CT215-7
Strings

Objectives

- String operations -- copy a string from a register/memory to a register/memory
- String instructions
  - **MOVS**
  - **LODS**
  - **STOS**
  - **CMPS**
  - **SCAS**
  - **REP**

String Instructions

- **A string instruction consists of a repeated process of an operation on each character (or each group of 2 or 4 characters) of a string.**
- **Each string instruction has a byte, a word, or a double-word version**
  - MOVSB MOVSW MOVSD
  - LODSB LODSW LODSD

String Instructions

- **String instructions assume the use of the** DS:SI **or** ES:DI **pair of registers.**
  - DS and ES have the initial address of the data segment
  - SI and DI are used to store offset of strings
  - initialization of ES register

```
MOV  AX, @data
MOV  DS, AX
MOV  ES, AX
```

Example

- **Using explicit operands**

```
Byte1    DB  'A'
Byte2    DB  '*'
         MOVS Byte2,Byte1
```

- **Using implicit operands**

```
LEA   DI,Byte2
LEA   SI,Byte1
MOVSB
```

Implied Operands

Instruction	Implied Operand
MOVS	**ES:DI, DS:SI**
LODS	**AX, DS:SI**
STOS	**ES:DI, AX**
CMPS	**DS:SI, ES:DI**
SCAS	**ES:DI, AX**

Example

```
Byte1   DB      'A'
Byte2   DB      '*'

        LEA DI,Byte2    ; address of Byte2
        LEA SI, Byte1   ; Address of Byte1
        MOVSB           ; copy  Byte1 to Byte2
        LODSB           ; copy Byte1 to AL
        STOSB           ; copy AL to Byte2
        CMPSB           ; compare Byte1 to Byte2
        SCASB           ; compare Byte2 to AL
```

- Note: Single character operations. No repetition

Example

```
String1  DB    'CS2401:Assembly Language'
String2  DB    24    DUP('  ')
         MOV CX,24      ; Number of Characters
         LEA DI,String2
         LEA SI,String1
         JCXZ OUT1      ;CX initially 0 get out
IN1:     MOV  AL, [SI]
         MOV  [DI], AL
         INC  DI
         INC  SI
         LOOP IN1 ;Decrement CX and repeat
OUT1:    ...
```

String Copy

```
        MOV   CX, 24 ; Number of Characters
        LEA   DI, STRING2
        LEA   SI, STRING1
        JCXZ  OUT1   ;CX initially 0 get out
IN1:    MOV   AL, [SI]
        MOV   [DI], AL
        INC   DI
        INC   SI
        LOOP  IN1          ; Decrement CX
OUT1:   ...
```

Right to Left String Copy

```
      MOV  CX,24 ;Number of characters
      LEA  DI,String2 + 23
      LEA  SI,String1 + 23
      JCXZ OUT1
IN1:  MOV  AL,[SI]
      MOV  [DI],AL
      DEC  DI
      DEC  SI
      LOOP IN1
OUT1: ...
```

String Copy

```
        MOV   CX, 12       ; Number of words
        LEA   DI, STRING2
        LEA   SI, STRING1
        JCXZ  OUT1
IN1:    MOV   AX, [SI]
        MOV   [DI], AX
        ADD   DI, 2
        ADD   SI, 2
        LOOP  IN1
OUT1:   ...
```

String Copy Instruction

```
MOV   CX, 24       ; Number of Characters
LEA   DI, STRING2
LEA   SI, STRING1
CLD                ; Clear DF to copy from
                   ; left to right
REP   MOVSB ; Copy 24  bytes
```

- REP repeats the MOVSB operation until CX becomes 0
- After each repetition, REP decrements CX by 1
- If DF is 0, DI and SI are incremented by 1 after each repetition

REP MOVSB

- **Copy [SI] to [DI]**
- **IF (DF = 0) Increment DI and SI by 1**

  ELSE IF (DF =1) Decrement DI and SI by 1
- **Decrement CX by 1**
- **IF (CX=0) stop**

  ELSE repeat step1-step4

Right to Left String Copy

```
MOV CX, 24              ;Number of Characters
LEA DI, String2 + 23
LEA SI, String1 + 23
STD                     ;Set Direction Flag to 1
                        ;to copy from right to left
REP MOVSB               ; Copy 24  bytes
```

- If DF is 1, DI and SI are decremented by 1 after each repetition

String Copy: Word by word

```
MOV CX, 12 ; Number of words
LEA DI, STRING2
LEA SI, STRING1
CLD              ; Clear Direction Flag
                 ; to copy from left to right
REP MOVSW        ; Copy 12 words
```

- After each repetition, REP increments SI and DI by 2

and decrements CX by 1 until it becomes repetition.

Load String

- **Copy one byte, word, or doubleword from DS:SI to AL, AX, or EAX.**

Example:

```
          STRING1     DB    'HELLO'
          ..........
          LEA   SI, STRING1

  LODSB   MOV   AL, [SI]
          INC   SI
```

Example

- **Search for a character in a string. Set BL to 1 if found; set BL to 2 otherwise.**

```
STRING1  DB    'HELLO'
         ...
         LEA   SI,STRING1
         MOV   CX,5
         MOV   BL,02 ;Not found yet
IN1:     LODSB
         CMP   AL,'*'
         JE    FOUND  ; if found search is over
         LOOP  IN1
         JMP   OUT1
FOUND:   MOV   BL,01  ; found
OUT1:    ...
```

Example

- **Copy a string in reverse order**

```
STRING1     DB    'HELLO'
STRING2     DB    5  DUP ('  ')
            ..........
            LEA   SI, STRING1
            LEA   DI, STRING2 + 4
            MOV   CX, 5
IN1:        LODSB
            MOV   [DI], AL
            DEC   DI
            LOOP  IN1
```

Store

- **Stores the contents of AL, AX, or EAX in ES:DI**

Example:

```
STRING1   DB   'HELLO'
          ...
          LEA  DI,STRING1
  STOSB    |    MOV  [DI],AL
           |    INC  DI
```

Example

- **Initialize a data area with a string.**

  Example: Clear a string

```
STRING1  DB     'CS2401:Aseembly Language'
         ...
         CLD                ;Clear DF
         MOV    AX, 2020H   ; AX = blank blank
         MOV    CX, 12      ; 12 blank words
         LEA    DI, STRING1
         REP    STOSW       ; repeat until CX is 0
```

CMPS

- **Used to compare strings addressed by DS:SI and ES:DI**
- **Strings are compared character by character**
- **Depending on the direction flag (DF), CMPS increments or decrements DI and SI after each repetition (by 1 for byte, by 2 for word, or 4 for doubleword)**

CMPS

- **CMPS also sets the AF, CF, OF, PF, SF, and ZF flags based on the comparison.**
- **The comparison is based on the ASCII values of the characters.**
- **For string comparison, use with REPE or REPNE**

REPnn

- **REP: repeat operation while CX is not 0**
- **REPE or REPZ: Repeat operation while equal (i.e. ZF=1) and CX is not 0. Stop repetition if CX is 0 or if not equal (i.e. ZF = 0).**
- **REPNE or REPNZ: Repeat operation while not equal (i.e. ZF=0) and CX is not 0. Stop repetition if CX is 0 or if equal (i.e. ZF = 1).**

Example

- **Compare two strings. If they are equal, set BL to 1; otherwise set BL to 2.**

```
STRING1   DB     'Jack'
STRING2   DB     'Jill'
          CLD
          MOV     CX,  4
          LEA     SI, STRING1
          LEA     DI, STRING2
          MOV     BL, 01   ; Assume they are equal
          REPE    CMPSB    ; if equal compare next two char
          JE      A10      ; If equal exit
          MOV     BL, 02   ; not equal set BL to 2
A10:      ...
```

SCAS

- **Used to scan a string in search for a specific byte, word or doubleword.**
- **SCAS compares the contents of ES:DI with the contents of AL, AX, or EAX (SCASB, SCASW, SCASD).**
- **SCAS set the AF, CF, OF, PF, SF and ZF flags based on the ASCII comparison.**

Example

- **Search for a character in a string**

```
STRING1 DB     'HELLO'
        ...
        LEA    SI, STRING1
        MOV    CX, 5
        MOV    BL, 02      ;Not found yet
 IN1:   LODSB
        CMP    AL, '*'
        JE     FOUND       ;if found search is over
        LOOP   IN1
        JMP    OUT1
FOUND:  MOV    BL, 01      ;found
OUT1:   ...
```

SCAS: Example

```
STRING1 DB     'HELLO'
        ...
        LEA    DI, STRING1
        CLD
        MOV    AL, '*'
        MOV    CX, 5
        MOV    BL, 02        ;Not found yet
        REPNE  SCASB
        JNE    OUT1
        MOV    BL, 01         ;found
OUT1:   ...
```