

# Assembly Instructions and Addressing Modes

## Topics

- Introduction of Assembly Instructions
  - Data transfer instructions
  - Basic arithmetic instructions
  - Repetitive move instructions
  - System call (Software Interrupt) instruction
- Addressing Modes for the operands of instructions
  - Register addressing
  - Immediate addressing
  - Direct memory addressing
  - Direct-offset addressing
  - Indirect memory addressing
  - Base displacement addressing
  - Base-index Addressing
  - Base-index with displacement addressing

## Data transfer instructions

### MOV Instruction

- `[label:] mov reg/mem, reg/mem/imm`
- move data from the source to the destination

### LEA Instruction

- `[label:] lea reg, mem`
- LEA: load effective address of the location name

**Arithmetic Instructions** ADD/SUB Instruction

- [label:] add/sub reg/mem, reg/mem/imm
- $\text{destination} \leftarrow \text{destination} + / - \text{source}$

## INC/DEC Instruction

- [label:] inc/dec reg/mem
- increment/decrement the destination

Enough to have an example now!

```

TITLE    A06MOVE (EXE)  Repetitive move operations
                .MODEL SMALL
                .STACK 64

;-----
                .DATA
HEADNG1 DB      'InterTech'
HEADNG2 DB      9 DUP ('*'), '$'
;-----
                .CODE
A10MAIN PROC    FAR
                MOV     AX,@data      ;Initialize segment
                MOV     DS,AX          ; registers
                MOV     ES,AX

                MOV     CX,09          ;Initialize to move 9 chars
                LEA     SI,HEADNG1     ;Initialize offset addresses
                LEA     DI,HEADNG2     ; of HEADNG1 and HEADNG2

A20:
                MOV     AL,[SI]         ;Get character from HEADNG1,
                MOV     [DI],AL         ; move it to HEADNG2
                INC     SI              ;Incr next char in HEADNG1
                INC     DI              ;Incr next pos'n in HEADNG2
                DEC     CX              ;Decrement count for loop
                JNZ     A20             ;Count not zero? Yes, loop
                ;Finished
                MOV     AH,09H          ;Request display
                LEA     DX,HEADNG2     ; of HEADNG2
                INT     21H

```

```
                MOV     AX,4C00H      ;End processing
                INT     21H
A10MAIN ENDP
                END     A10MAIN
```

## Interrupt Instruction

### INT Instruction

- Software interrupt instruction to trap to the operating system to perform system-related operations
- `int imm`
- Requires a code to be set in register AX
- Will use registers and therefore, you need to save registers onto the stack

## Addressing Modes

Addressing modes define the ways

- to get the data for the operands
- put the data into the destination

There are three sources for operands

- from registers — register mode
- from the instruction — immediate addressing mode
- from the memory — memory modes
  - direct
  - direct-offset
  - register-indirect
  - base displacement
  - base-index
  - base-index with displacement

### Register Addressing

- format: register name
- allowable registers: any register
- operands: the data in the named register
- example:

```
add ax, bx
```

### Immediate Addressing

- format: data of decimal, hexadecimal or binary systems
- example:

```
add ax, 124
```

```
add ax, 3DH
```



## Memory Addressing

- effective address: the address of the location where the operand is.
- the particular mode determines how to calculate the effective address.

## Direct Mode

- format: name
  - the name the data variable in the data section
- EA: the address of the variable plus the contents of DS — — DS:address
- example:

```
var1  DW  125
var2  DW  23H
....
      add ax, var1
      add ax, var2
```

### Direct-Offset Mode

- format: name[offset] or name+offset
- EA: the address of the variable plus the contents of DS — —  
DS:(address+offset)
- example:

```
array1  DW  10 DUP(?)  
.....  
    add ax, array1[0]  
    add ax, array1+6
```

### (Register) Indirect Mode

- format: [register]
- allowable registers: BX, DI, SI, BP
- EA: DS:BX, DS:DI, DS:SI, SS:BP
- example:

```
var1  DW  125
var2  DW  23H
....
      lea bx, var1
      mov [bx], var2
```

### Base Displacement Addressing

- format: [register+offset], offset[register]
- allowable registers: BX, DI, SI, BP
- EA: DS:(BX+offset), DS:(DI+offset), DS:(SI+offset), SS:(BP+offset)
- example:

```
var1 DW 125
array1 DW 10 DUP(?)
....
    lea bx, array1
    mov [bx+2], var1
    mov [bx+3], 2[bx]
```

### Base-Index Addressing

- format: [base-reg+index-reg]
- base registers: BX or BP
- index register: DI, SI
- EA: DS:(base-reg+index-reg)
- example:

```
mov [bx+di], [bx+si]
```

### Base-Index with Displacement Addressing

- format: [base-reg+index-reg+offset], offset[base-reg+index-reg]
- base registers: BX or BP
- index register: DI, SI
- EA: DS:(base-reg+index-reg+offset)
- example:

```
mov [bx+di+2], 3[bx+si]
```