

Experiment No 2: 16-bit Arithmetic Operations

Date: 28-08-2020

NAME: Gayathri M

REG.NO: 185001050

1. AIM:

Program for adding 2, 16-bit numbers.

ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load operand-1 to ax and operand-2 to bx
- Load 00h to ch register for carry
- Add ax and bx
- If there is no carry being generated, goto here segment else, increment ch by 1 and go to here segment
- In here segment,
 - Load ax to result
 - Load ch to carry
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
Start:	
mov ax,data mov ds,ax	Transferring address of data segment to ds
mov ax,opr1	Value of opr1 is loaded to ax
mov bx,opr2	Value of opr2 is loaded to bx
mov ch,00h	Initializing the value of ch
add ax,bx	ax=ax+bx
jnc here	Jump to "here" segment if no carry is generated
inc ch	Increments ch by 1

Here:	
mov result,ax	Load register value of ax to result
mov carry,ch	Load ch value to carry
mov ah,4ch int 21h	Termination of execution
code ends	Ending the segment with the segment name

UNASSEMBLED CODE:

The screenshot shows a DOSBox 0.74-3 window running 16ADD.EXE. The CPU speed is 3000 cycles. A Notepad window displays the assembly code for 16ADD.ASM. The code includes data and code segments with various instructions like MOV, ADD, JNC, INC, and INT. A DOSBox Status Window is also visible on the right.

```

(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [16ADD.EXE]:
List File [NUL.MAP1]:
Libraries [.LIB1]:
Warning: No STACK segment

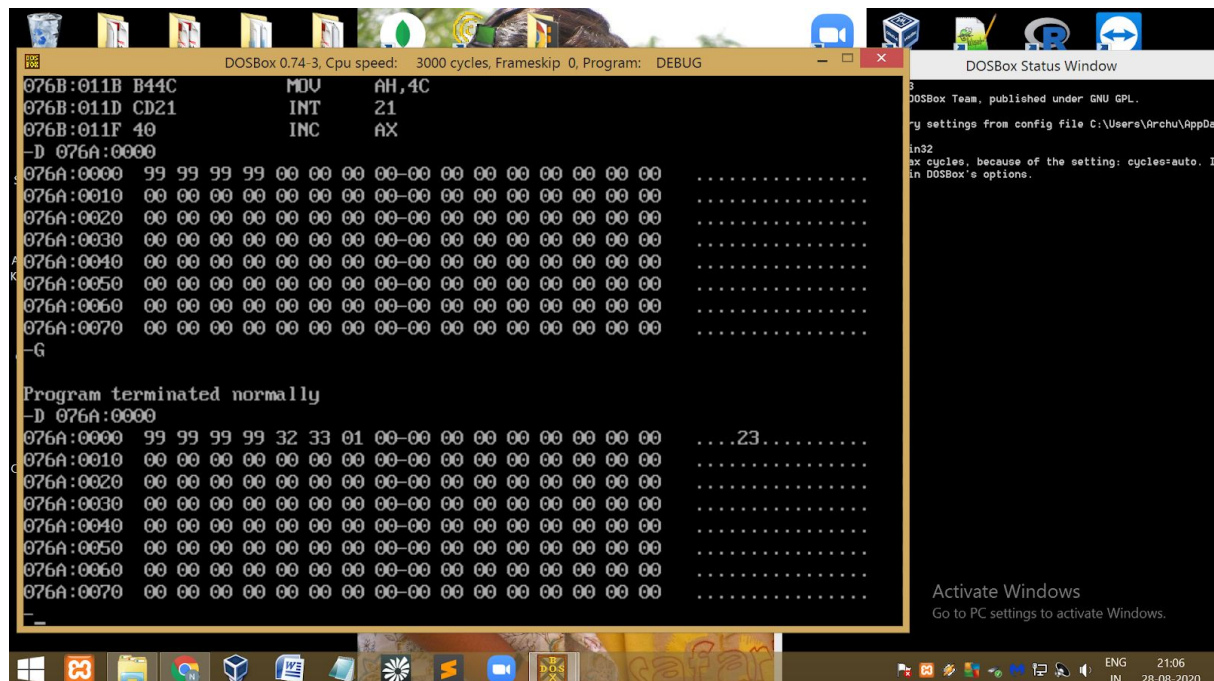
There was 1 error detected.

E:\>DEBUG 16ADD.EXE
-U
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 A10000      MOV     AX,[0000]
076B:0108 8B1E0200    MOV     BX,[0002]
076B:010C B500        MOV     CH,00
076B:010E 03C3        ADD     AX,BX
076B:0110 7302        JNB     0114
076B:0112 FEC5        INC     CH
076B:0114 A30400      MOV     [0004],AX
076B:0117 8B2E0600    MOV     [0006],CH
076B:011B B44C        MOV     AH,4C
076B:011D CD21        INT     21
076B:011F 40          INC     AX

assume cs:code,ds:data
data segment
    opr1 dw 9999h
    opr2 dw 9999h
    result dw 00H
    carry db 00H
data ends
code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov ax,opr1
        mov bx,opr2
        mov ch,00h
        add ax,bx
        jnc here
        inc ch
here:   mov result,ax
        mov carry,ch
        mov ah,4ch
        int 21h
        code ends
end start
  
```

SAMPLE INPUT/OUTPUT:

(ax=9999; bx=9999)



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
076B:011B B44C      MOV     AH,4C
076B:011D CD21      INT     21
076B:011F 40        INC     AX
-D 076A:0000
076A:0000 99 99 99 99 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
-G
Program terminated normally
-D 076A:0000
076A:0000 99 99 99 99 32 33 01 00-00 00 00 00 00 00 00 00 ....23.....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
-
```

RESULT:

The addition of 2, 16-bit numbers is thus shown.

2. AIM:

Program for subtracting 2, 16-bit numbers.

ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load operand-1 to ax and operand-2 to bx
- Load 00h to ch register and subtract ax and bx
- If ax is greater than bx, goto here segment else, increment ch by 1 and find the 2's complement of ah and goto segment here
- In here segment,
 - Load ax to result
 - Load ch to carry
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
Start: mov ax,data mov ds,ax mov ax,opr1 mov bx,opr2 mov ch,00h sub ax,bx jnc here inc ch neg ah	Transferring address of data segment to ds Value of opr1 is loaded to ax Value of opr2 is loaded to bx Initializing the value of ch ax=ax-bx Jump to "here" segment if ax>bx Increments ch by 1 2's complement of ah
Here: mov result,ax mov carry,ch mov ah,4ch int 21h code ends	Load register value of ax to result Load ch value to carry Termination of execution Ending the segment with the segment name

UNASSEMBLED CODE:

The screenshot shows a DOSBox emulator window with the following content:

```

(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [16SUB.EXE]:
List File [INUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 16SUB.EXE
-U
076B:0100 B8A07      MDU    AX,076A
076B:0103 8ED8      MDU    DS,AX
076B:0105 A10000     MDU    AX,[0000]
076B:0108 8B1E0200   MDU    BX,[0002]
076B:010C B500      MDU    CH,00
076B:010E 2BC3      SUB    AX,BX
076B:0110 7304      JNB    0116
076B:0112 FEC5      INC    CH
076B:0114 F7D8      NEG    AX
076B:0116 A30400     MDU    [0004],AX
076B:0119 882E0600   MDU    [0006],CH
076B:011D B44C      MDU    AH,4C
076B:011F CD21      INT    21
  
```

The Notepad window shows the source code of 16SUB.ASM:

```

File Edit Format View Help
assume cs:code,ds:data
data segment
    opr1 dw 5600h
    opr2 dw 4300h
    result dw 0000h
    carry db 00h
data ends
code segment
    org 0100h
start: mov ax,data
       mov ds,ax
       mov ax,opr1
       mov bx,opr2
       mov ch,00h
       sub ax,bx
       jnc here
       inc ch
       neg ax
here:  mov result,ax
       mov carry,ch
       mov ah,4ch
       int 21h
       code ends
  
```

The DOSBox Status Window shows the following information:

```

DOSBox Status Window

SBox Team, published under GNU GPL.
settings from config file C:\Users\Archu\AppData
32
cycles, because of the setting: cycles=auto.
DOSBox's options.
  
```

The taskbar at the bottom shows various icons, including the Windows logo, a clock showing 21:09, and a date showing 28-08-2020.

SAMPLE INPUT/OUTPUT

ax=4300; bx=5600 (ax<bx)

The screenshot shows a Windows 10 desktop with a taskbar at the bottom containing icons for Windows, Edge, File Explorer, Google Chrome, OneDrive, and several other applications. Two windows are open:

- DOSBox 0.74-3**: A window titled "DOSBox 0.74-3. Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG". It displays assembly code and memory dump:


```

076B:0119 882E0600      MOV     [0006],CH
076B:011D B44C       MOV     AH,4C
076B:011F CD21       INT     21
-D 076A:0000

076A:0000  00 56 00 43 00 00 00 00-00 00 00 00 00 00 00 00  .U.C.....
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
-G

Program terminated normally
-D 076A:0000

076A:0000  00 56 00 43 00 13 00 00-00 00 00 00 00 00 00 00  .U.C.....
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
      
```
- DOSBox Status Window**: A window titled "DOSBox Status Window" showing:
 - OSBox Team, published under GNU GPL.
 - settings from config file C:\Users\Archu\AppData\Local\DOSBox\config.txt
 - cycles, because of the setting: cycles=auto.
 - DOSBox's options.

At the bottom right, a Windows activation message is visible: "Activate Windows. Go to PC settings to activate Windows."

RESULT:

The subtraction of 2, 16-bit numbers is thus shown.

3. AIM:

Program for multiplication of 2, 16-bit numbers.

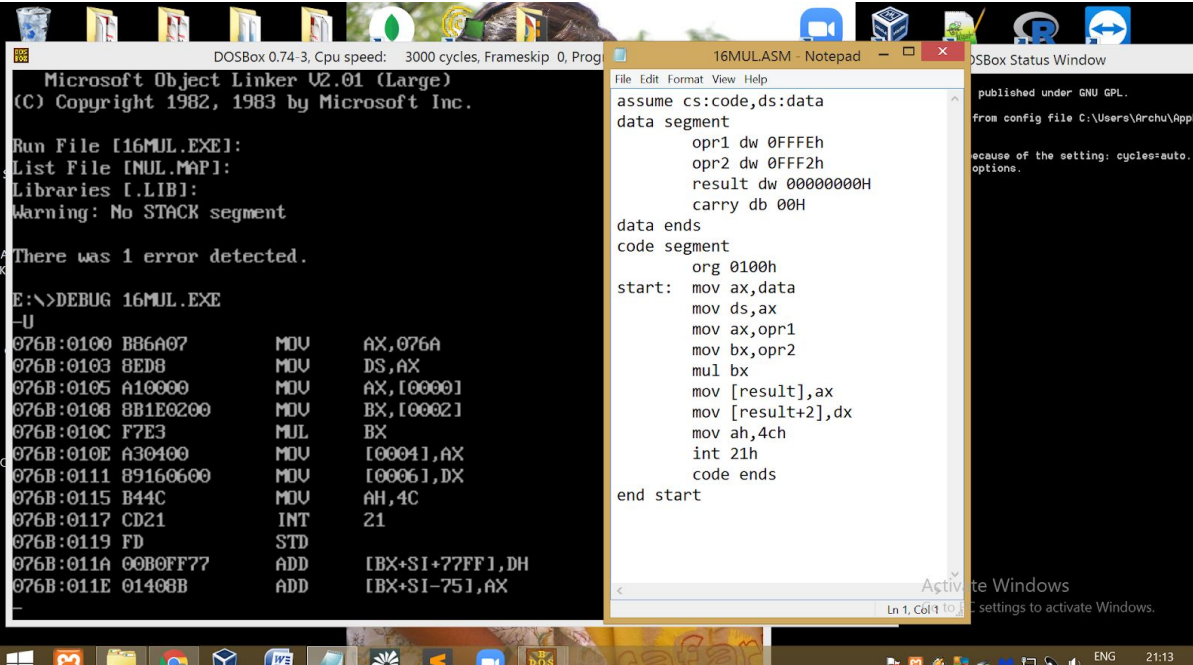
ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load operand-1 to ax and operand-2 to bx
- Multiply bx (dxax=ax x bx)
- Load ax to result
- Load dx to location result+2
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
Start: mov ax,data mov ds,ax	Transferring address of data segment to ds
mov ax,opr1	Value of opr1 is loaded to ax
mov bx,opr2	Value of opr2 is loaded to bx
mul bx	dxax=ax x bx
mov [result],ax	Load register value of ax to result
mov[result+2],dx	Load register value of dx to location [result+2]
mov ah,4ch int 21h	Termination of execution
code ends	Ending the segment with the segment name

UNASSEMBLED CODE:



The screenshot shows a DOSBox window with the linker output and a Notepad window displaying the assembly code for 16MUL.EXE.

Linker Output (DOSBox window):

```
Microsoft Object Linker U2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [16MUL.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

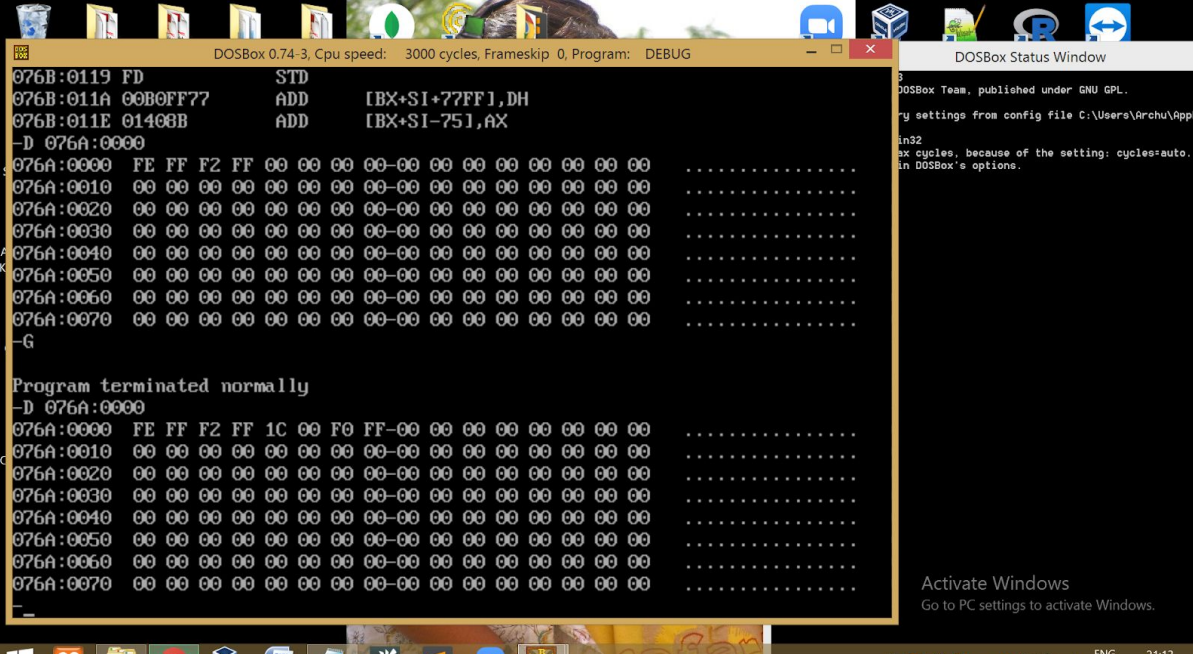
E:\>DEBUG 16MUL.EXE
-U
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 A10000      MOV     AX,[0000]
076B:0108 8B1E0200     MOV     BX,[0002]
076B:010C F7E3        MUL     BX
076B:010E A30400      MOV     [0004],AX
076B:0111 89160600     MOV     [0006],DX
076B:0115 B44C        MOV     AH,4C
076B:0117 CD21        INT     21
076B:0119 FD         STD
076B:011A 00B0FF77     ADD     [BX+SI+77FF],DH
076B:011E 01408B     ADD     [BX+SI-75],AX

Program terminated normally
-D 076A:0000
076A:0000 FE FF F2 FF 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
-G
```

Assembly Code (Notepad window):

```
File Edit Format View Help
assume cs:code,ds:data
data segment
    opr1 dw 0FFFFh
    opr2 dw 0FFF2h
    result dw 0000000H
    carry db 00H
data ends
code segment
    org 0100h
start:  mov ax,data
        mov ds,ax
        mov ax,opr1
        mov bx,opr2
        mul bx
        mov [result],ax
        mov [result+2],dx
        mov ah,4ch
        int 21h
        code ends
end start
```

SAMPLE INPUT/OUTPUT (ax=FFFE ; bx=FFF2)



The screenshot shows the DOSBox window displaying the memory dump and program termination.

Memory Dump (DOSBox window):

```
076B:0119 FD         STD
076B:011A 00B0FF77     ADD     [BX+SI+77FF],DH
076B:011E 01408B     ADD     [BX+SI-75],AX
-D 076A:0000
076A:0000 FE FF F2 FF 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
-G

Program terminated normally
-D 076A:0000
076A:0000 FE FF F2 FF 1C 00 F0 FF-00 00 00 00 00 00 00 00 .....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
```

RESULT:

The multiplication of 2, 16-bit numbers is thus shown.

4. AIM:

Program for division of 2, 16-bit numbers.

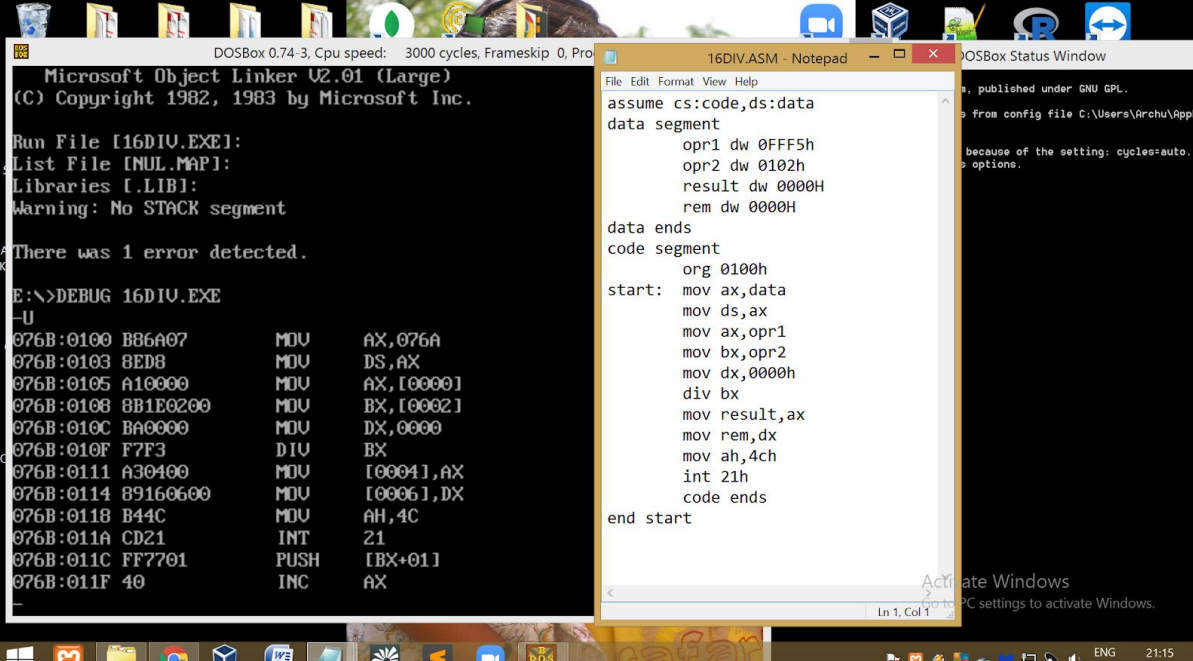
ALGORITHM:

- Initialize the data segment
- Move data segment address to ds
- Load operand-1 to ax and operand-2 to bx
- Load dx with 0000h
- Divide bx ($ax = dxax / bx$; remainder in dx)
- Load ax to result
- Load dx to rem (remainder)
- Terminate the program

PROGRAM:

PROGRAM	COMMENTS
Start:	
mov ax,data mov ds,ax	Transferring address of data segment to ds
mov dx,0000h	Register dx is loaded with 0000
mov ax,opr1	Value of opr1 is loaded to ax
mov bx,opr2	Value of opr2 is loaded to bx
div bx	$ax = dxax / bx$
mov result,ax	Load register value of ax to result
mov rem,dx	Load register value of dx to rem
mov ah,4ch int 21h	Termination of execution
code ends	Ending the segment with the segment name

UNASSEMBLED CODE:



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: 16DIV.EXE

Microsoft Object Linker U2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [16DIV.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
Warning: No STACK segment

There was 1 error detected.

E:\>DEBUG 16DIV.EXE
-U

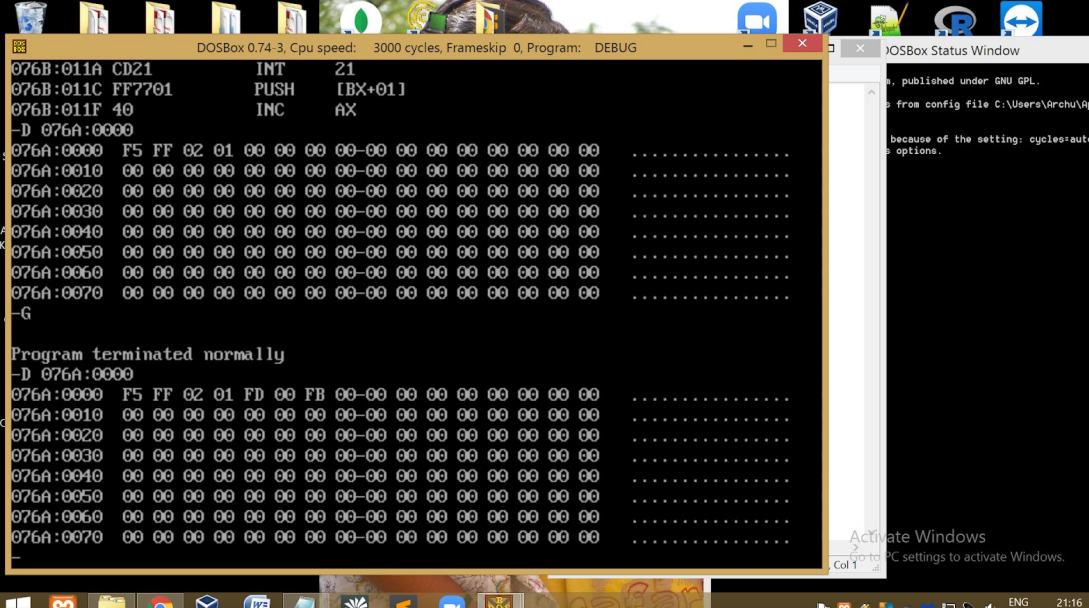
Address	Disassembly	Comment
076B:0100	B86A07	MOV AX,076A
076B:0103	8ED8	MOV DS,AX
076B:0105	A10000	MOV AX,[0000]
076B:0108	8B1E0200	MOV BX,[0002]
076B:010C	BA0000	MOV DX,0000
076B:010F	F7F3	DIV BX
076B:0111	A30400	MOV [0004],AX
076B:0114	89160600	MOV [0006],DX
076B:0118	B44C	MOV AH,4C
076B:011A	CD21	INT 21
076B:011C	FF7701	PUSH [BX+01]
076B:011F	40	INC AX

16DIV.ASM - Notepad

```
assume cs:code,ds:data
data segment
    opr1 dw 0FFF5h
    opr2 dw 0102h
    result dw 0000H
    rem dw 0000H
data ends
code segment
    org 0100h
start: mov ax,data
        mov ds,ax
        mov ax,opr1
        mov bx,opr2
        mov dx,0000h
        div bx
        mov result,ax
        mov rem,dx
        mov ah,4ch
        int 21h
        code ends
end start
```

SAMPLE INPUT/OUTPUT

(ax=FFF5 ; bx=0102)



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG

076B:011A CD21 INT 21
076B:011C FF7701 PUSH [BX+01]
076B:011F 40 INC AX

-D 076A:0000

Address	Disassembly	Comment
076A:0000	F5 FF 02 01 00 00 00 00 00 00 00 00 00 00 00 00
076A:0010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

-G

Program terminated normally

-D 076A:0000

Address	Disassembly	Comment
076A:0000	F5 FF 02 01 FD 00 FB 00 00 00 00 00 00 00 00 00
076A:0010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

RESULT:

The division of 2,16-bit numbers is thus shown.