Laboratory 7

Title of the Laboratory Exercise: String manipulation

1. Introduction and Purpose of Experiment

   Students will be able to perform all string manipulations in assembly language

2. Aim and Objectives

   Aim

   To develop assembly language program to perform all string operations like inserting a byte, deleting a byte and copying a string as a sub-string

   Objectives

   At the end of this lab, the student will be able to

   - Identify instructions for performing string manipulation
   - Use indexed addressing mode
   - Apply looping instructions in assembly language
   - Use data segment to represent arrays

3. Experimental Procedure

   1. Write algorithm to solve the given problem
   2. Translate the algorithm to assembly language code
   3. Run the assembly code in GNU assembler
   4. Create a laboratory report documenting the work

4. Questions

   Develop an assembly language program to perform the following

   1. Copy the contents of MSG1 to MSG2
   2. Copy the contents of MSG1 to MSG3 in reverse order
   3. Develop an assembly language program to compare two strings and print a message "Equal" if they are equal, "Not Equal" if they are not equal.

5. Calculations/Computations/Algorithms

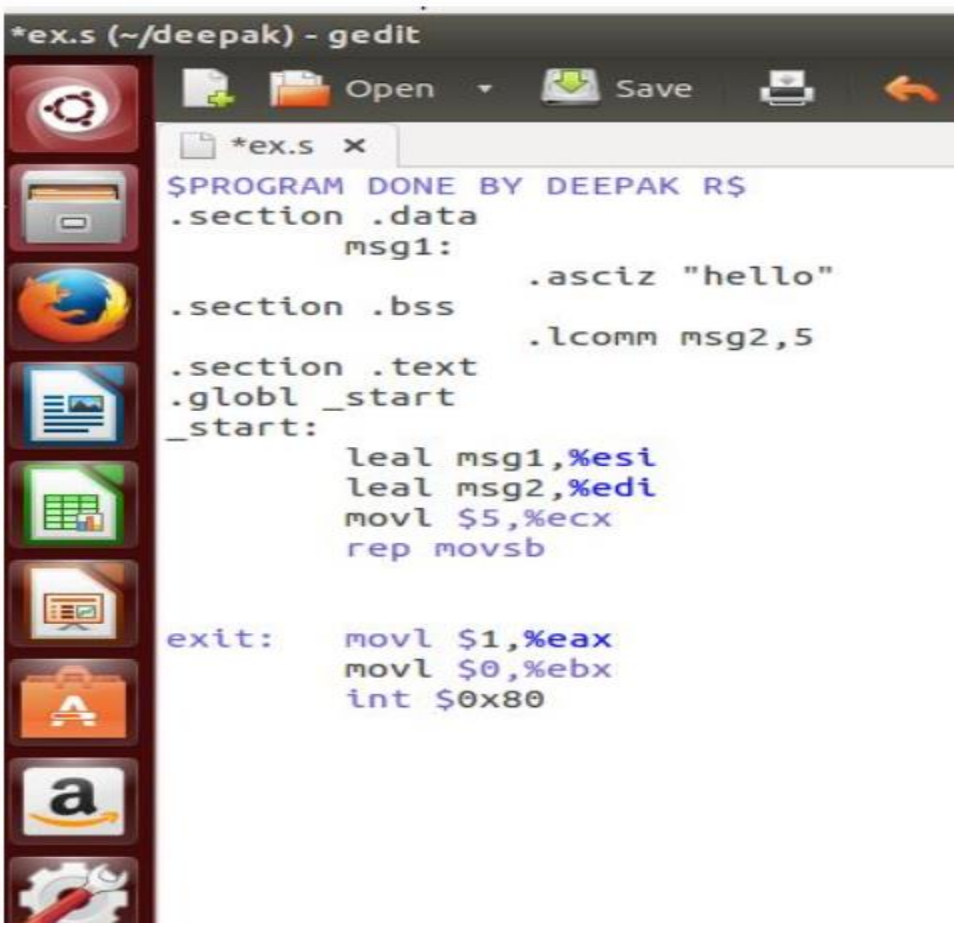

Fig 1 program to Copy the contents of MSG1 to MSG2

ex.s (~/deepak) - gedit

Open Save

ex.s

```
.section .data
        msg1:
                .asciz "hello"
        msg3:
                .asciz "      "
.section .text
.globl _start
_start:
        movl $4 ,%ecx
        movl $0 ,%edx
loop:
        movb msg1( ,%ecx,1),%al
        movb %al,msg3( ,%edx,1)
        addl $1,%edx
        subl $1,%ecx
        cmpl $5,%ecx
        jne loop

exit:   movl $1,%eax
        movl $0,%ebx
        int $0x80
```

Fig 2 program to Copy the contents of MSG1 to MSG3 in reverse order

ex.s (~/deepak) - gedit

Open Save

ex.s

```
.section .data
        s1:
                .asciz "hi welcome"
        s2:
                .asciz "hi welcome"
.section .text
.globl _start
_start:
        movl $10,%eax
        movl $10,%ebx
        cmpl %eax,%ebx
        jne exit
        movl $0,%esi
        movl $0,%edi
loop:
        movb s1( ,%esi,1),%cl
        movb s2( ,%edi,1),%dl
        cmpb %cl,%dl
        jne exit

        addl $1,%esi
        cmpl %esi,%eax
        jne loop2
        addl $1,%edi

        cmpl %edi,%ebx
        jne exit

        movl $1,%ebp
        jmp exit
```

```
        jmp exit
loop2:
        addl $1,%edi
        cmpl %edi,%ebx
        jne loop

exit:   movl $1,%eax
        movl $0,%ebx
        int $0x80
```

Fig 3 program to program to compare two strings

6. Presentation of Results

```
Starting program: /home/mplab/deepak/ex

Breakpoint 1, exit () at ex.s:15
15      exit:   movl $1,%eax
(gdb) x/s &msg2
0x804909c <msg2>:       "hello"
(gdb)
```

Fig 4 Result of program to Copy the contents of MSG1 to MSG2

```
13              movb %al,msg
(gdb) x/s &msg3
0x80490a9:      "olleh\200",
(gdb)
```

Fig 5 Result of program to Copy the contents of MSG1 to MSG3 in reverse order

```
Starting program: /home/mplab/deepak/ex

Breakpoint 1, exit () at ex.s:37
37      exit:   movl $1,%eax
(gdb) info registers
eax            0xa      10
ecx            0x65     101
edx            0x65     101
ebx            0xa      10
esp            0xbffff050       0xbffff050
ebp            0x1      0x1
esi            0xa      10
edi            0xa      10
eip            0x80480ba        0x80480ba <exit>
eflags         0x246    [ PF ZF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb)
```

Fig 6 Result of program to program to compare two strings

7. Conclusions

Instruction such as movsb, movsl, are used to move bytes and words from source register to destination register, which are esi and edi respectively.

To repeat an instruction, rep instruction is used, this is used to make a loop like construct the copy strings, and also to compare strings.

8. Comments

1. Limitations of Experiments

The length of the string to be copied has to be known to know how many characters has to be copied.

2. Limitations of Results

The destination memory which is assigned in the uninitialized bss segment is fixed size, hence strings of larger sizes could overflow the memory.

3. Learning happened

The concept of strings and various string operations in assembly is learnt in this lab.

4. Recommendations

The source and destination registers should be carefully taken and the DF flag must be cleared using the cld instruction

Signature and date

Marks