Addition of 2 16 bit numbers

MOV SI,5000 //starting address of stored data

MOV DI,6000 //destination address

Mov AX,[SI]

INC SI

INC SI

MOV BX,[SI]

ADD AX,BX

MOV [DI],AX

INT 03 //Display using Interrupt

Subtraction of 2 32 bit numbers

MOV SI,5000

MOV DI,6000

MOV AX,[SI]

INC SI

INC SI

MOV BX,[SI]

SUB AX,BX

MOV [DI],AX

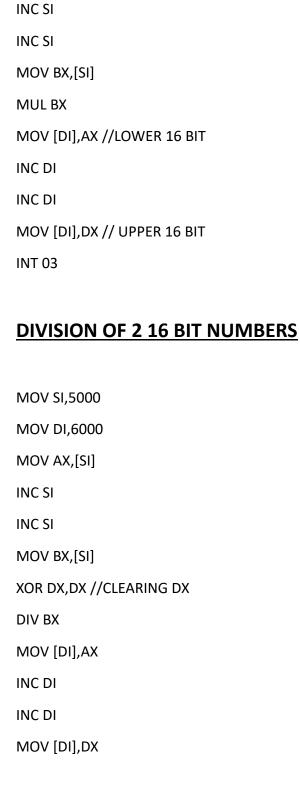
INT 03

MULTIPLICATION OF 16 BIT NUMBERS

MOV SI,5000

MOV DI,6000

MOV AX,[SI]



Array sorting of 16 bit numbers

STORE THE NUMBER OF NUMBEERS INTO 5000

```
MOV SI,5000
```

MOV CL,[SI]

DEC CL

Outer loop → MOV SI,5000

MOV CH,[SI]

DEC CH

INC SI

Inner loop-→ MOV AX,[SI]

INC SI

INC SI

CMP AX,[SI]

JC loop SKIPPING

XCHG AX,[SI]

DEC SI

DEC SI

XCHG AX,[SI]

INC SI

INC SI

LOOP SKIPPING→ DEC CH

JNZ //address of inner loop

DEC CL

JNZ //address of outer loop

INT 03

SEARCHING OF 16 BIT NUMBERS IN AN ARRAY

MOV SI,5000 //LENGTH STORED AT THIS ADDRESS
MOV CL,[SI]
MOV DI,6000 //KEY TO BE SEARCHED
MOV BX,[DI]
INC DI
INC DI
INC SI
INC SI
LOOP→ MOV AX,[SI]
CMP AX,BX
JE FOUND
INC SI
INC SI
DEC CL
JNZ LOOP
JMP NOT FOUND

FOUND: MOV DX,0FFFF

JMP STORE RESULT

NOT FOUND: MOV DX,0000

STORE RESULT: MOV [DI],DX

INT 03

INTERFACING

Interfacing with digital to analog coverter

MOV AL, 80h ; Load a sample value (80h) into AL

MOV DX, 0FFE6h ; Load port address 0FFE6h into DX

OUT DX, AL ; Output the value in AL to the DAC

LOOP_START:

MOV AL, 0FEh ; Load another value (0FEh) into AL to output

OUT DX, AL ; Output the value in AL to the DAC

MOV CX, 0FFFFh ; Set up a delay counter

DELAY:

NOP ; No operation (used to create delay)

NOP ; Additional delay

LOOP DELAY ; Decrement CX and loop until CX = 0

MOV AL, 00h ; Load 00h to reset the DAC

OUT DX, AL ; Output the reset value

MOV CX, OFFFFh ; Set up another delay counter

DELAY2:

NOP ; No operation (additional delay)

NOP ; Additional delay

LOOP DELAY2 ; Decrement CX and loop until CX = 0

JMP LOOP_START ; Repeat the loop

STEPPER MOTOR

MOV DX, 0FFE6h ; Port address for control signals

MOV AL, 80h ; Initial value for the stepper motor control

OUT DX, AL ; Send the initial control signal to the motor

MOV DX, 0FFE0h ; Port address for motor data

MOV CX, 400 ; Set up loop counter for the number of steps

Step_Loop:

MOV AL, 88h ; Load initial pattern to start stepping

OUT DX, AL ; Send the control signal to the motor

CALL Rotate_Signal; Call subroutine to rotate signal pattern

DEC CX ; Decrement step counter

JNZ Step_Loop ; Repeat if CX is not zero

JMP End_Program ; Jump to end program label

Rotate_Signal:

ROR AL, 1; Rotate AL right by 1 bit for the next step

OUT DX, AL ; Output the rotated pattern to the motor

RET ; Return from subroutine

End_Program:

; Program end. Add any necessary clean-up or HALT here.