16 – BIT ARITHMETIC OPERATIONS

Exp No.: 2 **Name:** S Vishakan

Date: 26-08-2020 **Reg. No:** 18 5001 196

AIM:

To write assembly language programs to perform 16-bit arithmetic operations and execute them.

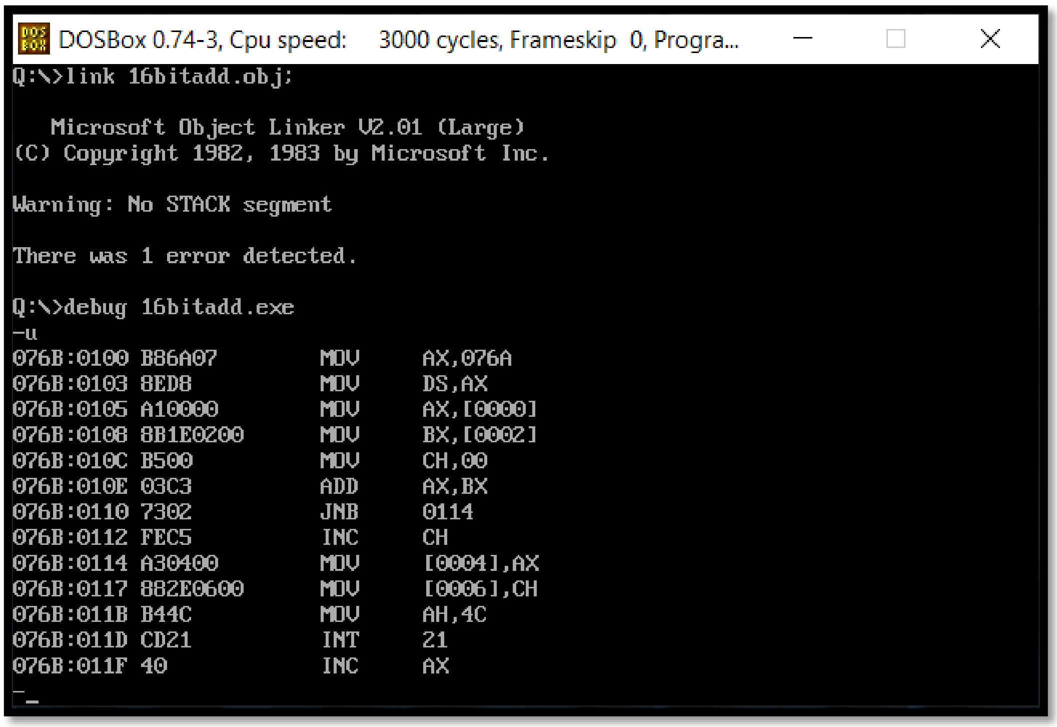
ALGORITHM:

- Begin.
- Open data segment.
- Initialize data segment with required operands, data types and values.
- Close the data segment.
- Open code segment.
- Set a preferred offset (preferably 100)
- Load the data segment content into AX register.
- Transfer the contents of AX register to DS register.
- Do the required operation (ADD, SUB, MUL, DIV) on the registers.
    - Jump (whenever ever carry/ overflow is a possibility)
    - Increment carry(add) or negate the value. (2's complement)
- Introduce an interrupt for safe exit. (INT 21h)
- Close the code segment.
- End.

PROGRAM – 1: 16 – BIT ADDITION:

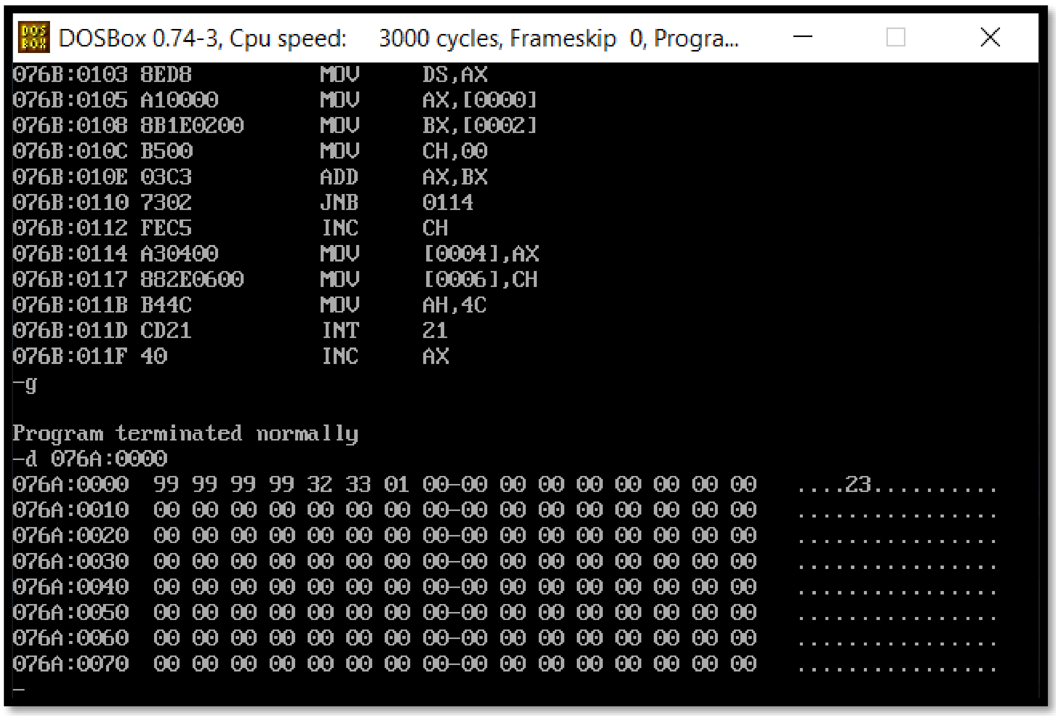
PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 dw 9999h	Stores operand 1.
opr2 dw 9999h	Stores operand 2.
result dw 0000h	Stores the result of the operation.
carry db 00h	Stores the carry, if any.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from memory location [0000] and [0001] to AL AND AH respectively.
mov ds, ax	Transfer data from memory location AX to DS.
mov ax, opr1	Transfer value of opr1 to AX.
mov bx, opr2	Transfer value of opr2 to BX.
mov ch, 00h	CH = 0.
add ax, bx	AX = AX + BX.
jnc here	Jump if no carry to “here”. Else, continue.
inc ch	CH = CH + 1
here: mov result, ax	Transfer value of AX to result.
mov carry, ch	Transfer value of CH to carry.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

UNASSEMBLED CODE:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Q:\>link 16bitadd.obj;

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>debug 16bitadd.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 882E0600      MOV     [0006],CH
076B:011B B44C          MOV     AH,4C
076B:011D CD21          INT     21
076B:011F 40            INC     AX
-_
```

SAMPLE I/O SNAPSHOT:

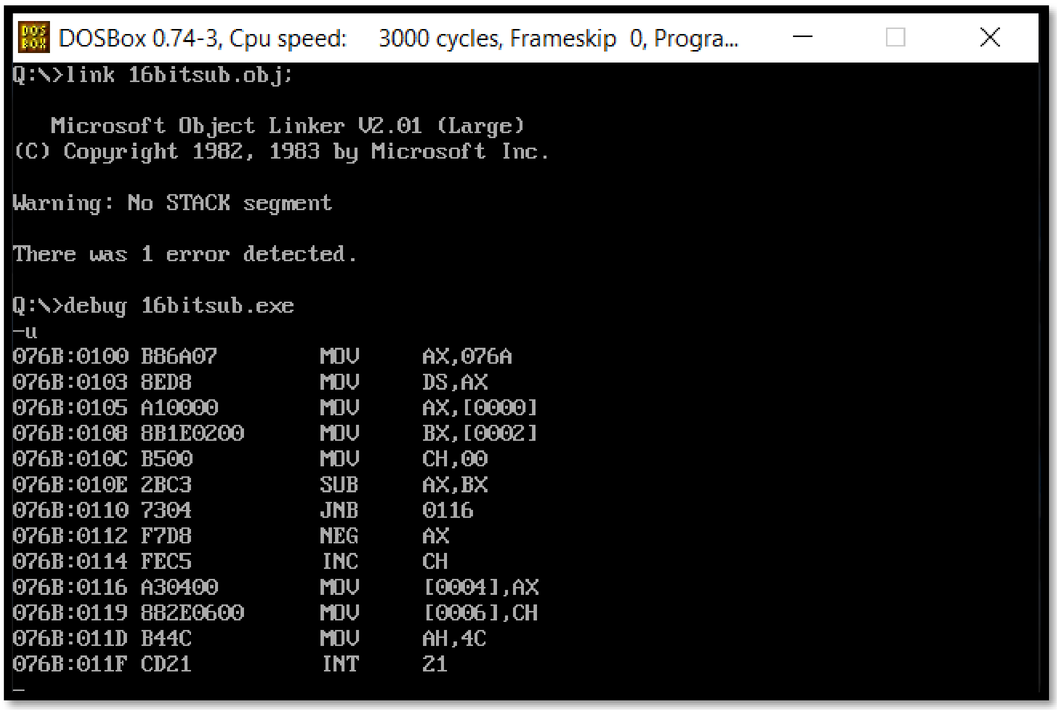

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
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 882E0600      MOV     [0006],CH
076B:011B B44C          MOV     AH,4C
076B:011D CD21          INT     21
076B:011F 40            INC     AX
-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   ................
-
```

PROGRAM – 2: 16 – BIT SUBTRACTION:

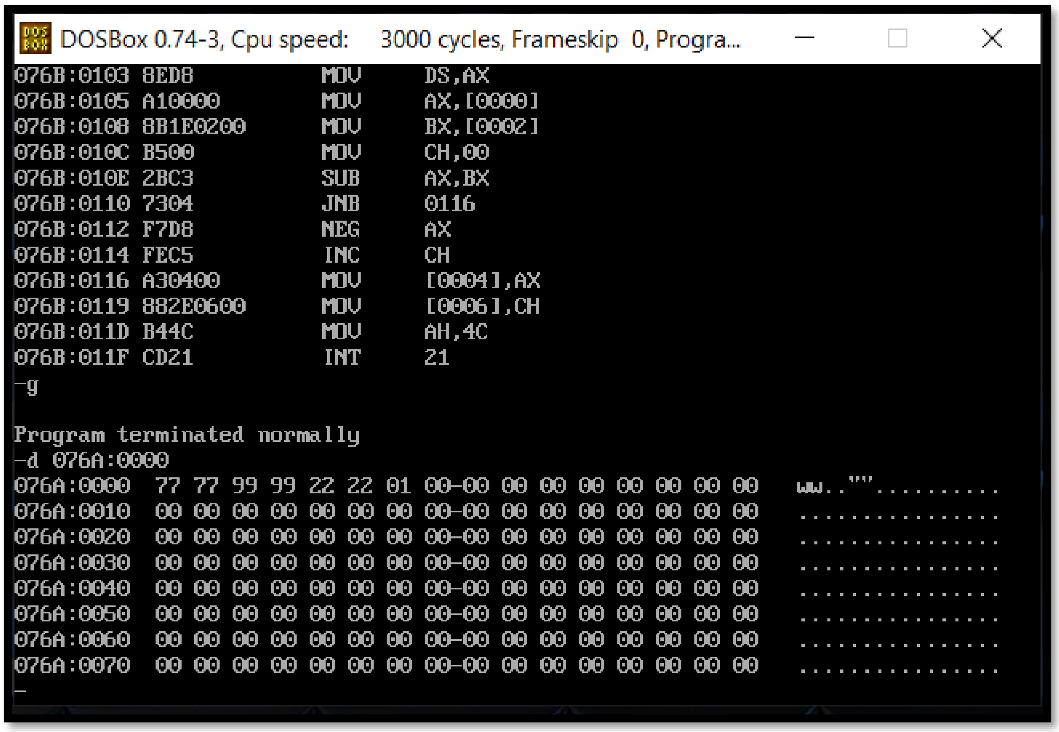
PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 dw 7777h	Stores operand 1.
opr2 dw 9999h	Stores operand 2.
diff dw 0000h	Stores the result of the operation.
sign db 00h	Stores the sign bit.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from memory location [0000] and [0001] to AL AND AH respectively.
mov ds, ax	Transfer data from memory location AX to DS.
mov ax, opr1	Transfer value of opr1 to AX.
mov bx, opr2	Transfer value of opr2 to BX.
mov ch, 00h	CH = 0.
sub ax, bx	AX = AX – BX.
jnc here	Jump if no sign change to “here”. Else, continue.
neg ax	Take 2’s Complement if negative value.
inc ch	CH = CH + 1
here: mov diff, ax	Transfer value of AX to diff.
mov sign, ch	Transfer value of CH to sign.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

UNASSEMBLED CODE:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Q:\>link 16bitsub.obj;

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>debug 16bitsub.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 2BC3          SUB     AX,BX
076B:0110 7304          JNB     0116
076B:0112 F7D8          NEG     AX
076B:0114 FEC5          INC     CH
076B:0116 A30400        MOV     [0004],AX
076B:0119 882E0600      MOV     [0006],CH
076B:011D B44C          MOV     AH,4C
076B:011F CD21          INT     21
-
```

SAMPLE I/O SNAPSHOT:

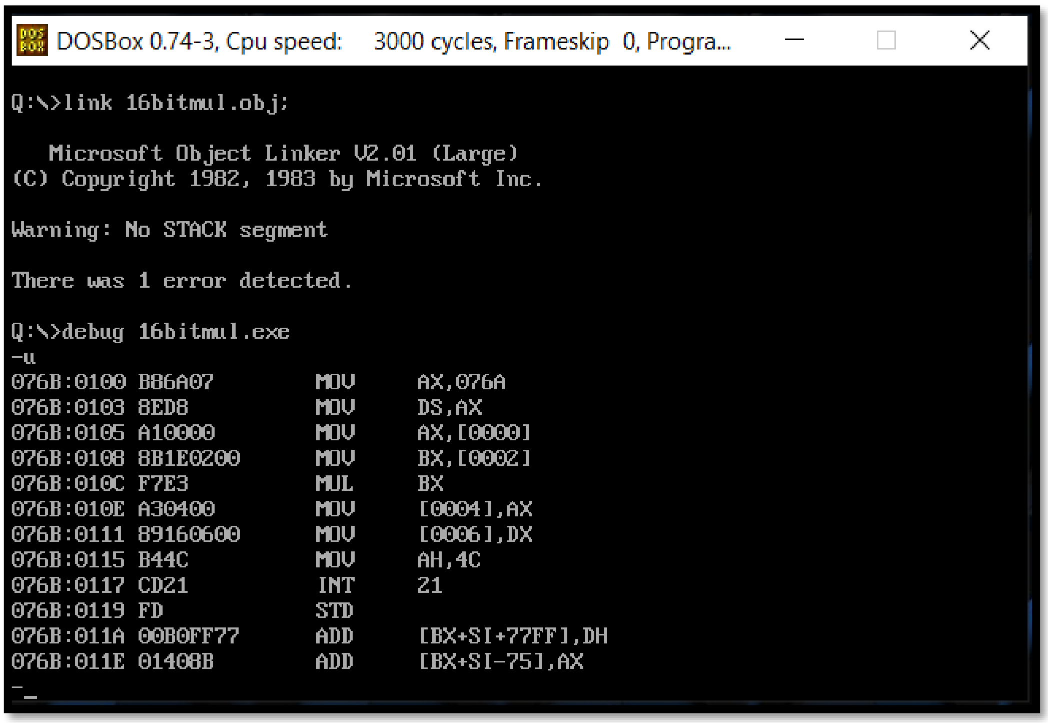

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
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 2BC3          SUB     AX,BX
076B:0110 7304          JNB     0116
076B:0112 F7D8          NEG     AX
076B:0114 FEC5          INC     CH
076B:0116 A30400        MOV     [0004],AX
076B:0119 882E0600      MOV     [0006],CH
076B:011D B44C          MOV     AH,4C
076B:011F CD21          INT     21
-g

Program terminated normally
-d 076A:0000
076A:0000  77 77 99 99 22 22 01 00-00 00 00 00 00 00 00 00   ww..""..........
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   ................
-
```

PROGRAM – 3: 16 – BIT MULTIPLICATION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 dw 1000h	Stores operand 1.
opr2 dw 1000h	Stores operand 2.
product1 dw 0000h	Stores the lower 16 bits of the operation.
product2 dw 0000h	Stores the higher 16 bits of the operation.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from memory location [0000] and [0001] to AL AND AH respectively.
mov ds, ax	Transfer data from memory location AX to DS.
mov ax, opr1	Transfer value of opr1 to AX.
mov bx, opr2	Transfer value of opr2 to BX.
mul bx	DXAX = AX * BX.
mov product1, ax	Transfer value of AX to product1.
mov product2, dx	Transfer value of DX to product2.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

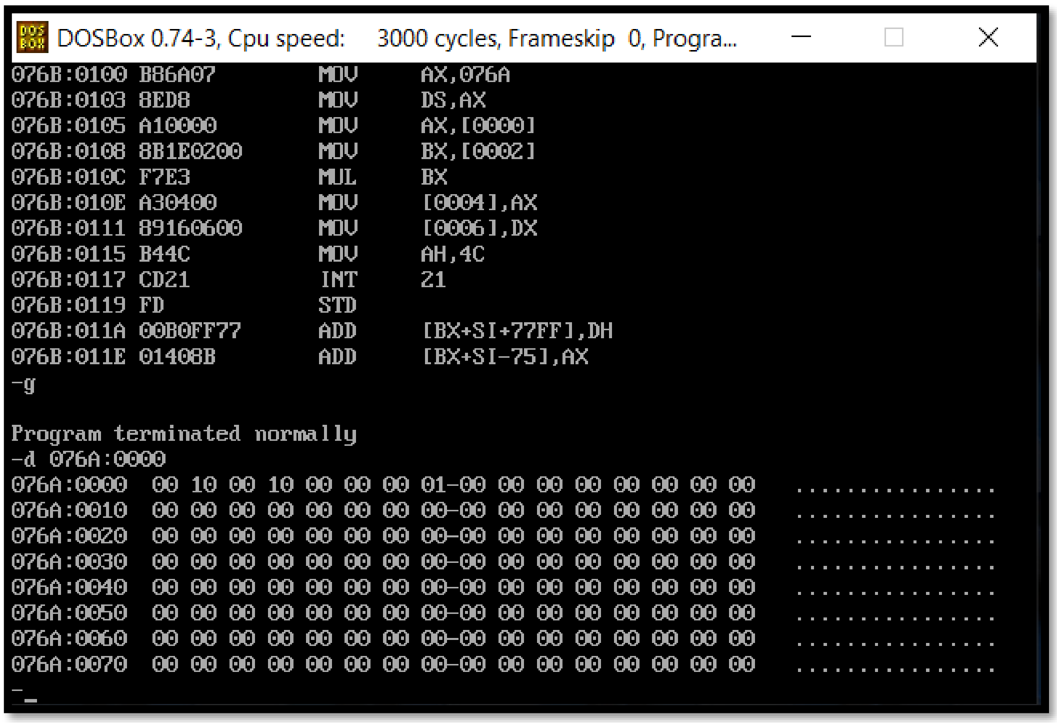
UNASSEMBLED CODE:



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Q:\>link 16bitmul.obj;

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>debug 16bitmul.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
-_
```

SAMPLE I/O SNAPSHOT:



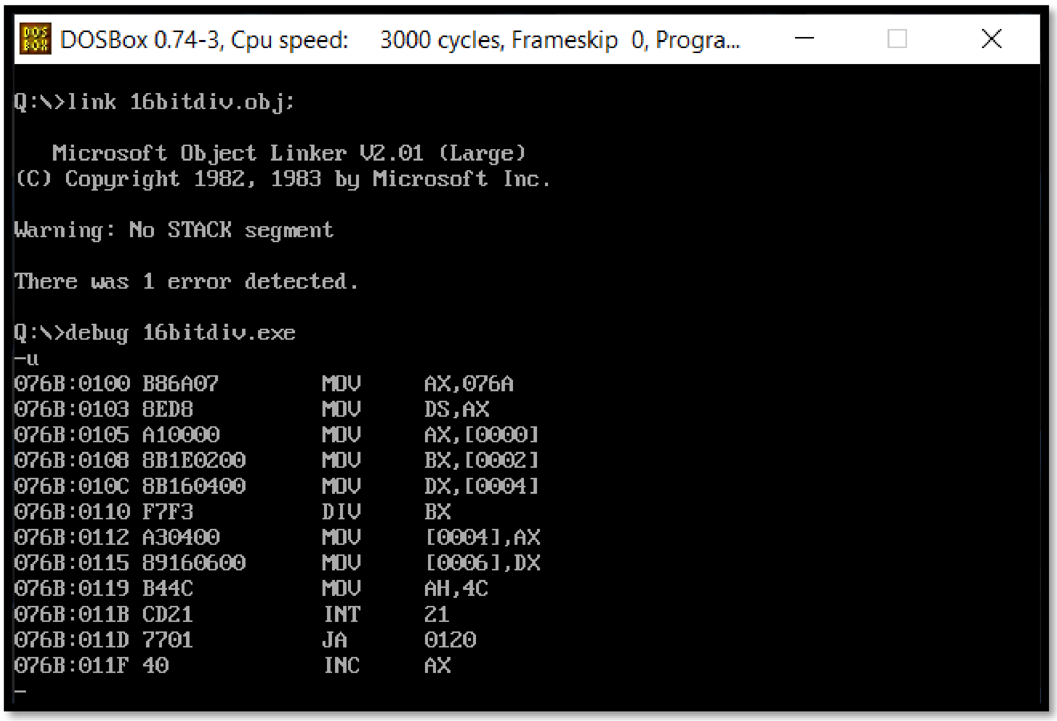
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
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
-g

Program terminated normally
-d 076A:0000
076A:0000  00 10 00 10 00 00 00 01-00 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   ................
-_
```

PROGRAM – 4: 16 – BIT DIVISION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 dw 1000h	Stores the dividend.
opr2 dw 0900h	Stores the divisor.
quot dw 0000h	Stores the quotient of the division.
rem dw 0000h	Stores the remainder of the division.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from memory location [0000] and [0001] to AL AND AH respectively.
mov ds, ax	Transfer data from memory location AX to DS.
mov ax, opr1	Transfer value of dividend to AX.
mov bx, opr2	Transfer value of divisor to BX.
mov dx, quot	Transfer value of quotient (0000h) to DX.
div bx	AX = DXAX / BL. (AX has quotient, DX has remainder)
mov quot, ax	Transfer value of AX to quot.
mov rem, dx	Transfer value of DX to rem.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

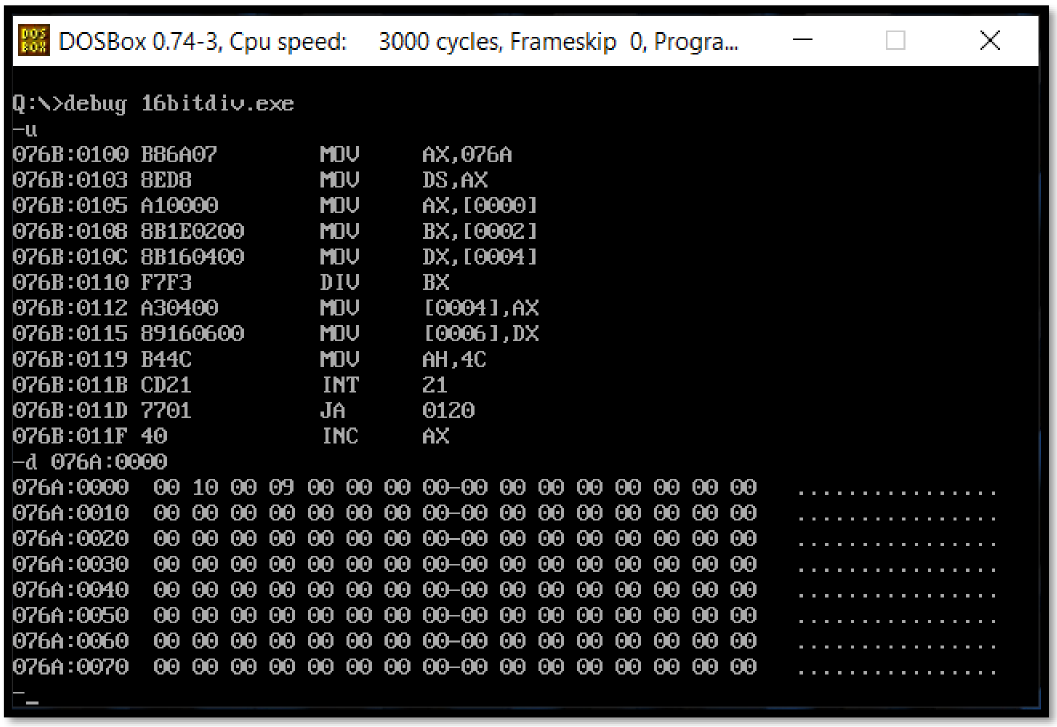
UNASSEMBLED CODE:



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Q:\>link 16bitdiv.obj;

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>debug 16bitdiv.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 8B160400      MOV     DX,[0004]
076B:0110 F7F3          DIV     BX
076B:0112 A30400        MOV     [0004],AX
076B:0115 89160600      MOV     [0006],DX
076B:0119 B44C          MOV     AH,4C
076B:011B CD21          INT     21
076B:011D 7701          JA      0120
076B:011F 40            INC     AX
-
```

SAMPLE I/O SNAPSHOT:



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Q:\>debug 16bitdiv.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 8B160400      MOV     DX,[0004]
076B:0110 F7F3          DIV     BX
076B:0112 A30400        MOV     [0004],AX
076B:0115 89160600      MOV     [0006],DX
076B:0119 B44C          MOV     AH,4C
076B:011B CD21          INT     21
076B:011D 7701          JA      0120
076B:011F 40            INC     AX
-d 076A:0000
076A:0000  00 10 00 09 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   ................
-_
```

RESULT:

The assembly level programs were written to perform the 16 – bit arithmetic operations and compiled. The results were observed and noted down.