

Name: Brendan Lucas, Div: SE COMP B, Roll No: 8953

Micro Processor Practical- 4.

1. a. HEX TO BCD:

edit: C:\emu8086\MySource\hextobcd.asm

```
01 ;Name: Brendan Lucas
02 ;Roll No: 8953
03 ;Div: SE Comps B
04
05
06
07 DATA SEGMENT
08     HEX DW 0FFFFH
09     BCD DW 5 DUP(0)
10 DATA ENDS
11
12 ASSUME CS:CODE, DS:DATA
13
14 CODE SEGMENT
15     START: MOV AX, DATA
16            MOV DS, AX
17            ; initialize data
18
19            LEA SI, BCD
20            MOV AX, HEX
21
22            MOV CX, 2710H
23            CALL SUB1
24
25            MOV CX, 03E8H
26            CALL SUB1
27
28            MOV CX, 0064H
29            CALL SUB1
30
31            MOV CX, 000AH
32            CALL SUB1
33
34            MOV [SI], AL
35
36            MOV AH, 4CH
37            INT 21H
38
39 SUB1 PROC NEAR
40     MOV BH, 0FFH
41     X1: INC BH
42     SUB AX, CX
43     JNC X1
44     ADD AX, CX
45     MOV [SI], BH
46     INC SI
47     RET
48 SUB1 ENDP
49
50 CODE ENDS
51 END START
```

variables

name	value
HEX	0FFFFH
BCD	00000005:0305:05:06h

emulator: hextobcd.exe

registers

register	value
AX	0000
BX	0000
CX	0000
DX	0000
SI	0000
DI	0000
BP	0000
SP	0000
IP	0000

BIOS D1

address	value
071000	FF 255 RES
071001	05 005
071002	06 006
071003	05 005
071004	05 005
071005	03 003
071006	05 005
071007	00 000 NULL
071008	00 000 NULL
071009	00 000 NULL
07100A	00 000 NULL
07100B	00 000 NULL
07100C	00 000 NULL
07100D	00 000 NULL
07100E	00 000 NULL
07100F	00 000 NULL
071010	00 000 NULL
071011	00 000 NULL
071012	07 007
071013	0E 142
071014	08 216
071015	05 005
071016	02 002
071017	00 000 NULL
071018	01 161
071019	00 000 NULL
07101A	00 000 NULL
07101B	09 185

1. b. BCD TO HEX:

edit: C:\emu8086\MySource\bcdtohex.asm

```
01 ;Name: Brendan Lucas
02 ;Roll No: 8953
03 ;Div: SE Comps B
04
05
06
07 DATA SEGMENT
08     BCD DB 06H, 05H, 05H, 03H, 05H
09     HEX DW 7
10 DATA ENDS
11
12 ASSUME CS:CODE, DS:DATA
13
14 CODE SEGMENT
15     START: MOV AX, DATA
16            MOV DS, AX
17            MOV CL, 05H
18            MOV BP, 000AH
19            MOV AX, 2710H
20            PUSH AX
21            MOV DI, 0000H
22            MOV SI, OFFSET BCD
23
24 X: MOV BL, [SI]
25    MUL BX
26    ADD DI, AX
27    POP AX
28    DIV BP
29    PUSH AX
30    INC SI
31    LOOP X
32    MOV HEX, DI
33
34    MOV AH, 4CH
35    INT 21H
36
37 CODE ENDS
38 END START
```

variables

name	value
BCD	00000005:0305:05:06h
HEX	0FFFFH

emulator: bcdtohex.exe

registers

register	value
AX	0000
BX	0000
CX	0000
DX	0000
SI	0000
DI	0000
BP	0000
SP	0000
IP	0000

BIOS D1

address	value
071000	06 006
071001	05 005
071002	05 005
071003	03 003
071004	05 005
071005	FF 255 RES
071006	05 005
071007	00 000 NULL
071008	00 000 NULL
071009	00 000 NULL
07100A	00 000 NULL
07100B	00 000 NULL
07100C	00 000 NULL
07100D	00 000 NULL
07100E	00 000 NULL
07100F	00 000 NULL
071010	00 000 NULL
071011	00 000 NULL
071012	07 007
071013	0E 142
071014	08 216
071015	05 005
071016	02 002
071017	00 000 NULL
071018	01 161
071019	00 000 NULL
07101A	00 000 NULL
07101B	09 185

edit: C:\emu8086\MySource\gcdsoftwonos.asm

3. Write a Program for addition of numbers taken from keyboard and display result on the screen

The image shows a screenshot of an x86 assembly editor and emulator. The editor window displays the source code for a program that adds two numbers entered from the keyboard and displays the result. The program is written in assembly language and includes comments in English. The emulator window shows the program running, with the input and output displayed on the screen.

Source Code:

```
0001 ;Name: Brendan Lucas
0002 ;Roll No: 8953
0003 ;Div: SE Comps B
0004
0005 .8086
0006 .model small
0007
0008 .data
0009     msg1 db 10,13,"Enter the no-$"
0010     sps db 10,13,"$"
0011     msg2 db "Grand Total is: $"
0012     msg3 db 10,13,10,13,"Grand Total is: $"
0013     arrn db 06 dup(00h)
0014     len db 06h
0015     gt du 0000h
0016
0017 .code
0018 start:
0019     mov ax,@data
0020     mov ds,ax
0021     mov ah,09h
0022     lea dx,msg1
0023     int 21h
0024     lea si,arrn
0025     mov cl,len
0026     mov h1,01h
0027
0028 bk1:
0029     mov ah,09h
0030     lea dx,sps
0031     int 21h
0032     mov ah,02h
0033     mov d1,h1
0034     add d1,30h
0035     int 21h
0036     inc h1
0037     mov ah,09h
0038     lea dx,msg2
0039     int 21h
0040     call getno
0041     mov d1,al
0042     inc si
0043     dec cl
0044     jnz bk1
0045
0046 calc:
0047     lea si,arrn
0048     mov cl,len
0049     mov ax,gt
0050
0051 bk2:
0052     add al,[si]
0053     daa
0054     jnc skp2
0055     mov bh,al
0056     mov al,ah
0057     add al,01h
0058     dec
0059
0060     proc getno
0061     push cx
0062     mov ah,01h
0063     int 21h
0064     mov cx,ax
0065     sub al,30h
0066     cmp al,09h
0067     jle g1
0068     sub al,07h
0069     g1:
0070     mov cl,04h
0071     rol al,cl
0072     mov ch,al
0073     mov ah,01h
0074     int 21h
0075     sub al,30h
0076     cmp al,09h
0077     jle g2
0078     sub al,07h
0079     g2:
0080     add al,ch
0081     pop cx
0082     ret
0083
0084 endp getno
0085
0086 proc display
0087     push cx
0088     mov al,[si]
0089     and al,0f0h
0090     mov cl,04h
0091     rol al,cl
0092     add al,30h
0093     cmp al,39h
0094     jle p1
0095     add al,07h
0096     p1:
0097     mov ah,02h
0098     mov d1,al
0099     int 21h
0100     mov al,[si]
0101     and al,0f0h
0102     add al,30h
0103     cmp al,39h
0104     jle p2
0105     add al,07h
0106     p2:
0107     mov ah,02h
0108     mov d1,al
0109     int 21h
0110     pop cx
0111     ret
0112
0113 endp display
0114 end start
```

Emulator Output:

```
Enter the no=
1:05
2:05
3:05
4:05
5:05
6:05

Grand Total is: 30
```