Exp. 1

Date: 19/08/2020

**8 BIT ARITHMETIC OPERATIONS**

Name: N Veeraraghavan

Reg. No: 185001191

**Aim:** To perform 8-bit addition, subtraction, multiplication and division using the MASM assembler.

**Procedure for executing MASM:**

1. The required 8086 program was created and saved with extension .asm in the mounted MASM folder.
2. After the file was created, it was assembled using the command “masm filename.asm”.
3. The file was linked using the command “link filename.obj”.
4. Next, the file was executed and memory contents were analysed using the command “debug filename.exe”.
   1. “u” : displayed unassembled code
   2. “d segment:offset” : used to view content of memory locations starting from segment:offset address
   3. “e segment:offset” : changed the value in that memory location
   4. “g” : executed the program
   5. “q” : exit from debug screen
5. After executing the program with “g” in debug, the result was checked using “d segment: offset”.

**Algorithm:**

1. Start
2. Declare names for code segment and data segments
3. In the data segment, declare variable names to identify memory locations
4. Start code segment:
   1. Move address value of data segment into AX and then to DS.
   2. Move operands into registers required to perform the operations.
   3. Perform required arithmetic instructions.
   4. Move the result to desired memory location.

3.5 Perform the interrupt routine.

End code segment

1. End

**Programs:**

1. 8 bit addition:

|  |  |
| --- | --- |
| **Program** | **Comments** |
| mov ah, opr1  mov bh, opr2 | Transferring opr1 [0000] and opr2 [0001] to ah and bh respectively |
| mov ch, 00h | Setting value of ch as 00h (zero) |
| add ah, bh | AH = AH + BH |
| jnc here | Jump if no carry to “here” |
| inc ch | If carry is present, ch is incremented by 1 |
| here: mov result, ah | Result of addition (sum) moved to memory location “result” [0002] |
| mov carry, ch | Carry obtained by addition moved to location “carry” [0003] |

1. 8 bit subtraction:

|  |  |
| --- | --- |
| **Program** | **Comments** |
| mov ah, opr1  mov bh, opr2 | Transferring opr1 [0000] and opr2 [0001] to ah and bh respectively |
| mov ch, 00h | Setting value of ch as 00h (zero) |
| sub ah, bh | AH = AH - BH |
| jnc here | Jump if no carry to “here” |
| inc ch | If carry is present, ch is incremented by 1 |
| neg ah | If carry is present, ah < bh which means answer is negative, so 2s complement is found |
| here: mov result, ah | Difference obtained is moved to memory location “result” [0002] |
| mov carry, ch | Carry obtained by addition moved to location “carry” [0003] - indicates whether number is negative or not |

1. 8 bit multiplication:

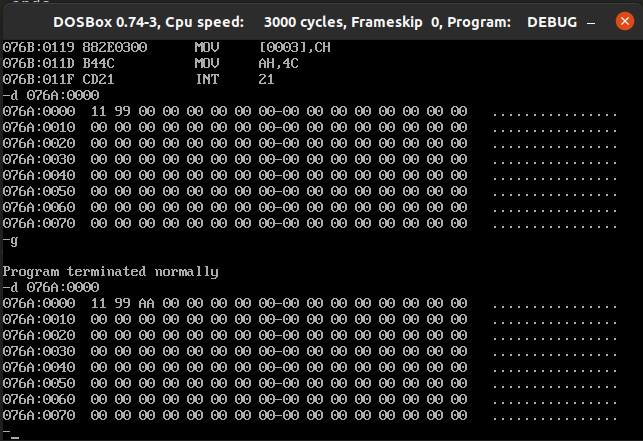
|  |  |
| --- | --- |
| **Program** | **Comments** |
| mov al, opr1  mov bl, opr2 | Transferring op1 [0000] and opr2 [0001] to al and bl respectively |
| mov ah, 00h | Setting value of ah as 00h (zero) |
| mul bl | AX = AL X BL |
| mov result1, ah  Mov result2, al | Product is moved to result1 [0002] (higher order byte) and to result2 [0003] |

1. 8 bit division:

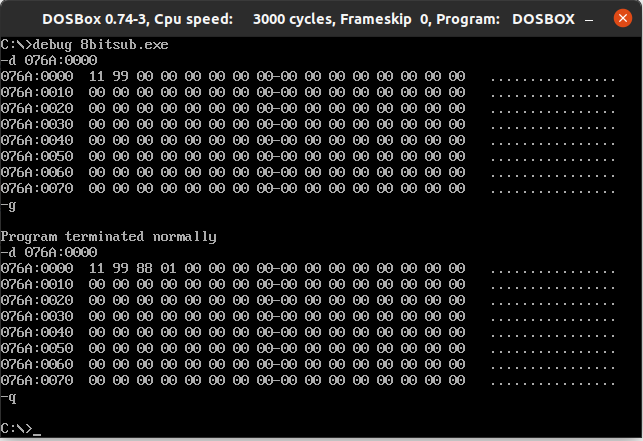
|  |  |
| --- | --- |
| **Program** | **Comments** |
| mov al, opr1  mov bl, opr2 | Transferring op1 [0000] and opr2 [0001] to al and bl respectively |
| mov ah, 00h | Setting value of ah as 00h (zero) |
| div bl | AL = AX / BL (Quotient)  AH = AX MOD BL (Remainder) |
| mov result1, al  mov result2, ah | Quotient is moved to result1 [0002] and remainder is moved to result2 [0003] |

**Sample Input and Output:**

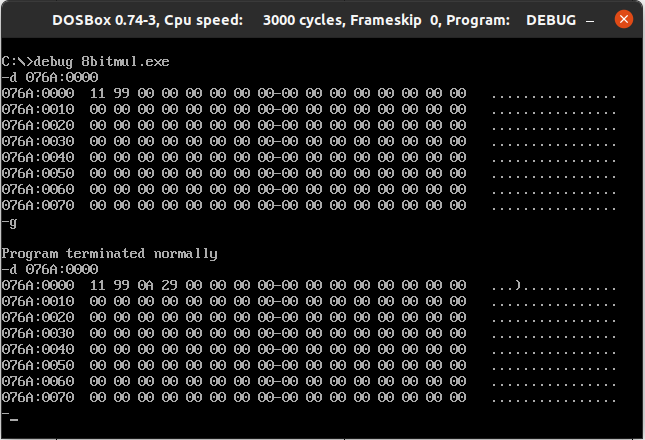
1. 8 bit addition -



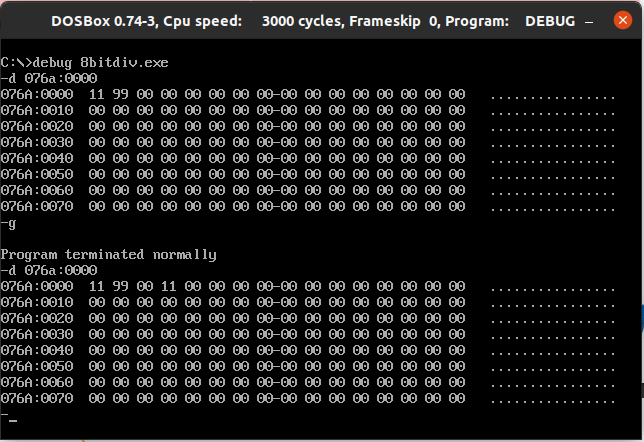
1. 8 bit subtraction -



1. 8 bit multiplication -



1. 8 bit division -



**Result:** 8 bit arithmetic operations such as addition, subtraction, multiplication and division were performed successfully.