



Università degli Studi di Palermo  
*Dipartimento di Ingegneria Informatica*



# Elaborazione di Immagini e Suoni / Riconoscimento e Visioni Artificiali 12 c.f.u.

Anno Accademico 2008/2009

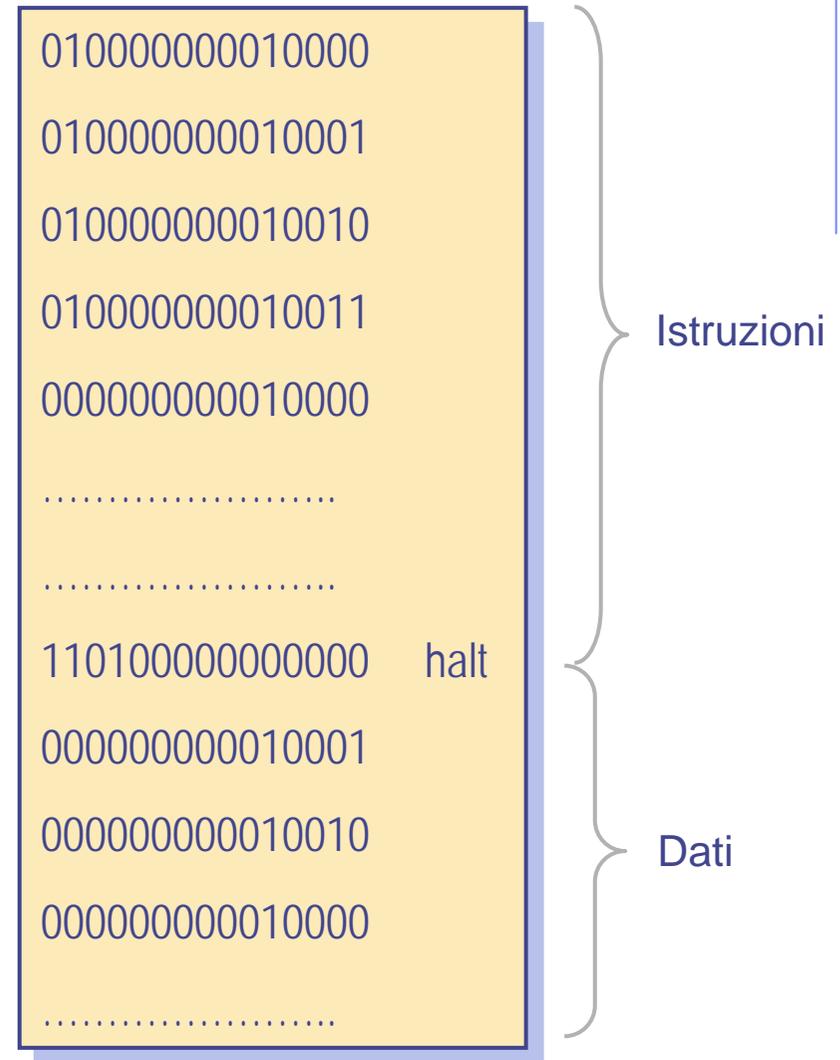
Docente: ing. Salvatore Sorce

## La Macchina Virtuale

Facoltà di Lettere e Filosofia

## Il programma

- Il programma è costituito da una sequenza di istruzioni caricate nella memoria centrale sotto forma di parole (quindi sequenze di bit)
- Esecuzione di un'istruzione:
  - Fase di acquisizione (Fetch)
  - Interpretazione (Decode)
  - Esecuzione (Execute)



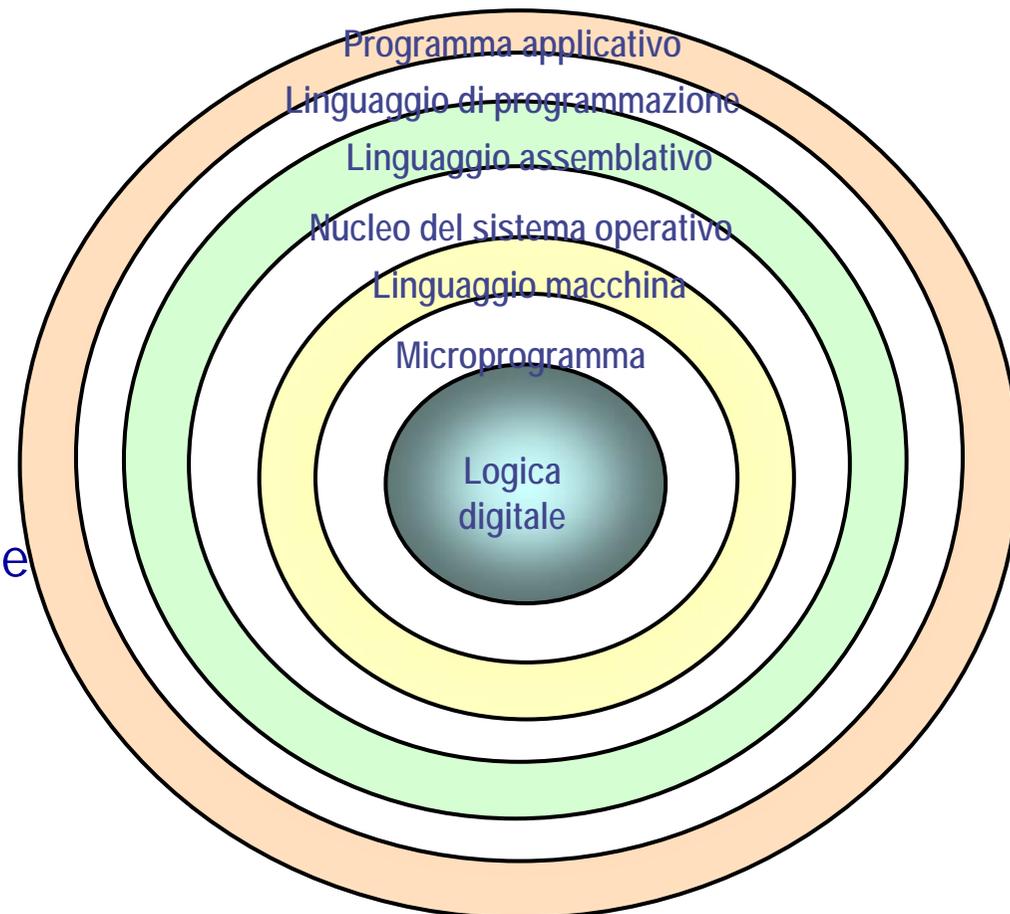


## Interfaccia tra hardware e software

- Nascondere all'utente i dettagli non necessari dell'hardware
- Presentare le informazioni
- Consentire all'utente un facile accesso alle risorse macchina disponibili
- Prevenire danni accidentali o intenzionali ad hardware, programmi e/o dati
  
- Analogia automobile: motore e cruscotto

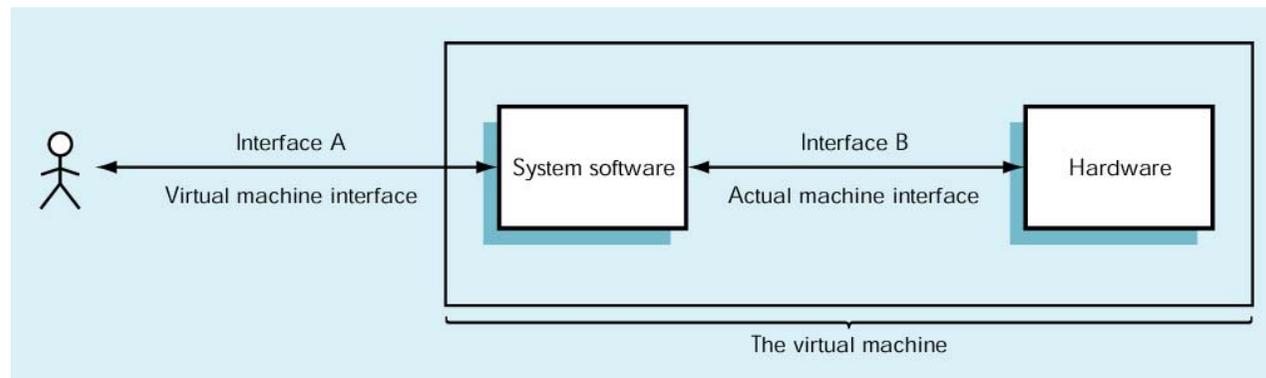
## Gerarchia del software

- Sei livelli di astrazione separano l'utente dall'hardware sottostante
- Microprogramma
- Linguaggio macchina
- Sistema operativo
- Linguaggio assembler
- Linguaggio di programmazione
- Programma applicativo



## Software di sistema

- Software di sistema
  - Raccolta di programmi per la gestione delle risorse di un calcolatore e della loro accessibilità
  - Agisce da intermediario tra utente e hardware
- Macchina virtuale
  - Insieme dei servizi e delle risorse generate dal sw di sistema
- Il software di sistema è l'analogo del cruscotto per una macchina di Von Neumann





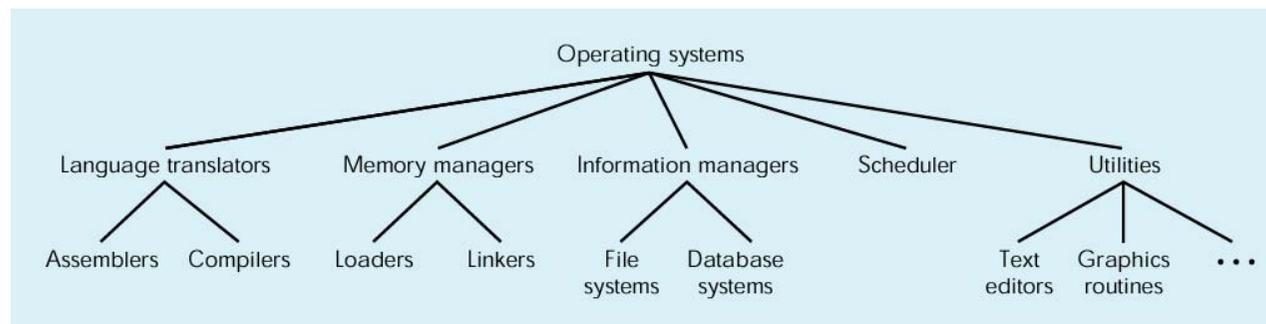
## Software di sistema

- Caratteristiche del software di sistema
  - Nascondere all'utente i dettagli dell'hw
  - Presentare le informazioni in modo semplice
  - Consentire un facile ed efficiente accesso alle risorse
  - Fornire un ambiente sicuro in cui operare
- Es.: somma di due numeri ( $a=b+c$ )
  - Carica registri ALU con i dati dalla memoria, agli indirizzi relativi alle variabili  $b$  e  $c$
  - Attiva ALU
  - Memorizza il risultato prodotto dalla ALU nella memoria, all'indirizzo relativo alla variabile  $a$



## Classificazione del software di sistema

- Sistema operativo
  - Programma che supervisiona tutte le operazioni di un calcolatore
  - Comunica con l'ambiente esterno, gestisce l'attivazione di periferiche e altre componenti sw
- Classi di programmi di sistema
  - Traduttori
  - Gestori della memoria
  - File system
  - Scheduler
  - Programmi di utilità





## Classificazione del software di sistema

- Traduttori
  - Assemblatori, compilatori ed interpreti
  - Consentono di descrivere algoritmi in un linguaggio orientato all'utente
- Gestori della memoria
  - Riservano spazio in memoria per dati e programmi
  - Caricano in memoria i programmi prima dell'esecuzione
- File system
  - Gestiscono la memorizzazione e il recupero di informazioni sui dispositivi di memoria di massa
- Schedulatore
  - Gestisce l'elenco con priorità dei programmi pronti per l'esecuzione
  - Seleziona il programma prossimo da eseguire (prioritarizzazione)
- Programmi di utilità
  - Librerie di programmi che forniscono servizi sia all'utente che ad altri programmi



## Applicazione programmi di sistema

- Scrivere un programma, eseguirlo e salvare i risultati
  1. Editor di testi  
Scrittura programma in linguaggio ad alto livello
  2. File system  
Memorizzare il programma come file di testo su disco fisso
  3. Traduttore  
Trasformare il programma dal linguaggio ad alto livello in linguaggio macchina
  4. Caricatore (loader)  
Riserva spazio in memoria per il programma, e caricare istruzioni per l'esecuzione
  5. Scheduler  
Esegue il programma ogni qualvolta è il suo turno
  6. File system  
Memorizza i dati generati
  7. Debugger  
In caso di errori, esegue il programma passo-passo e tracciare l'errore

## Linguaggio macchina

- Linguaggio macchina
  - Formato binario. Le istruzioni sono indistinguibili dai dati su cui operano
  - Non consente l'uso di etichette o simboli per indicare locazioni di memoria o istruzioni adibite a compiti specifici
  - Difficile da modificare. Gli indirizzi delle istruzioni si susseguono sequenzialmente a partire dalla prima.
  - Difficile creare dati. I dati possono solo essere rappresentati nel loro formato interno
- I calcolatori della prima generazione potevano essere programmati soltanto in linguaggio macchina!

## Linguaggio assembler

- Linguaggio assembler
  - Orientato sia alla macchina che all'utente
  - Linguaggio di seconda generazione, contrapposto al linguaggio macchina o di prima generazione
- Le istruzioni sono indicate con etichette comprensibili che vengono tradotte nel codice binario corrispondente dal traduttore
- Codici mnemonici
  - ADD – addizione
  - SUB – sottrazione
  - LOAD, STORE – carica da memoria, memorizza in memoria
  - JUMP – salta ad istruzione successiva
- Rapporto 1:1 con il linguaggio macchina
  - Ogni istruzione in linguaggio assembler è tradotta esattamente nella sua corrispondente in linguaggio macchina
  - Specifico per una particolare classe di microprocessori

## Esempio di file oggetto

Indirizzo	Opcode data	Significato
0000	1101 000000001001	IN X
0001	1101 000000001010	IN Y
0010	0000 000000001001	LOAD X
0011	0111 000000001010	COMPARE Y
0100	1001 00000000111	JUMPGT DONE
0101	1110 000000001001	OUT X
0110	1000 000000000000	JUMP LOOP
0111	1110 000000001010	OUT Y
1000	1111 000000000000	HALT
1001	0000 000000000000	CONST 0
1010	0000 000000000000	CONST 0

Linguaggio macchina

Linguaggio assembler

## Esempio di file oggetto

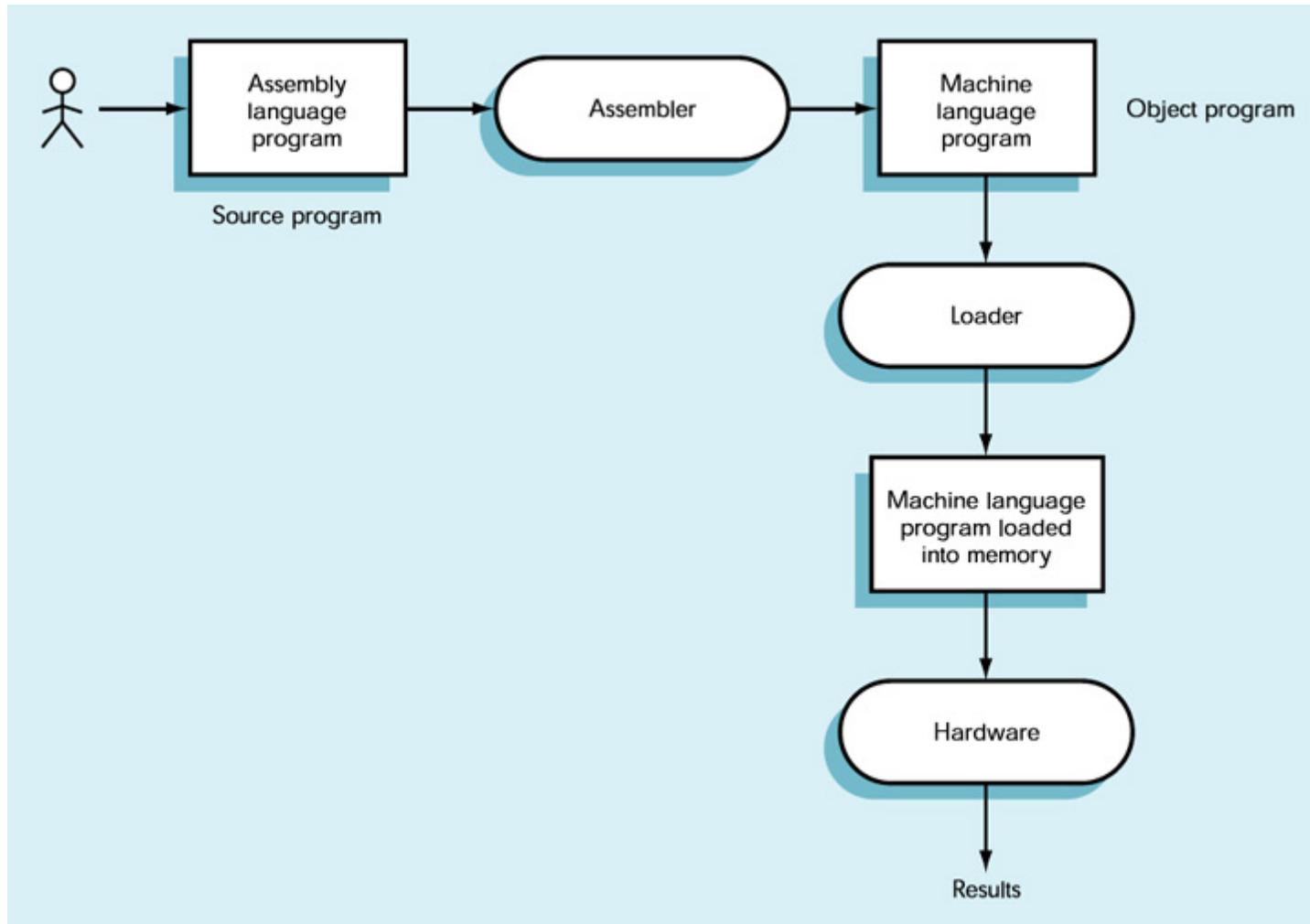
Indirizzo	Opcode data	Significato
0000	1101 000000001001	IN X
0001	1101 000000001010	IN Y
0010	0000 000000001001	LOAD X
0011	0111 000000001010	COMPARE Y
0100	1001 00000000111	JUMPGT DONE
0101	1110 000000001001	OUT X
0110	1000 000000000000	JUMP LOOP
0111	1110 000000001010	OUT Y
1000	1111 000000000000	HALT
1001	0000 000000000000	CONST 0
1010	0000 000000000000	CONST 0

Il primo indirizzo  
di un file oggetto  
è sempre 0

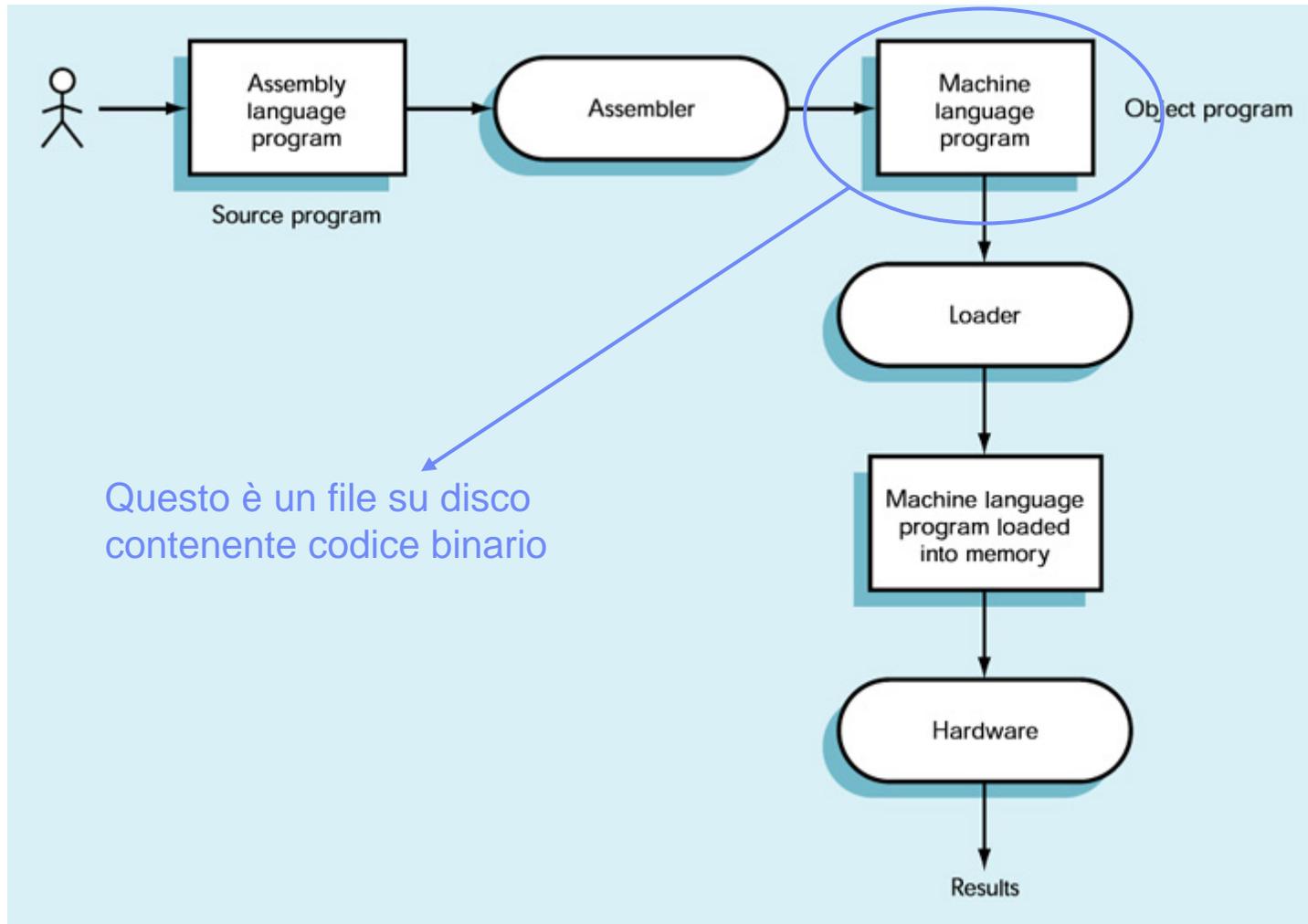
Linguaggio macchina    Linguaggio assembleativo



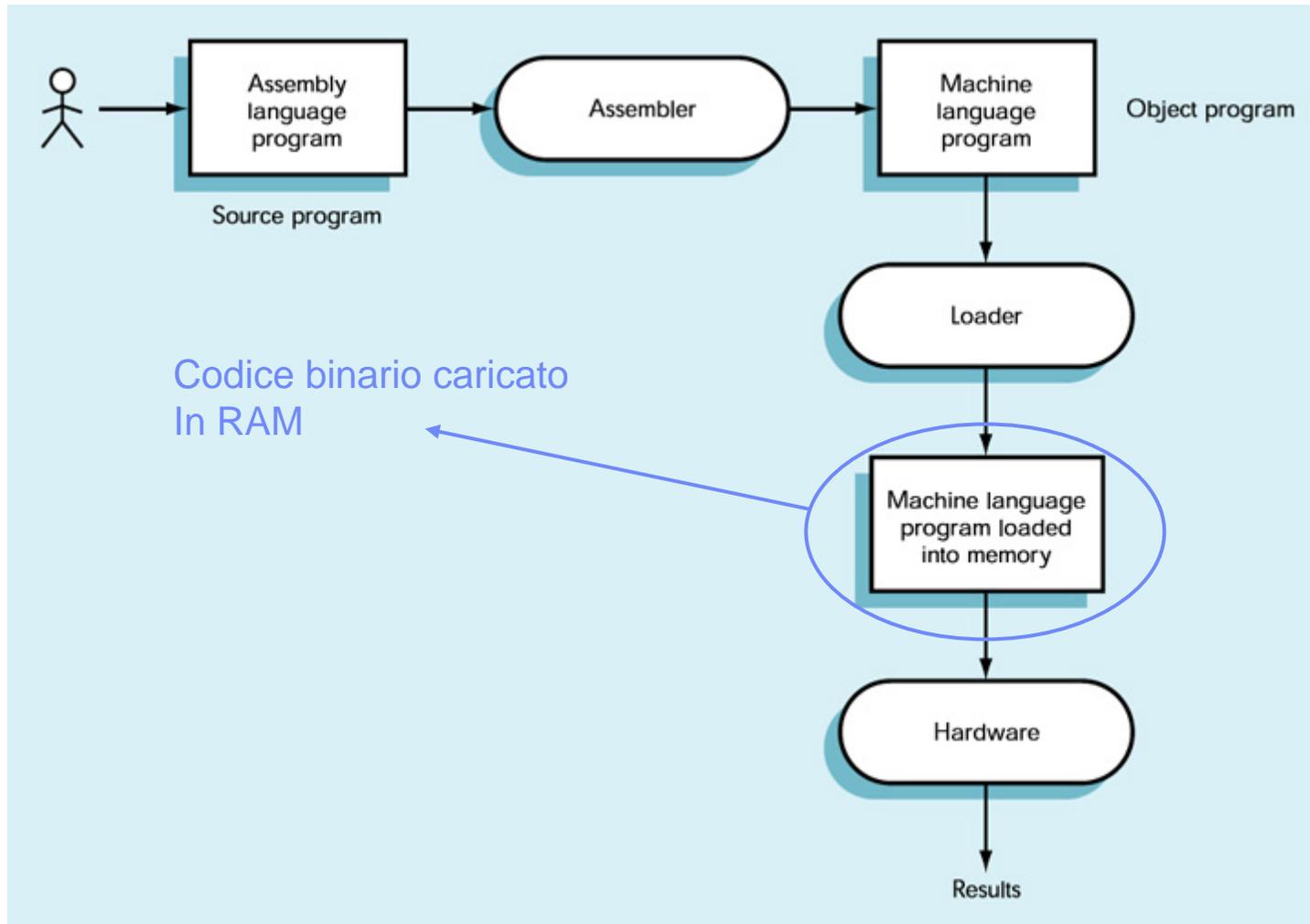
## Traduzione/caricamento/esecuzione



# Traduzione/caricamento/esecuzione



# Traduzione/caricamento/esecuzione



## Caratteristiche del linguaggio assembler

- Vantaggi rispetto al linguaggio macchina
  - Uso di codici operativi simbolici (mnemonici) anziché numerici
  - Uso di indirizzi di memoria simbolici anziché numerici
  - Pseudo-operazioni che forniscono servizi all'utente, come la generazione di dati
- Formato tipico di una istruzione  
etichetta: mnemonico      campo\_indirizzo -- commento
- Caratteristiche aggiuntive
  - Chiarezza dei programmi
  - Manutenibilità



## Traduzione e caricamento

- **Compiti dell'assemblatore**
  - Convertire i codici mnemonici in formato binario
  - Convertire gli indirizzi simbolici in formato binario
  - Eseguire i servizi richiesti dalle pseudo-operazioni
  - Caricare le istruzioni generate (codice oggetto) in un file per uso futuro
- **Il loader legge il file oggetto dal disco fisso e lo trasferisce in memoria principale per l'esecuzione**
- **Processo di assemblaggio**
  - Richiede due passaggi sul codice sorgente
  - Passaggio: processo di esame e traduzione di tutte le istruzioni, analizzate una dopo l'altra
- **Primo passaggio**
  - Costruzione tabella dei simboli
  - Collegamento (binding) degli indirizzi simbolici ad indirizzi fisici
- **Secondo passaggio**
  - Traduzione istruzioni simboliche in codici binari
  - Generazione file oggetto



## Evoluzione dei linguaggi di programmazione

- Nascita dei linguaggi ad alto livello (BASIC, Pascal, C++, Java)
  - Orientati all'utente (più vicini al linguaggio naturale)
  - Indipendenti dalla particolare macchina
  - Rapporto 1:N con il linguaggio macchina: una istruzione ad alto livello richiede tipicamente N istruzioni in linguaggio macchina

