**16bit arithmetic operations**

**Exp No:** 2

**Name:** Rahul Ram M

**Register Number:** 185001121

**Date:** 30/08/2020

**a) 16-bit addition:**

**Aim:**

Design 8086 program for 16-bit arithmetic addition.

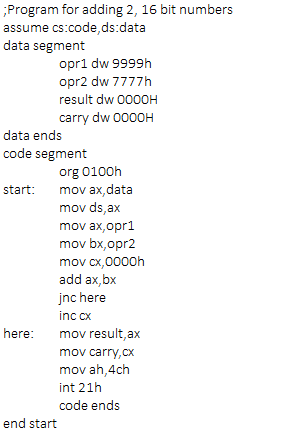
**Procedure for executing MASM:**

1. Run Dosbox and mount your masm folder to a drive in dosbox.
2. Goto the mounted drive.
3. Save the 8086 program with extension .asm in the same folder using command “edit”
4. After creating the file, assemble it using the command “masm filename.asm”
5. Link the file using the command “link filename.obj;”
6. Use debug command with filename.exe to execute and analyse the memory contents, “debug filename.exe”.
7. In debug, command “u” will display the unassembled code.
8. Use command “d segment:offset” to see the content of memory locations starting from segment:offset address.
9. To change the value in memory, use the command “e segment:offset”
10. Verify the memory contents to ensure the updates (using command “d”).
11. . Execute using the command “g” and check the outputs.
12. “q” to exit from debug and “exit” to exit from command prompt and to close the Dosbox.

**Algorithm:**

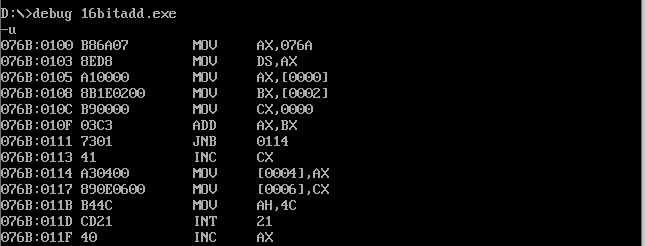
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AX register.
3. Move the data of second operator to BX register.
4. Load CX register with 0000H.
5. Then add AX and BX using ADD AX, BX and the value will be stored in AX register.
6. If the carry flag is not changed increment CH using “INC CX”.
7. If the carry flag is reset, jump to “here”.
8. Now store the data of AX register in result and CX register in result.
9. Move the hexadecimal value 4C into AH register.
10. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

**Program:**

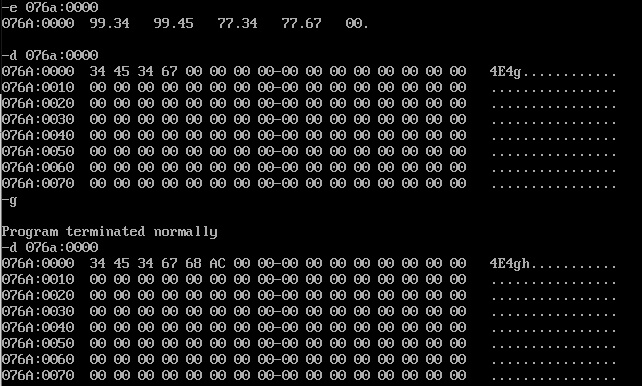


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0010H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, OPR1 | Transfer the data from opr1 to AX register. |
| MOV BX, OPR2 | Transfer the data from opr2 to BX register. |
| MOV CX, 0000H | Assign 0000h to CX register. |
| ADD AX, BX | AX = AX + BX |
| JNC HERE | Jump if no carry to here |
| INC CH | Increment the value in the CX register. |
| HERE: | MOV RESULT, AX | Move the data from AX register to result. |
| MOV CARRY, CH | Move the data from CX register to carry. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

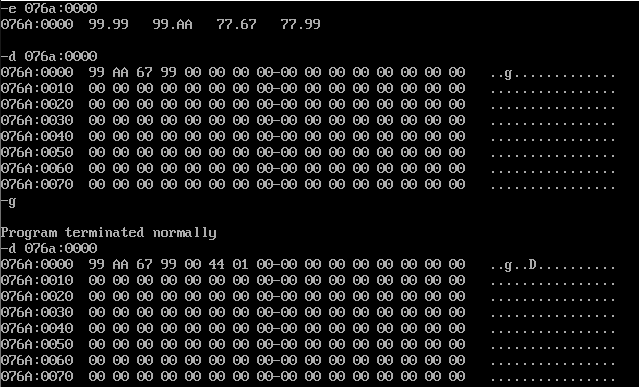
**Snapshot of sample input and output:**



**Without carry (op1 – 45 34, op2 – 67 34):**



**With carry (op1 – AA 99, op2 – 99 67):**



**Result:**

Thus the 8086 program for adding 2, 16-bit numbers is executed successfully in DOS-BOX.

**b) 16-bit subtraction:**

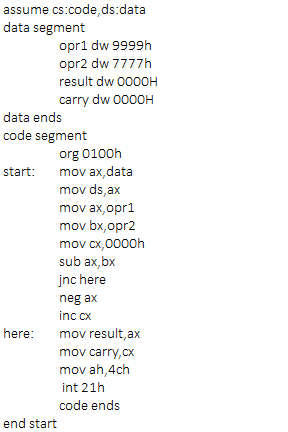
**Aim:**

Design 8086 program for 16-bit arithmetic subtraction.

**Algorithm:**

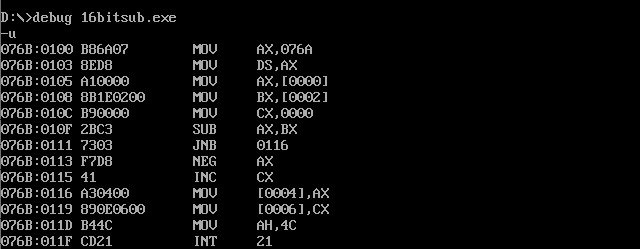
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AX register.
3. Move the data of second operator to BX register.
4. Load CX register with 0000H.
5. Then sub AH and BH using SUB AX, BH and the value will be stored in AX register.
6. If the carry flag is not changed increment CX using “INC CX” and find the 2’s compliment of the result in AH using “NEG AX”.
7. If the carry flag is reset, jump to “here”.
8. Now store the data of AX register in result and CX register in result.
9. Move the hexadecimal value 4C into AH register.
10. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

**Program:**

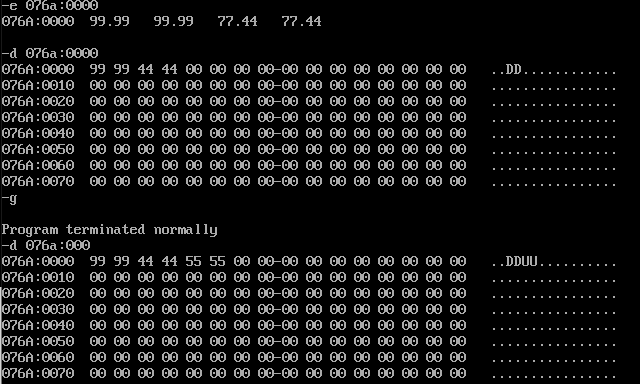


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, OPR1 | Transfer the data from opr1 to AX register. |
| MOV BX, OPR2 | Transfer the data from opr2 to BX register. |
| MOV CX, OOH | Assign 00h to CX register. |
| SUB AX, BX | AX = AX - BX |
| JNC HERE | Jump if no carry to here |
| INC CX | Increment the value in the CX register. |
| NEG AX | Takes 2’s compliment for data in AX register. |
| HERE: | MOV RESULT, AX | Move the data from AX register to result. |
| MOV CARRY, CX | Move the data from CX register to carry. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

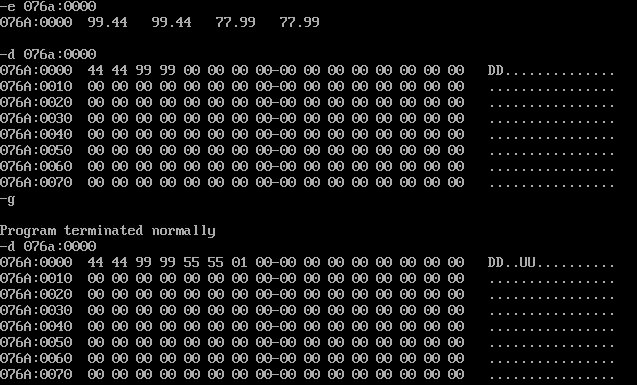
**Snapshot of sample input and output:**



**Without carry (op1 – 99 99, op2 – 44 44):**



**With carry (op1 – 44 44, op2 – 99 99):**



**Result:**

Thus the 8086 program for subtracting 2, 16-bit numbers is executed successfully in

DOS-BOX.

**c) 16-bit multiplication:**

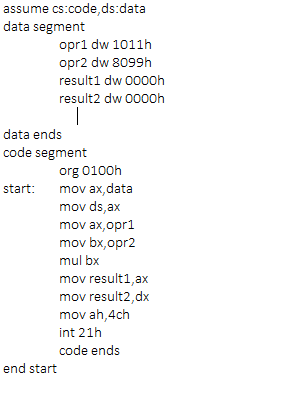
**Aim:**

Design 8086 program for 16-bit arithmetic multiplication.

**Algorithm:**

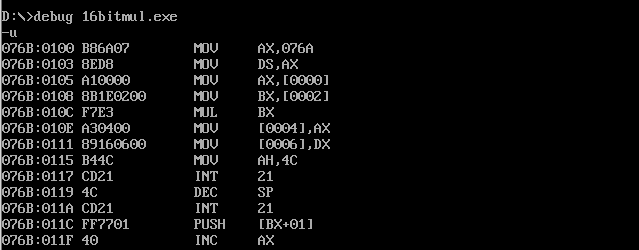
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AX register.
3. Move the data of second operator to BX register.
4. Then multiply Al and BL using “MUL BX”.
5. DX will store the higher order bits and dx will store the lower order bits.
6. Now store the data of AX in result1 and DX in result2.
7. Move the hexadecimal value 4C into AH register.
8. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

**Program:**

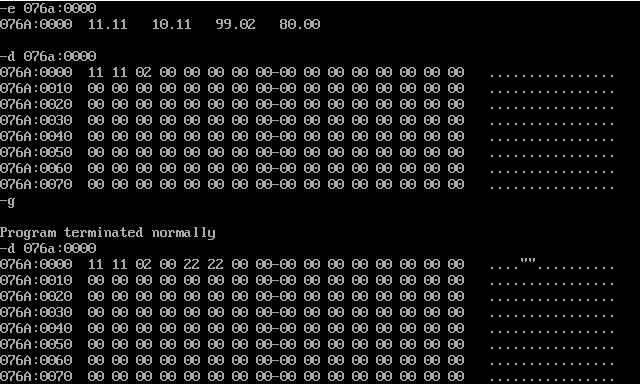


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, OPR1 | Transfer the data from opr1 to AX register. |
| MOV BX, OPR2 | Transfer the data from opr2 to BX register. |
| MUL BX | DXAX = AX x BX |
| MOV RESULT1, AX | Move the data from AX register to result1. |
| MOV RESULT2, DX | Move the data from DX register to result2. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

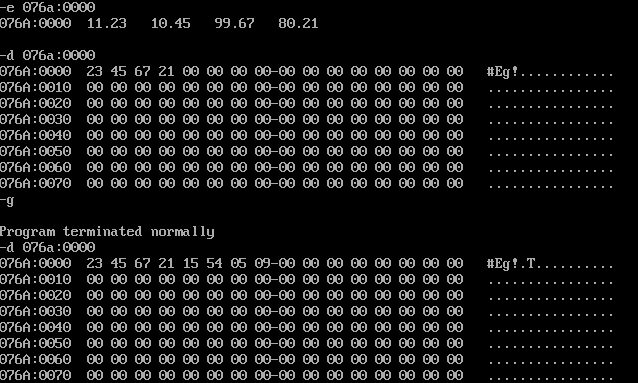
**Snapshot of sample input and output:**



**16-bit result (op1 – 11 11, op2 – 00 02):**



**32-bit result (op1 – 45 23, op2 – 21 67):**



**Result:**

Thus the 8086 program for multiplying 2, 16-bit numbers is executed successfully in DOS-BOX.

**d) 16-bit division:**

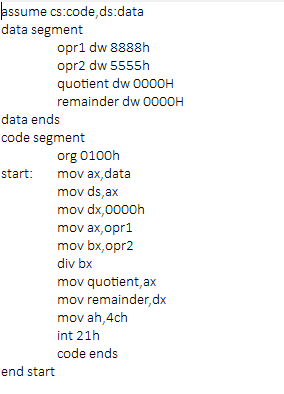
**Aim:**

Design 8086 program for 16-bit arithmetic division.

**Algorithm:**

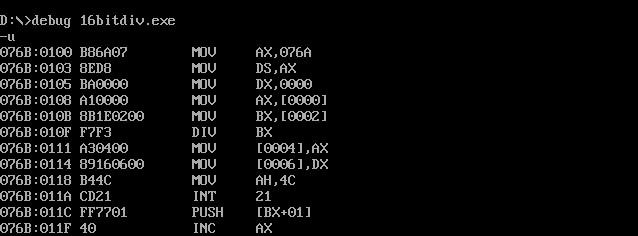
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Load DX register with 0000H.
3. Move the data of first operator to AX register.
4. Move the data of second operator to BX register.
5. Then divide DXAX by BX using “DIV BX”.
6. After execution, quotient will be stored in AX and remainder is stored in DX register.
7. Now move the data from AX register to quotient and DX register to remainder.
8. Move the hexadecimal value 4C into AH register.
9. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

**Program:**

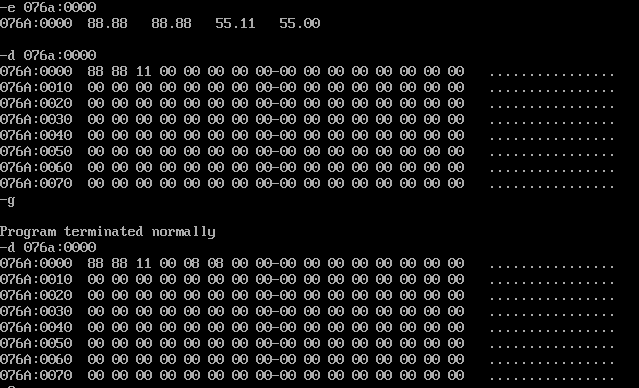


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV DX, 0000H | Assign 0000h to DX register. |
| MOV AX, OPR1 | Transfer the data from opr1 to AX register. |
| MOV BX, OPR2 | Transfer the data from opr2 to BX register. |
| DIV BX | Performs DXAX / BX. AX <- quotient. DX <- remainder. |
| MOV QUOTIENT, AX | Move the data from AX register to quotient. |
| MOV REMAINDER, DX | Move the data from DX register to remainder. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

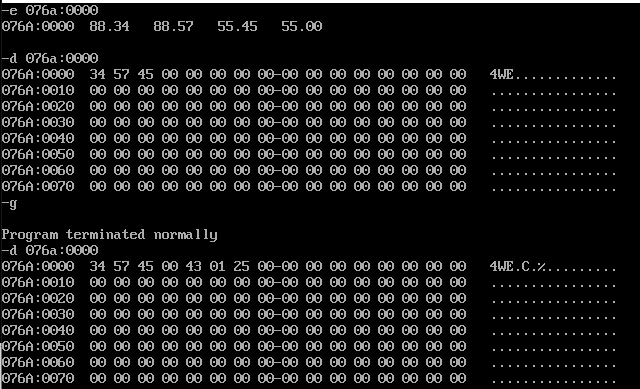
**Snapshot of sample input and output:**



**Without remainder (op1 – 88 88, op2 – 00 11):**



**With remainder (op1 – 57 34, op2 – 00 45):**



**Result:**

Thus the 8086 program for performing division with 2, 16-bit numbers is executed successfully in DOS-BOX.