

Experiment No. 7: BCD Addition and Subtraction

Date: 06-10-2020

NAME: Gayathri M

REG.NO: 185001050

A. AIM:

Program for performing addition of two 8-bit BCD numbers.

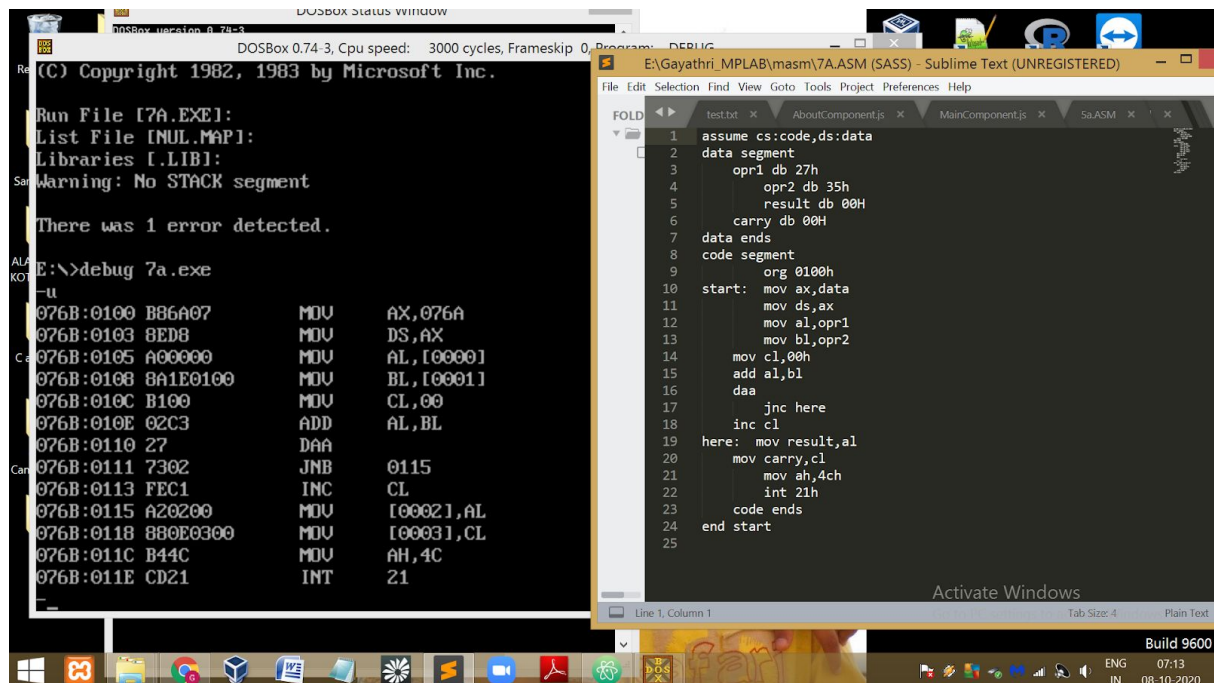
ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load opr1 to al and opr2 to bl
- Load 00h to cl register for carry
- Add al and bl
- Execute daa instruction to adjust the result of the addition of two packed BCD values to create a packed BCD result
- If there is no carry being generated, goto here segment else, increment cl by 1
- In here segment,
 - Load al to result
 - Load cl to carry
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
Start: mov ax,data mov ds,ax mov al,opr1 mov bl,opr2 mov cl,00h add al,bl daa jnc here inc cl	Transferring address of data segment to ds Value of opr1 is loaded to al Value of opr2 is loaded to bl Initializing the value of cl with 00h al=al+bl Add numbers represented in 8-bit packed BCD code Jump to "here" segment if no carry is generated Increments cl by 1
Here: mov result,al mov carry,cl mov ah,4ch int 21h	Load register value of al to result Load cl value to carry Terminate the program

UNASSEMBLED CODE:



The screenshot shows a DOSBox window on the left and a Sublime Text window on the right. The DOSBox window displays the execution of a program, showing memory addresses, hex values, and assembly instructions. The Sublime Text window shows the source assembly code for the program.

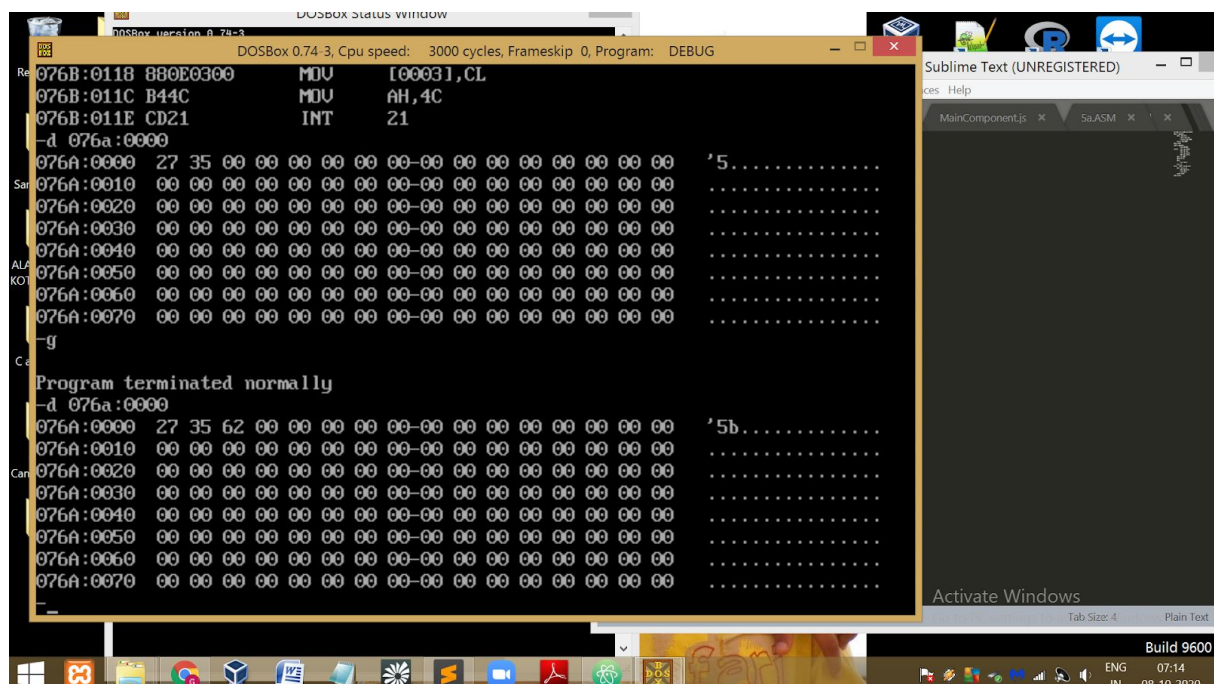
```
Run File [7A.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>debug 7a.exe
-u
076B:0100 B86A07      MOV     AX,076A
076B:0103 B8D8      MOV     DS,AX
076B:0105 A00000      MOV     AL,[0000]
076B:0108 BA1E0100     MOV     BL,[0001]
076B:010C B100      MOV     CL,00
076B:010E 02C3      ADD     AL,BL
076B:0110 27      DAA
076B:0111 7302      JNB     0115
076B:0113 FEC1      INC     CL
076B:0115 A20200      MOV     [0002],AL
076B:0118 8B0E0300     MOV     [0003],CL
076B:011C B44C      MOV     AH,4C
076B:011E CD21      INT     21
```

```
1  assume cs:code,ds:data
2  data segment
3      opr1 db 27h
4      opr2 db 35h
5      result db 00H
6      carry db 00H
7  data ends
8  code segment
9      org 0100h
10 start: mov ax,data
11        mov ds,ax
12        mov al,opr1
13        mov bl,opr2
14        mov cl,00h
15        add al,bl
16        daa
17        jnc here
18        inc cl
19 here:  mov result,al
20        mov carry,cl
21        mov ah,4ch
22        int 21h
23    code ends
24 end start
```

SAMPLE INPUT/OUTPUT:



The screenshot shows the DOSBox window displaying the memory dump and the program's termination. The memory dump shows the state of memory after the program has executed, with values for the carry flag and the result of the addition.

```
076B:0118 8B0E0300      MOV     [0003],CL
076B:011C B44C      MOV     AH,4C
076B:011E CD21      INT     21
-d 076a:0000
076A:0000 27 35 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  '5.....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
-g
Program terminated normally
-d 076a:0000
076A:0000 27 35 62 00 00 00 00 00-00 00 00 00 00 00 00 00 00  '5b.....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
```

RESULT:

Thus addition of two BCD numbers has been performed.

B. AIM:

Program for performing subtraction of two 8-bit BCD numbers.

ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load opr1 to al and opr2 to bl
- Load 00h to cl register
- Subtract al and bl
- Execute das instruction to adjust the result of the subtraction of two packed BCD values to create a packed BCD result
- If al is greater than bl, goto here segment else, increment cl by 1 and find the 10's complement of result and decimal adjust it.
- In here segment,
 - Load dl to result
 - Load cl to carry
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
mov ax,data mov ds,ax mov al,opr1 mov bl,opr2 mov cl,00h sub al,bl das jnc here inc cl mov dl,99h sub dl,al add dl,01h mov al,dl daa	Load data segment to ds Value of opr1 is loaded to al Value of opr2 is loaded to bl Initializing the value of cl with 00h al=al-bl Subtract numbers represented in 8-bit packed BCD code Jump to "here" segment if al>bl Increment value of cl Load dl with 99h dl=dl-al dl=dl+01h Load al with value of dl Add numbers represented in 8-bit packed BCD code
here: mov result,dl mov carry,cl mov ah,4ch int 21h	Load register value of dl to result Load cl value to carry Terminate the program

The image shows a DOSBox window on the left and a Sublime Text editor on the right, both displaying assembly code.

DOSBox Window:

```

Run File [7B.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>debug 7b.exe

-u
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 A00000      MOV     AL,[0000]
076B:0108 8A1E0100     MOV     BL,[0001]
076B:010C B100        MOV     CL,00
076B:010E 2AC3        SUB     AL,BL
076B:0110 2F          DAS
076B:0111 730C        JNB     011F
076B:0113 FEC1        INC     CL
076B:0115 B299        MOV     DL,99
076B:0117 2AD0        SUB     DL,AL
076B:0119 80C201     ADD     DL,01
076B:011C 8AC2        MOV     AL,DL
076B:011E 27          DAA
076B:011F 8B160200     MOV     [0002],DL
  
```

Sublime Text Editor:

```

1  assume cs:code,ds:data
2  data segment
3      opr1 db 64h
4      opr2 db 89h
5      result db 00H
6      carry db 00H
7  data ends
8  code segment
9      org 0100h
10 start: mov ax,data
11         mov ds,ax
12         mov al,opr1
13         mov bl,opr2
14         mov cl,00h
15         sub al,bl
16         das
17         jnc here
18         inc cl
19         mov di,99h
20         sub di,al
21         add di,01h
22         mov al,di
23         daa
24 here:   mov result,di
25         mov carry,cl
26         mov ah,4ch
27         int 21h
28     code ends
29 end start
30
  
```

The DOSBox window shows a warning about no stack segment and a message indicating one error was detected. The Sublime Text editor shows assembly code for a program that calculates the difference between two operands and stores the result in a variable named 'result'.

The screenshot shows a DOSBox emulator window titled "DOSBox Status Window" with the version "DOSBox version 0.74-3". The CPU speed is set to "3000 cycles, Frameskip 0, Program: DEBUG". The main window displays assembly code and memory dump. The assembly code shows instructions like MOV AL, DL, DAA, and MOV [0002], DL. The memory dump shows hex values and their corresponding ASCII characters. The program terminated normally.

Address	Hex	ASCII
076B:011C	8AC2	
076B:011E	27	
076B:011F	8B160200	
-d 076a:0000		
076A:0000	64 89 00 00 00 00 00 00-00 00 00 00 00 00 00 00	d.....
076A:0010	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0020	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0030	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-g		
Program terminated normally		
-d 076a:0000		
076A:0000	64 89 25 01 00 00 00 00-00 00 00 00 00 00 00 00	d.%.....
076A:0010	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0020	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0030	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00

Thus subtraction of two BCD numbers has been performed.