INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

System Software
CSN-252
Loaders and Linkers

Chapter 3 L. L. Beck

A Simple Bootstrap Loader

- Bootstrap Loader
  - When a computer is first turned on or restarted, a special type of absolute loader, called *bootstrap loader* is executed
  - This bootstrap loads the first program to be run by the computer -- usually an operating system
- SIC bootstrap loader (SIC Simulator)
  - The bootstrap itself begins at address 0 (DEV00)
  - It loads the OS / some pgm. starting at address 0x80 (DEVF1)
  - No header and end record or control information,
  - The object code is consecutive bytes of memory on device 00

SIC Bootstrap Loader Logic

Begin

X=0x80 (the address of the next memory location to be loaded

[LDA ZERO; LDX HEX80]

Loop

A←GETC (read from F1 and convert it from the ASCII character code to the value of the hexadecimal digit)

save value in the high-order 4 bits of rightmost byte of S

A←GETC (…)

combine the value to form one byte A← (A+S)

store the value (in A) to the address in register X

X←X+1

End

0~9 : 48

A~F : 65

```
GETC  A ← read one character
      if A=0x04 then jump to 0x80
      if A<48 then GETC
      A ← A-48 (0x30)
      if A<10 then return
      A ← A-7 (48+7=55)
      return
```

Reading Assignment: Study the assembly language code of bootstrap loader of SIC/XE

When power is applied (Intel Processor)

- Processor comes on in what is called 16-bit **Real Mode** with instruction pointer pointing to address 0xffff.fff0, the reset vector
- On reset, IP contains value 0xfff0 and CS has value 0xf000
- on reset, the system is in a "**special" Real Mode**, where the first 12 address lines are asserted
- These 12 address lines remain asserted until a long JMP is executed
- Processor executes the first instruction from BIOS region of the BIOS chip (boot firmware)
- One of the first things that the boot firmware does is switch to 32-bit mode. It is also "**protected mode**"
- Some initializations are done (including DRAM) and it searches for bootable device
- Once the BIOS has found a bootable device it loads the boot sector
- The boot sector code is the first-stage boot loader
- Control is handed over to next stage loader which is usually an OS loader like GRUB2 or LILO

Relocating Loaders

- Motivation
  - **efficient sharing of** the machine with larger **memory** and when several independent programs are to be run together
  - support the use of subroutine libraries efficiently
  - absolute addresses shouldn't be used in case of subroutines
- Two methods for specifying relocation
  - modification record
  - relocation bit
    - each instruction is associated with one relocation bit
    - these relocation bits in a Text record is gathered into bit masks

Modification Record

```
1          copy      start  0
2  0000    first     stl    retadr     172016
3  0003              ldb    #length
4                    base   length
5   0006   cloop     +jsub  rdrec      4b10101f
6   000a             lda    length
7   000d             comp   #0
8   0010             jeq    endfil
9   0013             j      cloop
10  0016   endfil     j      @retadr
11  0019   retadr     resw   1
12  001c   length     resw   1
13  001f   buffer     resb   4096
```

```
14     101f   rdrec   clear   x
15                    clear   a
16                    clear   s
17                    +ldt    #4096      75101000
18             loop   td      input
19                    jeq     loop
                      .
                      .
```

- A modification record is used to describe each part of the object code that must be changed when program is relocated

  M00000705+copy

```
begin
        Get PROGADDR from OS
        while not end of input do
        begin
                read next record
                while record type != 'E' do
                begin
                        read next input record
                        while record type = 'T' do
                                move object code from record to location
                                PROGADDR + specified address
                        while record type = 'M' do
                                add PROGADDR at the location
                                PROGADDR + specified address
                end
        end
end
```

```
1                copy        start   1000
2    1000        eof         byte    c'eof'      454f46
3    1003        zero        word    0           000000
4    1006        retadr      resw    1
5    1009        length      resw    1
6    100c        buffer      resw    4096
7     200c       first       stl     retadr      141006
8     200f       cloop       jsub    rdrec       482024
9     2012                   lda     length      001009
10    2015                   comp    zero        281003
11    2018                   jeq     endfil      30201e
12    201b                   j       cloop       30200f
13    201e       endfil      ldl     retadr      081006
14   2021                    rsub                4c0000
                             :
15  2024         rdrec       ldx     zero        041003
16  2027                     lda     zero        001003
```

Relocation Bit

- For machines that primarily use direct addressing and has a fixed instruction format

Text record
col 1: T
col 2-7: starting address
col 8-9: length (byte)
col 10-12: relocation bits
col 13-72: object code

- Relocation bit with each word / instruction
  - 0: no modification is necessary
  - 1: modification is needed
- Twelve-bit mask is used in each Text record
  - since each text record contains less than 12 words
  - unused words are set to 0
  - any value that is to be modified during relocation must coincide with one of these 3-byte segments
- If relocation bit corresponding to a word of object code is set to 1 – add program's starting address at the time of relocation

```
HCOPY__001000 00aaaa
T001000 06 000 454f46000000
T00200c 1E FEC 141006 482024 001009 281003 30201e 30200f
                  081006 4c0000 041003 001003
T…
E001000
….
While record type = 'T' {
        location = starting address;
        length =
        mask bits (M) =
        for (i = 0; i < length/3; i++) {
                move object code from  record to address
                                PROGADDR + location
                if (M_i = 1) then
                        add PROGADDR at the address
                        PROGADDR + location
        }
        read next record }
```

Starting address of the program in memory

1		copy	start	1000	
2	1000	eof	byte	c'eof'	454f46
3	1003	zero	word	0	000000
4	1006	retadr	resw	1	
5	1009	length	resw	1	
6	100c	buffer	resw	4096	
7	200c	first	stl	retadr	141006
8	200f	cloop	jsub	rdrec	482024
9	2012		lda	length	001009
10	2015		comp	zero	281003
11	2018		jeq	endfil	30201e
12	201b		j	cloop	30200f
13	201e	endfil	ldl	retadr	081006
14	2021		rsub		4c0000
15	2024	abc	byte	x'f1'	f1
			:		
15	2025	rdrec	ldx	zero	041003
16	2028		lda	zero	001003

HCOPY__001000 00aaaa
T001000 06 000 454f46000000
T00200c 1E FFC 141006 482024 001009 281003 30201e 30200f
081006 4c0000 041003 001003
T…
E001000

HCOPY__001000 00aaaa
T001000 06 000 454f46000000
T00200c 19 FF0 141006 482024…4c0000 f1
T002025 06 C00 041003 001003
E001000