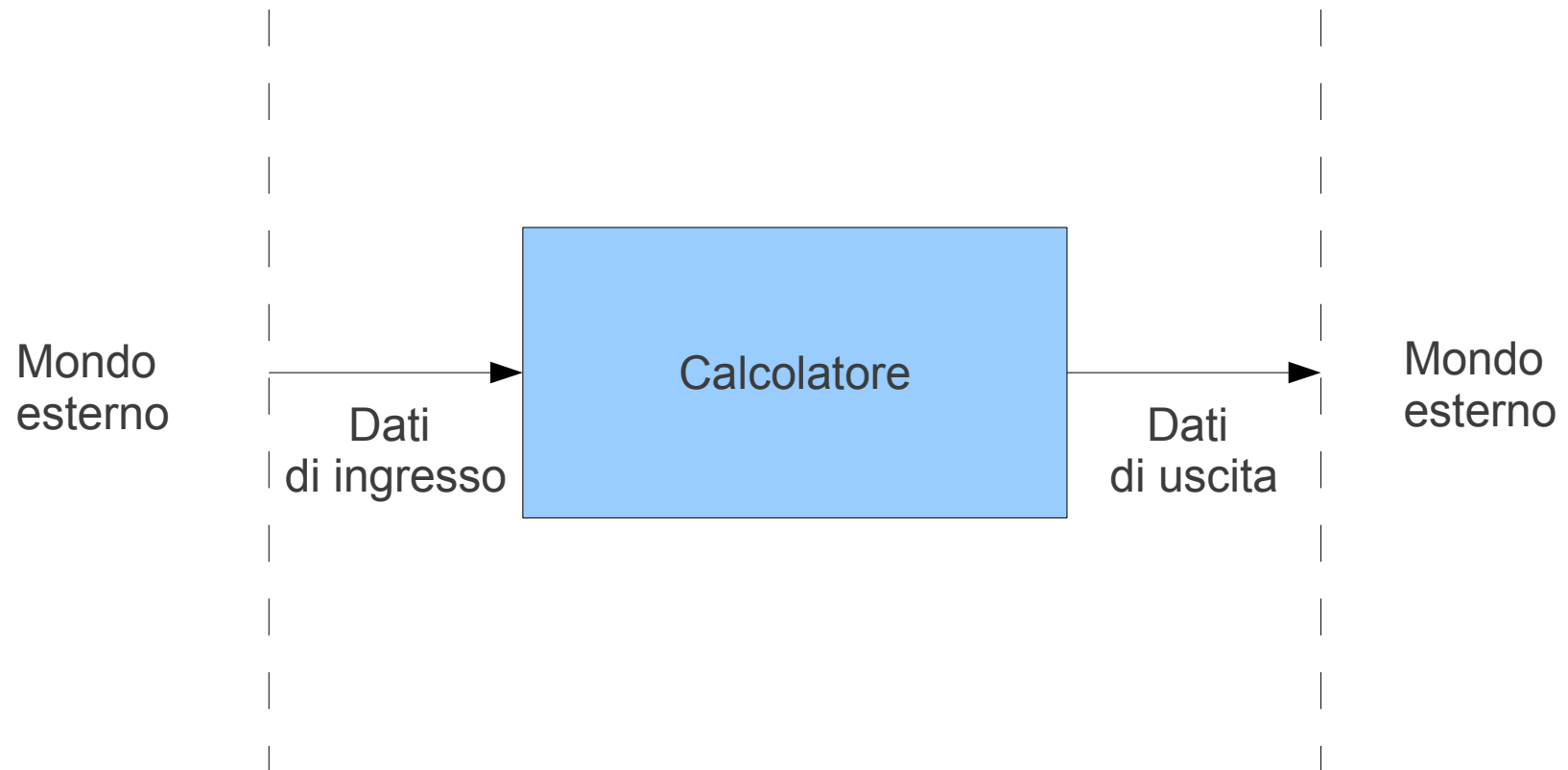


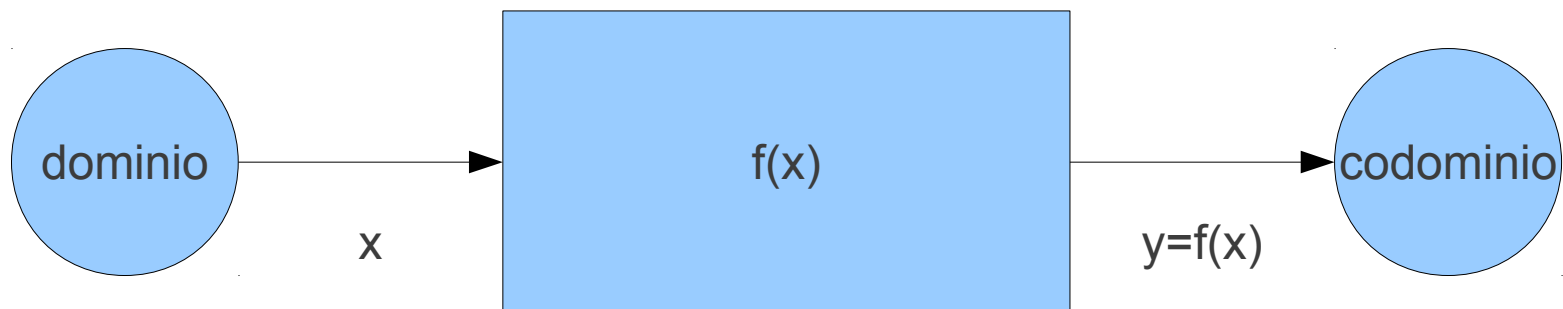
# Struttura del calcolatore



# Struttura del calcolatore

- Il calcolatore elabora dei dati di ingresso per ottenere dati in uscita
- In linea di principio, ma non solo, il **compito** svolto da un calcolatore è assimilabile a una funzione matematica
- I dati di ingresso appartengono al dominio della funzione
- I dati di uscita appartengono al codominio della funzione

# Struttura del calcolatore



# Struttura del calcolatore

- In questo schema tutte le situazioni reali
  - es. equazione di 2° grado:
    - Dominio: terne di numeri reali
    - Codominio: coppie di numeri complessi
  - Programma con interfaccia grafica:
    - Dominio: sequenze di numeri rappresentanti il movimento e i 'clic' del mouse
    - Codominio: sequenze di numeri rappresentanti i comandi inviati al monitor che accendono e spengono i punti luminosi in modo opportuno.

# Struttura del calcolatore

- Il **compito** svolto da un calcolatore è in realtà quello che viene chiamato il **processo** risultante dall'esecuzione di un **programma** che a sua volta rappresenta in forma opportunamente codificata l'**algoritmo** di soluzione di un **problema**.

# Struttura del calcolatore

- In origine abbiamo un **problema** da risolvere.
- Una prima formalizzazione consiste nell'individuare i dati in ingresso e in uscita
- Successivamente si deve trovare un 'metodo' di soluzione (es. formula di soluzione dell'eq. di 2° grado)
- Questo 'metodo', se soddisfa alcuni requisiti, costituisce **l'algoritmo** risolutivo

# Struttura del calcolatore

- Possiamo definire un algoritmo come:

*Un'insieme di operazioni che eseguite nell'ordine opportuno, producono in un **tempo finito** i risultati voluti*

- Rimangono indefinite alcune cose: quali sono le operazioni e come si stabilisce l'ordine di esecuzione?
- Soprattutto chi esegue le operazioni?
- E' fondamentale riferirsi ad un **esecutore** di riferimento.

# Struttura del calcolatore

- Sono stati proposti molti esecutori di riferimento che costituiscono i **modelli di calcolo** (o di computazione):
  - Macchina di Turing
  - Funzioni ricorsive
  - Sistemi di riscrittura
  - ...
- Il concetto di calcolo (o computazione) viene definito nell'ambito di questi formalismi
- Le operazioni elementari dipendono dal formalismo adottato



# Struttura del calcolatore

- Ogni modello di calcolo definisce l'insieme dei possibili algoritmi che è in grado di eseguire (e di problemi che si possono risolvere)
- Tutti i modelli di calcolo proposti si dimostrano equivalenti fra di loro (tesi di Church-Turing)
  - Per ogni algoritmo eseguibile in un modello ne esiste uno equivalente in un altro modello
- Esistono problemi che non si possono risolvere o meglio per i quali non è possibile scrivere un algoritmo in nessun modello di calcolo

# Struttura del calcolatore

- Una giustificazione dell'affermazione precedente (che peraltro si dimostra in modo matematicamente rigoroso):
  - Gli algoritmi esprimibili in un dato modello di calcolo sono una quantità **numerabile** (tanti quanto i numeri interi)
  - I problemi che si possono definire (ci possiamo anche limitare a funzioni da intero a intero) sono una quantità **non numerabile** (tanti quanto i numeri reali)
- Per cui esistono problemi **non risolvibili** con algoritmi

# Struttura del calcolatore

- **Macchina di Turing**: è il prototipo del calcolatore che utilizziamo quotidianamente. E' alla base del **paradigma imperativo di programmazione**.
- **Funzioni ricorsive**: definiscono gli algoritmi direttamente come funzioni matematiche a partire da alcune funzioni elementari. Sono alla base del **paradigma funzionale di programmazione**.
- **Sistemi di riscrittura**: definiscono gli algoritmi come predicati logici da dimostrare. Sono alla base del **paradigma logico di programmazione**.

# Struttura del calcolatore

- Una volta determinato l'algoritmo è necessario **codificarlo** (scriverlo, rappresentarlo) mediante un **linguaggio di programmazione** opportuno.
- Il risultato della codifica è il **programma**.
- Il programma deve essere inserito (**caricato**, loaded) nel calcolatore
- Il calcolatore **esegue** il programma dando origine ad un **processo**
- Ogni esecuzione del programma costituisce un processo diverso

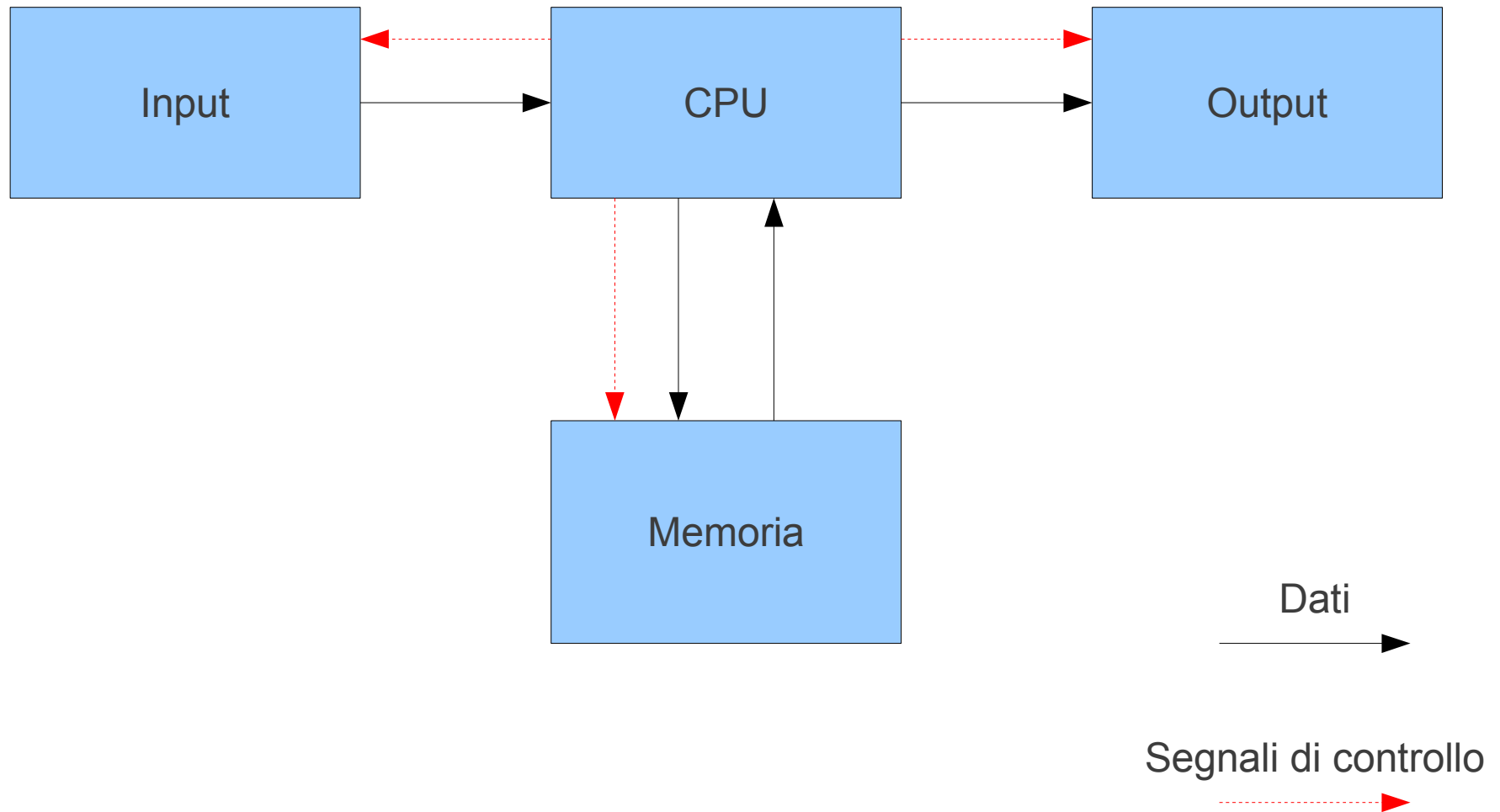
# Struttura del calcolatore

- I linguaggi di programmazione ricalcano i paradigmi di programmazione:
  - Linguaggi **imperativi**: C, C++, Pascal, Fortran, Basic, **Java** ecc.
  - Linguaggi **funzionali**: Lisp, ocaml ecc.
  - Linguaggi **logici**: Prolog ecc.

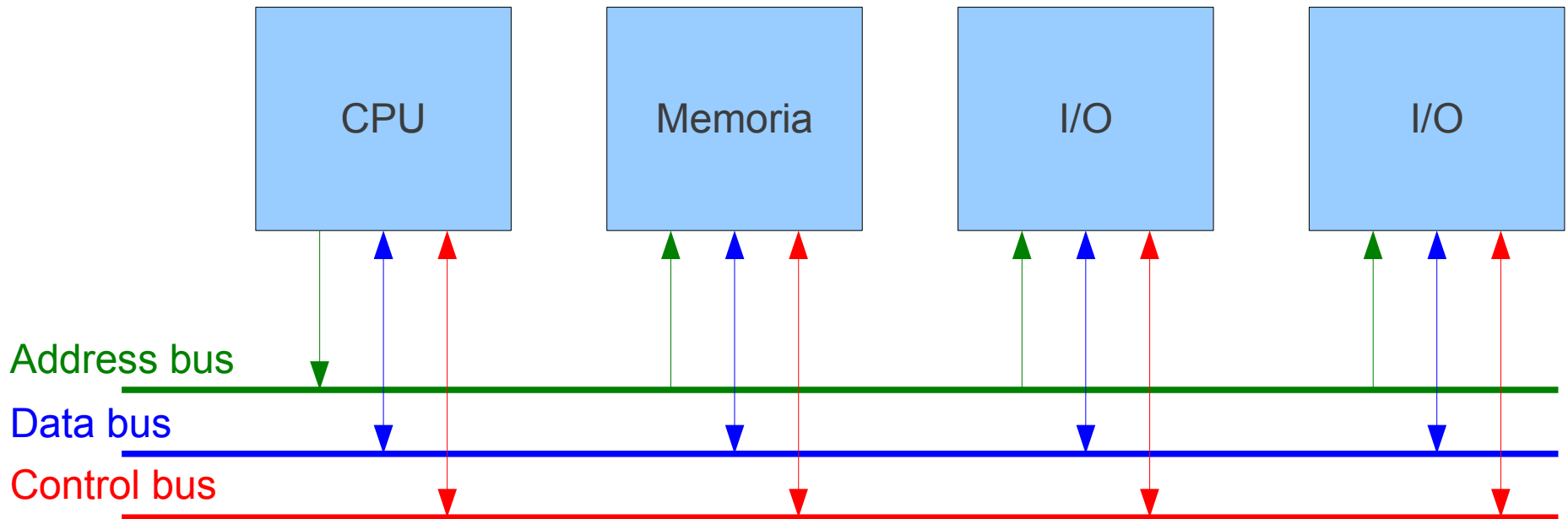
# Struttura del calcolatore

- Per definire correttamente un algoritmo dobbiamo quindi definire in modo preciso l'esecutore.
- La macchina di Turing è il modello di esecutore per la programmazione imperativa, ma è troppo elementare
- Per definire un esecutore, assimilabile ad un calcolatore come lo intendiamo noi, ci riferiamo alla [Macchina di Von Neumann](#)
- In ogni caso tale modello è equivalente la macchina di Turing

# Struttura del calcolatore



# Struttura a bus



Modulare e espandibile



# Struttura del calcolatore

- Nella programmazione imperativa un programma assume l'aspetto di una ricetta
- Una sequenza di operazioni da compiere nell'ordine stabilito dalla ricetta
- Le operazioni elementari dipendono dall'esecutore
  - per un grande chef potrebbe essere la preparazione di primo piatto completo
  - per uno che si arrangia, è necessario dettagliare le operazioni

# Struttura del calcolatore

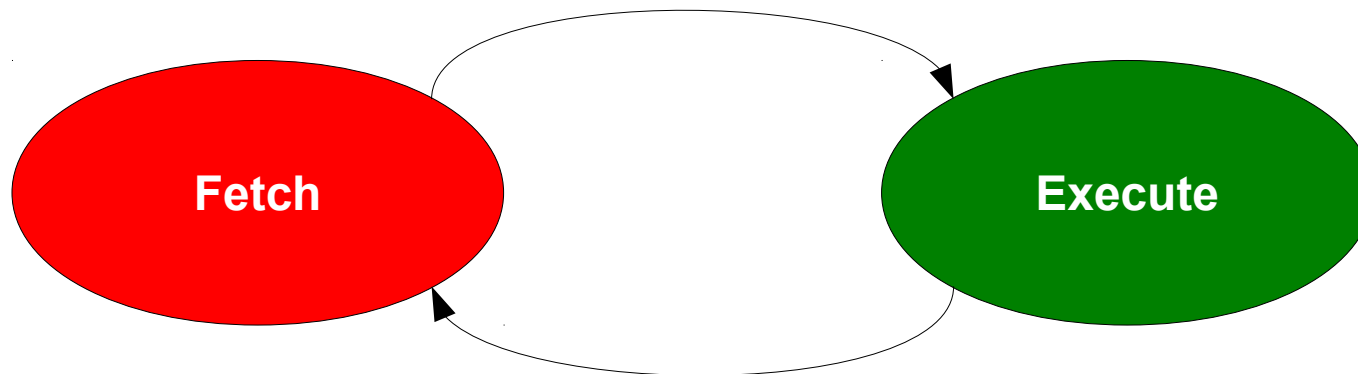
- La **CPU** (Central Processing Unit) è l'esecutore ed è in grado di eseguire un numero finito di operazioni elementari dette istruzioni e che costituiscono il **linguaggio macchina**.
- In base al programma, *che deve essere espresso in linguaggio macchina*, la CPU esegue le istruzioni una dopo l'altra leggendo i dati dall'unità di **input**, utilizzando la **memoria** per memorizzare i risultati intermedi del calcolo e scrivendo i risultati nell'unità di **output**

# Struttura del calcolatore

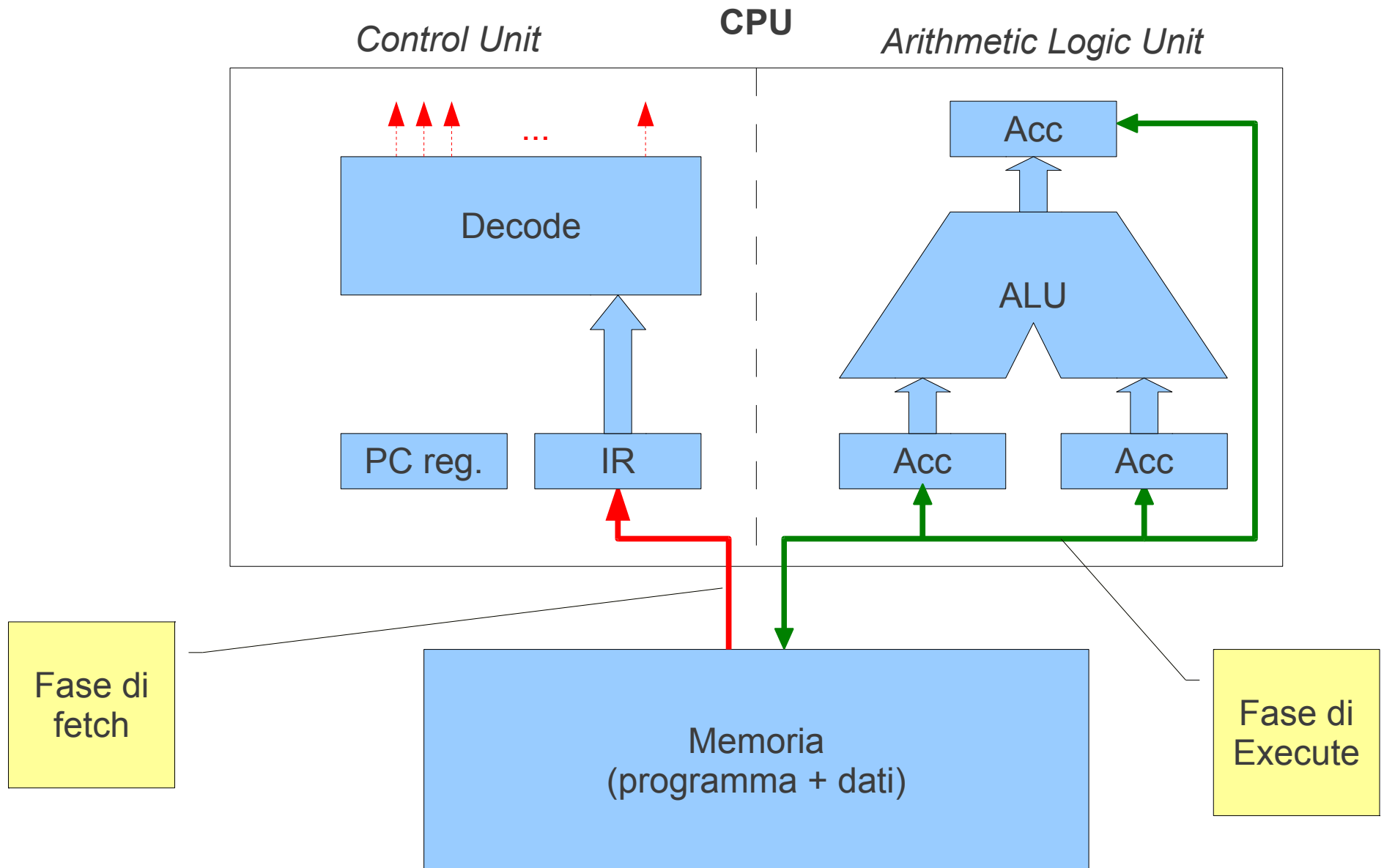
- Ma il programma dove risiede?
  - In linea di principio potrebbe essere cablato direttamente nella CPU, ma ciò ne farebbe un oggetto in grado di eseguire sempre lo stesso programma (in alcune applicazioni con i microcontrollori ciò è un requisito)
  - In un calcolatore general purpose, il programma risiede in memoria insieme ai dati (ma in zone diverse)

# Struttura del calcolatore

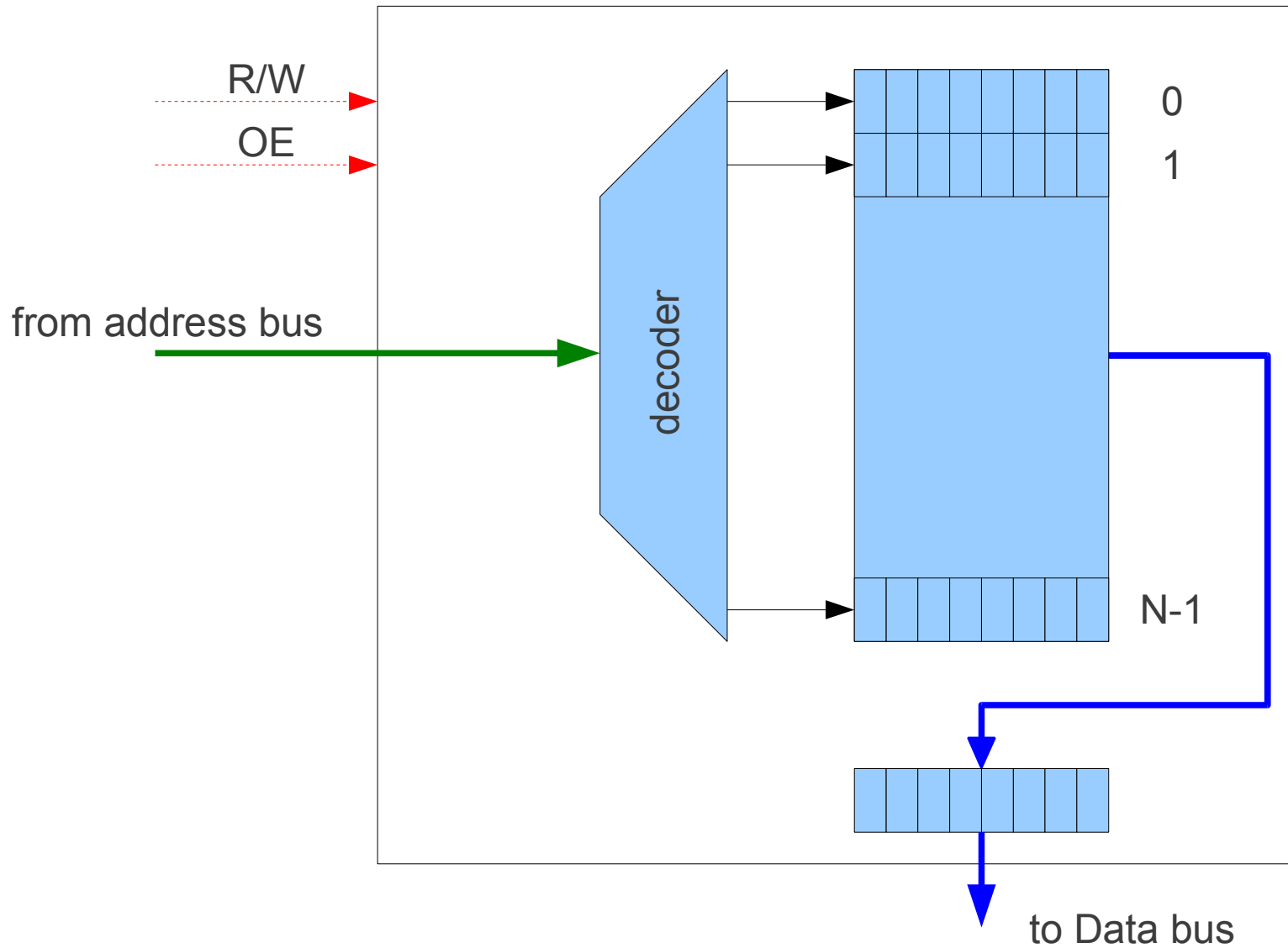
- Ne segue che la CPU deve accedere alla memoria per prelevare le istruzioni (fase di **fetch**) ed eseguirle (fase di **decode** and **execute**)



# CPU



# Memoria



# Memoria

- La memoria è organizzata in un insieme di N **locazioni** di memoria
- Ogni locazione di memoria può memorizzare un certo numero (in genere 8) di **informazioni binarie** (**bit**)
- Ogni locazione di memoria è individuata da un numero detto **indirizzo**
- N è detto **capacità** di memoria

# Memoria

- La CPU **legge** un dato dalla memoria fornendo l'indirizzo di una locazione sull'A.B., un comando di lettura sul C.B. e acquisendo l'informazione sul D.B.
- La CPU **scrive** un dato in memoria fornendo l'indirizzo di una locazione sull'A.B., un dato sul D.B. e un comando di scrittura sul C.B.
- La locazione di memoria è la minima unità indirizzabile



# Struttura del calcolatore

- Dal punto di vista tecnologico è facile realizzare oggetti (circuiti) a due stati (bistabili).
- La rappresentazione in memoria dei dati e dei programmi deve quindi essere realizzata utilizzando un sistema a 2 valori per
  - Le informazioni (dati) numeriche e alfanumeriche
  - Le istruzioni del linguaggio macchina