Lecture 11: String Instructions

80x86 is equipped with special instructions to handle string operations

String: A series of data words (or bytes) that reside in consecutive memory locations

Permits operations:

• Move data from one block of memory to a block elsewhere in memory,

•Scan a string of data elements stored in memory to look for a specific value,

•Compare two strings to determine if they are the same or different.

Five basic String Instructions define operations on one element of a string :

- Move byte or word string MOVSB/MOVSW
- Compare string CMPSB/CMPSW
- Scan string SCASB/SCASW
- Load string LODSB/LODSW
- Store string STOSB/STOSW

Repetition is needed to handle more than one element of a string.

Auto-indexing of String Instructions

Execution of a string instruction causes the address indices in SI and DI to be either automatically incremented or decremented. The decision to increment or Decrement is made based on the status of the direction flag.

Lecture 11: String Instructions

The direction Flag: Selects the auto increment (D=0) or the auto decrement (D=1) operation for the DI and SI registers during string operations.

Mnemonic	Meaning	Format	Operation	Flags Affected
CLD	Clear DF	CLD	(DF) 0	DF
STD	Set DF	STD	(DF) 1	DF

CLD Clears the D flag/ STD Sets the D flag

String Instructions

Mnemonic	Meaning	Format	Operation	Flags Affected
MOVS	Move string	MOVSB/ MOVSW	((ES))0+(DI) (DS)0+(SI) (SI) (SI)±1 or 2 (DI) (DI)±1 or 2	None
CMPS	Compare string	CMPSB/ CMPSW	Set flags as per ((DS))0+(SI) - (ES)0+(DI) (SI) (SI)±1 or 2 (DI) (DI)±1 or 2	CF,PF,AF,ZF,SF,OF
SCAS	Scan string	SCASB/ SCASW	Set flags as per (AL or AX) - (ES)0+(DI) (DI) (DI)±1 or 2	CF,PF,AF,ZF,SF,OF
LODS	load string	LODSB/ LODSW	(AL or AX) (DS)0+(SI) (SI) (SI)±1 or 2	None
STOS	Store string	STOSB/ STOSB	(ES)0+(DI) (AL or AX)±1 or 2 (DI) (DI)±1 or 2	None

String Instructions

EX: using string operation implement the previous example to copy block Of memory to another location.

Lecture 11: String Instructions

```
        MOV AX,DATASEGADDR
        MOV DS,AX
        MOV ES,AX
        MOV SI,BLK1ADDR
        MOV DI,BLK2ADDR
        CLD
NXTPT:  MOVSB
        LOOP NXTPT
        HTL
```

EX: Explain the function of the following sequence of instructions

```
        MOV DL, 05
        MOV AX, 0A00H
        MOV DS, AX
        MOV SI, 0
        MOV CX, 0FH
AGAIN:  INC SI
        CMP [SI], DL
        LOOPNE AGAIN
```

The first 5 instructions initialize internal registers and set up a data segment The loop in the program searches the 15 memory locations starting from Memory location A001H for the data stored in DL (05H). As long as the value In DL is not found the zero flag is reset, otherwise it is set. The LOOPNE Decrements CX and checks for CX=0 or ZF =1. If neither of these conditions is Met the loop is repeated. If either condition is satisfied the loop is complete. Therefore, the loop is repeated until either 05 is found or all locations in the address range A001H through A00F have been checked and are found not to contain 5.

EX: Implement the previous example using SCAS instruction.

Lecture 11: String Instructions

```
        MOV AX, 0H
        MOV DS, AX
        MOV ES, AX
        MOV AL, 05
        MOV DI, 0A000H
        MOV CX, 0FH
        CLD
AGAIN:  SCASB
        LOOPNE AGAIN
```

Ex: The following program loads the block of memory locations from A000H Through 0A00FH with number 5H.

```
        MOV AX, 0H
        MOV DS, AX
        MOV ES, AX
        MOV AL, 05
        MOV DI, 0A000H
        MOV CX, 0FH
        CLD
AGAIN:  STOSB
        LOOP AGAIN
```

Repeat String REP

Basic string operations must be repeated in order to process arrays of data; this is done by inserting a repeat prefix.

Prefix	Used with	Meaning
REP	MOVS STOS	Repeat while not end of string CX≠0
REPE/REPZ	CMPS SCAS	Repcat while not end of string and strings are equal CX≠0 and and ZF=1
REPNE/REPNZ	CMPS SCAS	Repeat while not end of string and strings are not equal CX≠0 and ZF=0

Lecture 11: String Instructions

EX: write a program to copy a block of 32 consecutive bytes from the block Of memory locations starting at address MASTER in the current data segment (DS) to a block of locations starting at address COPY in the current extra Segment (ES)

```
CLD
MOV AX, DATA_SEG
MOV DS, AX
MOV AX, EXTRA_SEG
MOV ES, AX
MOV CX, 20H
MOV SI, OFFSET MASTER
MOV DI, OFFSET COPY
REPZMOVSB
```

Example. Find and replace

Write a program that scans the name “Mr. Gones” and replaces the letter “G” with the letter “J”.

```
DATA1 DB  ‘Mr. Gones’,’$’
.CODE
        MOV ES, DS
        CLD                     ; reset auto increment bit D=0
        MOV DI, OFFSET DATA1
        MOV CX, 09              ; number of chars to be scanned
        MOV AL, ‘G’             ; char to be compared against
        REPNE SCASB             ; start scan   AL=? ES:[DI]
        JNE OVER                ; if z=0 (if G is not found)
        DEC DI                  ; z=1    (if found G)
        MOV BYTE PTR [DI], ‘J’
OVER:   MOV AH, 09
        MOV DX, OFFSER DATA1
        INT 21H                 ;display the resulting string
```