Introduzione storica al CALCOLO AUTOMATICO

Durante i secoli del basso medioevo si assistette in Europa a un fiorire su larga scala degli scambi commerciali, causa e, nello stesso tempo, effetto dello sviluppo demografico degli antichi borghi medioevali.

Il commerciò fu trascinato dalla navigazione e dalla ricerca scientifica., ma dall'altro ne fu anche promotore.

La navigazione si fondava molto sulle osservazioni astronomiche e con ciò esse riacquisirono l'antica importanza.

Ma i calcoli sui movimenti delle stelle coinvolgevano numeri molto grandi e le operazioni aritmetiche risultavano molto impegnative. Ciò portò da un lato all'introduzione di tecniche alternative (es: logaritmi) e dall'altro alla costruzione di macchine capaci di svolgere calcoli:

- Blaise Pascal (1623 – 1662), filosofo e matematico francese, costruì una macchina in grado di svolgere operazioni di somma e sottrazione;
- Gottfried Leibniz (1646 – 1716), filosofo e matematico tedesco, costruì una macchina in grado di svolgere operazioni di moltiplicazione e divisione.

Fin dall'inizio del XVIII secolo si cominciarono a costruire vari automatismi negli impianti manifatturieri, e fu in uno di questi impianti che nacque la prima macchina flessibile, capace di produrre un oggetto con caratteristiche definibili di volta in volta:

- Jacquard J. M. (1752 – 1834), artigiano francese, costruì un telaio che poteva produrre tessuti con disegno definito da una griglia di fori praticati su una scheda estraibile: i fori determinavano la posizione di aste uncinate che sollevavano oppure no i fili della trama (il disegno si creava col passaggio della trama sopra o sotto l'ordito). Era nato il telaio programmabile.

Nei primi decenni dell'800, la maturazione della rivoluzione industriale accentuò il problema del calcolo.

Charles Babbage (1791 – 1871), matematico inglese, intorno agli anni '20 progettò quella che chiamò la *macchina analitica*: con tecnologia meccanica e ispirandosi alle schede perforate di Jacquard, la macchina doveva

- possedere una memoria per i risultati parziali
- possedere una unità di calcolo
- essere capace di eseguire un programma scritto su schede perforate
- essere capace di una stampa automatica dei risultati

Era impostata come un moderno calcolatore, ma sembra che per carenza di finanziatori la macchina non fu mai costruita da Babbage.

La macchina di Babbage vide la luce verso la fine dell'800, quando, in occasione del censimento Usa del 1890, fu dato incarico a Herman Hollerit di costruire una macchina che avrebbe dovuto gestire i dati. La ditta che con Hollerith costruì il primo calcolare sarebbe poi divenuta IBM (International Business Machines).

Fu la seconda guerra mondiale a dare un notevole impulso alla costruzione di calcolatori.

USA:

- il problema del puntamento automatico dei cannoni della contraerea navale fu risolto con la costruzione di *calcolatori analogici*;
- la mole di calcoli per usi generici fu risolta con la costruzione (1944) di un *calcolatore ibrido*: il MARK 1, in parte analogico e in parte discreto, con tecnologia elettromeccanica.

GB: il problema della decodifica dei codici di trasmissione dei sottomarini tedeschi fu risolto con la costruzione di un *calcolatore interamente discreto*.

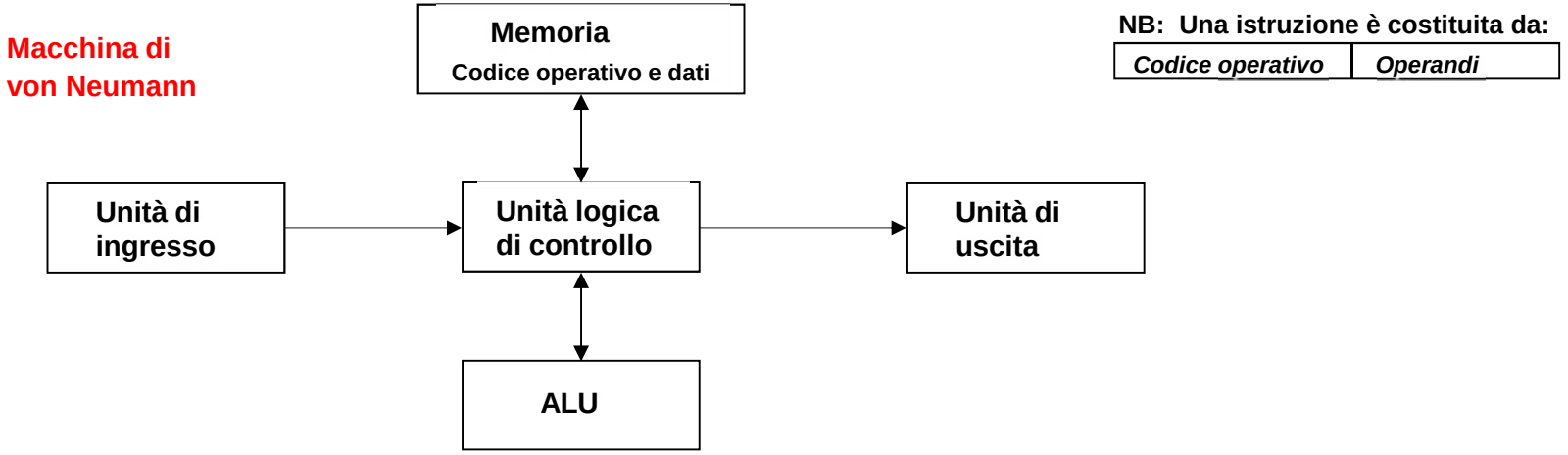
Nel 1946 fu costruito l'ENIAC, primo calcolatore elettronico, realizzato interamente a valvole. Tuttavia l'organizzazione era sostanzialmente la stessa del MARK 1.

John von Neumann (1903 – 1957), matematico di origine ungherese, naturalizzato americano, aveva visto funzionare il Mark 1 a Los Alamos nel 1944, dove veniva utilizzato per i calcoli necessari per la bomba atomica. Neumann non rimase soddisfatto del funzionamento del Mark 1:

- dati e istruzioni non venivano inseriti attraverso le stesse unità di ingresso e non venivano memorizzati nella stessa memoria
- l'esecuzione di un programma richiedeva la continua presenza di operatori umani.

Nel 1946 von Neumann pubblicò un articolo in cui proponeva una macchina alternativa, in cui:

- dati e istruzioni dovevano essere immessi attraverso le stesse unità di ingresso e dovevano essere memorizzati nella stessa memoria
- per l'esecuzione di un programma, si doveva caricarlo interamente in memoria.

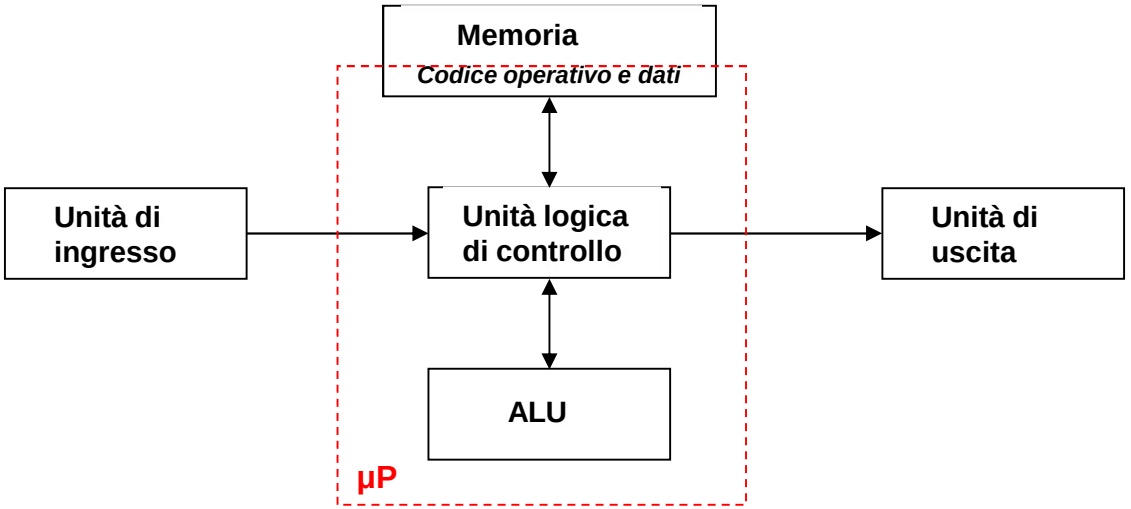


IL primo calcolatore con l'architettura di von Neumann fu costruito nel 1949: l'EDSAC, con tecnologia elettronica a valvole.

Nel 1947 era stato costruito il primo transistor, ma il primo calcolatore a transistor risale al 1959, realizzato dall'IBM.

Negli anni '60 l'integrazione dei componenti elettronici su un unico chip di silicio divenne molto grande, tanto che nel 1971 si arrivò a costruire il primo microprocessore (µP, CPU) nei laboratori della Intel per opera di Federico Faggin.

Il µP racchiudeva diversi blocchi dell'architettura di von Neumann:



NB: la piccola parte di memoria incorporata nel µP è denominata '*cache*'.

Nel 1975 fu costruito il primo calcolatore con µP. Si costruirono grandi calcolatori destinati ai centri di ricerca e a grandi utenze in generale (unico calcolatore con molti terminali: multiplazione a divisione di tempo).

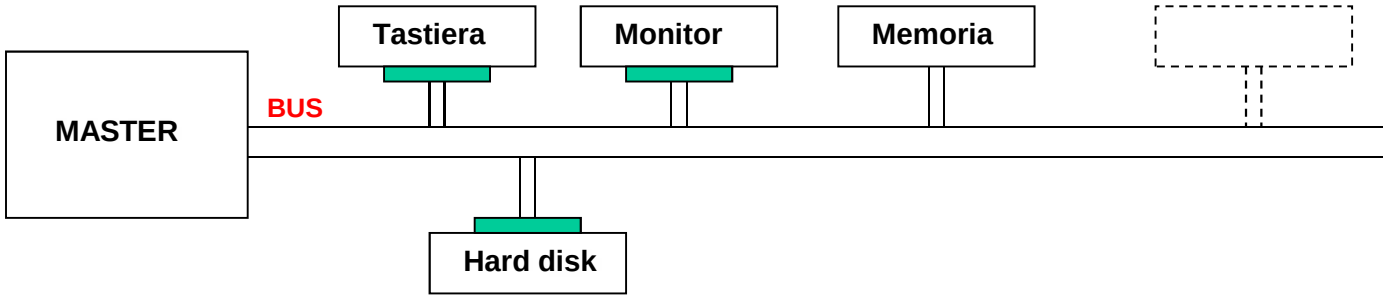
Nell'agosto del 1981 l'IBM presentò sul mercato un piccolo calcolatore destinato all'uso personale: il PC.

I PC , in quanto macchine economiche, nacquero con due difetti che li caratterizzano tuttora: **non sono**

- **deterministici**: non garantiscono un tempo massimo in cui la routine sarà eseguita;
- **real time**: velocità di risposta (esecuzione routine) non all'altezza di quella del processo esterno da controllare.

ARCHITETTURA A BUS

Il bus è un canale di comunicazione formato da più linee condivise da tutti i componenti del sistemi.



Caratteristiche:

- è possibile *un solo trasmettitore* (Tx) per volta: in caso contrario, nella più innocua delle ipotesi, il livello logico della linea potrebbe essere incerto
- sono possibili *più ricevitori* (Rx) contemporaneamente; esiste un limite massimo al numero dei Rx dipendente dal fan-out del trasmettitore
- il *grande vantaggio* del bus è la sua espandibilità, cioè la possibilità di aggiungere ulteriori componenti nel sistema mantenendo inalterato il circuito esistente.

Il bus può essere diviso in tre parti funzionali:

- address bus: linee che portano l'informazione dell'indirizzo del componente
- data bus: linee su cui viaggiano i dati che i componenti del sistema si scambiano
- control bus: linee su cui viaggiano i segnali di comando che gestiscono l'accesso al bus.

L'accesso al bus dei componenti, come Tx o come Rx, è gestito dal master (spesso un µP).

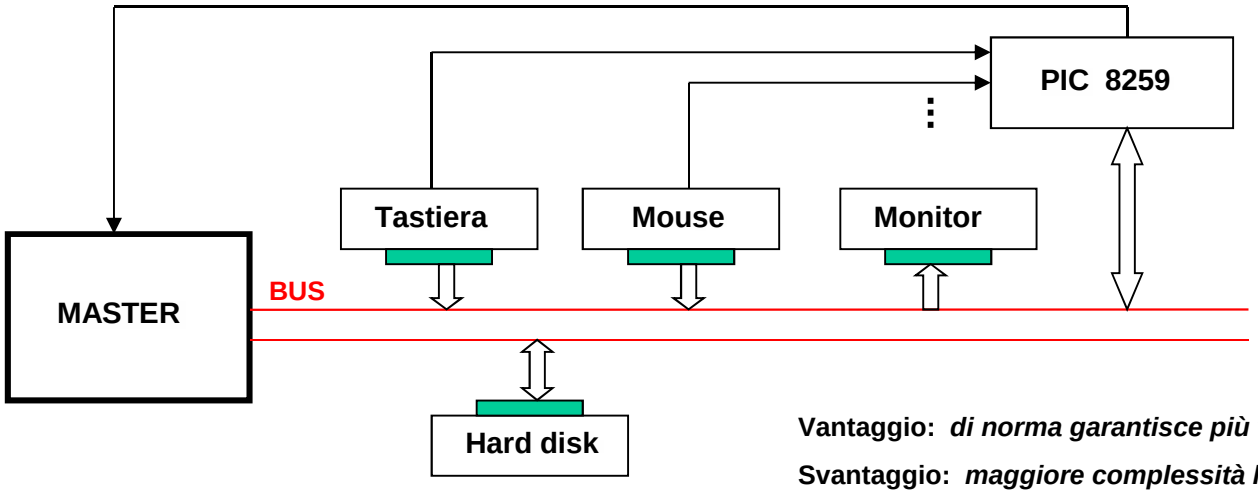
In generale si hanno due tecniche di accesso al bus:

- Polling: il master sonda periodicamente, in sequenza, lo stato di tutti i componenti, se qualcuno ha bisogno di impegnare il bus, viene accontentato.

  Vantaggio: *semplicità hardware*

  Svantaggio: *possibili tempi lunghi di attesa di accesso*.

- Interrupt: il componente che deve impegnare il bus invia una comunicazione al PIC (es: 8259), il quale esamina tutte le richieste e invia al master la richiesta con priorità più alta; il master, sulla base delle proprie esigenze interne può accogliere o meno la richiesta; nel caso la accolga chiede al PIC il codice del componente che ha chiesto l'impegno del bus e invia i relativi segnali di comando.



Vantaggio: ***di norma garantisce più veloci tempi di accesso***

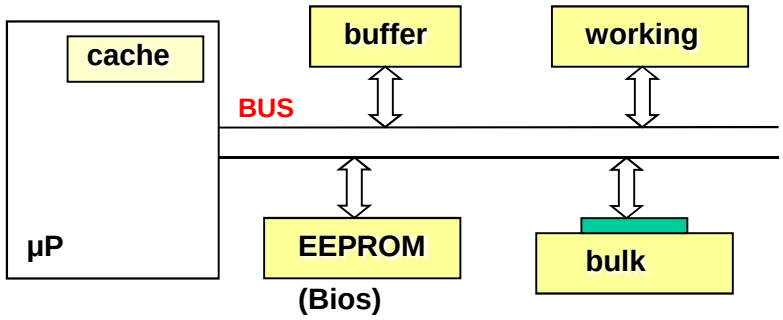
Svantaggio: ***maggiore complessità hardware.***

CLASSIFICAZIONE DELLA MEMORIA in un PC

Classificazione per tipologia secondo:

- il tipo di accesso:
  - sequenziale: nastri, alcuni registri
  - casuale (RAM): dischi (magnetici e ottici), chip seminconduttori
- l'uso:
  - solo lettura (ROM): chip sminconduttori, CD-ROM
  - lettura/scrittura: dischi (magnetici e ottici), chip seminconduttori
- la conservazione dei dati dopo aver tolto l'alimentazione:
  - non volatile: chip semiconduttori, dischi, nastri
  - volatile:
    - statica (SRAM): più veloce e più costosa,
    - dinamica (DRAM): bisogno di continuo rinfresco, maggiore integrazione

Classificazione per funzione:



cache: è chiamata di 1° livello.

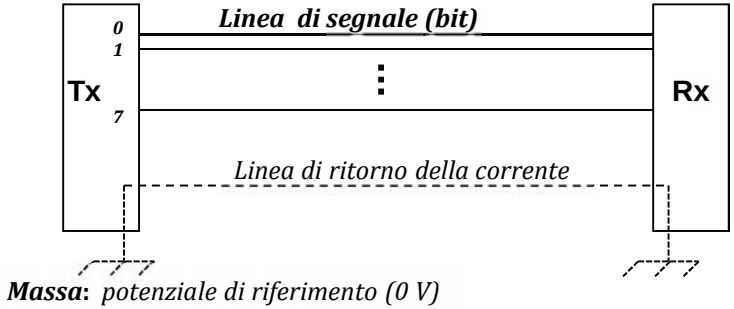
buffer memory: memoria tampone (SRAM), disposta vicina al µP, lavora a una frequenza più alta di quella di lavoro (cache di 2° livello).

working memory: memoria di lavoro (DRAM), memoria in cui è caricato il programma per l'esecuzione.

bulk memory: memoria di massa (dischi), per la funzione di archivio.

PORTE seriali e parallela in un PC

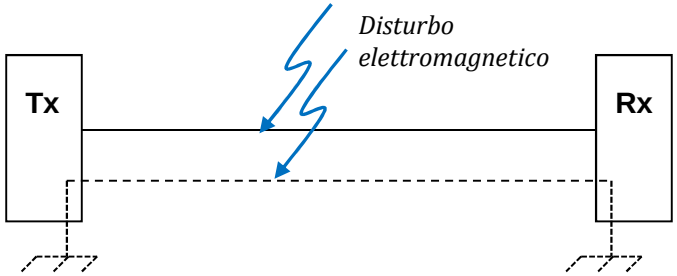
Trasmissione parallela: *tutti i bit del dato sono trasmessi contemporaneamente, ciascuno sulla propria linea.*



NB: l'informazione del bit è contenuta nel livello di potenziale (tensione) presente sulle singole linee di segnale.

Trasmissione seriale: *tutti i bit del dato sono trasmessi, uno dopo l'altro, sull'unica linea.*

Es: RS 232, la cui massima velocità di trasmissione nei PC è in genere di 115 200 baud (bit al secondo: *bps* oppure *b/s*)

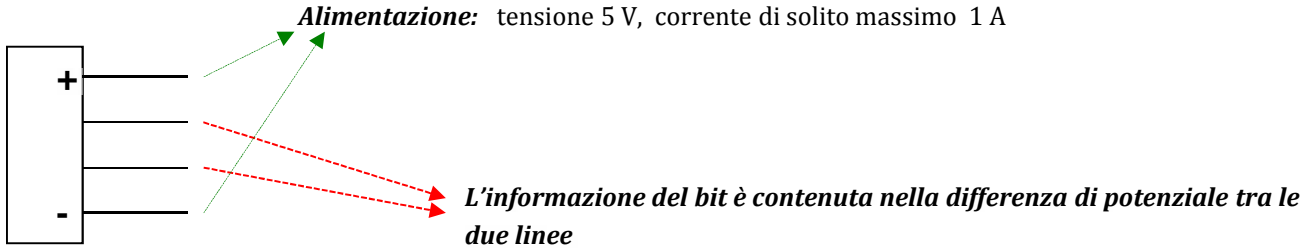


NB: a fronte di un disturbo, solo la linea di segnale può variare il suo potenziale (la linea di ritorno è vincolata a 0 V) e ciò potrebbe alterare il valore del bit.

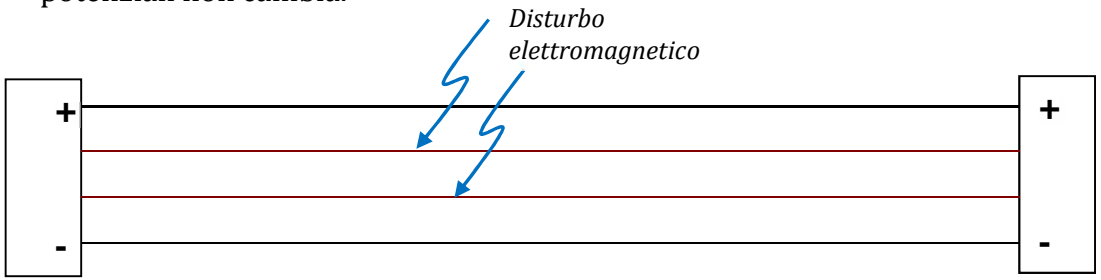
Es: **USB** (Universal Serial Bus), nuovo standard, che nei PC sta sostituendo sia le vecchie porte sia parallele che seriali.

I *nuovi chip* e la *trasmissione differenziale* consentono velocità molto alte:

USB 1.0 1.5 Mb/s
1.1 12 Mb/s
2.0 480 Mb/s
3.0 4.8 Gb/s



Vantaggio della trasmissione differenziale: un disturbo elettromagnetico colpisce in ugual misura i due conduttori (entrambi liberi di variare il loro potenziale) per cui la differenza dei potenziali non cambia.



ARCHITETTURA SOFTWARE di un PC

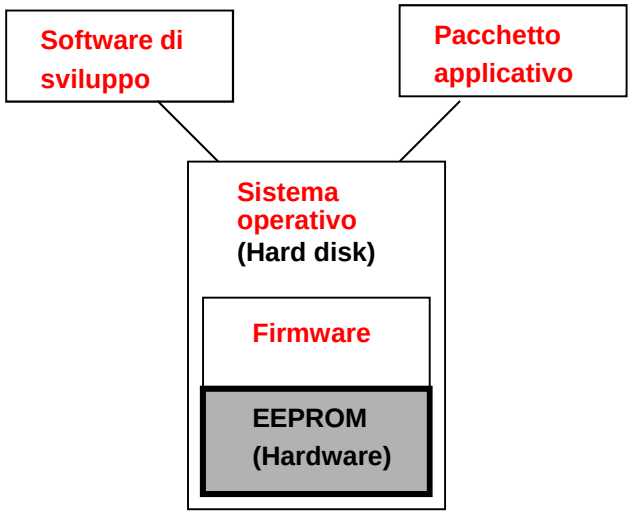
Il software di un PC è organizzato su più livelli:

FIRMWARE (BIOS per PC IBM compatibili): consiste in programmi (routine, cioè procedure) fornite dal costruttore del PC. All'atto dell'avviamento le routine del firmware programmano i chip del sistema, testano la memoria e caricano in memoria il sistema operativo.

SISTEMA OPERATIVO: consiste in una collezione di *programmi utili alla gestione delle risorse del computer*, che dispensano l'utente dalla fatica di dover scriverli in proprio. Si tratta di programmi scritti facendo ricorso alle routine del firmware.

PACCHETTI SOFTWARE APPLICATIVI: i programmi messi a disposizione dal sistema operativo (es: editor di testo) sono spesso poveri di funzionalità. Facendo ricorso alle routine del firmware e del sistema operativo si sono costruiti quindi dei programmi molto più ricchi, già pronti per l'utente.

SOFTWARE DI SVILUPPO: sono linguaggi di programmazione, che permettono la *creazione di applicazioni nuove*. Spesso infatti le esigenze degli utenti non possono essere soddisfatte dai pacchetti applicativi già esistenti.



LINGUAGGI DI PROGRAMMAZIONE

Il primo linguaggio di programmazione è stato il linguaggio macchina: codice operativo e dati erano rappresentati da sequenze di 0 e 1.

Esempio: per µP compatibili con l'8086 della Intel l'operazione *'metti il numero 4 nel registro AX*' costringeva i programmatori a scrivere il seguente codice:

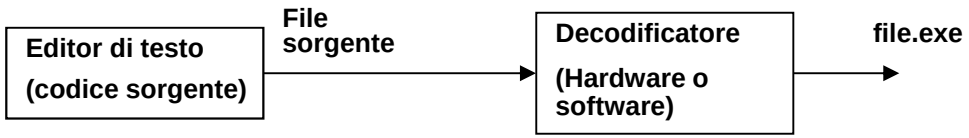
1011 1000 0000 0100

Con il conseguente problema di frequenti *errori di scrittura* e *scarsa leggibilità*

La soluzione fu trovata nella *codifica esadecimale*: B 8 0 4

Si producevano meno errori, ma permaneva la scarsa leggibilità.

Con la codifica esadecimale, dopo aver scritto il programma (*codice sorgente*) si rende necessaria una *decodifica in binario* del codice esadecimale.



Il passo successivo fu la creazione di linguaggi di programmazione con codici mnemonici (assembly):

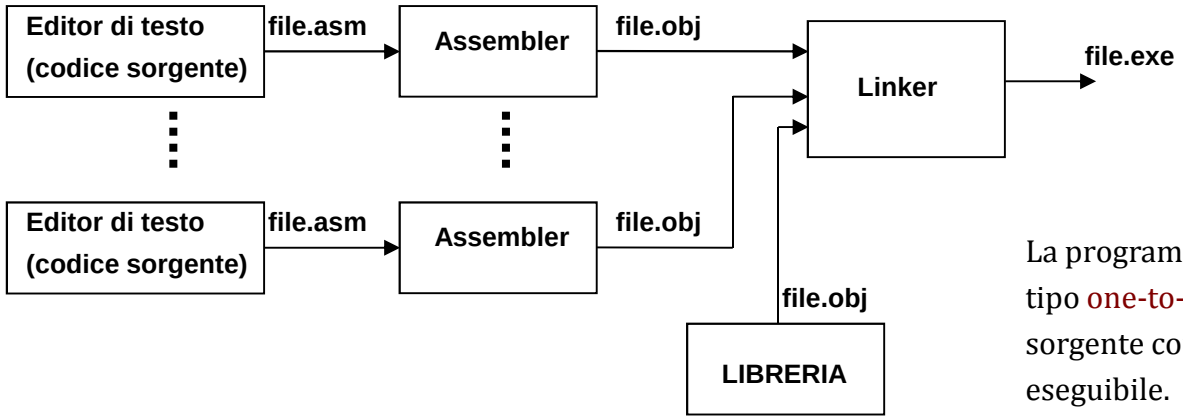
```
MOV  AX,  4
```

L'istruzione diveniva più chiara: il codice operativo MOV esprime chiaramente il senso dell'operazione (mnemonico).

I linguaggi assembly sono definiti di *basso livello*, in quanto richiedono comunque la piena conoscenza del microprocessore da programmare.

La trasformazione del codice sorgente (estensione .asm) in binario è svolta da un programma chiamato assembler, il quale si occupa anche di controllare la sintassi. L'assembler crea un file binario con estensione .obj (file oggetto). Se il programma sorgente è formato da più file, l'assembler esamina un file per volta. E' successivamente compito di un altro programma, il linker, collegare i vari file oggetto e aggiungere il codice delle librerie utilizzate.

Il linker crea il file eseguibile (.exe) senza fissare un indirizzo assoluto cui il programma sarà caricato nella memoria RAM. Si è voluto lasciare al sistema operativo la libertà di caricare il file eseguibile nella zona di memoria al momento più conveniente. Un programma che permette questo al sistema operativo si dice rilocabile.



La programmazione in assembly è del tipo one-to-one: ad una istruzione sorgente corrisponde una istruzione eseguibile.

Difficoltà presenti nella programmazione in assembly:

- la realizzazione di programmi complessi rende molto impegnativa la programmazione
- l'obbligo per il programmatore della piena conoscenza del µP che dovrà eseguire il programma.

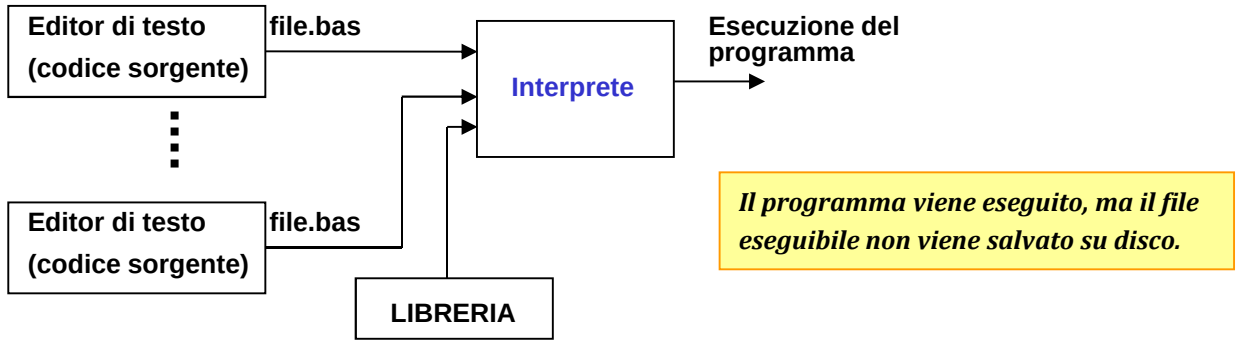
Per superare le difficoltà proprie della programmazione assembly sono stati creati *linguaggi di programmazione evoluti* (detti anche di *alto livello*: Basic, C, Pascal, ...). Presentano il vantaggio

- di consentire la programmazione anche a utenti senza alcuna conoscenza dell'hardware del computer
- di implementare istruzioni compatte (facendo ricorso a librerie) che riducono il lavoro del programmatore.

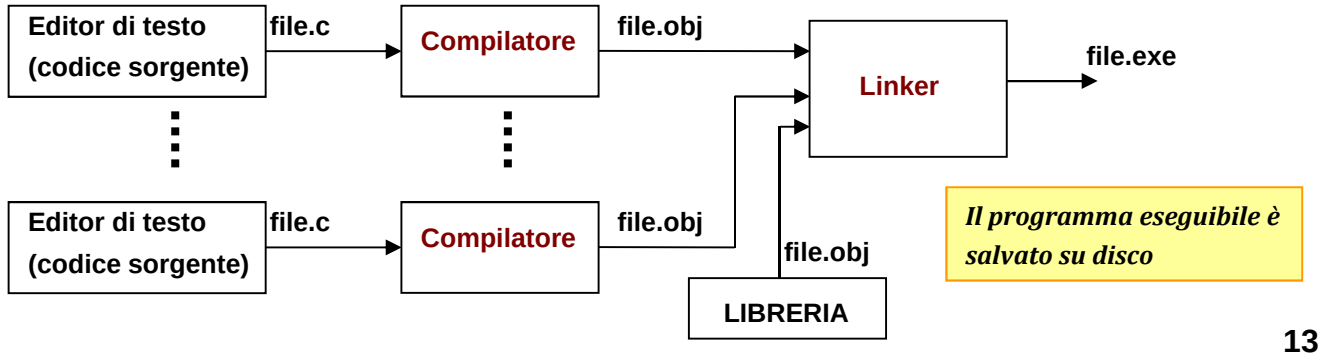
Questi vantaggi sono ottenuti al costo di uno sfruttamento meno ottimale delle risorse del computer (per la caratteristica di essere one-to-many e per la presenza di codice superfluo).

La traduzione del file sorgente in eseguibile può essere realizzata con programmi:

Interprete:



Compilatore:



I *programmi interpreti* sono ottimizzati per una veloce fase di messa a punto del programma che si sta scrivendo (debugging) , ma ogni volta che occorre eseguire un file sorgente lo si deve trasformare in eseguibile. Sul disco non resta alcun file oggetto o eseguibile. I linguaggi di programmazione i cui programmi sono eseguiti per mezzo di un interprete sono detti linguaggi interpretati (es: Basic, Html).

I programmi compilatori sono invece ottimizzati per l'avviamento veloce dell'esecuzione dei file eseguibili, in quanto memorizzano su disco sia i file oggetto (.obj) che quelli eseguibili (.exe), sono però più lenti in fase di messa a punto. I linguaggi di programmazione i cui programmi sono eseguiti per mezzo di un compilatore sono detti linguaggi compilati.

Per il *linguaggio di programmazione C*, si sono costruiti normalmente solo compilatori per cui lo si considera un *linguaggio compilato.*

Confronto Interprete – Compilatore:

> L'esecuzione di un programma sorgente mediante un interprete è meno efficiente, poiché impegna più memoria ed è meno veloce in quanto il codice macchina è generato durante l'esecuzione stessa.
>
> L'esecuzione tramite un compilatore parte invece da un codice già in linguaggio macchina, che viene caricato e immediatamente eseguito.
>
> Tuttavia un interprete consente una maggiore flessibilità nella fase di scrittura e prova di un programma.

Negli ultimi anni si è sviluppato un approccio *ibrido*: software di sviluppo che consentono sia l'interpretazione che la compilazione del codice sorgente.