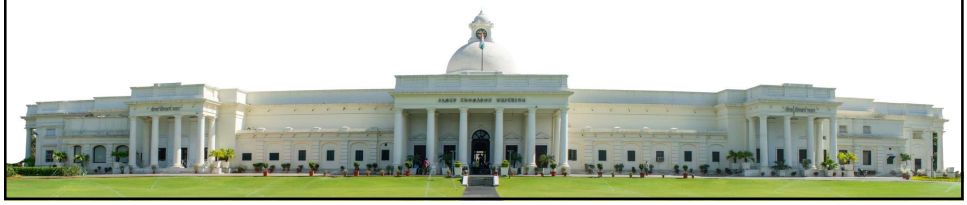
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

System Software
CSN-252
Assembler



```
1  COPY      START  0
2  FIRST     STL    RETADR
3            LDB    #LENGTH
4            BASE   LENGTH
5  CLOOP     +JSUB  RDREC
6            LDA    LENGTH
7            COMP   #0
8            JEQ    ENDFIL
9            J      CLOOP
10 ENDFIL    J      @RETADR
11 RETADR    RESW   1
12 LENGTH    RESW   1
13 BUFFER    RESB   4096
             :
14 RDREC     CLEAR  X
15           CLEAR  A
16           CLEAR  S
17           +LDT   #4096
18 RLOOP     TD     INPUT
19           JEQ    RLOOP
20           RD     INPUT
21           COMPR  A,S
22           JEQ    EXIT
23           STCH   BUFFER,X
24           TIXR   T
25           JLT    RLOOP
26 EXIT      STX    LENGTH
27           RSUB
28 INPUT     BYTE   X'F3'
29           END    FIRST
```

- ✓ **Starting address = 0?**
- ✓ It is often desirable to have **more than one program** at a time sharing the memory and other resources of the machine.
- ✓ Should assembler decide the location of these programs in memory?
- ✓ We do not know in advance that exactly which programs will execute concurrently
- ✓ **Desirable:** loader should have the liberty to load the program into the memory wherever there is a room for it.
- ✓ Actual starting address of the program is not known until load time.

Assembly of SIC/XE instructions

```
    COPY     START  0
0   FIRST    STL    RETADR
3            LDB    #LENGTH
             BASE   LENGTH
6   CLOOP    +JSUB  RDREC
a            LDA    LENGTH
d            COMP   #0
10           JEQ    ENDFIL
13           J      CLOOP
16  ENDFIL   J      @RETADR
19  RETADR   RESW   1
1c  LENGTH   RESW   1
1f  BUFFER   RESB   4096
             :

101f RDREC   CLEAR  X
1021         CLEAR  A
1023         CLEAR  S
1025         +LDT   #4096
1029 RLOOP   TD     INPUT
102c         JEQ    RLOOP
102f         RD     INPUT
1032         COMPR  A,S
1034         JEQ    EXIT
1037         STCH   BUFFER,X
103a         TIXR   T
103c         JLT    RLOOP
103f EXIT    STX    LENGTH
1042         RSUB
1045 INPUT   BYTE   X'F1'
1046         END    FIRST
```

Assembly of instructions

```
0000        STL         RETADR        172016
```

```
    COPY      START  0
0   FIRST     STL    RETADR
3             LDB    #LENGTH
              BASE   LENGTH
6   CLOOP     +JSUB  RDREC
a             LDA    LENGTH
d             COMP   #0
10            JEQ    ENDFIL
13            J      CLOOP
16  ENDFIL    J      @RETADR
19  RETADR    RESW   1
1c  LENGTH    RESW   1
1f  BUFFER    RESB   4096
              :

101f  RDREC   CLEAR  X
1021          CLEAR  A
1023          CLEAR  S
1025          +LDT   #4096
1029  RLOOP    TD     INPUT
102c          JEQ    RLOOP
102f          RD     INPUT
1032          COMPR  A,S
1034          JEQ    EXIT
1037          STCH   BUFFER,X
103a          TIXR   T
103c          JLT    RLOOP
103f  EXIT    STX    LENGTH
1042          RSUB
1045  INPUT   BYTE   X'F1'
1046          END    FIRST
```

Assembly of instructions (Immediate addressing)

```
1025          +LDT          #4096
*0003           LDB          #LENGTH
```

*combines program-counter relative addressing with immediate addressing

75101000
692016

```
    COPY      START  0
0   FIRST     STL    RETADR
3             LDB    #LENGTH
              BASE   LENGTH
6   CLOOP     +JSUB  RDREC
a             LDA    LENGTH
d             COMP   #0
10            JEQ    ENDFIL
13            J      CLOOP
16  ENDFIL    J      @RETADR
19  RETADR    RESW   1
1c  LENGTH    RESW   1
1f  BUFFER    RESB   4096
              :

101f  RDREC   CLEAR  X
1021          CLEAR  A
1023          CLEAR  S
1025          +LDT   #4096
1029  RLOOP    TD     INPUT
102c          JEQ    RLOOP
102f          RD     INPUT
1032          COMPR  A,S
1034          JEQ    EXIT
1037          STCH   BUFFER,X
103a          TIXR   T
103c          JLT    RLOOP
103f  EXIT    STX    LENGTH
1042          RSUB
1045  INPUT   BYTE   X'F1'
1046          END    FIRST
```

Indirect addressing

Example.

```
16    J          @RETADR
```

3E2000

Register-Register

```
COMPR         A, S      // A is 0 and S is 4
```

A004