UCS1512 – Microprocessors Lab

16 BIT ARITHMETIC OPERATIONS

Exp no : 2 Name: Sreedhar V

Date : 28-08-2020 Reg no: 185001161

AIM:

To program and execute the 16 bit arithmetic operations like addition, subtraction, multiplication and division in 8086 using an emulator.

16 - Bit Addition:

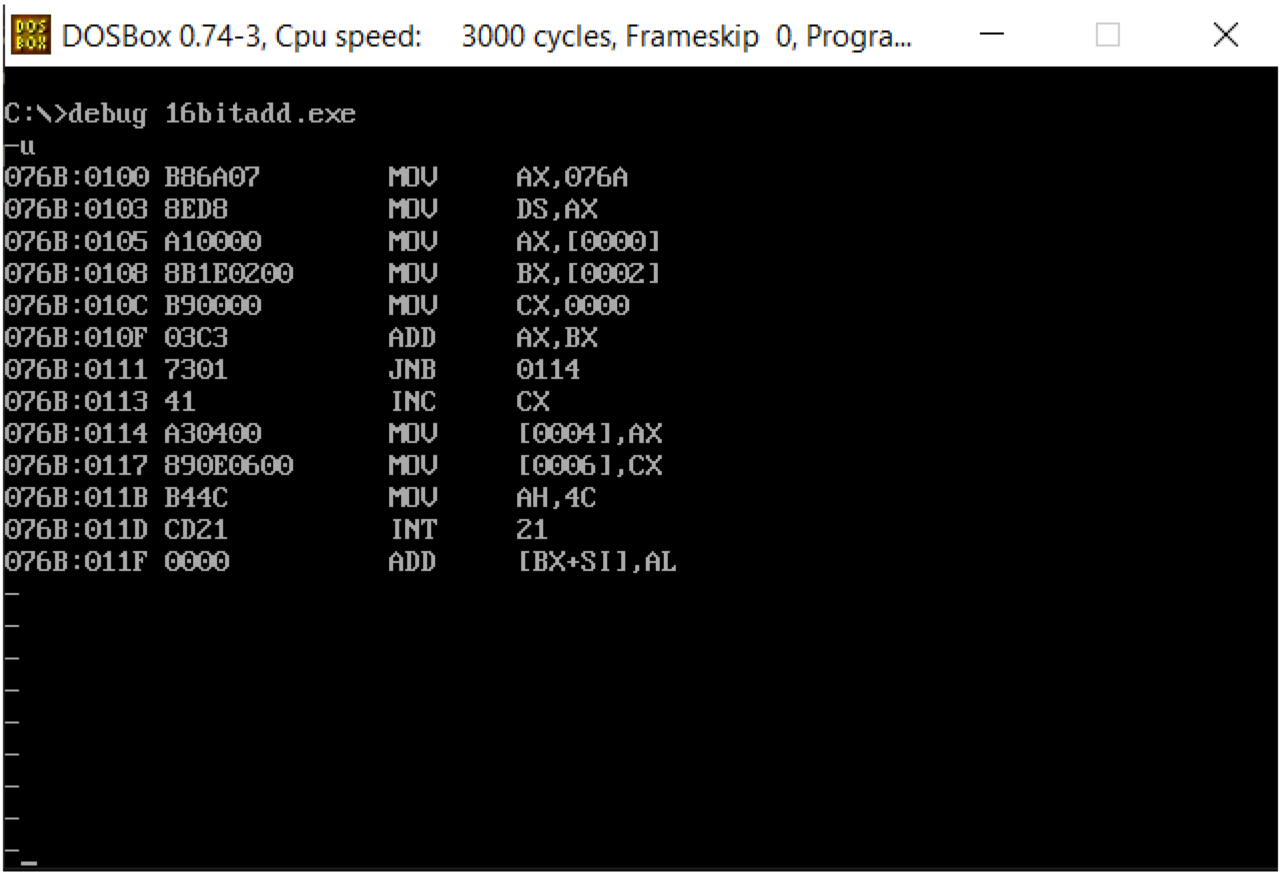
Algorithm:

- Program is set to run from any specified memory position.
- Load data from opr1 to register AX (first number)
- Load data from opr2 to register BX (second number)
- Add these two numbers (contents of register AX and register BX)
- Initialise carry to 0.
- Jump to final steps if there is no carry.
- Increment carry.
- Store additional values to result.
- Terminate the program.

Program:

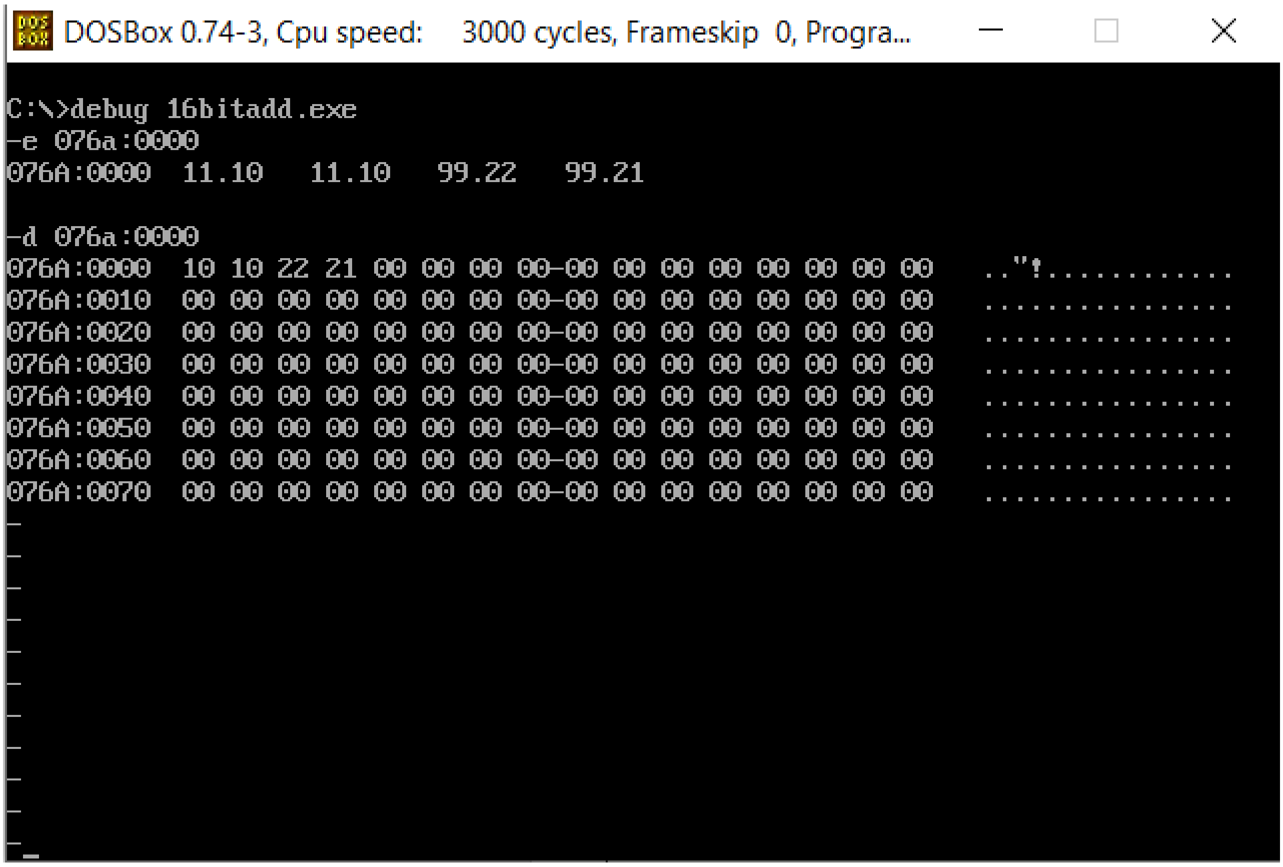
CODE	COMMENT
;Program for adding 2, 16 bit numbers	
assume cs:code,ds:data	
data segment	Data segment initialized
opr1 dw 1111h	opr1 initialised and set to 1111
opr2 dw 9999h	opr2 initialised and set to 9999
result dw 0000H	result initialised and set to 0000
carry dw 0000H	carry initialised and set to 0000
data ends	
code segment	Code segment begins
org 0100h	Originating address is set at 0100
start:	
mov ax,data	Address of data segment moved to ax
mov ds,ax	From ax, transferred to ds
mov ax,opr1	Value of opr1 transferred to ax
mov bx,opr2	Value of opr2 transferred to bx
mov cx,0000h	cx is initialised and set to 0
add ax,bx	Addition takes palce
jnc here	Junction created • If carry is not generated jump to here
inc cx	• Else increment cx
here:	
mov result,ax	data transferred from ax to result
mov carry,cx	data transferred from cx to carry
mov ah,4ch	
int 21h	Program terminates
code ends	
end start	

Unassembled code:

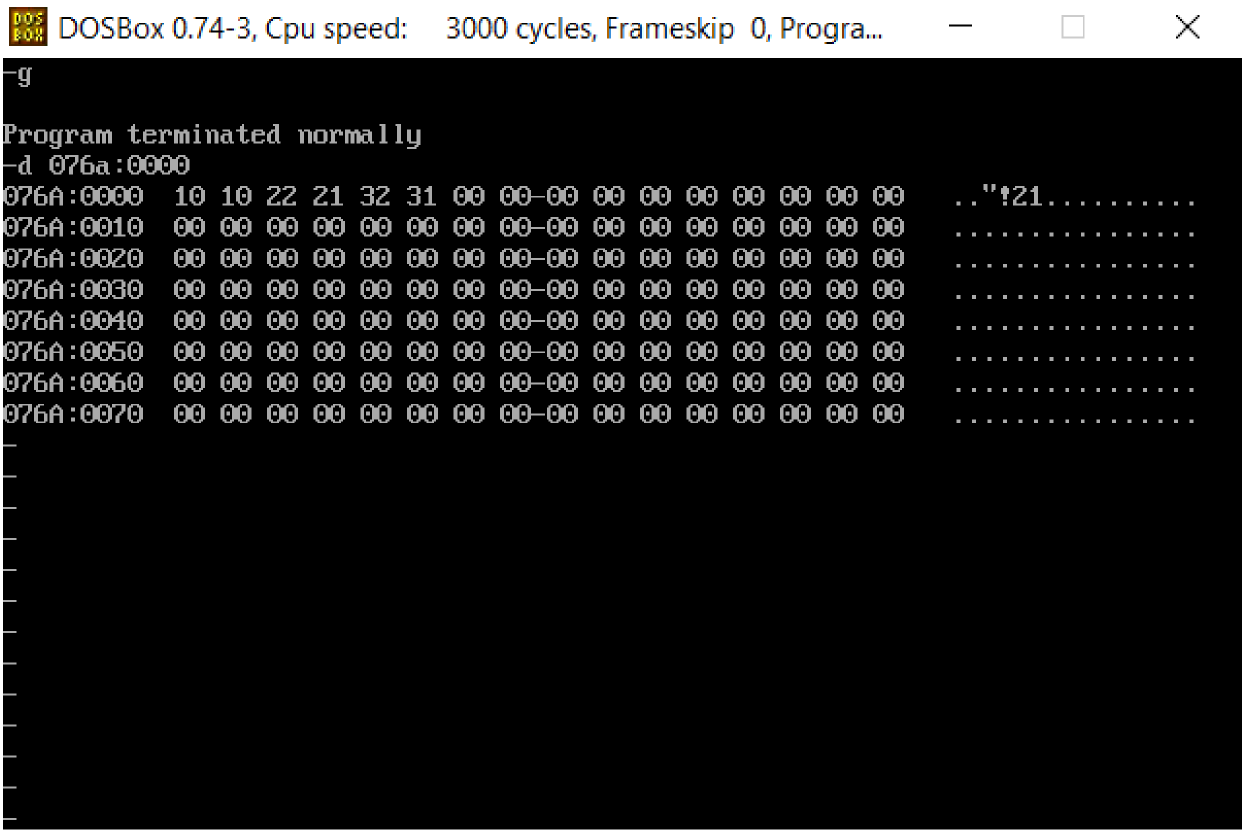

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
C:\>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 B90000        MOV     CX,0000
076B:010F 03C3          ADD     AX,BX
076B:0111 7301          JNB     0114
076B:0113 41            INC     CX
076B:0114 A30400        MOV     [0004],AX
076B:0117 890E0600      MOV     [0006],CX
076B:011B B44C          MOV     AH,4C
076B:011D CD21          INT     21
076B:011F 0000          ADD     [BX+SI],AL
-
-
-
-
-
-
-
-
-_
```

Sample input and output:


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

```
C:\>debug 16bitadd.exe
-e 076a:0000
076A:0000  11.10   11.10   99.22   99.21

-d 076a:0000
076A:0000  10 10 22 21 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   ................
-
-
-
-
-
-
-
-
-
-
-_
```


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

```
-g

Program terminated normally
-d 076a:0000
076A:0000  10 10 22 21 32 31 00 00-00 00 00 00 00 00 00 00   .."!21..........
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:

16 bit addition is executed and verified using an emulator.

16 - Bit Subtraction:

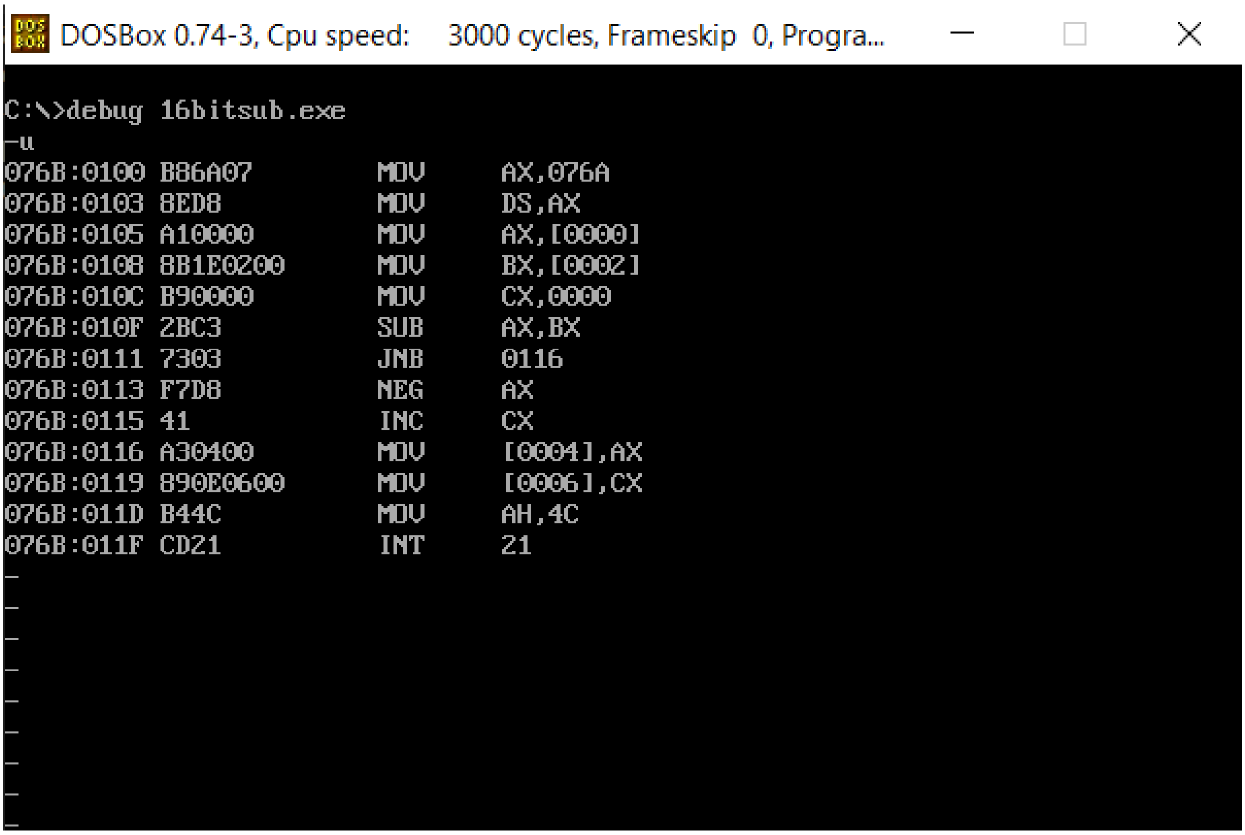
Algorithm:

- Program is set to run from any specified memory position.
- Load data from opr1 to register AX (first number)
- Load data from opr2 to register BX (second number)
- Subtract these two numbers (contents of register AX and register BX)
- Initialise carry to 0.
- Jump to final steps if there is no carry.
- Increment carry.
- And the result is negated.
- Store answer to result.
- Terminate the program.

Program:

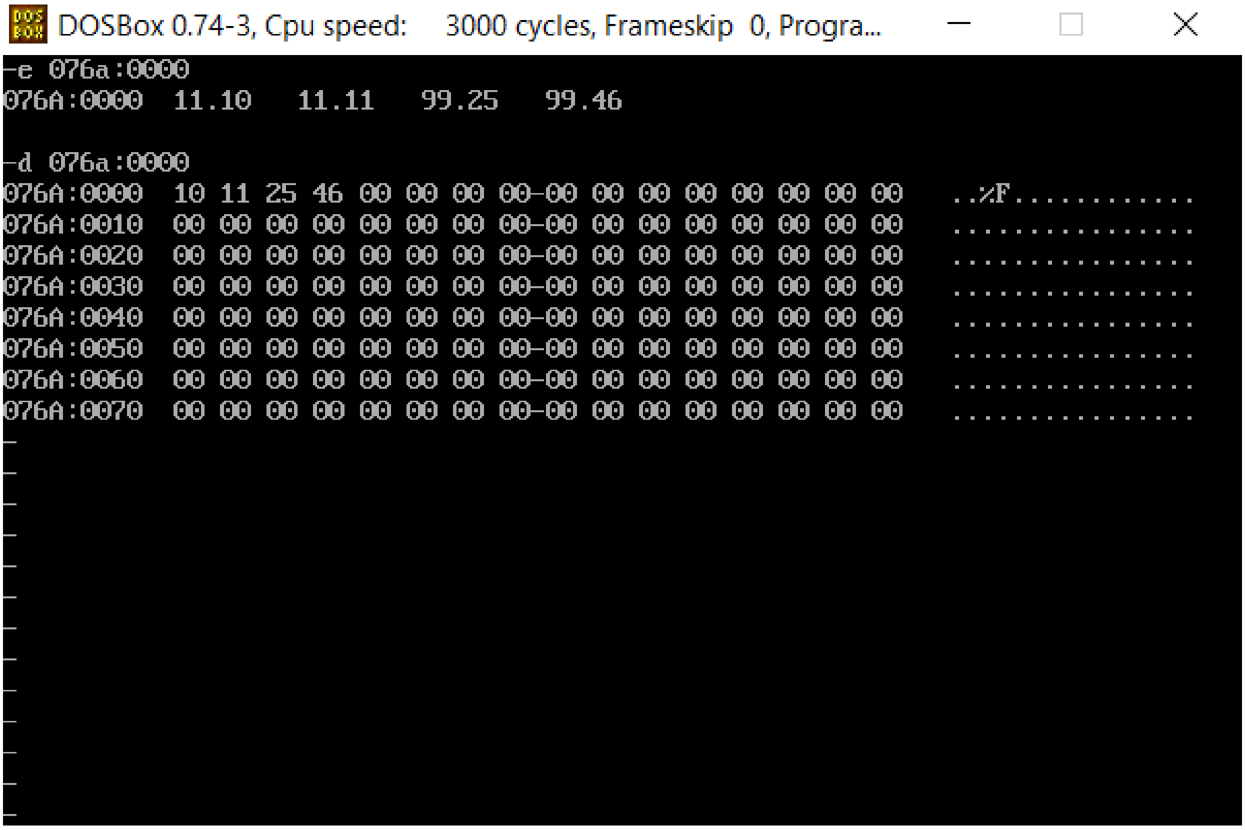
CODE	COMMENT
Program for Subtracting 2, 16-bit numbers	
assume cs:code,ds:data	
data segment	Data segment initialized
opr1 dw 1111h	opr1 initialised and set to 1111
opr2 dw 9999h	opr2 initialised and set to 9999
result dw 0000H	result initialised and set to 0000
carry dw 0000H	carry initialised and set to 0000
data ends	
code segment	Code segment begins
org 0100h	Originating address is set at 0100
start:	
mov ax,data	Address of data segment moved to ax
mov ds,ax	From ax, transferred to ds
mov ax,opr1	Value of opr1 transferred to ax
mov bx,opr2	Value of opr2 transferred to bx
mov cx,0000h	cx is initialised and set to 0
sub ax,bx	Substarction takes palce
jnc here	Junction created
neg ax	• Jump if no carry
inc cx	• Else: negate ah and increment cx
here:	
mov result,ax	data transferred from ax to result
mov carry,cx	data transferred from cx to carry
mov ah,4ch	
int 21h	Program terminates
code ends	
end start	

Unassembled code:

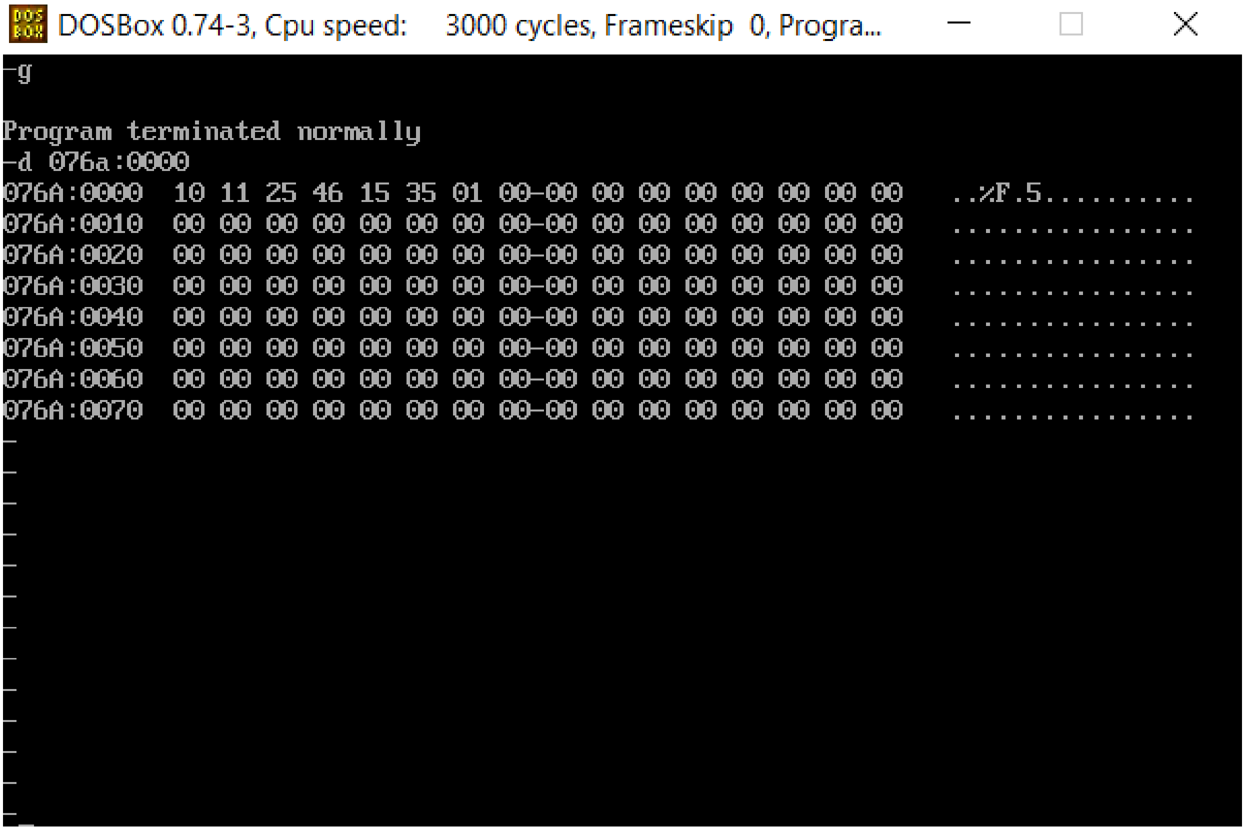

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
C:\>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 B90000        MOV     CX,0000
076B:010F 2BC3          SUB     AX,BX
076B:0111 7303          JNB     0116
076B:0113 F7D8          NEG     AX
076B:0115 41            INC     CX
076B:0116 A30400        MOV     [0004],AX
076B:0119 890E0600      MOV     [0006],CX
076B:011D B44C          MOV     AH,4C
076B:011F CD21          INT     21
-
-
-
-
-
-
-
-
-
```

Sample input and output:


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

```
-e 076a:0000
076A:0000  11.10   11.11   99.25   99.46

-d 076a:0000
076A:0000  10 11 25 46 00 00 00 00-00 00 00 00 00 00 00 00   ..%F............
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 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
-g

Program terminated normally
-d 076a:0000
076A:0000  10 11 25 46 15 35 01 00-00 00 00 00 00 00 00 00   ..%F.5..........
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:

16 bit subtraction is executed and verified using an emulator.

16 - Bit Multiplication:

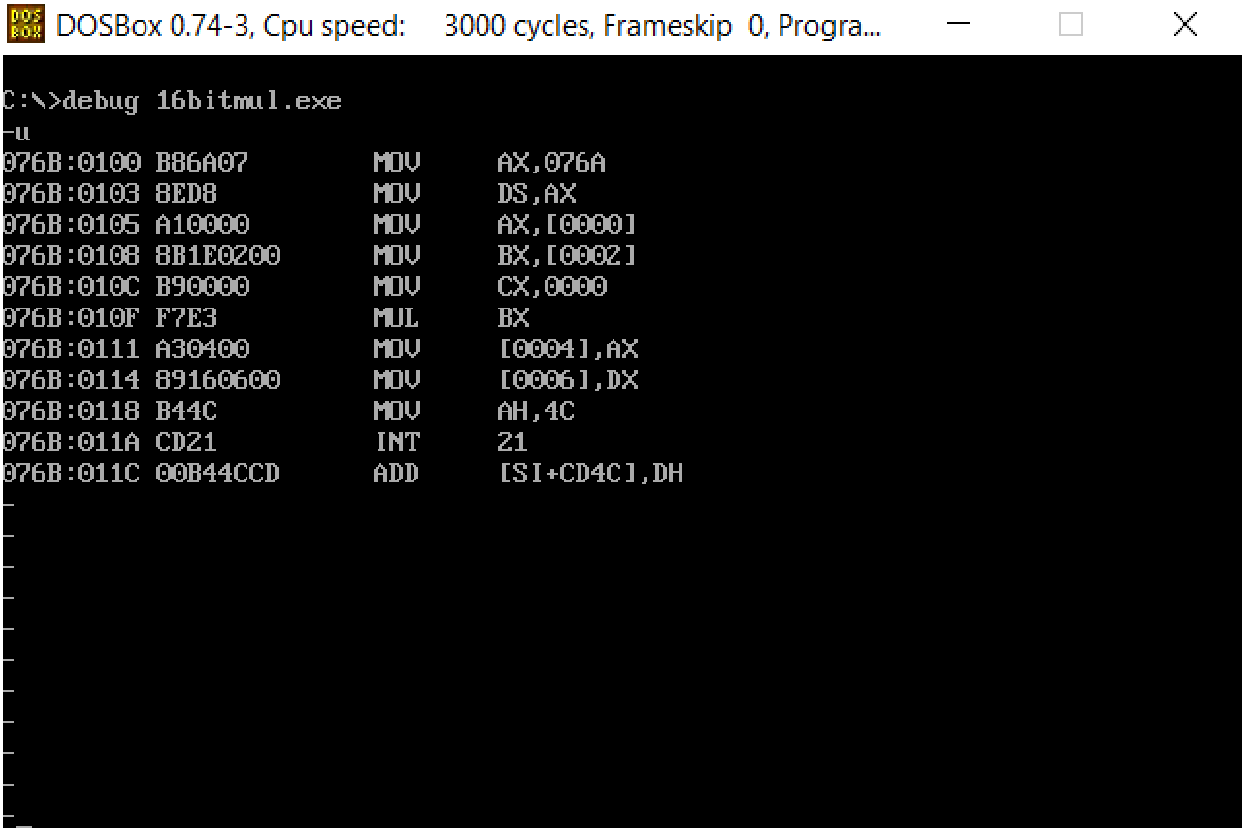
Algorithm:

- Program is set to run from any specified memory position.
- Load data from opr1 to register AX (first number)
- Load data from opr2 to register BX (second number)
- Multiply these two numbers (contents of register AX and register BX)
- Initialise carry to 0.
- Multiplied values is stored in ax and dx
- These two values are stored in different locations for better representation.
- Terminate the program.

Program:

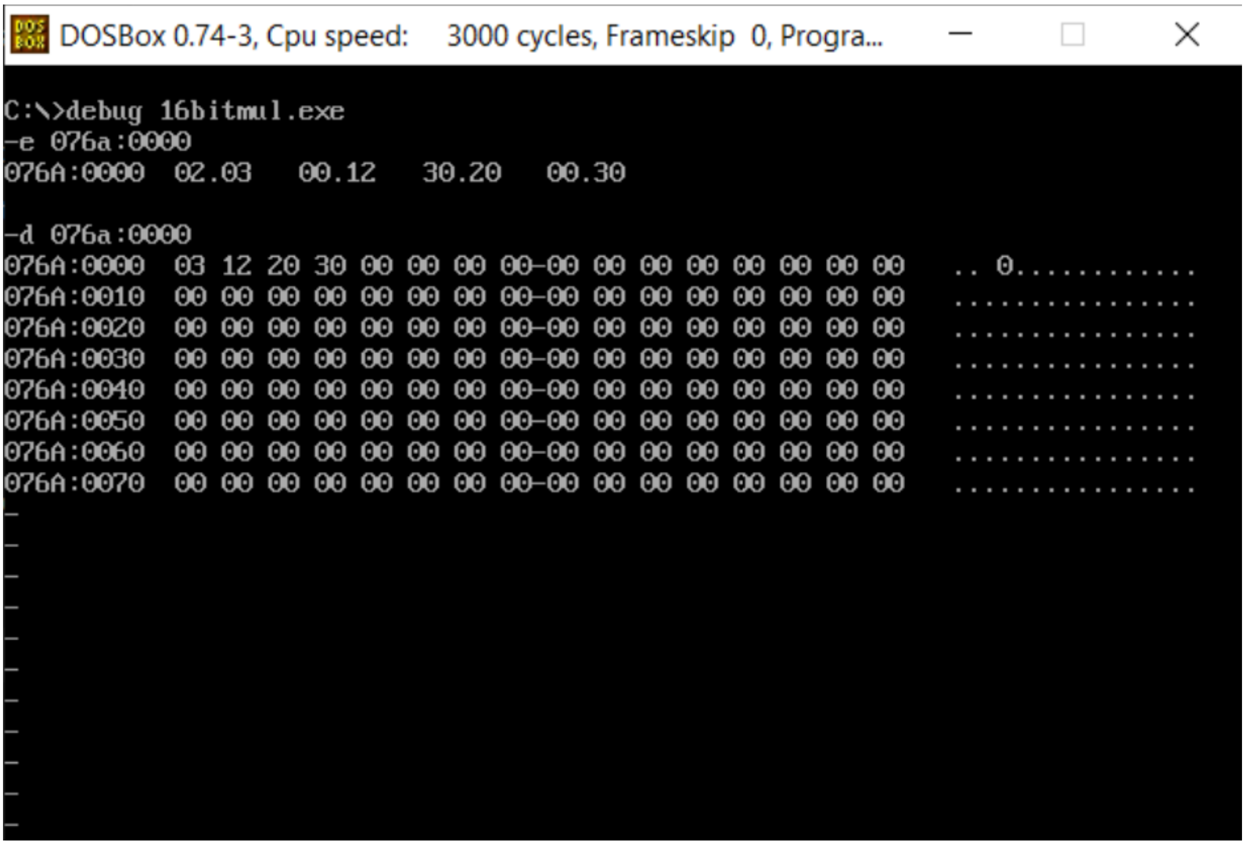
CODE	COMMENT
;Program for Multiplying 2, 16 bit numbers	
assume cs:code,ds:data	
data segment	Data segment initialized
opr1 dw 0002h	opr1 initialised and set to 0002
opr2 dw 0030h	opr2 initialised and set to 0030
result dw 0000H	result initialised and set to 0000
res dw 0000H	res initialised and set to 0000
data ends	
code segment	Code segment begins
org 0100h	Originating address is set at 0100
start:	
mov ax,data	Address of data segment moved to ax
mov ds,ax	From ax, transferred to ds
mov ax,opr1	Value of opr1 is transfers to ax
mov bx,opr2	Value of opr2 is transferred to bx
mul bx	Multiply ax and bx
mov result,ax	data transferred from ax to result
mov res,dx	data transferred from dx to res
mov ah,4ch	
int 21h	Program terminates
code ends	
end start	

Unassembled code:

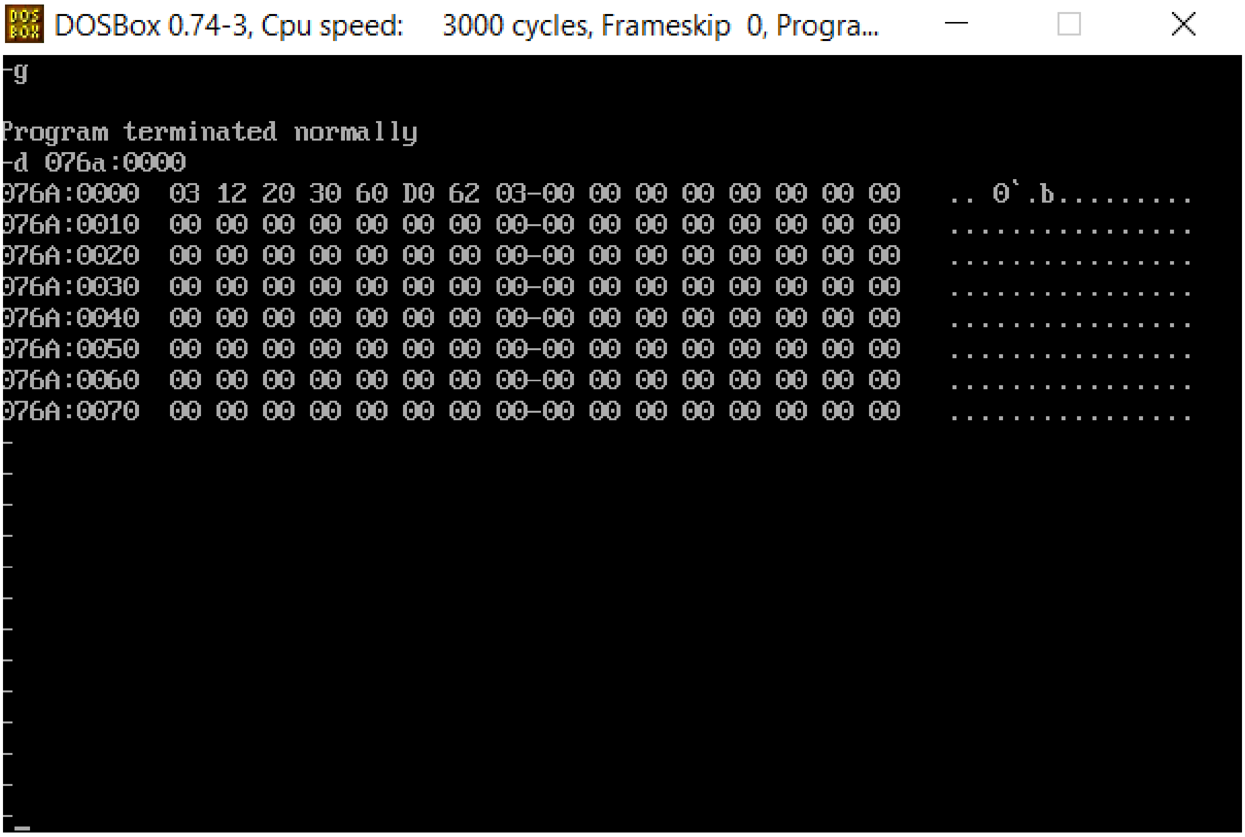

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
C:\>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 B90000        MOV     CX,0000
076B:010F F7E3          MUL     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 00B44CCD      ADD     [SI+CD4C],DH
-
-
-
-
-
-
-
-
-
-
-
```

Sample input and output:


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

```
C:\>debug 16bitmul.exe
-e 076a:0000
076A:0000  02.03   00.12   30.20   00.30

-d 076a:0000
076A:0000  03 12 20 30 00 00 00 00-00 00 00 00 00 00 00 00   .. 0............
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 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
-g

Program terminated normally
-d 076a:0000
076A:0000  03 12 20 30 60 D0 62 03-00 00 00 00 00 00 00 00   .. 0`.b.........
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:

16 bit multiplication is executed and verified using an emulator.

16 -Bit Division:

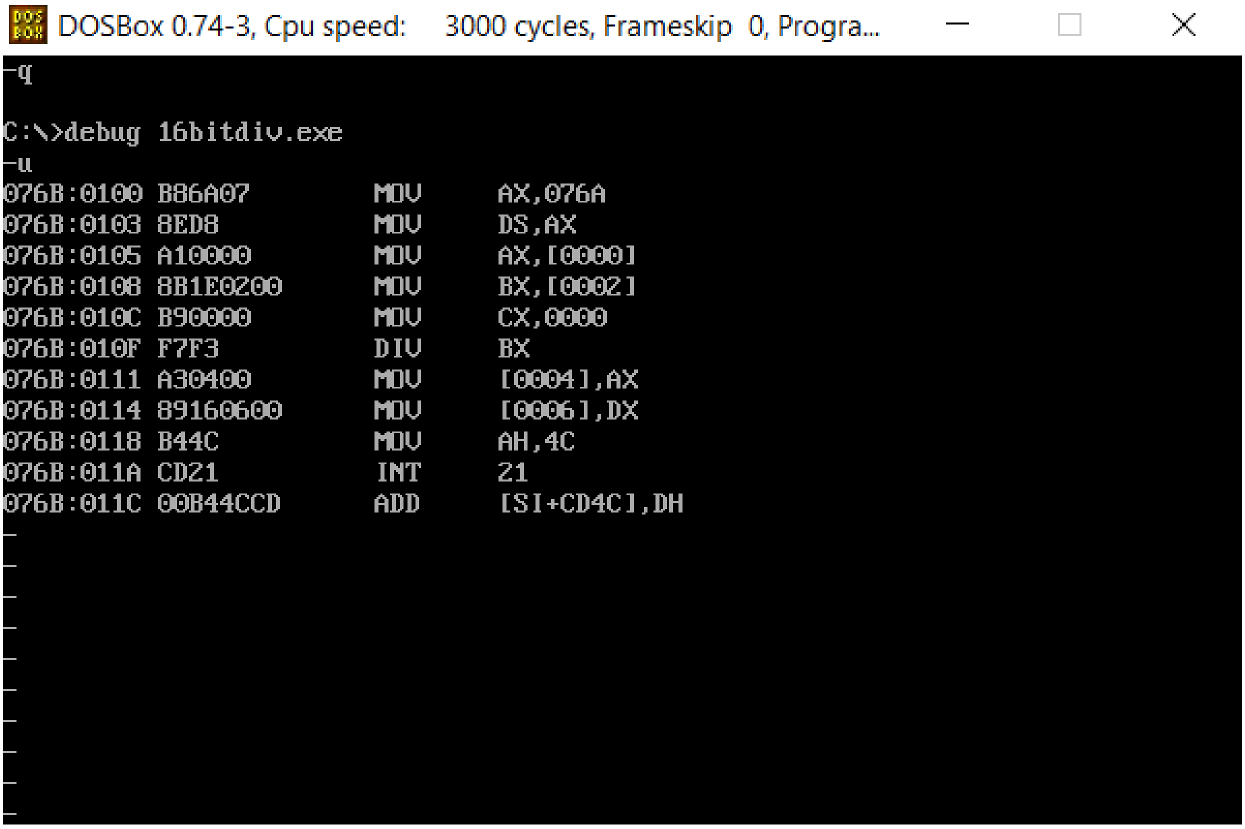
Algorithm:

- Load data from opr1 to register AX (first number)
- Load data from opr2 to register BX (second number)
- Initialise a variable for remainder to 0.
- Divide these two numbers (contents of register AX and register BX)
- Move ax value to quotient variable.
- Move dx value to remainder variable.
- Terminate the program.

Program:

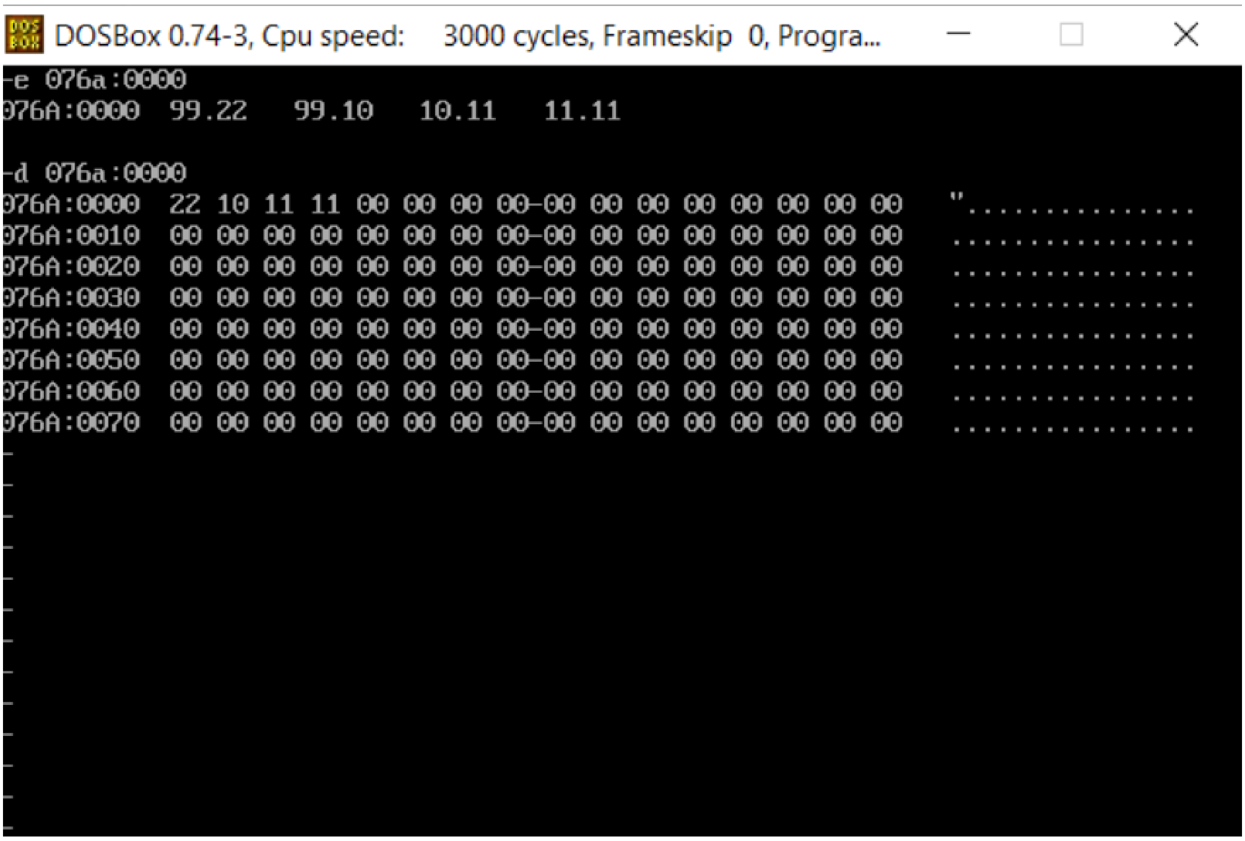
CODE	COMMENT
;Program for Dividing 2, 16 bit numbers	
assume cs:code,ds:data data	
segment	Data segment initialized
opr1 dw 9999h	opr1 initialised and set to 9999
opr2 dw 1111h	opr2 initialised and set to 1111
quotient dw 0000H	quotient initialised and set to 00
rem dw 0000H	rem initialised and set to 0000
data ends	
code segment	Code segment begins
org 0100h	Originating address is set at 0100
start:	
mov ax,data	Address of data segment moved to ax
mov ds,ax	From ax, transferred to ds
mov ah,0000h	ax is initialised and set to 0000
mov ax,opr1	Value of opr1 transferred to ax
mov bx,opr2	Value of opr2 transferred to bx
div bx	Division takes place
mov quotient,ax	data transferred from ax to quotient
mov rem,dx	data transferred from dx to rem
mov ah,4ch	
int 21h	Program terminates
code ends	
end start	

Unassembled code:

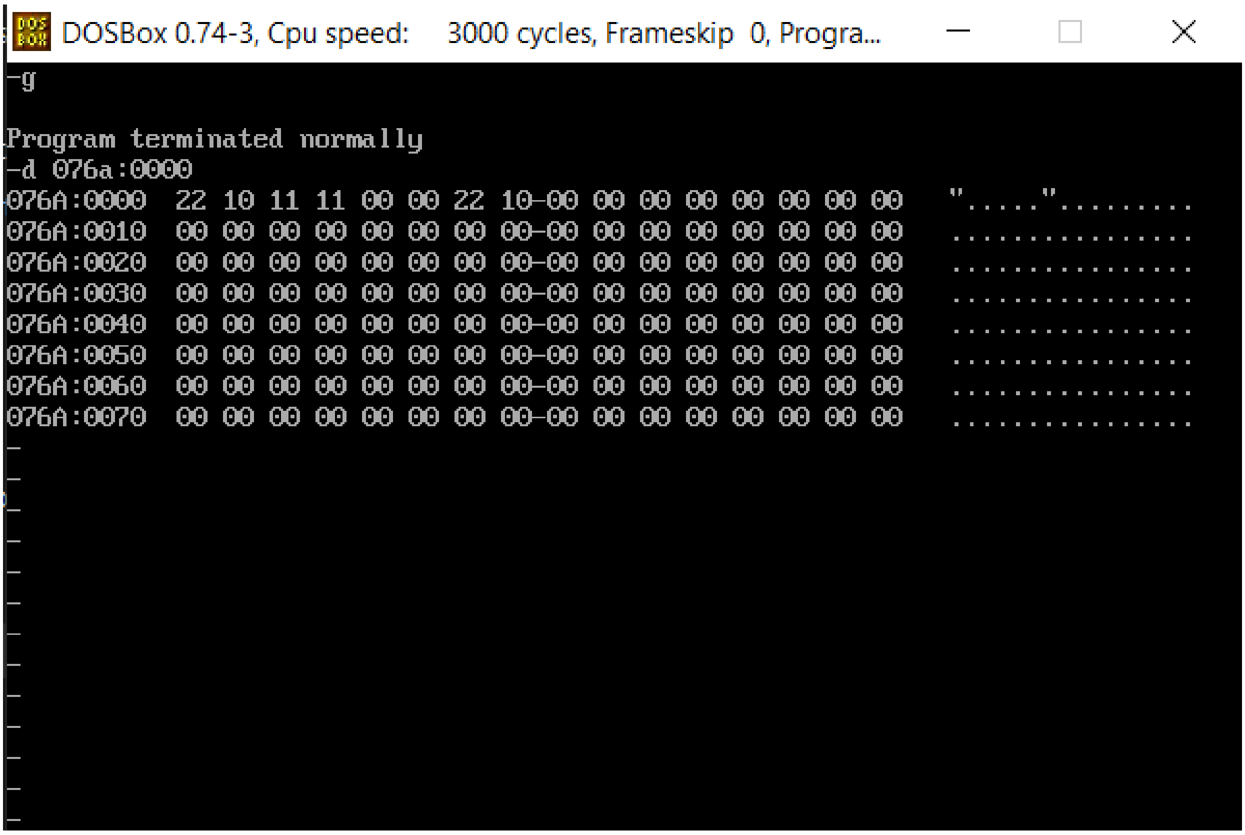

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
-q

C:\>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 B90000        MOV     CX,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 00B44CCD      ADD     [SI+CD4C],DH
-
-
-
-
-
-
-
-
-
-
```

Sample input and output:


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

```
-e 076a:0000
076A:0000  99.22   99.10   10.11   11.11

-d 076a:0000
076A:0000  22 10 11 11 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   ................
-
-
-
-
-
-
-
-
-
-
-
-
-
```


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

```
-g

Program terminated normally
-d 076a:0000
076A:0000  22 10 11 11 00 00 22 10-00 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:

16 bit division is executed and verified using an emulator.