Module 9

Linking - Loading

Module Nine

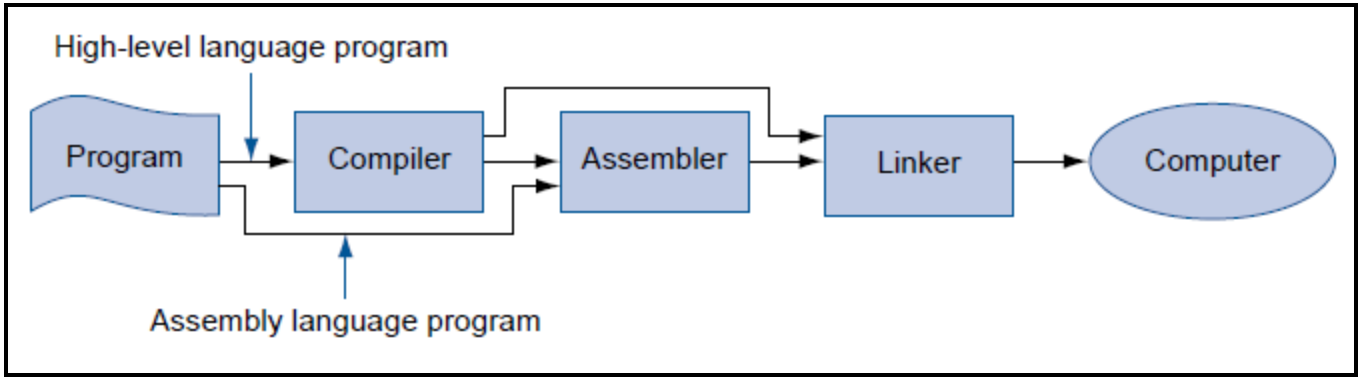
- This week, we are going to talk about :
- Program Loading and Linking
  - Basic Functions
  - Algorithms
  - Program Relocation
  - Additional Link-Editor Features
  - Design Options
- Material adapted from L. Beck, System Software,1997

Linking - Loader

- System Software
- Reads Object program file created by Assembler or Compiler
  - Processes each record one at a time
- Creates executable file
- Copies Machine Language into memory

Program Loading - Linking

- Program Loading and Linking
  - Combine two or more separate object programs and supply the information needed to allow references between the programs.
  - Copy the program into memory
  - LINKING LOADER



Copy program into memory

- But first …
- Program called a Loader
- System software
  - Simple loader
  - Relocating loader
  - Virtual memory loader
  - Dynamic link loader
- Operating system sets parameters
  - Memory Address

Basic Functions

- Basic Loader or Absolute Loader
  - Copy object program into memory and start the execution.
  - Absolute program from a basic Assembler
  - Linking not required
  - Relocation not required
- Process
  - Check Header record for program name and size.
  - Put object code into memory as directed by the Text records.
  - Jump to the location specified by the End record to start execution.

Object File

- Object file refresher
- Created by the Assembler
- Standard format specified by the Operating System
- a.out

Object File

- Example from Unix

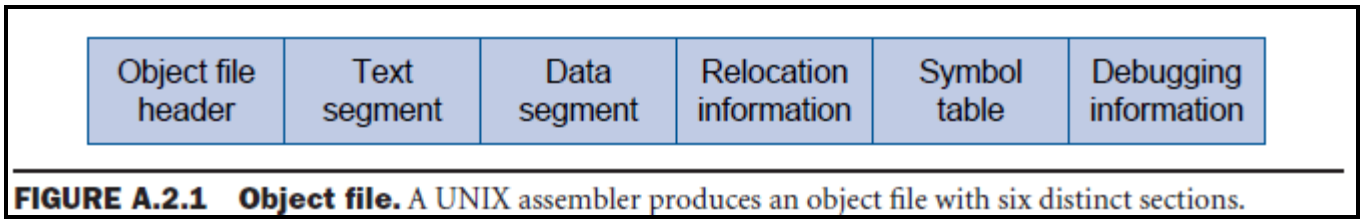


FIGURE A.2.1 Object file. A UNIX assembler produces an object file with six distinct sections.

  - Text - machine language code for instructions in source file.
  - Data - binary representation of the data in source file.
  - Relocation - identifies instructions and data that depend on absolute addresses. The references must change when the program is moved in memory.
  - Symbol table - external labels defined in source, and unresolved label references.

A simple Object File format

- Simple - Concepts without unnecessary details

- Header Record H COPY 001000 00107A
- Text Record T 001000 1C 27BDFFE0 AFBF0014 AFA40020
- End Record E 001000

Simple Object File Format (continue)

- Header Record

(1)	H
(2) - (7)	Program name
(8) - (13)	Starting Address (hex)
(14) - (19)	Program Length (hex)

```
H COPY    001000 00107A
^         ^      ^      ^
```

Simple Object File Format (continue)

- Text Record

(1)	T
(2) - (7)	Starting Address of this object code (hex)
(8) - (9)	Length of this record in bytes (hex)
(11) - (69)	Actual Object code (hex)

```
T 001000 1C 27BDFFE0 AFBF0014 AFA40020
^        ^  ^        ^        ^        ^
```

Simple Object File Format (continue)

- End Record

(1)	E
(2) - (7)	Address of first executable instruction (hex)

E 001000
^ ^

Simple Object File Format (continue)

- Sample

```
H COPY   001000 00107A
T 001000 1C  27BDFFE0 AFBF0014 AFA40020  ….
T 00101C 0C  25C80001  29010065  AFA8001C
E 001000
```

Algorithm

- Process
  - Get **Program Address** value from Operating system
  - Read Header record
  - Verify name and length
  - Read Text records
  - Insert machine code values at specified locations
  - Read End record
  - Jump to first executable instruction

Boot Strap Loader

- How to start an idle machine ?
- Power ON
- Simple Absolute loader - predefined fixed address
- Hard wired
- Operator switch setting

Boot Strap Loader

- ROM - read only memory
- Read one record from selected device
- Bootstrap loader
  - First record read that starts the process
  - Program can be fewer than 80 bytes
- Copies Operating System
  - BIOS

Summary

- Program Loading
  - Basic Function
  - Object file
  - Algorithm

Next: Program Linking