Array String Instructions

Ned Nedialkov

McMaster University
Canada

SE 3F03
March 2014

Outline

String Instructions

LOADSx, STOSx

MOVSx

Example

REP prefix

CMPSx, SCASx

Examples

String Instructions

- Defined to work with arrays
- Use **ESI** and **EDI**
- Automatically increment/decrement them
- SDT sets the direction flag **DF**. If set, **ESI** and **EDI** are decremented
- **CLD** clears the direction flag
  ESI and **EDI** are incremented

LOADSx, STOSx

- Load a byte/word/double word

instruction		source	incr/decr
LODSB	**AL**	[DS:ESI]	ESI = ESI ± 1
LODSW	**AX**	[DS:ESI]	ESI = ESI ± 2
LODSD	**EAX**	[DS:ESI]	ESI = ESI ± 4

- Store a byte/word/double word

instruction		source	incr/decr
STOSB	**AL**	[ES:EDI]	EDI = EDI ± 1
STOSW	**AX**	[ES:EDI]	EDI = EDI ± 2
STOSD	**EAX**	[ES:EDI]	EDI = EDI ± 4

MOVSx

- Move instructions

instruction	[ES:EDI] ← [DS:ESI]	incr/decr
MOVSB	byte	EDI, EDI± 1
MOVSW	word	EDI, EDI± 2
MOVSD	byte	EDI, EDI± 4

Load/store example

From http://www.drpaulcarter.com/pcasm/

```
        segment .data
array1 dd 1,2,3,4,5,6,7,8,9,10
        segment .bss
array2  resd 10
        segment .text
        cld                ;clear direction flag
        mov    esi, array1 ;store addresses
        mov    edi, array2
        mov    ecx, 10     ;set counter to 10
lp:
        lodsd  ;load dword from array1
        stosd  ;store dword into array2
loop lp
```

REP prefix

- Repeat the next instruction **ecx** times
- Example

```
lp:
          lodsd  ;load dword from array1
          stosd  ;store dword into array2
loop lp
          ;; is the same as
          rep movsd
```

- **REPE, REPZ**
  - repeat while ZF=1 or at most **ECX** times
- **REPNE, REPNZ**
  - repeat while ZF=0 or at most **ECX** times

Comparison instructions

- `CMPSx` compares `x = B/W/D` at [DS:ESI] and [ES:EDI]
  Increments/decrements by 1/2/4
- `SCASx` compares **AL**/**AX**/**EAX** and B/W/D at [ESI:EDI]
  Increments/decrements by 1/2/4

Examples

- From http://www.drpaulcarter.com/pcasm/

```
;; compare two blocks of memory
segment .text
cld
mov     esi, block1 ;set addresses
mov     edi, block2
mov     ecx, size   ;block size
repe    cmpsb
je  ...
;; exit at the first two different bytes
;;
```

- **repe** if two bytes are not equal, exit
  - ZF is cleared
- **repe** if all bytes are equal
  - **ecx** is 0
  - ZF is set

From http://www.drpaulcarter.com/pcasm/

```
        ;; copy string
        ;; void asm_strcpy( char * dest, char *src)
%define dest [ebp + 8]
%define src [ebp + 12]
_asm_strcpy:
        enter 0,0
        push esi
        push edi
        mov edi, dest
        mov esi, src
        cld
cpy_loop:
        lodsb                  ;load from src into al
        stosb                  ;store into dest
        or al,al               ;if both not 0 repeat
        jnz cpy_loop
        pop     edi
        pop     esi
        leave
        ret
```