Fall Semester 2021-2022

Microprocessor and Interfacing

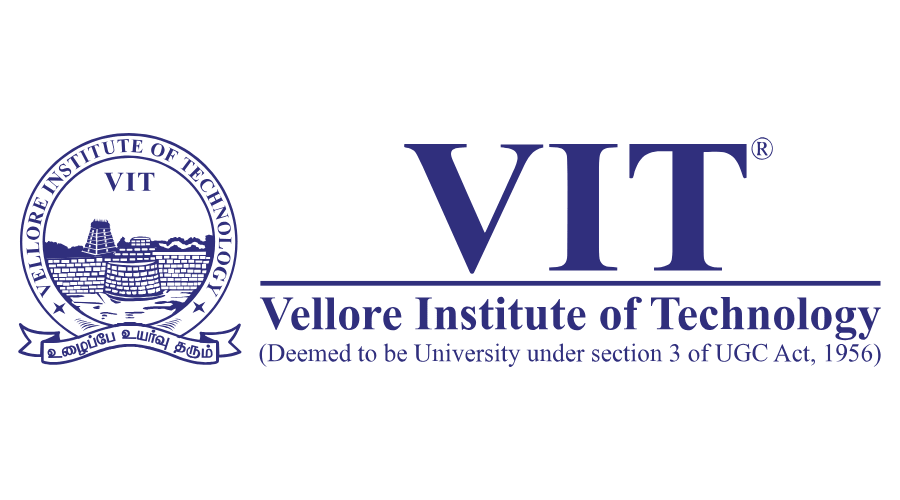
Lab Report

Digital Assignment-4

Experiment No: 5 Task No: 4

Course Code: CSE2006

Slot: L7+L8



Submitted By: Alokam Nikhitha

Reg. Numb: 19BCE2555

Submitted To: Dr. Abdul Majed KK

**EXPERIMENT 5:**

**Aim:**

**1. Write an ALP Program to find LCM of a given numbers**

**2. Write an ALP program to find the average of N numbers.**

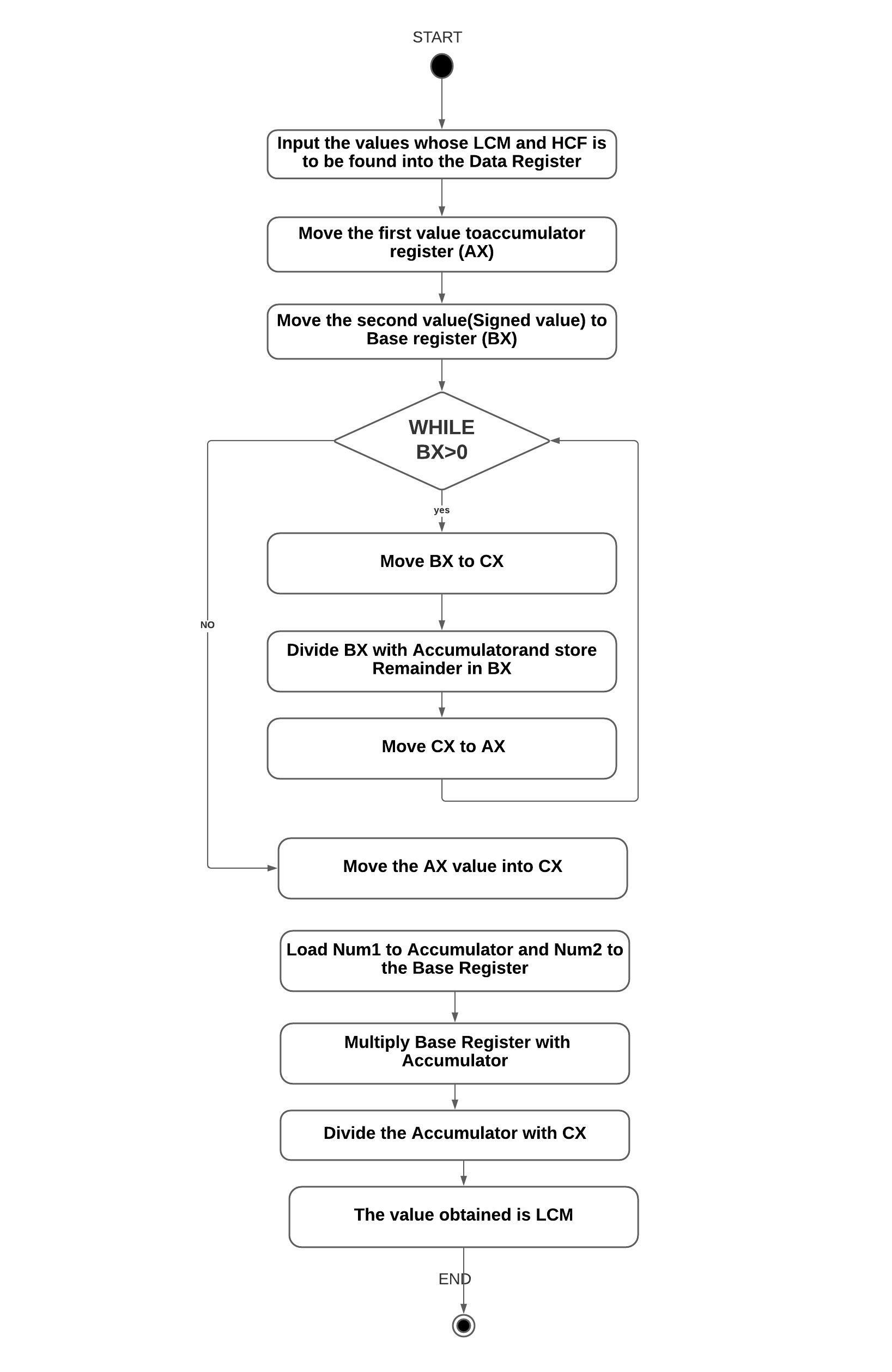
**3. Write an ALP to find the greatest among two numbers.**

**Tool Used: emu8086 simulator**

**1. Write an ALP Program to find LCM of a given numbers**

**Algorithm:**

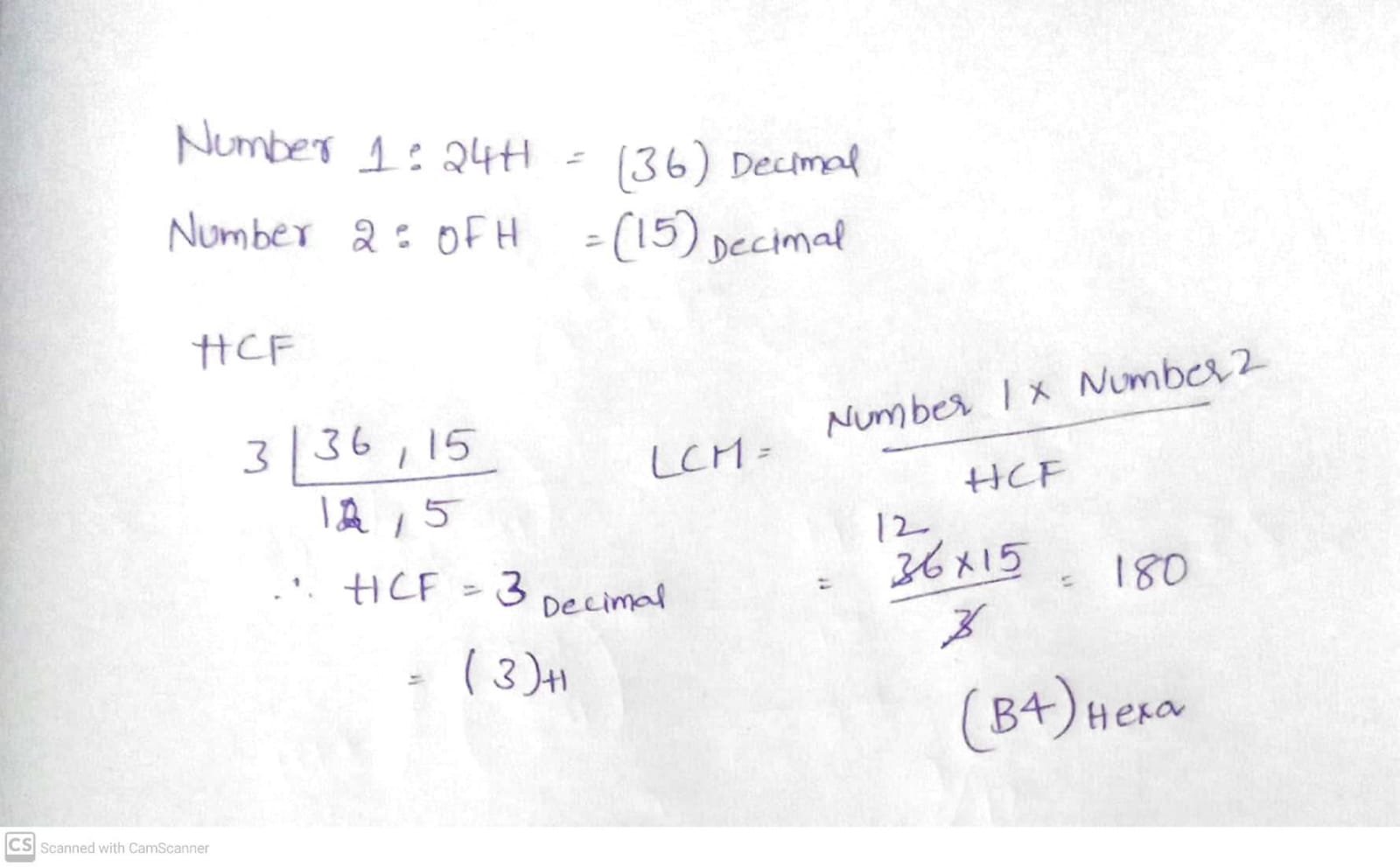
* **Input the values whose LCM and HCF is to be found into the Data Register**
* **Move the first value to accumulator register (AX).**
* **Move the second value(Signed value) to Base register (BX).**
* **Create a while loop until until BX becomes 0**
* **Move BX to CX.**
* **Divide BX with Accumulator and store Remainder in BX**
* **Move CX to AX and repeat the While loop**
* **After the While loop ends the value in AX is HCF**
* **Move the AX value into CX**
* **Load Num1 to Accumulator and Num2 to the Base Register**
* **Multiply Base Register with Accumulator**
* **Divide the Accumulator with CX(HCF)**
* **The value obtained is LCM(stored in AX)**
* **Halt the overall process.**

**Flow Chart: **

**Design and Calculations:**

**Input the values whose LCM and HCF is to be found into the Data Register.Move the first value to accumulator register (AX).Move the second value(Signed value) to Base register (BX).Create a while loop until until BX becomes 0.Move BX to CX.**

**Divide BX with Accumulator and store Remainder in BX.Move CX to AX and repeat the While loop.After the While loop ends the value in AX is HCF.Move the AX value into CX.Load Num1 to Accumulator and Num2 to the Base Register .Multiply Base Register with Accumulator.Divide the Accumulator with CX(HCF).The value obtained is LCM(stored in AX)**

****

**Program Code:**

**ASSUME CS:CODE DS:DATA**

**DATA SEGMENT**

**NUM1 DW 24H**

**NUM2 DW 0FH**

**HCF DW ?**

**LCM DW ?**

**ENDS**

**CODE SEGMENT**

**ASSUME DS:DATA CS:CODE**

**START:**

**MOV AX,DATA**

**MOV DS,AX**

**MOV AX,NUM1**

**MOV BX,NUM2**

**WHILE:MOV DX,0**

**MOV CX,BX**

**DIV BX**

**MOV BX,DX**

**MOV AX,CX**

**CMP BX,0**

**JNE WHILE**

**MOV HCF,AX**

**MOV CX,AX**

**MOV AX,NUM1**

**MOV BX,NUM2**

**MUL BX**

**DIV CX**

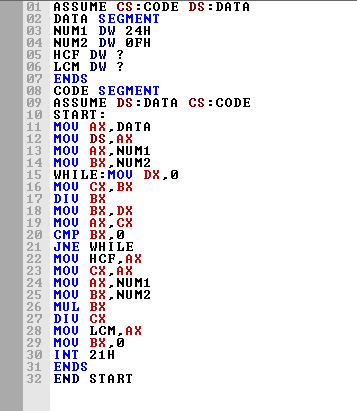
**MOV LCM,AX**

**MOV BX,0**

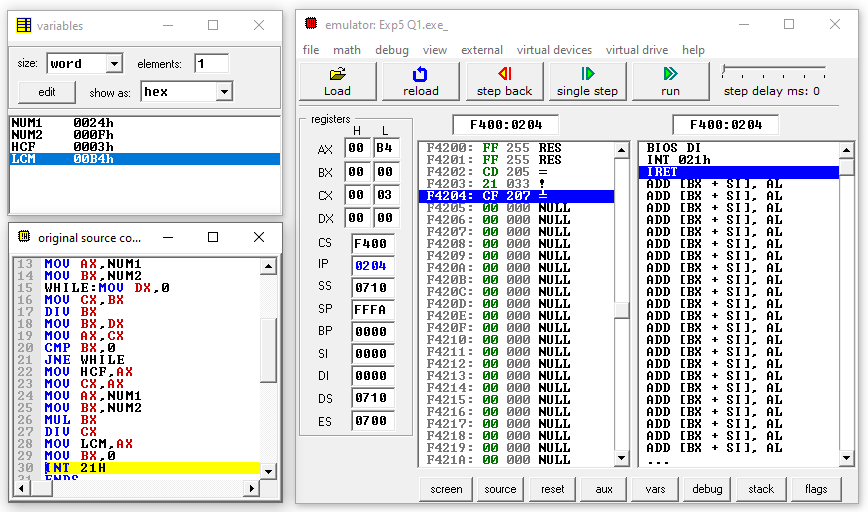
**INT 21H**

**ENDS**

**END START**

****

**Output:**

****

**Result and Inference:**

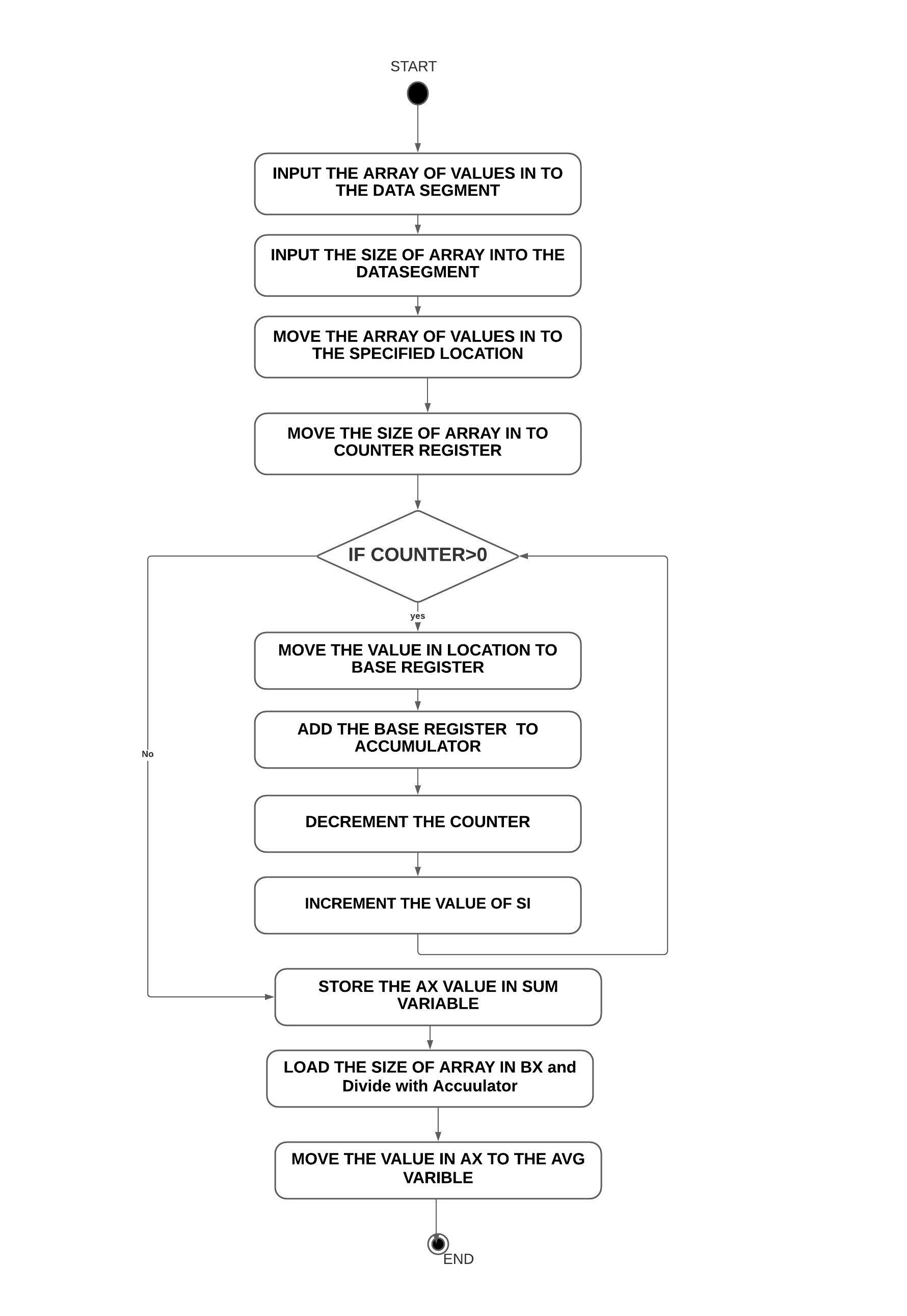
**-The HCF of the 2 input values in Stored in CX and LCM is stored in AX and Variable we can see that HCF is 0003H and LCM is 00B4H**

**2) Write an ALP program to find the average of N numbers.**

**Algorithm**

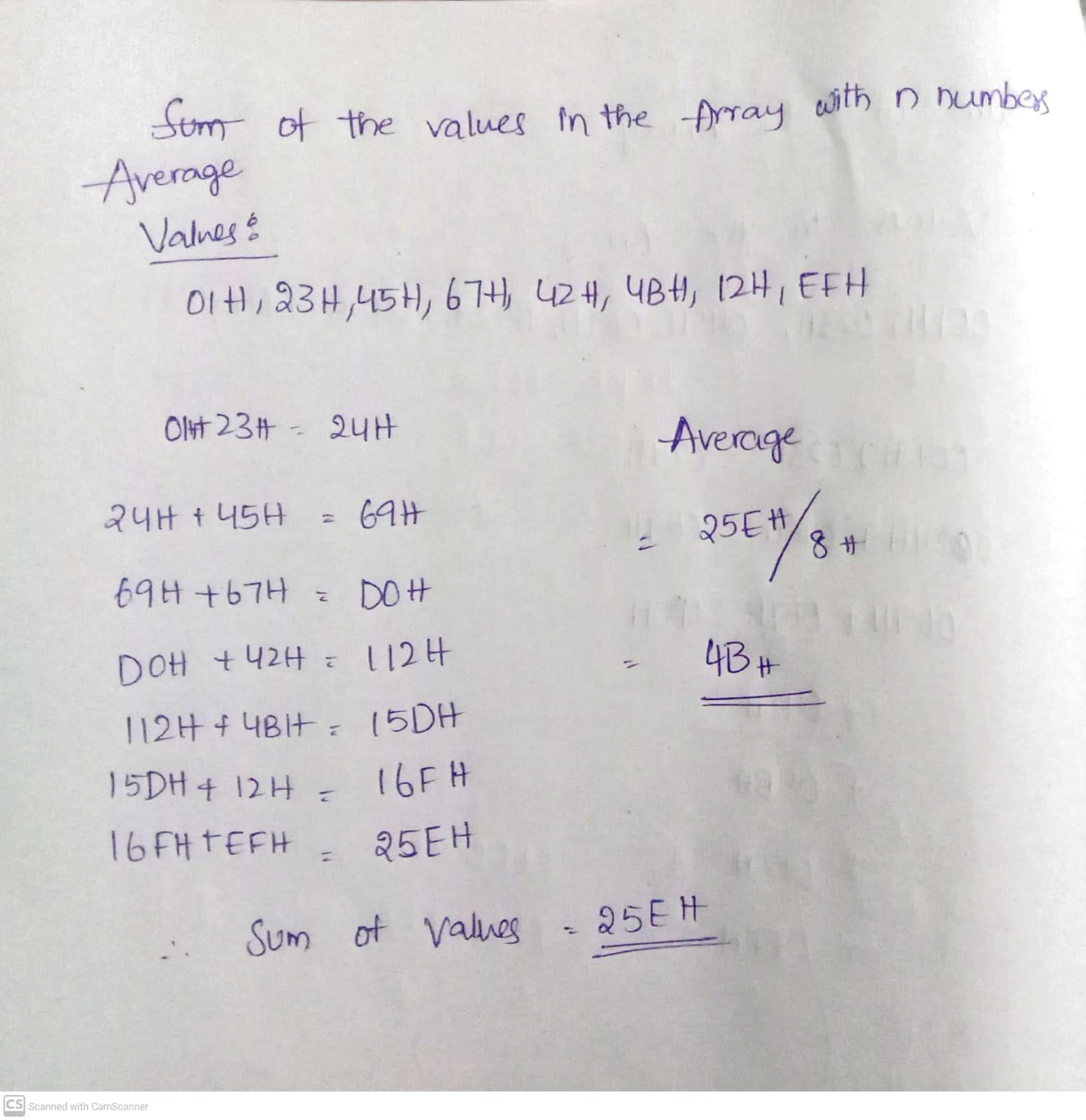
* **Input the Array of Values in Data Segment**
* **Input the Size of the Array in Data segment**
* **Take SUM and AVG in DataSegment**
* **Move all the values of the Array to a specified Memory Location(SI) in Code Segment**
* **Move the size of Array from Data segment in to Counter Register**
* **Now Run a loop and Add move the values in SI location of to Base Register and add that to Accumulator .Increment the SI value and**
* **Decrement the Counter Register**
* **Repeat the Process until the Counter becomes 0**
* **The Sum is stored in Accumulator.**
* **Move Size of Array in to BX**
* **-Divide the Sum with BX**
* **-Move the value to AVG**

**Flow chart:**

****

**Design and Calculations:**

**Input the Array of Values(001H, 023H, 045H, 067H, 042H, 04BH, 012H, 0EFH ) in Data Segment .Input the Size of the Array(8) in Data segment.Move all the values of the Array to a specified Memory Location(SI) in Code Segment.Move the size of Array from Data segment in to Counter Register.Now Run a loop and Add move the values in SI location of to Base Register and add that to Accumulator .Increment the SI value and Decrement the Counter Register.Repeat the Process until the Counter becomes 0 .The Sum is stored in Accumulator. Move Size of Array in to BX.Divide the Sum with BX.Move the value to AVG**

****

**Program Code:**

**ASSUME CS: CODE ,DS: DATA**

**DATA SEGMENT**

**ARR DB 001H, 023H, 045H, 067H, 042H, 04BH, 012H, 0EFH**

**N DW 08H**

**SUM DW 01 DUP (?)**

**AVG DW 01 DUP (?)**

**DATA ENDS**

**CODE SEGMENT**

**START:**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV CX, N**

**MOV AX, 0000H**

**MOV SI, OFFSET ARR**

**ABC:**

**MOV BL, [SI]**

**INC SI**

**ADD AX, BX**

**DEC CX**

**JNZ ABC**

**MOV SUM, AX**

**MOV CX, AX**

**MOV BX, N**

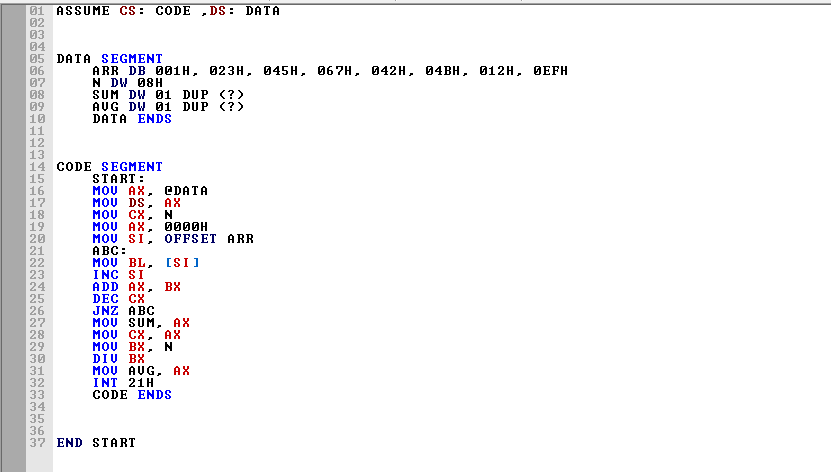
**DIV BX**

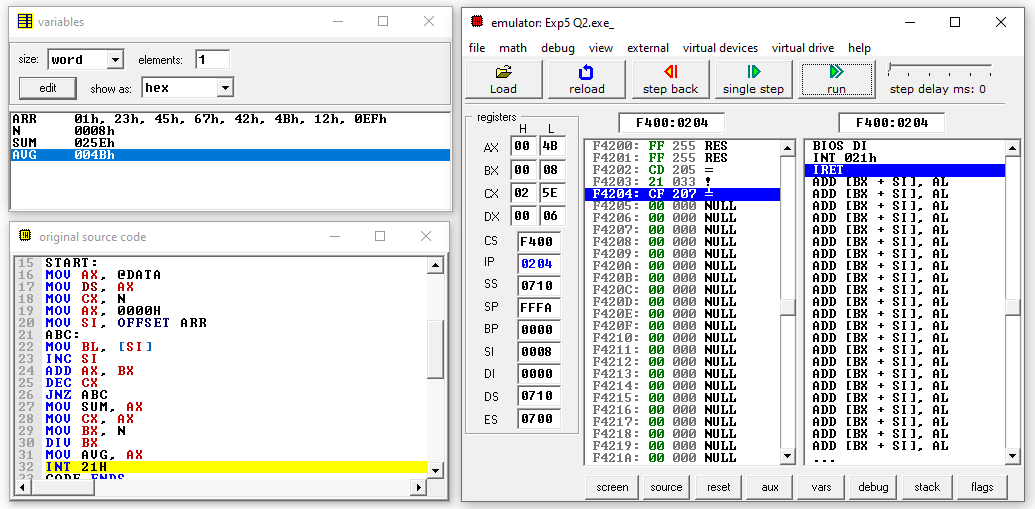
**MOV AVG, AX**

**INT 21H**

**CODE ENDS**

**END START**

****

**Output:** 

**Result and Inference:**

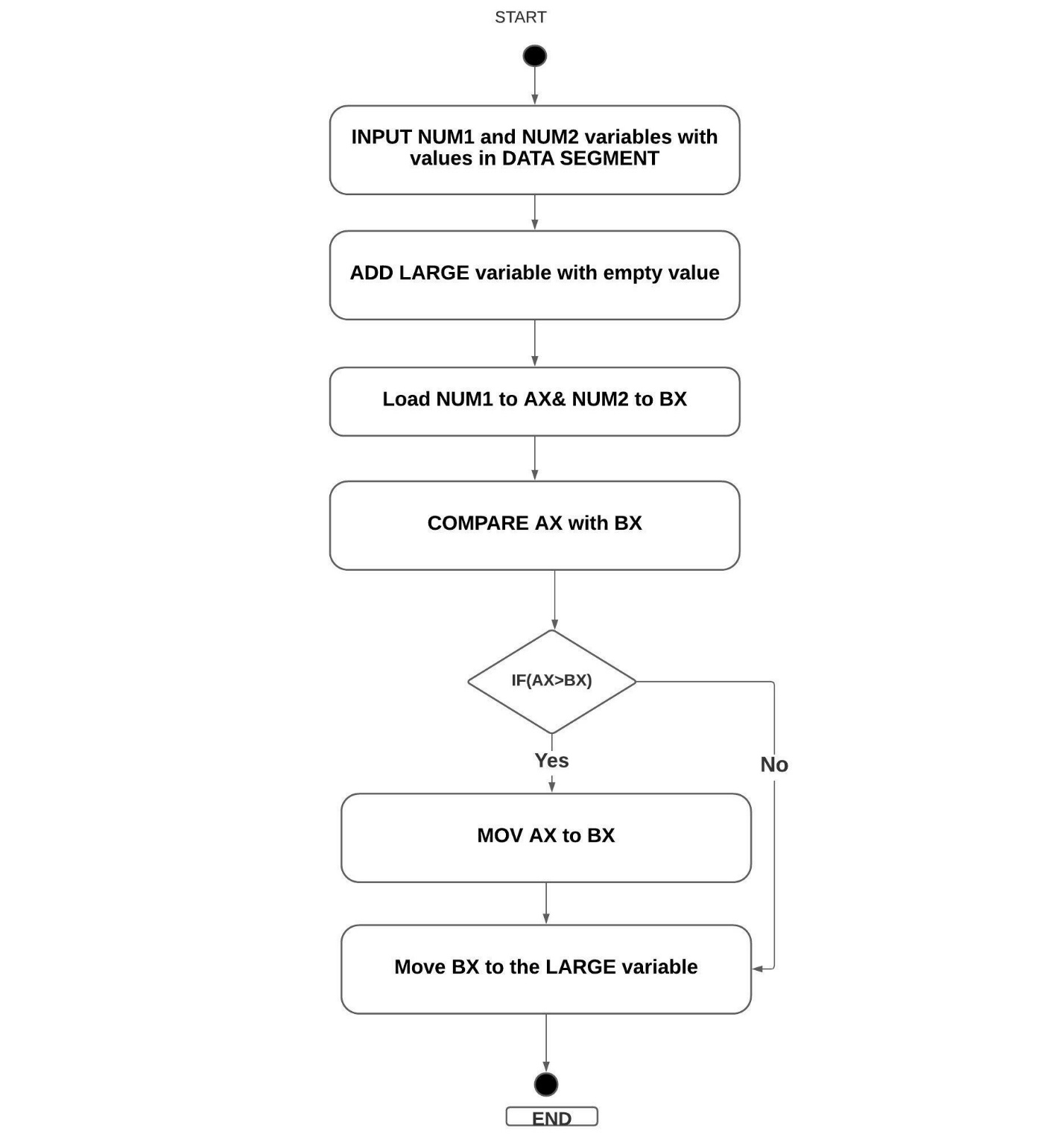
**-The sum of the Values is stored in CX, and Average is Stored in Accumulator. In the Variables We can see the value of Sum(25EH) and Average(4BH)**

**3) Write an ALP to find the greatest among two numbers.**

**ALGORITHM**

* **Input the 2 numbers NUM1 and NUM2 in the Data segment**
* **Move NUM1 into AX and NUM2 into BX**
* **Compare AX with BX**
* **If carry is obtained(ie if AX is smaller than BX) .Jump to Next**
* **Move AX to BX (if BX greater than AX)**
* **Next Loop: Move the BX value to LARGE variable**

**FLOWCHART**

****

**Design and Calculations:**

**Input the 2 numbers NUM1 and NUM2 in the Data segment.Then move NUM1 into AX and NUM2 into BX.Compare AX with BX .If carry is obtained(ie if AX is smaller than BX) .Jump to Next.Move AX to BX (if BX greater than AX).In Next Loop Move the BX value to LARGE variable**

**Program Code:**

**DATA SEGMENT**

**NUM1 DW 1529H**

**NUM2 DW 1231H**

**LARGE DW ?**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME DS:DATA,CS:CODE**

**START:**

**MOV AX,DATA**

**MOV DS,AX**

**MOV CX, 01h**

**MOV BX, NUM2**

**UP:**

**MOV AX, NUM1**

**CMP AX, BX**

**JL NXT**

**MOV BX,AX**

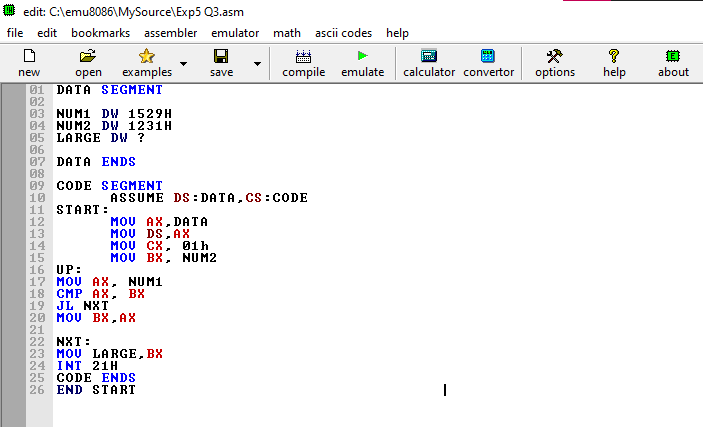
**NXT:**

**MOV LARGE,BX**

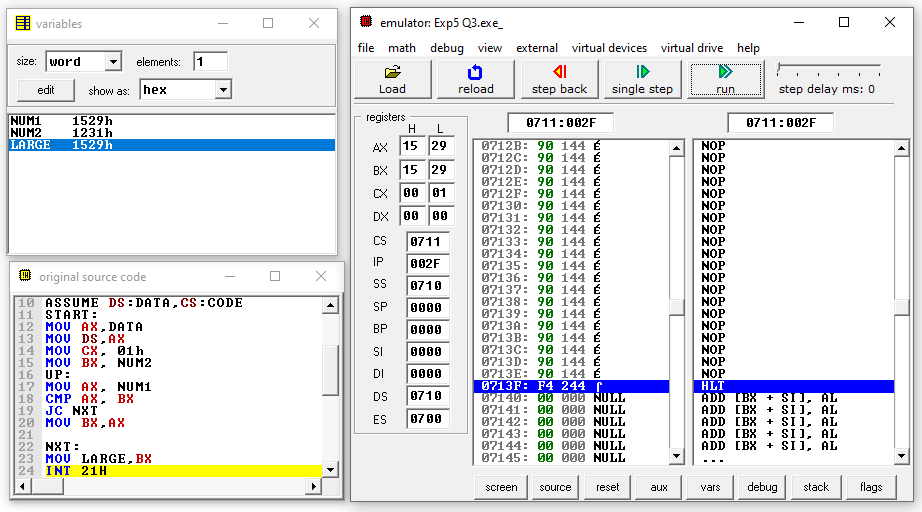
**INT 21H**

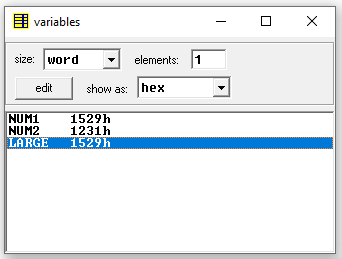
**CODE ENDS**

**END START**

****

**Output:**

****

****

**Result and Inference:**

**Since AX is greater than BX. AX is moved to BX and BX value is moved to LARGE variable( 1529H)**