**SSN College of Engineering Department of Computer Science and Engineering**

**III year - UCS1512 – Microprocessors Lab**

**String Manipulations**

**Exp No:** 04

**Name:** Rahul Ram M

**Register Number:** 185001121

**Date:** 15/08/2020

**Code Conversion**

**a) BCD to Hexadecimal:**

**Aim:**

Design 8086 program for converting BCD to hexadecimal.

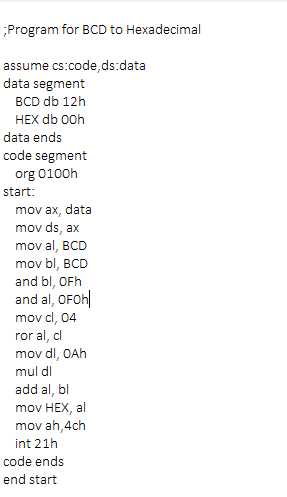
**Procedure for executing MASM:**

1. Run Dosbox and mount your masm folder to a drive in dosbox.
2. Goto the mounted drive.
3. Save the 8086 program with extension .asm in the same folder using command “edit”
4. After creating the file, assemble it using the command “masm filename.asm”
5. Link the file using the command “link filename.obj;”
6. Use debug command with filename.exe to execute and analyse the memory contents, “debug filename.exe”.
7. In debug, command “u” will display the unassembled code.
8. Use command “d segment:offset” to see the content of memory locations starting from segment:offset address.
9. To change the value in memory, use the command “e segment:offset”
10. Verify the memory contents to ensure the updates (using command “d”).
11. . Execute using the command “g” and check the outputs.
12. “q” to exit from debug and “exit” to exit from command prompt and to close the Dosbox.

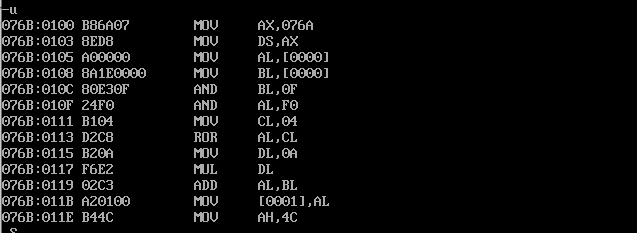
**Algorithm:**

1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the value of the variable BCD to AL and BL register.
3. Separating the numbers in BCD into two bytes with its value in lower order.
4. This is done by performing AND operation with AL and BL register and rotating 4 bits (specified in CL register) of AL register to the right using ROR AL, CL.
5. Move 0AH to DL register.
6. Multiply AL and DL using MUL DL and store the value in AX register.
7. Add AL and BL using ADD AL, BL and result is stored in AL register which is the hexadecimal value of the given BCD number.
8. Move the value of the AL register to HEX.
9. Move the hexadecimal value 4C into AH register.
10. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

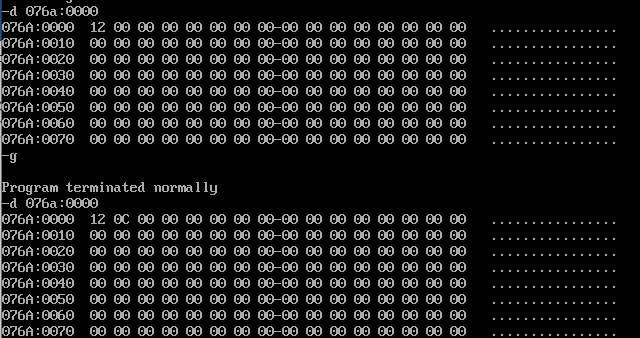
**Program:**



|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0010H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AL, BCD | Move value from BCD to AL register. |
| MOV BL, BCD | Move value from BCD to BL register. |
| AND BL, 0FH | BL <- BL AND 0FH |
| AND AL, 0F0H | AL <- AL AND F0H |
| MOV CL, 04H | CL <- 04H |
| ROR AL, CL | Rotate AL to the right specified by CL register. |
| MOV DL, 0AH | DL <- 0AH |
| MUL DL | DL <- AL x DL |
| ADD AL, BL | AL <- AL + BL |
| MOV AH, 4CH  INT 21H | Terminates the program. |

**Snapshot of sample input and output:**

**Converting BCD (12) to Hexadecimal (0CH):**



**Result:**

Thus the 8086 program for converting BCD to hexadecimal is executed successfully in DOS-BOX.

**b) Hexadecimal to BCD:**

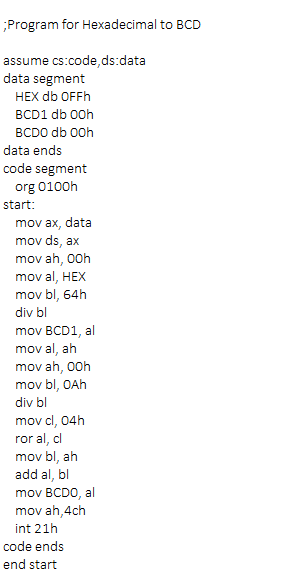
**Aim:**

Design 8086 program for converting hexadecimal to BCD.

**Algorithm:**

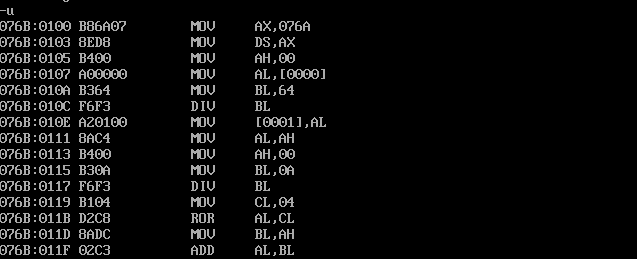
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move 00H to AH register.
3. Move the value of HEX to AL register.
4. Move 64H which is the hexadecimal representation for BCD value 100 to BL register.
5. Divide AX by BL using DIV BL.
6. Quotient is in AL register which is moved to BCD1.
7. Move the value from AH register(remainder) to AL register.
8. Move 00H to AH register.
9. Move 0AH which is the hexadecimal representation for BCD value 10 to BL register.
10. Divide AX by BL using DIV BL.
11. Pack the quotient and remainder into a BCD value.
12. Move 04H to CL register and rotate the value of the AL register by 4 bits right using ROR Al, CL.
13. Move the value of ah to bl.
14. Add AL and BL register and result is in AL register.
15. Move the value of AL register to BCD0.
16. Move the hexadecimal value 4C into AH register.
17. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

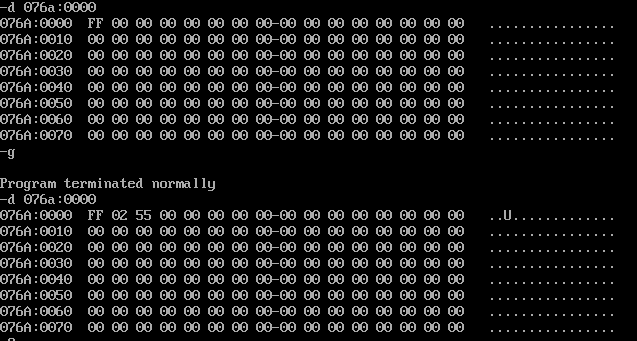
**Program:**



|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0010H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AH, 00H | AH <- 00H |
| MOV AL, HEX | AL <- HEX |
| MOV BL, 64H | BL <- 64H |
| DIV BL | AX / BL => quotient -> AL, remainder -> AH |
| MOV BCD1, AL | AL <- BCD1 |
| MOV AL, AH | AL <- AH |
| MOV AH, 00H | AH <- 00H |
| MOV BL, 0AH | BL <- 0AH |
| DIV BL | AX / BL => quotient -> AL, remainder -> AH |
| MOV CL, 04H | CL <- 04H |
| ROR AL, CL | Rotate AL to the right specified by CL register. |
| MOV BL, AH | BL <- AH |
| ADD AL, BL | AL + BL -> AL |
| MOV BCD0, AL | BCD0 <- AL |
| MOV AH, 4CH  INT 21H | Terminates the program. |

**Snapshot of sample input and output:**

**Converting Hexadecimal (FFH) to BCD (255):**



**Result:**

Thus the 8086 program for converting hexadecimal to BCD is executed successfully in

DOS-BOX.