**8 bit arithmetic operations**

**Exp No :** 1 **Name :** Rahul Ram M

**Date :** 23/08/2020  **Register Number :** 185001121

**a) 8 bit addition:**

**Aim:**

Design 8086 program for 8-bit arithmetic addition.

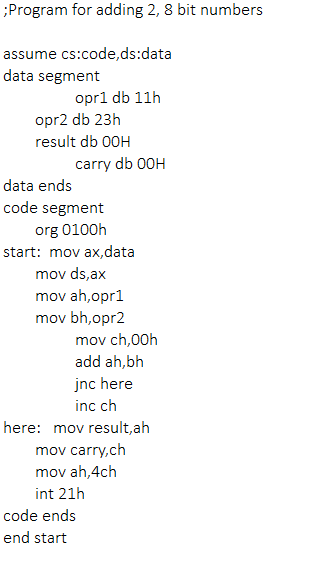
**Procedure for executing MASM:**

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:**

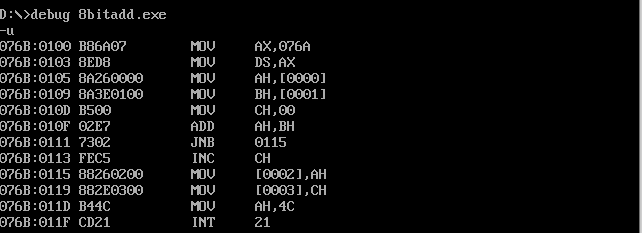
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AH register.
3. Move the data of second operator to BH register.
4. Load CH register with 00H.
5. Then add AH and BH using ADD AH, BH and the value will be stored in AH register.
6. If the carry flag is not changed increment CH using “INC CH”.
7. If the carry flag is reset, jump to “here”.
8. Now store the data of AH register in result and CH 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 AH, OPR1 | Transfer the data from opr1 to AH register. |
| MOV BH, OPR2 | Transfer the data from opr2 to BH register. |
| MOV CH, OOH | Assign 00h to CH register. |
| ADD AH, BH | AH = AH + BH |
| JNC HERE | Jump if no carry to here |
| INC CH | Increment the value in the CH register. |
| HERE: | MOV RESULT, AH | Move the data from AH register to result. |
| MOV CARRY, CH | Move the data from CH register to carry. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

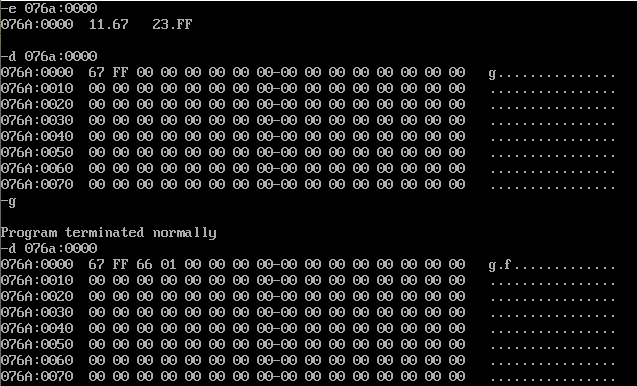
**Snapshot of sample input and output:**

****

**Without carry(op1 – 11, op2 – 23):**

****

**With carry(op1 – 67, op2 – FF):**

****

**Result:**

Thus the 8086 program for adding 2, 8bit numbers is executed successfully in DOS-BOX.

**b) 8 bit subtraction:**

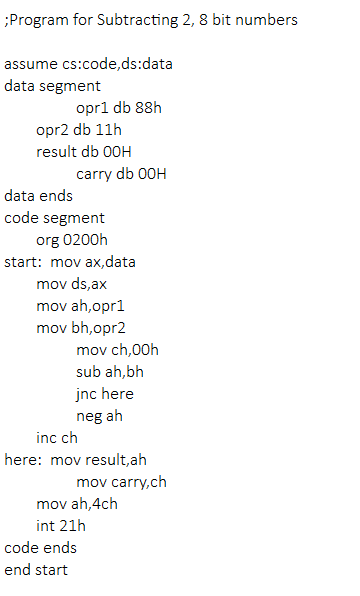
**Aim:**

Design 8086 program for 8-bit arithmetic subtraction.

**Algorithm:**

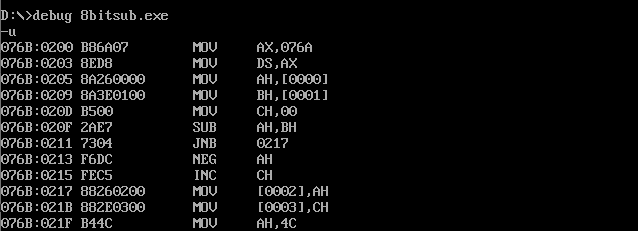
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AH register.
3. Move the data of second operator to BH register.
4. Load CH register with 00H.
5. Then sub AH and BH using SUB AH, BH and the value will be stored in AH register.
6. If the carry flag is not changed increment CH using “INC CH” and find the 2’s compliment of the result in AH using “NEG AH”.
7. If the carry flag is reset, jump to “here”.
8. Now store the data of AH register in result and CH 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:**

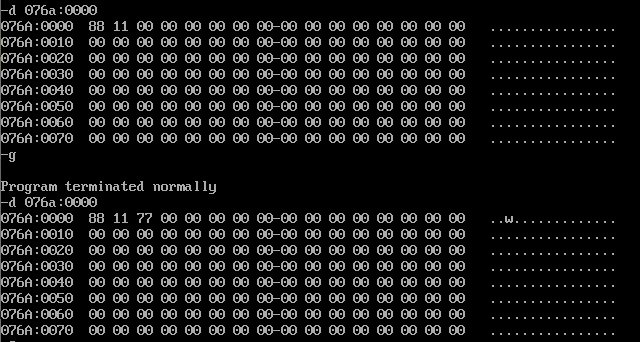
****

|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0200H | Memory instruction starts from 0200H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AH, OPR1 | Transfer the data from opr1 to AH register. |
| MOV BH, OPR2 | Transfer the data from opr2 to BH register. |
| MOV CH, OOH | Assign 00h to CH register. |
| SUB AH, BH | AH = AH - BH |
| JNC HERE | Jump if no carry to here |
| INC CH | Increment the value in the CH register. |
| NEG AH | Takes 2’s compliment for data in AH register. |
| HERE: | MOV RESULT, AH | Move the data from AH register to result. |
| MOV CARRY, CH | Move the data from CH register to carry. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

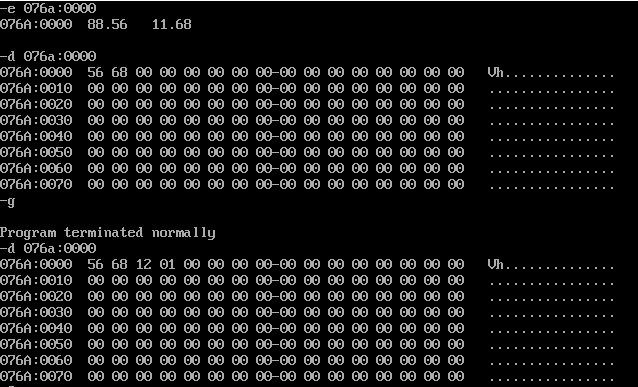
**Snapshot of sample input and output:**

****

**Without carry(op1 – 88, op2 – 11):**

****

**With carry(op1 – 56, op2 – 68):**

****

**Result:**

Thus the 8086 program for subtracting 2, 8bit numbers is executed successfully in

DOS-BOX.

**c) 8 bit multiplication:**

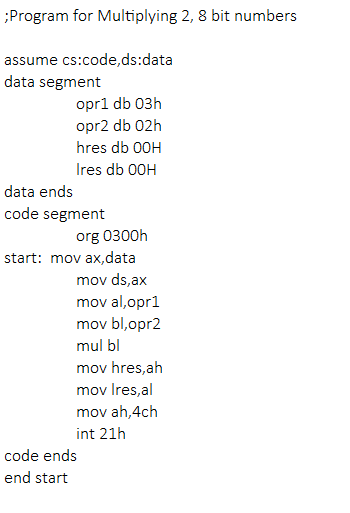
**Aim:**

Design 8086 program for 8-bit arithmetic multiplication.

**Algorithm:**

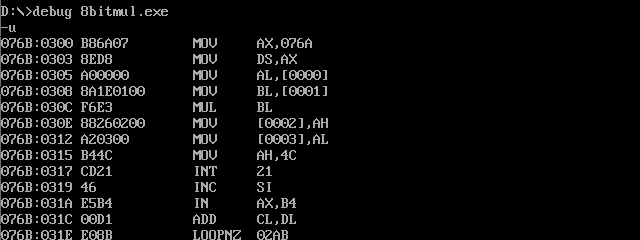
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the data of first operator to AL register.
3. Move the data of second operator to BL register.
4. Then multiply Al and BL using “MUL BL”.
5. AX(both higher and lower order) register will have the result of that multiplication.
6. Now store the data of AH register in hres and AL register in lres.
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:**

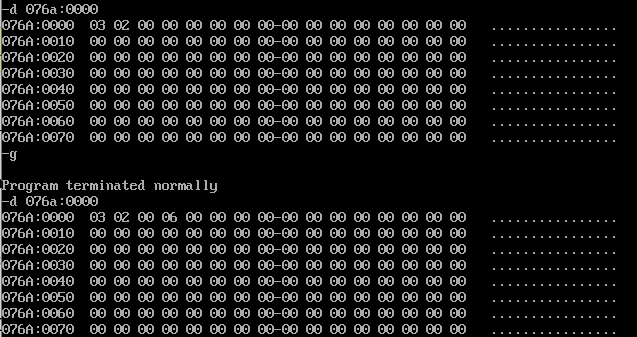
****

|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0300H | Memory instruction starts from 0300H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AL, OPR1 | Transfer the data from opr1 to AL register. |
| MOV BL, OPR2 | Transfer the data from opr2 to BL register. |
| MUL BL | AX = AL x BL |
| MOV HRES, AH | Move the data from AH register to hres. |
| MOV LRES, AL | Move the data from AL register to lres. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

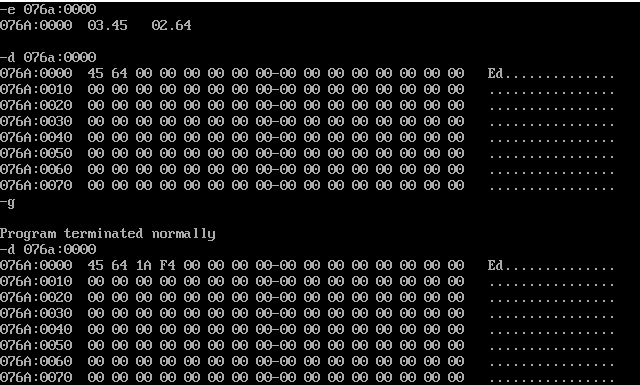
**Snapshot of sample input and output:**



**8bit result(op1 – 3, op2 – 2):**

****

**16bit result(op1 – 45, op2 – 64):**

****

**Result:**

Thus the 8086 program for multiplying 2, 8bit numbers is executed successfully in DOS-BOX.

**d) 8 bit division:**

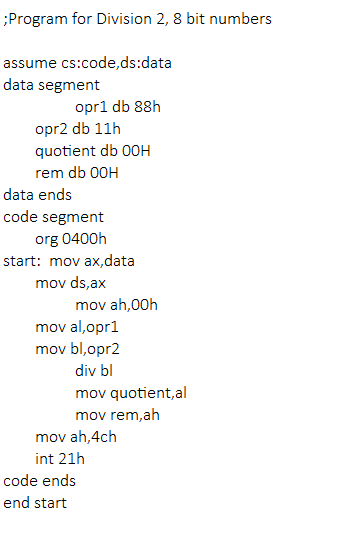
**Aim:**

Design 8086 program for 8-bit arithmetic division.

**Algorithm:**

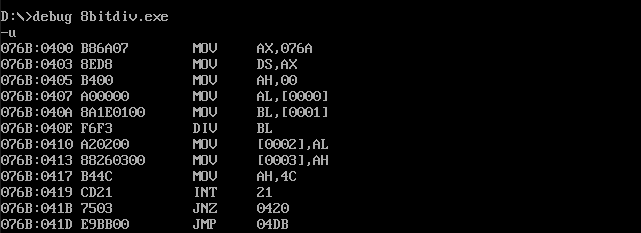
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Load AH register with 00H.
3. Move the data of first operator to AL register.
4. Move the data of second operator to BL register.
5. Then divide AX by BL using “DIV BL”.
6. After execution, quotient will be stored in AL and remainder is stored in AH register.
7. Now move the data from AL register to quotient and AH register to rem.
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:**

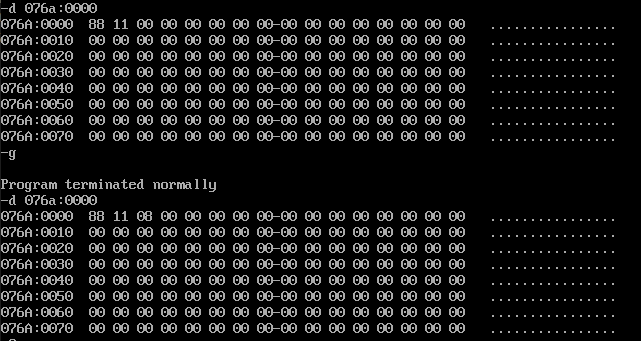
****

|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0400H | Memory instruction starts from 0400H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AH, OOH | Assign 00h to AH register. |
| MOV AL, OPR1 | Transfer the data from opr1 to AL register. |
| MOV BL, OPR2 | Transfer the data from opr2 to BL register. |
| DIV BL | Performs AX / BL. AL <- quotient. AH <- remainder. |
| MOV QUOTIENT, AL | Move the data from AL register to quotient. |
| MOV REM, AH | Move the data from AH register to rem. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

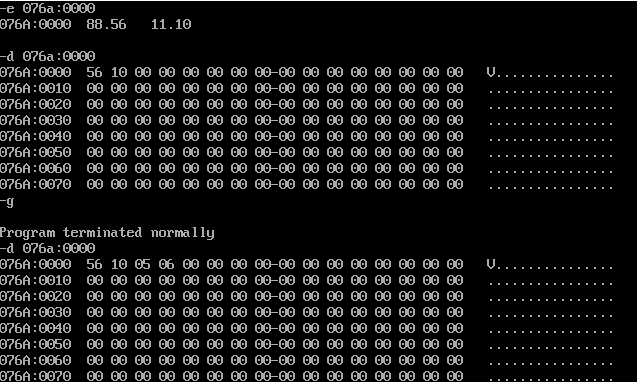
**Snapshot of sample input and output:**



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

****

**With rem(op1 – 56, op2 – 10):**

****

**Result:**

Thus the 8086 program for performing division with 2, 8bit numbers is executed successfully in DOS-BOX.