**SSN College of Engineering Department of Computer Science and Engineering**

**III year - UCS1512 – Microprocessors Lab**

**String Manipulations**

**Exp No:** 03

**Name:** Rahul Ram M

**Register Number:** 185001121

**Date:** 14/08/2020

**a) Moving a string of bytes:**

**Aim:**

Design 8086 program for moving a string of bytes.

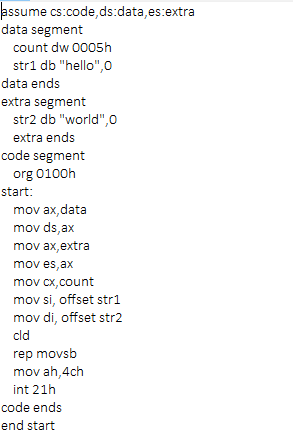
**Procedure for executing MASM:**

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:**

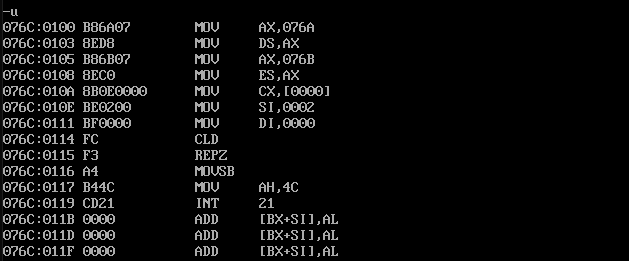
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the starting address of extra segment to AX register and move the data from AX register to ES register.
3. Move the value of the variable count to CX register.
4. Move the offset of the str1 to the SI register.
5. Move the offset of the str2 to the DI register.
6. Clear the direction flag using the command CLD so as to make the command REP MOVSB start moving the character from left to right.
7. REP MOVSB move the byte from the location specified by the SI register to the location specified by the DI register, incrementing the value of the SI and DI register by 2bytes at each iteration till CX register becomes 0.
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:**

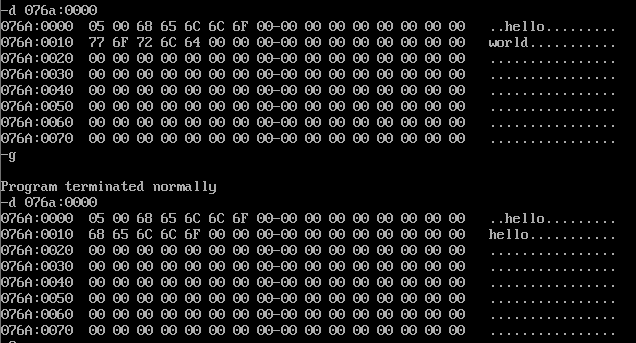


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0010H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, EXTRA  MOV ES, AX | Transferring the data from EXTRA to AX register and  from AX register to ES register. |
| MOV CX, count | Assign value of the count to CX register. |
| MOV SI, OFFSET STR1 | Move the offset value of STR1 To SI register. |
| MOV DI, OFFSET STR2 | Move the offset value of STR2 to DI register. |
| CLD | Clear the direction flag. |
| REP MPVSB | Move the string of bytes from SI’s location to the DI’s location. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

**Snapshot of sample input and output:**



**Moving ‘hello’ in the place of ‘world’:**



**Result:**

Thus the 8086 program for moving a string of bytes is executed successfully in DOS-BOX.

**b) Comparing 2 strings of bytes:**

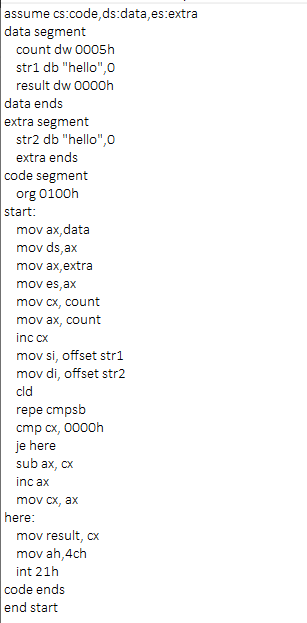
**Aim:**

Design 8086 program for comparing 2 strings of bytes.

**Algorithm:**

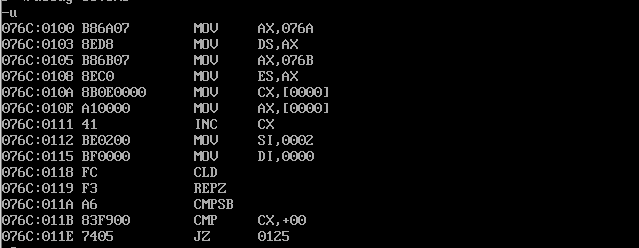
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the starting address of extra segment to AX register and move the data from AX register to ES register.
3. Move the value of the variable count to CX register.
4. Move the value of the variable count to AX register.
5. Increment the CX register.
6. Move the offset of the str1 to the SI register.
7. Move the offset of the str2 to the DI register.
8. Clear the direction flag using the command CLD so as to make the command REPE CMPSB start comparing the character from left to right.
9. REPE CMPSb decrements the value of count at each iteration before comparing. It compares the byte in the SI register with DI register. If no difference is found, the iteration will stop when count reaches 0 else it stops as soon as a different character is found.
10. Comparing CX register with 0000H.
11. If same then unconditional jump to ‘HERE’.
12. Subtract the value of the CX register with AX (length of the string) register.
13. Increment the value in the AX register and move it to CX register.
14. HERE: Move the value of the CX register to RESULT.
15. Move the hexadecimal value 4C into AH register.
16. 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:**

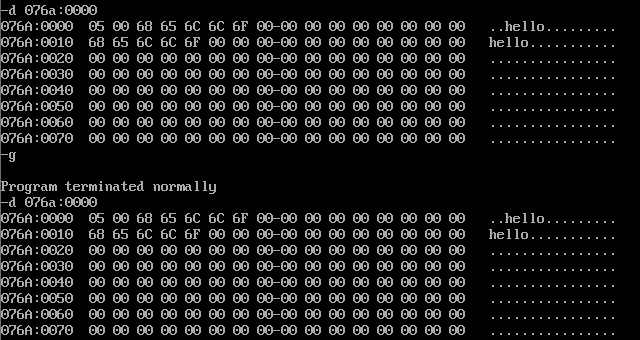


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, EXTRA  MOV ES, AX | Transferring the data from EXTRA to AX register and from AX register to ES register. |
| MOV CX, count | Assign value of the count to CX register. |
| MOV SI, OFFSET STR1 | Move the offset value of STR1 To SI register. |
| MOV DI, OFFSET STR2 | Move the offset value of STR2 to DI register. |
| CLD | Clear the direction flag. |
| REPE CMPSB | Compare 2 strings of bytes from SI and DI register. |
| CMP CX, 0000H | Comparing the value of CX register with 0000H. |
| JE HERE | Jump to HERE if equal. |
| SUB AX, CX | Subtract CX from AX and store the result in AX. |
| INC AX | Increment the AX register. |
| MOV CX, AX | Move data from AX register to CX register. |
| HERE: | MOV RESULT, CX | Move the data from CX register to result. |
| MOV AH, 4CH  INT 21H | Terminates the program. |

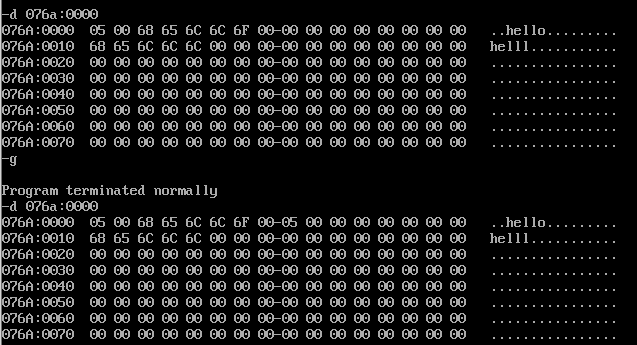
**Snapshot of sample input and output:**



**Comparing ‘hello’ and ‘hello’ (Answer = 0 Same string):**



**Comparing ‘hello’ and ‘helll’ (Answer = 05 index:5):**



**Result:**

Thus the 8086 program for comparing 2 strings of bytes is executed successfully in

DOS-BOX.

**c) Searching a byte in a string:**

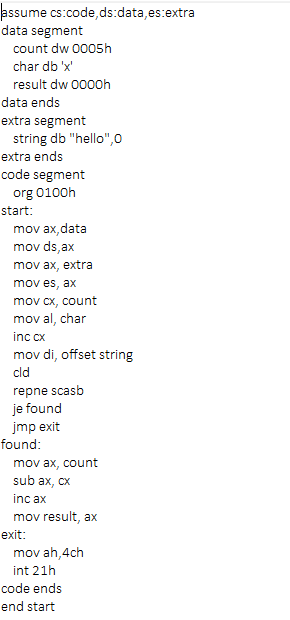
**Aim:**

Design 8086 program for searching a byte in a string.

**Algorithm:**

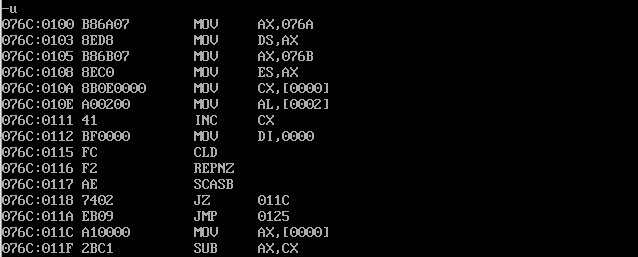
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the starting address of extra segment to AX register and move the data from AX register to ES register.
3. Move the value of the variable count to CX register.
4. Move the value of the variable char to AL register.
5. Increment the CX register.
6. Move the offset of the string to the DI register.
7. Clear the direction flag using the command CLD so as to make the command REPNE SCASB start searching for the character from left to right.
8. REPNE CMPSb decrements the value of count at each iteration before comparing. It compares the byte in the SI register with DI register. If no matching is found, the iteration will stop when count reaches 0 else it stops as soon as it finds a match.
9. If a match is found jump to ‘FOUND’ using JE (jump if equal).
10. Else Jump to ‘EXIT’ using unconditional jump.
11. FOUND: Move the value of count (length of the string) to AX register.
12. Subtract the value of the CX register with AX (length of the string) register.
13. Increment the value in the AX register and move it to RESULT.
14. EXIT: Move the hexadecimal value 4C into AH register.
15. 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:**

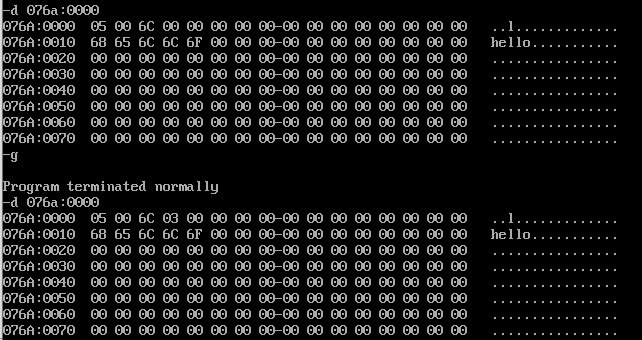


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV AX, EXTRA  MOV ES, AX | Transferring the data from EXTRA to AX register and from AX register to ES register. |
| MOV CX, count | Assign value of the count to CX register. |
| MOV AL, CHAR | Move the value of char to AL register. |
| INC CX | Increment the CX register. |
| MOV DI, OFFSET STRING | Move the offset value of STRING to DI register. |
| CLD | Clear the direction flag. |
| REPNE SCASB | Search for character in Al register in the string offset by DI register. |
| JE FOUND | Jump to FOUND if equal (if the character is found in the string). |
| JMP EXIT | Jump to EXIT. |
| FOUND: | MOV AX, COUNT | Move COUNT to AX register. |
| SUB AX, CX | Subtract CX from AX and store the result in AX. |
| INC AX | Increment the AX register. |
| MOV RESULT, AX | Move the data from AX register to RESULT. |
| EXIT: | MOV AH, 4CH  INT 21H | Terminates the program. |

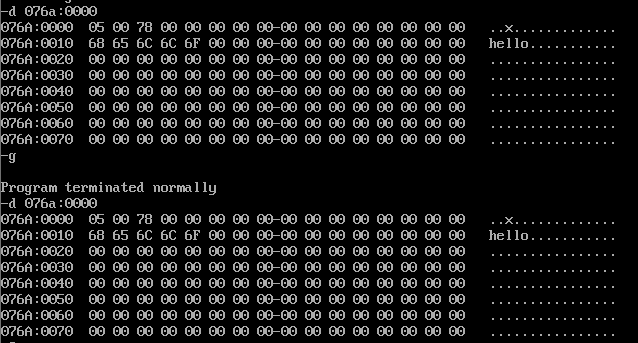
**Snapshot of sample input and output:**



**Searching ‘l’ in ‘hello’ (Answer = 3 index:3):**



**Searching ‘x’ in ‘hello’ (Answer = 0 Not found):**



**Result:**

Thus the 8086 program for searching a character in a string is executed successfully in DOS-BOX.

**d) Moving a string without using string instructions:**

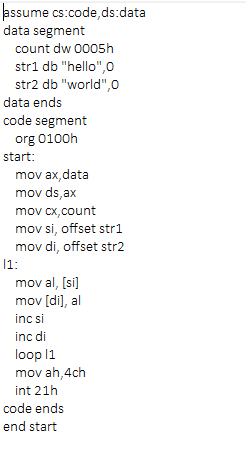
**Aim:**

Design 8086 program for moving a string without using string instructions.

**Algorithm:**

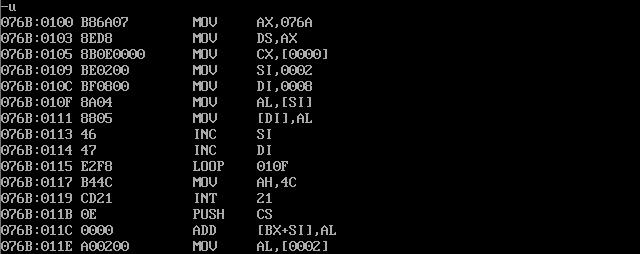
1. Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move the value of the variable count to CX register.
3. Move the offset of the str1 to the SI register.
4. Move the offset of the str2 to the DI register.
5. L1: Move the value in the location specified by SI register to Al register.
6. Move the value in the AL register to the location specified by DI register.
7. Increment SI and DI register.
8. Loop to L1.
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:**

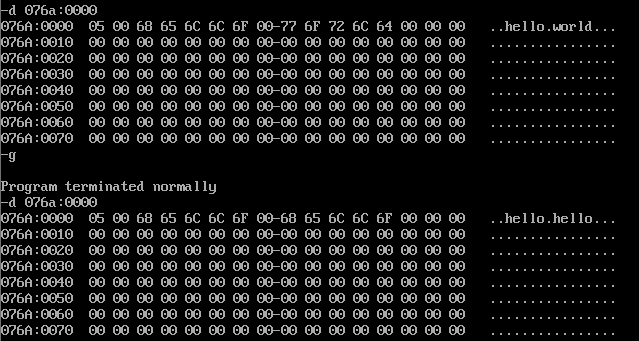


|  |  |  |
| --- | --- | --- |
|  | **Program** | **Comments** |
| START: | ORG 0100H | Memory instruction starts from 0100H. |
| MOV AX, DATA  MOV DS, AX | Transferring the data from DATA to AX register and  from AX register to DS register. |
| MOV CX, count | Assign value of the count to CX register. |
| MOV SI, OFFSET STR1 | Move the offset value of STR1 To SI register. |
| MOV DI, OFFSET STR2 | Move the offset value of STR2 to DI register. |
| L1: | MOV AL, [SI] | Move the value at SI to AL register |
| MOV [DI], AI | Move the AL to DI. |
| INC SI | Increment SI register. |
| INC DI | Increment DI register. |
| LOOP L1 | Loop to L1 |
| MOV AH, 4CH  INT 21H | Terminates the program. |

**Snapshot of sample input and output:**



**Moving ‘hello’ in the place of ‘world’:**



**Result:**

Thus the 8086 program for moving a string without using string instructions is executed successfully in DOS-BOX.