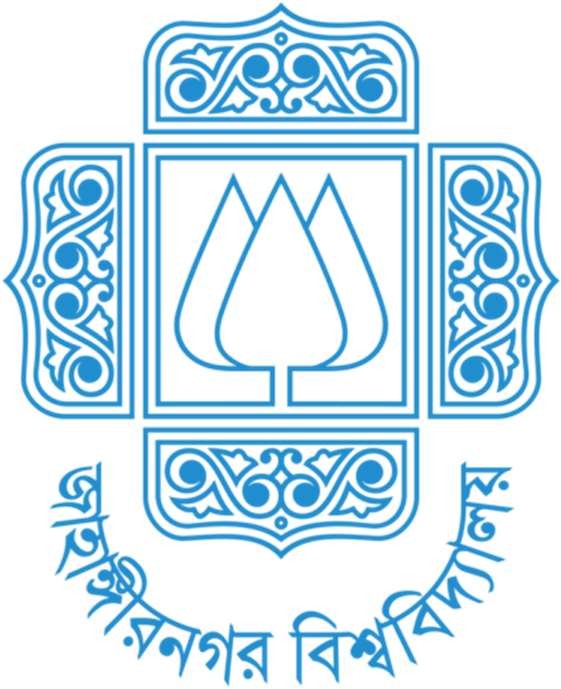
Jahangirnagar University (JU)



**Institute of Information Technology**

**Lab Report-2**

Assembly Language

**Name:** Md Shakil Hossain **Roll:** 2023

# Experiment 1:

Multiplication of two 8-bit numbers (Using an assembly language program) [input as hex].

# Algorithm:

Step1: Start

Step2: Initialize data segment through AX register in the DS register. Step3: Read first digit in AL register through keyboard.

Step4: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step5: Move contents of AL register to a BL.

Step6: Rotate the contents of BL register by 4 positions at left side. Step7: Read a second digit in AL register through keyboard

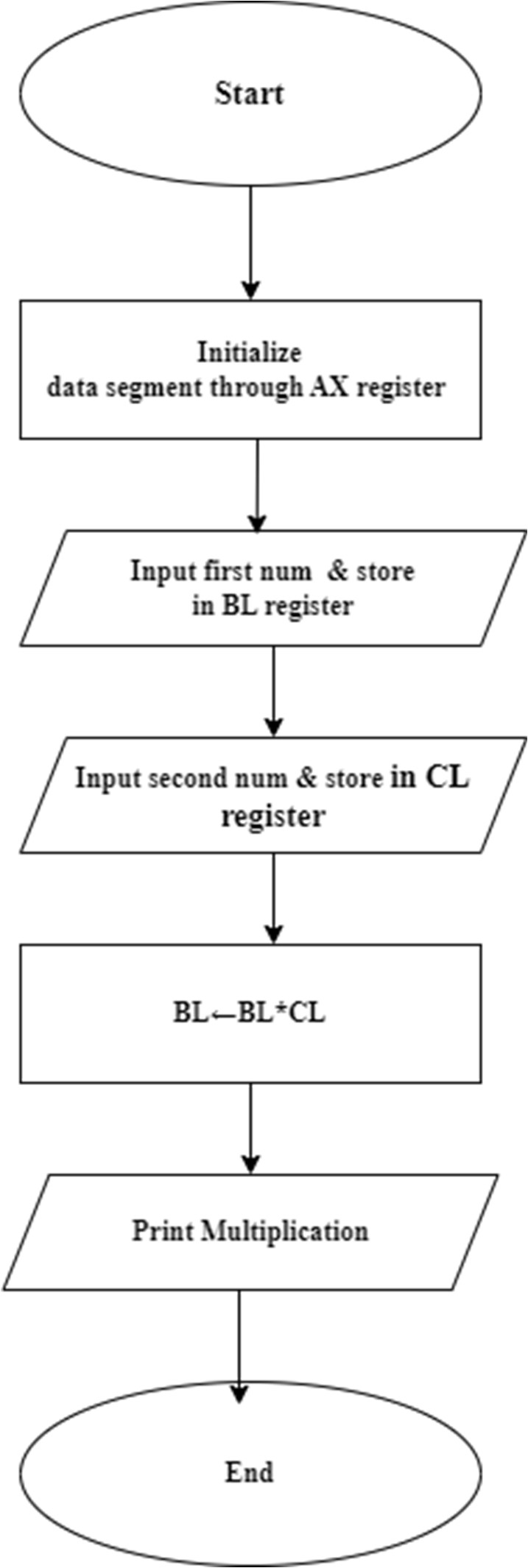
Step8: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step9: Add the contents of BL and AL store the result in BL.

Step10: Follow step 2 to 10 to input second number and store the result of number in CL Step11: Multiply the contents of BL and CL and result gets stored in BL

Step12: Display the multiplication result in the screen Step13: Stop

# Flow Chart



**Program Source Code:**

Data segment

msg db 0dh,0ah,"Enter first number: $" msg1 db 0dh,0ah,"Enter second number: $" result db 0dh,0ah,"The Result is: $"

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 ah,01h int 21h

call AsciitoHex mov bl,al

rol bl,4

mov ah,01h int 21h

call AsciitoHex add bl,al

mov dx,offset msg1 mov ah,09h

int 21h

mov ah,01h int 21h

call AsciitoHex mov cl,al

rol cl,4

mov ah,01h int 21h

call AsciitoHex add cl,al

mov al,bl mov bl,cl mul bl

mov cx,ax

mov dx,offset result mov ah,09h

int 21h

and ax,0f000h ror ax,12

mov bl,al

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov ax,cx

and ax,0f00h ror ax,8

mov bl,al

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov ax,cx

and ax,00f0h ror ax,4

mov bl,al

call AsciiConv mov dl,bl

mov ah,02h int 21h

mov ax,cx

and ax,000fh mov bl,al

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov ah,4ch int 21h

AsciiConv proc cmp bl,0ah

jc skip add bl,07h

skip: add bl,30h ret

endp

AsciitoHex proc cmp al,41h

jc skippy sub al,07h

skippy: sub al,30h ret

endp Code ends end start

**Sample Input:** 05,03

**Sample Output:** 000F



# Experiment 2:

Division of two 8bit numbers (Using an assembly language program) [input as hex]

# Algorithm:

Step1: Start

Step2: Initialize data segment through AX register in the DS register. Step3: Read first digit in AL register through keyboard.

Step4: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step5: Move contents of AL register to a BL.

Step6: Rotate the contents of BL register by 4 positions at left side. Step7: Read a second digit in AL register through keyboard

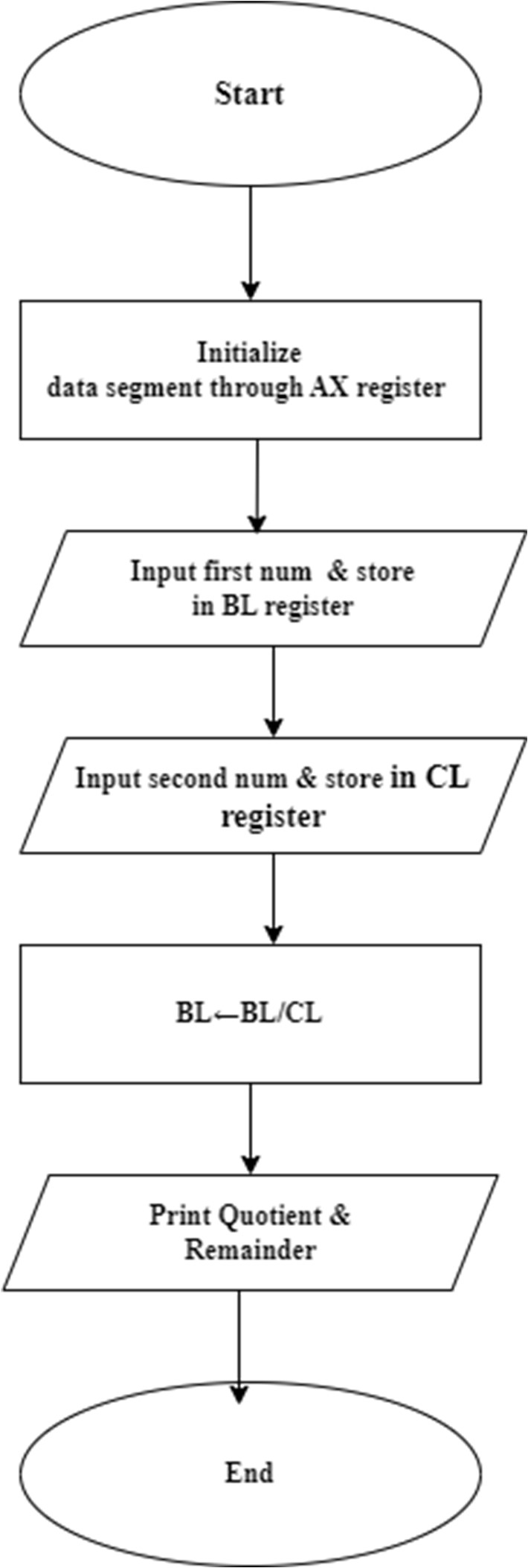
Step8: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step9: Add the contents of BL and AL store the result in BL.

Step10: Follow step 2 to 10 to input second number and store the result of number in CL Step11: Divide the contents of BL and CL and result gets stored in BL

Step12: Display the multiplication result in the screen Step13: Stop

# Flow Chart:



**Program Source Code:**

## Data segment

msg db 0dh,0ah,"Enter Dividend: $" msg1 db 0dh,0ah,"Enter Divisor: $" resq db 0dh,0ah,"The Quotient is: $" resr db 0dh,0ah,"The Remainder is: $"

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 ah,01h int 21h

call AsciitoHex mov bl,al

rol bl,4

mov ah,01h int 21h

call AsciitoHex add bl,al

mov dx,offset msg1 mov ah,09h

int 21h

mov ah,01h int 21h

call AsciitoHex mov cl,al

rol cl,4

mov ah,01h int 21h

call AsciitoHex add cl,al

and ax,0000h mov al,bl mov bl,cl

div bl mov cx,ax

mov dx,offset resq mov ah,09h

int 21h

mov ax,cx

and al,0f0h ror al,4

mov bl,al

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov ax,cx and al,0fh

mov bl,al

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov dx,offset resr mov ah,09h

int 21h

mov ax,cx

and ah,0f0h ror ah,4

mov bl,ah

call AsciiConv

mov dl,bl mov ah,02h int 21h

mov ax,cx and ah,0fh mov bl,ah

call AsciiConv

mov dl,bl mov ah,02h

int 21h

mov ah,4ch int 21h

AsciiConv proc cmp bl,0ah

jc skip add bl,07h

skip: add bl,30h ret

endp

**Sample Input:** 72,07

**Sample Output:** Quotient 10

## Remainder 02

**Experiment 3:**

Addition of two 8-bit numbers (Using an assembly language program). [input as hex]

# Algorithm:

Step1: Start

Step2: Initialize data segment through AX register in the DS register. Step3: Read first digit in AL register through keyboard.

Step4: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step5: Move contents of AL register to a BL.

Step6: Rotate the contents of BL register by 4 positions at left side. Step7: Read a second digit in AL register through keyboard AL=35h

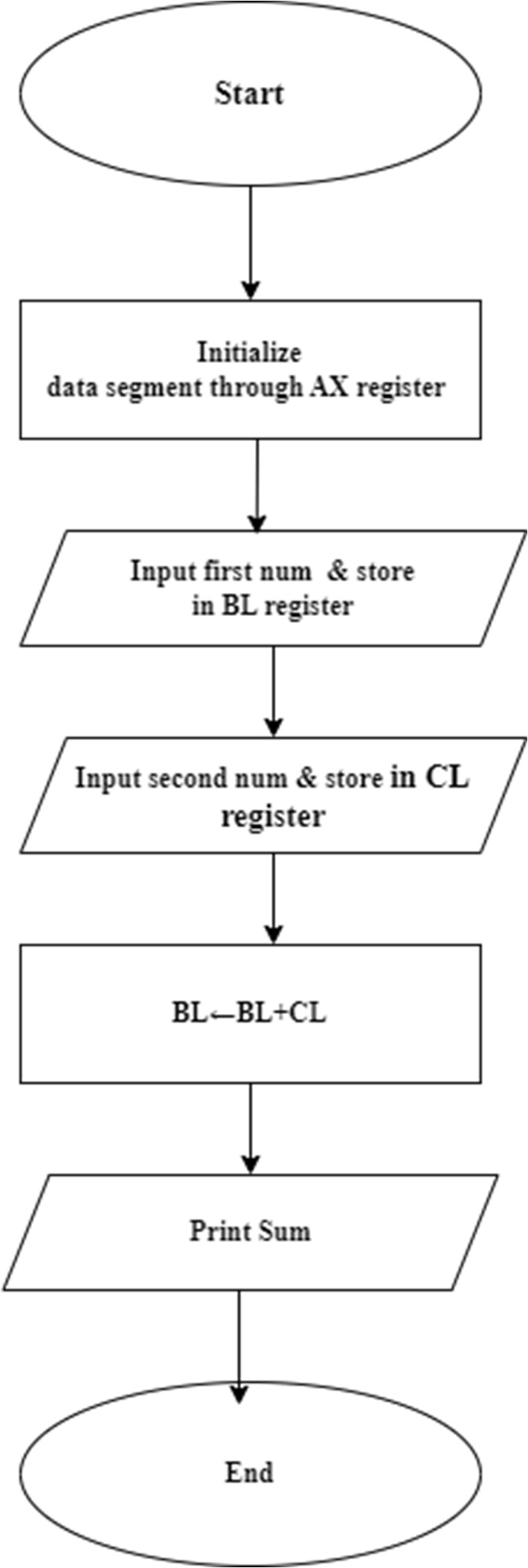
Step8: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step9: Add the contents of BL and AL store the result in BL.

Step10: Follow step 2 to 10 to input second number and store the result of number in CL Step11: Add the contents of BL and CL and result gets stored in BL

Step12: Display the addition result in the screen Step13: Stop

# Flow Chart:



**Program Source Code:**

Data segment

msg db 0dh,0ah,"Enter first number: $" msg1 db 0dh,0ah,"Enter second number: $" result db 0dh,0ah,"The sum is: $"

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 ah,01h ; int 21h

sub al,30h ; mov bl,al rol bl,4

mov ah,01h ; int 21h

sub al,30h ; add bl,al ;

mov dx,offset msg1 ; mov ah,09h

int 21h

mov ah,01h ; int 21h

sub al,30h ; mov cl,al rol cl,4

mov ah,01h ; int 21h

sub al,30h ; add cl,al ; add bl,cl ;

mov dx,offset result ; mov ah,09h

int 21h mov cl,bl ;

and bl,0f0h ; ror bl,4

call AsciiConv ; mov dl,bl ;

mov ah,02h

int 21h mov bl,cl ; and bl,0fh ;

call AsciiConv ;

mov dl,bl ; mov ah,02h int 21h

mov ah,4ch ; int 21h

AsciiConv proc ; cmp bl,0ah ;

jc skip add bl,07h

skip: add bl,30h ret

endp Code ends end star

**Sample Input:** 08,07

# Sample Output: 0F



**Experiment 4:**

Subtraction of two 8-bit numbers (Using an assembly language program). [input as hex]

# Algorithm:

Step1: Start

Step2: Initialize data segment through AX register in the DS register. Step3: Read first digit in AL register through keyboard.

Step4: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step5: Move contents of AL register to a BL.

Step6: Rotate the contents of BL register by 4 positions at left side. Step7: Read a second digit in AL register through keyboard AL=35h

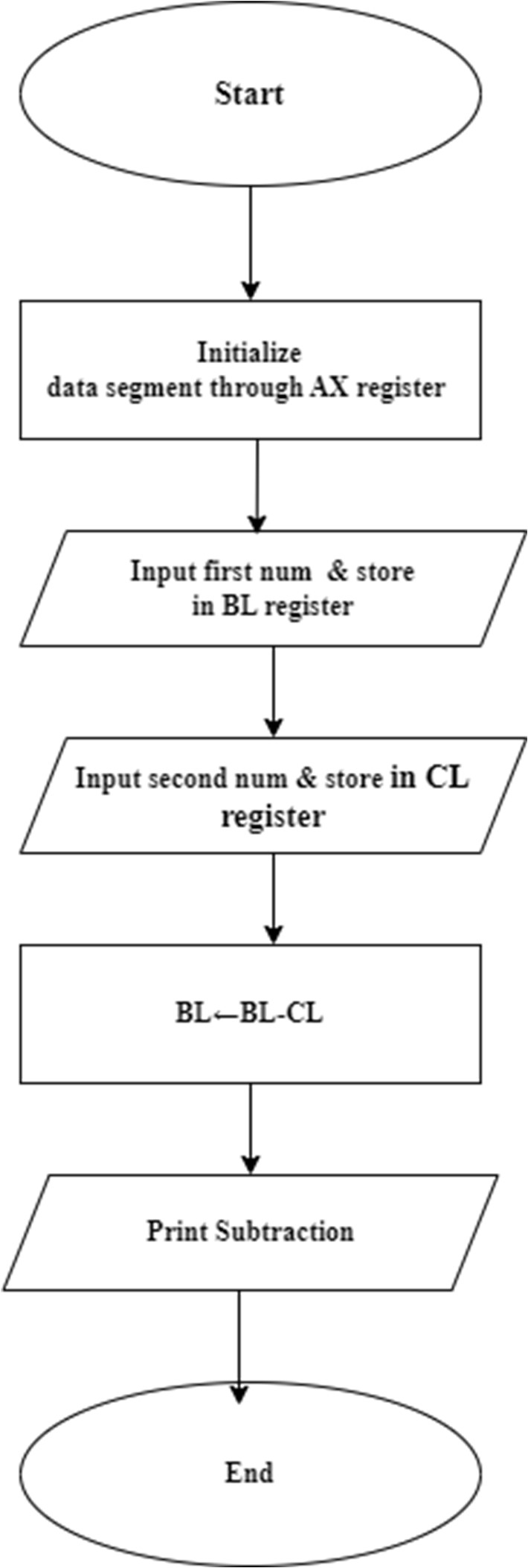
Step8: Call Input procedure to make a number from ASCII hexadecimal to a normal hexadecimal number.

Step9: Add the contents of BL and AL store the result in BL.

Step10: Follow step 2 to 10 to input second number and store the result of number in CL Step11: Subtract the contents of BL and CL and result gets stored in BL

Step12: Display the subtraction result in the screen Step13: Stop

# Flow Chart:



**Program Source Code:**

## Data segment

msg db 0dh,0ah,"Enter first number: $" msg1 db 0dh,0ah,"Enter second number: $" result db 0dh,0ah,"The Subtraction is: $"

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 ah,01h ; int 21h

sub al,30h ; mov bl,al rol bl,4

mov ah,01h ; int 21h

sub al,30h ; add bl,al ;

mov dx,offset msg1 ; mov ah,09h

int 21h

mov ah,01h ;

int 21h

sub al,30h ; mov cl,al rol cl,4

mov ah,01h ; int 21h

sub al,30h ; add cl,al ; sub bl,cl ;

mov dx,offset result ; mov ah,09h

int 21h mov cl,bl ;

and bl,0f0h ; ror bl,4

call AsciiConv ; mov dl,bl ;

mov ah,02h

int 21h mov bl,cl ; and bl,0fh ;

call AsciiConv ; mov dl,bl ;

mov ah,02h int 21h

mov ah,4ch ; int 21h

AsciiConv proc ; cmp bl,0ah ;

jc skip add bl,07h

skip: add bl,30h ret

endp Code ends end start

**Sample Input:** 0F,0A

# Sample Output: 05

