Software Systems2

RELOCATING LOADER LECTURE02 2023

Dr. Azam E. Al-Rawachy
SOFTWARE DEPARTMENT | COLLEGE OF COMPUTER SCIENCE AND MATHEMATICS

1 Introduction

With the advent of operating systems, relocating loaders separate from linkers and libraries became necessary. Before operating systems, each program had the machine's entire memory at its disposal, so the program could be assembled and linked for fixed memory addresses, knowing that all addresses in the computer would be available. But with operating systems, the program had to share the computer's memory with the operating system and perhaps even with other programs, this means that the actual addresses at which the program would be running weren't known until the operating system loaded the program into memory.

Linkers and loaders now divided up the work, with linkers doing part of the address binding, assigning relative addresses within each program, and the loader doing a final relocation step to assign actual addresses. A relocating loader loads a program in a designated area of memory relocates it so that it can execute correctly in that area of memory and passes control to it for execution.

Address Binding

Usually, a program resides on a disk as a binary executable file. To be executed, the program must be brought into memory and placed within a process. Depending on the memory management in use, the process may be moved between disk and memory during its execution. The processes on the disk are waiting to be brought into memory for execution from the input queue.

The normal procedure is to select one of the processes in the input queue and to load that process into memory. As the process is executed, it accesses instructions and data from memory. Eventually, the process terminates, and its memory space is declared available.

Addresses in the source program are generally symbolic (such as count). a compiler will typically bind these symbolic addresses to relocatable addresses. The linkage editor or loader will in turn bind the relocatable addresses to absolute addresses. Each binding is mapping from one address space to another. Address Binding is a process that fixes a physical address to the logical address of a process's address space.

There are two different types of address expression:

• An address expression that has a fixed value, independent of run-time considerations such as where the program is located in memory, is called an **absolute address** expression.

• An address expression with a value that depends on run-time considerations is called a **relocatable address** expression.

An example of Relocating Loader is written below: Let: PRG: Source Program

```
BEGIN:   JMP     MAIN          EB --
N1       DW      ?
N2       DW      -6
RES      DW      0, 0
MAIN:    MOV     AX,N1         A1 -- --
         MUL     WORD PTR N2   F6 26 -- --
         CMP     DX, 20H       83 FA --
         JB      CONT          72 --
         XOR     DX, DX        32 D2
CONT:    MOV     RES,AX        A3 -- --
         MOV     RES+2, DX     89 16 -- --
         RET                   C3
         END
```

PASS-1

Intermediate File:

Address	Mnemonic Field			Error Flag
0000	BEGIN:	JMP	MAIN	0
0002	N1	DW	?	0
0004	N2	DW	-6	0
0006	RES	DW	0, 0	0
000A	MAIN:	MOV	AX, N1	0
000D		MUL	WORD PTR N2	0
0011		CMP	DX, 20H	0
0014		JB	CONT	0
0016		XOR	DX, DX	0
0018	CONT:	MOV	RES, AX	0
001B		MOV	RES+2, DX	0
001F		RET		0
0020		END		0

SYMTAB:

Symbol	Value	Error Flag
BEGIN:	0000	0
N1	0002	0
N2	0004	0
RES	0006	0
MAIN:	000A	0
CONT:	0018	0

PASS-2

Listing File:

Line No.	Address	Mnemonic Field			Machine Code	Error Flag
1	0000	BEGIN:	JMP	MAIN	EB 08	0
2	0002	N1	DW	?	00 00	0
3	0004	N2	DW	-6	FA FF	0
4	0006	RES	DW	0, 0	00 00 00 00	0
5	000A	MAIN:	MOV	AX, N1	A1 02 00	0
6	000D		MUL	WORD PTR N2	F6 26 04 00	0
7	0011		CMP	DX, 20	83 FA 20	0
8	0014		JB	CONT	72 02	0
9	0016		XOR	DX, DX	33 D2	0
10	0018	CONT:	MOV	RES, AX	A3 06 00	0
11	001B		MOV	RES+2, DX	89 16 08 00	0
12	001F		RET		C3	0
13	0020		END			0

Object Program:

H PRG --- 0000 0020

T 0000 10 EB 08 00 00 FA FF 00 00 00 00 A1 02 00 F6 26 04

T 0010 10 00 83 FA 20 72 02 33 D2 A3 06 00 89 16 08 00 C3

M 000B + PRG

M 000F + PRG

M 0019 + PRG

M 001D +PRG

E 0000

	Address	M.C
BEGIN	0000	EB
	0001	08
N1	0002	00
	0003	00
N2	0004	FA
	0005	FF
RES	0006	00
	0007	00
	0008	00
	0009	00
MAIN:	000A	A1
Modify	000B	02
	000C	00
	000D	F6
	000E	26
Modify	000F	04
	0010	00
	0011	83
	0012	FA
	0013	20
	0014	72
	0015	02
	0016	33
	0017	D2
CONT:	0018	A3
Modify	0019	06
	001A	00
	001B	89
	001C	16
Modify	001D	08
	001E	00
	001F	C3

PROGADDR (program load address) from OS=100h

CSADDR Control section = PROGADDR

Symbol Address Length = 20H

After loading the Machine code into memory starting address at 100H

	Address	M.C
BEGIN	0100	EB
	0101	08
N1	0102	00
	0103	00
N2	0104	FA
	0105	FF
RES	0106	00
	0107	00
	0108	00
	0109	00
MAIN:	010A	A1
	010B	02
	010C	01
	010D	F6
	010E	26
	010F	04
	0110	01
	0111	83
	0112	FA
	0113	20
	0114	72
	0115	02
	0116	33
	0117	D2
CONT:	0118	A3
	0119	06
	011A	01
	011B	89
	011C	16
	011D	08
	011E	01
	011F	C3