

NAME: GINI CHACKO

SEMESTER: IV

CLASS: SE COMPS B

BATCH: B

ROLL: 8942

TOPIC: MP EXPERIMENT 5:

To Perform Code Conversion

- **Hex to BCD**
- **BCD to Hex**
- **ASCII to BCD**
- **BCD to ASCII**

CODE:

```
.8086
.model small
.data
num db 37h
num1 db 35h
num2 db 22h
num3 db 35h

res db ?
res1 db ?
res2 db ?
result db ?

msg2 db 'option 1: Ascii to BCD [ascii value is 37 ] $'
msg3 db 'option 2: BCD to Ascii [ascii value is 35 ] $'
msg4 db 'option 3: Hex to BCD [ascii value is 22 ] $'
msg5 db 'option 4: BCD to Hex [ascii value is 35 ] $'
msg1 db 'Enter option $'

.code
Start:
mov ax,@data
mov ds, ax

lea dx,msg2
mov ah,09h
int 21h

lea dx,msg3
mov ah,09h
int 21h

lea dx,msg4
mov ah,09h
int 21h

lea dx,msg5
mov ah,09h
```

int 21h

lea dx,msg1
mov ah,09h
int 21h

mov ah,08h
int 21h

cmp al,31h
jnz next
;AsciiToBCD
 mov al, num
 sub al, 30h
 mov res ,al
 jmp exit
next :cmp al,32h
jnz next1

;BCDtoAscii
 mov al,num1
 and al,0F0h
 ror al, 4
 add al,30h
 mov bl,num1
 and bl,0Fh
 add bl,30h
 mov res1, al
 mov res2, bl
 jmp exit
next1:cmp al,33h
jnz next2

;HextoDEC
 mov al,num2
 mov ah,00h
 mov bl,0Ah
 div bl
 ror al,04h
 add al,ah

```

        jmp exit

next2:cmp al,34h
;BCDtoHEX
        mov al,num3
        mov dl,00h
        mov bl,al
        and bl, 0F0h
        mov cl,04h
        ror bl,cl
back: add dl, 0Ah
        dec bl
        jnz back
        and al,0Fh
        add dl, al
        mov result ,dl
        jmp exit

exit:int 3h
end start

```

OUTPUT:

```

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TD

C:\TASM>tasm exp5.asm
Turbo Assembler Version 3.0 Copyright (c) 1988, 1991 Borland International

Assembling file:   exp5.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  474k

C:\TASM>tlink exp5.obj
Turbo Link Version 2.0 Copyright (c) 1987, 1988 Borland International
Warning: no stack

C:\TASM>td exp5
Turbo Debugger Version 3.1 Copyright (c) 1988,92 Borland International
option 1: Ascii to BCD [ascii value is 37 ] option 2: BCD to Ascii [ascii value
is 35 ] option 3: Hex to BCD [ascii value is 22 ] option 4: BCD to Hex [ascii va
lue is 35 ] Enter option _

```

1.] ASCII to BCD (Option 1)

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TD

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

CS:009E CC	int	03	ax	0807	c=0
CS:009F 0037	add	[bx],dh	bx	0000	z=0
CS:00A1 352235	xor	ax,3522	cx	0000	s=0
CS:00A4 07	pop	es	dx	00B8	o=0
CS:00A5 0000	add	[bx+si],al	si	0000	p=0
CS:00A7 006F70	add	[bx+70],ch	di	0000	a=0
CS:00AA 7469	je	0115	bp	0000	i=1
CS:00AC 6F	outsw		sp	0000	d=0
CS:00AD 6E	[.] = Dump				
CS:00AE 2031	ds:0000	37 35 22 35 07 00 00 00			75 "5
CS:00B0 3A20	ds:0008	6F 70 74 69 6F 6E 20 31			option 1
CS:00B2 41	ds:0010	3A 20 41 73 63 69 69 20			: Ascii
CS:00B3 7363	ds:0018	74 6F 20 42 43 44 20 5B			to BCD [
es:0000 CD 20 FF 9F 00 EA FF FF	=	f r			
es:0008 AD DE E0 01 C5 15 AA 01	:	[x] [S] [0]			
es:0010 C5 15 89 02 20 10 92 01	:	[S] [0] [A] [0]			
es:0018 01 03 01 00 02 FF FF FF		[0] [0] [0] [0]			
ss:0002 6474					
ss:0000 0000					

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

2.] BCD to ASCII (Option 2)

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TD

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

CS:009E CC	int	03	ax	0833	c=0
CS:009F 0037	add	[bx],dh	bx	0035	z=0
CS:00A1 352235	xor	ax,3522	cx	0000	s=0
CS:00A4 0033	add	[bp+di],dh	dx	00B8	o=0
CS:00A6 35006F	xor	ax,6F00	si	0000	p=1
CS:00A9 7074	jo	011F	di	0000	a=0
CS:00AB 696F6E2031	imul	bp,[bx+6E],31	bp	0000	i=1
CS:00B0 3A20	cmp	ah,[bx+si]	sp	0000	d=0
CS:00B2 41	[.] = Dump				
CS:00B3 7363	ds:0000	37 35 22 35 00 33 35 00			75 "5 35
CS:00B5 696920	ds:0008	6F 70 74 69 6F 6E 20 31			option 1
CS:00BA 204243	ds:0010	3A 20 41 73 63 69 69 20			: Ascii
CS:00BD 44	ds:0018	74 6F 20 42 43 44 20 5B			to BCD [
es:0000 CD 20 FF 9F 00 EA FF FF	=	f r			
es:0008 AD DE E0 01 C5 15 AA 01	:	[x] [S] [0]			
es:0010 C5 15 89 02 20 10 92 01	:	[S] [0] [A] [0]			
es:0018 01 03 01 00 02 FF FF FF		[0] [0] [0] [0]			
ss:0002 6474					
ss:0000 0000					

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

3.] Hex to BCD (Option 3)

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TD

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

cs:009E>CC	int	03	ax 0434	c=0
cs:009F 0037	add	[bx],dh	bx 000A	z=0
cs:00A1 352235	xor	ax,3522	cx 0000	s=0
cs:00A4 0000	add	[bx+si],al	dx 00B8	o=0
cs:00A6 0000	add	[bx+si],al	si 0000	p=0
cs:00A8 6F	outsw		di 0000	a=0
cs:00A9 7074	jo	011F	bp 0000	i=1
cs:00AB 696F6E2031	imul	bp,[bx+6E],31	sp 0000	d=0

cs:00B0 3A20 [.] = Dump 2=[.]

cs:00B2 41 ds:0000 37 35 22 35 00 00 00 00 75"5

cs:00B3 7363 ds:0008 6F 70 74 69 6F 6E 20 31 option 1

cs:00B5 696920 ds:0010 3A 20 41 73 63 69 69 20 : Ascii

cs:00BA 204243 ds:0018 74 6F 20 42 43 44 20 5B to BCD I

es:0000 CD 20 FF 9F 00 EA FF FF = f Ω

es:0008 AD DE E0 01 C5 15 AA 01 i |x|S-

es:0010 C5 15 89 02 20 10 92 01 +Se >ff

es:0018 01 03 01 00 02 FF FF FF 00 00

ss:0002 6474

ss:0000 0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

4.] BCD to Hex (Option 4)

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TD

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

cs:009E>CC	int	03	ax 0805	c=0
cs:009F 0037	add	[bx],dh	bx 0000	z=0
cs:00A1 352235	xor	ax,3522	cx 0004	s=0
cs:00A4 0000	add	[bx+si],al	dx 0023	o=0
cs:00A6 0023	add	[bp+di],ah	si 0000	p=0
cs:00A8 6F	outsw		di 0000	a=1
cs:00A9 7074	jo	011F	bp 0000	i=1
cs:00AB 696F6E2031	imul	bp,[bx+6E],31	sp 0000	d=0

cs:00B0 3A20 [.] = Dump 2=[.]

cs:00B2 41 ds:0000 37 35 22 35 00 00 00 23 75"5 #

cs:00B3 7363 ds:0008 6F 70 74 69 6F 6E 20 31 option 1

cs:00B5 696920 ds:0010 3A 20 41 73 63 69 69 20 : Ascii

cs:00BA 204243 ds:0018 74 6F 20 42 43 44 20 5B to BCD I

es:0000 CD 20 FF 9F 00 EA FF FF = f Ω

es:0008 AD DE E0 01 C5 15 AA 01 i |x|S-

es:0010 C5 15 89 02 20 10 92 01 +Se >ff

es:0018 01 03 01 00 02 FF FF FF 00 00

ss:0002 6474

ss:0000 0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

POSTLAB QUESTIONS:

1. Write any 10 Bit manipulation instructions with example

Ans:

These instructions are used to perform operations where data bits are involved, i.e. operations like logical, shift, etc.

Following is the list of instructions under this group –

➤ Instructions to perform logical operation

- **NOT** – Used to invert each bit of a byte or word.
- **AND** – Used for adding each bit in a byte/word with the corresponding bit in another byte/word.
- **OR** – Used to multiply each bit in a byte/word with the corresponding bit in another byte/word.
- **XOR** – Used to perform Exclusive-OR operation over each bit in a byte/word with the corresponding bit in another byte/word.
- **TEST** – Used to add operands to update flags, without affecting operands.

➤ Instructions to perform shift operations

- **SHL/SAL** – Used to shift bits of a byte/word towards left and put zero(S) in LSBs.
- **SHR** – Used to shift bits of a byte/word towards the right and put zero(S) in MSBs.
- **SAR** – Used to shift bits of a byte/word towards the right and copy the old MSB into the new MSB.

➤ Instructions to perform rotate operations

- **ROL** – Used to rotate bits of byte/word towards the left, i.e. MSB to LSB and to Carry Flag [CF].
- **ROR** – Used to rotate bits of byte/word towards the right, i.e. LSB to MSB and to Carry Flag [CF].

- **RCR** – Used to rotate bits of byte/word towards the right, i.e. LSB to CF and CF to MSB.
- **RCL** – Used to rotate bits of byte/word towards the left, i.e. MSB to CF and CF to LSB.

2. Explain 8086 in Minimum and Maximum mode.

Ans:

MINIMUM MODE	MAXIMUM MODE
1. In minimum mode there can be only one processor i.e. 8086.	1. In maximum mode there can be multiple processors with 8086, like 8087 and 8089.
2. MN/MX is 1 to indicate minimum mode.	2. MN/MX is 0 to indicate maximum mode.
3. ALE for the latch is given by 8086 as it is the only processor in the circuit.	3. ALE for the latch is given by 8288 bus controller as there can be multiple processors in the circuit.
4. DEN and DT/R for the trans-receivers are given by 8086 itself.	4. DT/R for the trans-receivers are given by 8288 bus controller.
5. Direct control signals M/IO, RD and WR are given by 8086.	5. Instead of control signals, each processor generates status signals called S2, S1 and S0.
6. Control signals M/IO, RD and WR are decoded by a 3:8 decoder like 74138	6. Status signals S2, S1 and S0 are decoded by a bus controller like 8288 to produce control signals.
7. INTA is given by 8086 in response to an interrupt on INTR line.	7. INTA is given by 8288 bus controller in response to an interrupt on INTR line.
8. HOLD and HLDA signals are used for bus request with a DMA controller like 8237.	8. RQ/GT lines are used for bus requests by other processors like 8087 or 8089.
9. The circuit is simpler.	9. The circuit is more complex.
10. Multiprocessing cannot be performed hence performance is lower.	10. As multiprocessing can be performed, it can give very high performance.