8086 Microprocessor Laboratory Experiments

Experiment 2: Instruction Format and The MOVE Instruction

Murtadha Hssayeni, Ph.D.

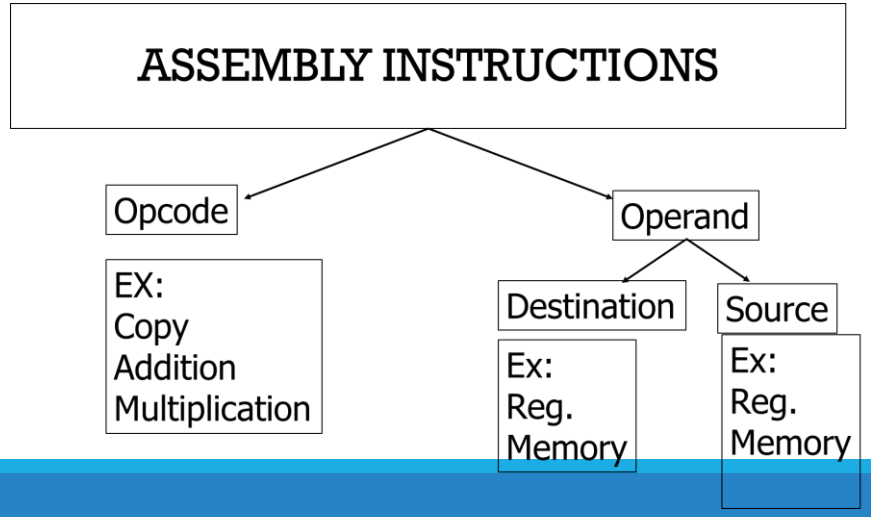
m.hssayeni@uobabylon.edu.iq

Outlines

- Instructions in Assembly language
- MOV Instruction
- How Are Instructions Saved in Memory?
- Logical address and Physical Address
- Procedure and Discussion

Instructions in Assembly language

- The sequence of commands used to tell a microcomputer what to do is called a program.
- An instruction can be divided into two parts:
  - Its operation code (**opcode**) and its (**operands**).
  - The **opcode** is the part of the instruction that identifies the operation to be performed (commands to the CPU).
  - **Operands** describe the data that are to be processed as the microprocessor carries out the operation specified by the opcode.



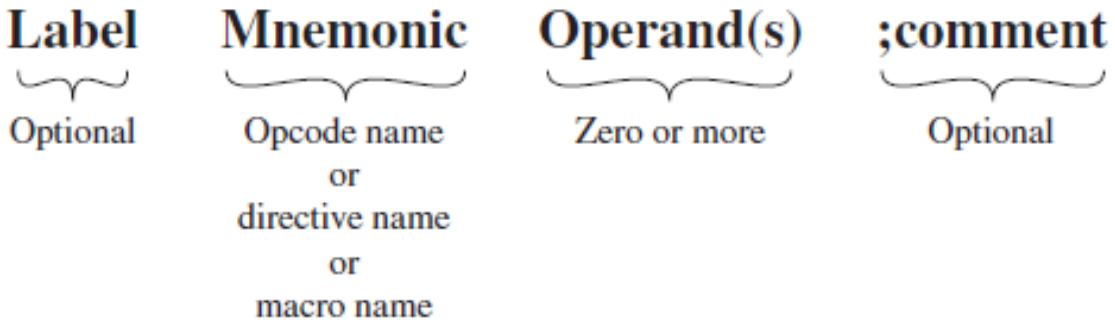


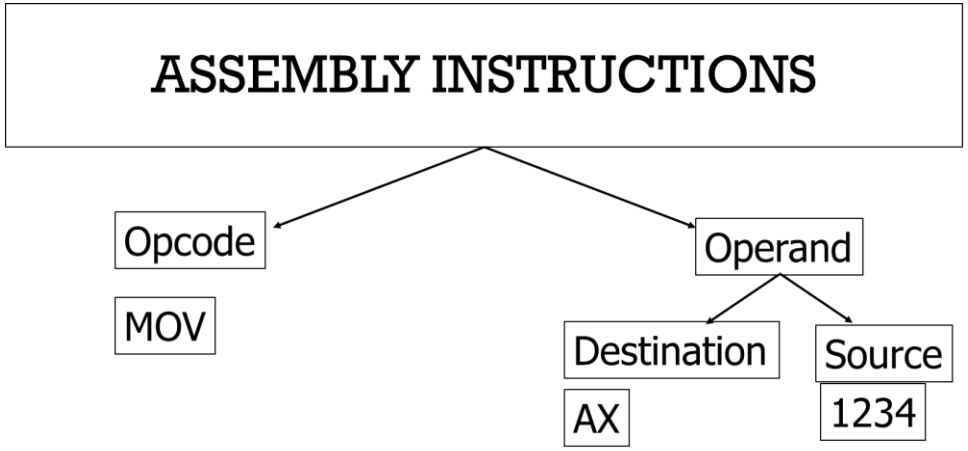
Figure B.1 Assembly-Language Statement Structure

MOV Instruction

- The MOV instruction copies the second operand (Source) to the first operand (Destination)
- It has the following format:

  MOV destination, source

- Example*: MOV AX, 1234H*



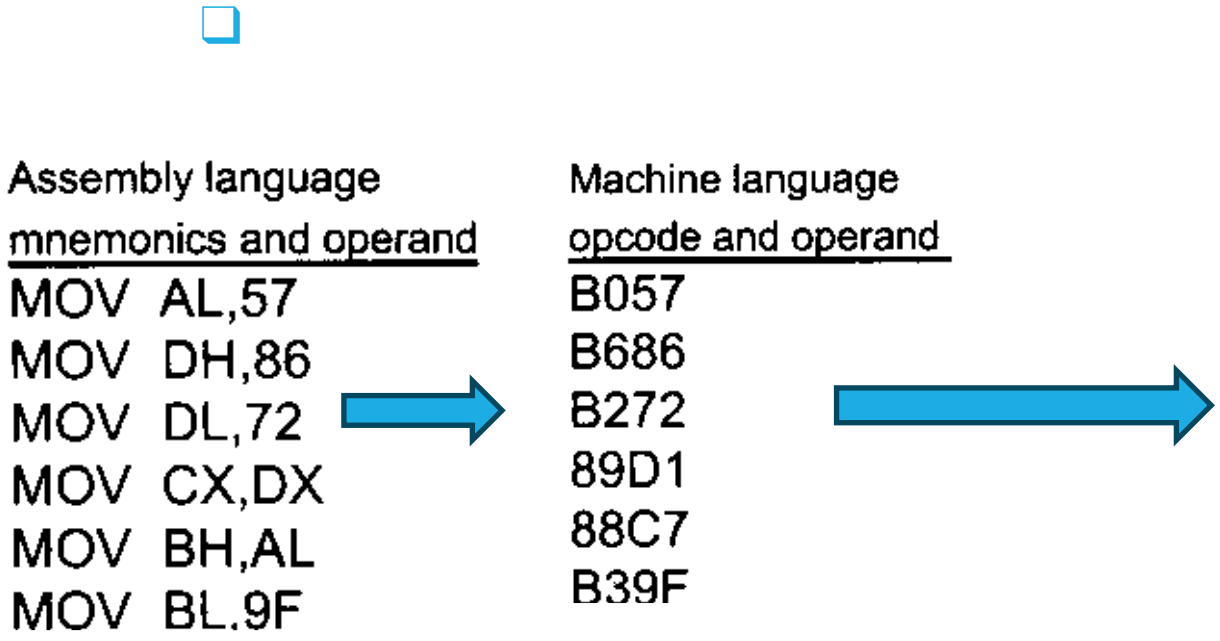
MOV Instruction

- Both **operands must be the same size**, which can be a byte or a word. These types of operand are supported:
  - MOV REG, memory
  - MOV memory, REG
  - MOV SREG, REG
  - MOV memory, immediate
  - MOV REG, immediate
    - REG: AX, BX, CX, DX, AH, AL, BH, BL, CH, CL, DH, DL, DI, SI, BP, SP.
    - Memory: [BX], [SI]+BX, [DI]+BX, etc.
    - SREG: DS, ES, SS, and only the second operand: CS.
    - Immediate: 5, -24, 3FH, 1000101B, etc.

MOV Instruction

- The **MOV** instruction **cannot**:
  - Set the value of the CS and IP registers.
  - **Copy value of one segment register to another segment register (should copy to general register first).**
  - Copy immediate value to segment register (should copy to general register first).
- Examples on MOV instruction:
  - *MOV AX, 58FCH ; move 58FCH into AX (LEGAL)*
  - *MOV CS, 3F47H ; move 3F47H into CS (**ILLEGAL**)*
  - *MOV DH, 6678H ; move 6678H into DH (**ILLEGAL**)*
  - *MOV BL, 99H ; move 99H into BL (LEGAL)*
  - *MOV DS, CS ; (**ILEGAL**)*

How Are Instructions Saved in Memory?



Memory

Logical address	Physical address	Machine code contents
1132:0100	11420	B0
1132:0101	11421	57
1132:0102	11422	B6
1132:0103	11423	86
1132:0104	11424	B2
1132:0105	11425	72
1132:0106	11426	89
1132:0107	11427	D1
1132:0108	11428	88
1132:0109	11429	C7
1132:010A	1142A	B3

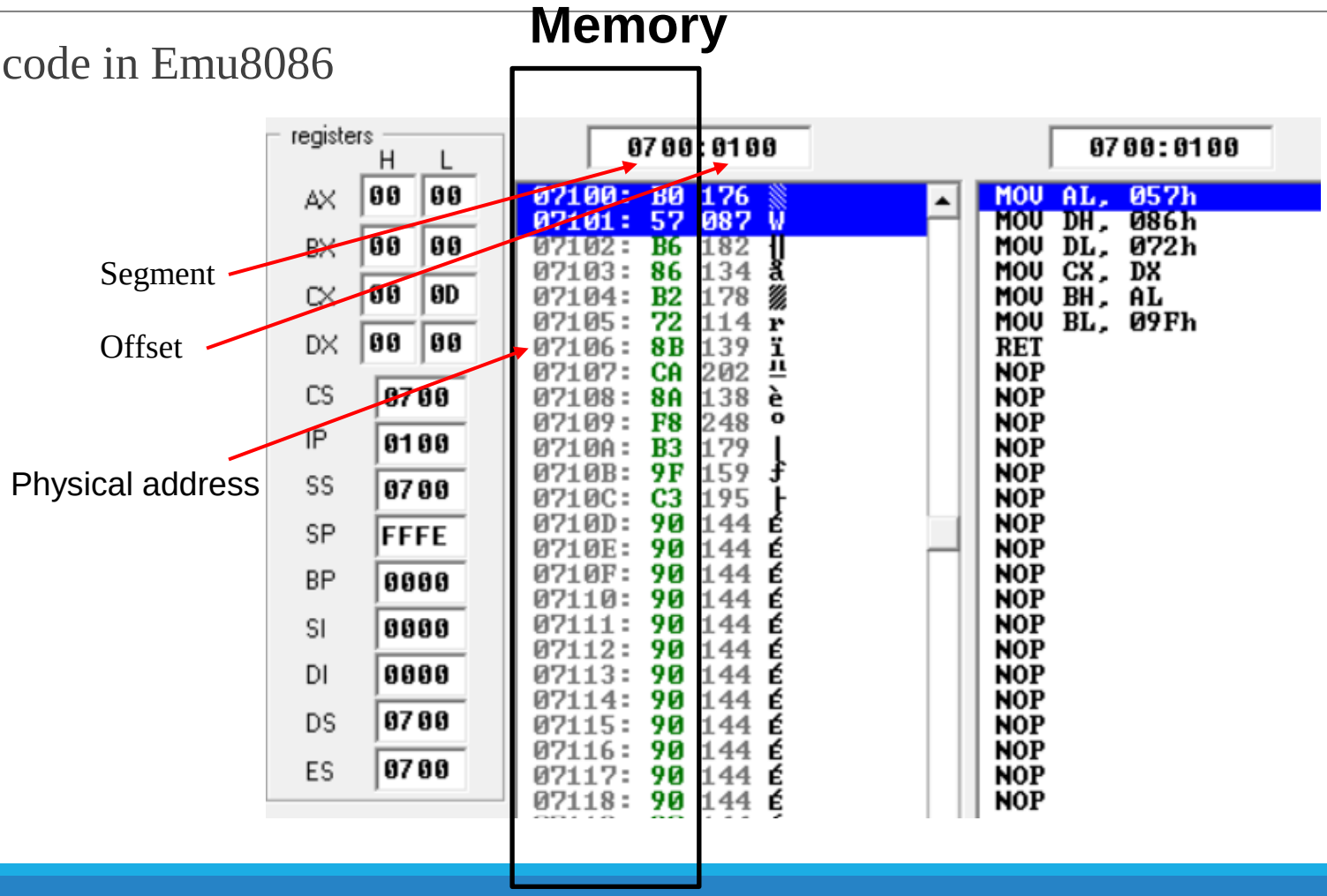
Example

- Executing the previous code in Emu8086

```
org 100h

MOV AL, 57H
MOV DH, 86H
MOV DL, 72H
MOV CX, DX
MOV BH, AL
MOV BL, 9FH

ret
```



Logical address and Physical Address

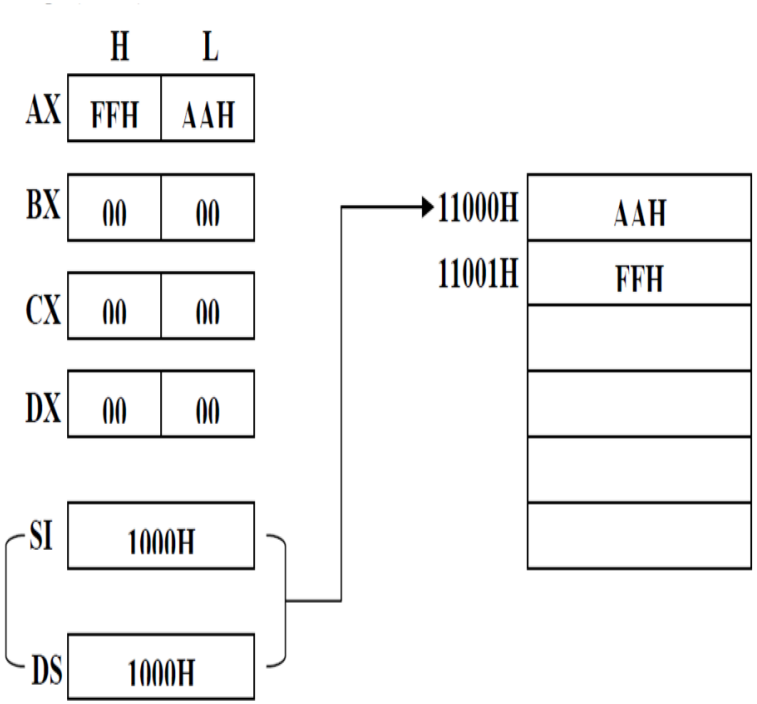
- There are two main types of addresses
  - The **physical address**, and the **logical address**.

1. The **physical address** is the 20-bit address
   - A range of 00000H to FFFFFH for the 8086 microprocessor
   - Its an actual physical location in RAM or ROM within the 1 megabyte memory range.
2. The **logical address** consists of a segment value and an offset address in the form segment:offset.
   - The **offset address** is 16-bit address which is a location within a 64K-byte segment range.
     - A range from 0000H to FFFFH.

- **Logical address -> Physical address**

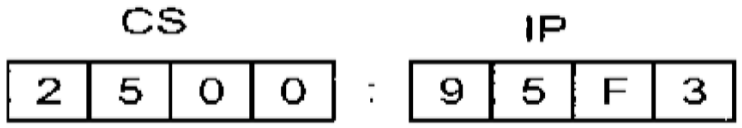
  Physical address = Segment *10h + offset

- Now, why do you think there are different address types?



Logical address and Physical Address: Accessing instructions

- **To execute a program**, the 8086 fetches the instructions (opcodes and operands) from the code segment.
  - **The CS and IP are used to define the logical address**
- The physical address for the location of the instruction is generated by shifting the **CS** left one hex digit and then adding it to the **IP**.
- To clarify this important concept,
  - Assume values in CS and IP as shown



  - The offset address is contained in IP; in this case 95F3H.
  - The logical address is 2500:95F3H.
  - The physical address will be 2500 * 10h + 95F3 = 2E5F3H.
  - The microprocessor will retrieve the instruction from memory locations starting at 2E5F3

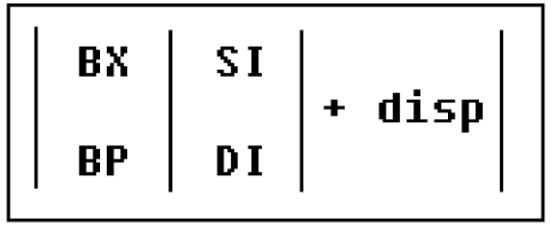
Logical address and Physical Address: Accessing Data

- **To access memory value:**
  - **segment registers** work together **with general purpose register**.
- **Valid segment registers to define segment:**
  - By default, **DS** segment register is used for all modes except those with BP register, for these **SS** segment register is used.
- **Valid general-purpose register to define offset: BX, SI, DI, and BP**.
  - combining these registers inside [ ] symbols, we can get different memory locations.
  - All those possible combinations are summarized in this chart:

BX	SI	+ disp
BP	DI	

- Valid combinations:
  - Taking only one item from each column or skipping the column
  - BX and BP never go together.
  - SI and DI also don't go together.

Logical Address and Physical Address: Accessing Data



- **Displacement** can be an immediate value or offset of a variable, or even both.
  - It can be inside or outside of the [ ] symbols
  - It is a signed value, so it can be both positive and negative.
- Example:
  - if we would like to access memory at the physical address 12345h (hexadecimal), we may set the DS = 1230h and SI = 0045h.

Logical address and Physical Address: Summary

- **To access a program in memory:**
  - The **CS** and **IP** are used to define the logical address

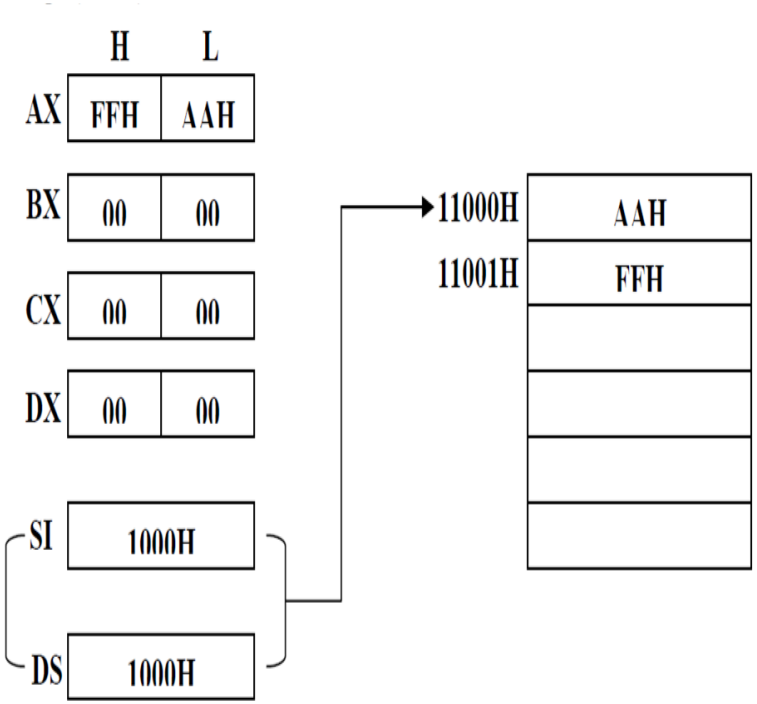
- **To access memory data:**
  - Valid segment registers to define segment: By default, **DS** segment register is used for all modes except those with BP register, for these **SS** segment register is used.
  - Valid general-purpose register to define offset: **BX, SI, DI, and BP**.

- The 8086 microprocessor provides various **addressing modes (next lecture).**

Logical address and Physical Address: Example

- We want to store the data in AX register at the memory location (11000H):
  - We must set the data segment and the offset of the above location as:
    - Offset = 1000H
    - Segment DS = (11000 - 1000) /10H = 1000H
  - Assume the data in the AX register is (FFAA H)
  - The AH=FFH and the AL=AAH

```
org 100h
MOV AX, 1000 H
MOV DS, AX
MOV AX, 0FFAA H
MOV SI, 1000 H
MOV [SI], AX
ret
```



- After we execute the program:
  - The memory location (11000H) and AH in (11001H) according to the rule:
    - lower byte = lower address

Procedure

1. Write down the following program:

MOV AX, 0B820H
MOV DS, AX
MOV CL, 'H'
MOV CH, 11110000B
MOV BX, 14CH
MOV [BX], CX

2. Find the offset (BX), and segment (DS), then calculate the physical address of the above program.

3. Locate the calculated physical memory in Emu8086

   - by executing the program and write down the results of the output register and the memory address that the program writes to.

Discussion

1. Write a program to store (10101010B) to the memory location 47511H.
2. Run the following program:

```
MOV AX, 3000H
MOV DS, AX
MOV AX, 4000H
MOV SS, AX
MOV BX, 0600H
MOV BP, 0700H
MOV DX, 0A6E2H
MOV SI, 0040H
MOV DI, 00E0H
MOV [BX]+SI, AH
MOV [BX]+DI, DH
MOV [BP]+DI, DL
MOV [BX]-20H, DH
MOV [DI]+4H, DL
MOV [BP]+100H, AL
```

3. Give the physical address of all accessed memory addresses for the previous program.
4. You have DS=4000H, AH=7FH, AL=44H, SI=1000, what is the physical address of the instruction

   MOV [SI]+10H, AX

5. And how the data is stored in the memory?