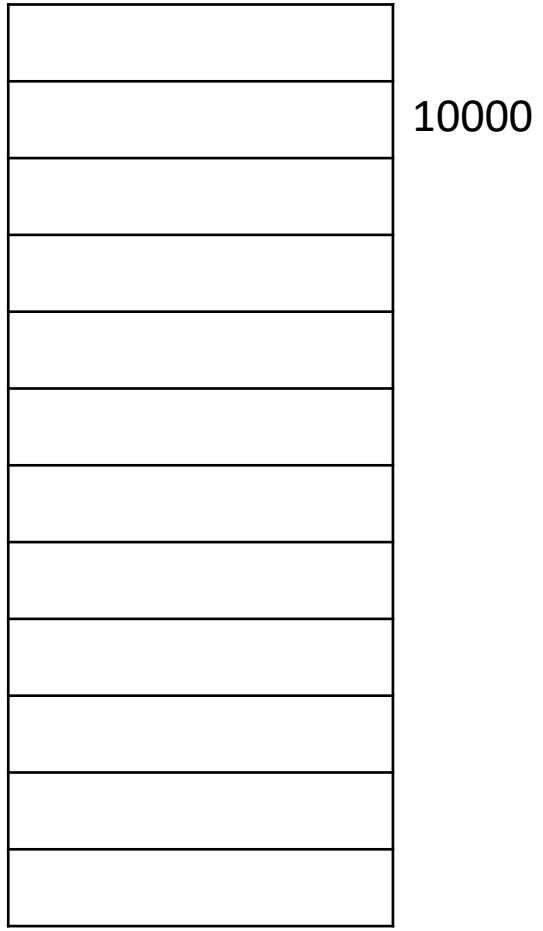


Parallelismo

- Ogni cella di memoria contiene una quantità fissa di bit (word):
  - identica per tutte le celle (di una certa unità di memoria)
  - accessibile con un'unica istruzione
  - è un multiplo del byte
    - Tipicamente la dimensione della cella di memoria coincide con quella dei registri per consistenza e semplicità

Memoria e programmazione

- Cosa succede quando creiamo, leggiamo, scriviamo un dato?



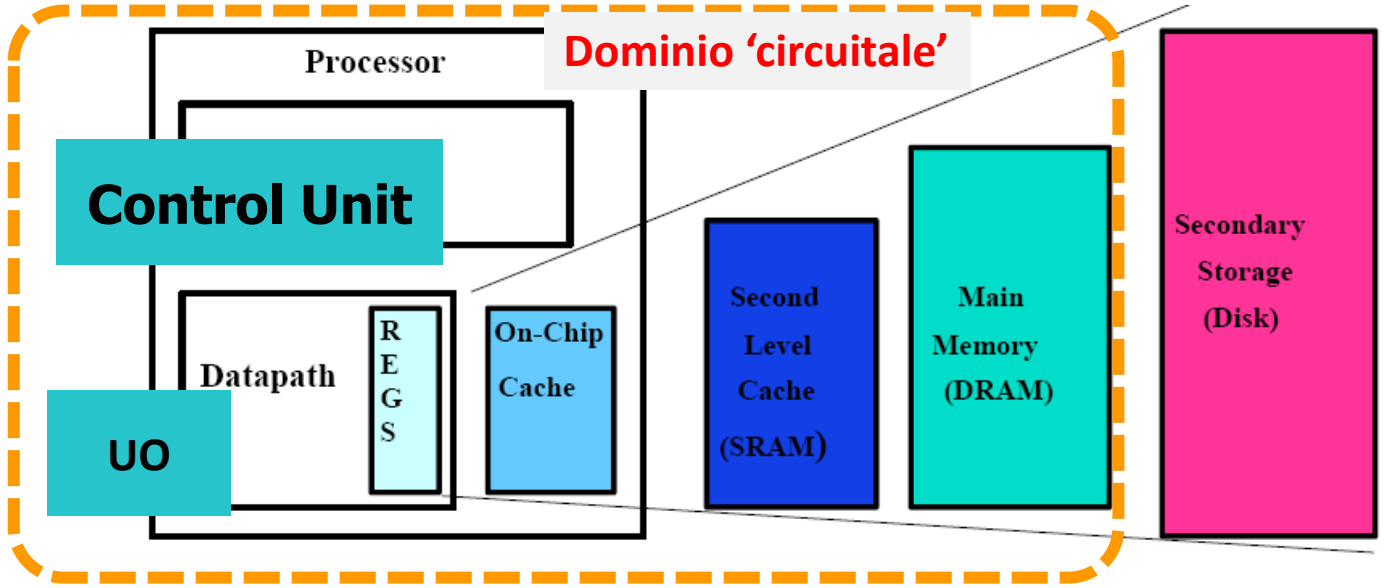
Gerarchia di memoria

- Idealmente la memoria dovrebbe essere
  - Più grande possibile
  - Più economica possibile
  - Più veloce possibile
  - Mantenere le informazioni indefinitamente (anche in assenza di alimentazione) – non volatili

- Una simile memoria non esiste (ancora)
  - Le memorie veloci hanno un costo relativamente alto e sono volatili
  - Le memorie non volatili costano relativamente poco ma sono relativamente lente
- Necessaria una gerarchia di memoria

Gerarchia di memoria

- Per ottimizzare tempo medio di accesso e costo medio, si organizza la memoria a livelli
  - Più vicino al processore le memorie più veloci, costose (e anche volatili, per motivi tecnologici)
  - Più lontano dal processore le memorie più lente, economiche e non volatili



Gerarchia di memoria

- Registri
  - Costituiscono la memoria più veloce, interna al processore
  - Memorizzano temporaneamente dati o istruzioni
  - Sono gestiti dal compilatore (che alloca le variabili ai registri, gestisce i trasferimenti allo spazio di memoria)

- Cache
  - Memoria veloce e di piccole dimensioni posta fra la CPU e la memoria principale
  - Il suo scopo è di velocizzare l'esecuzione dei programmi: è utilizzata per recuperare velocemente dati e programmi che si prevede debbano essere utilizzati nel breve termine

Gerarchia di memoria

- Memoria principale: RAM (Random Access Memory)
  - Accesso casuale
    - Permette l'accesso diretto a qualunque indirizzo di memoria con lo stesso tempo di accesso
    - Si può accedere alle celle in lettura specificandone l'indirizzo
    - Si può accedere alle celle in scrittura specificandone l'indirizzo e il dato da scrivere
  - Volatile
    - Nella memoria RAM vengono copiati (caricati) i programmi che la CPU deve eseguire
    - Una volta chiuso il programma, le modifiche effettuate, se non opportunamente salvate su altre memorie, verranno perse

Gerarchia di memoria

- Memoria principale: RAM (Random Access Memory)
  - Di due tipi:
    - SRAM: Static Random Access Memory
      - Consentono di mantenere le informazioni per un tempo infinito ma, perdono le informazioni in esse contenute se non alimentate da corrente
      - Più veloce e costosa della DRAM
      - Usata per la cache
    - DRAM: Dynamic Random Access Memory
      - Basata su condensatori: dopo un (breve) lasso di tempo il suo contenuto diventa completamente inaffidabile e quindi richiede un refresh periodico

Gerarchia di memoria

- Memoria secondaria (o memoria di massa):
  - La memoria principale non può essere troppo grande a causa del suo costo elevato, e non consente la memorizzazione permanente dei dati (volatilità)

Gerarchia di memoria

- Memoria secondaria (o memoria di massa):
  - La memoria principale non può essere troppo grande a causa del suo costo elevato, e non consente la memorizzazione permanente dei dati (volatilità)
  - Oltre alla memoria principale veloce, volatile, di dimensioni relativamente piccole, i computer hanno una memoria secondaria:
    - Più lenta e meno costosa
    - Con capacità di memorizzazione maggiore ed in grado di memorizzare i dati in modo permanente

Gerarchia di memoria

- Memoria secondaria (o memoria di massa):
  - La memoria principale non può essere troppo grande a causa del suo costo elevato, e non consente la memorizzazione permanente dei dati (volatilità)
  - Oltre alla memoria principale veloce, volatile, di dimensioni relativamente piccole, i computer hanno una memoria secondaria:
    - Più lenta e meno costosa
    - Con capacità di memorizzazione maggiore ed in grado di memorizzare i dati in modo permanente
  - La memoria secondaria viene utilizzata per mantenere tutti i programmi e tutti i dati che possono essere utilizzati dal computer
    - Il processore non può utilizzare direttamente la memoria di massa per l'elaborazione dei dati
    - Quando si vuole eseguire un programma, questo dovrà essere copiato dalla memoria di massa a quella principale (caricamento)

Gerarchia di memoria

- Flash
  - Le chiavette USB, le SD card, e i più recenti hard disk a stato solido sono basati su tecnologia elettronica, ma, a differenza delle RAM, non sono volatili

- Disco rigido
  - Gli hard disk sono dei supporti di plastica o vinile, su cui è depositato del materiale magnetizzabile
  - Nel corso delle operazioni i dischi vengono mantenuti in rotazione a velocità costante e le informazioni vengono lette e scritte da testine
  - Meccanico, quindi lento!

Confronto tra memorie

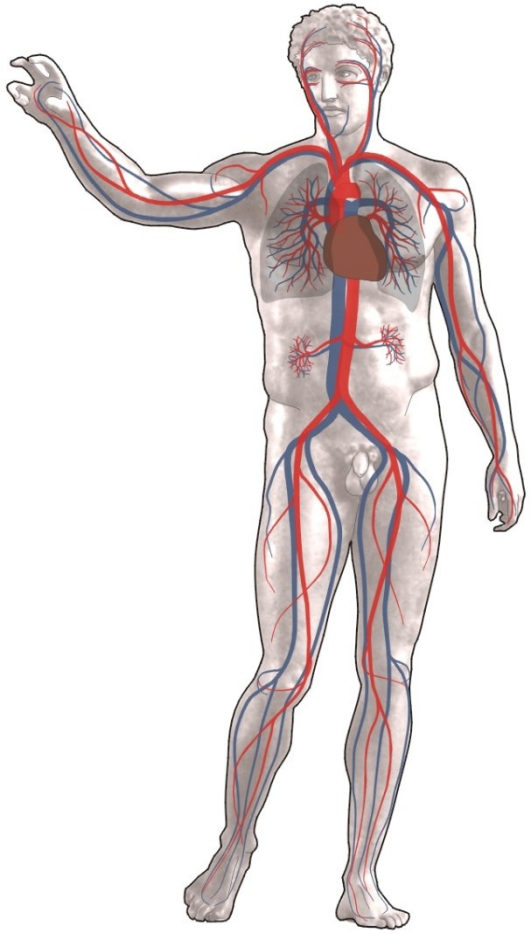
MAGGIORE VELOCITA'
MAGGIORI COSTI

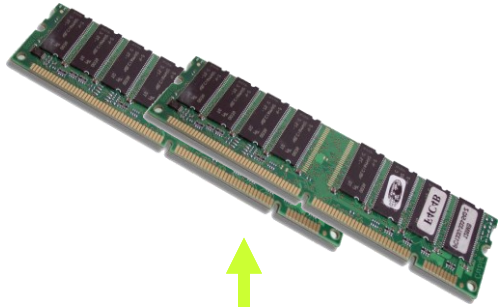
Metrica	SRAM (cache)	DRAM	FLASH	Disco
Dimensione	MB (1-16)	GB (4-16)	GB (16-512)	TB (>1)
Velocità	1-5 ns	50-150 ns	~10ms*	~10ms
Costo	10^{-5} $/byte	10^{-8} $/byte	10^{-9} $/byte	10^{-10} $/byte
Persistenza	Volatile	Volatile	Non volatile	Non volatile

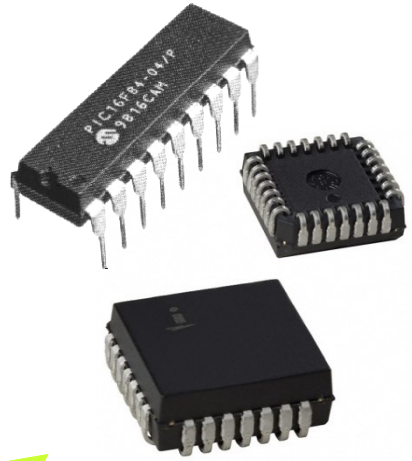
MAGGIORE CAPACITA'
MAGGIORI TEMPI DI ACCESSO

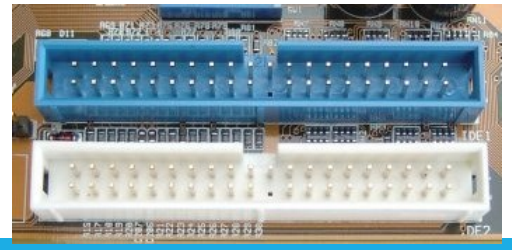
Le interconnessioni (bus)

I Bus (sistema circolatorio del PC)



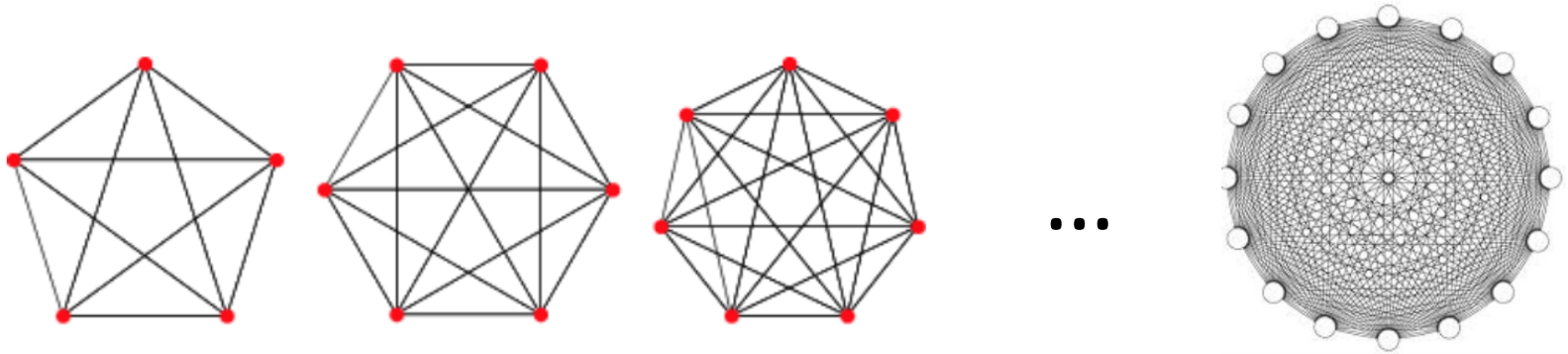






I bus

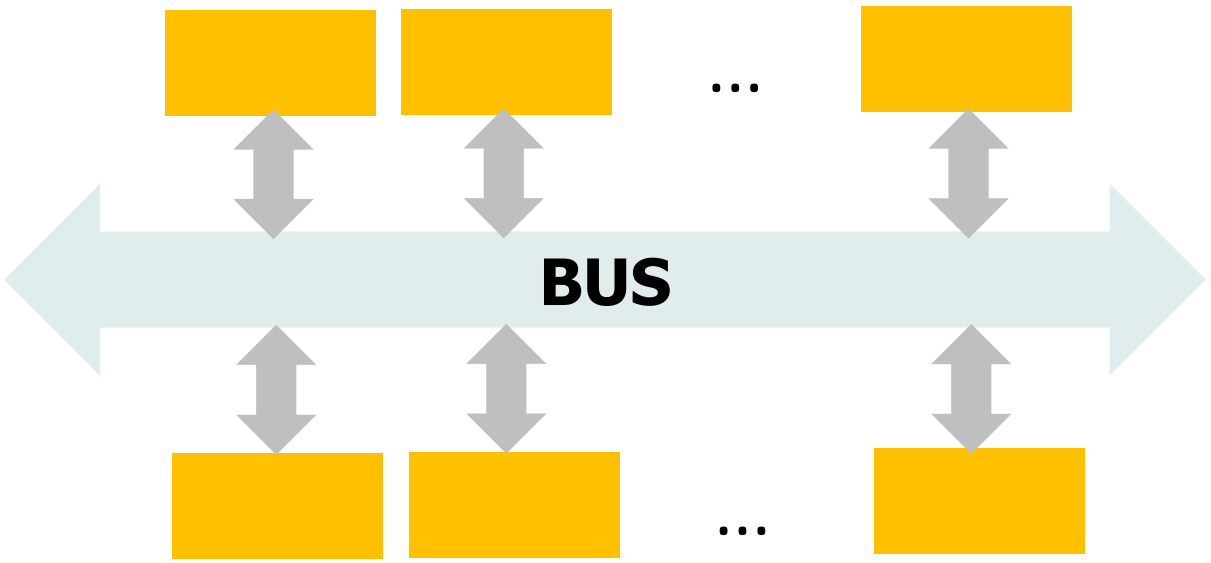
- Come connettere n unità (CPU con memorie e vari controllori di I/O) in modo efficiente?
- La connessione punto-punto non è pratica!
  - Il numero di connessioni cresce con il quadrato del numero di unita' da connettere!



n=5 n=6 n=7 n=16

I bus

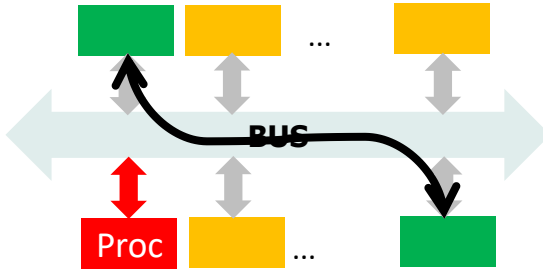
- Usiamo un’unica linea di connessione per connettere tutti i componenti
  - Fisicamente “agganciati” a questa linea



I bus

- Vantaggi:
  - costi ridotti di produzione
  - Estendibilità (scalabilità)
    - aggiunta di nuovi dispositivi molto semplice
  - Standardizzabilità
    - regole per la comunicazione da parte di dispositivi diversi

- Svantaggi:
  - Lentezza
    - utilizzo in mutua esclusione del bus
  - Sovraccarico del processore (CPU)
    - funge da “master” sul controllo del bus



Caratteristiche di un bus

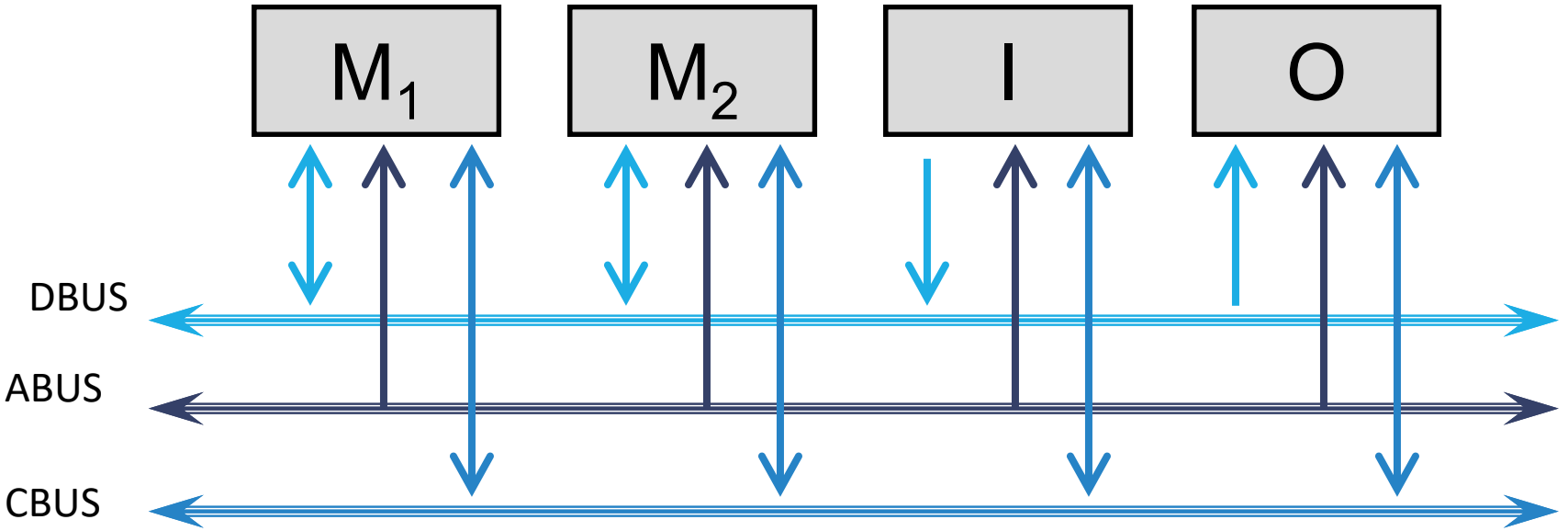
- Trasporto di un solo dato per volta

- Frequenza = n. di dati trasportati al secondo
- Ampiezza = n. di bit di cui è costituito un singolo dato

- Se mal dimensionato, potrebbe essere un collo di bottiglia

Tipi fondamentali di bus

- Un singolo bus è suddiviso in tre “sotto bus”, detti:
  - bus dati (DBus)
  - bus degli indirizzi (ABus)
  - bus di controllo (CBus)



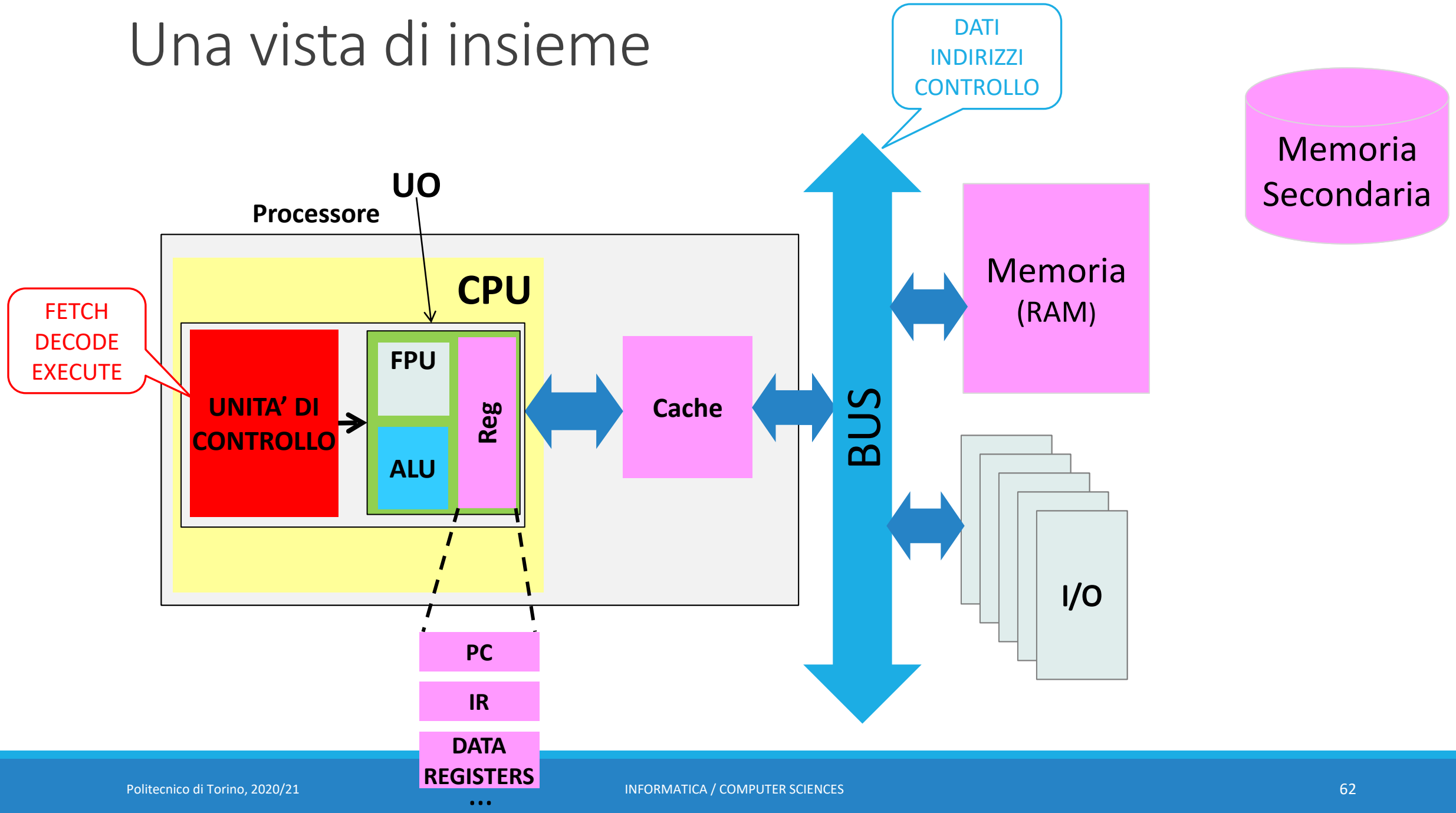
Massima memoria interna (fisicamente presente)

- La dimensione dell'Abus determina il max numero di celle di memoria indirizzabili
- La dimensione del Dbus "indica" la dimensione di una cella di memoria

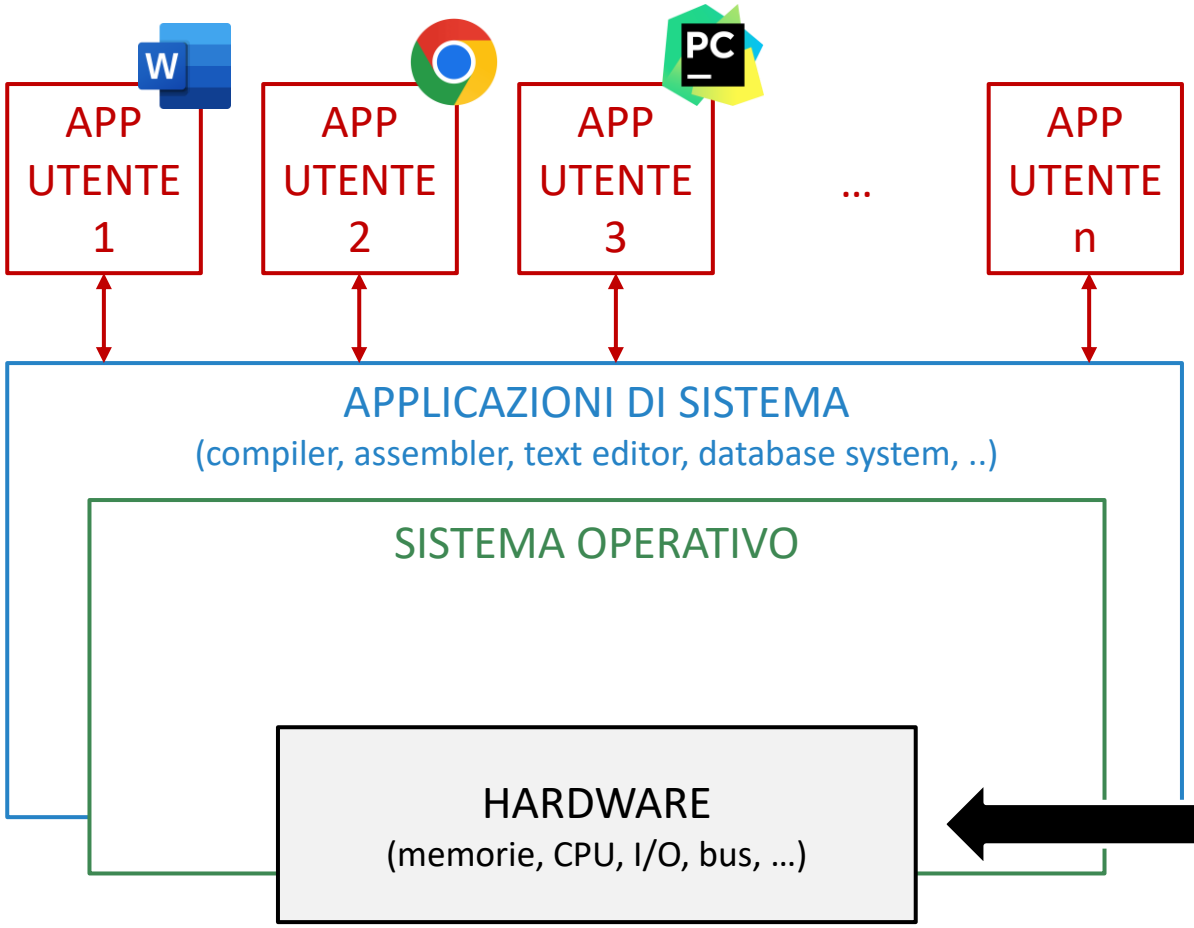
- **max mem = $2^{|Abus|}$ x |Dbus| bit**

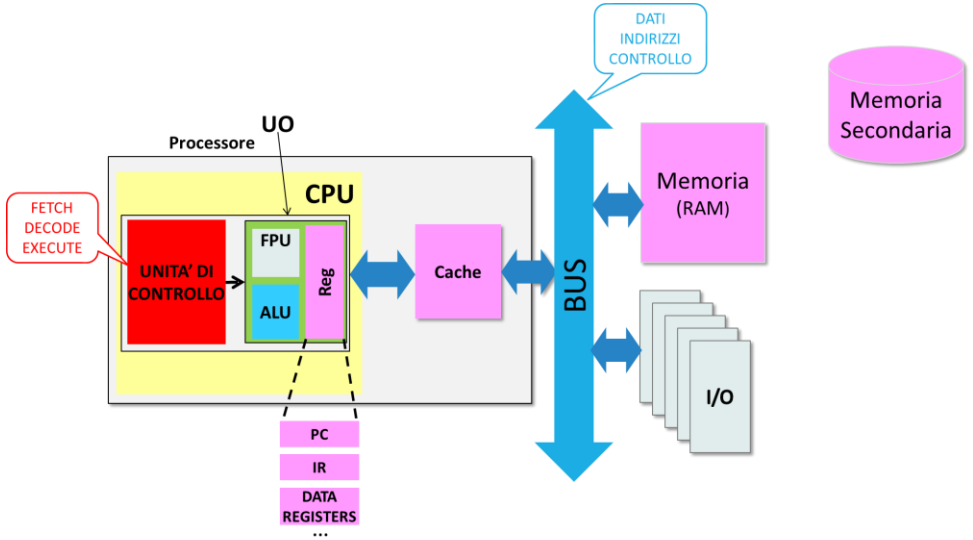
- Esempio (Abus da 20 bit, Dbus da 16 bit):
  - max mem = 2^{20} x 2 byte = 2 MB
  - ossia 1 M celle di memoria, ognuna da 2 byte

Una vista di insieme



Un po' di prospettiva…





Sistema Operativo

- Gestisce le risorse hardware e software della macchina, fornendo servizi di base ai software applicativi

- Obiettivi
  - Eseguire comandi e programmi (rendere più facile la soluzione di problemi)
    - Controlla il funzionamento e le operazioni dei dispositivi, e l'esecuzione dei programmi utente
  - Rendere il sistema più facile da utilizzare
  - Usare l'hardware in modo efficiente
    - Astrae i vari dispositivi del sistema (ad es. i dischi sono visti attraverso i driver e manipolati attraverso il relativo filesystem)
    - Permette la gestione delle risorse in maniera semplificata

Sistema Operativo

- Moduli e servizi tipici di un SO
  - Interprete dei comandi
  - Gestione dei processi
  - Gestione della memoria principale
  - Gestione della memoria secondaria
  - Gestione dei dispositivi di I/O
  - Gestione file e file system
  - Implementazione dei meccanismi di protezione
  - Gestione delle reti e sistemi distribuiti

Sistema Operativo

- Gestione dei processi
  - Un processo (unità attiva) è un programma (unità passiva) in esecuzione
  - Per essere eseguito richiede risorse (CPU, memoria, perferiche, etc.)
  - Il SO deve Creare, sospendere e cancellare un processo
- Gestione dei dispositivi di I/O
  - I dispositivi di I/O non possono essere gestiti direttamente dagli utenti (complessità, driver, condivisione, etc.)
  - Il SO deve:
    - Nascondere i dettagli di tali dispositivi agli utenti fornendo un'interfaccia generica tra ogni dispositivo e il relativo driver
    - Fornire operatività sui dispositivi

Sistema Operativo

- Gestione della memoria principale
  - Dati e istruzioni sono immagazzinate nella memoria principale durante l'esecuzione di un programma
    - Logicamente la memoria è un vettore di elementi (word)
  - Il SO deve:
    - Organizzare l'uso della memoria (quali word sono utilizzate e quali sono libere)
    - Decidere quali processi allocare/deallocare dalla memoria
    - Ottimizzare l'accesso ai dati da parte della CPU

- Gestione della memoria secondaria
  - Dato che la memoria centrale è volatile e piccola, i dati sono contenuti in maniera permanente su memoria di massa
  - Il SO deve:
    - Organizzare le informazioni nello spazio disponibile
    - Allocare/deallocare lo spazio richiesto
    - Gestire lo spazio libero
    - Ottimizzare le scheduling delle operazioni di R/W

Un po' di prospettiva…