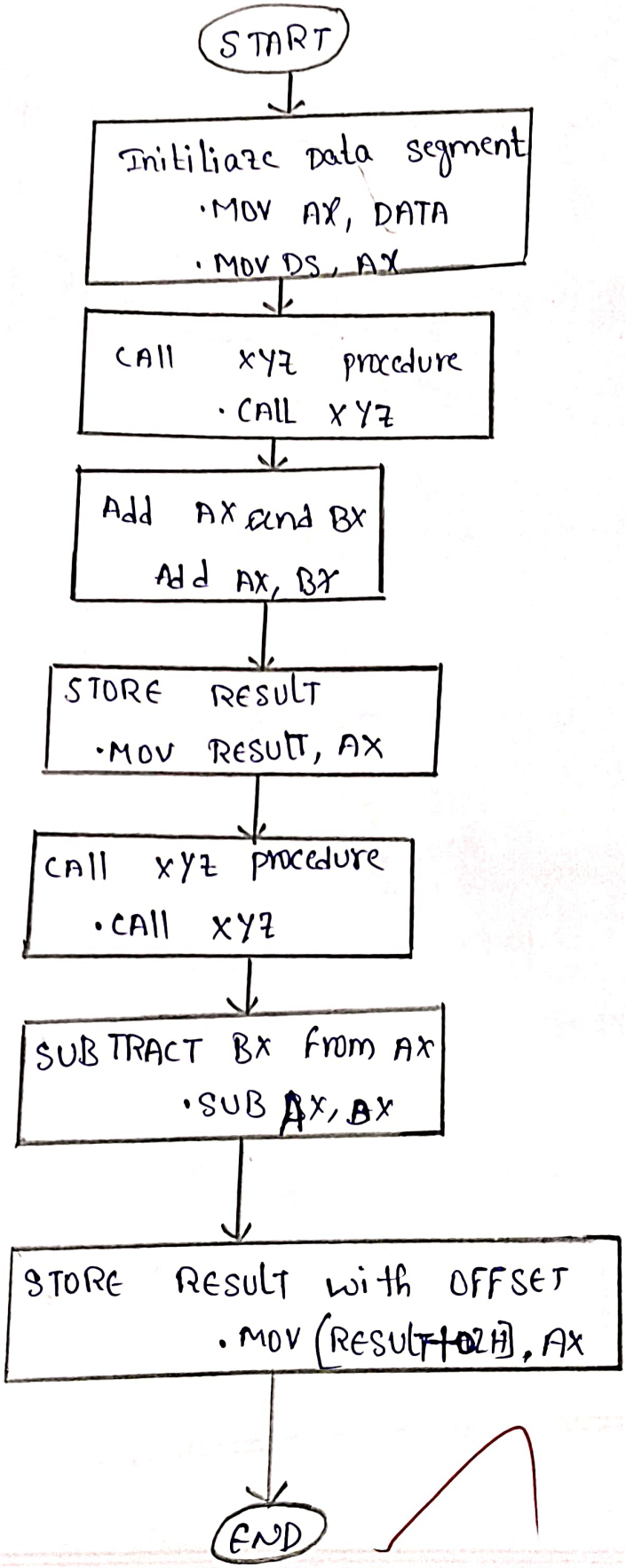
Flow chart:-


START
Initiliaze data segment
·MOV AX, DATA
·MOV DS, AX
CALL XYZ procedure
·CALL XYZ
Add AX and BX
Add AX, BX
STORE RESULT
·MOV RESULT, AX
call xyz procedure
·CALL XYZ
SUBTRACT BX from AX
·SUB AX, BX
STORE RESULT with OFFSET
·MOV (RESULT+02H), AX
END

13/08/25

Experiment - 5

Aim:- a) write an assembly language program for implementation of near procedure.

Apparatus:-

1.) Pc with windows.
2.) MASM Software

Program:-

```
ASSUME CS:CODE, DS:DATA
DATA SEGMENT                    ; Start of data segment
N1 DB 56H, 12H                  ; Two numbers: 56H
                                              12H
RESULTS DB ?                    ; Reserve a byte for the
                                  result.
DATA ENDS                       ; END of the data segment

CODE SEGMENT                    ; Start of CODE segment

START: MOV AX, DATA             ; Load base address of data
       MOV DS, DATA                segment to AX
                                ; Move AX to DS register

       MOV SI, OFFSETN1         ; SI points to the address of
                                  N1
       MOV DI, OFFSET RESULT    ; DI points to RESULT address
       CALL LOAD                ; call LOAD Again for Second operation
       SUB AL, BL               ; subtract BL from AL
       MOV 01H[DI], AL          ; store result at Result+1
       HLT                      ; stop execution
```

```
LOAD PROC                    ; start of LOAD procedure
        MOV AL, [SI]         ; Move data at SI to AL (first number)
        MOV BL, 01H[SI]      ; Move next data at SI+1 to BL
                                (second number).
        RET                  ; Return from Procedure.
LOAD ENDP                    ; End of LOAD procedure
  CODE ENDS                  ; End of CODE SEGMENT
  END START                  ; END of program, entry point
                               START
  END
```

RESULT :-

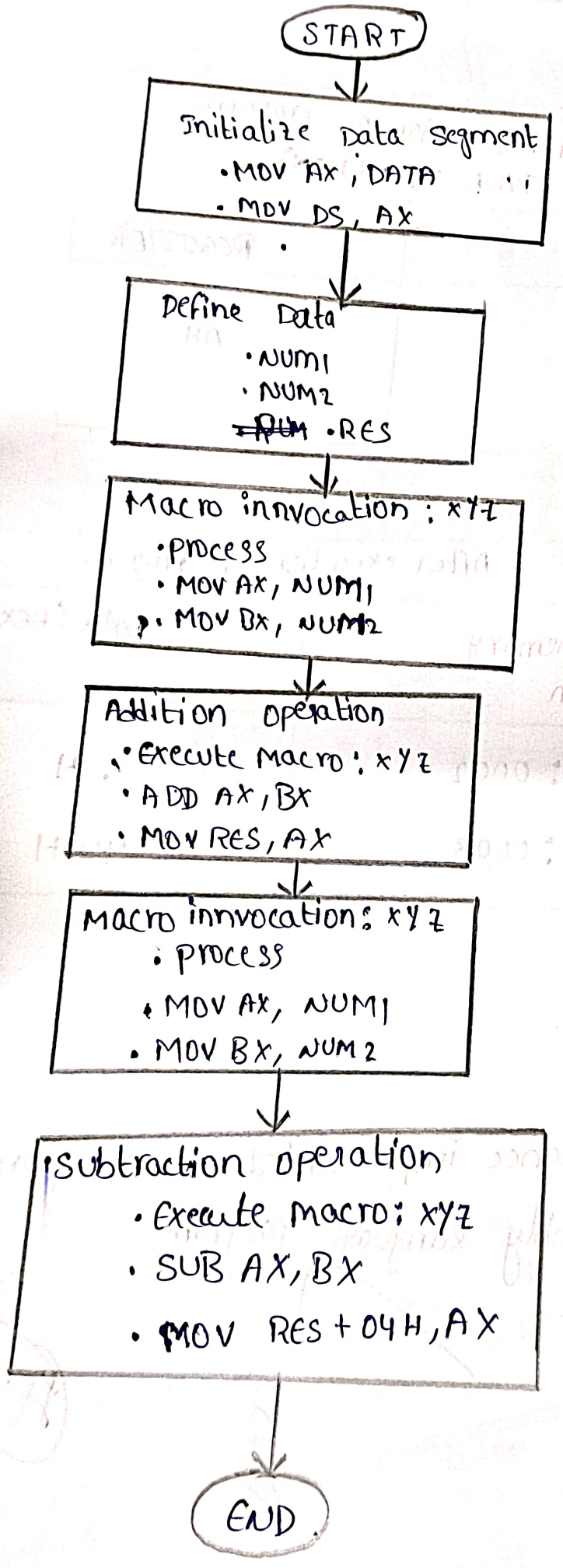
Before execution of program DATA SET-1 (HEX).	
DATA	REGISTER
56H	AH
12H	BH

After execution of program	
In Destination memory location	Data (HEX)
076A : 0002	68H
076A : 0003	44 H

Conclusion :-

Hence implemented a near procedure using Assembly language program.

Flow chart :-


START
Initialize Data Segment
•MOV AX, DATA
•MOV DS, AX
Define Data
•NUM1
•NUM2
•RES
Macro invocation : XYZ
•process
•MOV AX, NUM1
•MOV Bx, NUM2
Addition operation
•Execute Macro : xyz
•ADD AX, BX
•MOV RES, AX
Macro invocation : XYZ
•process
•MOV AX, NUM1
•MOV BX, NUM2
Subtraction operation
•Execute Macro : xyz
•SUB AX, BX
•MOV RES+04H, AX
END

20/08/25

Experiment - 6

Aim:- write an Assembly language program for implementation of MACRO.

Apparatus:-

1.) PC with windows
2.) MASM Software

Program:-

```
ASSUME CS:CODE, DS:DATA ;
DATA SEGMENT                ; START Of data segment
N1 DB 56H, 12H              ; Two numbers: 56H and 12H stored in memory.
RESULT DB ?                 ; Reserve space for storing element
DATA ENDS                   ; End of data segment
CODE SEGMENT                ; start of CODE segment
START: MOV AX, DATA         ; Load base address of data segment
       MOV DS, AX           ; set DS register to point to dat segment.
       MOV SI, OFFSET N1    ; SI points to first number (N1)
       MOV DI, OFFSET RESULT ; DI points to result storage location.

XYZ MACRO                   ; start of xyz macro definition
    MOV AL, [SI]            ; Load first number into AL register
    MOV BL, 01H[SI]         ; load second number into BL register
ENDM                        ; End of Macro de finition
    XYZ                     ; call xyz macro to load numbers
    ADD AL, BL              ; Add AL and BL, result stored in AL
    MOV [DI], AL            ; store Addition result in RESULT.
```

```
xyz                     ; call xyz macro again to reload numbers
SUB AL, BL              ; subtract BL from AL.
MOV 01H[DI], AL         ; store subtraction result in RESULT+1

HLT                     ; stop program execution
CODE ENDS               ; End of code segment
END START               ; End of program, entry point is START

END
```

RESULT:-

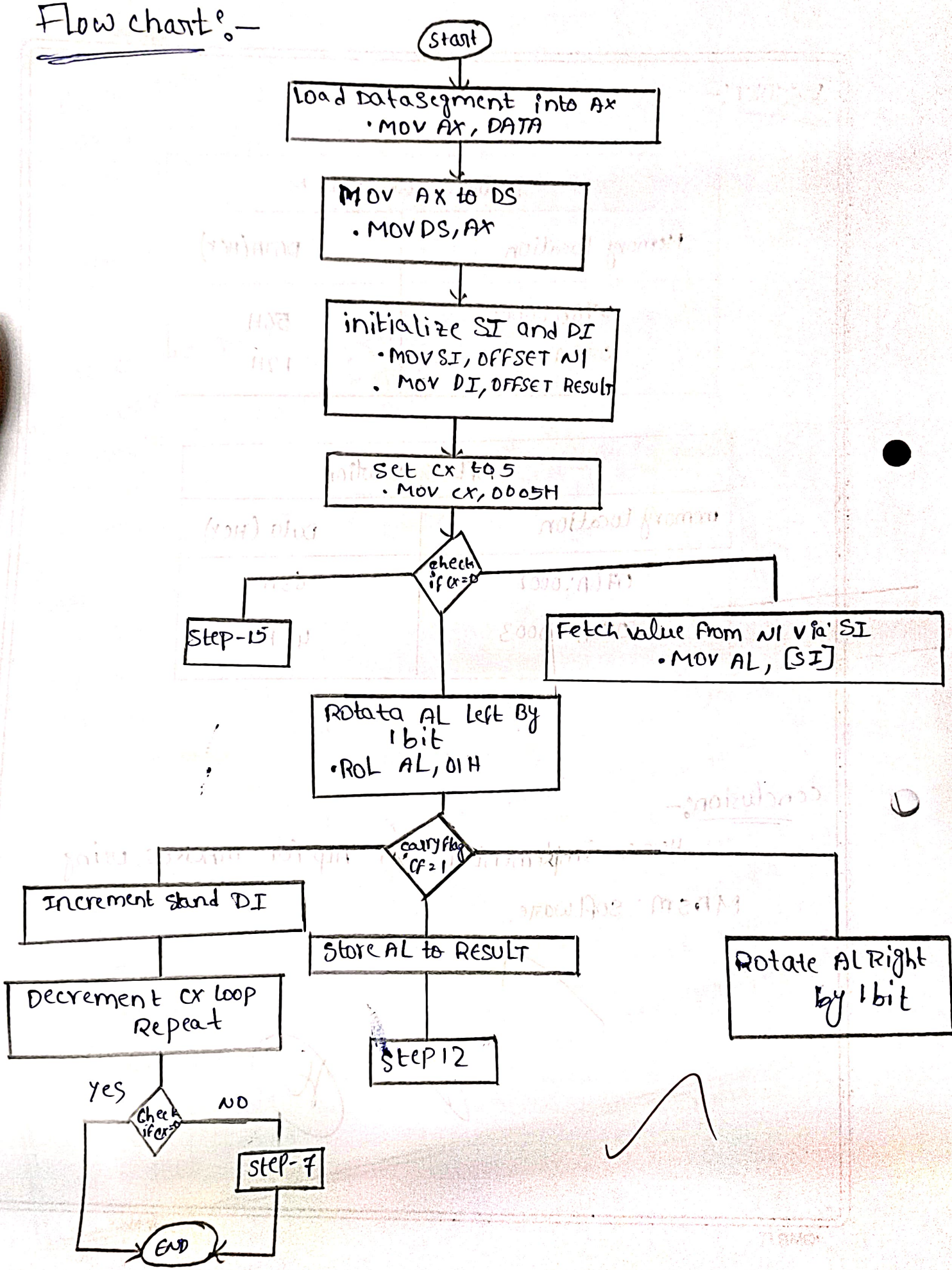
BEFORE EXECUTION	
Memory Location	DATA (HEX)
076A:0000	56H
076A:0001	12H

After execution	
memory location	Data (HEX)
076A:0002	68H
076A:0003	44H

conclusion:-

Hence implemented an ALP for MACROS, using MASM software.

Flow chart:-


Start
Load Datasegment into Ax
• MOV AX, DATA
MOV AX to DS
• MOV DS, AX
initialize SI and DI
• MOV SI, OFFSET N1
• MOV DI, OFFSET Result
Set CX to 05
• MOV CX, 0005H
check if CX=0
Step-15
Fetch value from N1 via SI
• MOV AL, [SI]
Rotate AL Left By 1 bit
• ROL AL, 01H
carry flag CF=1
Increment stand DI
Store AL to RESULT
Rotate AL Right by 1 bit
Decrement CX Loop Repeat
Step 12
yes
Check if CX=0
NO
step-7
END

20/08/25

Experiment-7

Aim:- write an Alp of 8086 microprocessor to reject negative numbers from a series of five bytes and store the positive numbers in memory.

Apparatus:-

1.) PC with windows
2.) MASM software

Program:-

```
ASSUME CS:CODE , DS:DATA
DATA SEGMENT                    ; START of Data segment
N1 DB 34H, 40H, 85H, 0A3H, 70H  ; Five numbers in memory
RESULT DB?                      ; space for storing positive results
DATA ENDS                       ; End of data segment
CODE SEGMENT                    ; Start of code segment
START: MOV AX, DATA             ; Load base address of data segment
MOV DS,AX                       ; set DS register to point to data segment
MOV SI, OFFSET N1               ; SI points to the input array N1
MOV DI, OFFSET RESULT           ; set DI points to result storage
                                  Location
MOV CX, 0005H                   ; CX is set to Repeat 5 times
                                  (for each byte).
REPEAT: MOV AL, [SI]            ; Load current value from N1 into AL,
ROL AL, 01H                     ; Rotate AL left by 1 bit (check sign bit)
JC REJECT                       ; store positive number if carry is
                                  set, number is negative. jump to
                                  Reject
ROR AL, 01H                     ; Rotate AL right back to original
                                  (Restore value)
MOV [DI], AL                    ; store positive number into Result
INC DI                          ; MOV to next result position.
REJECT: INC SI                  ; MOV to next input number
LOOP REPEAT                     ; Repeat until cx is zero (All numbers
                                  checked).
```

```
HLT                    ; stop program execution

CODE ENDS              ; End of CODE segment

END START              ; End of program, entry point is
                         START.

END
```

RESULT:-

BEFORE execution	
Memory Location	DATA(HEX)
0000 H	34H
0001 H	40H
0002H	85H
0003 H	0A3H
0004 H	70H

After Execution

After execution	
memory location	DATA HEX
0003 H	34H
0006 H	40 H
0007 H	70H

conclusion:-

Hence verification for an ALP of 8086 microprocessor negative numbers from a series of the bytes and store the positive numbers in memory.