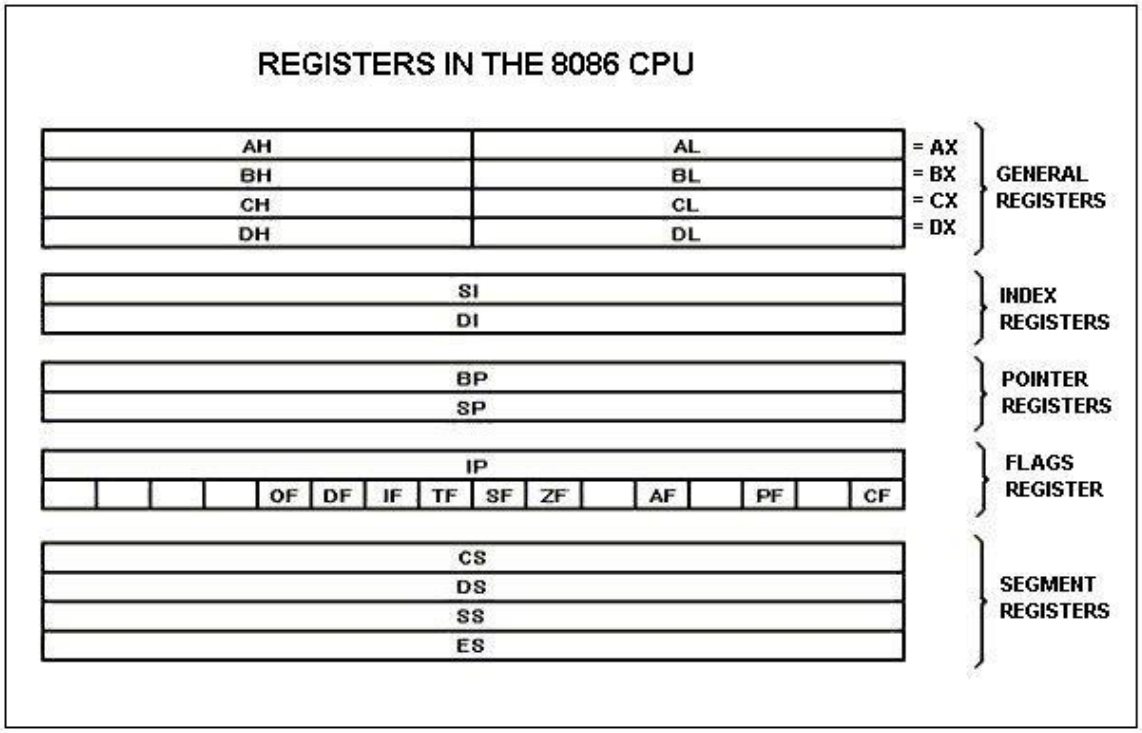
Experiment 03

Learning Objective: Student should be able to Convert HEX to BCD and BCD to HEX using stack in ALP.

Tools: TASM/MASM

Theory:

Software Architecture / Register Set/ Programmer's model of Intel 8086 Microprocessor:



GENERAL PURPOSE REGISTERS

8086 CPU has 8 general purpose registers, each register has its own name:

AX - the accumulator register (divided into AH / AL):

1. Generates shortest machine code
2. Arithmetic, logic and data transfer
3. One number must be in AL or AX
4. Multiplication & Division
5. Input & Output

BX - the base address register (divided into BH / BL).

CX - the count register (divided into CH / CL):

1. Iterative code segments using the LOOP instruction
2. Repetitive operations on strings with the REP command
3. Count (in CL) of bits to shift and rotate

DX - the data register (divided into DH / DL):

1. DX:AX concatenated into 32-bit register for some MUL and DIV operations
2. Specifying ports in some IN and OUT operations

SI - source index register:

1. Can be used for pointer addressing of data
2. Used as source in some string processing instructions
3. Offset address relative to DS

DI - destination index register:

1. Can be used for pointer addressing of data
2. Used as destination in some string processing instructions
3. Offset address relative to ES

BP - base pointer:

1. Primarily used to access parameters passed via the stack
2. Offset address relative to SS

SP - stack pointer:

1. Always points to top item on the stack
2. Offset address relative to SS
3. Always points to word (byte at even address)
4. An empty stack will had SP = FFFEh

SEGMENT REGISTERS

CS - points at the segment containing the current program.

DS - generally points at segment where variables are defined.

ES - extra segment register, it's up to a coder to define its usage.

SS - points at the segment containing the stack.

Flag Register of 8086:

- A flag is a flip-flop which indicates some condition produced by the execution of an instruction or controls certain operations of the EU.
- The Flag Register is a special register associated with the ALU.
- A 16-bit flag register in the EU contains nine active flags.
- Fig. shows the location of the nine flags in the flag register

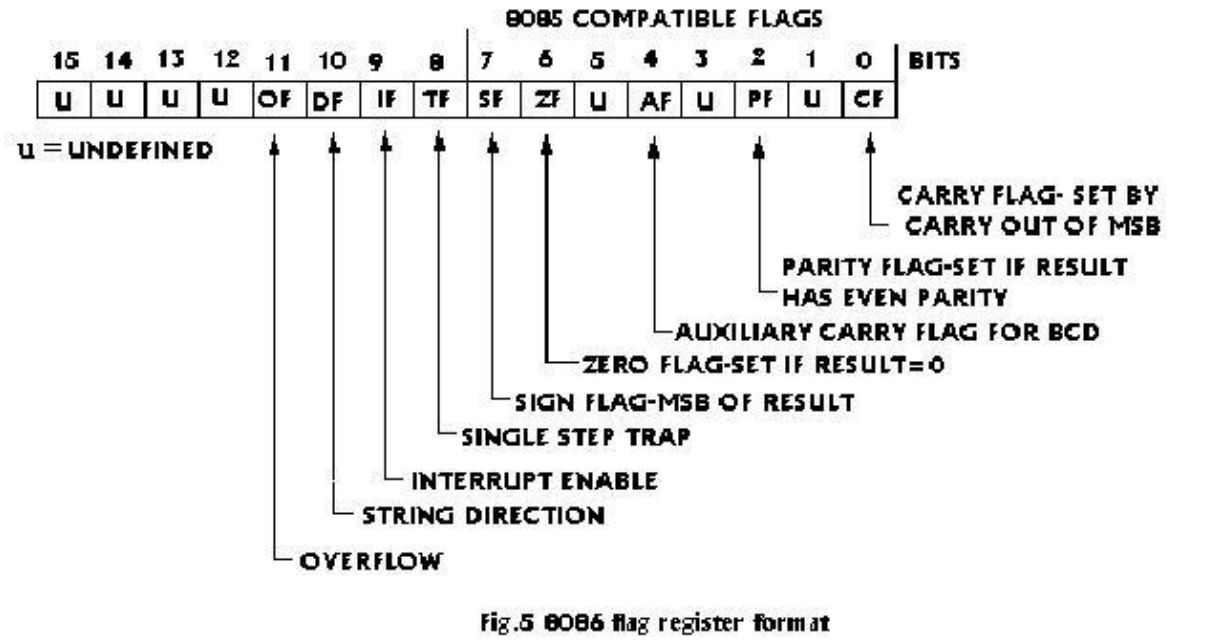


Fig.5 8086 flag register format

Fig. 1.4 Flag Register structure

Flags is a 16-bit register containing 9 1-bit flags:

- Overflow Flag (OF) - set if the result is too large positive number, or is too small negative number to fit into destination operand.
- Direction Flag (DF) - if set then string manipulation instructions will auto-decrement index registers. If cleared then the index registers will be auto-incremented.
- Interrupt-enable Flag (IF) - setting this bit enables maskable interrupts.
- Single-step Flag (TF) - if set then single-step interrupt will occur after the next instruction.
- Sign Flag (SF) - set if the most significant bit of the result is set.
- Zero Flag (ZF) - set if the result is zero.
- Auxiliary carry Flag (AF) - set if there was a carry from or borrow to bits 0-3 in

the AL register.

- Parity Flag (PF) - set if parity (the number of "1" bits) in the low-order byte of the result is even.
- Carry Flag (CF) - set if there was a carry from or borrow to the most significant bit during last result calculation.

Procedure to Convert 4 digit Hex number to its equivalent BCD number.

We have a 4 digit Hex number whose equivalent binary number is to be found.i.e. FFFF H. Initially we compare FFFF H with decimal 10000 (2710 H in Hex). If number is greater than 10,000 we add it to DH register. Also, we subtract decimal 10,000 from FFFF H, each time comparison is made. Then we compare the number obtained in AX by 1000 decimal. Each time we subtract 1000 decimal from AX and add 1000 decimal to BX. Then we compare number obtained in AX by 100 decimals. Each time we subtract 100 decimal from AX and add 100 decimal to BX to obtain BCD equivalent. Then we compare number obtained in AX with 10 decimal. Each time we subtract 10 decimal from AX and we add 10 decimal to BX. Finally we add the result in BX with remainder in AX. The final result is present in register DH with contains the 5th bit if present and register AX. Display the result.

Algorithm:

Step I: Initialize the data segment.

Step II: Initialize BX = 0000 H and DH = 00H.

Step III: Load the number in AX.

Step IV: Compare number with 10000 decimal. If below goto step VII else goto

Step V.

Step V: Subtract 10,000 decimal from AX and add 1 decimal to DH

Step VI: Jump to step IV.

Step VII: Compare number in AX with 1000, if below goto step X else goto

Step VIII.

Step VIII: Subtract 1000 decimal from AX and add 1000 decimal to BX.

Step IX: Jump to step VII.

Step X: Compare the number in AX with 100 decimal if below goto step XIII

Step XI: Subtract 100 decimal from AX and add 100 decimal to BX.

Step XII: Jump to step X

Step XIII: Compare number in AX with 10. If below goto step XVI

Step XIV: Subtract 10 decimal from AX and add 10 decimal to BX.

Program

```
.model small

disp macro a
mov ah,09h
lea dx,a
int 21h
endm


.data
msg1 db 10,13,"1.Hex to BCD.$"
msg2 db 10,13,"2.BCD to Hex.$"
msg3 db 10,13,"3.Exit.$"
msg4 db 10,13,"Enter your choice:$"
msg5 db 10,13,"Invalid choice.Re-enter choice.$"
msg6 db 10,13,"$"
msg7 db 10,13,"Enter the HEX no:$"
msg8 db 10,13,"The corresponding BCD no is:$"
msg9 db 10,13,"Enter the BCD no:$"
msg10 db 10,13,"The corresponding HEX no is:$"
result dw 0000h
temp dw 000Ah
cnt db 00h

.code

main:
mov ax,@data
mov ds,ax
```

```
menu:
disp msg6
disp msg1
disp msg2
disp msg3
disp msg4
mov ah,01h
int 21h
mov bl,al
sub bl,30h
cmp bl,01
je h2b
cmp bl,02
je b2h
cmp bl,03
je exit
disp msg5
jmp menu

exit:
mov ah,4ch
int 21h

h2b:
disp msg7
mov ax,0000h
call accept
disp msg8
mov ax,bx
```

```
up1:
mov dx,0000
div temp
push dx
inc cnt
cmp ax,0000h
jnz up1

up2:
pop dx
add dl,30h
mov ah,02h
int 21h
dec cnt
jnz up2

jmp menu

b2h:
disp msg9
mov bx,0000h
mov cl,05h

up3:
mov ax,000Ah
mul bx
mov bx,ax
mov ah,01h
int 21h
sub al,30h
mov ah,00h
```

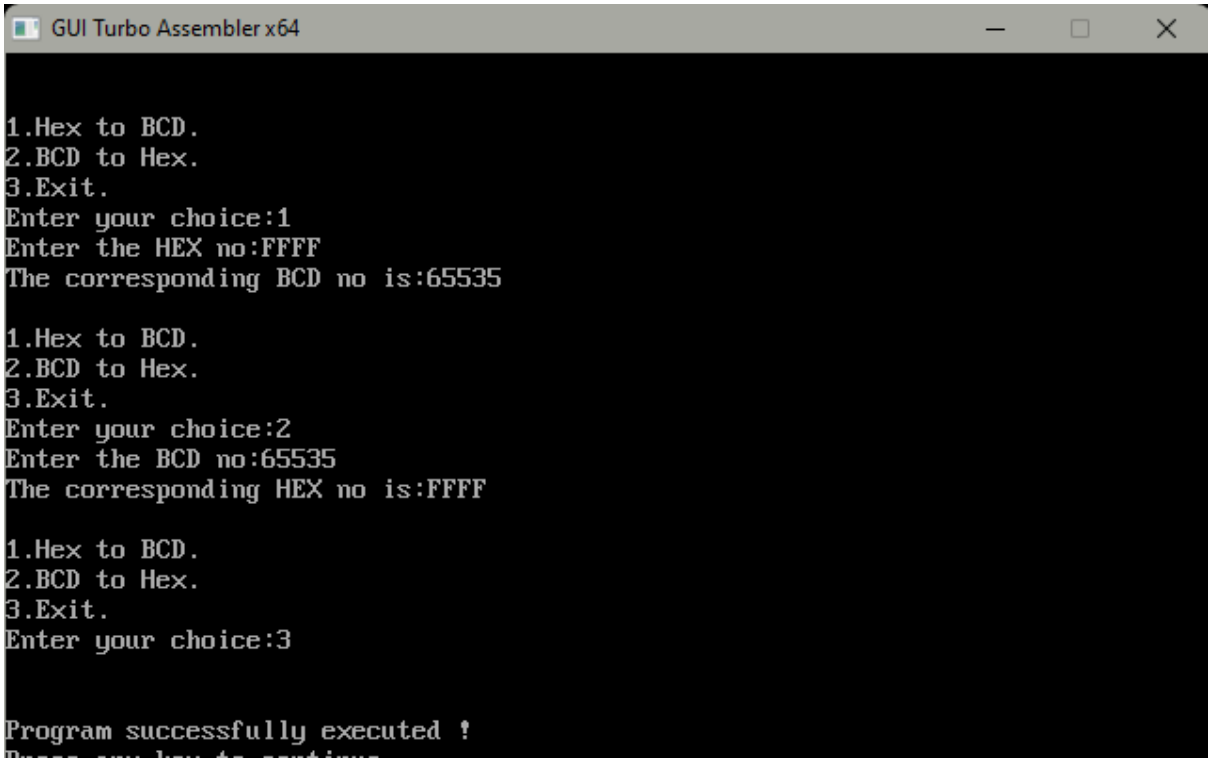
```
add bx,ax
dec cl
jnz up3
mov result,bx
disp msg10
call print

jmp menu

accept proc near

mov ah,01h
int 21h
sub al,30h
cmp al,09h
jle l1
sub al,07h

l1:
mov ah,00
shl ax,12
mov bx,ax

mov ah,01h
int 21h
sub al,30h
cmp al,09h
jle l2
sub al,07h

l2:
```

```
mov ah,00
shl ax,08
add bx,ax

mov ah,01h
int 21h
sub al,30h
cmp al,09h
jle l3
sub al,07h

l3:
mov ah,00
shl ax,04
add bx,ax

mov ah,01h
int 21h
sub al,30h
cmp al,09h
jle l4
sub al,07h

l4:
mov ah,00
add bx,ax

ret
accept endp

print proc near
```

```
mov bx,result
shr bx,0Ch
cmp bl,09h
jle l5
add bl,07h

l5:
add bl,30h
mov dl,bl
mov ah,02h
int 21h

mov bx,result
shr bx,08h
and bl,0Fh
cmp bl,09h
jle l6
add bl,07h

l6:
add bl,30h
mov dl,bl
mov ah,02h
int 21h

mov bx,result
shr bx,04h
and bl,0Fh
cmp bl,09h
jle l7
```

```
add bl,07h

l7:
add bl,30h
mov dl,bl
mov ah,02h
int 21h

mov bx,result
and bl,0Fh
cmp bl,09h
jle l8
add bl,07h

l8:
add bl,30h
mov dl,bl
mov ah,02h
int 21h

ret
print endp

end main
end
```

Output


GUI Turbo Assembler x64
1.Hex to BCD.
2.BCD to Hex.
3.Exit.
Enter your choice:1
Enter the HEX no:FFFF
The corresponding BCD no is:65535
1.Hex to BCD.
2.BCD to Hex.
3.Exit.
Enter your choice:2
Enter the BCD no:65535
The corresponding HEX no is:FFFF
1.Hex to BCD.
2.BCD to Hex.
3.Exit.
Enter your choice:3
Program successfully executed !

Result and Discussion:

PUSH and POP instructions were used here and the concept was understood . BCD to HEX and vice versa was done using Stack.

Learning Outcomes: The student should have the ability to

LO1: Draw and explain the format of PUSH and POP instructions.
LO2: Explain the concept of Number systems.
LO3: Apply stack instructions to convert HEX to BCD and BCD to HEX.

Course Outcomes: Upon completion of the course students will be able to make use of instructions of 8086 to build assembly and Mixed language programs.

Conclusion: Students were able to make use of instructions of 8086 to build assembly and Mixed language programs.

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				