Exp No: 4

Date: 22/09/2020

Code Conversion

Name: Swetha Saseendran

Reg No: 185001183

Aim:

To write assembly language programs to perform the following code conversions:

1. BCD to Hexadecimal Code Conversion
2. Hexadecimal to BCD Code Conversion

Programs:

(i) BCD to Hexadecimal Code Conversion

Algorithm:

- Move the data segment to DS register through AX register.
- Move the extra segment to ES register through AX register.
- Clear AH register.
- Load the BCD value to AL.
- Load 10H to BL.
- Divide the value at AL by BL.
- Load the LSB at AH to DL.
- Multiple AL by 10 and add it to value at DL.
- Move the result at AL to HEX. 16.

Program	Comments
ASSUME CS:CODE,DS:DATA	Using assume directive to declare data,extra and code segment
DATA SEGMENT BCD DB 12H HEX DB ? DATA ENDS	Declaring and initialising variables in data segment
CODE SEGMENT *ORG 0100h*	Set location for code segment at 0100h

STARTt:	
MOV AX,data	Move the content of Data segment to AX register
MOV DS,AX	Move the content of AX register to DS register
MOV AL,BCD	Move the content of BCD to AL register
MOV AH,0H	Move the 0H to AH register..
MOV BL,10H	Assign the offset of source to SI register.
DIV BL	Divide AX by BL. (Quotient in AL, Remainder in AH)
MOV BL,0AH	Transfer 10 to BL.
MOV DL,AH	Copy the contents of AH to DL.
MOV AH,0H	Clear AH register.
MUL BL	AX = AL * BL (Multiply MSB by 10)
ADD AL,DL	AL = AL + DL (Add LSB to the hex result)
MOV AH,4CH	Moves the hexadecimal value 4c to ah.
INT 21H	When Software interrupt 21 is called with AH=4C, then current process terminates
CODE ENDS	Ending the code segment
END START	Ending start segment

Unassembled Code:

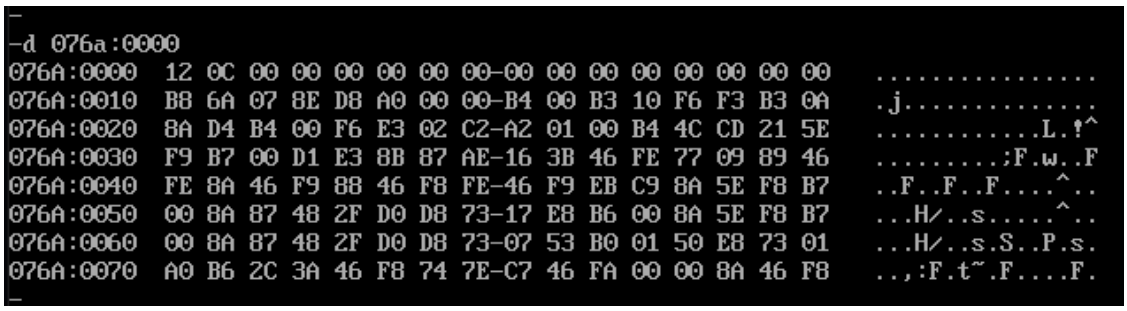
```
-u
076B:0000 B86A07        MOV     AX,076A
076B:0003 8ED8          MOV     DS,AX
076B:0005 A00000        MOV     AL,[0000]
076B:0008 B400          MOV     AH,00
076B:000A B310          MOV     BL,10
076B:000C F6F3          DIV     BL
076B:000E B30A          MOV     BL,0A
076B:0010 8AD4          MOV     DL,AH
076B:0012 B400          MOV     AH,00
076B:0014 F6E3          MUL     BL
076B:0016 02C2          ADD     AL,DL
076B:0018 A20100        MOV     [0001],AL
076B:001B B44C          MOV     AH,4C
076B:001D CD21          INT     21
076B:001F 5E            POP     SI
```

Snapshot of sample input and output:

INPUT: Packed BCD No- 12

```
-d 076a:0000
076A:0000  12 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  B8 6A 07 8E D8 A0 00 00-B4 00 B3 10 F6 F3 B3 0A   .j..............
076A:0020  8A D4 B4 00 F6 E3 02 C2-A2 01 00 B4 4C CD 21 5E   ............L.!^
076A:0030  F9 B7 00 D1 E3 8B 87 AE-16 3B 46 FE 77 09 89 46   .........;F.w..F
076A:0040  FE 8A 46 F9 88 46 F8 FE-46 F9 EB C9 8A 5E F8 B7   ..F..F..F....^..
076A:0050  00 8A 87 48 2F D0 D8 73-17 E8 B6 00 8A 5E F8 B7   ...H/..s.....^..
076A:0060  00 8A 87 48 2F D0 D8 73-07 53 B0 01 50 E8 73 01   ...H/..s.S..P.s.
076A:0070  A0 B6 2C 3A 46 F8 74 7E-C7 46 FA 00 00 8A 46 F8   ..,:F.t~.F....F.
```

OUTPUT: Hexadecimal equivalent- 0C



(ii) Hexadecimal to BCD Code Conversion

Algorithm:

- Move the data segment to DS register through AX register.
- Move the extra segment to ES register through AX register.
- Load 100(64H) to BL.
- Divide the value at AX by BL.
- Move the MSB at AL to CL.
- Move the LSBs at AH to AL.
- Clear AH register.
- Load the 10(0AH) to BL.
- Dive the value at AX by BL.
- Move the second bit of BCD to CH.
- Move the LSB of BCD to DL.
- Apply [CL]*100 + [CH]*10 + [DL] and store the result at AX.
- Move the result at AX to BCD.

Program	Comments
ASSUME CS:CODE,DS:DATA,	Using assume directive to declare data, and code segment
DATA SEGMENT HEX DB0FFH BCD DW ? DATA ENDS	Declaring and initialising variables in data segment
CODE SEGMENT *ORG 0100H*	Set location for code segment at 0100h

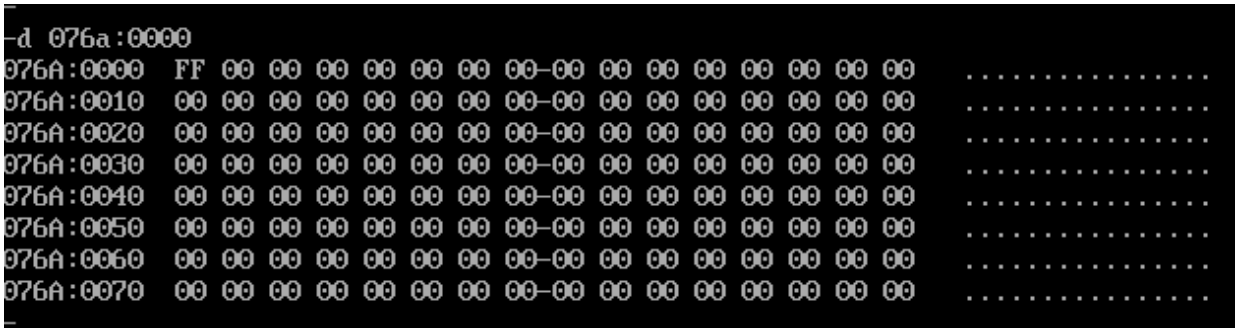
START:	
MOV AX,DATA	Move the content of Data segment to AX register
MOV DS,AX	Move the content of AX register to DS register
MOV AL, HEX	Transfer the given BCD byte to AL.
MOV AH, 0H	Clear AH register. mov bl, 64h Transfer 100 to BL.
MOV BL, 64H	Transfer 100 to BL.
DIV BL	Divide AX by BL. (Quotient in AL, Remainder in AH)
MOV CL, AL	Transfer the quotient to CL register. (MSB of BCD)
MOV AL, AH	Transfer the remainder to AL register.
MOV AH, 0H	Clear AH register.
MOV BL, 0AH	Transfer 10 to BL.
DIV BL	Divide AX by BL.
MOV CH, AL	Transfer the quotient to CH register. (2nd MSB of BCD)
MOV DL, AH	Transfer the remainder to DL register. (LSB of BCD)
MOV BL, 10H	Transfer 16 to BL.
MOV AL, CL	Transfer the MSB of BCD to AL register.
MUL BL	AX = AL * BL (Multiply MSB by 10)
ADD AL, CH	AL = AL + CH (Add 2 nd MSB to the BCD result)
MUL BL	AX = AL * BL (MSB * 100 + 2nd MSB * 10)
ADD AL, DL	AL = AL + DL (MSB * 100 + 2nd MSB * 10 + LSB)
MOV BCD, AX	Store the value in AX as the final BCD converted code
MOV AH,4CH	Moves the hexadecimal value 4c to ah.
INT 21H	When Software interrupt 21 is called with AH=4C, then current process terminates
CODE ENDS	Ending the code segment
END START	Ending start segment

Unassembled Code:

```
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 A00000        MOV     AL,[0000]
076B:0108 B400          MOV     AH,00
076B:010A B364          MOV     BL,64
076B:010C F6F3          DIV     BL
076B:010E 8AC8          MOV     CL,AL
076B:0110 8AC4          MOV     AL,AH
076B:0112 B400          MOV     AH,00
076B:0114 B30A          MOV     BL,0A
076B:0116 F6F3          DIV     BL
076B:0118 8AE8          MOV     CH,AL
076B:011A 8AD4          MOV     DL,AH
076B:011C B310          MOV     BL,10
076B:011E 8AC1          MOV     AL,CL
```

Snapshot of sample input and output:

INPUT: Hexadecimal value-FF



OUTPUT: BCD output in packed form- location1 = 02, location2 = 55

Result:

The assembly level programs were written to perform the above specified code conversions and the output was verified.