Lab 2

Assembly

Name: Abdelrahman Mahmoud Kamal Mahmoud Nour

ID: 39

First Program

A)

Write a program that searches for a certain byte in a string, if found, put the address of the byte in register A else, put 0xFFFFFF

Code:

c: > FPC > prog > srcfile.txt

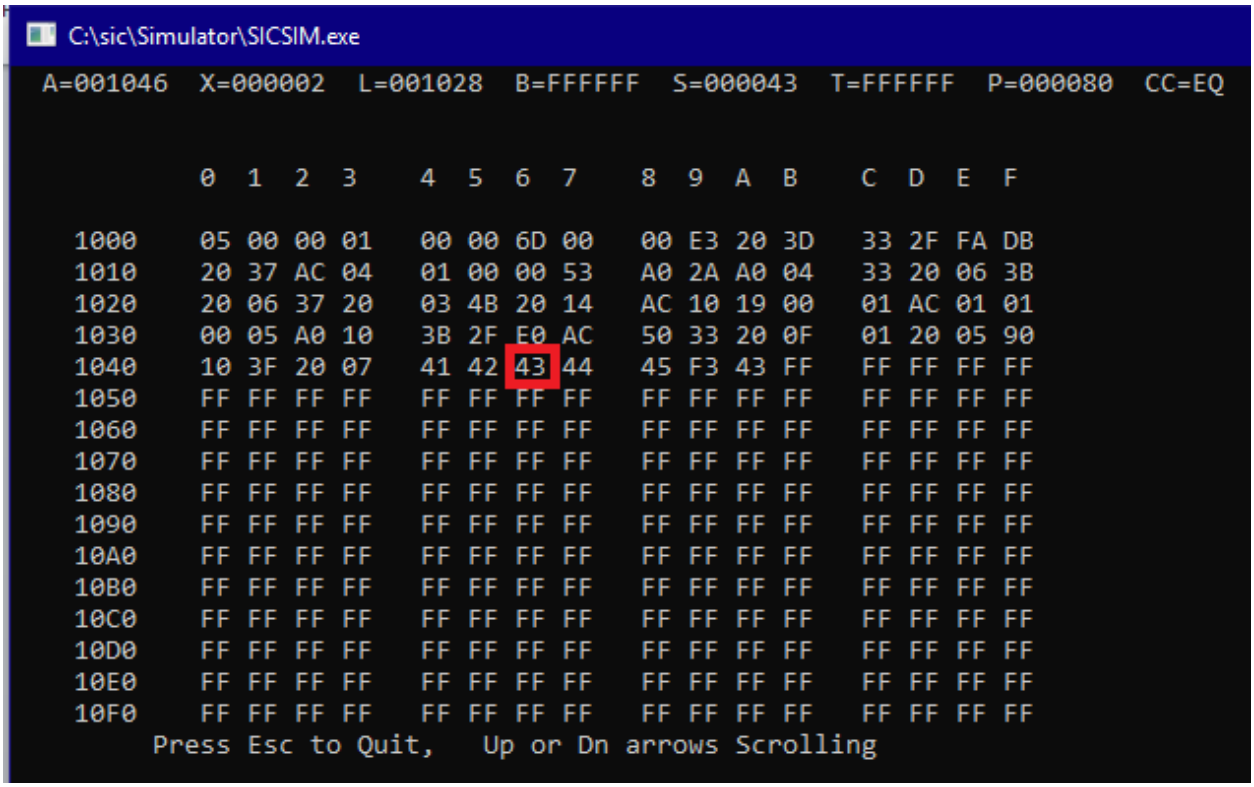
```
1   .2345678901234567890
2   PROB1   START   1000
3           LDX     #0
4           LDA     #0
5           LDS     #0
6   KHAL    TD      INDEV
7           JEQ     KHAL
8           RD      INDEV
9           RMO     A,S
10          LDA     #0
11  LOOP    LDCH    STR,X
12          COMPR   A,S
13          JEQ     DO
14          JLT     DOF
15          JGT     DOF
16  DO      JSUB    DOE
17  DOF     RMO     X,A
18          ADD     #1
19          RMO     A,X
20          LDA     #5
21          COMPR   X,A
22          JLT     LOOP
23          RMO     T,A
24          JEQ     EXIT
25  DOE     LDA     #STR
26          ADDR    X,A
27          J       EXIT
28  STR     BYTE    C'ABCDEF'
29  INDEV   BYTE    X'F3'
30  EXIT    END     PROB1
```

Assumptions:

1- The string is known.
2- The size of the string in known.
3- The character is read from device F3.

Sample Runs:

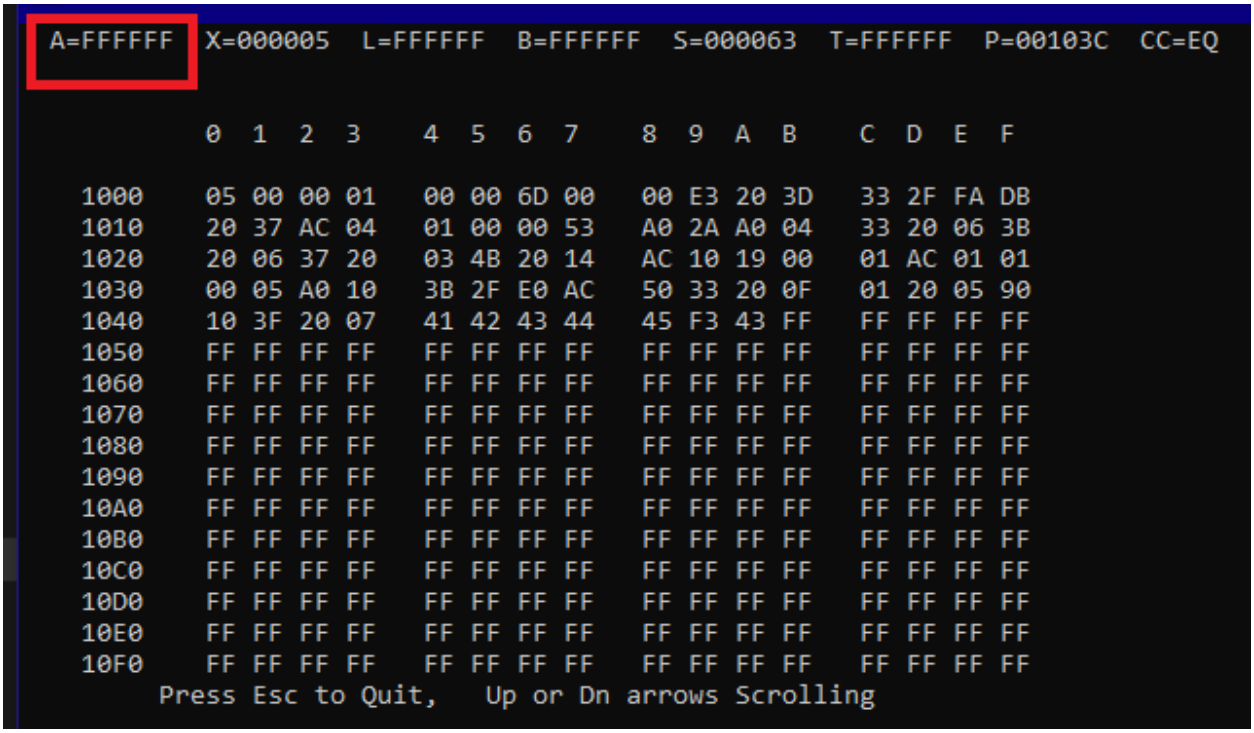
1-Search for character ‘C’ in ’ABCDE’



The desired character ‘C’ address is “001046” in the Accumulator which is written as “43” in hexadecimal.

2-Search for ‘c’ in ‘ABCDE’ which does not exist

Desired output in the Accumulator: FFFFFF



Program 2

B) Read a string from an input device and print it reversed to an output device.

Code:

c: > FPC > prog > srcfile.txt

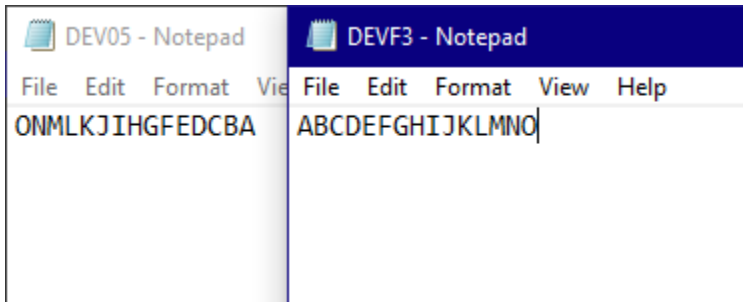
```
2   PROB1    START   1000
3            LDX     #0
4            LDA     #0
5   LOOP1    LDS     #4
6   AGAIN1   TD      INDEV
7            JEQ     AGAIN1
8            RD      INDEV
9            COMPR   A,S
10           JGT     ZWD
11           JEQ     MID
12  ZWD      STCH    ARR,X
13           RMO     X,A
14           ADD     #1
15           RMO     A,X
16           JGT     LOOP1
17  MID      RMO     X,A
18           SUB     #1
19           RMO     A,X
20           J       KHAL
21  KHAL     TD      OUTDEV
22           JEQ     KHAL
23           LDCH    ARR,X
24           WD      OUTDEV
25           RMO     X,A
26           SUB     #1
27           RMO     A,X
28           COMP    #0
29           JGT     KHAL
30           JEQ     KHAL
31  INDEV    BYTE    X'F3'
32  OUTDEV   BYTE    X'05'
33  ARR      RESB    10
34           END     PROB1
```

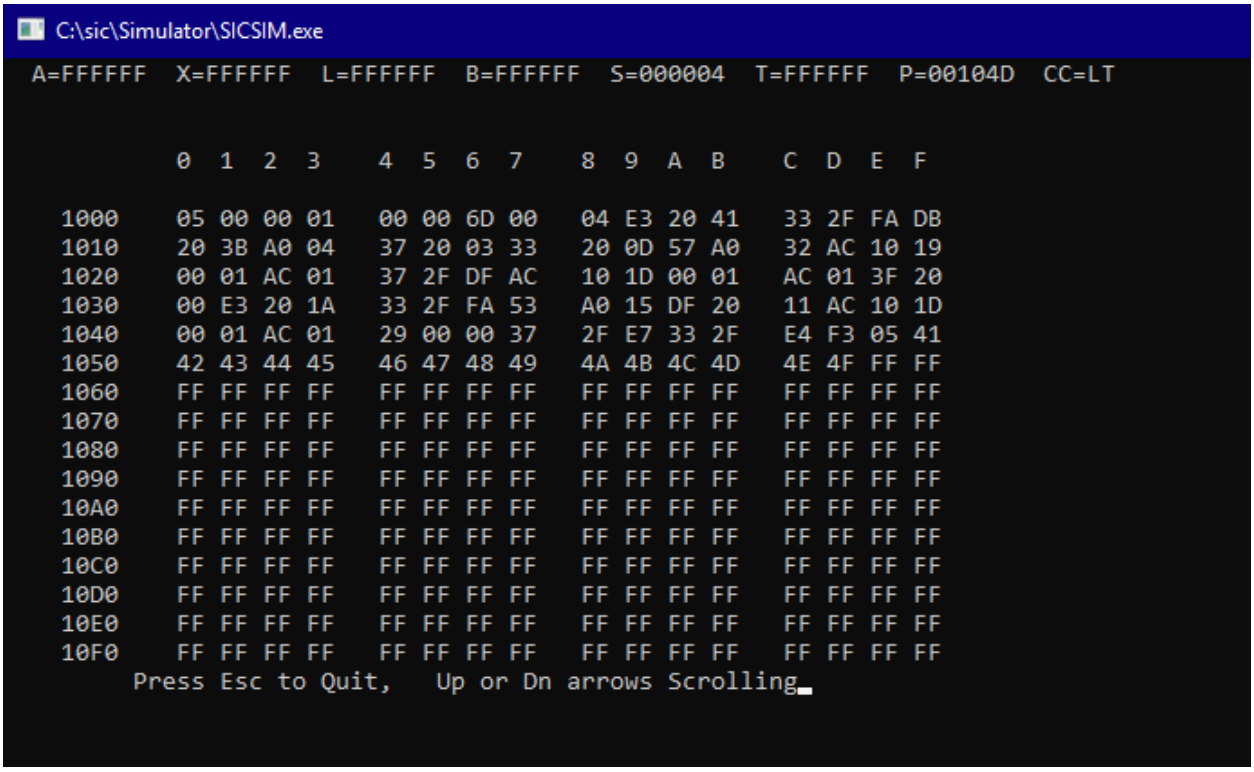
Assumptions:

1- The input is printed reversed to the output, but it is not reversed in the memory.

Sample Runs:

1- Reverse “ABCDEFGHIJ KLMNO”





When printing the M character which is the 12th character and has index C in hexadecimal.

Program 3

C. Read a string from an input device and print the string to an output device after converting it to UPPER case.
(You must use at least two subroutines in this problem .One for converting a letter to an upper case. Another one for writing one character to an output device)

Code:

c: > FPC > prog > srcfile.txt

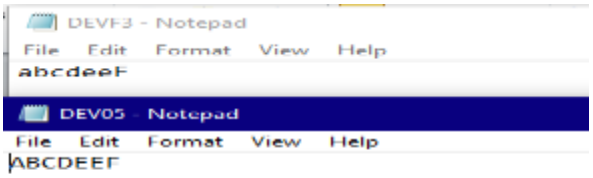
```
1   .234567890123456789
2   PROB1    START   1000
3            LDA     #0
4   LOOP1    LDS     #4
5   AGAIN1   TD      INDEV
6            JEQ     AGAIN1
7            RD      INDEV
8            COMPR   A,S
9            JEQ     EXIT
10           JLT     EXIT
11           JSUB    CONV
12           JSUB    WRITE
13           J       AGAIN1
14  CONV     COMP    #90
15           JGT     DO
16           RSUB
17  DO       SUB     #32
18           RSUB
19  WRITE    TD      OUTDEV
20           JEQ     WRITE
21           WD      OUTDEV
22           RSUB
23  INDEV    BYTE    X'F3'
24  OUTDEV   BYTE    X'05'
25  EXIT     END     PROB1
```

Assumptions:

No specific assumption

Sample Runs:

1- Print “abcdeeF” in upper case .




```
A=000004  X=FFFFFF  L=00101D  B=FFFFFF  S=000004  T=FFFFFF  P=001014  CC=EQ

          0  1  2  3    4  5  6  7    8  9  A  B    C  D  E  F

  1000    01 00 00 6D   00 04 E3 20   32 33 2F FA   DB 20 2C A0
  1010    04 33 20 29   3B 20 26 4B   20 06 4B 20   12 3F 2F E6
  1020    29 00 5A 37   20 03 4F 00   00 1D 00 20   4F 00 00 E3
  1030    20 0A 33 2F   FA DF 20 04   4F 00 00 F3   05 FF FF FF
  1040    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  1050    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  1060    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  1070    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  1080    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  1090    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10A0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10B0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10C0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10D0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10E0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
  10F0    FF FF FF FF   FF FF FF FF   FF FF FF FF   FF FF FF FF
       Press Esc to Quit,   Up or Dn arrows Scrolling
```

When reading “a” character and then printing it after converting it to upper case. (from 61 to 41 in hexadecimal)

```
            S t a t e   A f t e r   F e t c h   C y c l e

                    Contents of Registers
     A        x        L        B        S        T        PC       CC
  000061   FFFFFF   FFFFFF   FFFFFF   000004   FFFFFF   001011     LT


                       I n s t r u c t i o n
        location         instruction      format        mnemonic
         00100F             A004            2            compr


               Operands are :
               R 1        R 2
               0          4
               A          S


          Press any key to execute , Esc to continue
```

```
            S t a t e   A f t e r   F e t c h   C y c l e

                    Contents of Registers
     A        x        L        B        S        T        PC       CC
  000041   FFFFFF   00101D   FFFFFF   000004   FFFFFF   001038     LT


                       I n s t r u c t i o n
        location         instruction      format        mnemonic
         001035             DF2004          3             wd


                       O p e r a n d
  indxd      Calculation of Target Address        TA           mode
    no                adrs + (PC)                 01660       direct


         Press any key to execute , Esc to continue
```

Program 4

D. Implement the bubble sort to sort characters in a string.

Code:

c: > FPC > prog > srcfile.txt

```
1   .234567890123456789
2   PROB1   START   1000
3           LDS     #0
4           LDX     #0
5   LOOP1   LDS     #1
6           LDA     I
7           ADD     #1
8           STA     I
9           COMP    LAST
10          LDA     #6
11          STA     J
12          JLT     LOOP2
13          JGT     TEST
14          JEQ     TEST
15  LOOP2   LDA     J
16          SUB     #1
17          COMP    I
18          STA     J
19          JLT     LOOP1
20          JEQ     LOOP1
21          LDX     J
22          LDCH    ARR,X
23          RMO     A,T
24          RMO     X,A
25          SUB     #1
26          RMO     A,X
27          LDCH    ARR,X
28          COMPR   A,T
29          RMO     X,A
30          ADD     #1
31          RMO     A,X
32          JGT     DO
33          JLT     LOOP2
34          JEQ     LOOP2
35  DO      LDS     #1
36          JSUB    SWAP
37          J       LOOP2
38          STX     J
39          LDA     J
40          SUB     #1
41          COMP    I
42          JGT     LOOP2
43          JLT     LOOP1
44          JEQ     LOOP1
45  SWAP    LDCH    ARR,X
46          STCH    TEMP
47          RMO     X,A
48          STA     FI
49          SUB     #1
50          STA     SI
51          LDX     SI
52          LDCH    ARR,X
53          LDX     FI
54          STCH    ARR,X
55          LDX     SI
56          LDCH    TEMP
57          STCH    ARR,X
58          RSUB
59  TEST    LDX     #0
60  KHAL    TD      OUTDEV
61          JEQ     KHAL
62          LDCH    ARR,X
63          WD      OUTDEV
64          RMO     X,A
```

```
65          ADD     #1
66          COMP    #5
67          RMO     A,X
68          JGT     EXIT
69          JEQ     KHAL
70          JLT     KHAL
71  TEMP    RESB    1
72  ARR     BYTE    C'azbgjw'
73  LINDX   WORD    5
74  FI      RESW    1
75  SI      RESW    1
76  I       WORD    -1
77  J       WORD    6
78  LAST    WORD    5
79  OUTDEV  BYTE    X'05'
80  EXIT    END     PROB1
```

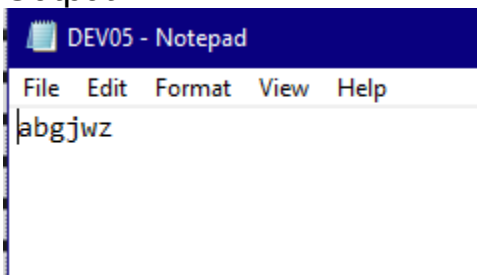
Assumptions:

1- The string and its size is known in advance and is written in the code.

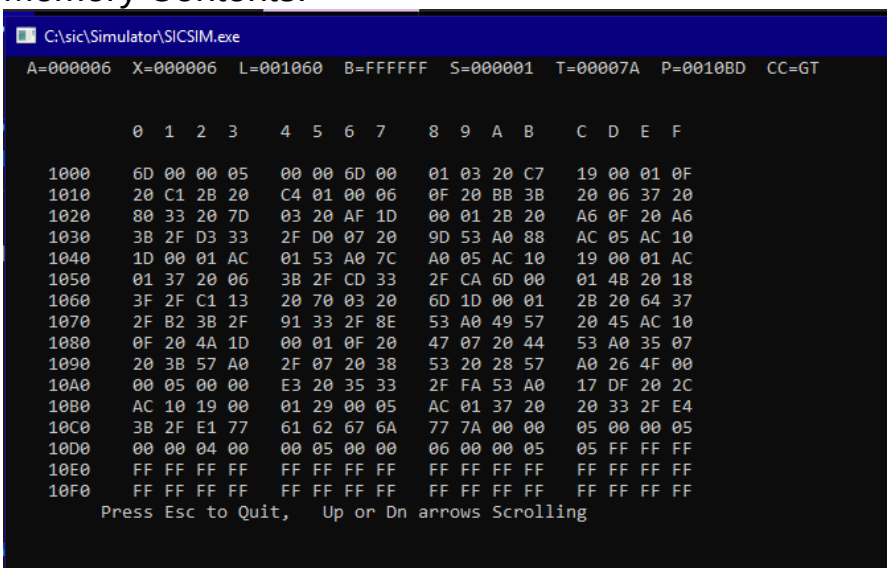
Sample Runs:

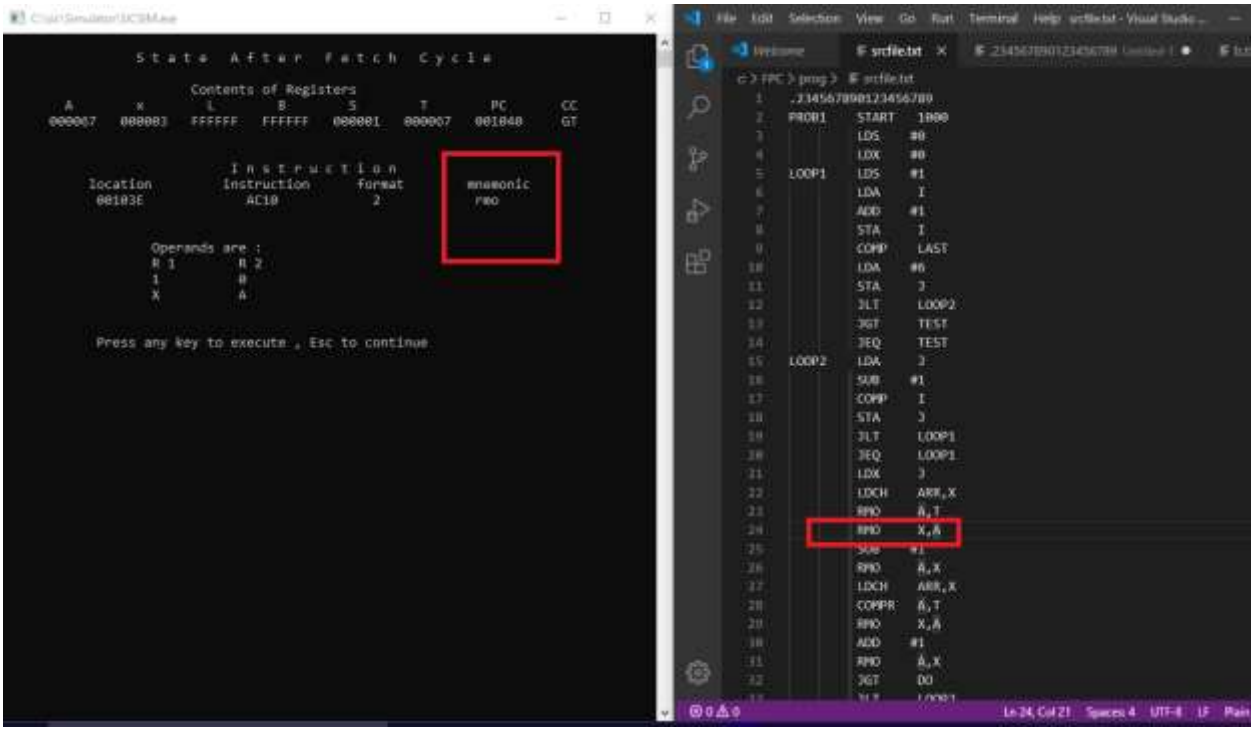
1- Sort “azbgjw”

Output:

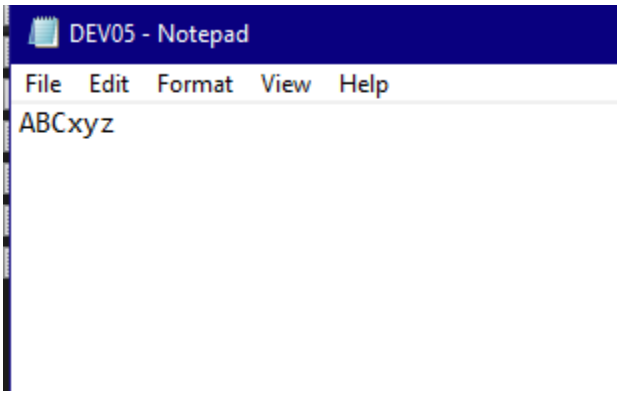


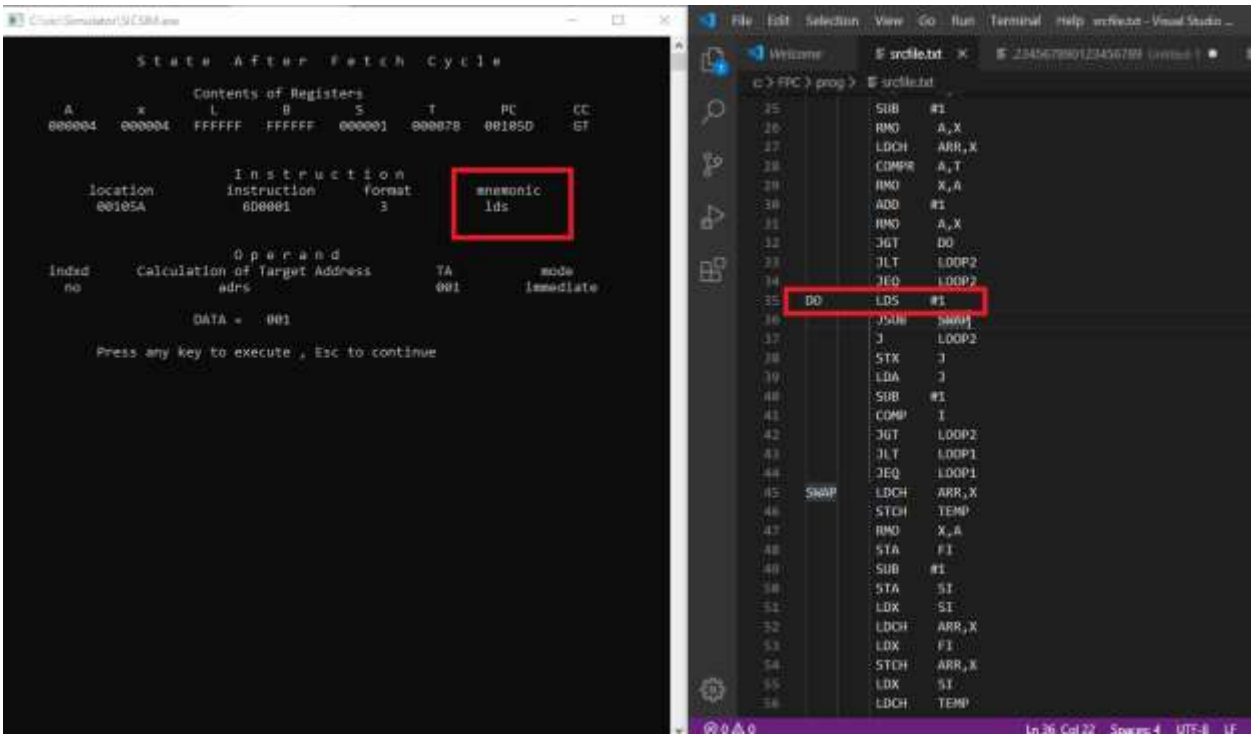
Memory Contents:

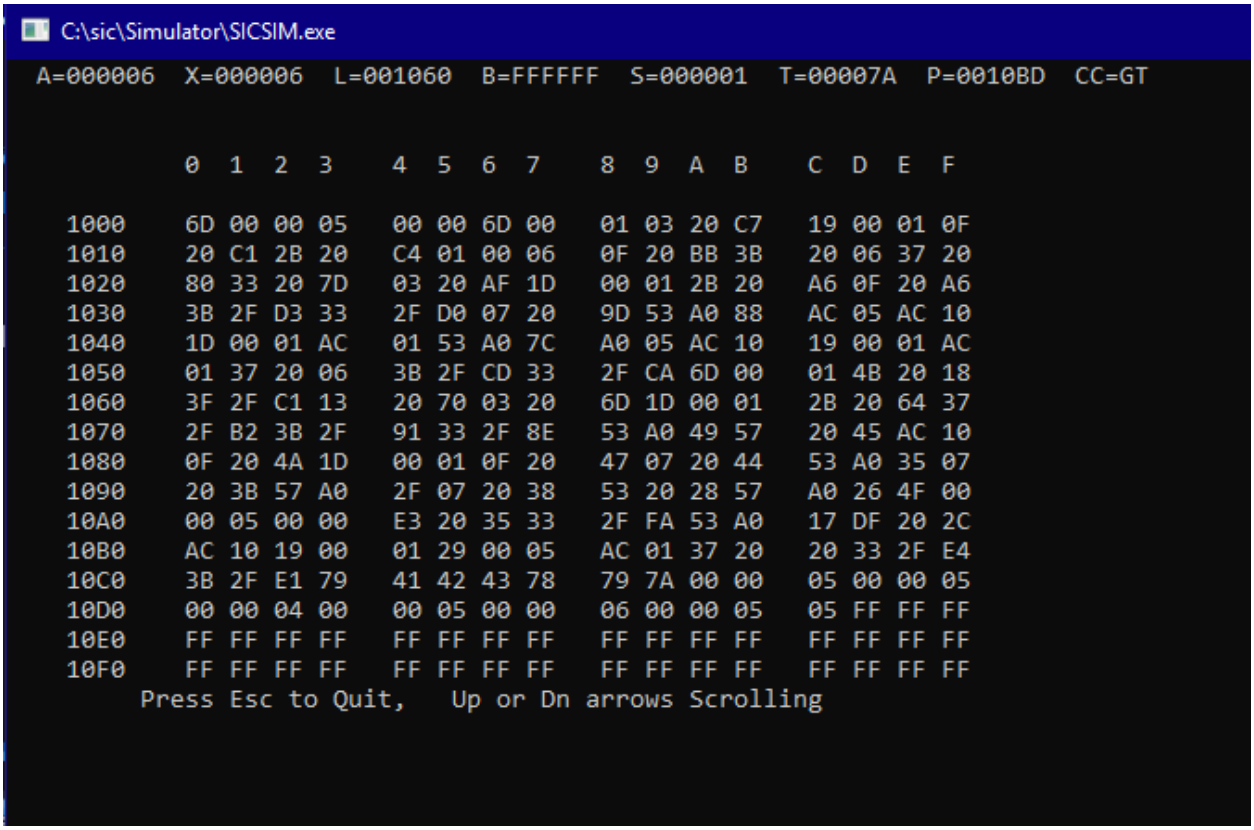




2- Sort “ABCzxy”




State After Fetch Cycle
Contents of Registers
A X L B S T PC CC
000004 000004 FFFFFF FFFFFF 000001 000078 00185D GT
Instruction
location instruction format mnemonic
00185A 6D0001 3 lds
Operand
Indxd Calculation of Target Address TA mode
no adrs 001 immediate
DATA = 001
Press any key to execute , Esc to continue
SUB #1
RMO A,X
LDCH ARR,X
COMPR A,T
RMO X,A
ADD #1
RMO A,X
JGT DO
JLT LOOP2
JEQ LOOP2
DO LDS #1
J LOOP2
STX J
LDA J
SUB #1
COMP I
JGT LOOP2
JLT LOOP1
JEQ LOOP1
SWAP LDCH ARR,X
STCH TEMP
RMO X,A
STA FI
SUB #1
STA SI
LDX SI
LDCH ARR,X
LDX FI
STCH ARR,X
LDX SI
LDCH TEMP


C:\sic\Simulator\SICSIM.exe
A=000006 X=000006 L=001060 B=FFFFFF S=000001 T=00007A P=0010BD CC=GT
0 1 2 3 4 5 6 7 8 9 A B C D E F
1000 6D 00 00 05 00 00 6D 00 01 03 20 C7 19 00 01 0F
1010 20 C1 2B 20 C4 01 00 06 0F 20 BB 3B 20 06 37 20
1020 80 33 20 7D 03 20 AF 1D 00 01 2B 20 A6 0F 20 A6
1030 3B 2F D3 33 2F D0 07 20 9D 53 A0 88 AC 05 AC 10
1040 1D 00 01 AC 01 53 A0 7C A0 05 AC 10 19 00 01 AC
1050 01 37 20 06 3B 2F CD 33 2F CA 6D 00 01 4B 20 18
1060 3F 2F C1 13 20 70 03 20 6D 1D 00 01 2B 20 64 37
1070 2F B2 3B 2F 91 33 2F 8E 53 A0 49 57 20 45 AC 10
1080 0F 20 4A 1D 00 01 0F 20 47 07 20 44 53 A0 35 07
1090 20 3B 57 A0 2F 07 20 38 53 20 28 57 A0 26 4F 00
10A0 00 05 00 00 E3 20 35 33 2F FA 53 A0 17 DF 20 2C
10B0 AC 10 19 00 01 29 00 05 AC 01 37 20 20 33 2F E4
10C0 3B 2F E1 79 41 42 43 78 79 7A 00 00 05 00 00 05
10D0 00 00 04 00 00 05 00 00 06 00 00 05 05 FF FF FF
10E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
Press Esc to Quit, Up or Dn arrows Scrolling