

Name : Khan Mohd. Owais Raza**Registration No. : 20BCD7138****Course : Computer Organization & Architecture****Course Code : ECE2002****Question :**

Write a program for searching the existence of a certain data in a given data array using 8086. When found, add the searched number to the last 4 digits of your Reg no. Display this result in White color text in the middle row, starting of the column of the output screen at row 20, column 10, page no. 0 using interrupts.

Solution –

Algorithm of 8086 program :-

- 1) Move 2000 in AX and assign it to ES
- 2) Assign value 600 to DI
- 3) Move 25 in AI
- 4) Move the contents of CX to BX
- 5) Clear the value of directional flag DF
- 6) Repeat step 7 till ZF is not set
- 7) Scan byte from [DI] and check its difference with contents of AL
- 8) Update value of DI
- 9) Decrease the value of DI by 1
- 10) Subtract value of BX by CX
- 11) Decrease value of BX by 1
- 12) Program halts

1] **8086 program to put numbers in an array** :-

```
01 ; KHAN MOHD OWAIS RAZA 20BCD7138
02 ; ECE2002_Computer Organization & Architecture_Digital Assessment-1
03 ; -----
04 ; 8086 program to put elements in an array
05 ORG 100H
06 MOV SI,1200H
07 MOV CL,10H
08 NEXT:
09 MOV [SI],CL
10 INC SI
11 INC CL
12 CMP SI,1200H
13 JNZ NEXT
14 MOV SI,1200H
15 MOV AL,[SI]
16 INC SI
17 AGAIN:
18 ADD AL,[SI]
19 INC SI
20 CMP SI,120AH
21 JNZ NEXT
22 MOV [120H],AL
23 RET
```

2] 8086 program to search element in an array :-

```

01 ; KHAN MOHD OWAIS RAZA 20BCD7138
02 ; ECE2002_Computer Organization & Architecture_Digital Assessment-1
03 ;
04 ; 8086 program to search an element in an array
05 MOV SI,1100H
06 MOV DI,1200H
07 MOV DL,[DI]
08 MOV BL,01H
09 MOV AL,[SI]
10 AGAIN:
11 CMP AL,DL
12 JZ AVAIL
13 INC SI
14 INC BL
15 MOV AL,[SI]
16 CMP AL,20H
17 JNZ AGAIN
18 NODATA:
19 MOV CX,0000H
20 MOV [DI+1],CX
21 MOV [DI+3],CX
22 AVAIL:
23 MOV [DI+1],BH
24 MOV [DI+2],BL
25 MOV [DI+3],SI
26 DATA SEGMENT
27 A DB 09H
28 B DB 02H
29 C DW ?
30 DATA ENDS
31 CODE SEGMENT
32 ASSUME CS:CODE,DS:DATA
33 START:
34 MOV AX,DATA
35 MOV DS,AX
36 MOV AL,A
37 MOV BL,B
38 ADD AL,BL
39 MOV C,AX
40 INT 8
41 OVER:
42 HLT

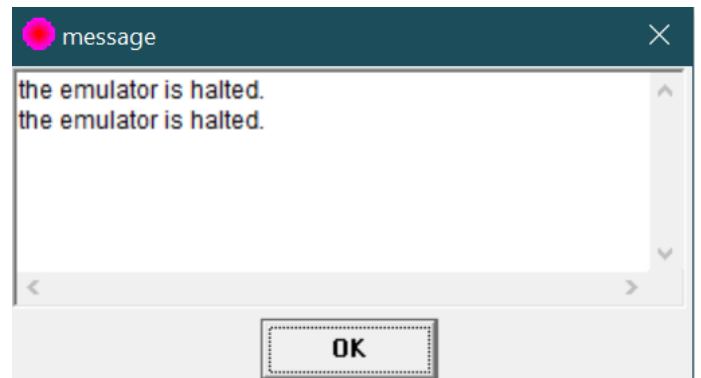
```

The screenshot shows an 8086 emulator interface. At the top, there are control buttons: Load, reload, step back, single step, run, and a step delay slider set to 0 ms. Below these are the registers, organized into two columns (H and L) for each register pair (AX, BX, CX, DX, CS, IP, SS, SP, BP, SI, DI, DS, ES). The values are as follows:

Register	H	L
AX	00	71
BX	00	01
CX	00	00
DX	00	00
CS	0100	
IP	0053	
SS	0100	
SP	FFFE	
BP	0000	
SI	1100	
DI	1200	
DS	0000	
ES	0100	

Below the registers, there are tabs for screen, source, reset, aux, vars, debug, stack, and flags. The main window displays memory addresses and their contents. The current address is 0100:0053. The memory contents are as follows:

Address	Content
01053:	F4 244
01054:	90 144
01055:	90 144
01056:	90 144
01057:	90 144
01058:	90 144
01059:	90 144
0105A:	90 144
0105B:	90 144
0105C:	90 144
0105D:	90 144
0105E:	90 144
0105F:	90 144
01060:	90 144
01061:	90 144
01062:	90 144
01063:	90 144
01064:	90 144
01065:	90 144
01066:	90 144
01067:	90 144
01068:	F4 244

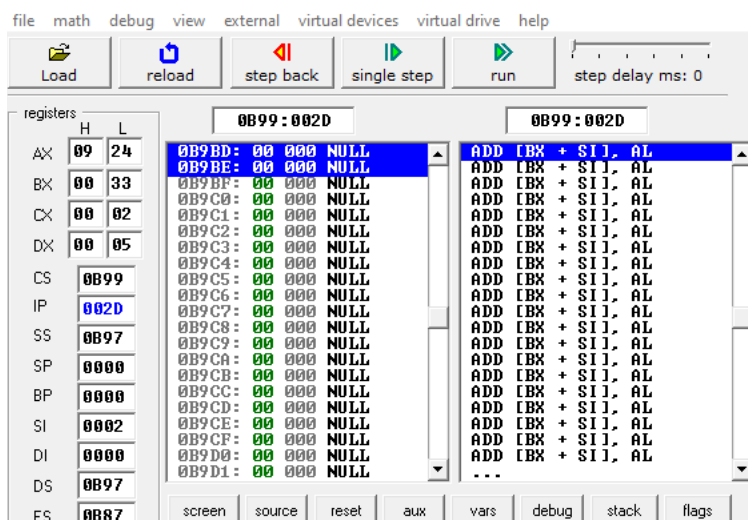
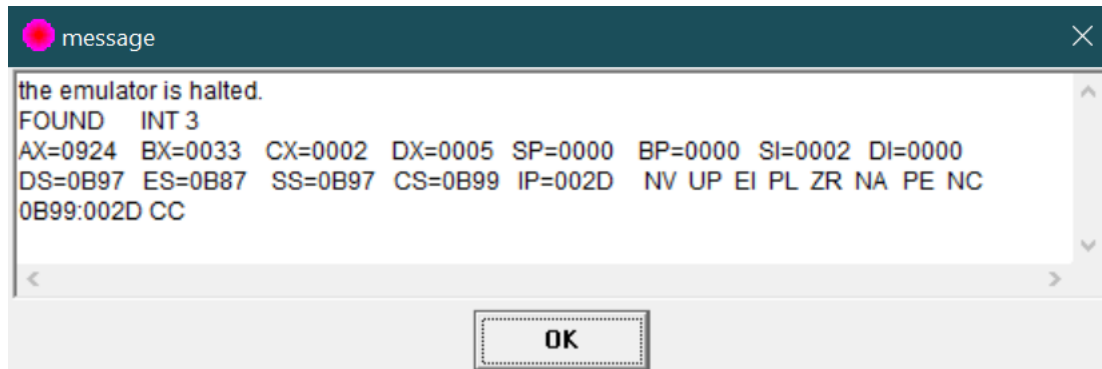


8086 program to search element in array (second method) :-

```

01 ; KHAN MOHD OWAIS RAZA 20BCD7138
02 ; ECE2002 Computer Organization & Architecture Digital Assessment-1
03 ;
04 ; 8086 program to search an element in an array
05 org 100h
06 DATA SEGMENT
07 STRING1 DB 11H,22H,33H,44H,55H
08 MSG1 DB "FOUND$"
09 MSG2 DB "NOT FOUND$"
10 SE DB 33H
11 DATA ENDS
12
13 PRINT MACRO MSG
14 MOV AH, 09H
15 LEA DX, MSG
16 INT 21H
17 INT 3
18 ENDM
19
20 CODE SEGMENT
21 ASSUME CS:CODE, DS:DATA
22 START:
23 MOV AX, DATA
24 MOV DS, AX
25 MOV AL, SE
26 LEA SI, STRING1
27 MOV CX, 04H
28
29 UP:
30 MOV BL, [SI]
31 CMP AL, BL
32 JZ FO
33 INC SI
34 DEC CX
35 JNZ UP
36 PRINT MSG2
37 JMP END1
38
39 FO:
40 PRINT MSG1
41
42 END1:
43 INT 3
44 CODE ENDS
45 END START
46 ret

```



3] 8086 program to add the number to last four digits of registration number :-

Registration number = 20BCD7138

Thus, last four digits are = 7, 1, 3, 8

8086 code to perform addition -

```
01 ; KHAN MOHD OWAIS RAZA 20BCD7138
02 ; ECE2002 Computer Organization & Architecture Digital Assessment-1
03 ;
04
05 ; 8086 program to add element of array with last four digits of registration number
06 ; Registration Number = 20BCD7138
07 .MODEL SMALL
08 .STACK 100H
09 .DATA
10 .CODE
11 MAIN PROC
12     MOV AH,1
13     INT 21H
14     MOV DL,AL
15     MOV AH,1
16     INT 21H
17     ADD DL,AL
18     SUB DL,48
19     MOV AH,2
20     INT 21H
21     MOV AH,4CH
22     INT 21H
23     MAIN ENDP
24 END MAIN
25
```

original source code

```
01 ; KHAN MOHD OWAIS RAZA 20BCD7138
02 ; ECE2002 Computer Organization & Architecture Digital Assessment-1
03 ;
04
05 ; 8086 program to add element of array with last four digits of registration number
06 ; Registration Number = 20BCD7138
07 .MODEL SMALL
08 .STACK 100H
09 .DATA
10 .CODE
11 MAIN PROC
12     MOV AH,1
13     INT 21H
14     MOV DL,AL
15     MOV AH,1
16     INT 21H
17     ADD DL,AL
18     SUB DL,48
19     MOV AH,2
20     INT 21H
21     MOV AH,4CH
22     INT 21H
23     MAIN ENDP
24 END MAIN
25
```

emulator: ADDITION.exe_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers	H	L
AX	00	00
BX	00	00
CX	01	17
DX	00	00
CS	0720	
IP	0000	
SS	0710	
SP	0100	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

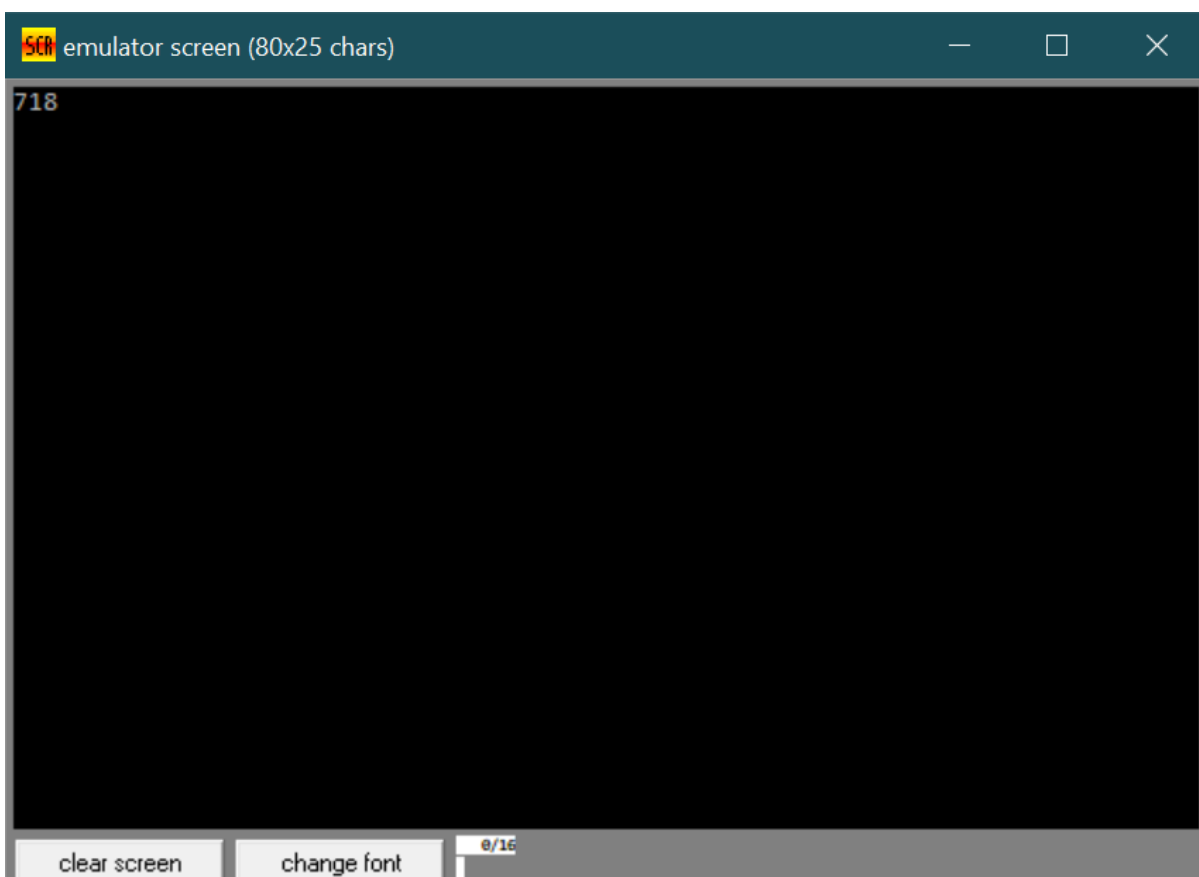
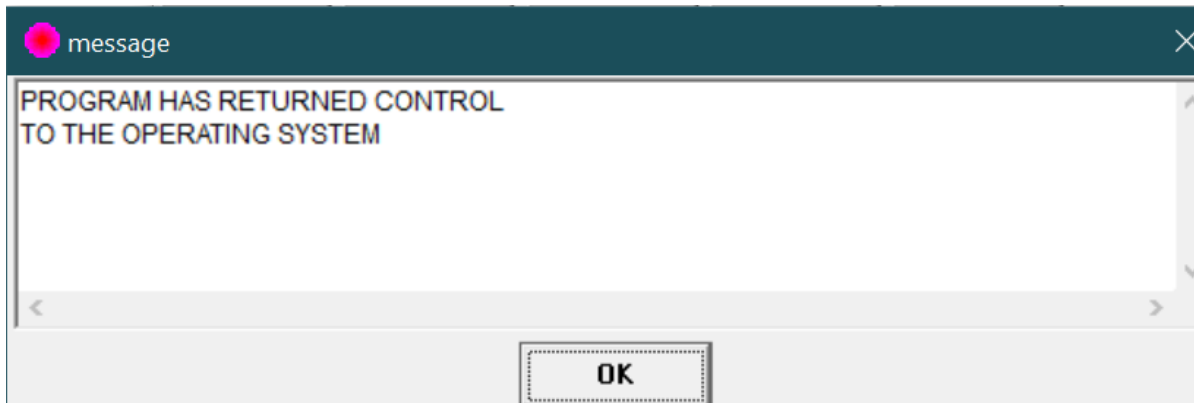
0720:0000 0720:0000

Address	Hex	Dec	Symbol
07200: B4	180	↓	
07201: 01	001	⊙	
07202: CD	205	=	
07203: 21	033	!	
07204: 8A	138	è	
07205: D0	208	u	
07206: B4	180	↓	
07207: 01	001	⊙	
07208: CD	205	=	
07209: 21	033	!	
0720A: 02	002	⊙	
0720B: D0	208	u	
0720C: 80	128	Ç	
0720D: EA	234	Ω	
0720E: 30	048	0	
0720F: B4	180	↓	
07210: 02	002	⊙	
07211: CD	205	=	
07212: 21	033	!	
07213: B4	180	↓	
07214: 4C	076	L	

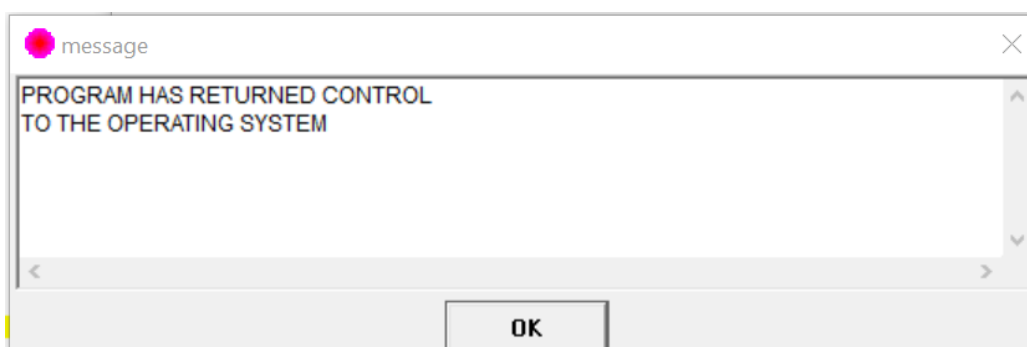
MOV AH, 01h
INT 021h
MOV DL, AL
MOV AH, 01h
INT 021h
ADD DL, AL
SUB DL, 030h
MOV AH, 02h
INT 021h
MOV AH, 04Ch
INT 021h
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...

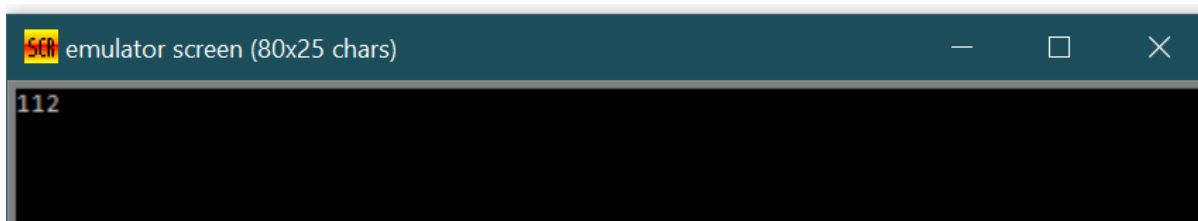
screen source reset aux vars debug stack flags

- 1) When we take first digit 7,
Inputs = 7, 1 & Output = 8

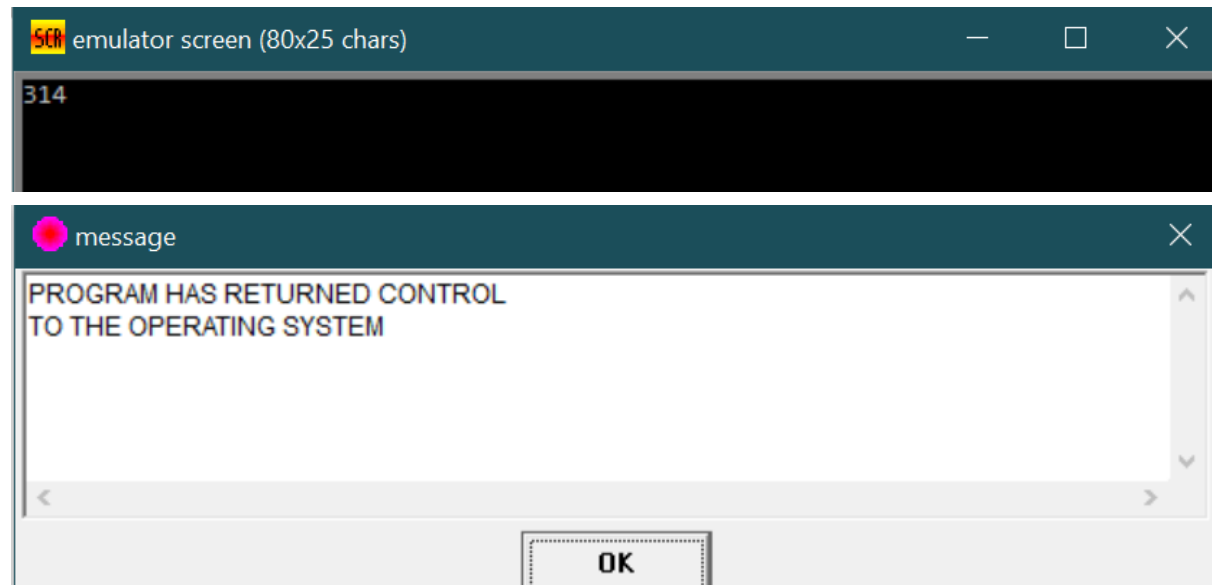


- 2) When we take second digit 1,
Inputs = 1, 1 & Output = 9





3) When we take third digit 3,
Inputs = 3, 1 & Output = 4



4) When we take fourth digit 8,
Inputs = 8, 1 & Output = 9

