UCS1512 – Microprocessors Lab

Exp No : 4

Code Conversion

Date : 19/09/2020

Name : Gayathri M
Reg No. : 185001050

4A. **Aim :**

To write an assembly level program to convert a packed BCD number to its equivalent hexadecimal number using an 8086 microprocessor.

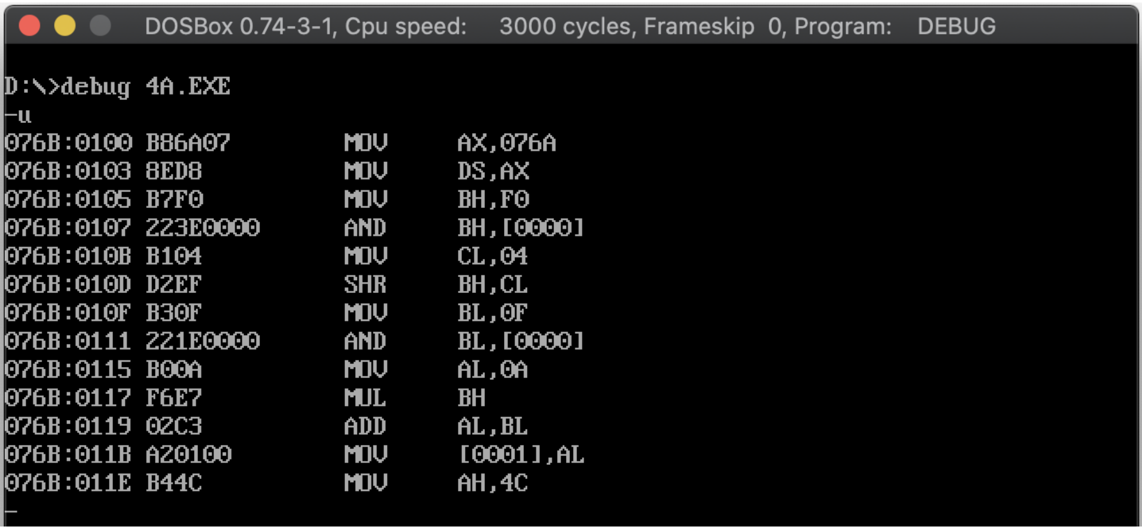
Algorithm :

1. Initialize the data segment.
2. Initialize the extra segment.
3. Perform bitwise AND of given BCD and 0F0h and shift right by 4 bits to extract tens digit and store it in BH.
4. Perform bitwise AND of given BCD and 0Fh to extract ones digit.
5. Initialize AL with 0Ah which is hexadecimal equivalent of 10.
6. Multiply BH with AL.
7. Add AL and BL.
8. Copy AL to hexval.
9. Terminate the program.

Program :

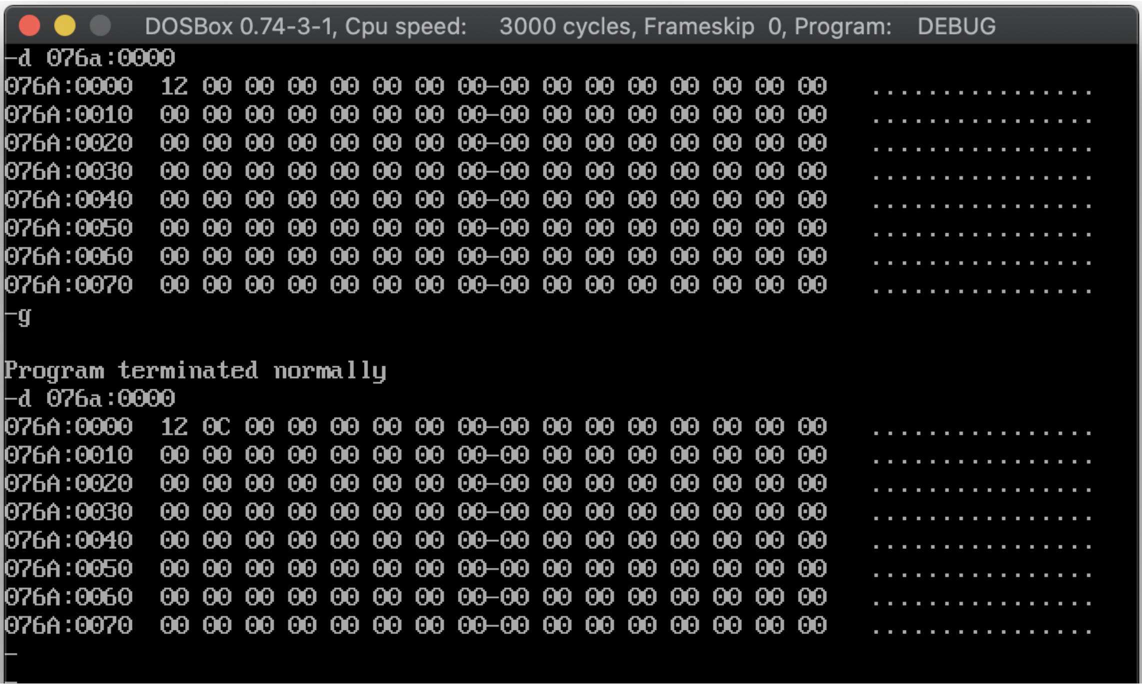
PROGRAM	COMMENTS
`start:`	
`mov ax,data`	Transfer address of data segment to ds
`mov ds,ax`	
`mov bh,0F0h`	Copy F0h to BH
`and bh,bcd`	Bitwise AND BH and given BCD (to extract 10’s digit)
`mov cl,04h`	Initialize CL to 04h
`shr bh,cl`	Shift right BH by value in CL to get 10’s digit
`mov bl,0Fh`	Copy 0Fh to BL
`and bl,bcd`	Bitwise AND BLand given BCD (to extract 1’s digit)
`mov al, 0Ah`	Initialize AL to 0Ah
`mul bh`	Multiply BH
`add al,bl`	Add AL and BL, store in AL
`mov hexval, al`	Copy value in AL to hexval
`mov ah,4ch`	Termination of program.
`int 21h`	
`code ends`	
`end start`	

Unassembled Code :



```
D:\>debug 4A.EXE
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 B7F0          MOV     BH,F0
076B:0107 223E0000      AND     BH,[0000]
076B:010B B104          MOV     CL,04
076B:010D D2EF          SHR     BH,CL
076B:010F B30F          MOV     BL,0F
076B:0111 221E0000      AND     BL,[0000]
076B:0115 B00A          MOV     AL,0A
076B:0117 F6E7          MUL     BH
076B:0119 02C3          ADD     AL,BL
076B:011B A20100        MOV     [0001],AL
076B:011E B44C          MOV     AH,4C
-
```

Sample Input/Output:



```
-d 076a:0000
076A:0000  12 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
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  12 0C 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
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   ................
-
```

Result:

Thus assembled and executed an 8086 program for converting a BCD number to a hexadecimal number successfully.

4B. Aim :

To write an assembly level program to convert a hexadecimal number to its equivalent packed BCD number using an 8086 microprocessor.

Algorithm :

1. Initialize the data segment.
2. Initialize the extra segment.
3. Initialize AL to the given hexadecimal number.
4. Divide the number by 64h (100) to get 100's digit and store the quotient to higher order byte of the BCD result
5. Now move the remainder of the previous division to the AL register.
6. Divide the value in AL register by 0Ah (10) to get 10's place value.
7. Shift left the quotient in AL by 4 bits and add with remainder AH. This gives the packed lower order byte of BCD result.
8. Copy the result in AL to bcd.
9. Terminate the program

Program :

PROGRAM	COMMENTS
`start:`	
`mov ax,data` `mov ds,ax`	Transfer address of data segment to ds
`mov al,hexval`	Initialize AL to hexvalue
`mov bl,64h`	Initialise BL to 64h
`mov ah,00h`	Initialise AH to 00h
`div bl`	Divide AX by BL and store quotient in Al and remainder in AH

`mov byte ptr bcd+1,al`	Copy value in AL to higher order by of bcd

`mov al,ah`
`mov bl,0Ah`
`mov ah,00h`
	Copy AH to AL
	Copy 0Ah to BL
	Copy 00 to AH

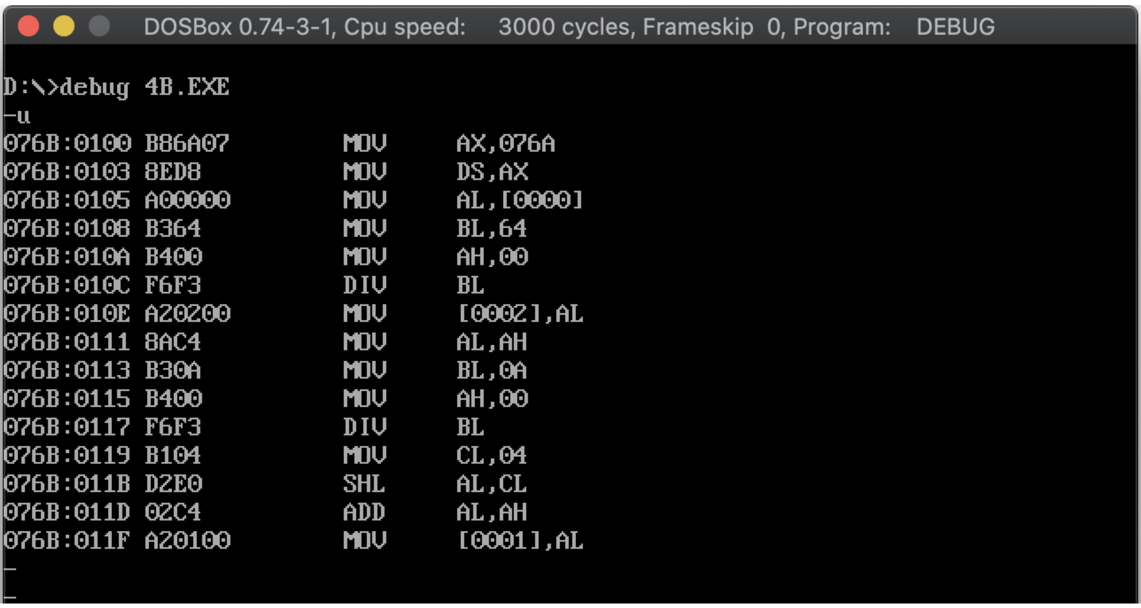
`div bl`	Divide AX by BL and store quotient in Al and remainder in AH

`mov cl,04h`
`shl al,cl`
	Copy 04 to CL
	Shift left AL by value in CL to multiply by 10

`add al,ah`
`mov byte ptr bcd,al`
	Add AL and AH, store in AL
	Copy value in AL to bcd

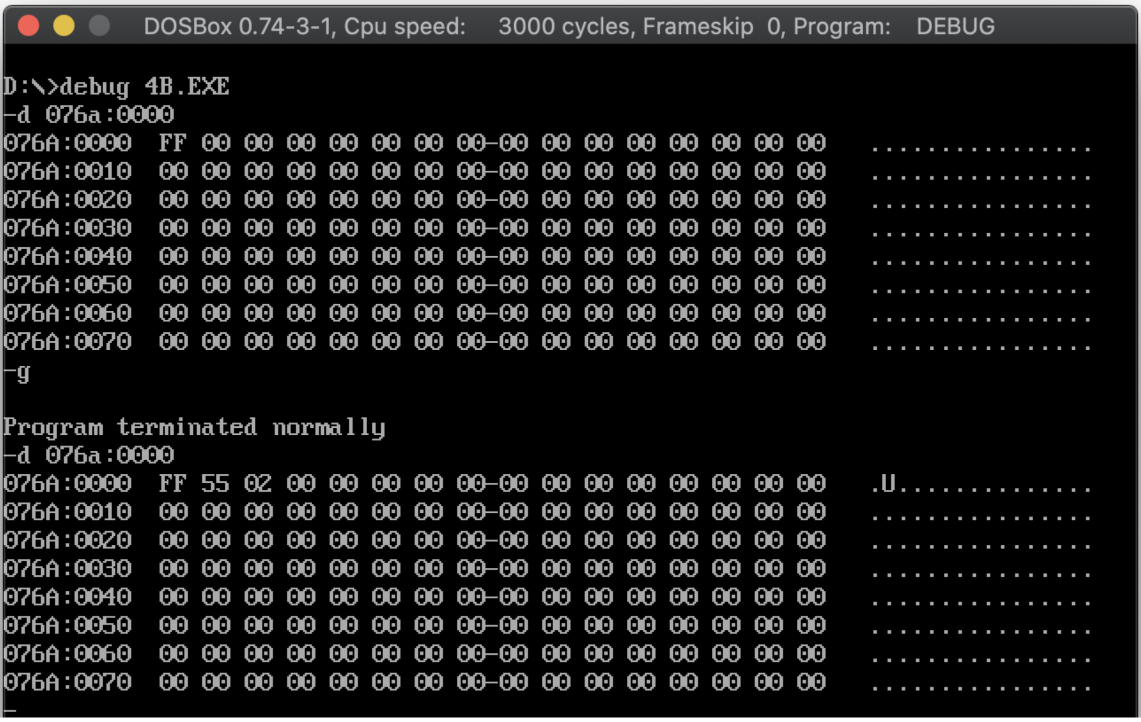
`mov ah,4ch`
`int 21h`
`code ends`
`end start`
	Termination of program.

Unassembled Code :



```
D:\>debug 4B.EXE
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 A00000        MOV     AL,[0000]
076B:0108 B364          MOV     BL,64
076B:010A B400          MOV     AH,00
076B:010C F6F3          DIV     BL
076B:010E A20200        MOV     [0002],AL
076B:0111 8AC4          MOV     AL,AH
076B:0113 B30A          MOV     BL,0A
076B:0115 B400          MOV     AH,00
076B:0117 F6F3          DIV     BL
076B:0119 B104          MOV     CL,04
076B:011B D2E0          SHL     AL,CL
076B:011D 02C4          ADD     AL,AH
076B:011F A20100        MOV     [0001],AL
-
```

Sample Input/Output:


DOSBox 0.74-3-1, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG

```
D:\>debug 4B.EXE
-d 076a:0000
076A:0000  FF 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
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  FF 55 02 00 00 00 00 00-00 00 00 00 00 00 00 00   .U..............
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   ................
-
```

Result:

Thus assembled and executed an 8086 program for converting a hexadecimal number to its equivalent packed BCD number successfully.