L’architettura del calcolatore (Prima parte)

Percorso di Preparazione agli Studi di Ingegneria

Università degli Studi di Brescia

Docente: Massimiliano Giacomin

Calcolatore astratto e reale

- **Concetto astratto** di calcolatore: cos'è in sé [cfr. cap.1 del libro di testo]
  - Concetto di problema (classe di domande omogenee, alle quali si possa dare risposta con una procedura uniforme), istanza, soluzione
  - Concetto di algoritmo (specifica attraverso una sequenza di istruzioni come produrre una soluzione per ogni istanza)
  - Il calcolatore come **esecutore universale di algoritmi**

- Come "in pratica" i calcolatori attuali sono costruiti: dobbiamo analizzare l'**architettura** del calcolatore
- CAPIRLA (abbastanza) bene è scopo del corso!!!!!

Offerta speciale!

PC Desktop

Intel Pentium 4 524 3.06 GHz

RAM 2048 MB

HD 250 GB

DVD+CDRW

Scheda audio on board

Sistema Operativo Windows Vista

Monitor LCD 19”

699 E

799 E

Notebook

Intel Pentium M 740 1.73 GHz

2 MB L2 RAM 1024 MB

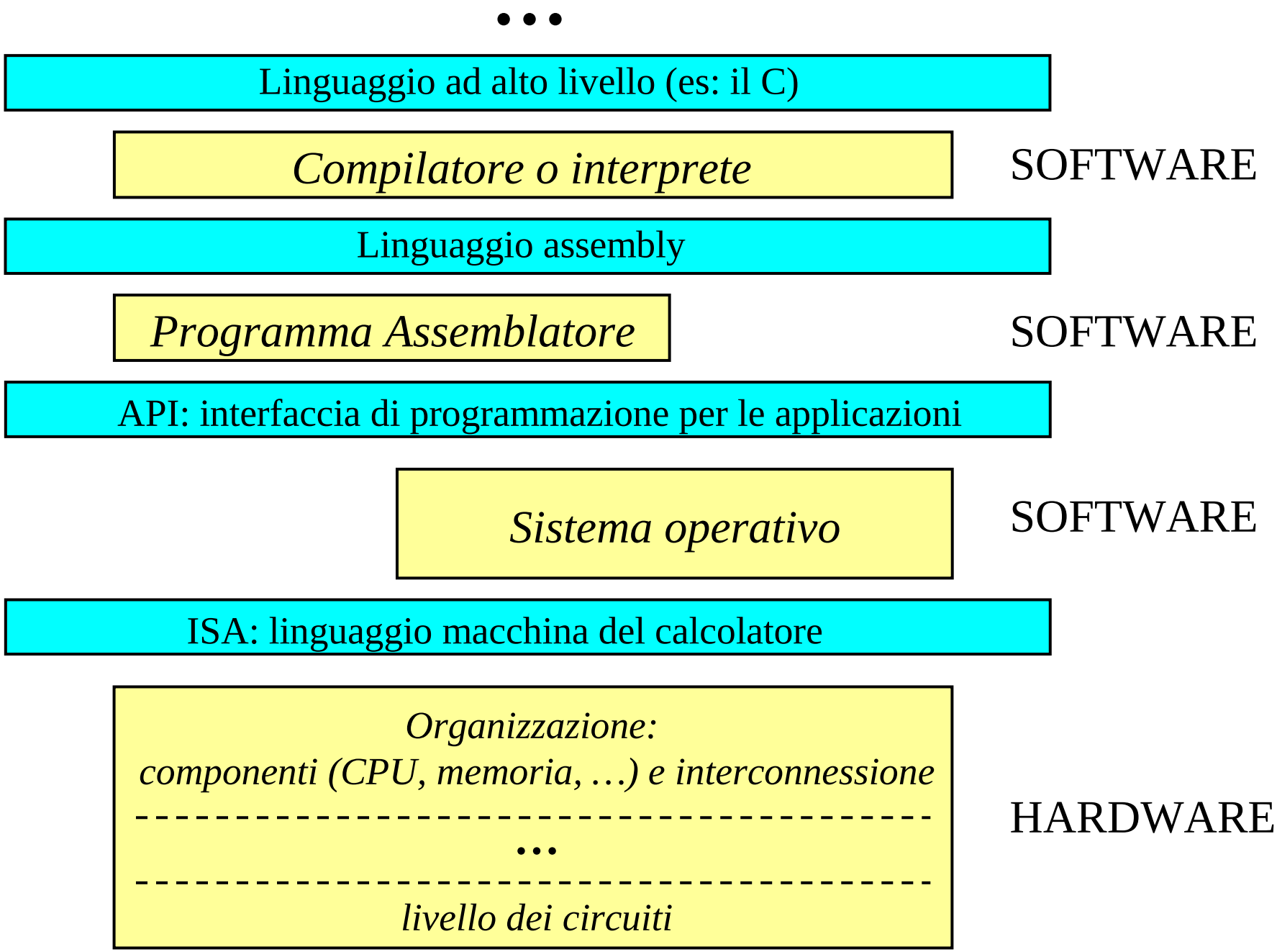
Display 15,4” TFT WXGA TrueBrite

HD 100 GB

DVD/CD-RW

LAN 10/100 - Modem 56k - Wi-Fi 802.11b/g

3 USB 2.0 - IEEE 1394


• • •
Linguaggio ad alto livello (es: il C)
Compilatore o interprete
SOFTWARE
Linguaggio assembly
Programma Assemblatore
SOFTWARE
API: interfaccia di programmazione per le applicazioni
Sistema operativo
SOFTWARE
ISA: linguaggio macchina del calcolatore
Organizzazione:
componenti (CPU, memoria, …) e interconnessione
• • •
livello dei circuiti
HARDWARE

Hardware e software

- **Hardware** (lett. “ferramenta”): i componenti fisici del calcolatore

- **Software** (lett. “oggetto morbido”): l’insieme dei programmi che vengono eseguiti dal calcolatore (accezione “ristretta”)

- NB: distinzione non netta (dipende da costo, velocità, evoluzioni previste del sistema, ecc.)

ISA: linguaggio macchina del calcolatore

Organizzazione:
componenti (CPU, memoria, …) e interconnessione
- -
•••
- -
livello dei circuiti

HARDWARE

- Tecnologie elettroniche:
  - elementi fondamentali: transistor
  - sono considerati due livelli di tensione (alta/bassa)
- Tecnologie magnetiche:
  - memorie costituite da materiale magnetizzabile (es: HD)
  - due stati di polarizzazione (positiva/negativa)
- Tecnologie ottiche:
  - materiali con prop. ottiche rilevate da raggio laser (es: CD)
  - due stati (es: assenza o presenza di un pit)

ISA: linguaggio macchina del calcolatore

Organizzazione:
componenti (CPU, memoria, …) e interconnessione
- -
•••
- -
livello dei circuiti

HARDWARE

A questi due stati si associano convenzionalmente i valori “0” e “1”

P. es. +5 V — Valore 1

0 V — Valore 0

CONSEGUENZA:

Il linguaggio macchina del calcolatore è binario!

00101010 1010001001 01010100001 00000….

- A livello ISA, abbiamo quindi a disposizione due simboli:

  Alfabeto binario = {0, 1}

- I simboli 0 e 1 sono detti **cifre binarie** o

  BIT (*Binary digIT*)

- Ad esempio, sentirete parlare di “registri” a 32 bit…

  00000000101000010000000000011000

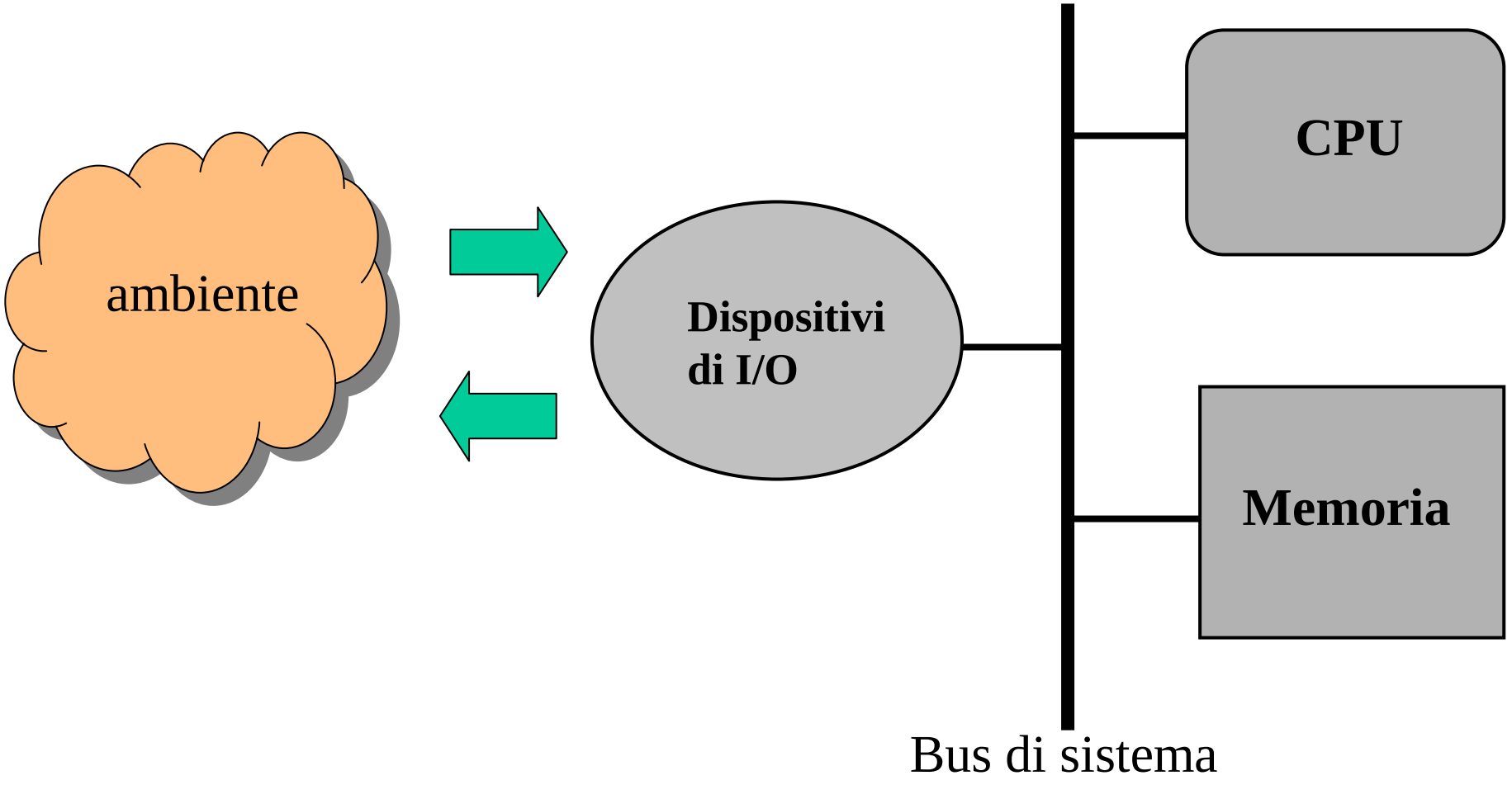
- Abbiamo quindi il problema di rappresentare tutte le informazioni di interesse (numeri, testi, immagini, filmati, ecc.) in linguaggio binario: ci occuperemo della ***codifica binaria*** dell’informazione

ISA: linguaggio macchina del calcolatore

Alcune elementari nozioni sull'organizzazione del calcolatore a livello hardware…

L’architettura di Von Neumann

CPU = *Central Processing Unit (Unità centrale)*
detta oggi Microprocessore o processore



Funzioni dei componenti principali nella Architettura di Von Neumann

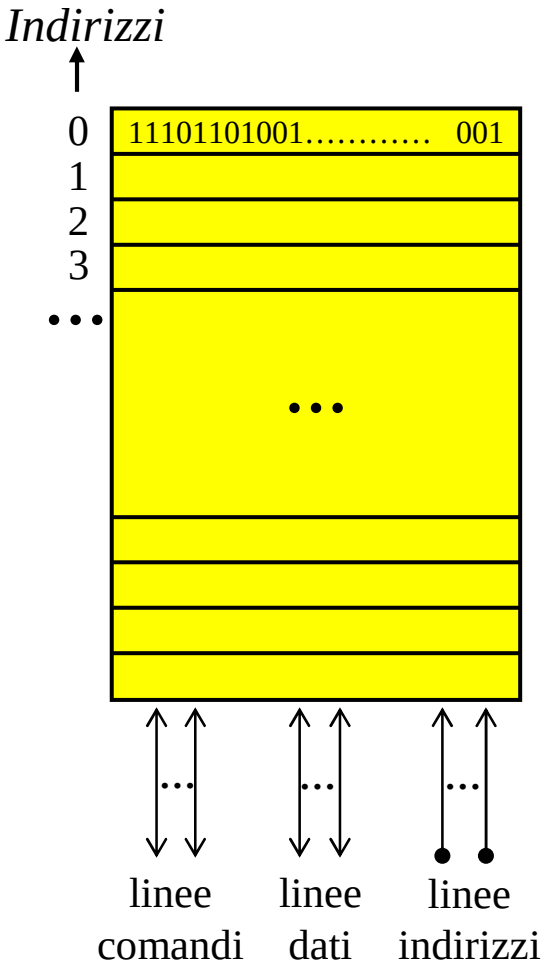
- La ***memoria centrale*** memorizza:
  - le ***istruzioni*** (binarie!) del programma da eseguire
  - i ***dati e i risultati*** (binari!) elaborati dal programma

  [sistema a ***programma memorizzato***]

- L'***unità centrale*** esegue le istruzioni contenute in memoria ***in modo sequenziale***, ripetendo ciclicamente i seguenti passi:
  - Prelievo (fetch) dell'istruzione da eseguire
  - Decodifica (decode) dell'istruzione
  - Esecuzione (execute) delle operazioni previste

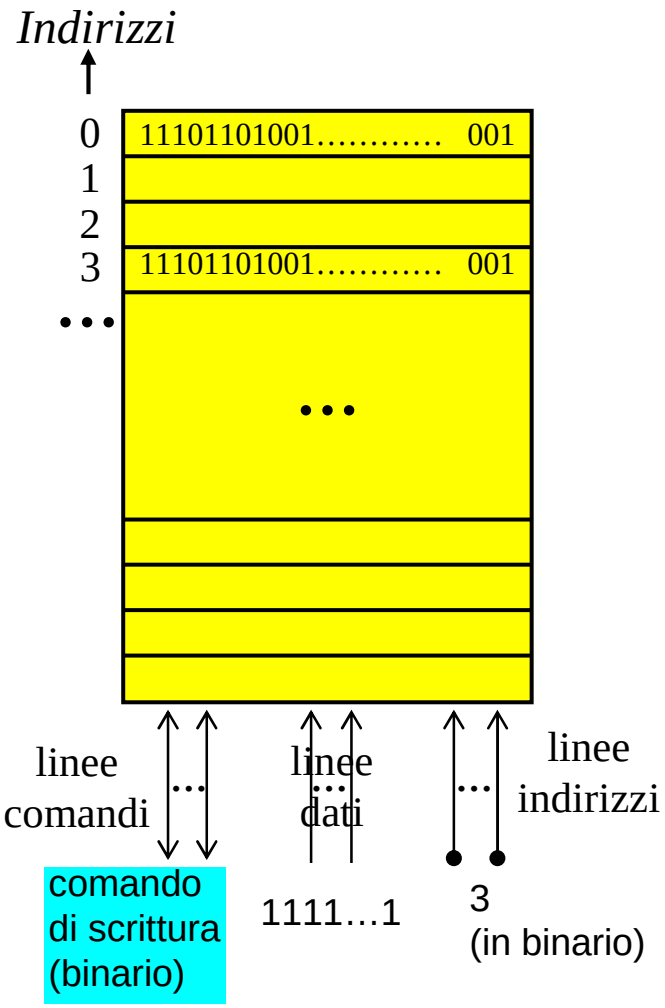
- Il ***bus di sistema*** interconnette i diversi componenti

LA MEMORIA CENTRALE

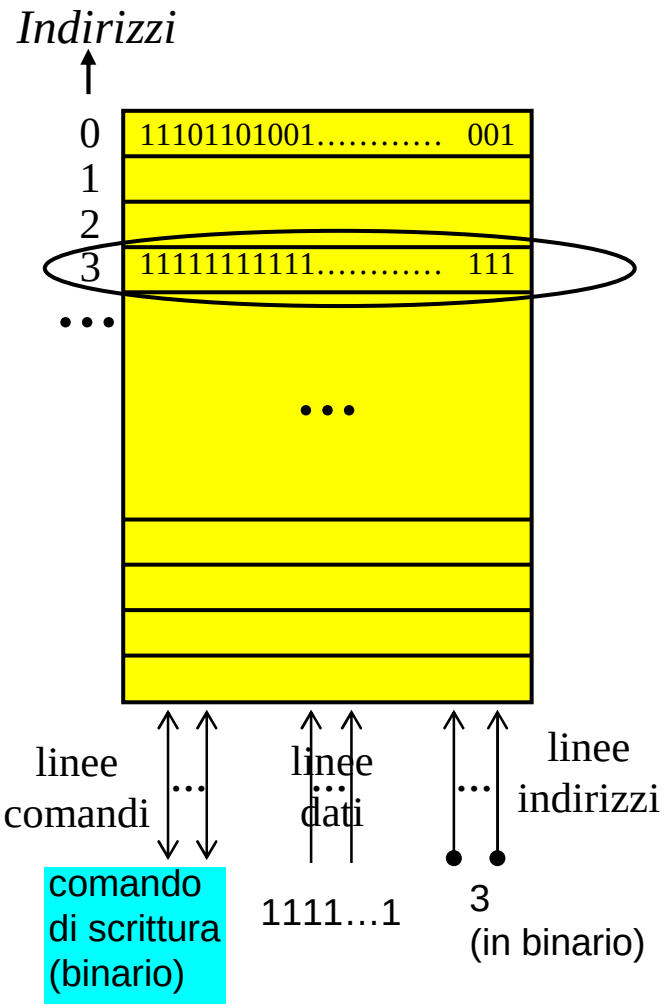


- Un insieme di ***parole di memoria*** consecutive, ciascuna identificata da un ***indirizzo***
- Ogni parola memorizza una sequenza di *n* bit, dove *n* è lo stesso per tutte le parole e dipende dal calcolatore (es: 16, 32, 64 bit)
- Due operazioni possibili: ***lettura*** e ***scrittura*** di una parola all'indirizzo specificato ("cancellazione" non ha senso!)

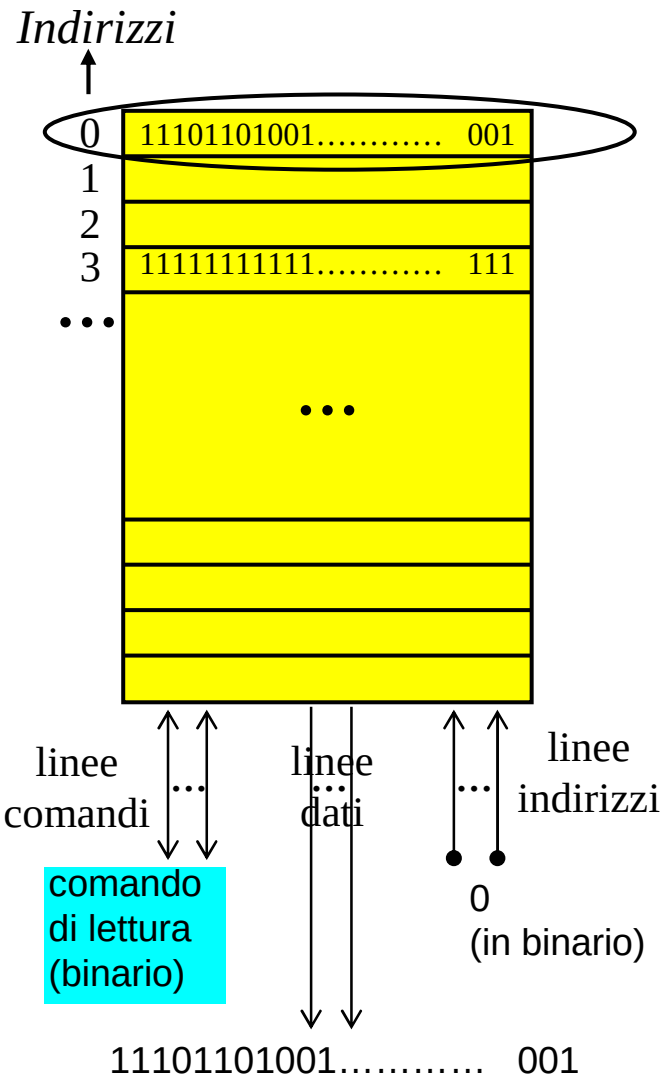
ESEMPIO: scrittura (nella parola di memoria) all'indirizzo 3 del valore (binario) 11111111….1



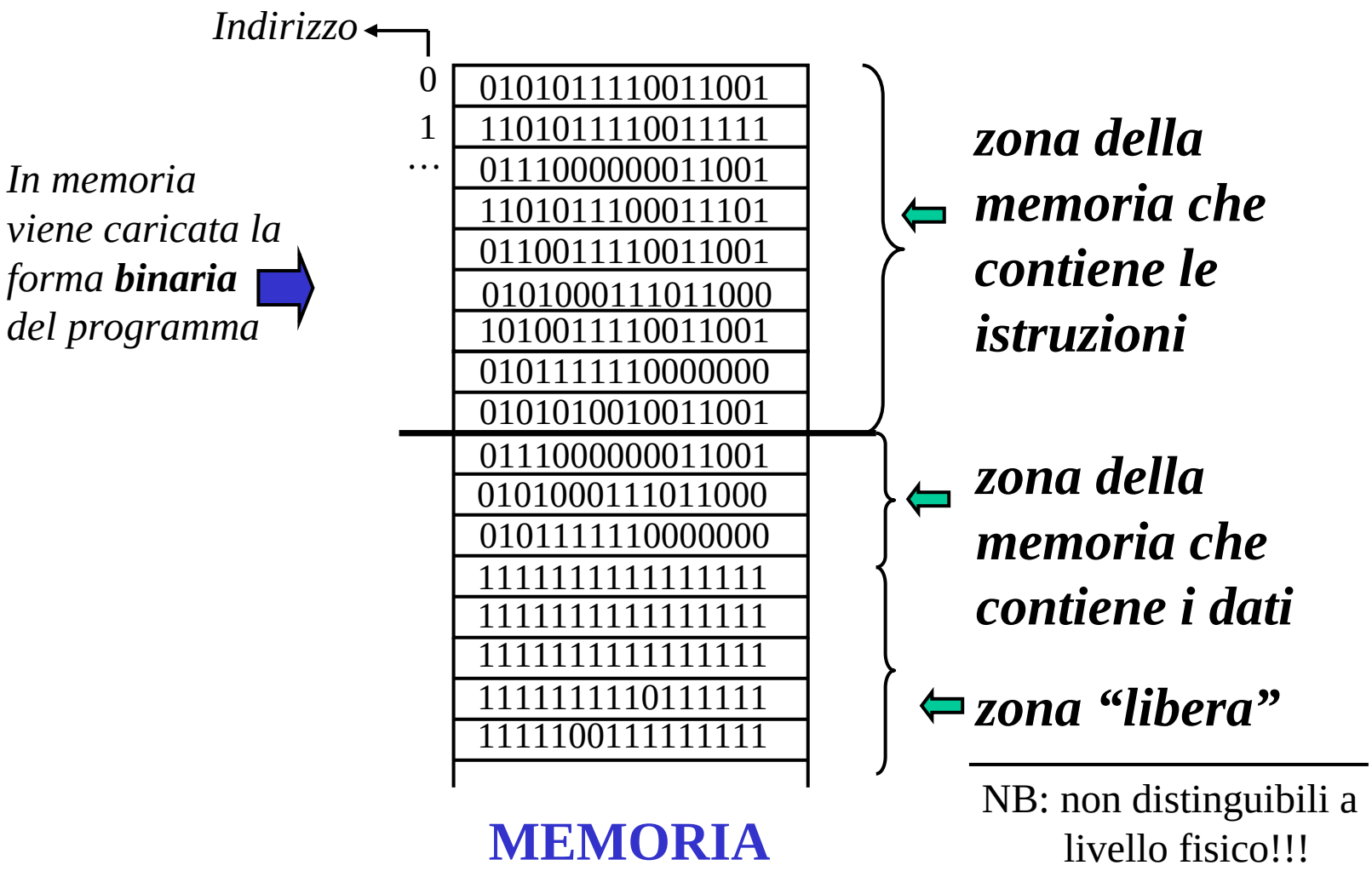
ESEMPIO: scrittura (nella parola di memoria) all'indirizzo 3 del valore (binario) 11111111….1



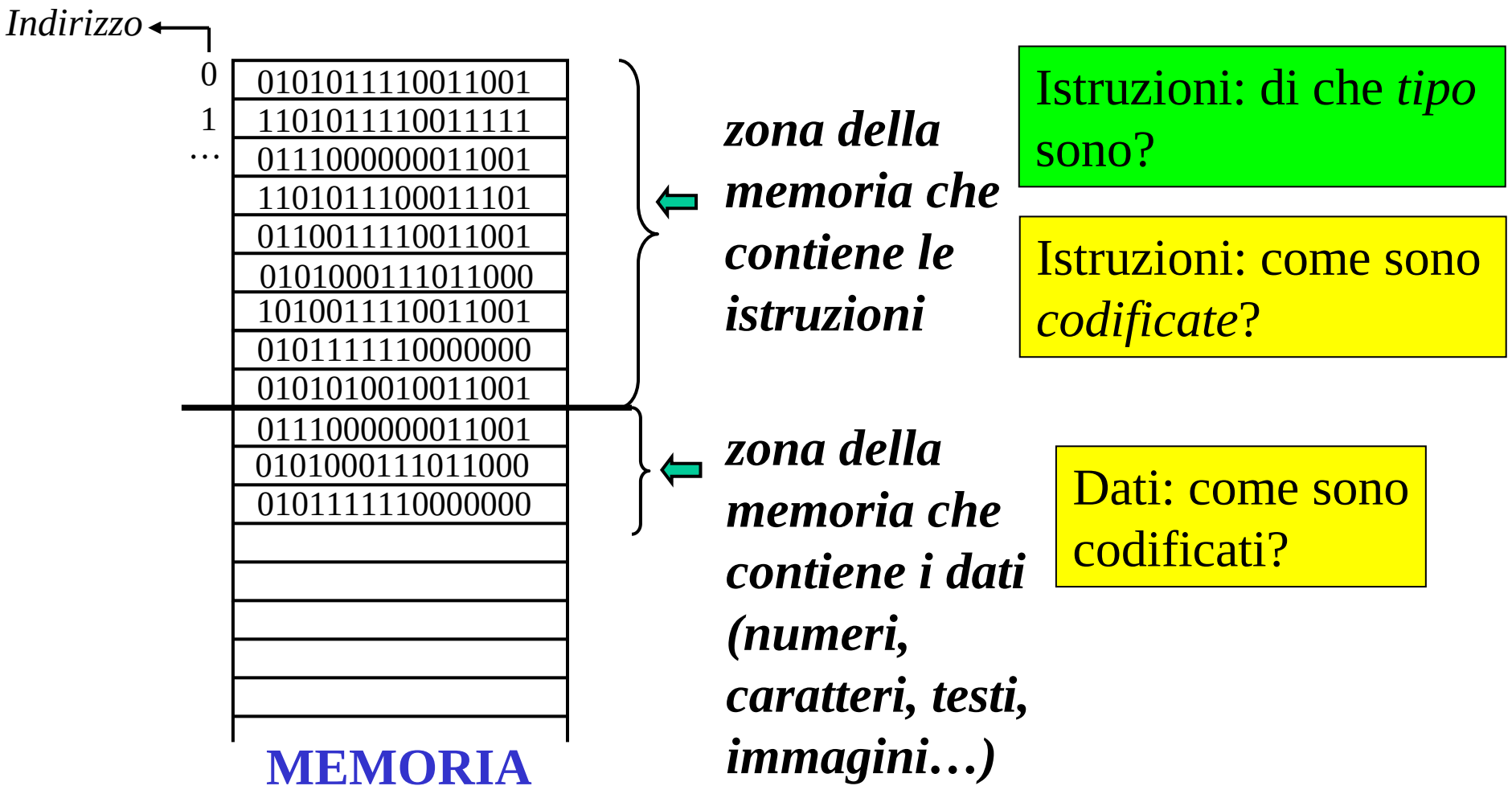
ESEMPIO: lettura (dalla parola di memoria) all'indirizzo 0
del valore (binario) contenuto



Programma e dati in memoria



Programma e dati in memoria: domande lecite



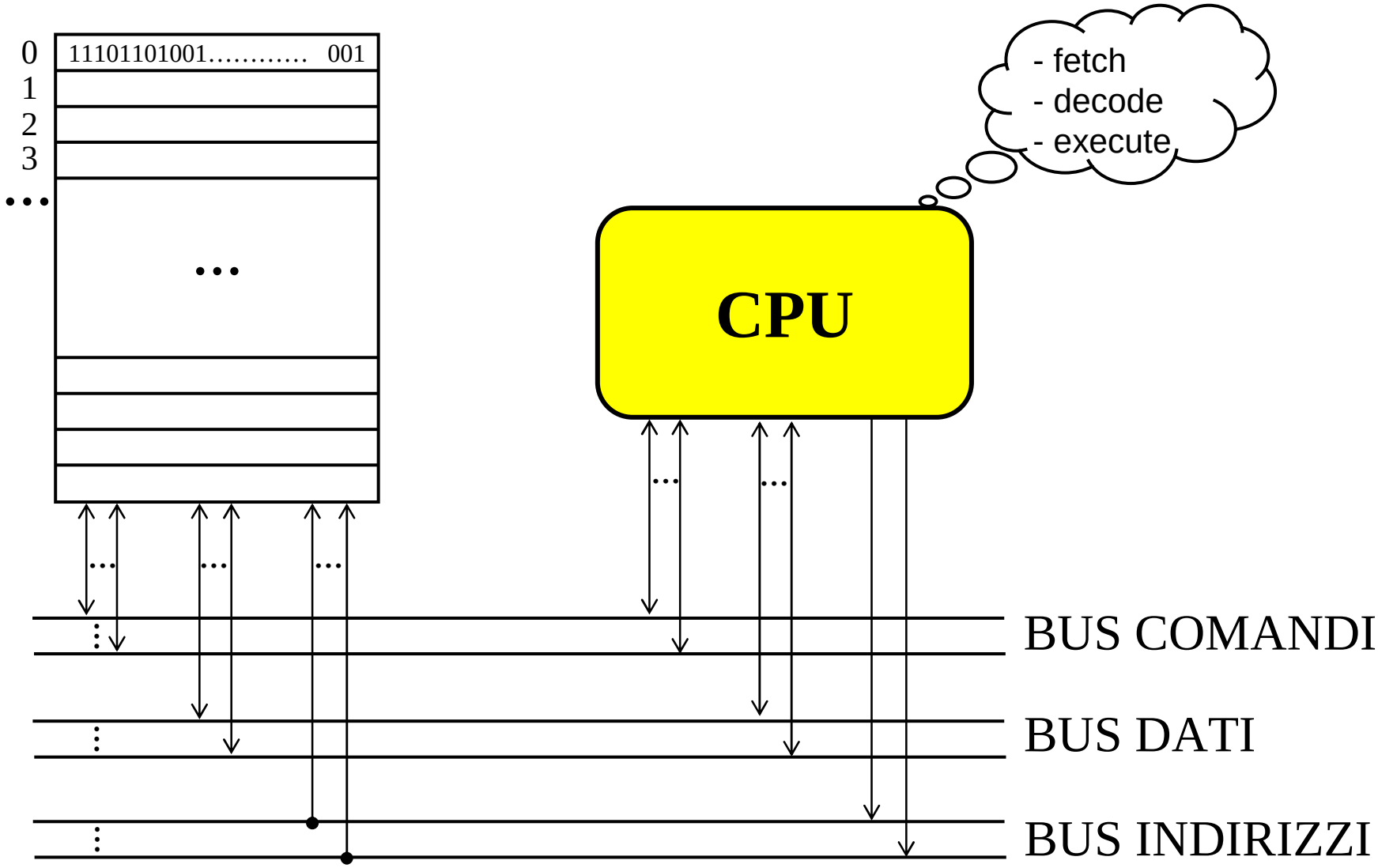
RISPONDIAMO ALLA PRIMA DOMANDA…

IL BUS DI SISTEMA

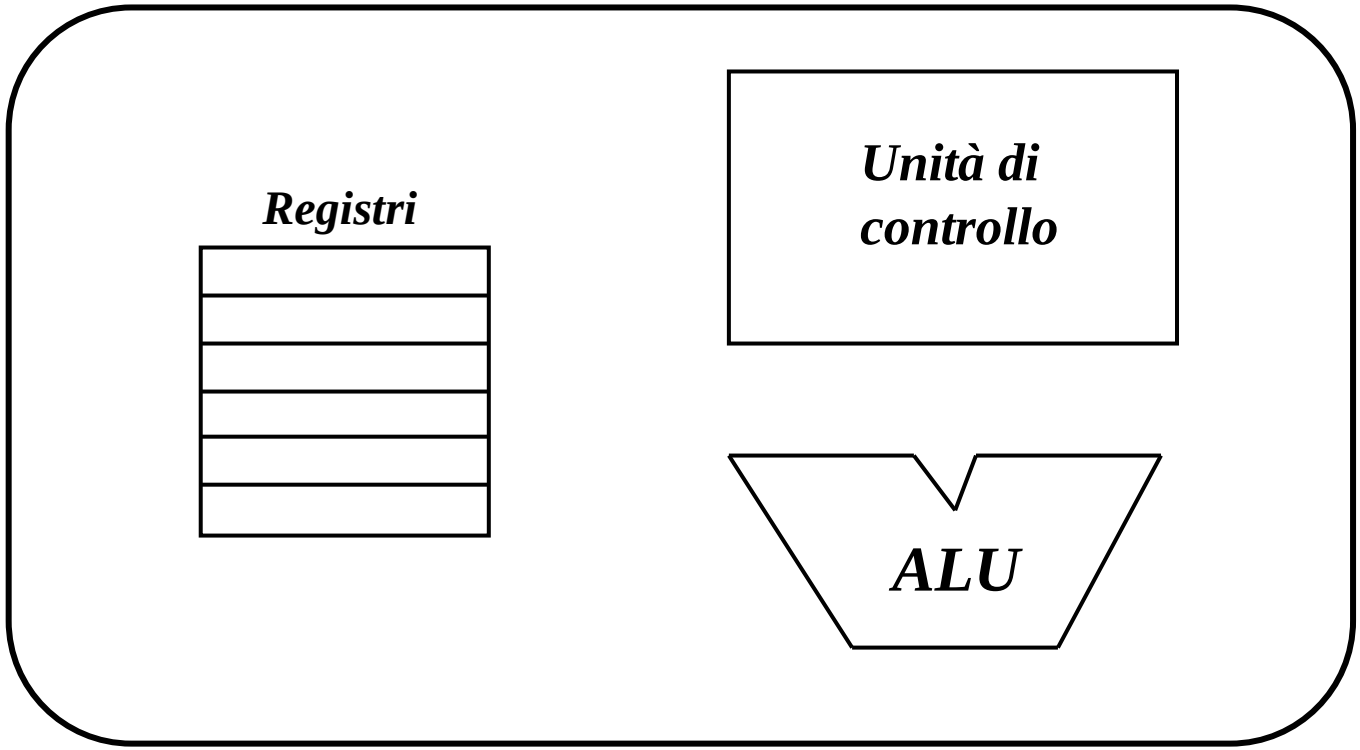
- Insieme di collegamenti (linee) che permettono di trasferire dati da più sorgenti a più destinazioni (componenti del calcolatore)

- Il bus può essere suddiviso funzionalmente in:
  - Bus dati (trasferisce i dati scambiati tra componenti)
  - Bus indirizzi (selezionano le parole della memoria o le interfacce di ingresso-uscita coinvolte nel trasferimento)
  - Bus comandi o “di controllo” (segnali di controllo che regolano le operazioni del sistema di elaborazione)

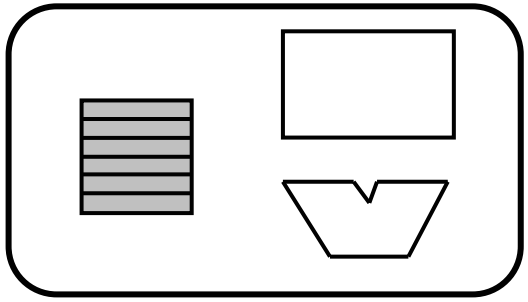
L'UNITA' CENTRALE (CPU - processore)



Componenti della CPU



Processori commerciali: Intel Pentium, AMD Atlon, Cell, ...



I registri della CPU

- Sono **celle di memoria a n bit**

0 00000000101000010000000000011000

1 00001100101000010000000000011000

…

Identificati da un numero

- Quanti bit per registro?: esistono processori a 16, 32, 64 bit…
- Quanti registri? Dipende dal processore

POSSONO MEMORIZZARE OPERANDI DELLE ISTRUZIONI

→ Costituiscono la “memoria a breve termine” del calcolatore

Il linguaggio macchina

- Linguaggio macchina: costituito da **istruzioni macchina, eseguite dalla CPU**
- Ogni CPU ha un proprio linguaggio macchina (**ISA – Instruction Set Architecture**): per esempio, le istruzioni dei processori Intel X86 sono diverse da quelle del processore MIPS
  - esistono CPU di marca diversa con diversa struttura fisica che risultano **compatibili** (es. Intel e AMD)
- Le istruzioni del linguaggio macchina sono costituite da stringhe di bit, suddivise in:
  - **Codice operativo** → tipo istruzione
  - **Operandi** → indicano i dati su cui l'istruzione opera (sorgenti) e dove memorizzare il risultato (destinazione)

Tipologie di istruzioni

- ***Istruzioni aritmetico-logiche e di manipolazione di bit*** (Elaborazione dati)
  - Somma, Sottrazione, Divisione, …
  - And, Or, Xor, …
  - Maggiore, Minore, Uguale, Minore o uguale, …
- ***Istruzioni di trasferimento***
  - Trasferimento dati e istruzioni tra CPU e memoria
- ***Istruzioni di controllo***
  - Salti condizionati
  - Salti incondizionati
- ***Istruzioni di ingresso e uscita***
  - Trasferimento dati e istruzioni tra CPU e dispositivi di ingresso/uscita

Faremo riferimento ad un esempio di ISA "load-store"

- ***Istruzioni aritmetico-logiche e di manipolazione di bit***
  - Lavorano solo su registri

- ***Istruzioni di trasferimento***
  - Trasferiscono valori da memoria a registri (**load**) o da registri a memoria (**store**)

Quindi, per elaborare più valori in memoria, il programma dovrà:

1. Trasferire i valori dalla memoria ad opportuni registri (load)
2. Elaborare i valori dei registri con opportune istruzioni aritmetico-logiche
3. Trasferire i valori ottenuti in memoria (store)

Esempi di istruzioni aritmetiche (prese a riferimento)

```
add     $r1, $r3, $r4     # somma il contenuto di $r3 col contenuto
                          # di $r4 e poni il risultato in $r1
sub     $r0, $r1, $r2     # sottrai dal contenuto di $r1 il contenuto di
                          # $r2 e poni il risultato in $r0
```

Stato registri: prima dell'esecuzione

$r0	4
$r1	3
$r2	5
$r3	10
$r4	2

…

Esempi di istruzioni aritmetiche (prese a riferimento)

```
add     $r1, $r3, $r4     # somma il contenuto di $r3 col contenuto
                          # di $r4 e poni il risultato in $r1
sub     $r0, $r1, $r2     # sottrai dal contenuto di $r1 il contenuto di
                          # $r2 e poni il risultato in $r0
```

Stato registri: prima dell'esecuzione

$r0	4
$r1	3
$r2	5
$r3	10
$r4	2

…

Stato registri: dopo la prima istruzione

$r0	4
$r1	**12**
$r2	5
$r3	10
$r4	2

…

Esempi di istruzioni aritmetiche (prese a riferimento)

```
add     $r1, $r3, $r4     # somma il contenuto di $r3 col contenuto
                          # di $r4 e poni il risultato in $r1
sub     $r0, $r1, $r2     # sottrai dal contenuto di $r1 il contenuto di
                          # $r2 e poni il risultato in $r0
```

Stato registri: prima dell'esecuzione

$r0	4
$r1	3
$r2	5
$r3	10
$r4	2

…

Stato registri: dopo la prima istruzione

$r0	4
$r1	**12**
$r2	5
$r3	10
$r4	2

…

Stato registri: dopo la seconda istruzione

$r0	**7**
$r1	**12**
$r2	5
$r3	10
$r4	2

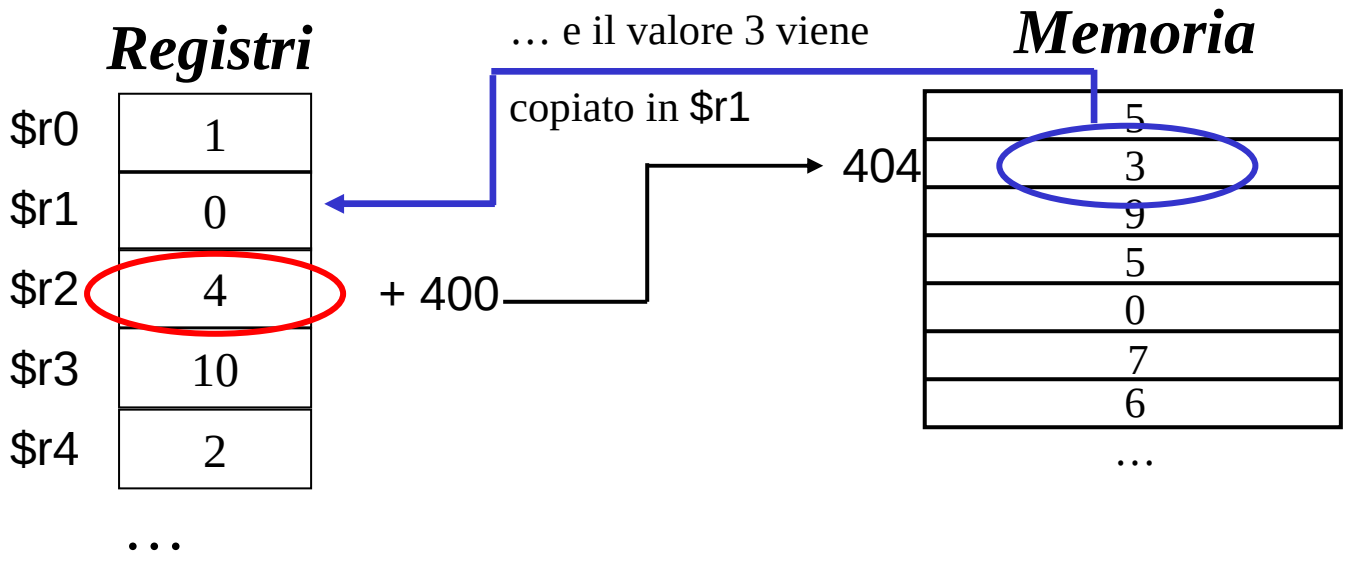
…

Esempi di istruzioni di trasferimento

- Istruzione lw (*load word*)

  lw $r1, $r2, 400

  significa: carica nel registro $r1 il contenuto della cella di memoria il cui indirizzo si trova sommando 400 al contenuto di $r2

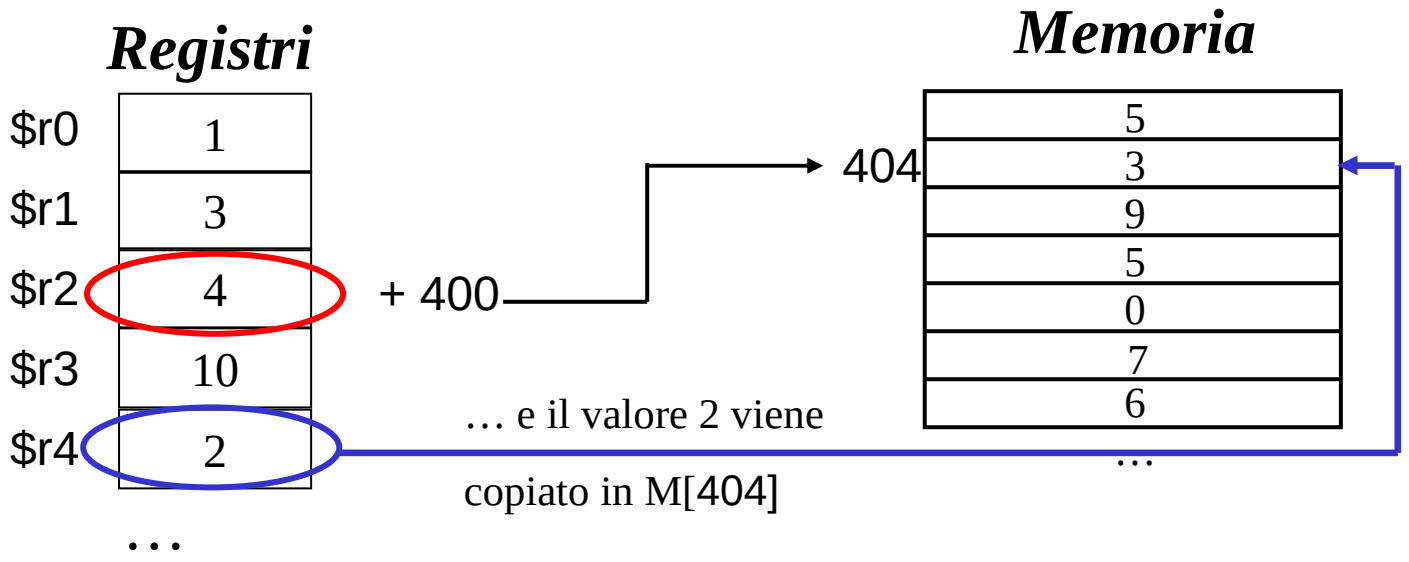


Esempi di istruzioni di trasferimento

- Istruzione sw (*store word*)

  sw $r4, $r2, 400

  significa: salva il contenuto del registro $r4 nella cella di memoria il cui indirizzo si trova sommando 400 al contenuto di $r2



ESERCIZIO PER CASA

Supponendo che nel registro $r0 sia contenuto il valore 0, scrivere un programma che somma i valori delle due celle di memoria di indirizzo rispettivamente 100 e 200, ponendo il risultato nella cella di memoria di indirizzo 300.

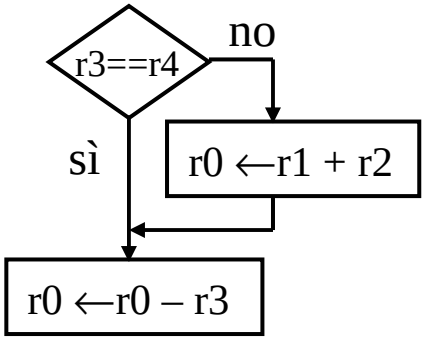
Istruzioni di controllo

- Normalmente, le istruzioni sono eseguite dalla CPU in sequenza
- Talvolta si vuole alterare questo comportamento, soprattutto per poter prendere delle decisioni
- Istruzioni di “salto”:
  - salto incondizionato (***jump***): salta ad un’istruzione
  - salto condizionato (***branch***):
    salta ad una istruzione solo se si verifica una determinata condizione (ad esempio se due registri hanno lo stesso valore)

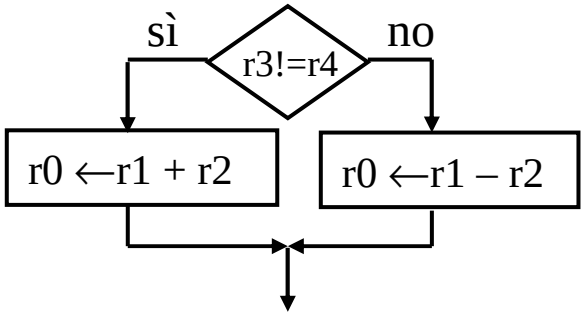
Esempi di istruzioni di controllo

- Salto incondizionato *j* (jump)
- Salto condizionato *beq* (branch if equal) e *bne* (branch if not equal)

```
        beq     $r3, $r4, L1
        add     $r0, $r1, $r2
L1:     sub     $r0, $r0, $r3
```

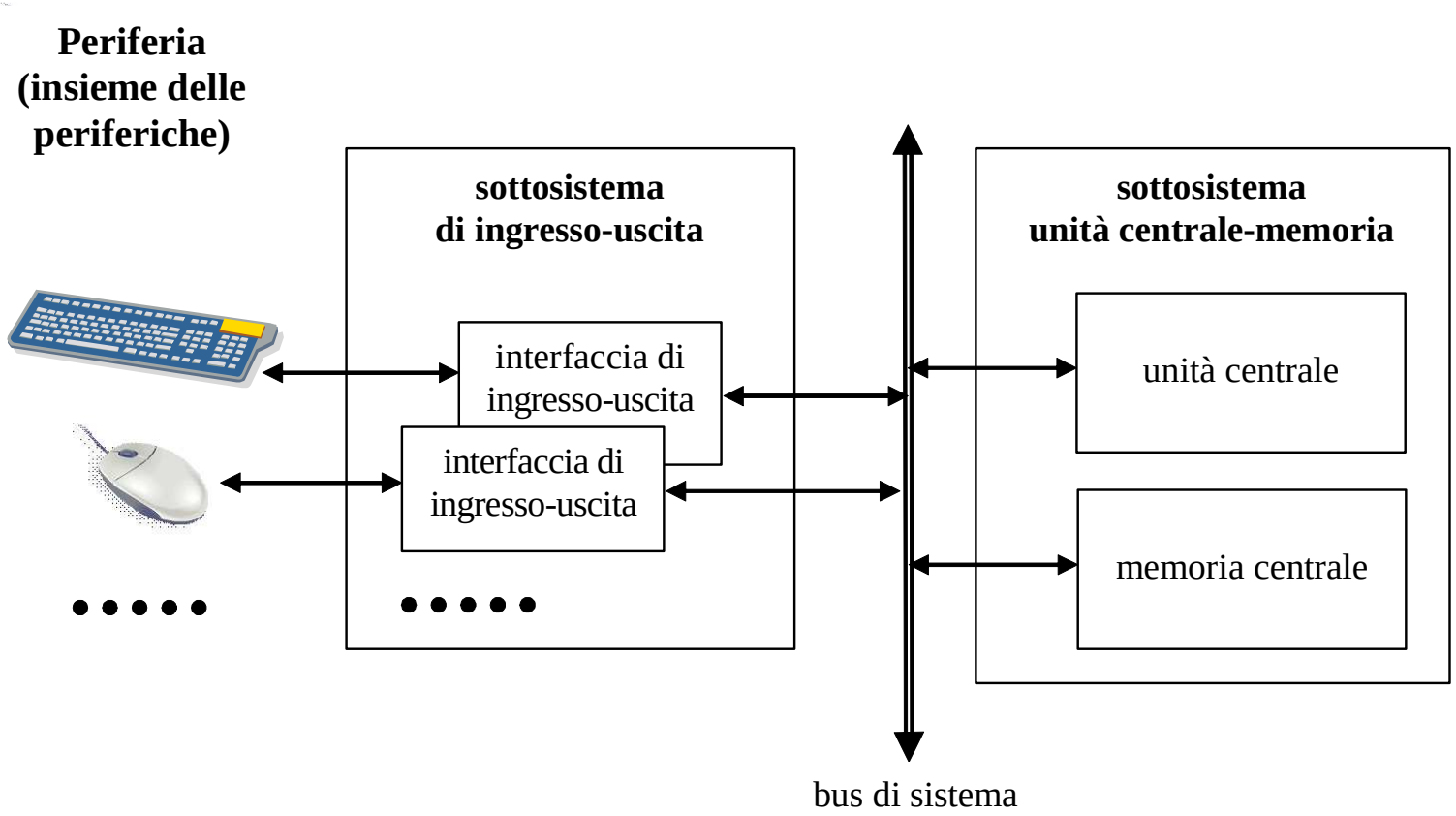



```
        bne     $r3, $r4, Allora
        sub     $r0, $r1, $r2
        j       Esci
Allora: add     $r0, $r1, $r2
Esci:   …
```



Istruzioni di ingresso e uscita (I/O)

- Permettono trasferimenti tra registri e interfacce I/O



ALTRE DOMANDE LECITE