**Experiment 2**

Logical operations:

AND:

org 100h

mov ax, 21h

And Ax, 12h

hlt

OR:

org 100h

mov ax, 1221h

or Ax, 0f0fh

hlt

NEG:

org 100h

mov ax, 0000h

mov bx, 0000h

mov cx, 0000h

mov dx, 0000h

mov ax, 0021h

mov bx, 0031h

neg ax

NOT:

org 100h

mov ax, 0000h

mov bx, 0000h

mov cx, 0000h

mov dx, 0000h

mov ax, 0021h

mov bx, 0031h

not ax

XOR:

org 100h

mov ax, 0021h

mov bx, 0031h

xor ax, bx

mov ah, 4Ch

int 21h

**Experiment 3:**

Arithmetic operations:

ADD:  
org 100h

mov ax, 06h

mov bx, 03h

add ax,bx

SUB:

org 100h

mov ax, 09h

mov bx, 04h

sub ax,bx

MUL:

org 100h

mov ax, 1100h

mov bx, 1102h

mul, bx

DIV:  
org 100h

mov al, 09h

mov bl, 03h

div bl

**Experiment 4:**

16 bit Addition:

ORG 100H

MOV SI, 1100H

MOV CX, 00H

MOV AX, [SI]

MOV BX, [SI+2]

ADD AL, BL

DAA

MOV DL, AL

ADC AH,BH

DAA

MOV DH,AH

JNC JUMP

INC CL

JUMP:

MOV [SI+4], DX

MOV [SI+6], CL

HLT

**Experiment 5:**

Memory Exchange:

ORG 100H

MOV SI, 1100H

MOV DI, 2200H

MOV CX, 05H

LOOP:

MOV AX, [SI]

MOV [DI], AX

INC SI

INC DI

DEC CX

JNZ LOOP

HLT

**Experiment 6:**

EVEN & ODD Numbers:

MOV SI,2200H

MOV BH,00H

MOV BL,00H

MOV CL,[SI]

INC SI

UP: MOV AL, [SI]

ROR AL,01H

JC LOOP

INC BL

JMP NEXT

LOOP:

INC BH

NEXT:

INC SI

DEC CL

JNZ UP

MOV [SI +1], BL

MOV [SI +2], BH

HLT

**Experiment 7:**

Fibonacci Series:

MOV AL, 00H

MOV SI, 1100H

MOV [SI], AL

ADD SI, 01H

ADD AL, 01H

MOV [SI], AL

MOV CX, [0000H]

SUB CX, 0002H

L1:

MOV AL, [SI-1]

ADD AL, [SI]

ADD SI, 01H

MOV [SI], AL

LOOP L1

HLT

**Experiment 8:**

Search for a number in the list:

.model small

.data

Numbers db 10, 20, 30, 40, 50

count db 5

key db 30

foundMsg db &#39

notFoundMsg db &#39

.code

main:

mov ax, @data

mov ds, ax

mov cl, count

mov si, 0

search\_loop:

mov al, numbers[si]

cmp al, key

je found

inc si

dec cl

jnz search\_loop

not\_found:

mov dx, offset notFoundMsg

mov ah, 09h

int 21h

mov ah, 4Ch

int 21h

found:

mov dx, offset foundMsg

mov ah, 09h

int 21h

mov ah, 4Ch

int 21h

ret

**Experiment 9:**

To display 8 bit flag registers:

Data Segment

msg db 0dh,0ah

newline db 0dh,0ah

flag dw ?

data ends

code segment

assume CS: Code, DS: Data

start:

MOV AX, Data

MOV DS, AX

MOV DX, offset msg

MOV AH, 09h

int 21h

MOV DX, offset newline

MOV AH,09h

int 21h

CLI

STC

STD

PUSHF

POP BX

MOV FLAG, BX

MOV CX, 16

MOV BX, 8000H

AGAIN:

MOV AX, FLAG

AND AX, BX

JZ ZERO

MOV DL, 31H

MOV AH, 02h

INT 21h

JMP SPACE

ZERO:

MOV DL, 30h

MOV AH, 02H

INT 21H

SPACE:

MOV DL, &#39

MOV AH, 02H

INT 21H

MOV AH, 02H

INT 21H

ROR BX, 1

LOOP AGAIN