CMPE 310
Systems Design and Programming

L20: Chapter 4 – *Data Movement Instructions*

UMBC
AN HONORS UNIVERSITY IN MARYLAND

Review: Addressing modes

* Addressing modes: how data is specified
  * Register: MOV AX, BX
  * Immediate: MOV AX, 1234H
  * Memory operands: Address of form SBA:EA
    * Direct: MOV AX, [1010H]
      * EA = DA =1010H
    * Register indirect: MOV AX, [SI]
      * EA = contents of SI
    * Based: MOV AX, [BX + 1234H]
      * EA = (contents of BX) + 1234H
    * Indexed: MOV AX, [SI + 1234H]
      * EA = (contents of SI) + 1234H
    * Based-indexed: MOV AX, [BX + SI + 1234H]
      * EA = (contents of BX) + (contents of SI) + 1234H

L20 Objectives

* Data transfer instructions
  * Gain familiarity with different type of data transfer instruction
    * MOV, XCHG, LEA, and Load full pointer
    * Flags are unaffected

Instruction types

* 8086 instruction types
  * **Data Transfer instructions**
  * Input/output instructions
  * Arithmetic instructions
  * Logic instructions
  * String Instructions
  * Control transfer instructions
  * Processor control

Move Instruction

Mnemonic	Meaning	Format	Operation	Flags affected
MOV	Move	MOV D, S	(S) → (D)	None

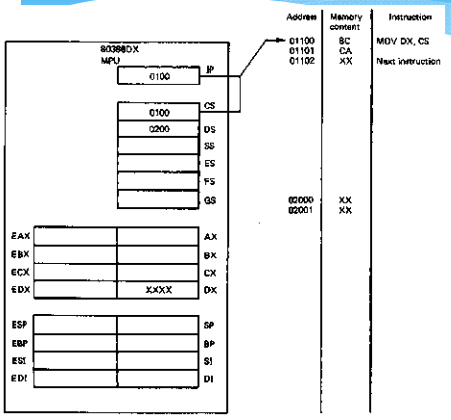
(a)

Destination	Source
Memory	Accumulator
Accumulator	Memory
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Seg-reg	Reg16
Seg-reg	Mem16
Reg16	Seg-reg
Mem16	Seg-reg
Register	Spec-reg
Spec-reg	Register

(b)

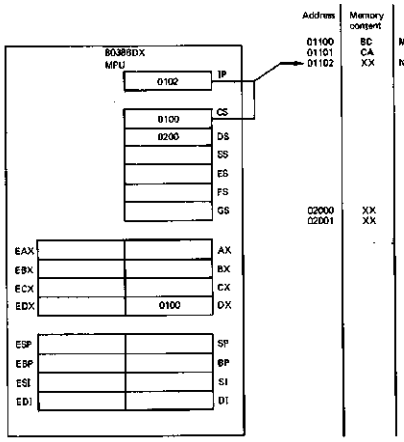
- Used to move (copy) data between:
  - Registers
  - Register and memory
  - Immediate operand to a register or memory
- General format: MOV D,S
- Operation: Copies the content of the source to the destination
  (S) → (D)
  - Source contents unchanged
  - Flags unaffected
- Allowed operands
  - Register
  - Memory
  - Accumulator (AH,AL,AX,EAX)
  - Immediate operand (S only)
  - Segment register (Seg-reg)
- Example:
- MOV [SUM],AX
- (AL) → (address SUM)
- (AH) → (address SUM+1)

Warming Up Example



- Example
  MOV DX,CS

Warming Up Example



* Example (continued)
* State after execution

Sign-Extend and Zero-Extend Move Instruction

Mnemonic	Meaning	Format	Operation	Flags affected
MOVSX	Move with sign-extend	MOVSX D, S	(S) → (D) MSBs of D are filled with sign bit of S	None
MOVZX	Move with zero-extend	MOVZX D, S	(S) → (D) MSBs of D are filled with 0	None

(a)

Destination	Source
Reg16	Reg8
Reg32	Reg8
Reg32	Reg16
Reg16	Mem8
Reg32	Mem8
Reg32	Mem16

(b)

* Sign-Extend Move instruction
  * Used to move (copy) data between two registers or memory and a register and extend the value with the value of the sign bit
  * General format:
    MOVSX D,S
  * Operation: Copies the content of the source to the destination
    (S) → (D); source contents unchanged
    * Sign bit → extended through bit 16 or 32
    * Flags unaffected
  * Examples: MOVSX EBX,AX
    Where: (AX) = FFFFH
    S = Reg16
    D = Reg32
    Sign bit (AX) = 1
    FFFFFFFFH → (EBX)
* Zero-Extend Move instruction—MOVZX
  * Operates the same as MOVSX except extends with zeros
  * Example: MOVZX CX, Byte Pointer [DATA_BYTE]
  * Where: (DATA_BYTE) = FFH,S = Mem8,D = Reg16
    00FFH → (CX)

Exchange Instruction

Mnemonic	Meaning	Format	Operation	Flags affected
XCHG	Exchange	XCHG D, S	(D) ↔ (S)	None

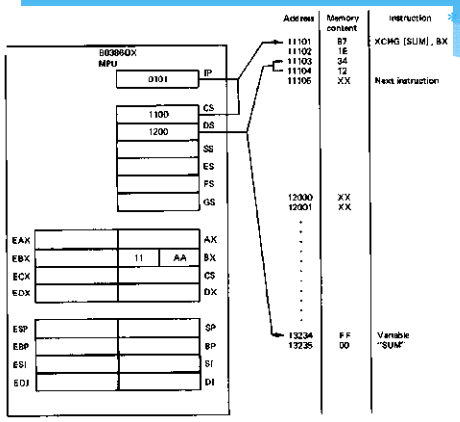
(a)

Destination	Source
Accumulator	Reg16
Accumulator	Reg32
Memory	Register
Register	Register

(b)

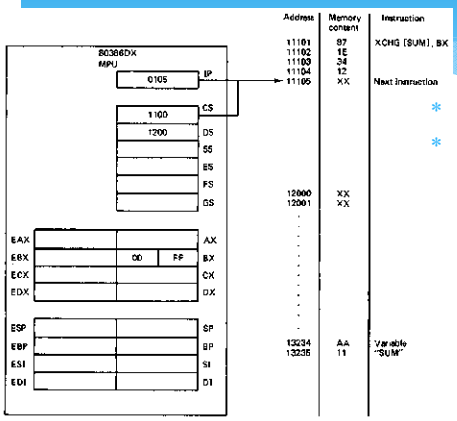
- Used to exchange the data between two data registers or a data register and memory
- General format:
  XCHG D,S
- Operation: Swaps the content of the source and destination
  - Both source and destination change
    (S) → (D)
    (D) → (S)
  - Flags unaffected
- Special accumulator destination version executes faster
- Examples: XCHG AX,DX
  (Original value in AX) → (DX)
  (Original value in DX) → (AX)

Example of Exchange Instruction



- Example
  XCHG [SUM],BX

Example of Exchange Instruction



- Example (continued)
- State after execution

Review: Data transfer instructions

- MOV: basic data transfer
  - Can use registers, memory, immediate
  - If segment reg. is destination, source must be register
- MOVSX/MOVZX
  - Sign-extend or zero-extend register/memory value
  - Moving byte from memory: BYTE POINTER
- XCHG
  - Exchange contents of source, dest

Load Effective Address

Mnemonic	Meaning	Format	Operation	Flags affected
LEA	Load effective address	LEA Reg16, EA LEA Reg32, EA	(EA) → (Reg16) (EA) → (Reg32)	None None
LDS	Load register and DS	LDS Reg16, EA	(EA) → (Reg16) (EA + 2) → (DS)	None
		LDS Reg32, EA	(EA) → (Reg32) (EA + 4) → (DS)	None
LSS	Load register and SS	LSS Reg16, EA	(EA) → (Reg16) (EA + 2) → (SS)	None
		LSS Reg32, EA	(EA) → (Reg32) (EA + 4) → (SS)	None
LES	Load register and ES	LES Reg16, EA	(EA) → (Reg16) (EA + 2) → (ES)	None
		LES Reg32, EA	(EA) → (Reg32) (EA + 4) → (DS)	None
LFS	Load register and FS	LFS Reg16, EA	(EA) → (Reg16) (EA + 2) → (FS)	None
		LFS Reg32, EA	(EA) → (Reg32) (EA + 4) → (FS)	None
LGS	Load register and GS	LGS Reg16, EA	(EA) → (Reg16) (EA + 2) → (GS)	None
		LGS Reg32, EA	(EA) → (Reg32) (EA + 4) → (GS)	None

- Load effective address instruction
  - Used to load the effective address of memory operand into a register
  - General format:

    LEA Reg16/32,EA
  - Operation:

    EA → (Reg16/32)
    - Source unaffected:
    - Flags unaffected

Load Full Pointer Instructions

Mnemonic	Meaning	Format	Operation	Flags affected
LEA	Load effective address	LEA Reg16, EA LEA Reg32, EA	(EA) → (Reg16) (EA) → (Reg32)	None None
LDS	Load register and DS	LDS Reg16, EA	(EA) → (Reg16) (EA + 2) → (DS)	None
		LDS Reg32, EA	(EA) → (Reg32) (EA + 4) → (DS)	None
LSS	Load register and SS	LSS Reg16, EA	(EA) → (Reg16) (EA + 2) → (SS)	None
		LSS Reg32, EA	(EA) → (Reg32) (EA + 4) → (SS)	None
LES	Load register and ES	LES Reg16, EA	(EA) → (Reg16) (EA + 2) → (ES)	None
		LES Reg32, EA	(EA) → (Reg32) (EA + 4) → (DS)	None
LFS	Load register and FS	LFS Reg16, EA	(EA) → (Reg16) (EA + 2) → (FS)	None
		LFS Reg32, EA	(EA) → (Reg32) (EA + 4) → (FS)	None
LGS	Load register and GS	LGS Reg16, EA	(EA) → (Reg16) (EA + 2) → (GS)	None
		LGS Reg32, EA	(EA) → (Reg32) (EA + 4) → (GS)	None

- Used to load a full address pointer from memory into a segment register and register
- General formats and operation for LDS and LSS

  LDS Reg16/32,EA

  (EA) → (Reg16/32)

  (EA+2/4) →(DS)

  LSS Reg16/32,EA

  (EA) → (Reg16/32)

  (EA+2/4) →(SS)
- LES, LFS, and LGS operate the same

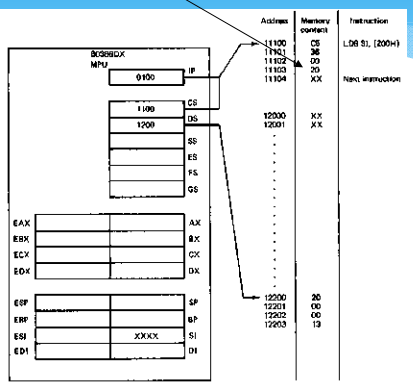
LES Reg16/32,EA (EA) →(Reg16/32),(ES)

LFS Reg16/32,EA (EA) → (Reg16/32),(FS)

LGS Reg16/32,EA (EA) → (Reg16/32),(GS)

Load Full Pointer Instructions (example)

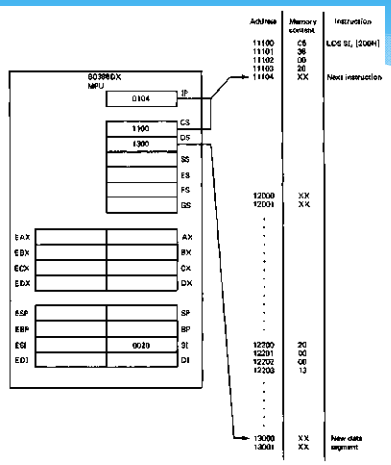
Error 02 not 20



* Example

  LDS SI,[200H]

Load Full Pointer Instructions (example)



* Example (continued)
* State after execution

Example
Usage of Move Instruction

```
MOV  AX,2000H
MOV  DS, AX
MOV  ES, AX
MOV  AX,3000H
MOV  SS,AX
MOV  AX,0H
MOV  BX,AX
MOV  CX,0AH
MOV  DX,100H
MOV  SI,200H
MOV  DI,300H
```

* Example—Initialization of internal registers with immediate data and address information
  * What is the final state of all affected registers?
  * Why is AX used to initialize segment registers?

Usage of Move Instruction

```
MOV  AX,2000H
MOV  DS, AX
MOV  ES, AX
MOV  AX,3000H
MOV  SS,AX
MOV  AX,0H
MOV  BX,AX
MOV  CX,0AH
MOV  DX,100H
MOV  SI,200H
MOV  DI,300H
```

* Example—Initialization of internal registers with immediate data and address information

Next time

* Arithmetic instructions