UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

Catena programmativa e Assembly

Prof.ssa Giulia Cisotto

giulia.cisotto@units.it

Trieste, 21 marzo 2025

AGENDA DI IERI/OGGI

- *Organizzazione della memoria centrale*
- *Istruction Set Architecture (ISA)*
- **Catena programmativa**

DAL PROGRAMMA SORGENTE ALL'ESECUZIONE

Linguaggio naturale, input nel PC

Rappresentazione dell'informazione in linguaggio macchina, flusso dati nel PC

Output dal PC, linguaggio naturale

UNIVERSITÀ DEGLI STUDI DI TRIESTE

DAL PROGRAMMA SORGENTE ALL'ESECUZIONE

Programmatore: scrive il testo del programma in linguaggio sorgente di alto livello (più vicino al linguaggio «naturale» umano)

Compilatore: Traduce il codice sorgente scritto in un linguaggio di alto livello in codice assembly MIPS, ottimizzando le istruzioni per migliorare le prestazioni.

Assemblatore: Converte il codice assembly in codice macchina (binario) generando *file oggetto* contenenti le istruzioni eseguibili dal processore MIPS. Questo codice è quello che viene caricato in memoria*.

Linker: Combina diversi file oggetto e librerie, risolvendo i riferimenti ai simboli (funzioni e variabili), per produrre un programma eseguibile.

***Loader:** Carica l'eseguibile nella memoria del sistema, preparando l'ambiente di esecuzione e avviando l'esecuzione del programma sul processore MIPS.

Nota: spesso con il termine compilazione si indica l'intero processo di traduzione da linguaggio ad alto livello a linguaggio macchina (essendo l'assemblatore spesso integrato con il compilatore)

LINGUAGGI DI PROGRAMMAZIONE

Linguaggi ad alto livello (Python, Matlab, R, Java, ..)

- **Notazione vicina al linguaggio naturale** e alta notazione algebrica (maggiore espressività e leggibilità)
- Incremento di produttività
- La programmazione è **svincolata dalla conoscenza dei dettagli architetturali della macchina**
- Indipendenza dalle caratteristiche dell'architettura (processore) su cui il programma sarà eseguito (**portabilità**)
- Ideati non per essere compresi direttamente da macchine reali, ma da macchine astratte, in grado di effettuare operazioni di più alto livello rispetto alle operazioni dei processori reali
- Permettono l'uso di librerie di funzionalità già scritte (riusabilità del codice)

Linguaggio assembler (Assembly)

Vantaggi. La dipendenza dall'architettura del calcolatore permette di:

- Ottimizzare le prestazioni (maggiore efficienza)
- Programmi (potenzialmente) più compatti
- Massimo sfruttamento delle potenzialità dell'hardware sottostante
- Molto importante per programmare controller di processi e macchinari (e.g., real-time), o per apparati limitati (e.g., embedded computer, portatili)

Svantaggi:

- Minore espressività: es., strutture di controllo limitate
- Necessario conoscere i dettagli dell'architettura
- Mancanza di portabilità su architetture diverse
- Difficoltà di comprensione
- Lunghezza maggiore dei programmi

CATENA PROGRAMMATIVA

Compiler, Assembler, Linker

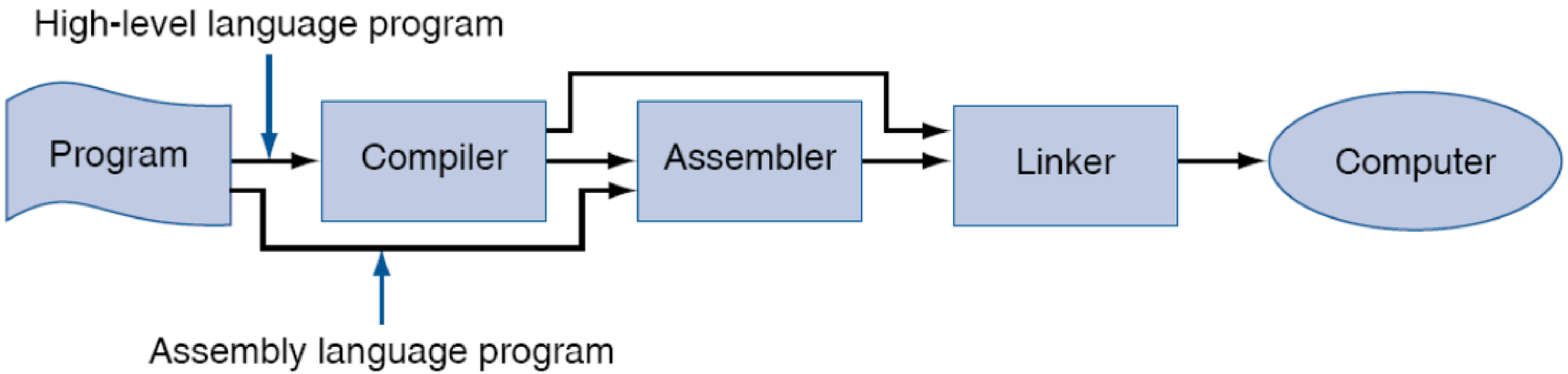

FIGURE A.1.6 Assembly language either is written by a programmer or is the output of a compiler.

CATENA PROGRAMMATIVA

Assembler, Linker

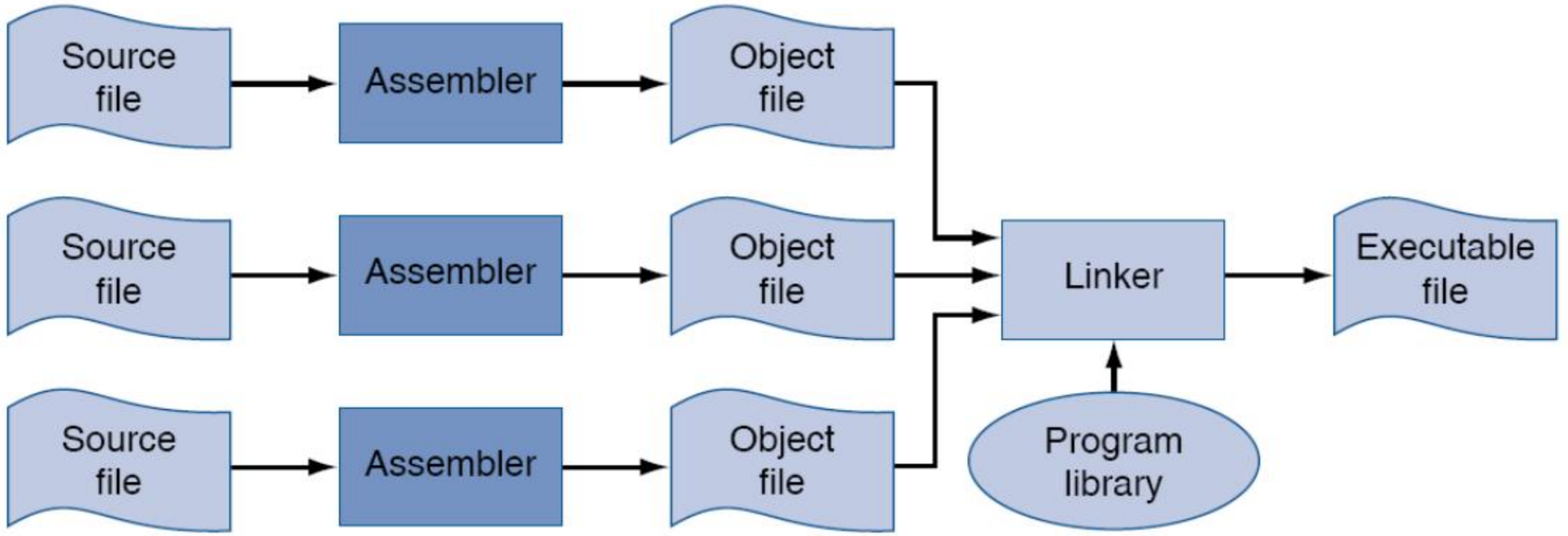

FIGURE A.1.1 The process that produces an executable file. An assembler translates a file of assembly language into an object file, which is linked with other files and libraries into an executable file.

ASSEMBLER (1/3)

- Converte un programma assembler (file sorgente) in linguaggio macchina (file oggetto)
- Gestisce le etichette
- Gestisce pseudoistruzioni
- Gestisce numeri in base diverse (e.g., binario, decimale, esadecimale)

L'assemblaggio è un procedimento sequenziale che esamina, riga per riga, il codice sorgente Assembly, traducendo ciascuna riga in un'istruzione del linguaggio macchina.

Applicato modulo per modulo al programma e costituisce per ogni modulo la **tabella dei simboli** del modulo.

2 passi importanti:

1. Traduce i codici mnemonici (simbolici) delle istruzioni nei corrispondenti codici binari
2. Traduce i riferimenti simbolici (variabili, registri, etichette di salto, parametri) nei corrispondenti indirizzi numerici.

ASSEMBLER (2/3)

Poiche' le etichette di salto generano il problema dei riferimenti in avanti (ossia, riferimenti ad etichette successive o contenute in altri file), **l'assemblatore deve leggere il programma sorgente due volte**.

Ogni lettura del programma sorgente è chiamata **passo** e **l'assemblatore è chiamato traduttore a due passi.**

Materiale per la lezione

- *Hennessy-Patterson, cap. 1 pp.14-16*
- *Appendix A.2, A.3, A.4, A.5*

Prossima lezione: 26 ***marzo, h.9:00****, aula 4C*