8 – BIT ARITHMETIC OPERATIONS

Exp No.: 1	Name: S Vishakan
Date: 19-08-2020	Reg. No: 18 5001 196

AIM: To write assembly language programs to perform 8-bit arithmetic operations and execute them.

PROCEDURE FOR EXECUTING MASM:

Assemble the file using	:	MASM <FILENAME>.ASM
Link the file using	:	LINK <FILENAME>.OBJ
Debug the file using	:	DEBUG <FILENAME>.EXE
Options in Debugging	:	

-u	: Un-assemble the instructions
-d mem_start:mem_end	: Memory dump
-e mem_start:mem_end	: To change the values stored in memory
-g	: Execute the instructions
-q	: Quit the debugging

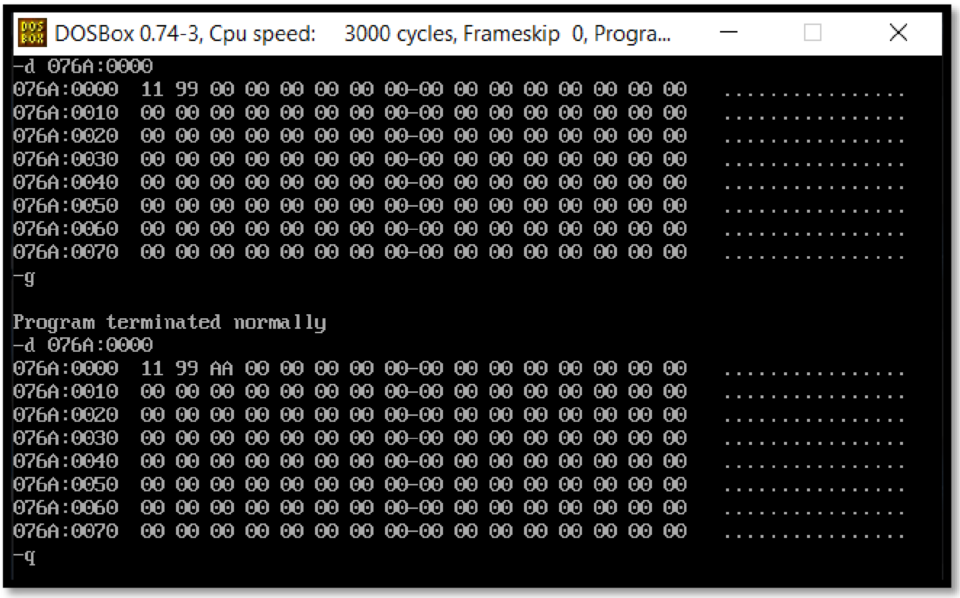
ALGORITHM:

- 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 compliment)
- Introduce an interrupt for safe exit (int 21h)
- Close the code Segment
- End

PROGRAM – 1: 8 – BIT ADDITION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 db 11h	Stores operand 1.
opr2 db 99h	Stores operand 2.
result db 00h	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 ah, opr1	Transfer value of opr1 to AH.
mov bh, opr2	Transfer value of opr2 to BH.
mov ch, 00h	CH = 0.
add ah, bh	AH = AH + BH.
jnc here	Jump if no carry to “here”. Else, continue.
inc ch	CH = CH + 1
here: mov result, ah	Transfer value of AH 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	

SAMPLE I/O SNAPSHOT:

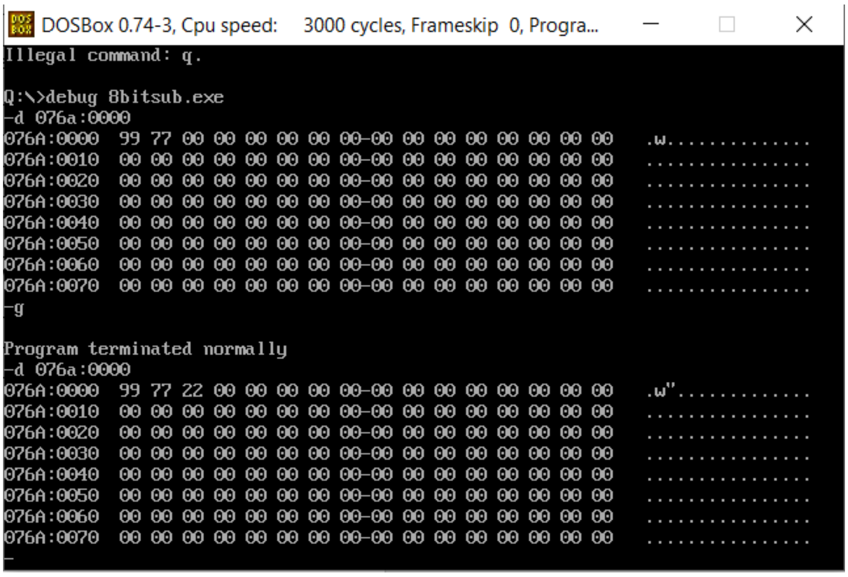

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
-d 076A:0000
076A:0000  11 99 00 00 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  11 99 AA 00 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   ................
-q
```

PROGRAM – 2: 8 – BIT SUBTRACTION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 db 11h	Stores operand 1.
opr2 db 99h	Stores operand 2.
diff db 00h	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 ah, opr1	Transfer value of opr1 to AH.
mov bh, opr2	Transfer value of opr2 to BH.
mov ch, 00h	CH = 0.
sub ah, bh	AH = AH – BH.
jnc here	Jump if no sign change to “here”. Else, continue.
neg ah	Take 2’s Complement if negative value.
inc ch	CH = CH + 1
here: mov diff, ah	Transfer value of AH 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	

SAMPLE I/O SNAPSHOT:

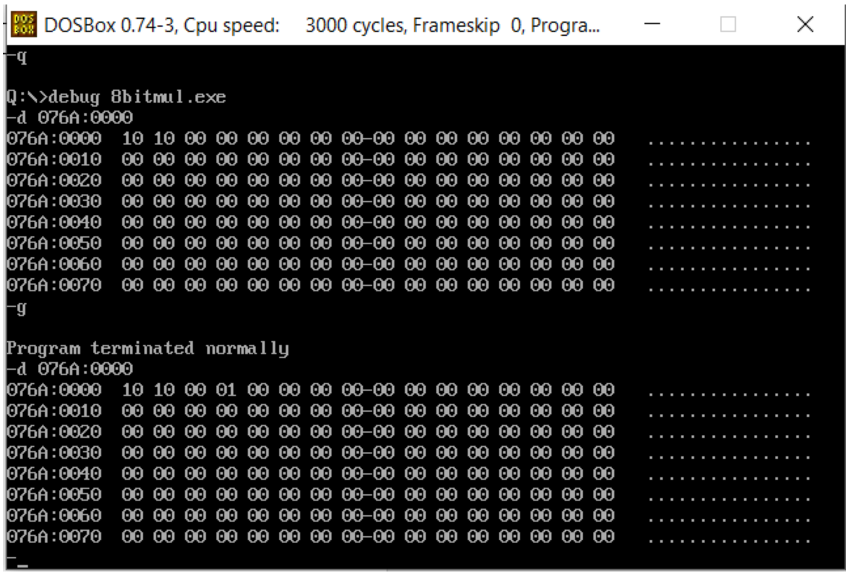

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
Illegal command: q.

Q:\>debug 8bitsub.exe
-d 076a:0000
076A:0000  99 77 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .w..............
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 77 22 00 00 00 00 00-00 00 00 00 00 00 00 00   .w".............
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: 8 – BIT MULTIPLICATION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 db 10h	Stores operand 1.
opr2 db 10h	Stores operand 2.
product dw 0000H	Stores the result 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 al, opr1	Transfer value of opr1 to AL.
mov bl, opr2	Transfer value of opr2 to BL.
mul bl	AX= AL * BL.
mov product, ax	Transfer value of AX to product.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

SAMPLE I/O SNAPSHOT:

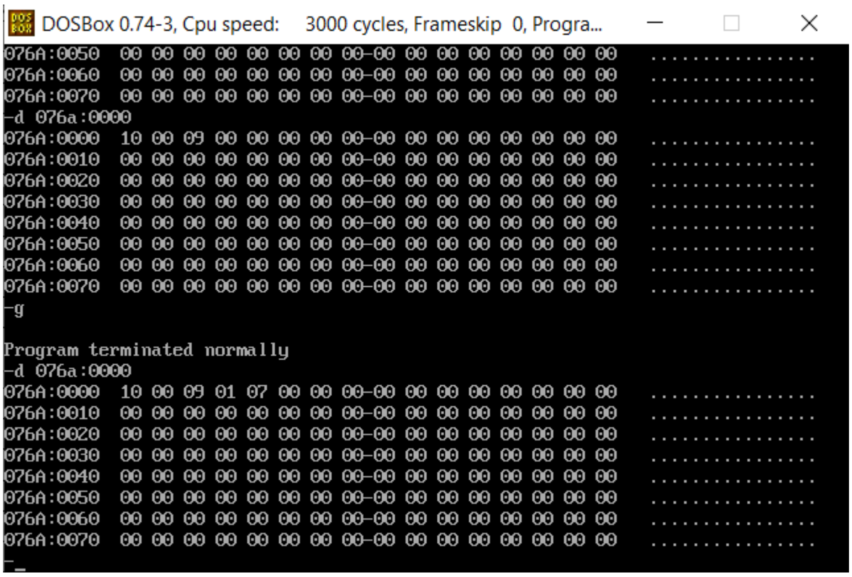

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
-q

Q:\>debug 8bitmul.exe
-d 076A:0000
076A:0000  10 10 00 00 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  10 10 00 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   ................
-_
```

PROGRAM – 4: 8 – BIT DIVISION:

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
opr1 dw 0010h	Stores operand 1.
opr2 db 09h	Stores operand 2.
quot db 00h	Stores the quotient of the division.
rem db 00h	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 opr1 to AX.
mov bl, opr2	Transfer value of opr2 to BL.
div bl	AX = AX / BL. (AL has quotient, AH has remainder)
mov quot, al	Transfer value of AL to quot.
mov rem, ah	Transfer value of AH to rem.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

SAMPLE I/O SNAPSHOT:



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

```
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   ................
-d 076a:0000
076A:0000  10 00 09 00 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  10 00 09 01 07 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 8 – bit arithmetic operations and compiled. The results were observed and noted down.