CC410: System Programming

Dr. Manal Helal – Fall 2014 – Lecture 4 - Assembler 1

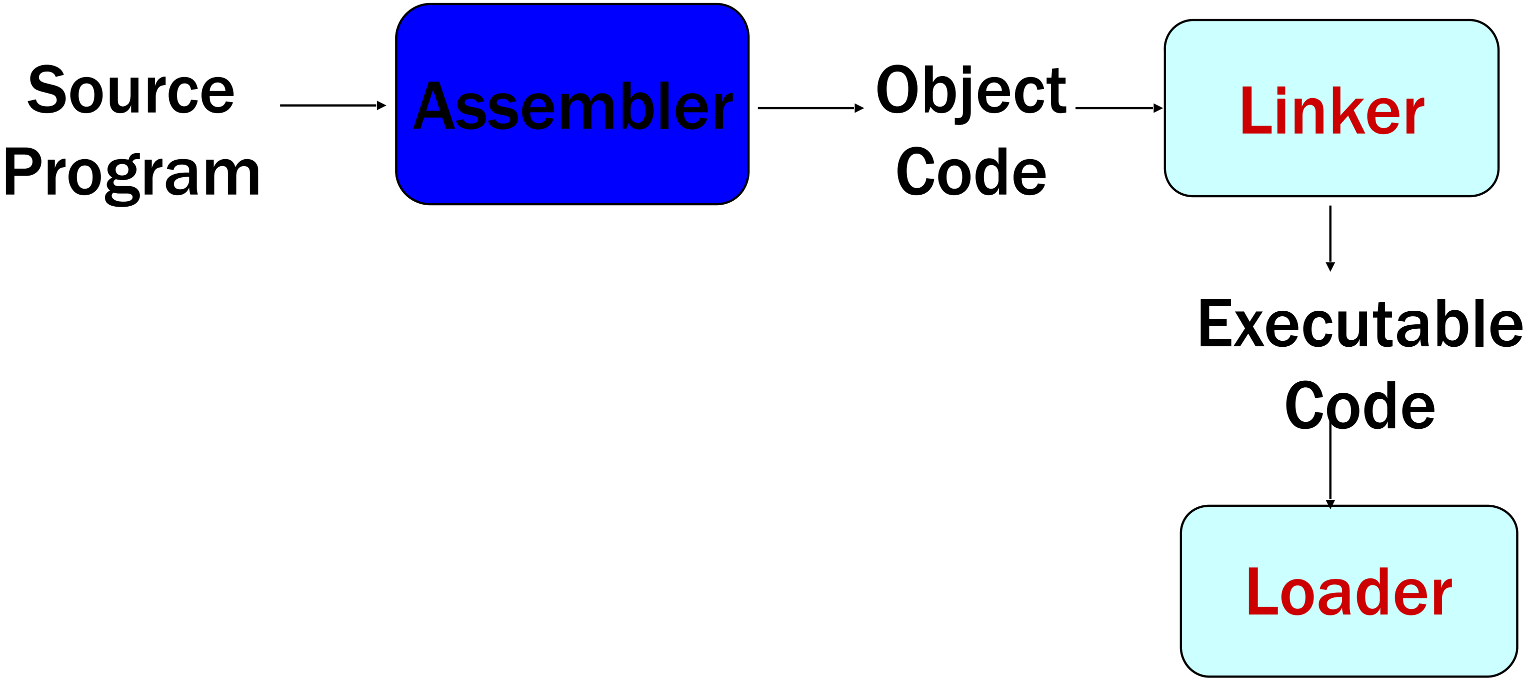
Learning Objectives

- Understand Assemblers functions
- Differentiate machine dependant vs machine independent features

Assembler

Definition: An Assembler is a Program that has the following tasks:

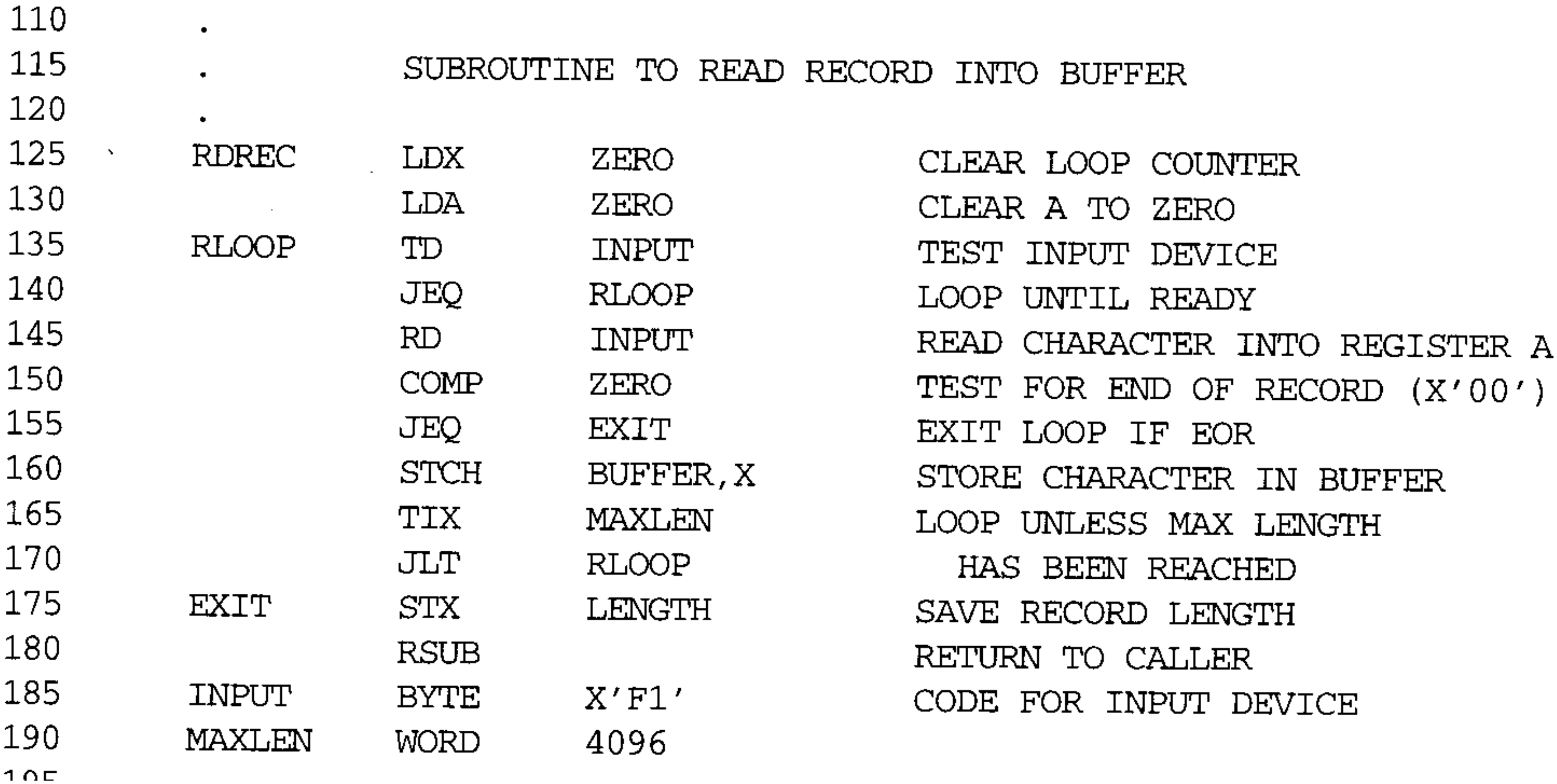
1. Transform assembly instructions (source code, such as MOV) into machine code (binary, such as 100010)
2. Assign memory addresses to symbolic labels
3. Create an object code file



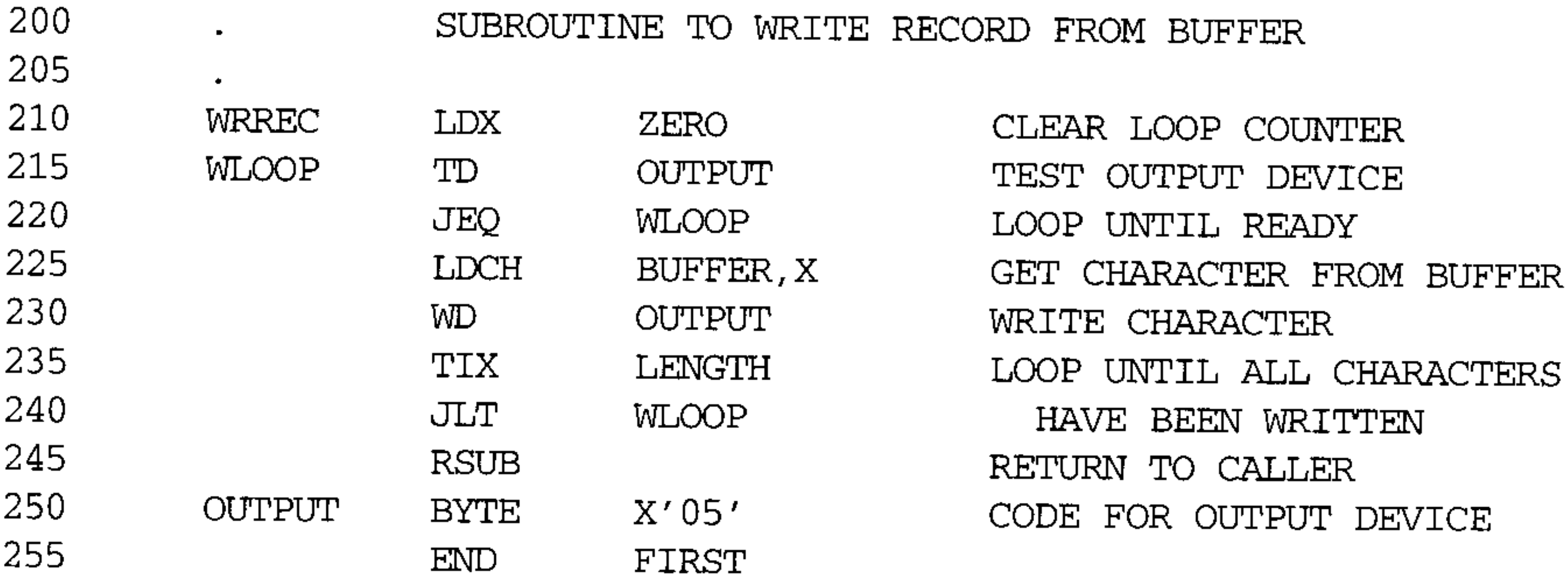
2.1 Basic Assembler Functions

- Figure 2.1 shows an assembler language program for SIC.
  - The line numbers are for reference only.
  - Indexing addressing is indicated by adding the modifier “X”
  - Lines beginning with “.” contain comments only.
  - Reads records from input device (code F1)
  - Copies them to output device (code 05)
  - At the end of the file, writes EOF on the output device, then RSUB to the operating system

```
Line      Source statement

  5     COPY      START   1000       COPY FILE FROM INPUT TO OUTPUT
 10     FIRST     STL     RETADR     SAVE RETURN ADDRESS
 15     CLOOP     JSUB    RDREC      READ INPUT RECORD
 20               LDA     LENGTH     TEST FOR EOF (LENGTH = 0)
 25               COMP    ZERO
 30               JEQ     ENDFIL     EXIT IF EOF FOUND
 35               JSUB    WRREC      WRITE OUTPUT RECORD
 40               J       CLOOP      LOOP
 45     ENDFIL    LDA     EOF        INSERT END OF FILE MARKER
 50               STA     BUFFER
 55               LDA     THREE      SET LENGTH = 3
 60               STA     LENGTH
 65               JSUB    WRREC      WRITE EOF
 70               LDL     RETADR     GET RETURN ADDRESS
 75               RSUB               RETURN TO CALLER
 80     EOF       BYTE    C'EOF'
 85     THREE     WORD    3
 90     ZERO      WORD    0
 95     RETADR    RESW    1
100     LENGTH    RESW    1          LENGTH OF RECORD
105     BUFFER    RESB    4096       4096-BYTE BUFFER AREA
110
```



```
110        .
115        .           SUBROUTINE TO READ RECORD INTO BUFFER
120        .
125        RDREC       LDX       ZERO               CLEAR LOOP COUNTER
130                    LDA       ZERO               CLEAR A TO ZERO
135        RLOOP       TD        INPUT              TEST INPUT DEVICE
140                    JEQ       RLOOP              LOOP UNTIL READY
145                    RD        INPUT              READ CHARACTER INTO REGISTER A
150                    COMP      ZERO               TEST FOR END OF RECORD (X'00')
155                    JEQ       EXIT               EXIT LOOP IF EOR
160                    STCH      BUFFER,X           STORE CHARACTER IN BUFFER
165                    TIX       MAXLEN             LOOP UNLESS MAX LENGTH
170                    JLT       RLOOP                HAS BEEN REACHED
175        EXIT        STX       LENGTH             SAVE RECORD LENGTH
180                    RSUB                         RETURN TO CALLER
185        INPUT       BYTE      X'F1'              CODE FOR INPUT DEVICE
190        MAXLEN      WORD      4096
195
```



```
200     .          SUBROUTINE TO WRITE RECORD FROM BUFFER
205     .
210     WRREC      LDX       ZERO             CLEAR LOOP COUNTER
215     WLOOP      TD        OUTPUT           TEST OUTPUT DEVICE
220                JEQ       WLOOP            LOOP UNTIL READY
225                LDCH      BUFFER,X         GET CHARACTER FROM BUFFER
230                WD        OUTPUT           WRITE CHARACTER
235                TIX       LENGTH           LOOP UNTIL ALL CHARACTERS
240                JLT       WLOOP              HAVE BEEN WRITTEN
245                RSUB                       RETURN TO CALLER
250     OUTPUT     BYTE      X'05'            CODE FOR OUTPUT DEVICE
255                END       FIRST
```

Figure 2.1 Example of a SIC assembler language program.

2.1 Basic Assembler Functions

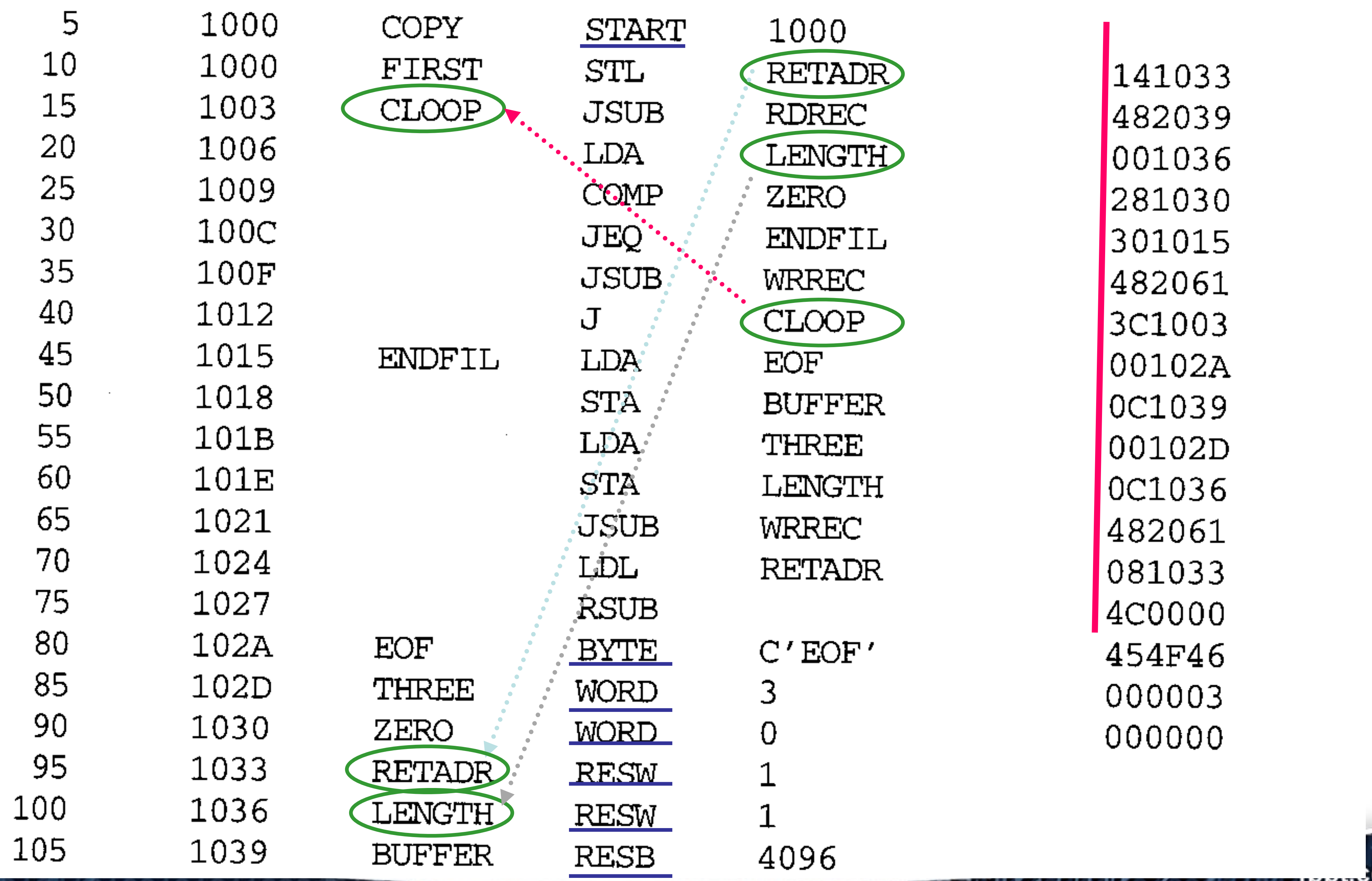
- Assembler *directives* (pseudo-instructions)
  - START, END, BYTE, WORD, RESB, RESW.
  - These statements are not translated into machine instructions.
  - Instead, they provide instructions to the assembler itself.

2.1 Basic Assembler Functions

- Data transfer (RD, WD)
  - A buffer is used to store record
  - Buffering is necessary for different I/O rates
  - The end of each record is marked with a null character (00_{16})
  - Buffer length is 4096 Bytes
  - The end of the file is indicated by a zero-length record
- Subroutines (JSUB, RSUB)
  - RDREC, WRREC
  - Save link (L) register first before nested jump

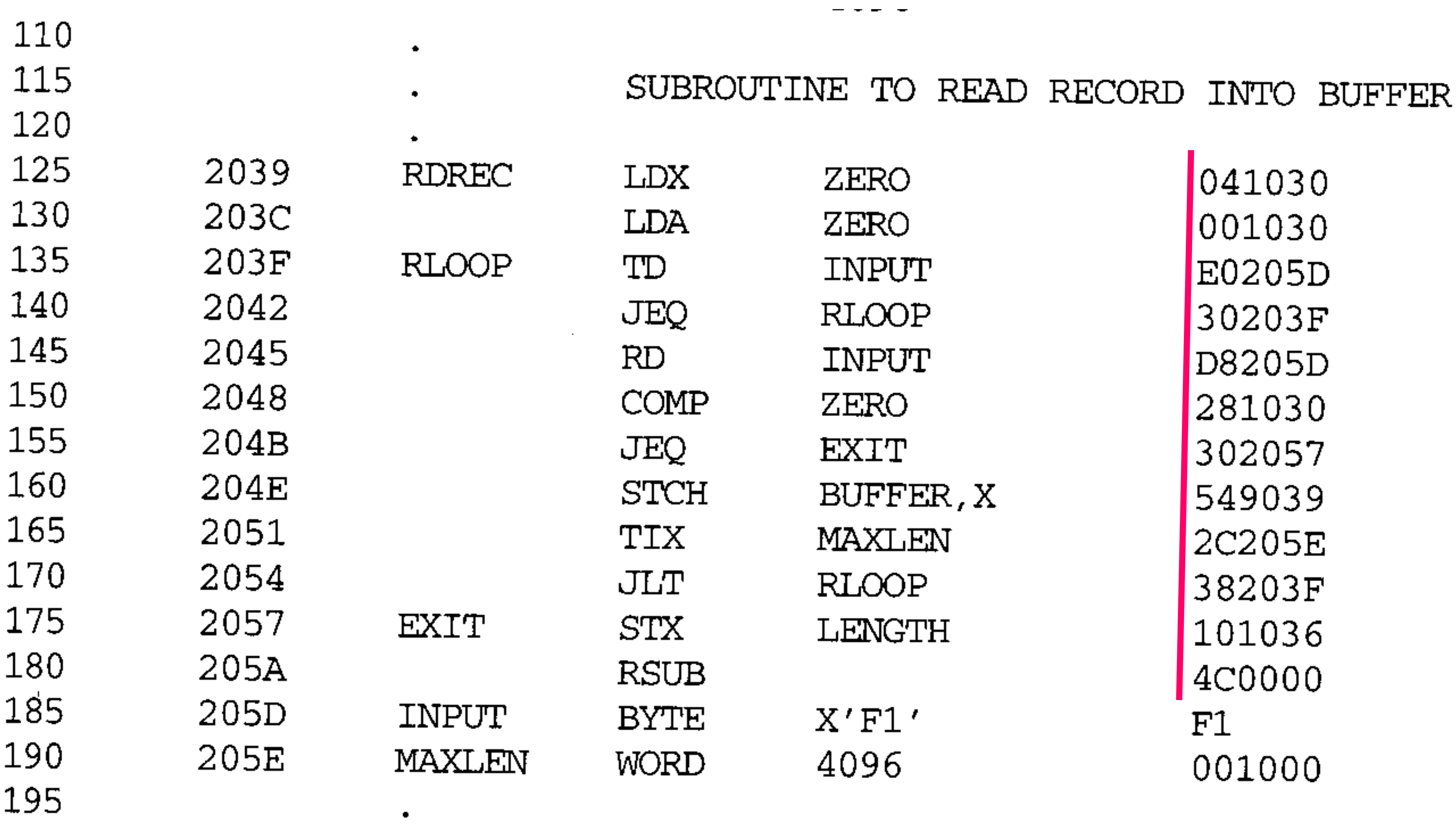
2.1.1 A simple SIC Assembler

- Figure 2.2 shows the generated object code for each statement.
  - Loc gives the machine address in Hex.
  - Assume the program starting at address 1000.
- Translation functions
  - Translate STL to 14.
  - Translate RETADR to 1033.
  - Build the machine instructions in the proper format (,X).
  - Translate EOF to 454F46.
  - Write the object program and assembly listing.



Line	Loc	Source statement			Object code
5	1000	COPY	START	1000	
10	1000	FIRST	STL	RETADR	141033
15	1003	CLOOP	JSUB	RDREC	482039
20	1006		LDA	LENGTH	001036
25	1009		COMP	ZERO	281030
30	100C		JEQ	ENDFIL	301015
35	100F		JSUB	WRREC	482061
40	1012		J	CLOOP	3C1003
45	1015	ENDFIL	LDA	EOF	00102A
50	1018		STA	BUFFER	0C1039
55	101B		LDA	THREE	00102D
60	101E		STA	LENGTH	0C1036
65	1021		JSUB	WRREC	482061
70	1024		LDL	RETADR	081033
75	1027		RSUB		4C0000
80	102A	EOF	BYTE	C'EOF'	454F46
85	102D	THREE	WORD	3	000003
90	1030	ZERO	WORD	0	000000
95	1033	RETADR	RESW	1	
100	1036	LENGTH	RESW	1	
105	1039	BUFFER	RESB	4096	

```
Line      Loc       Source statement                                   Object code

110                 .
115                 .          SUBROUTINE TO READ RECORD INTO BUFFER
120                 .
125       2039      RDREC      LDX        ZERO                         041030
130       203C                 LDA        ZERO                         001030
135       203F      RLOOP      TD         INPUT                        E0205D
140       2042                 JEQ        RLOOP                        30203F
145       2045                 RD         INPUT                        D8205D
150       2048                 COMP       ZERO                         281030
155       204B                 JEQ        EXIT                         302057
160       204E                 STCH       BUFFER,X                     549039
165       2051                 TIX        MAXLEN                       2C205E
170       2054                 JLT        RLOOP                        38203F
175       2057      EXIT       STX        LENGTH                       101036
180       205A                 RSUB                                    4C0000
185       205D      INPUT      BYTE       X'F1'                        F1
190       205E      MAXLEN     WORD       4096                         001000
195                 .
```



```
Line    Loc     Source statement                          Object code

200             .        SUBROUTINE TO WRITE RECORD FROM BUFFER
205             .
210     2061    WRREC    LDX     ZERO                     041030
215     2064    WLOOP    TD      OUTPUT                   E02079
220     2067             JEQ     WLOOP                    302064
225     206A             LDCH    BUFFER,X                 509039
230     206D             WD      OUTPUT                   DC2079
235     2070             TIX     LENGTH                   2C1036
240     2073             JLT     WLOOP                    382064
245     2076             RSUB                             4C0000
250     2079    OUTPUT   BYTE    X'05'                    05
255                      END     FIRST
```

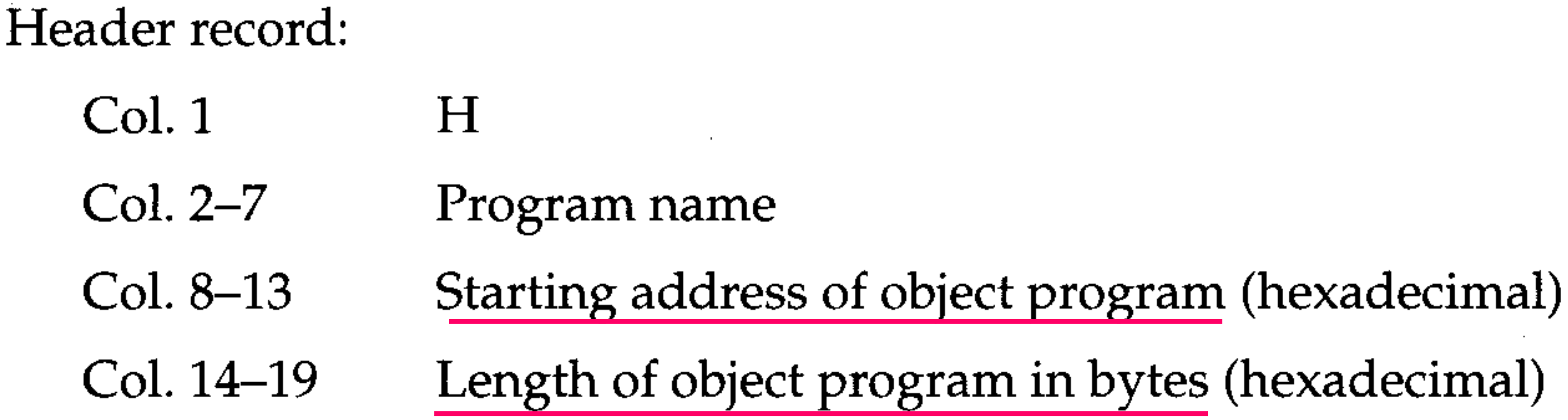
Figure 2.2 Program from Fig. 2.1 with object code.

2.1.1 A simple SIC Assembler

- A forward reference
  - 10 1000 FIRST STL RETADR 141033
  - A reference to a label (RETADR) that is defined later in the program
- Most assemblers make two passes over source program.
  - Pass 1 scans the source for label definitions and assigns address (Loc).
  - Pass 2 performs most of the actual translation.

2.1.1 A simple SIC Assembler

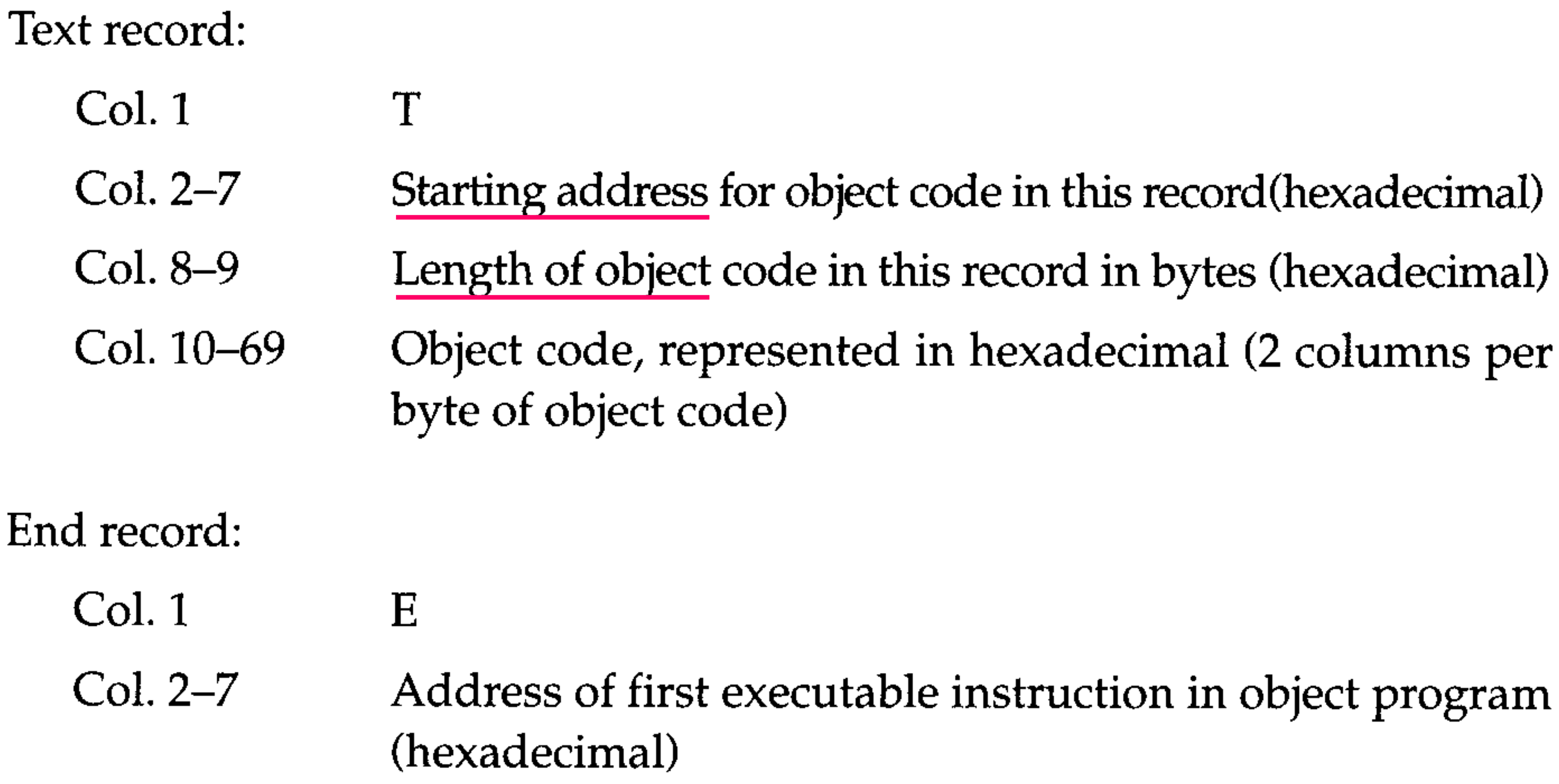
- The object program (OP) will be loaded into memory for execution.
- Three types of records
  - Header: program name, starting address, length.
  - Text: starting address, length, object code.
  - End: address of first executable instruction.



Header record:

Col. 1	H
Col. 2–7	Program name
Col. 8–13	Starting address of object program (hexadecimal)
Col. 14–19	Length of object program in bytes (hexadecimal)

2.1.1 A simple SIC Assembler



Text record:

Col. 1 T

Col. 2–7 Starting address for object code in this record(hexadecimal)

Col. 8–9 Length of object code in this record in bytes (hexadecimal)

Col. 10–69 Object code, represented in hexadecimal (2 columns per byte of object code)

End record:

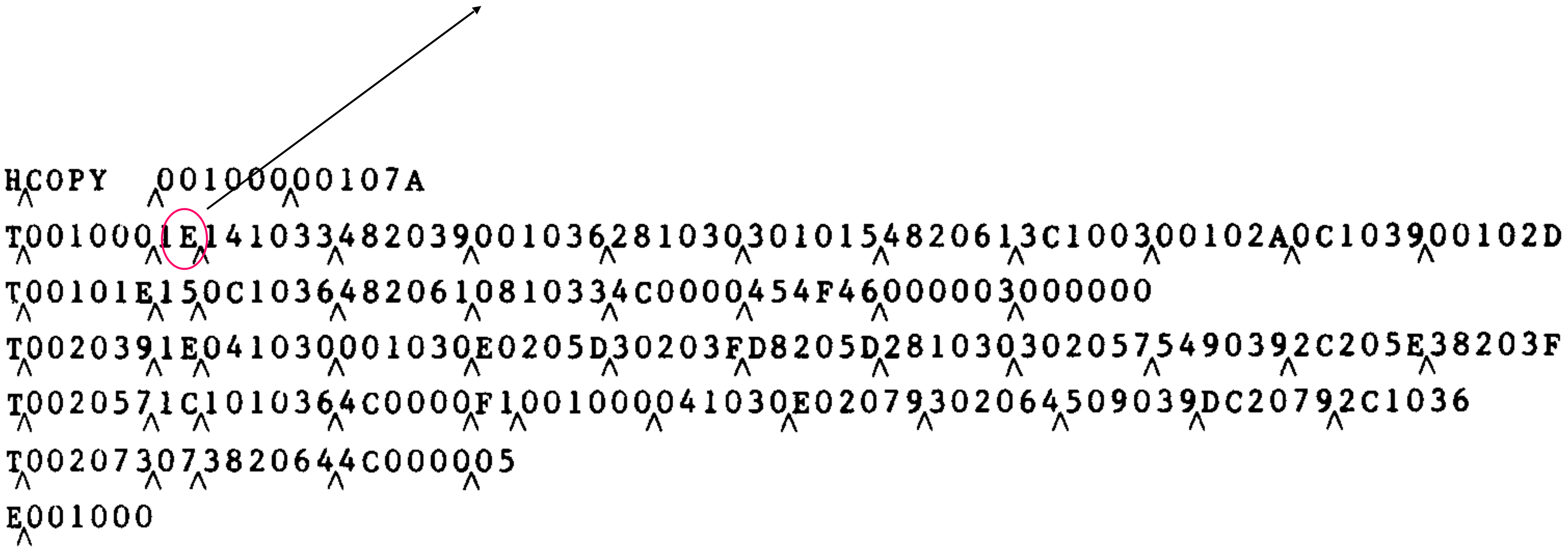
Col. 1 E

Col. 2–7 Address of first executable instruction in object program (hexadecimal)

2.1.1 A simple SIC Assembler

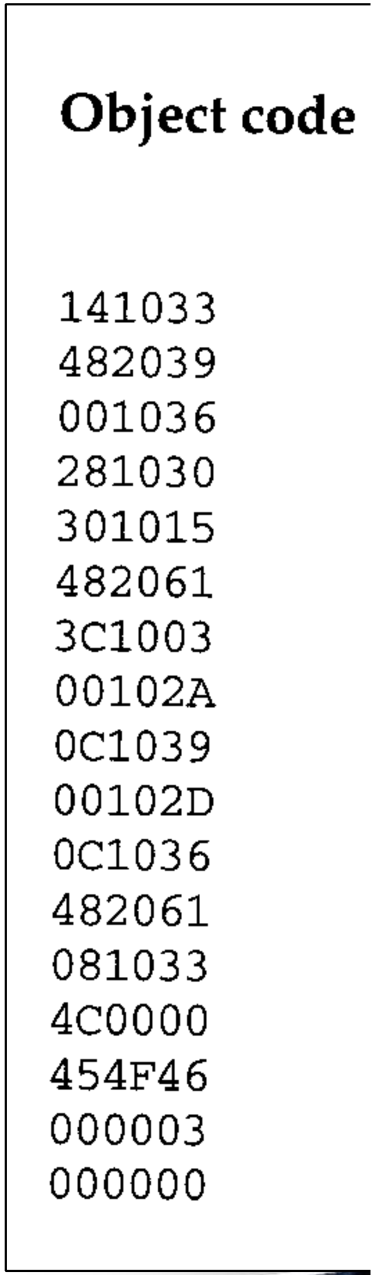
- The symbol ^ is used to separate fields.
  - Figure 2.3

1E(H)=30(D)=16(D)+14(D)



```
H^COPY  ^001000^00107A
T^001000^1E^141033^482039^001036^281030^301015^482061^3C1003^00102A^0C1039^00102D
T^00101E^15^0C1036^482061^081033^4C0000^454F46^000003^000000
T^002039^1E^041030^001030^E0205D^30203F^D8205D^281030^302057^549039^2C205E^38203F
T^002057^1C^101036^4C0000^F1^001000^041030^E02079^302064^509039^DC2079^2C1036
T^002073^07^382064^4C0000^05
E^001000
```

Figure 2.3 Object program corresponding to Fig. 2.2.



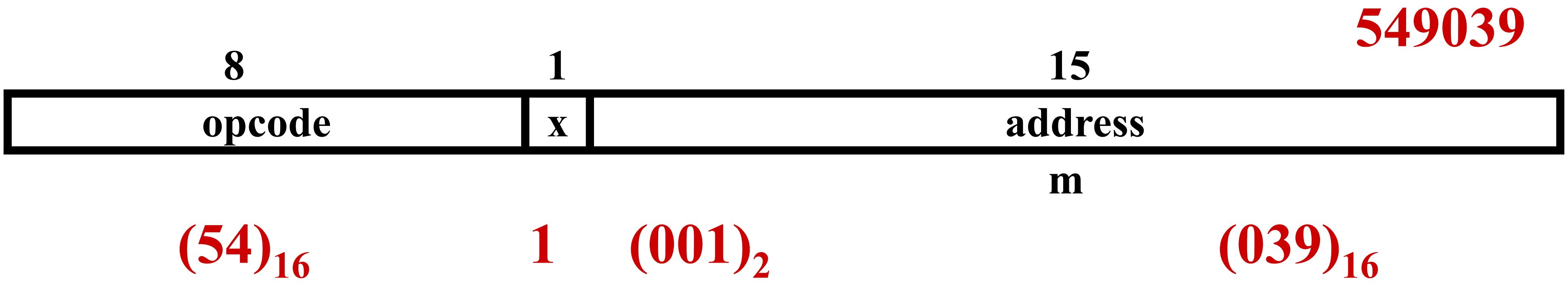
```
Object code

141033
482039
001036
281030
301015
482061
3C1003
00102A
0C1039
00102D
0C1036
482061
081033
4C0000
454F46
000003
000000
```

2.1.1 A simple SIC Assembler

- Assembler's Functions
  - Convert mnemonic operation codes to their machine language equivalents
    - STL to 14
  - Convert symbolic operands (referred label)_to their equivalent machine addresses
    - RETADR to 1033
  - Build the machine instructions in the proper format
  - Convert the data constants to internal machine representations
  - Write the object program and the assembly listing

2.1.1 A simple SIC Assembler

- Example of Instruction Assemble
  - Forward reference
  - STCH BUFFER, X

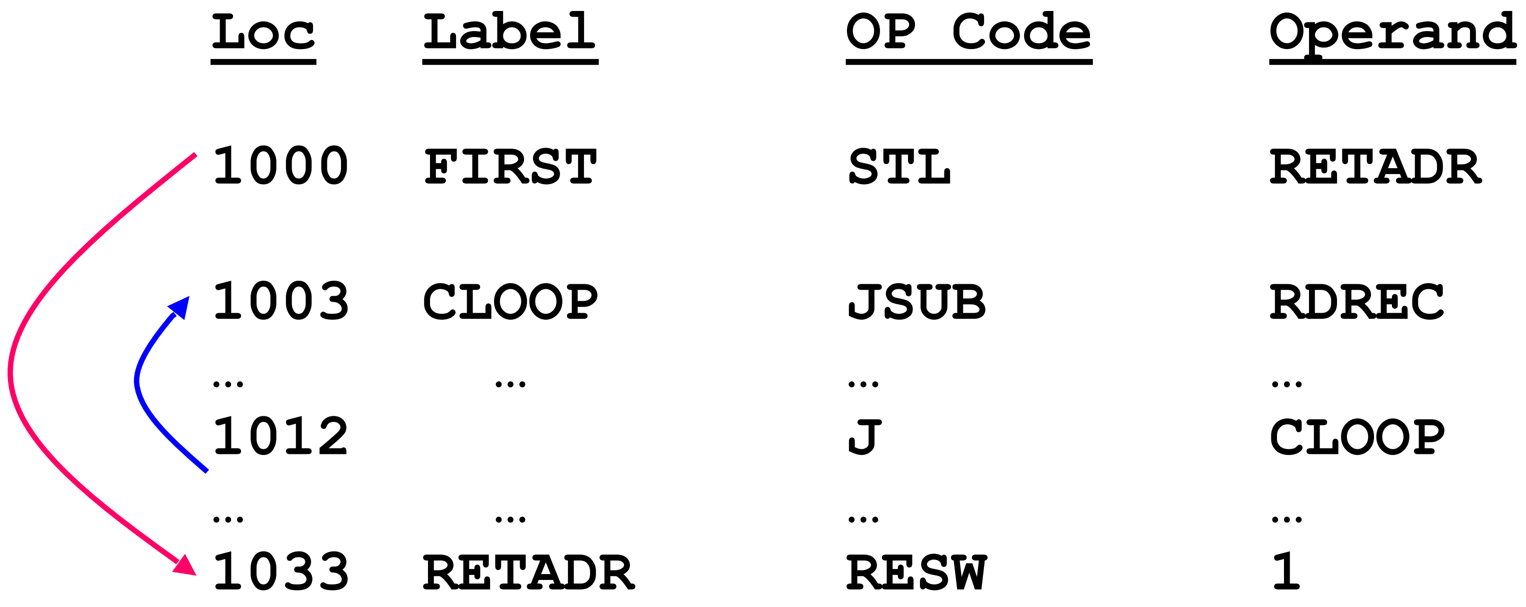


549039

8	1	15
opcode	x	address
		m
$(54)_{16}$	1	$(001)_2$ $(039)_{16}$

2.1.1 A simple SIC Assembler

- Forward reference
  - Reference to a label that is defined later in the program.



Loc	Label	OP Code	Operand
1000	FIRST	STL	RETADR
1003	CLOOP	JSUB	RDREC
…	…	…	…
1012		J	CLOOP
…	…	…	…
1033	RETADR	RESW	1

2.1.1 A simple SIC Assembler

- The functions of the two passes assembler.
- Pass 1 (define symbol)
  - Assign addresses to all statements (generate LOC).
  - Save the values (address) assigned to all labels for Pass 2.
  - Perform some processing of assembler directives.
- Pass 2
  - Assemble instructions.
  - Generate data values defined by BYTE, WORD.
  - Perform processing of assembler directives not done during Pass 1.
  - Write the OP (Fig. 2.3) and the assembly listing (Fig. 2.2).