|  |  |  |
| --- | --- | --- |
|  | BCD ADDITION AND SUBTRACTION |  |
| Exp No.: 7 |  | **Name:** S Vishakan |
| Date: 07-10-2020 |  | **Reg. No:** 18 5001 196 |

**AIM:**

To write assembly language programs to perform the following BCD arithmetic operations:

1. BCD Addition.
2. BCD Subtraction.

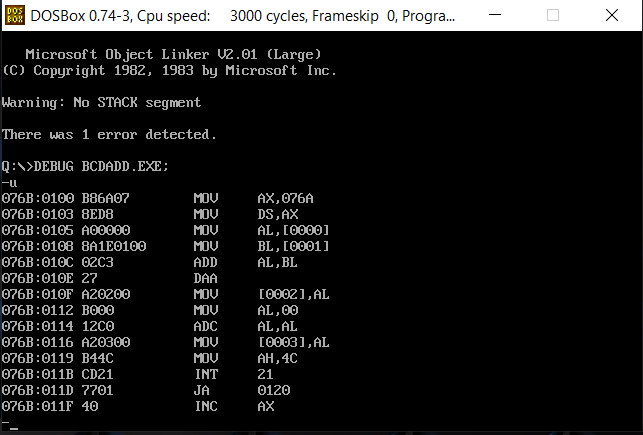
**PROGRAM – 1: BCD ADDITION:**

**ALGORITHM:**

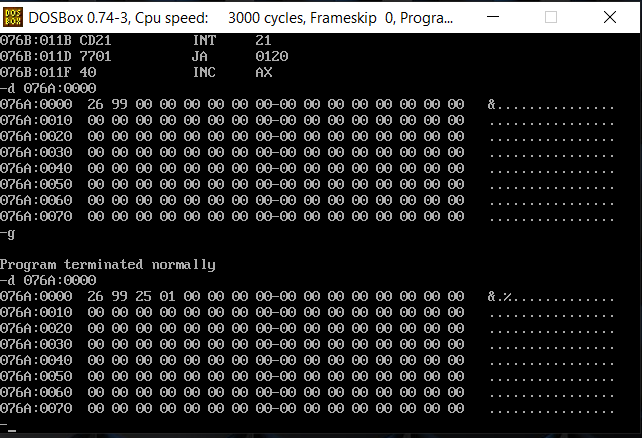
1. Begin.
2. Declare the data segment.
3. Initialize data segment with the 2 BCD numbers and variables for storing their sum and carry.
4. Close the data segment.
5. Declare the code segment.
6. Set a preferred offset (preferably 100h)
7. Load the data segment content into AX register.
8. Transfer the contents of AX register to DS register.
9. Move the contents of the two numbers num1 and num2 to AL and BL register.
10. Add them and store the value in AL.
11. Move the contents of AL to sum.
12. Perform decimal adjust after addition on AL to get BCD result (HEX to BCD)
13. Check if the above adjustment produced a carry.
    1. If carry was produced, set the variable carry to 1.
    2. Else, continue.
14. Transfer the adjusted addition result to the variable sum.
15. Introduce an interrupt for safe exit. (INT 21h)
16. Close the code segment.
17. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| assume cs:code, ds:data | Declare code and data segment. |
|  |  |
| data segment | Initialize data segment with values. |
| num1 db 26h | Stores the first BCD number. |
| num2 db 99h | Stores the second BCD number. |
| res db ? | Variable to store the sum of the 2 numbers. |
| carry db ? | Variable to store the carry of the above sum. |
| data ends |  |
|  |  |
| code segment | Start the code segment. |
| org 0100h | Initialize an offset address. |
| start: mov ax, data | Transfer data from “data” to AX. |
| mov ds, ax | Transfer data from memory location AX to DS. |
| mov al, num1 | Copy num1 to AL. |
| mov bl, num2 | Copy num2 to BL. |
| mov cl, 00h | Clear CL register. |
| add al, bl | AL = AL + BL |
| daa | Adjust HEX result to BCD after subtraction. |
| jnc resume | If carry was not produced, jump to “resume”. |
| inc cl | Increment CL register by 1. |
| resume: mov res, al | Transfer AL contents to variable res. |
| mov carry, cl | Transfer CL contents to variable carry. |
| break: mov ah, 4ch |  |
| int 21h | Interrupt the process with return code and exit. |
| code ends |  |
| end start |  |

**UNASSEMBLED CODE:**



**SAMPLE I/O SNAPSHOT:**



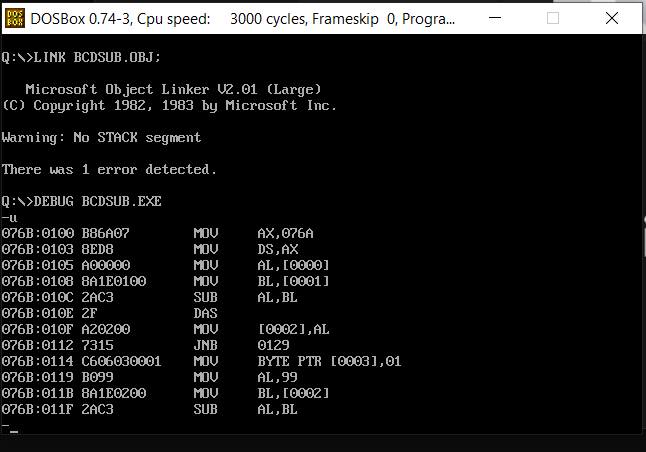
**PROGRAM – 2: BCD SUBTRACTION:**

**ALGORITHM:**

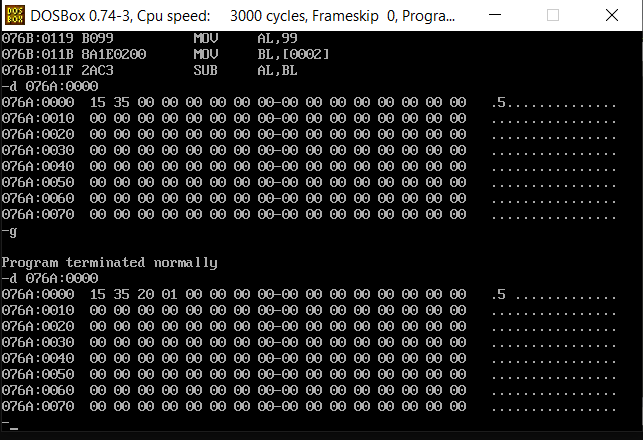
1. Begin.
2. Declare the data segment.
3. Initialize data segment with the 2 BCD numbers and variables for storing their difference (diff) and sign.
4. Close the data segment.
5. Declare the code segment.
6. Set a preferred offset (preferably 100h)
7. Load the data segment content into AX register.
8. Transfer the contents of AX register to DS register.
9. Move the contents of the two numbers num1 and num2 to AL and BL register.
10. Subtract them and store the value in AL.
11. Transfer the contents of AL to diff.
12. If carry flag is set: (Performing 10’s complement)
    1. Set sign as 01h.
    2. Move the contents of diff to BL register.
    3. Move 99h to AL register.
    4. Subtract BL from AL and store the value in AL register.
    5. Move 01h to BL register.
    6. Add AL and BL.
    7. Perform decimal adjust on the addition in AL. (HEX to BCD).
    8. Transfer the contents of AL to diff.
13. Introduce an interrupt for safe exit. (INT 21h)
14. Close the code segment.
15. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| assume cs:code, ds:data | Declare code and data segment. |
|  |  |
| data segment | Initialize data segment with values. |
| num1 db 26h | Stores the first BCD number. |
| num2 db 99h | Stores the second BCD number. |
| diff db ? | Variable to store the difference of the 2 numbers. |
| sign db ? | Variable to store the sign of the above difference. |
| data ends |  |
|  |  |
| code segment | Start the code segment. |
| org 0100h | Initialize an offset address. |
| start: mov ax, data | Transfer data from “data” to AX. |
| mov ds, ax | Transfer data from memory location AX to DS. |
| mov al, num1 | Copy num1 to AL. |
| mov bl, num2 | Copy num2 to BL. |
| sub al, bl | AL = AL – BL |
| das | Adjust HEX result to BCD after subtraction. |
| mov diff, al | Transfer AL contents to diff. |
| jnc break | If carry was not produced, jump to “break”. |
| mov sign, 01h | If carry was produced, set sign to 1. |
| mov al, 99h | Set AL = 99h to perform 9’s complement. |
| mov bl, diff | Transfer diff to BL. |
| sub al, bl | AL = 99h – BL (9’s complement) |
| mov bl, 01h | Set BL = 01h. |
| add al, bl | AL = AL + BL |
| daa | AL value is decimal adjusted after addition (HEX to BCD) |
| mov diff, al | Transfer AL contents to diff. |
| break: mov ah, 4ch |  |
| int 21h | Interrupt the process with return code and exit. |
| code ends |  |
| end start |  |

**UNASSEMBLED CODE:**



**SAMPLE I/O SNAPSHOT:**



**RESULT:**

The assembly level programs were written to perform the above specified BCD arithmetic operations and their output was verified.