WORKING WITH STRINGS | SEC204

Overview

- Introduction
- Strings functions

INTRODUCTION

INTERACTING WITH HUMANS

- C provides functio interaction with humans. Assembly only provides basi ality
  - Moving strings
  - Storing and loading strings
  - Comparing strings
  - Scanning
  - Finding string length

- Character is 1 byte long in C. What more do we need to represent strings?

STRINGS FUNCTIONS

MOVING STRINGS

- movs
  Moves string from source location. It implies as source the ESI register (memory location of the string) and destination the EDI register (memory location of the destination.

- Can define the size of bytes to be moved
  - **movsl**: l for 32-bit long word value
  - **movsw**: w for 16-bit word value
  - **movsb**: b for 8-bit byte value
- What does the movstest1 program do?

In command line:

```
$ as –gstabs –o movtest1.o movtest1.s
$ ld –o movtest1 movtest1.o
$ gdb movtest1

(gdb) x/s &output
```

```
# movstest1.s - MOVS instructions
.section .data
value1:
    .ascii "This is a test string.\n"
.section .bss
    .lcomm output, 23
.section .text
.globl _start
_start:
    nop
    leal value1, %esi
    leal output, %edi
    movsb
    movsw
    movsl

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

REP STOS - REPEAT STORE STRING

- REP is not an instruction by itself
  - It repeats a single instruction a number of times. The number of repetitions is controlled by the value in the %ECX register
  - All rep operations use ECX register as a "counter" to determine how many times to loop through the instruction. Each time it executes, it decrements ECX. Once ECX = 0, it stops the repetition.
- STOS is an instruction. It places a string value in another location.
  - Stosb, stosw, stosl moves one byte, 1 word or 1 dword respectively at a time
  - The EDI register holds the implied destination.
- So there are 3 pieces which must happen before the actual REP STOS occurs:
  - set EDI to the destination;
  - Set EAX/AL to the value to store;
  - Set ECX to the number of times to repeat store

REP STOS EXAMPLE

```
# stostest1.s - An example of the STOS instruction
.section .data
space:
   .ascii " "
.section .bss
   .lcomm buffer, 256
.section .text
.globl _start
_start:
   nop
   leal space, %esi
   leal buffer, %edi
   movl $256, %ecx
   cld
   lodsb
   rep stosb

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


In gdb:

```
(gdb) step
(gdb) step
(gdb) print/x $eax
(gdb) x/10b &buffer
(gdb) step
(gdb) x/10b &buffer
```

FURTHER READING

- Professional Assembly Language, chapter 10