CPU, RAM, ROM e BUS

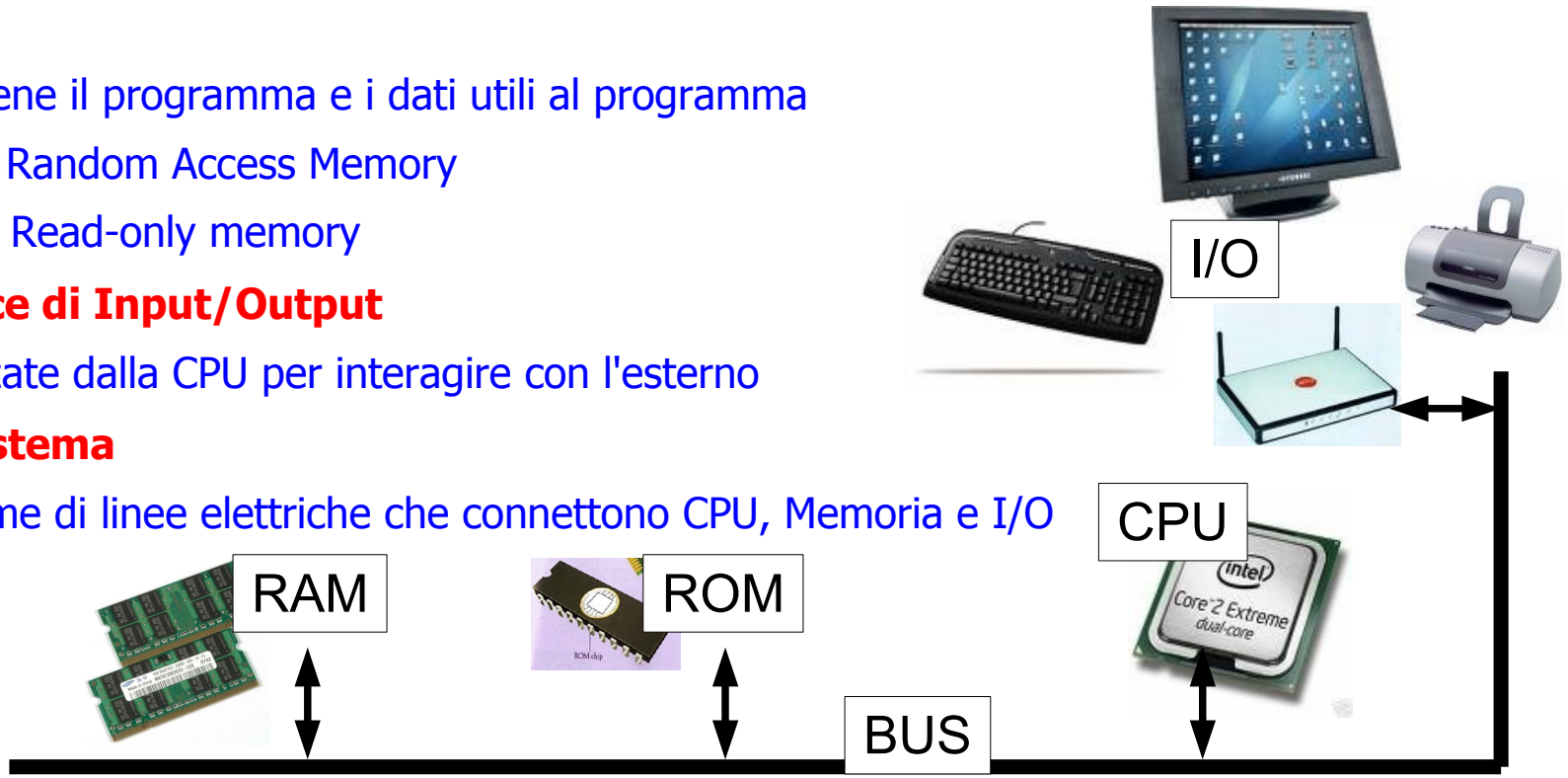
Corso di Abilità Informatiche
Laurea in Fisica

prof. ing. Corrado Santoro

A.A. 2009-10

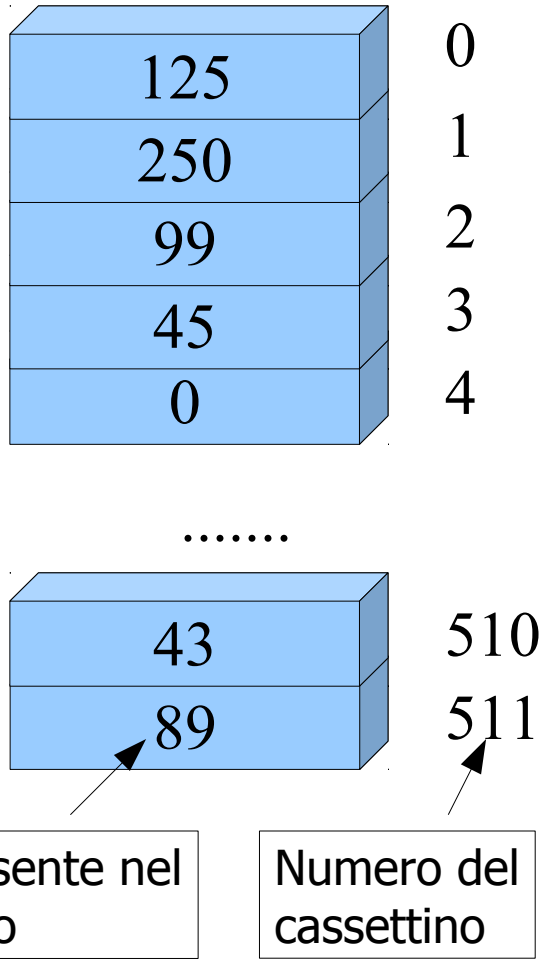
Ripassiamo: Struttura di un Computer

- **CPU**
  - Regola il funzionamento del computer
  - E' in grado di eseguire **istruzioni semplici**
  - Realizza la funzionalità desiderata attraverso l'esecuzione di un **programma**, composto da un insieme di istruzioni semplici
- **Memoria**
  - Contiene il programma e i dati utili al programma
  - RAM: Random Access Memory
  - ROM: Read-only memory
- **Interfacce di Input/Output**
  - Utilizzate dalla CPU per interagire con l'esterno
- **Bus di sistema**
  - Insieme di linee elettriche che connettono CPU, Memoria e I/O



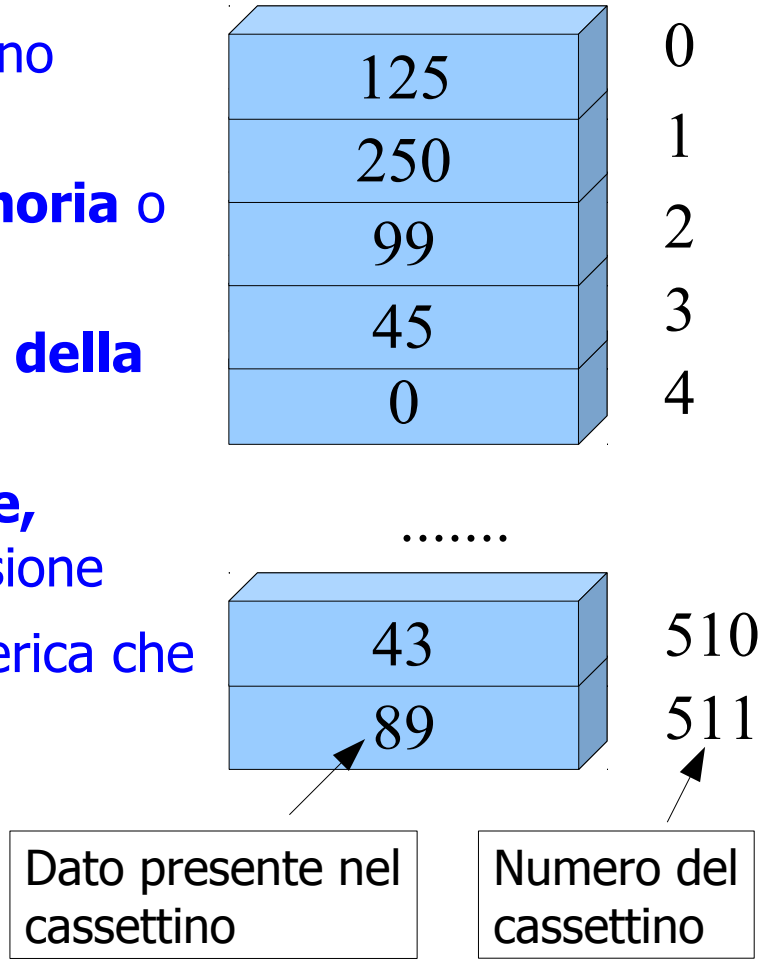
Ripassiamo: RAM e ROM

- Sono circuiti elettronici, ma possiamo pensarli come virtualmente composti da un insieme di "**cassettini**", ognuno dei quali può contenere un singolo "**dato**":
  - Ogni cassettino è numerato progressivamente, da 0 fino al numero che rappresenta la capacità massima della memoria considerata
  - Ogni dato è in realtà un'informazione numerica intera che può assumere un valore da **0 a 255**
  - Possiamo considerare quest'informazione come equivalente ad un **carattere alfanumerico**



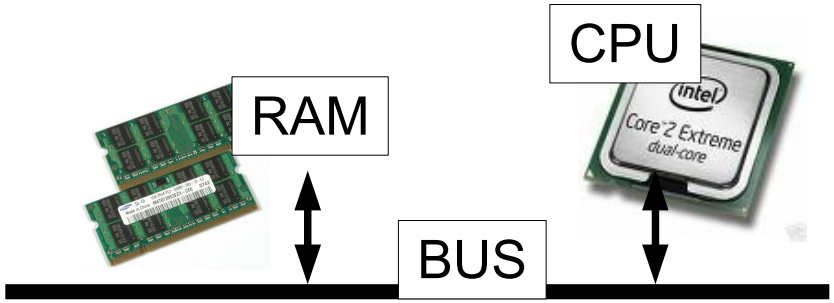
Ripassiamo: RAM, ROM e misura dello spazio

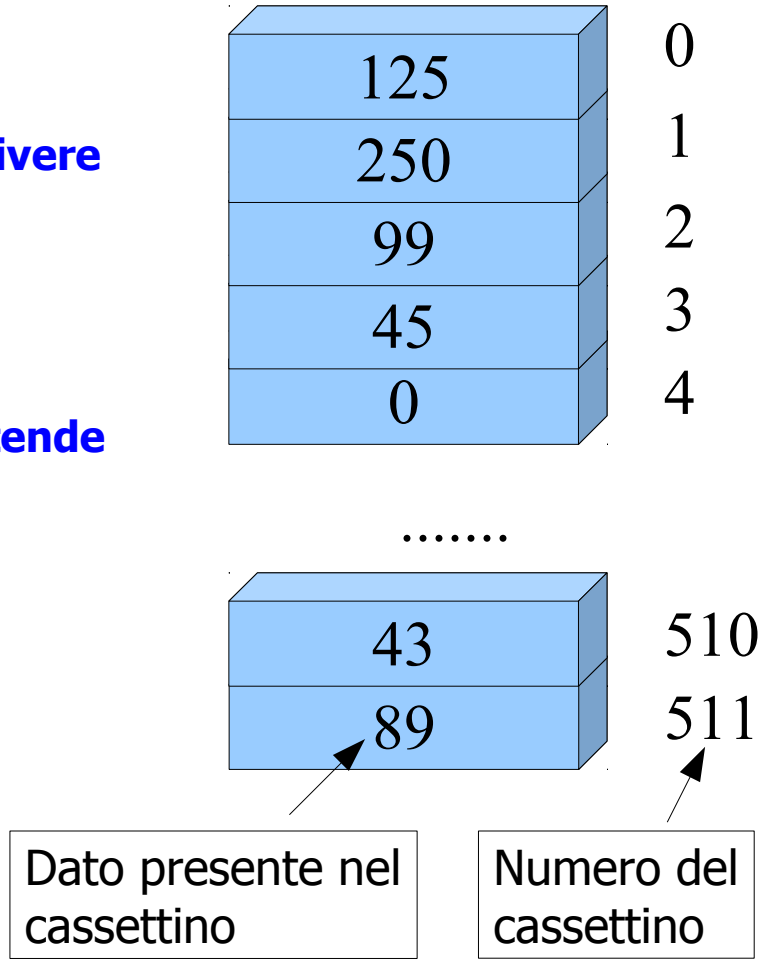
- E' possibile leggere o scrivere un cassettino per volta
- La scrittura di un nuovo dato in un cassettino provoca la sostituzione del vecchio valore
- Ogni cassettino è detto **locazione di memoria** o **cella di memoria**
- Il "numero di cassettino" è detto **indirizzo della locazione/cella di memoria**
- Il dato presente nel cassettino è detto **byte,** termine usato anche per indicare la dimensione
- Un **byte** equivale ad un'informazione numerica che può assumere un valore da **0 a 255**



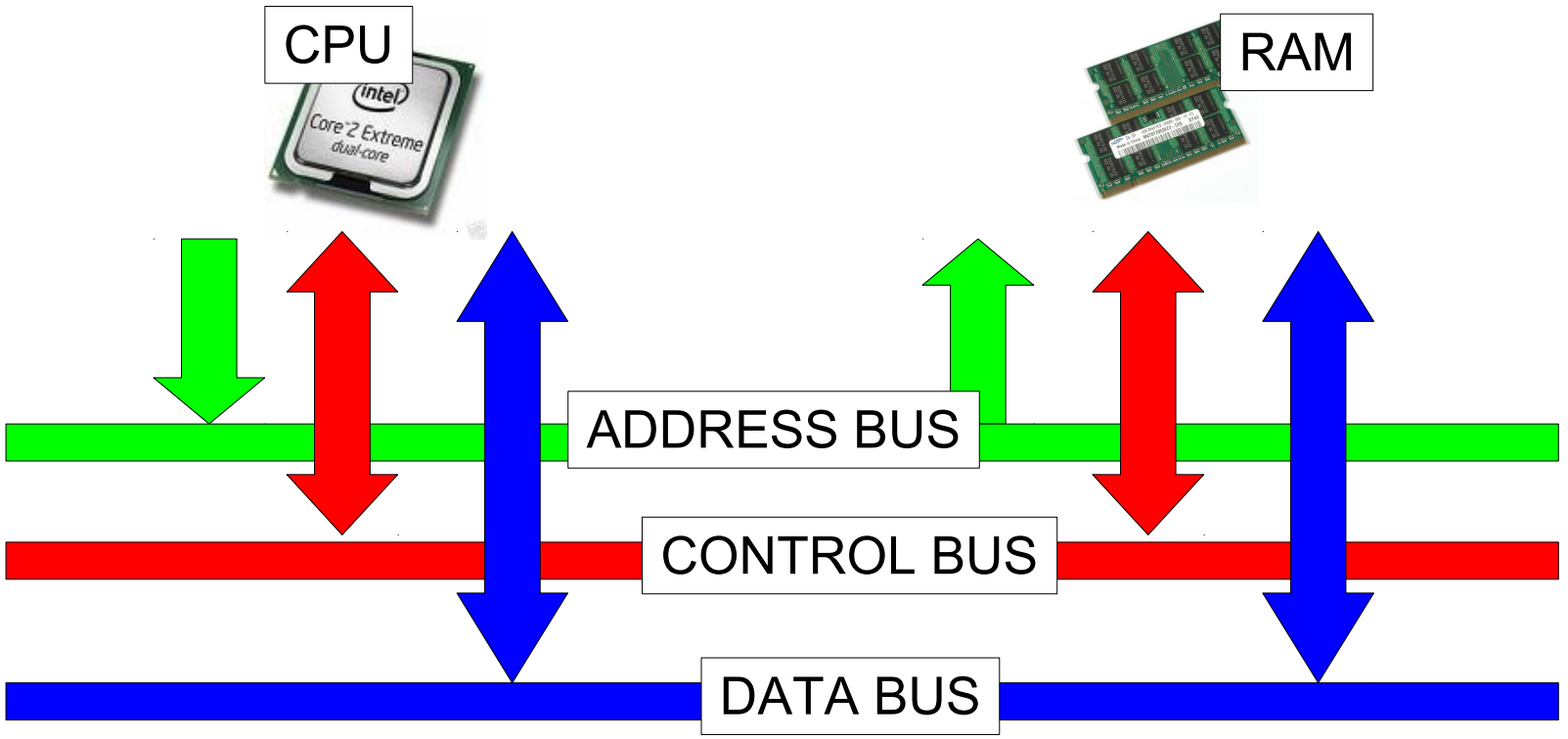
CPU, Memoria e BUS

- La connessione tra CPU e RAM (ROM) avviene tramite le linee elettriche del BUS
- Le linee del BUS sono tutte **digitali** (valore "0" o "1")
- La CPU, per scrivere sulla memoria, deve:
  - **Comunicare l'indirizzo di memoria a cui scrivere**
  - **Comunicare il dato che vuole scrivere**
  - **Inviare il comando di scrittura**
- La CPU, per leggere dalla memoria, deve:
  - **Comunicare alla memoria l'indirizzo che intende leggere**
  - **Inviare il comando di lettura**
  - **"Prelevare" il dato dalla memoria**



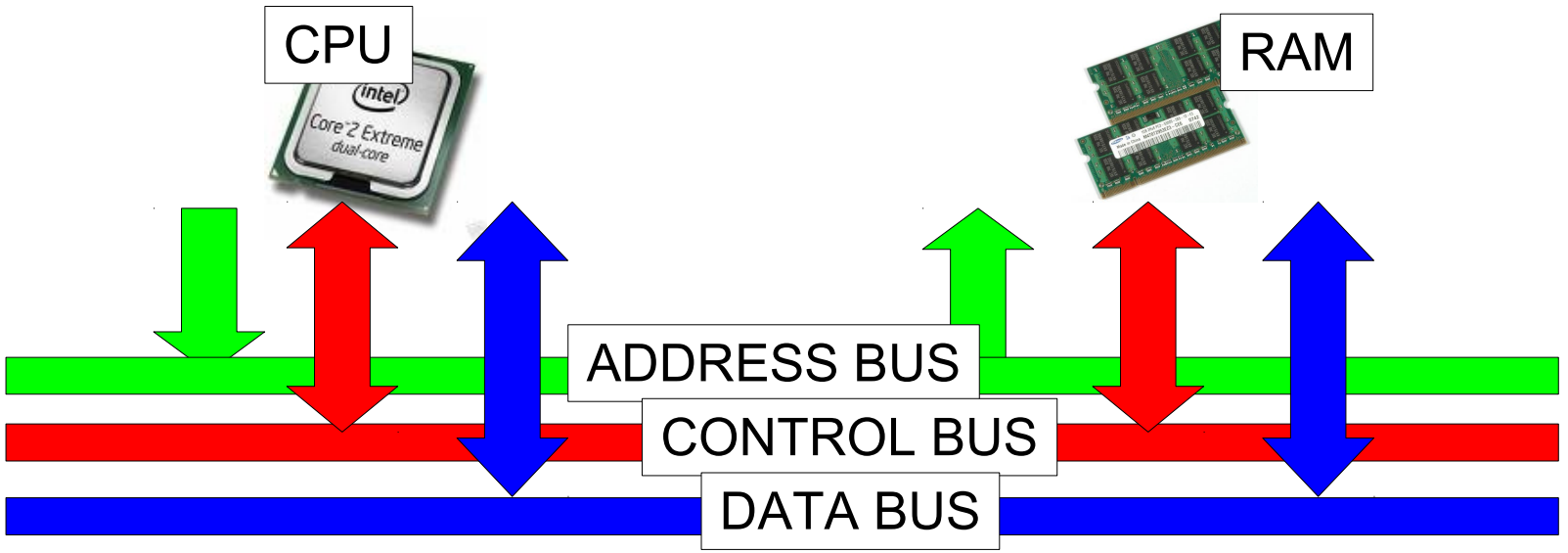


CPU, Memoria e BUS



- Linee dove la CPU pone l'**indirizzo**
- Linee dove la CPU invia i **comandi di lettura/scrittura**
- Linee per il transito dei **dati** da CPU a Memoria e viceversa

Quante linee ha il BUS?



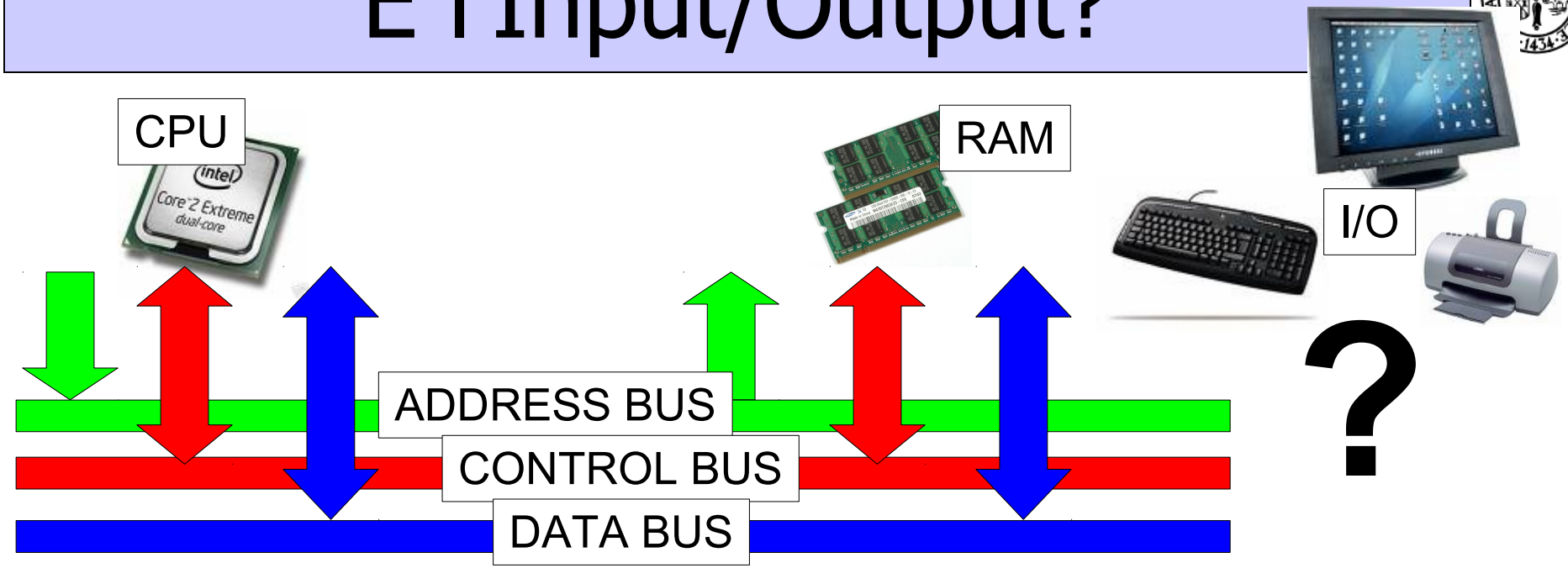
- **Linee dati: 8, 8 bit, [0, 2^8-1] = [0, 255]**
- **Linee di controllo: almeno 2, MEMR, MEMW**
  - **(in realtà le linee di controllo sono molte)**
- **Linee indirizzi:** non c'è un vincolo, ma il loro numero influenza la **dimensione massima** della memoria

Address BUS e dimensioni memoria

- 8 linee indirizzi: 2^8 = 256 byte
- 16 linee indirizzi: 2^{16} = 65536 byte = 64KB
- 32 linee indirizzi: 2^{32} = 4294967296 byte = 4 GB

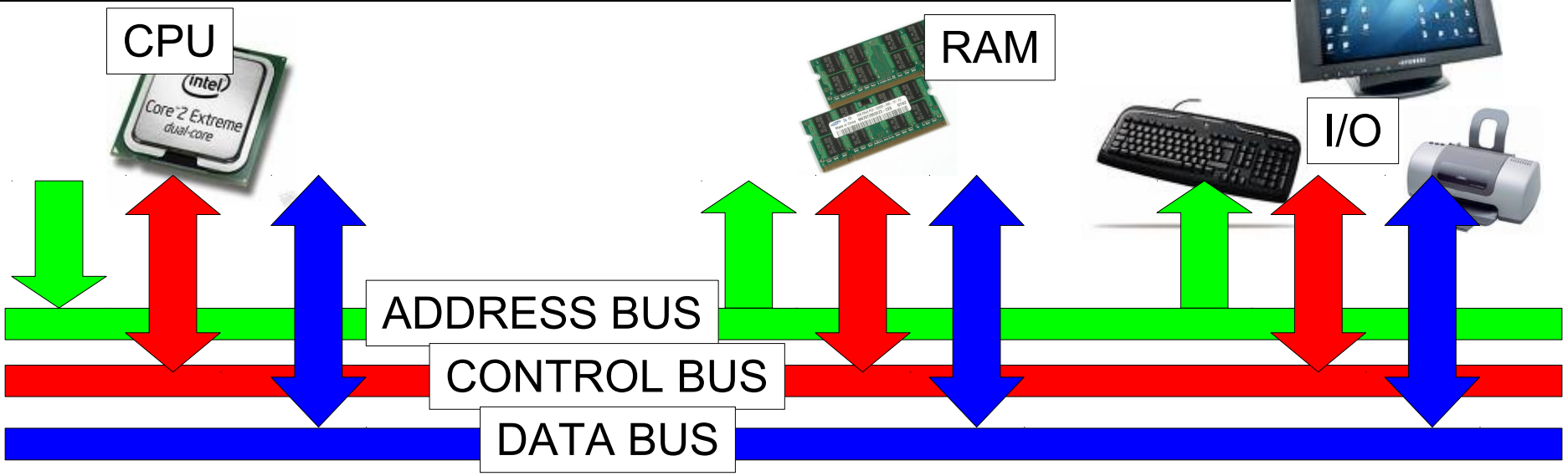
- Le CPU dei moderni PC hanno **32 linee di indirizzi**

E l'Input/Output?



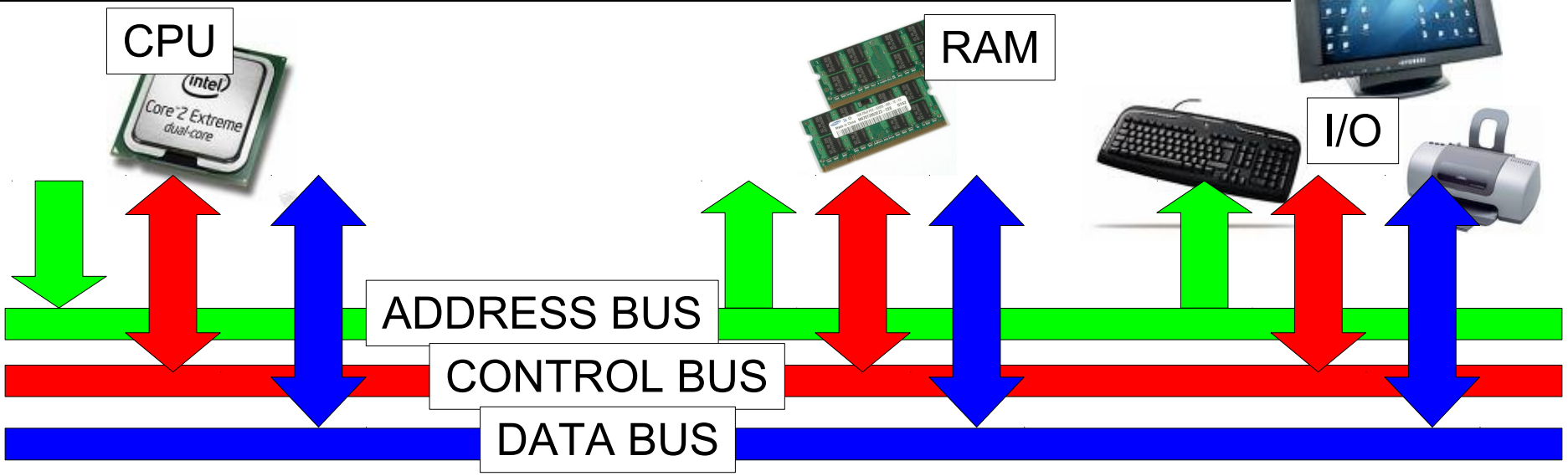
- **Come può la CPU "comunicare" con le periferiche di Input/Output?**
- **Riflessione:** se abbiamo trovato un meccanismo "elettrico" per comunicare con circuiti quali la memoria ...
- ... perché non possiamo usare esattamente lo stesso meccanismo per comunicare con circuiti come le periferiche?

Input/Output



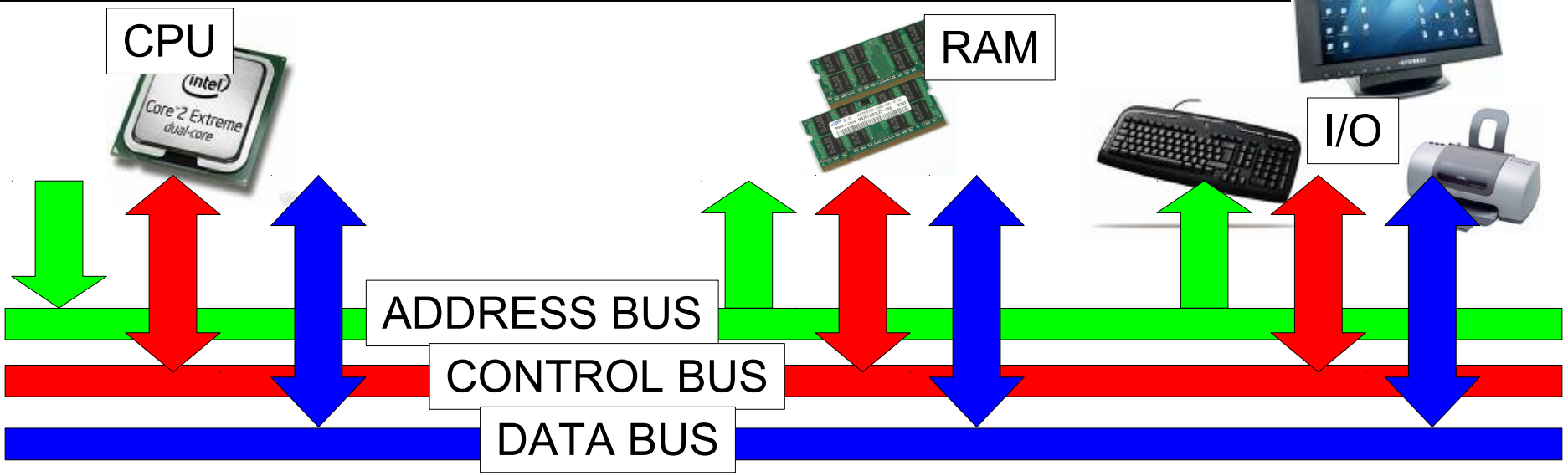
- **Considerazione:** Se "**non attiviamo**" le linee di controllo **MEMR**, **MEMW**, la memoria è "silente" ...
- ... allora possiamo fare ciò che vogliamo su data bus e address bus
- **Idea:** utilizziamo un meccanismo di indirizzamento analogo a quello della memoria, e altre due linee di controllo, che chiamiamo **IOR** e **IOW**, per poter interagire con le periferiche di I/O

Input/Output



- **Idea:** utilizziamo un meccanismo di indirizzamento analogo a quello della memoria, e altre due linee di controllo, che chiamiamo **IOR** e **IOW**, per poter interagire con le periferiche di I/O
- Costruiamo elettricamente le periferiche in modo che ognuna di esse sia in grado di rispondere ad **un indirizzo specifico** o ad un **insieme di indirizzi specifici**

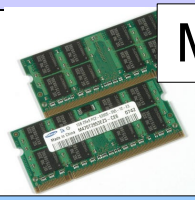
Esempio: la tastiera

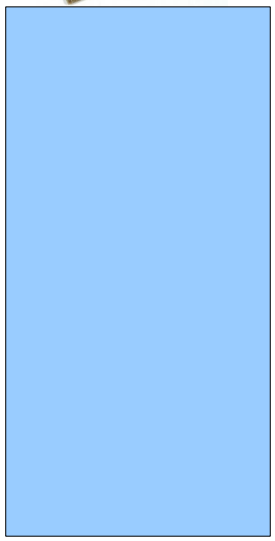


- Costruiamo elettricamente la tastiera in modo che "risponda" all'indirizzo specifico 60 Hex, allora:
  - Se **Address_Bus = 60Hex** e **MEMR attivato**
    - **LETTURA BYTE DALL'INDIRIZZO 60H DELLA RAM**
  - Se **Address_Bus = 60Hex** e **IOR attivato**
    - **LETTURA CARATTERE DALLA TASTIERA**

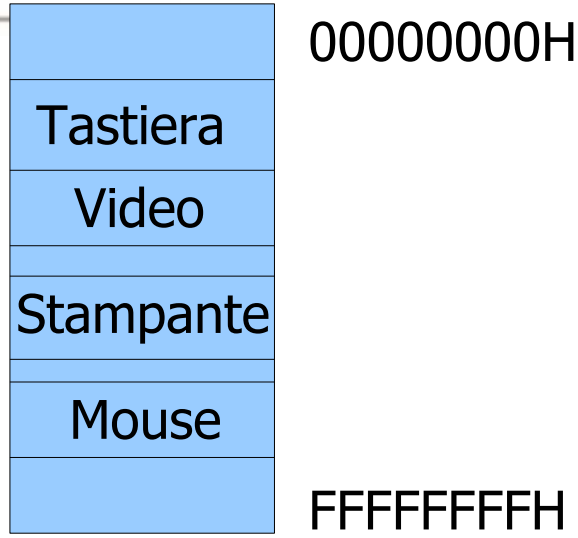
Memoria e I/O

Indirizzi a 32 bit





MEMR, MEMW



IOR, IOW

- Rispetto alla memoria, la zona degli indirizzi di I/O è **estremamente specializzata**, perché ogni indirizzo ha un significato diverso, funzione della periferica che risponde a quello specifico indirizzo

Riassumendo...

- CPU, Memoria e I/O sono interconnessi tramite il bus
- Il bus è suddiviso in
  - **BUS DATI** (bidirezionale, a 8 bit)
  - **BUS INDIRIZZI** (unidirezionale, da CPU verso l'esterno, normalmente 32 bit)
  - **BUS DI CONTROLLO** (bidirezionale, linee per attivare le funzionalità di lettura/scrittura da memoria e I/O)
- La CPU possiede due spazi di indirizzamento
  - **Memoria:** attivato tramite le linee di controllo MEMR e MEMW
  - **I/O:** attivato tramite le linee di controllo IOR e IOW