(Do not include this page in your Report, go through next page)

Programming Language: Assembly Language

Lab days: As informed by lecturer

Full Marks: 20

Report: 5

Attendance: 5

Project: 10

Report must include:

1. Related Theory
2. Problem explanation with algorithms
3. Program code
4. Output with sample Inputs
5. Conclusion

Note:

- Report must be submitted in A4 size white paper written in both side, leaving proper margins in each side and with proper plastic binding within given date.
- Front cover must be in printed form with college logo, College name, University name, subject name etc. with smart design.
- Report must be signed regularly to the supervisor on every next lab.
- There will be no excuse for those who don't follow the above rules for any reasons.

INDEX

LAB 1: Introduction to 8085 Microprocessor and its architecture.

8085 is pronounced as "eighty-eighty-five" microprocessor. It is an 8-bit microprocessor designed by Intel in 1977 using NMOS technology.

It has the following configuration –

- 8-bit data bus
- 16-bit address bus, which can address up to 64KB
- A 16-bit program counter
- A 16-bit stack pointer
- Six 8-bit registers arranged in pairs: BC, DE, HL
- Requires +5V supply to operate at 3.2 MHZ single phase clock

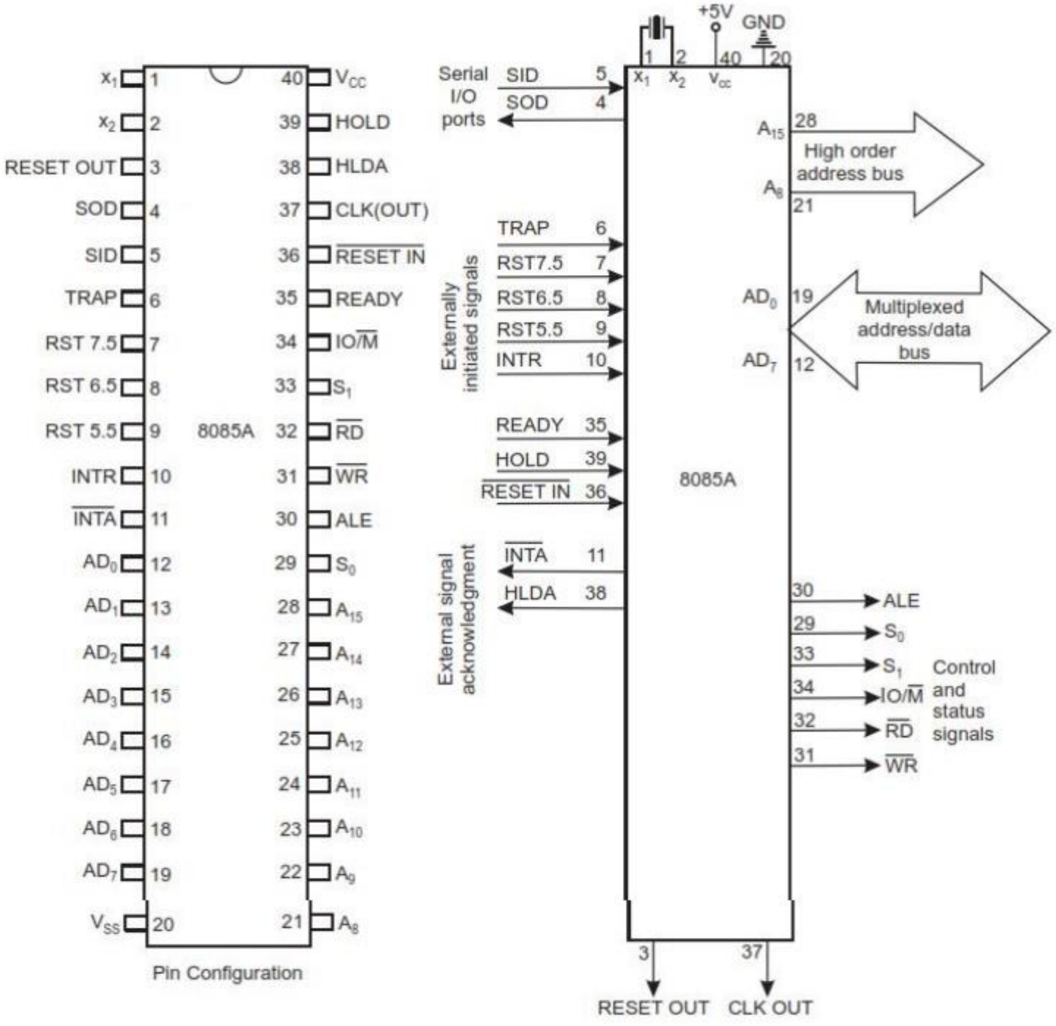


Fig 1.2 Pin Diagram of 8085

Flag register

It is an 8-bit register having five 1-bit flip-flops, which holds either 0 or 1 depending upon the result stored in the accumulator.

These are the set of 5 flip-flops –

- Sign (S)
- Zero (Z)
- Auxiliary Carry (AC)
- Parity (P)
- Carry (C)

Its bit position is shown in the following table –

D7	D6	D5	D4	D3	D2	D1	D0
S	Z		AC		P		CY

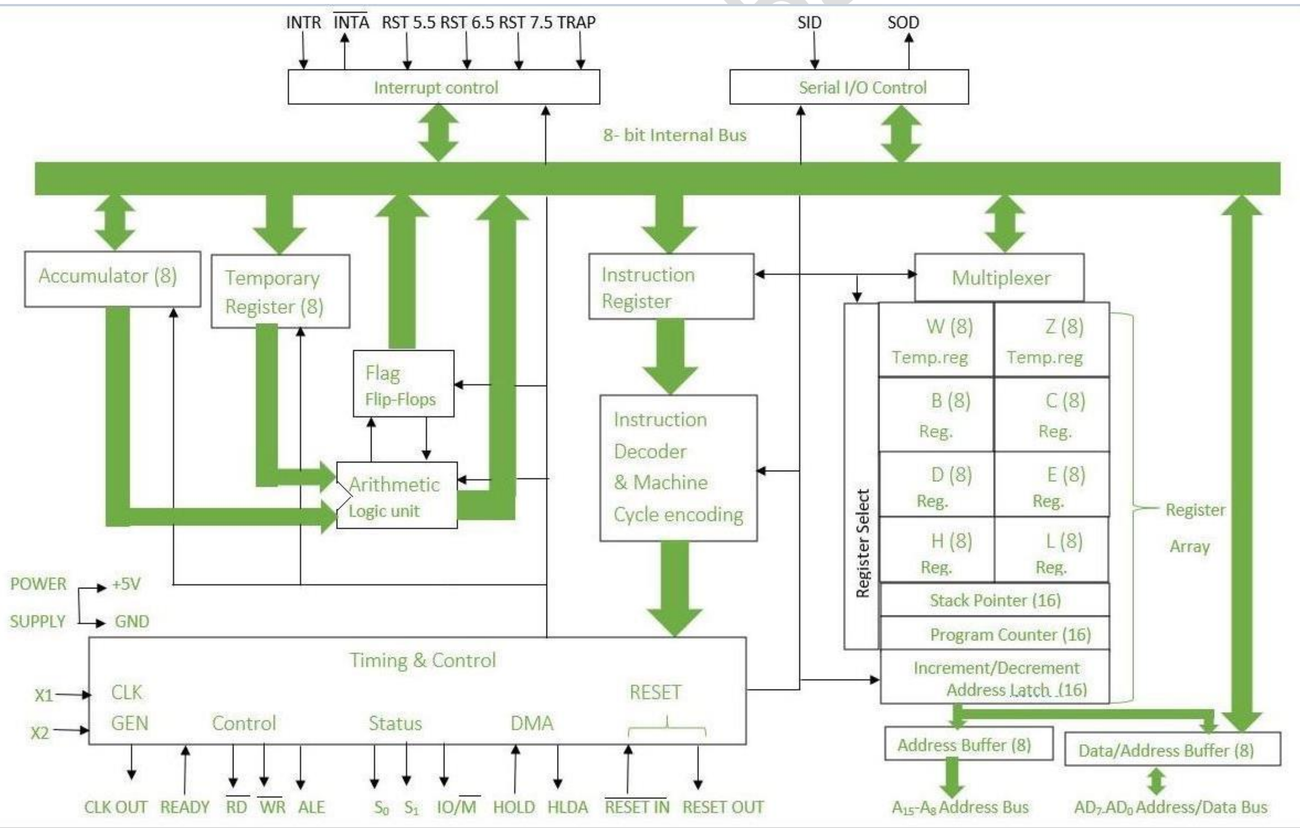
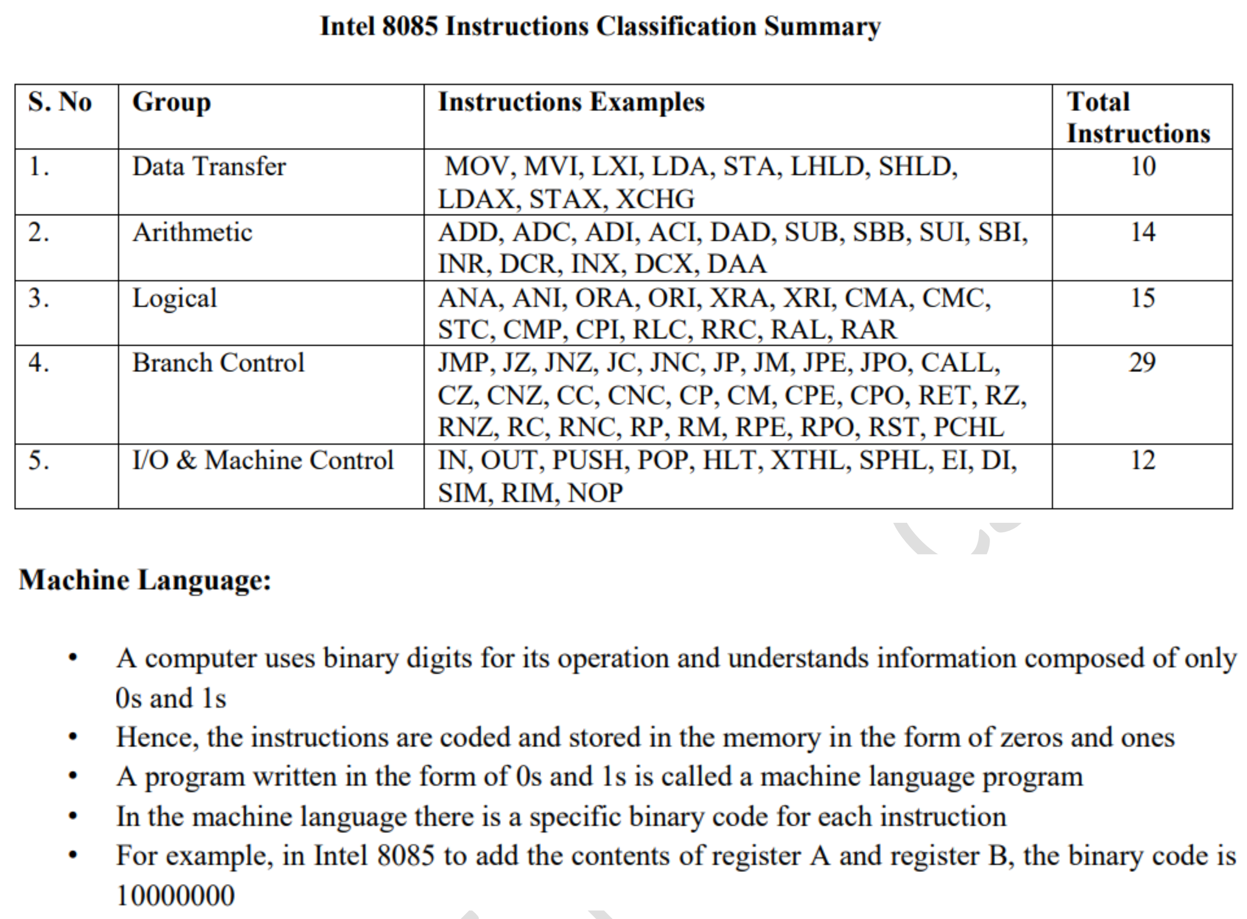


Fig: Architecture of 8085 Microprocessor



Intel 8085 Instructions Classification Summary

S. No	Group	Instructions Examples	Total Instructions
1.	Data Transfer	MOV, MVI, LXI, LDA, STA, LHLD, SHLD, LDAX, STAX, XCHG	10
2.	Arithmetic	ADD, ADC, ADI, ACI, DAD, SUB, SBB, SUI, SBI, INR, DCR, INX, DCX, DAA	14
3.	Logical	ANA, ANI, ORA, ORI, XRA, XRI, CMA, CMC, STC, CMP, CPI, RLC, RRC, RAL, RAR	15
4.	Branch Control	JMP, JZ, JNZ, JC, JNC, JP, JM, JPE, JPO, CALL, CZ, CNZ, CC, CNC, CP, CM, CPE, CPO, RET, RZ, RNZ, RC, RNC, RP, RM, RPE, RPO, RST, PCHL	29
5.	I/O & Machine Control	IN, OUT, PUSH, POP, HLT, XTHL, SPHL, EI, DI, SIM, RIM, NOP	12

Machine Language:

- A computer uses binary digits for its operation and understands information composed of only 0s and 1s
- Hence, the instructions are coded and stored in the memory in the form of zeros and ones
- A program written in the form of 0s and 1s is called a machine language program
- In the machine language there is a specific binary code for each instruction
- For example, in Intel 8085 to add the contents of register A and register B, the binary code is 10000000

Assembly Language:

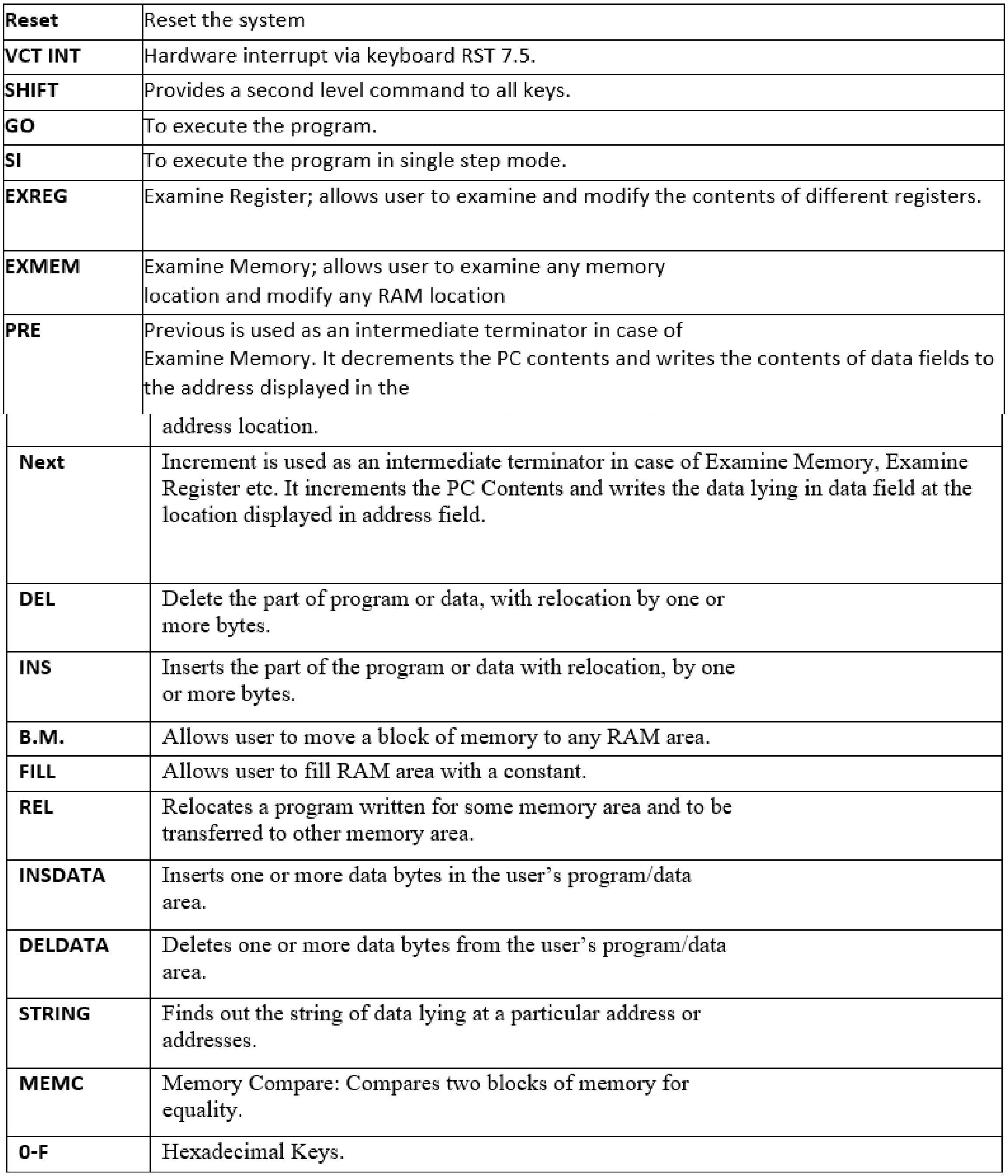
- A programmer can easily write a program in alphanumeric symbols instead of 0s and 1s
- Meaningful and easily rememberable symbols are chosen for the purpose
- Examples are: ADD for addition, SUB for subtraction, CMP for comparison etc…, such symbols called mnemonics
- A program written in mnemonics is known as assembly language program
- The writing of a program in assembly language is much easier and faster as compared to machine language
- Both machine language and assembly language are microprocessor specific

Procedure to execute a program in 8085 Trainer kit

1. Press Reset Key.
2. Press 'REL EXMEM' Key
3. Type starting address of the program (e.g.: - 8000)
4. Press 'Next' key
5. Type the Opcode of the program from starting to End by Pressing Next Key. (Next Key is used to go one by one-by-one memory location.)
6. Press Reset key

7. Press ‘GO’ then type starting address. (i.e., 2000)
8. Press ‘Fill’ key then press reset key.
9. Press ‘REL EXMEM’ key then type the o/p address (i.e., 9000)
10. Press ‘Next’ key
11. Data will be displayed on seven segment.

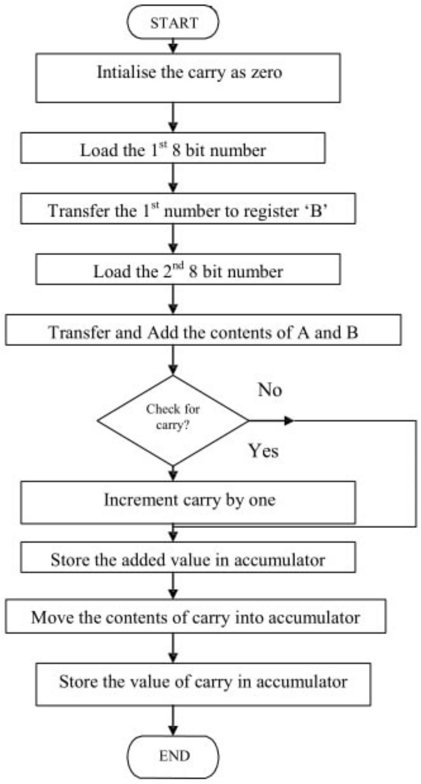
Some Commonly Used Command Keys



Reset	Reset the system
VCT INT	Hardware interrupt via keyboard RST 7.5.
SHIFT	Provides a second level command to all keys.
GO	To execute the program.
SI	To execute the program in single step mode.
EXREG	Examine Register; allows user to examine and modify the contents of different registers.
EXMEM	Examine Memory; allows user to examine any memory location and modify any RAM location
PRE	Previous is used as an intermediate terminator in case of Examine Memory. It decrements the PC contents and writes the contents of data fields to the address displayed in the address location.
Next	Increment is used as an intermediate terminator in case of Examine Memory, Examine Register etc. It increments the PC Contents and writes the data lying in data field at the location displayed in address field.
DEL	Delete the part of program or data, with relocation by one or more bytes.
INS	Inserts the part of the program or data with relocation, by one or more bytes.
B.M.	Allows user to move a block of memory to any RAM area.
FILL	Allows user to fill RAM area with a constant.
REL	Relocates a program written for some memory area and to be transferred to other memory area.
INSDATA	Inserts one or more data bytes in the user’s program/data area.
DELDATA	Deletes one or more data bytes from the user’s program/data area.
STRING	Finds out the string of data lying at a particular address or addresses.
MEMC	Memory Compare: Compares two blocks of memory for equality.
0-F	Hexadecimal Keys.

Lab 2: Assembly Language Program for 8085 microprocessors to add two 8-bit numbers.

Flow chart:


START
Intialise the carry as zero
Load the 1st 8 bit number
Transfer the 1st number to register 'B'
Load the 2nd 8 bit number
Transfer and Add the contents of A and B
Check for carry?
No
Yes
Increment carry by one
Store the added value in accumulator
Move the contents of carry into accumulator
Store the value of carry in accumulator
END

INPUT

The 1st number _your input is in the memory location 2600 H.
The 2nd number _your input is in the memory location 2601 H.

Program:

```
MVI C, 00H
LDA 2600H
MOV B, A
LDA 2601H
ADD B
JNC LOOP
INR C
LOOP: STA 2602H
MOV A, C
STA 2603H
HLT
```

Do not write this in your report, for report fill this to the table below.

Mnemonics	Hex-code	Comment	Memory Address

RESULT

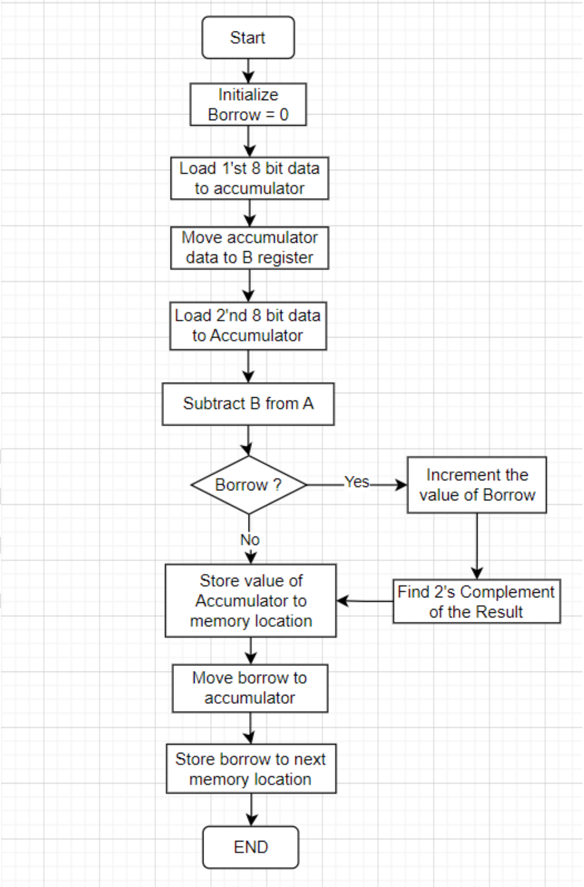
The result observed in memory location 2602 is: _____write from your experiment result________
The result observed in memory location 2603 is: _____ write from your experiment result________

Lab 3: Assembly language program for 8085 microprocessors to subtract two 8-bit numbers.

Flow Chart:

INPUT

The 1st number _your input is in the memory location 2500 H.
The 2nd number _your input is in the memory location 2501 H.


Start
Initialize Borrow = 0
Load 1'st 8 bit data to accumulator
Move accumulator data to B register
Load 2'nd 8 bit data to Accumulator
Subtract B from A
Borrow ?
Yes
Increment the value of Borrow
No
Find 2's Complement of the Result
Store value of Accumulator to memory location
Move borrow to accumulator
Store borrow to next memory location
END

PROGRAM

```
MVI C, 00H
LDA 2500H
MOV B, A
LDA 2501H
SUB B
 JNC LINE1
INR C
CMA
INR A
LINE1: STA 2502H
MOV A, C
STA 2503H
HLT
```

Do not write this in your report, for report fill this to the table below.

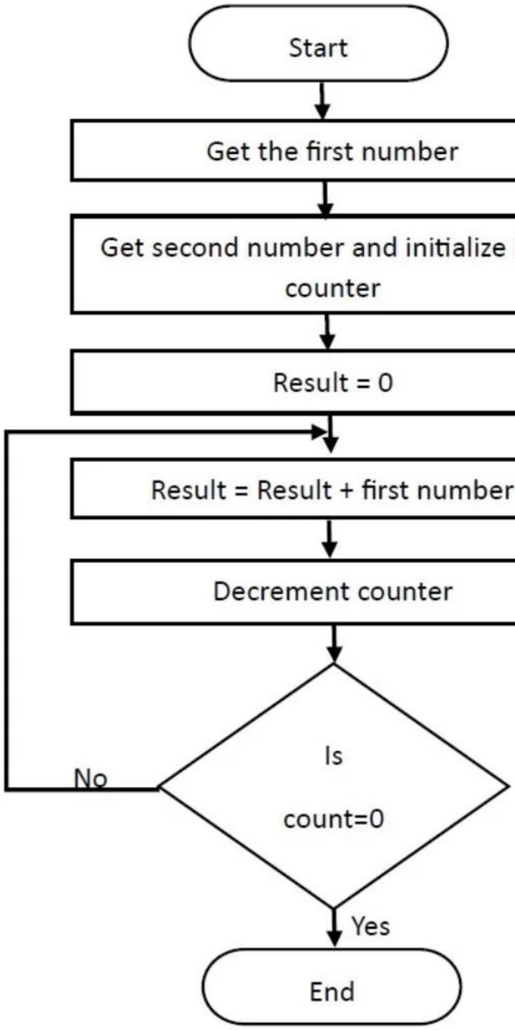
RESULT

The result observed in memory location 2502 is: _____write from your experiment result_______
The result observed in memory location 2503 is: ____ write from your experiment result_______

Mnemonics	Hex-code	Comment	Memory Address

Lab 4: Assembly language program for 8085 microprocessors to multiply two 8-bit numbers.

Flow Chart:


Start
Get the first number
Get second number and initialize it as counter
Result = 0
Result = Result + first number
Decrement counter
Is
count=0
No
Yes
End

INPUT

The 1st number _your input is in the memory location 2600 H.

The 2nd number _your input is in the memory location 2601 H.

PROGRAM

```
MVI D,00H       // Initialize Carry
LDA 2600H
DCR A
MOV C, A
LDA 2601H
MOV B, A

LINE1: ADD B
JNC LOOP
INR D
LOOP: DCR C
JNZ LINE1
STA 2602H

MOV A, D
STA 2603H
HLT
```

Do not write this code in your report, for report fill this to the table below.

RESULT

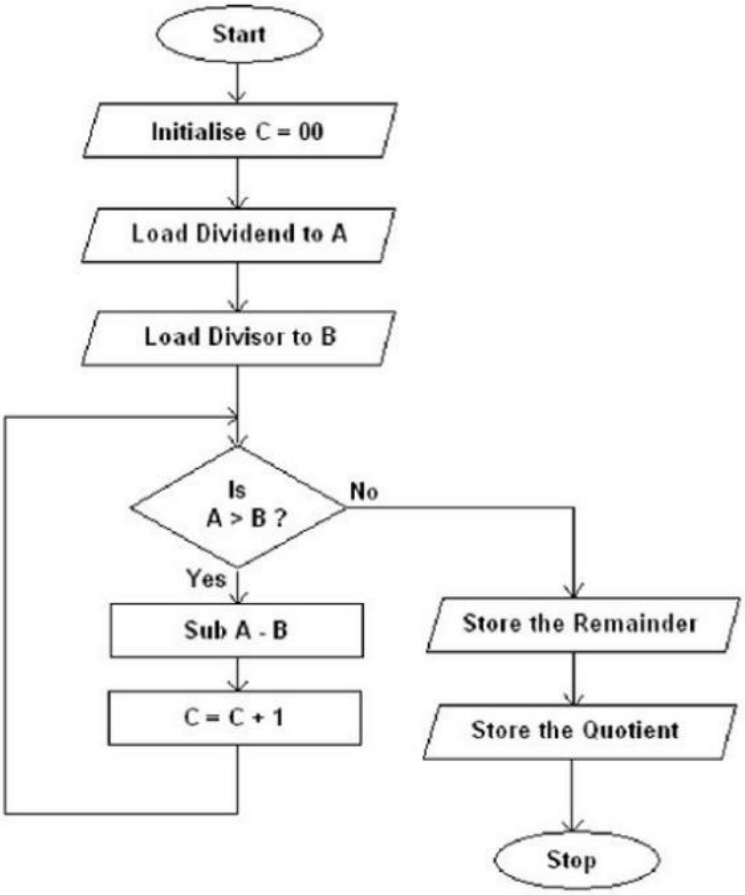
The result observed in memory location 2602 is: _____write from your experiment result________

The result observed in memory location 2603 is: ____ write from your experiment result________

Mnemonics	Hex-code	Comment	Memory Address

Lab 5: Assembly language program for 8085 microprocessors for Division of two 8-bit data.

Flow Chart:



INPUT

The 1st number _your input is in the memory location 2500 H.
The 2nd number _your input is in the memory location 2501 H.

PROGRAM

```
MVI C, 00H

LDA 2500H
LXI H, 2501H
LABEL: CMP M

JC LINE1
SUB M
INR C
JMP LABEL

LINE1:  STA 2503H
MOV A, C
STA 2502H
HLT
```

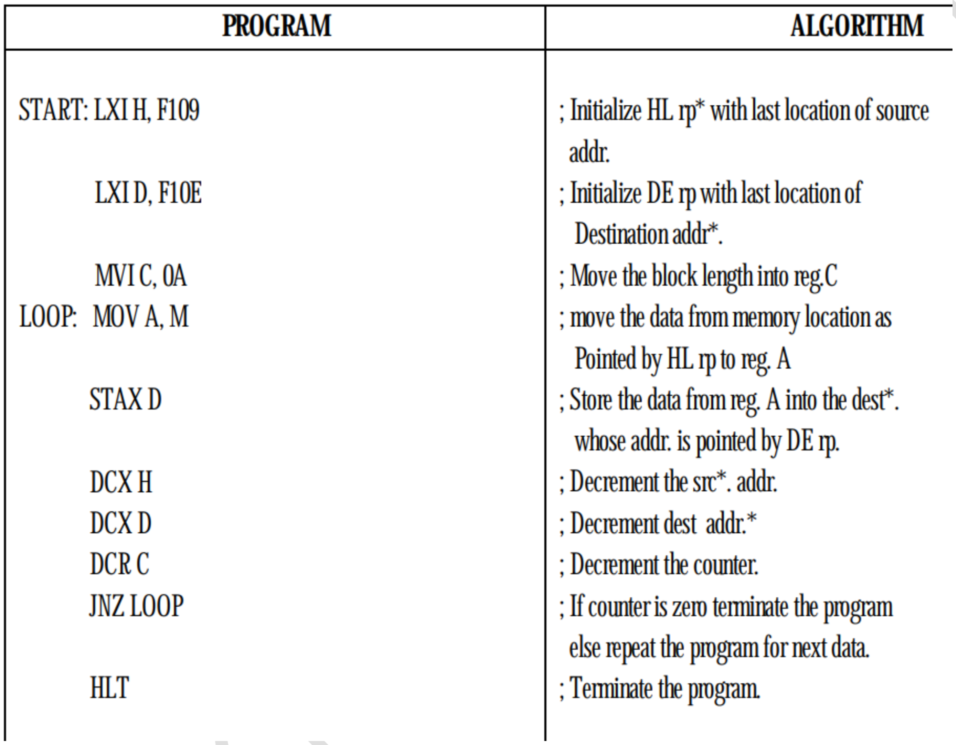
RESULT

The result observed in memory location 2502 is: _____write from your experiment result_______
The result observed in memory location 2503 is: ____ write from your experiment result_______
That is Quotient is _______ and Remainder is ___________.

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

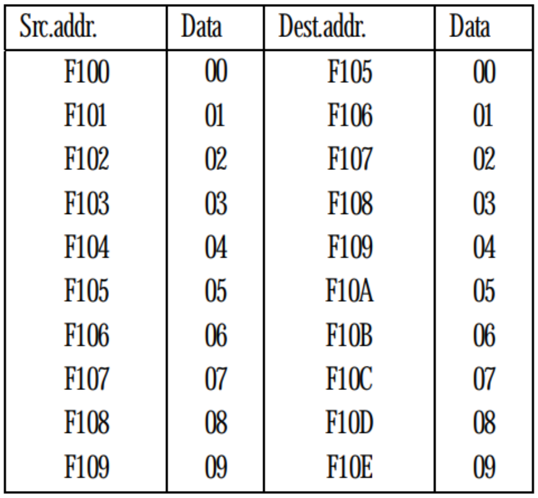
Lab 6: Assembly language program for 8085 microprocessors to move a block of data starting at location 'X' to location starting at 'Y'.

Starting Source Address: F109H
Starting Destination Address: F10E
Block size: 0AH



PROGRAM	ALGORITHM
START: LXI H, F109	; Initialize HL rp* with last location of source addr.
LXI D, F10E	; Initialize DE rp with last location of Destination addr*.
MVI C, 0A	; Move the block length into reg.C
LOOP: MOV A, M	; move the data from memory location as Pointed by HL rp to reg. A
STAX D	; Store the data from reg. A into the dest*. whose addr. is pointed by DE rp.
DCX H	; Decrement the src*. addr.
DCX D	; Decrement dest addr.*
DCR C	; Decrement the counter.
JNZ LOOP	; If counter is zero terminate the program else repeat the program for next data.
HLT	; Terminate the program.

Sample Input/Output



Src.addr.	Data	Dest.addr.	Data
F100	00	F105	00
F101	01	F106	01
F102	02	F107	02
F103	03	F108	03
F104	04	F109	04
F105	05	F10A	05
F106	06	F10B	06
F107	07	F10C	07
F108	08	F10D	08
F109	09	F10E	09

Lab 7: Assembly language program for 8085 microprocessors to add two 16-bit numbers.

Method 1: using DAD Instruction

Algorithm:

1. Start
2. Initialize carry = 0
3. Load 16-bit data to H-L register pair
4. Exchange the content of HL pair with DE pair
5. Double add register pair HL and DE using DAD instruction
6. If there is no carry jump to step 8
7. Increment the carry register
8. Move the content of L register to Accumulator
9. Store accumulator value to desired memory location
10. Move the content of H register to Accumulator
11. Store accumulator value to desired memory location
12. Move the carry value to accumulator
13. Store the accumulator value to desired memory location
14. End

INPUT

The 1st 16-bit number _your input.
The 2nd 16 bit-number _your input.

Program: (Write only in tabular form as below)

```
MVI C, 00H
LXI H, ABCDH
XCHG
LXI H,6789H
DAD D
JNC LOOP
INR C
LOOP: MOV A, L

STA 2501H
MOV A, H
STA 2502H
MOV A, C
STA 2503H
HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

RESULT

The result observed in memory location 2501 is: _____write from your experiment result_______
The result observed in memory location 2502 is: ____ write from your experiment result_______
The result observed in memory location 2503 is: ____ write from your experiment result____ is carry.

Method two: Without using DAD instruction.

Input

Memory Location	8-bit Hex-Data
2501H	Your input data
2502H	"
2503H	"
2504H	"

Algorithm: (Examine the program below and write the algorithm steps according. Write yourself as in previous experiment.)

Program: (Write only in tabular form as in previous experiment)

```
MVI C,00H
LXI H, 2503H
LDA 2501H
ADD M
LOOP: STA 2600H
INX H

LDA 2502H
ADC M
JNC LINE1
INR C

LINE1: STA 2601H

MOV A,C
STA 2603H
HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Result:

Memory Location	8-bit Hex-Data
2600H	Your result data
2601H	"
2602H	"

Lab 8: Assembly language program for 8085 microprocessors to subtract two 16-bit numbers.

Input

Memory Location	8-bit Hex-Data
2501H	Your input data
2502H	″
2503H	″
2504H	″

Algorithm: (Examine the program below and write the algorithm steps according. Write yourself as in previous experiment.)

Program: (Write only in tabular form as in previous experiment)

```
MVI C,00H
LXI H, 2503H
LDA 2501H
SUB M
LOOP: STA 2600H
INX H

LDA 2502H
SBB M
JNC LINE1
INR C
LINE1: STA 2601H

MOV A, C
STA 2603H
HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Result:

Memory Location	8-bit Hex-Data
2600H	Your result data
2601H	″
2602H	″

Lab 9: Assembly language program for 8085 microprocessors to multiply two 16-bit numbers.

Algorithm:

1. Load the first data in the HL pair.
2. Move content of HL pair to the stack pointer.
3. Load the second data in the HL pair and move it to DE.
4. Make H register as 00H and L register as 00H.
5. ADD HL pair and stack pointer.
6. Check for carrying if carry increment it by 1 else moves to the next step.
7. Then move E to A and perform OR operation with accumulator and register D.
8. If the value of the operation is zero, then store the value else go to step 3.

Program:

```
LHLD 2501H

SPHL
LHLD 2503H
XCHG
LXI H,0000H
LXI B,0000H

LOOP: DAD SP
JNC LINE1
INX B
LINE1: DCX D
MOV A, E
ORA D
JNZ LOOP

SHLD 2601H
MOV L, C
MOV H, B
SHLD 2603H
HLT
```

SAMPLE INPUT/OUTPUT

```
INPUT:
        (2501H) = 05H
        (2502H) = 07H
        (2503H) = 03H
        (2504H) = 04H
OUTPUT:
         (2601H) = 0FH
         (2602H) = 29H
         (2603H) = 1CH
         (2604H) = 00H
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Lab 10: Assembly language program for 8085 microprocessors to divide two 16-bit numbers.

Algorithm:

1. Initialize register BC as 0000H for Quotient.
2. Load the divisor in HL pair and save it in DE register pair.
3. Load the dividend in HL pair.
4. Subtract the content of accumulator with E register.
5. Move the content A to C and H to A.
6. Subtract with borrow the content of A with D.
7. Move the value of accumulator to H.
8. If CY=1, go to step 10, otherwise next step.
9. Increment register B and jump to step 4.
10. ADD both contents of DE and HL.
11. Store the remainder in memory.
12. Move the content of C to L & B to H.
13. Store the quotient in memory.

Program:

```
LXI B,0000H     //quotient
LHLD 2503H      //DE HOLDS DIVISOR
XCHG
LHLD 2501H

LOOP2: MOV A, L
SUB E
MOV L, A
MOV A, H
SBB D
MOV H, A
JC LOOP1

INX B
JMP LOOP2
LOOP1: DAD D

SHLD 2601H
MOV L, C
MOV H, B
SHLD 2603H
HLT
```

SAMPLE INPUT/OUTPUT

```
INPUT:
        (2501H) = 05H
        (2502H) = 07H
        (2503H) = 03H
        (2504H) = 04H
OUTPUT:
         (2601H) = F8H
         (2602H) = 00H
         (2603H) = 06H
         (2604H) = 00H
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Lab 11: Assembly language program for 8085 microprocessors to find 2's complement of 16-bit numbers.

Algorithm: (Examine the program below and write the algorithm steps according. Write yourself as in previous experiment.)

Input

Memory Location	8-bit Hex-Data
2501H	Your input data
2502H	"

Program:

```
LXI H, 2501H
MOV A, M

MVI B, 00H
CMA
ADI 01H
STA 2503H
JNC SECOND
INR B

SECOND:INX H

MOV A, M
CMA
ADD B
STA 2504
HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Result:

Memory Location	8-bit Hex-Data
2503H	Your result data
2504H	"

Lab 12: Assembly language program for 8085 microprocessors to find the sum of odd numbers (8 bits data) stored in a list.

Algorithm: (Examine the program below and write the algorithm steps according. Write yourself as in previous experiment.)

Input

Memory Location	8-bit Hex-Data
6000H to	Your input data
. . . . .	"
600AH	Total 10 inputs

Program:

```
MVI B, 0AH      // Ten numbers of input data
MVI C, 00H      // Initialize carry = 0
MVI D, 00H      // Initialize Sum = 0

LXI H, 6000H
LOOP: MOV A, M
ANI 01H
JZ EVEN
MOV A, M
ADD D
JNC NOCARRY
INR C

NOCARRY:MOV D, A
EVEN: INX H
DCR B
JNZ LOOP

MOV A, D
STA 6010H
MOV A, C
STA 6011H

HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Result:

Memory Location	8-bit Hex-Data
6010H	Your result data
6011H	"

Lab 13: Assembly language program for 8085 microprocessors to sort the data in ascending order (Bubble Sort Technique)

Algorithm

1. Initialize HL pair as memory pointer.
2. Get the count at 8000h into C – register.
3. Copy it in D – register (for bubble sort (N-1) times required).
4. Get the first value in Accumulator from memory.
5. Compare it with the value at next location.
6. If they are out of order, exchange the contents of Accumulator and Memory.
7. Decrement D –register content by 1.
8. Repeat steps 5 and 7 till the value in D- register become zero.
9. Decrement contents of C –register by 1.
10. Repeat steps 3 to 9 till the value in C – register becomes zero.

Program

```
LDA 8000H

MOV C, A
DCR
REPEAT: MOV D, C

LXI H, 8001H
LOOP: MOV A, M
INX H
CMP M

JC LINE1
JZ LINE1

MOV B, M

MOV M, A
DCX H
MOV M, B
INX H

LINE1:  DCR D

JNZ LOOP
DCR C
JNZ REPEAT
HLT
```

Sample Input	
Memory Location	**8-bit Hex-Data**
8000H	07H
8001H	12H
8002H	B1H
8003H	2CH
8004H	05H
8005H	68H
8006H	1BH
8007H	55H

Sample Output	
Memory Location	**8-bit Hex-Data**
8000H	05
8001H	12
8002H	1B
8003H	.
8004H	.
8005H	.
8006H	.
8007H	.

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Lab 14: Understanding the operation of PUSH, POP and PSW instructios.

Problem: Write an assembly language program in 8085 microprocessors to access Flag register and exchange the content of flag register F with register B.

Algorithm –

-

1. Push the value of PSW in-memory stack with the help of PUSH instruction
2. Pop the value of the Flag register and store it in register H with help of POP instruction
3. Move the value of register L in register C [H -> ACCUMULATOR VALUE, L-> FLAG REGISTER VALUE]
4. Move the value of register B in register L
5. Move the value of register C in register B
6. Push the value of register H in-memory stack with the help of PUSH instruction
7. Pop the value of PSW from the memory stack using POP instruction

Input:

Memory Location	8-bit Hex-Data
Register B	3F
Flag register	01

Program:

```
LXI SP, 2006H
MVI B, 3FH

STC

PUSH PSW

POP H
MOV C, L
MOV L, B
MOV B, C
PUSH H
POP PSW
HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Output:

Memory Location	8-bit Hex-Data
Register B	?
Flag register	?

Lab 15: Assembly language program for 8085 microprocessors find the square of number using lookup table.

Algorithm

1. Load the input from memory address
2. Verify it is in table lookup range
3. Set HL pair so that it can point to memory address where the square of input data matched.
4. Store the output in desired memory location.

Program

```
LDA 2000H
CPI 0BH          // CHECK THE INPUT IS GREATER THAN 0B OR NOT
JC LINE1         // IF YES THEN ERROR!
MVI A, FFH
STA 3000H
HLT

LINE1: ADI 50H
MOV L,A
MVI H,20H
MOV A,M
STA 3000H
HLT

#ORG 2050H
#DB 00H,01H,04H,09H,10H,19H,24H,31H,40H,51H,64H,79H,90H

#ORG 2000H
#DB 0CH
```

Hint: It is not necessary to include the #part of program codes in the table.

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Lab 16: Assembly language program for 8085 microprocessors find multiplication table of given input number.

Algorithm: (Examine the program below and write the algorithm steps according. Write yourself as in previous experiment.)

Input:

Memory Location	8-bit Hex-Data
2500h	Ex. 03H

Program:

```
MVI C, 00H
LDA 2500H
MOV B,A
LXI H, 2600H

LOOP:MOV M,A
INR C
MOV A,C
CPI 0AH
JNC LINE1
MOV A,M
ADD B

INX H
JMP LOOP
LINE1: HLT
```

Mnemonics	Hex-code	Comment	Memory Address
		Fill this table	

Output:

Memory Location	8-bit Hex-Data
2600H	?
2601H	?
2602H	?
2603H	?
2604H	?
2605H	?
2606H	?
2607H	?
2608H	?
2609H	?

Lab 17: Assembly language program for 8086 microprocessors involving data transfer instructions. [Byte and word data transfer in different addressing modes]

```
DATA SEGMENT
DATA1 DB 23H
DATA2 DW 1234H
DATA3 DB 0H
DATA4 DW 0H
DATA5 DW 2345H,6789H
DATA ENDS

CODE SEGMENT
 ASSUME          CS:CODE,          DS:DATA

START: MOV AX,DATA               ;Initialize DS to point to start of the memory
 MOV DS,AX                      ;set aside for storing of data
 MOV AL,25H                     ;copy 25H into 8 bit AL register
 MOV AX,2345H                   ;copy 2345H into 16 bit AX register
 MOV BX,AX                      ;copy the content of AX into BX register(16 bit)
 MOV CL,AL                      ;copy the content of AL into CL register
 MOV AL,DATA1                   ;copies the byte contents of data segment memory ;location DATA1 into
                                8 bit AL
 MOV AX,DATA2                   ;copies the word contents of data segment memory ;location DATA2
                                into 16 bit AX
 MOV DATA3,AL                   ;copies the AL content into the byte contents of data ;segment memory
                                location DATA3
 MOV DATA4,AX                   ;copies the AX content into the word contents of ;data segment
                                memory location DATA4
 MOV BX,OFFSET DATA5            ;The 16 bit offset address of DS memory location  ; DATA5 is copied into
                                BX
 MOV AX,[BX]                    ; copies the word content of data segment ;memory location addressed
                                by BX into ;AX(register indirect addressing)
 MOV DI,02H                     ;address element
 MOV AX,[BX+DI}                 ; copies the word content of data segment ;memory location addressed
                                by BX+DI into ;AX(base plus indirect addressing)
 MOV AX,[BX+0002H]              ; copies the word content of data segment ;(16 bit)
 MOV AL,[DI+2]                  ;register relative addressing
 MOV AX,[BX+DI+0002H]           ;copies the word content of data segment ;memory location addressed
                                by BX+DI+0002H ;into AX(16 bit)
 MOV AH,4CH                     ; Exit to DOS with function call 4CH
 INT 21H
CODE ENDS                       ; Assembler stop reading
END START
```

Lab 18: Assembly language program for 8086 microprocessors to understand different directives and comments.

The directives are the number of statements that enables us to control the way in which the source program assembles and lists. These statements called directives act only during the assembly of program and generate no machine-executable code.

I. The page and title listing directives
II. SEGMENT directive
III. PROC Directives
IV. END Directive
V. ASSUME Directive
VI. Processor directive
VII. Dn Directive (Defining data types)
VIII. DUP Directive
IX. The EQU directive

EXAMPLE: Program written in Conventional full segment directive

```
Page 60,132
TITLE SUM program to add two numbers
;-----------------------------------------------------

STACK SEGMENT PARA STACK Stack
DW 32 DUP(0)
STACK ENDS
;-----------------------------------------------------

DATA SEGMENT PARA Data
NUM1 DW 3291
NUM2 DW 582
SUM DW ?
DATA SEG ENDS
;------------------------------------------------------

CODE SEGMENT PARA Code
MAIN PROC FAR

ASSUME SS: STACK, DS:DATASEG, CS:CODESEG
MOV AX, @DATA
MOV DS, AX
MOV AX, NUM1
ADD AX, NUM2
MOV AX, 4C00H
INT 21H
MAIN ENDP
CODE ENDS
END MAIN
```

EXAMPLE: Program written using simplified segment directives.

Program:

```
Page 60, 132
TITLE Sum program to add two numbers.
.MODEL SMALL
.STACK 64
.DATA
NUM1 DW 3241
NUM2 DW 572
SUM DW ?
.CODE
MAIN PROC FAR
MOV AX, @DATA ; set address of data segment in DS
MOV DS, AX
MOV AX, NUM1
ADD AX, NUM2
MOV SUM, AX
MOV AX, 4C00H ; End processing
INT 21H
MAIN ENDP ; End of procedure
END MAIN ; End of program
```

Lab 19: Assembly language program for 8086 microprocessors to display a string “This is a microprocessor Lab.”

Program:

```
.model small
.data
String1 db "This is a microprocessor lab.$"
.code
Main proc
MOV BX, @data
MOV CX, offset String1
MOV DS, BX
MOV AH, 09H
INT 21H
MOV AH, 4CH
INT 21H
Main endp
End Main
```

Lab 20: Write an ALP to find factorial of number for 8086.

Program:

```
MOV AX, 05H
MOV CX, AX
DEC CX

BACK:MUL CX
DEC CX
JZ STOP
JMP back
; results stored in AX
; to store the result at 7000H
STOP:MOV [7000H],AX
HLT
```

Lab 21: Write a program to reverse the given string for 8086.

Program:

```
.model small
.data
String1 db " Assembly language program.$"  ; "Assembly language program is a given string"
Length dw $-String1-2
Main proc
MOV AX, @data
MOV DS, AX
MOV SI, offset String1
MOV CX, Length
ADD SI, CX
Back: MOV DL, [SI]
MOV AH, 02H
INT 21H
DEC SI
LOOP Back
MOV AH, 4CH
INT 21H
Main endp
End Main
```

THE END