|  |  |  |
| --- | --- | --- |
| Exp No: 4 | **Code Conversion** |  |
| Date: 20/9/2020 |  | Name: Anirudh H  Register Number: 185001019 |

**Aim:** To write assembly program to do following Code Conversion:

1. BCD to Hexadecimal
2. Hexadecimal to BCD

**Procedure:**

1. Install all the required file for executing MASM programs.(Masm, edit, link, debug etc..).
2. Write the assembly program in any editor before mounting the folder to the MASM.
3. Mount the folder that contains the assembly program with any name such as “d”.
   * mount d e:\masm
4. Create the object file of the assembly program using masm.
   * masm 16BITADD.asm
5. Use the link to create the executable file of the object file created from the above step.
   * Link 16BITADD.obj
6. Run the executable file using debug.
   * debug 16BITADD.exe
7. By un-assembling the program you can check the code segment of the program
   * u 076b:0100
8. To check the data memory segment, you can use the memory option to view the data stored.
   * d 076a:0000
9. To enter your own values, you can use the enter option which will prompt for new values.
   * e 076a:0000
10. To execute the program, you can use go option
    * G
11. After successful execution and termination of the program, you can check the result by checking the data memory segment
    * d 076a:0000
12. The result can be viewed in the respective address mentioned in the program.

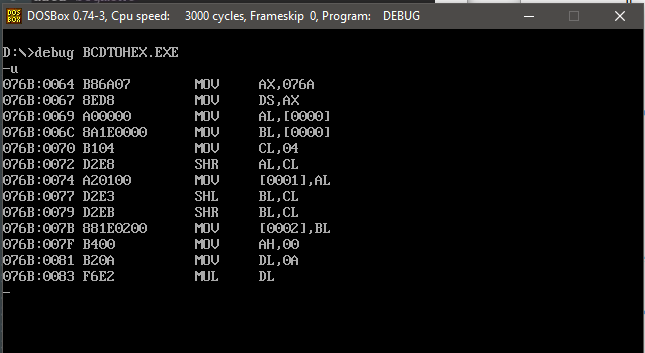
**4 a) BCD to Hexadecimal**

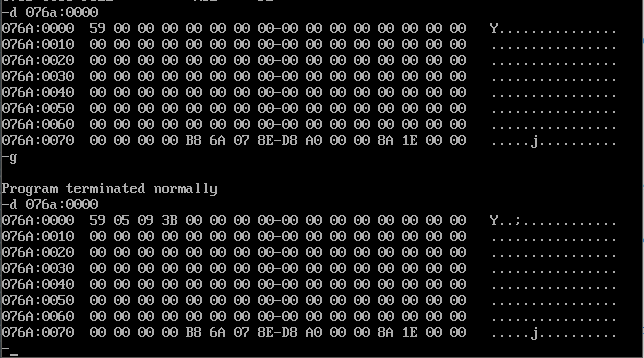
**Algorithm:**

1. Assign data to ax register
2. Load contents of memory location ax in register ds
3. Load contents of memory location bcd in register al
4. Load contents of memory location bcd in register bl
5. Load contents 04h in register cl
6. Shift bits to right of contents in cl and store it in al
7. Load contents of memory location al in one
8. Shift bits to left of contents in cl and store it in bl
9. Shift bits to right of contents in cl and store it in bl
10. Load contents of memory location bl in two
11. Load contents 00h in register ah
12. Load contents 0Ah in register dl
13. Multiply dl => ax/bl
14. Add al => al+bl
15. Load contents of memory location al in register hex
16. Load content 4ch termination code to ah register
17. Stops execution of the program

**Program:**

|  |  |
| --- | --- |
| **Program** | **Comments** |
| assume cs:code,ds:data,es:extra | Initializing the code, data and extra segments to assembler |
| data segment | Data segment |
| bcd db 59h | bcd is declared and initialized to 59h |
| one db 00h | one is declared and initialized to 00h |
| two db 00h | two is declared and initialized to 00h |
| hex db 00h | hex is declared and initialized to 00h |
| data ends |  |
| code segment | Code segment |
| org 0100h | assemble the code starting from address range 0100h |
| start: mov ax,data | Transferring the data from memory location data to ax |
| mov ds,ax | Transferring the data from memory location ax to ds |
| mov al,bcd | Transferring the data from memory location bcd to al |
| mov bl,bcd | Transferring the data from memory location bcd to bl |
| mov cl,04h | Transferring the 04h to cl |
| shr al,cl | Shift bits to right of contents in cl and store it in al |
| mov one,al | Transferring the data from memory location al to one |
| shl bl,cl | Shift bits to left of contents in cl and store it in bl |
| shr bl,cl | Shift bits to right of contents in cl and store it in bl |
| mov two,bl | Transferring the data from memory location bl to two |
| mov ah,00h | Transferring the 00h to ah |
| mov dl,0Ah | Transferring the 0Ah to dl |
| mul dl | Multiply dl => ax/bl |
| add al,bl | Add al => al+bl |
| mov hex,al | Transferring the data from memory location al to hex |
| mov ah,4ch | Transferring the termination code 4ch to ah |
| int 21h | Termination |
| code ends | Code ends |
| end start |  |

**Unassembled code:**

**Sample Input and output:**

**Result:**

Thus, the assembly program for BCD to Hexadecimal conversion is written and executed.

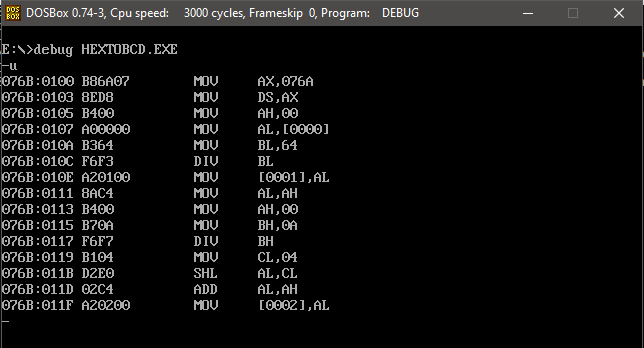
**4 b) Hexadecimal to BCD**

**Algorithm:**

1. Assign data to ax register
2. Load contents of memory location ax in register ds
3. Load contents 00h in register ah
4. Load contents of memory location input in register al
5. Load contents 64h in register bl
6. Div bl => ax/bl
7. Load contents of memory location al in out1
8. Load contents of memory location ah in al
9. Load contents 00h in register ah
10. Load contents 0Ah in register bh
11. Div bl => ax/bl
12. Load contents 04h in register cl
13. Shift bits to left of contents in cl and store it in bl
14. Add al => al+ah
15. Load contents of memory location al in register out2
16. Load content 4ch termination code to ah register
17. Stops execution of the program

**Program:**

|  |  |
| --- | --- |
| **Program** | **Comments** |
| assume cs:code,ds:data,es:extra | Initializing the code, data and extra segments to assembler |
| data segment | Data segment |
| input db 0AAh | input is declared and initialized to 0AAh |
| out1 db 00h | Out1 is declared and initialized to 00h |
| Out2 db 00h | Out2 is declared and initialized to 00h |
| data ends |  |
| code segment | Code segment |
| org 0100h | assemble the code starting from address range 0100h |
| start: mov ax,data | Transferring the data from memory location data to ax |
| mov ds,ax | Transferring the data from memory location ax to ds |
| mov al,input | Transferring the data from memory location input to al |
| mov bl,64h | Transferring the data 64h to bl |
| div bl | Div bl => ax/bl |
| mov out1,al | Transferring the data from memory location al to out1 |
| mov al,ah | Transferring the data from memory location ah to al |
| mov ah,00h | Transferring the data 00h to ah |
| mov bh,0Ah | Div bl => ax/bl |
| div bh | Div bl => ax/bl |
| mov cl,04h | Transferring the data 04h to cl |
| shl al,cl | Shift bits to left of contents in cl and store it in al |
| add al,ah | Add al => al+ah |
| mov out2,al | Transferring the data from memory location al to out2 |
| mov ah,4ch | Transferring the termination code 4ch to ah |
| int 21h | Termination |
| code ends | Code ends |
| end start |  |

**Unassembled code:**

**Sample Input and output:**

**Result:**

Thus, the assembly program for Hexadecimal to BCD conversion is written and executed.