Lecture 4

SINGLE PASS ASSEMBLER FOR IBM PC - 1/7

(Architecture of Intel 8088)

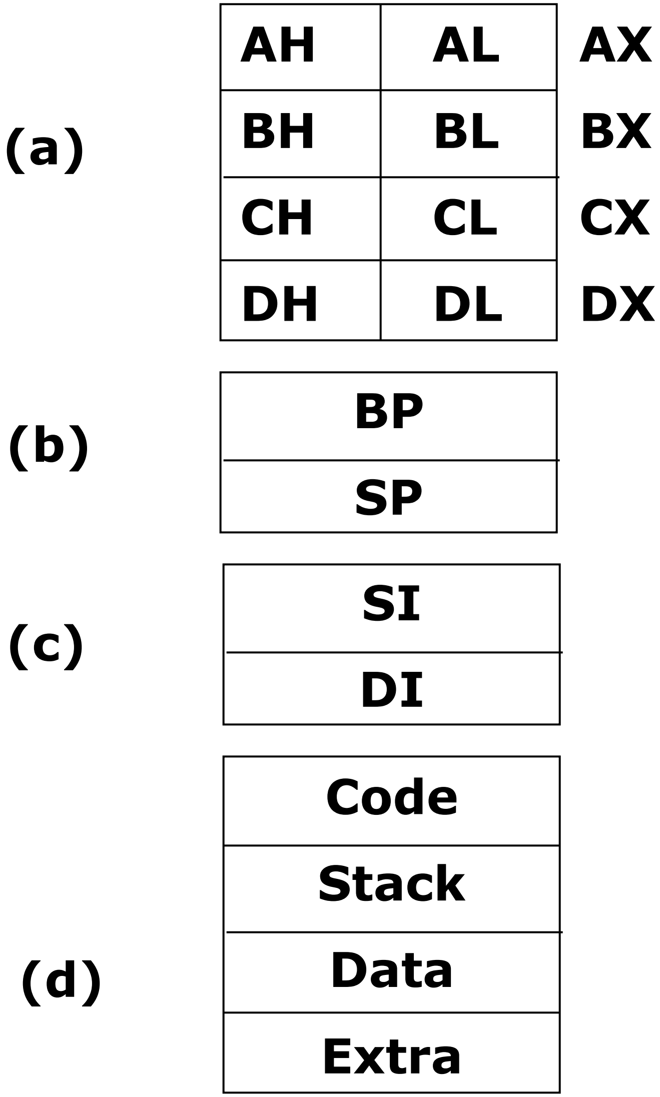
- Supports 8 and 16 bit arithmetic

- CPU contains following features
  - Data registers AX, BX, CX and DX
  - Index registers SI and DI
  - Stack pointer registers BP and SP
  - Segment registers Code, Stack, Data and Extra

SINGLE PASS ASSEMBLER FOR IBM PC - 2/7

(Architecture of Intel 8088)

Fig:
(a)Data
(b)Base
(c) Index
(d) Segment



(a)

AH	AL	AX
BH	BL	BX
CH	CL	CX
DH	DL	DX

(b)

BP
SP

(c)

SI
DI

(d)

Code
Stack
Data
Extra

SINGLE PASS ASSEMBLER FOR IBM PC - 3/7

(Architecture of Intel 8088)

- Each data register is 16 bits in size - split into upper and lower halves
- Either half can be used for 8 bit arithmetic, while two halves together constitute data register for 16 bit arithmetic
- SI and DI are used to index source and destination addresses in string manipulation instructions

SINGLE PASS ASSEMBLER FOR IBM PC - 4/7

(Architecture of Intel 8088)

- Two stack pointer registers called SP and BP address the stack
- SP points into stack implicitly used by architecture to store subroutine and interrupt return addresses
- BP used by programmer – Push and Pop instructions are provided

SINGLE PASS ASSEMBLER FOR IBM PC - 5/7

(Architecture of Intel 8088)

- Memory used to store 3 components of a program
  - Program code
  - Stack
  - Data

- Code, Stack and Data registers contain the start addresses of the three above

- Extra segment register points to another memory which can be used to store data

SINGLE PASS ASSEMBLER FOR IBM PC - 6/7

(Architecture of Intel 8088)

- Addressing modes tells where and how to locate the data to be accessed

- There are 7 addressing modes divided into 3 categories in 8086
  - Register operand addressing
    - Register addressing mode
  - Immediate operand addressing
    - Immediate addressing mode

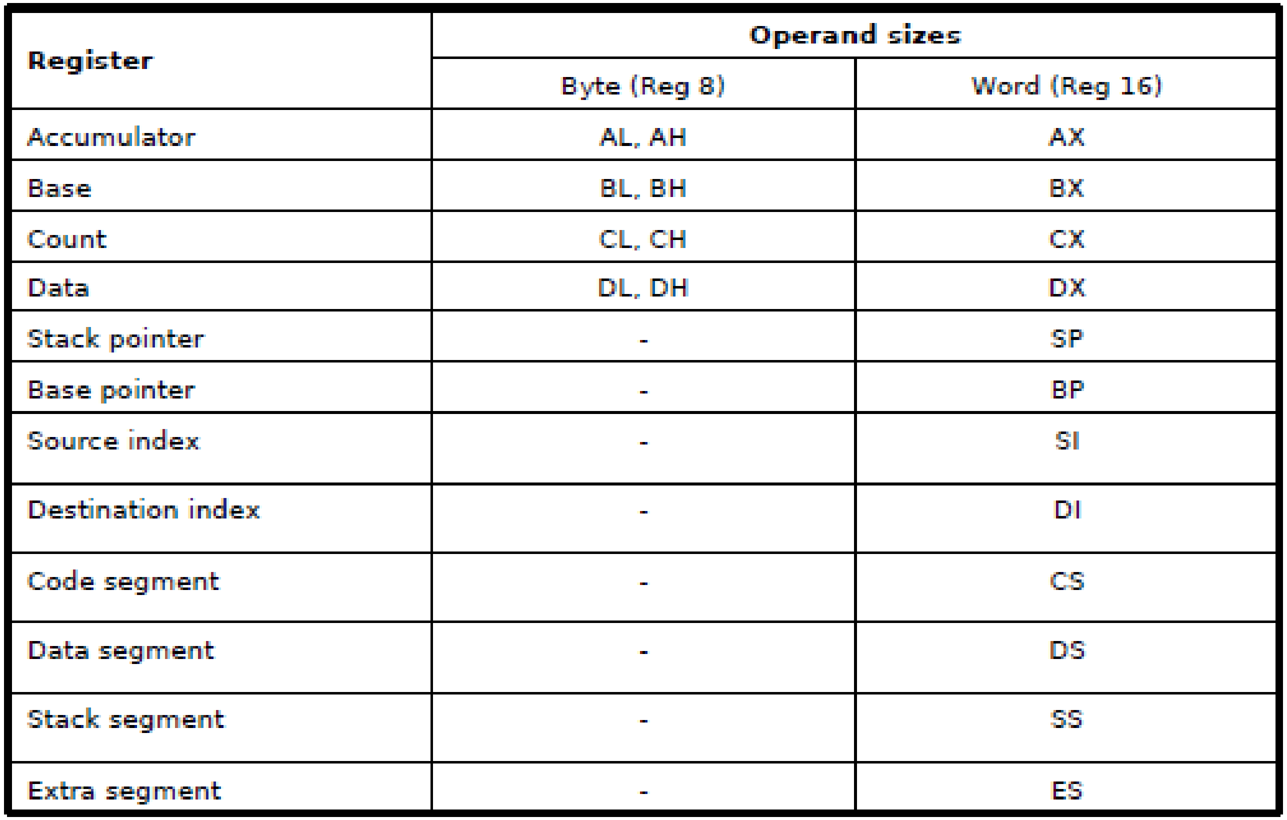
SINGLE PASS ASSEMBLER FOR IBM PC - 7/7

(Architecture of Intel 8088)

– Memory operand addressing
    - Direct addressing mode
    - Register indirect addressing mode
    - Based addressing mode
    - Indexed addressing mode
    - Based-indexed addressing mode

Register operand addressing – 1/2

- Operand to be accessed is inside internal register
- Internal registers that can be used as source or destination



Register	Operand sizes	
	Byte (Reg 8)	Word (Reg 16)
Accumulator	AL, AH	AX
Base	BL, BH	BX
Count	CL, CH	CX
Data	DL, DH	DX
Stack pointer	-	SP
Base pointer	-	BP
Source index	-	SI
Destination index	-	DI
Code segment	-	CS
Data segment	-	DS
Stack segment	-	SS
Extra segment	-	ES

Register operand addressing – 2/2

- Example

```
MOV AX, BX
MOV CH, DH
```

Immediate operand addressing – 1/3

- If an operand is part of an instruction instead of the contents of a register or memory location, it is called an immediate operand

- The operand can either be an 8-bit or 16-bit data

- Immediate operands normally represent **constant data**

Immediate operand addressing – 2/3

- This addressing mode can only be used to specify a source operand

```
MOV AL, 15H
```

- In this instruction, the source operand is 15H, while the destination operand is register AL

- Thus this instruction uses both the immediate and register addressing modes

Immediate operand addressing – 3/3

- Example : Write an instruction that will move the immediate value 1234H into the CX register

- The instruction must use immediate addressing mode for the source operand and register addressing mode for the destination operand, i.e.

```
MOV CX, 1234H
```

Memory Operand Addressing – 1/4

- The **operands are inside the memory**
- To access an operand in memory, 8086 must calculate the physical address (PA) of the memory location before it can read or write the operand
- **The physical address is computed from a segment based address (SBA) and an effective address (EA)**

  PA = SBA + EA

Memory Operand Addressing – 2/4

- **SBA identifies the starting location of the segment in memory**, and **EA represents the offset of the operand** from the beginning of this segment of memory

- The value of EA can be specified in a variety of ways

Memory Operand Addressing – 3/4

- **EA can be made up from as many as 3 elements: base, index and displacement**

PA = SBA: EA

PA = Segment base: Base + Index + Displacement

$$PA = \left\{\begin{matrix} CS \\ SS \\ DS \\ ES \end{matrix}\right\} : \left\{\begin{matrix} BX \\ BP \end{matrix}\right\} + \left\{\begin{matrix} SI \\ DI \end{matrix}\right\} + \left\{\begin{matrix} \text{8-bit displacement} \\ \text{16-bit displacement} \end{matrix}\right\}$$

Memory Operand Addressing – 4/4

- Hence, three addressing modes are defined by using various combination of these elements:
  - Register indirect addressing mode
  - Based addressing mode
  - Indexed addressing mode
  - Based-indexed addressing mode

Direct Addressing Mode – 1/5

- In this addressing mode, the **effective address (EA) of the operand is specify directly in the instruction**

- The effective address is used directly as the **16-bit offset** of the memory location of the operand from the segment base address specified by the selected segment register

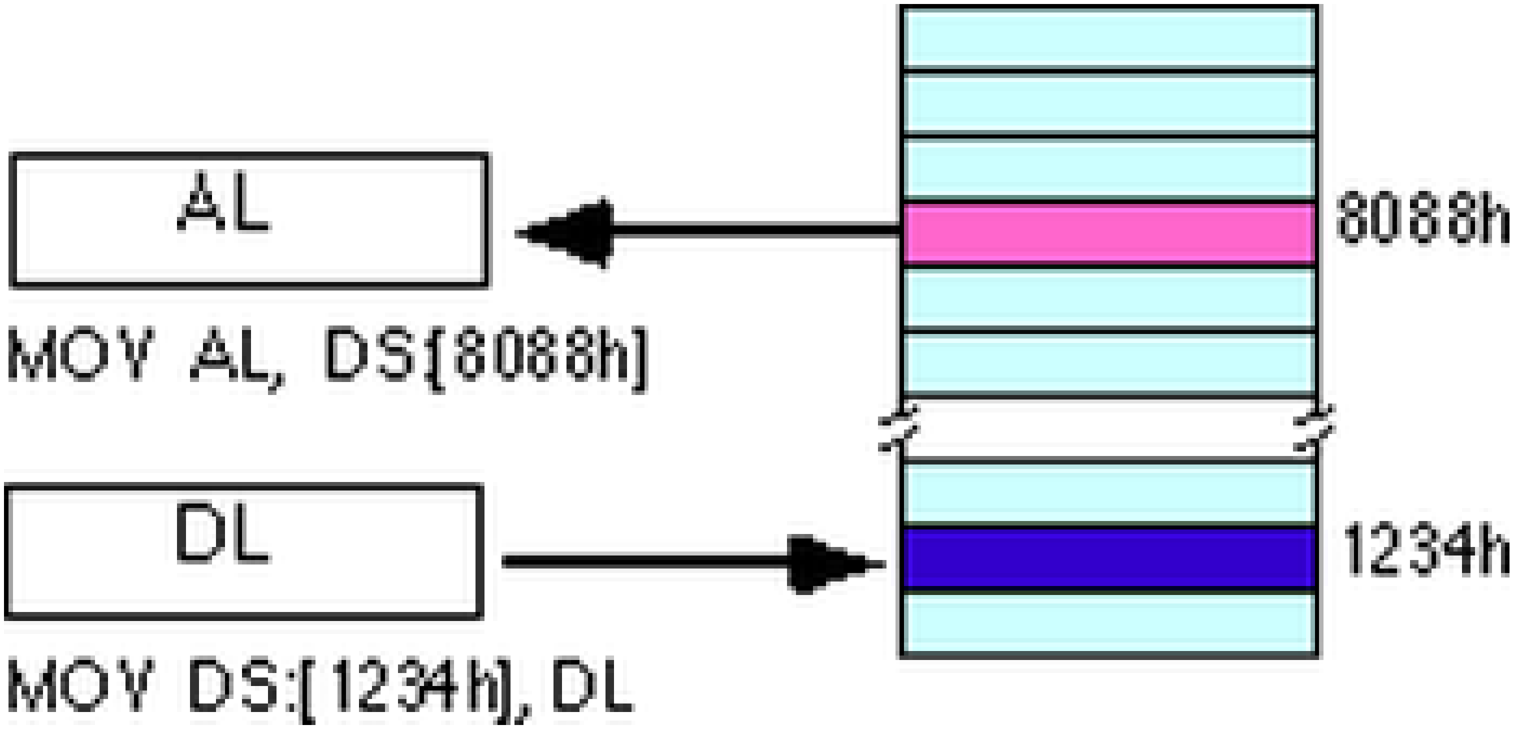
Direct Addressing Mode – 2/5

$$PA = \begin{Bmatrix} CS \\ SS \\ DS \\ ES \end{Bmatrix} : \left\{ \text{Direct address} \right\}$$

- The default segment register is **DS**

- But using a **segment over-ride prefix**, any of the four segment registers can be used

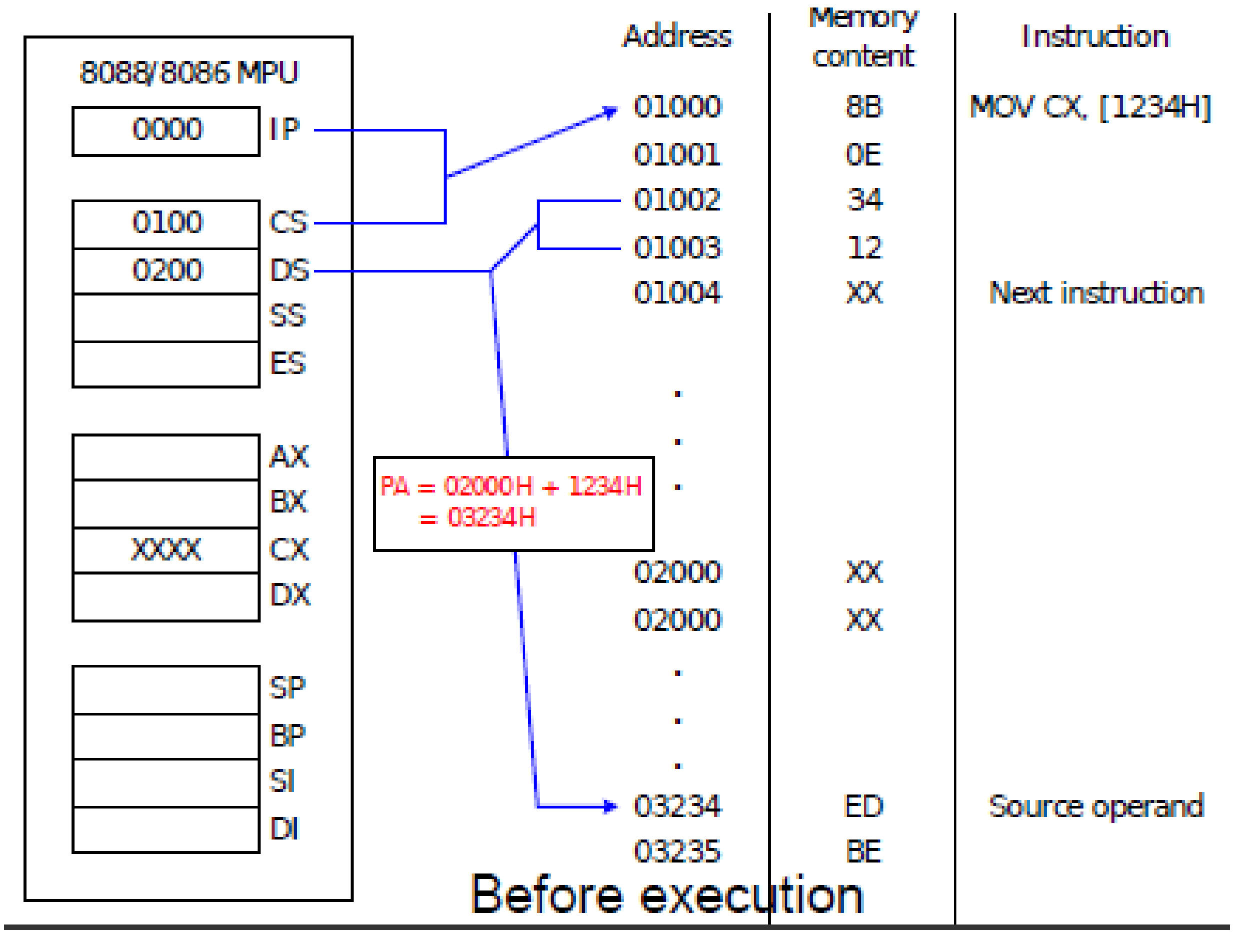
Direct Addressing Mode – 3/5



Direct Addressing Mode – 4/5

MOV CX, [1234H]

- Move the contents of the memory location with offset 1234H in the current data segment into register CX



8088/8086 MPU

0000	IP
0100	CS
0200	DS
	SS
	ES
	AX
	BX
XXXX	CX
	DX
	SP
	BP
	SI
	DI

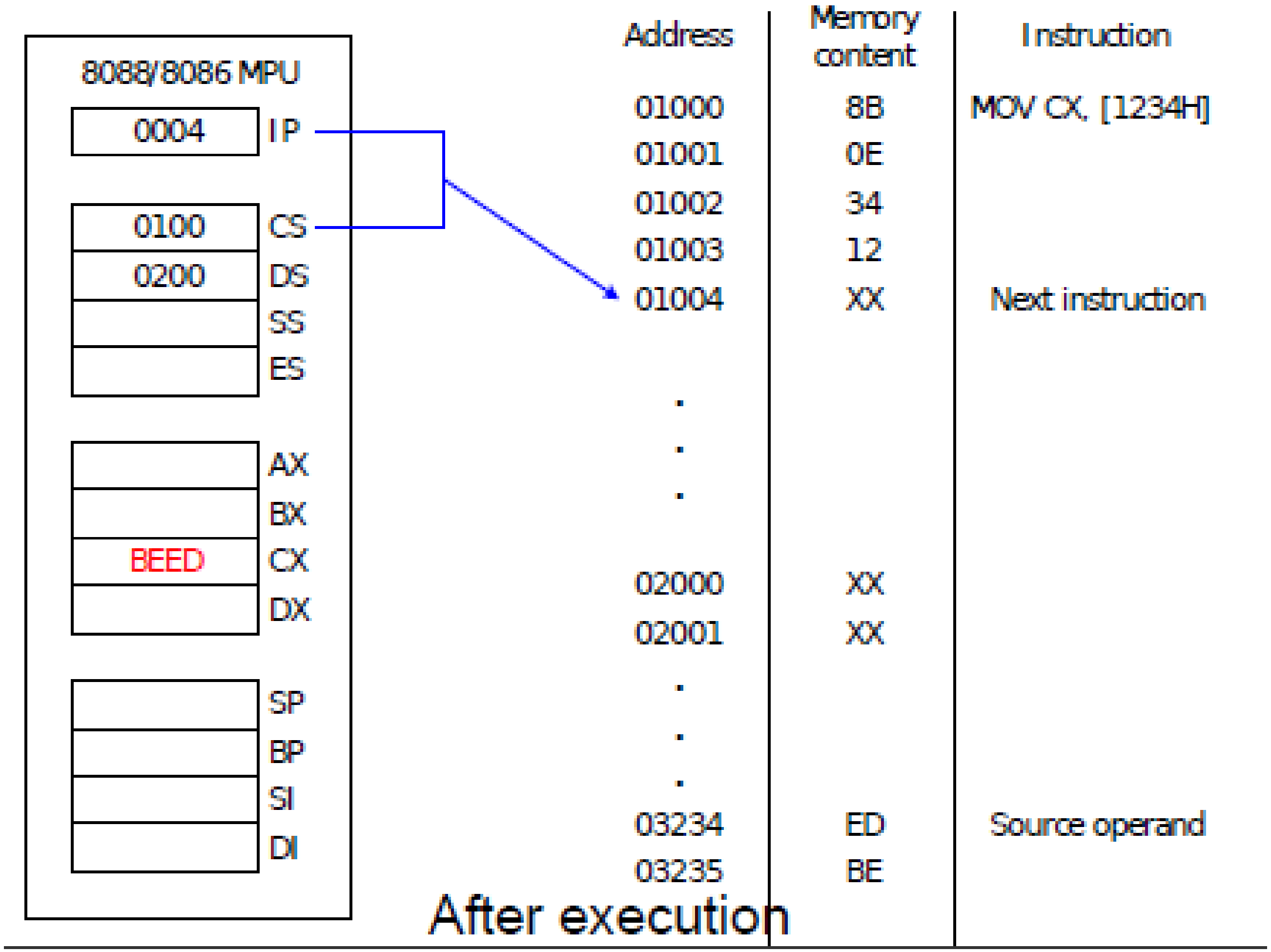
PA = 02000H + 1234H
= 03234H

Address	Memory content	Instruction
01000	8B	MOV CX, [1234H]
01001	0E	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02000	XX	
.		
.		
.		
03234	ED	Source operand
03235	BE	

Before execution

Direct Addressing Mode – 5/5

- After calculating the PA of the operand, the 8086 reads the word of data starting at that address (which is BEEDH) and loads it into the CS register



8088/8086 MPU

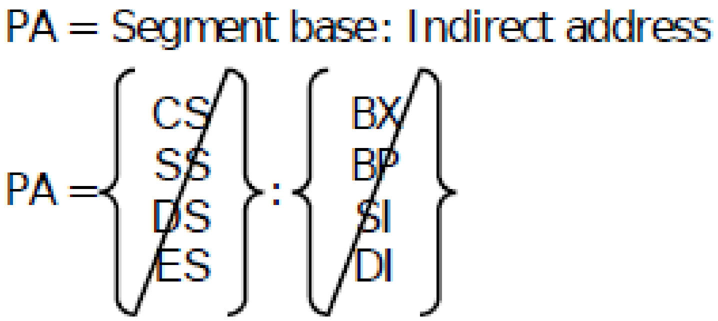
0004	IP
0100	CS
0200	DS
	SS
	ES
	AX
	BX
BEED	CX
	DX
	SP
	BP
	SI
	DI

Address	Memory content	Instruction
01000	8B	MOV CX, [1234H]
01001	0E	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
03234	ED	Source operand
03235	BE	

After execution

Register Indirect Addressing Mode – 1/4

- In this addressing mode, the **effective address is specified either in a base register (BX or BP) or index register (SI or DI)**

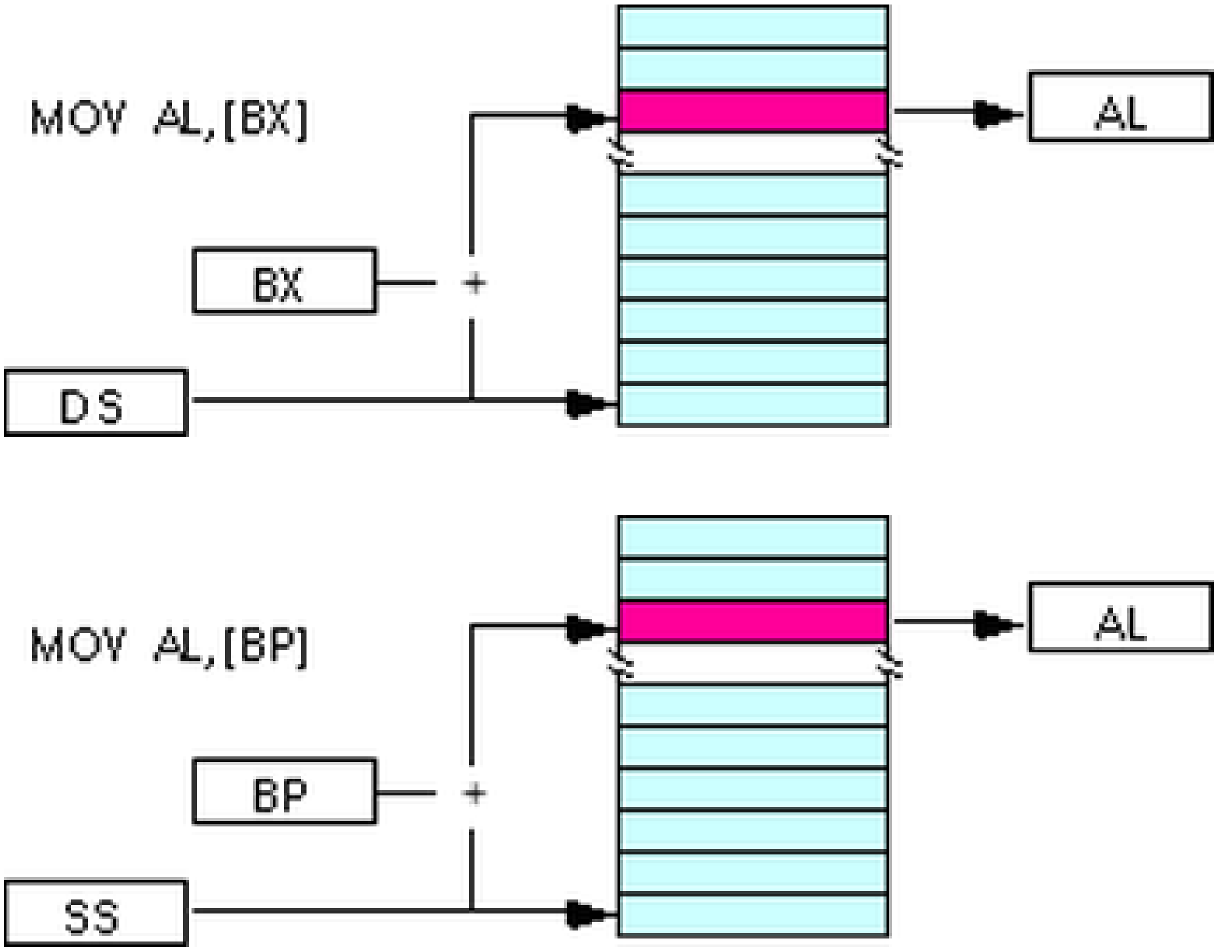


- This effective address will be combined with a segment base address in a segment register (default is DS register) to form a physical address

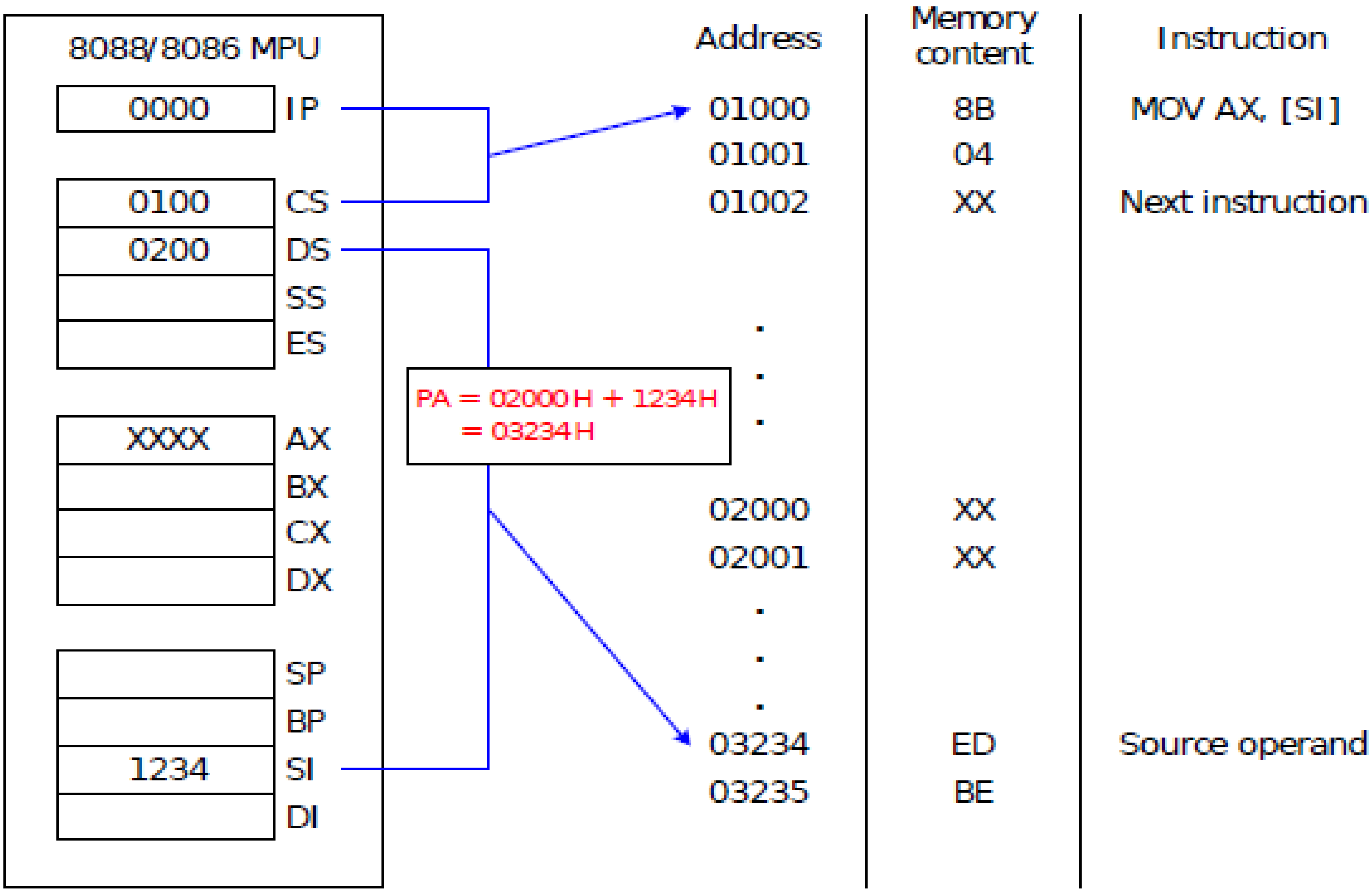
MOV AX, [SI]

- This instruction will move the contents of the memory location at the PA given by the combination of the current data segment and the effective address in register SI into AX register

Register Indirect Addressing Mode – 2/4



Register Indirect Addressing Mode – 3/4



8088/8086 MPU

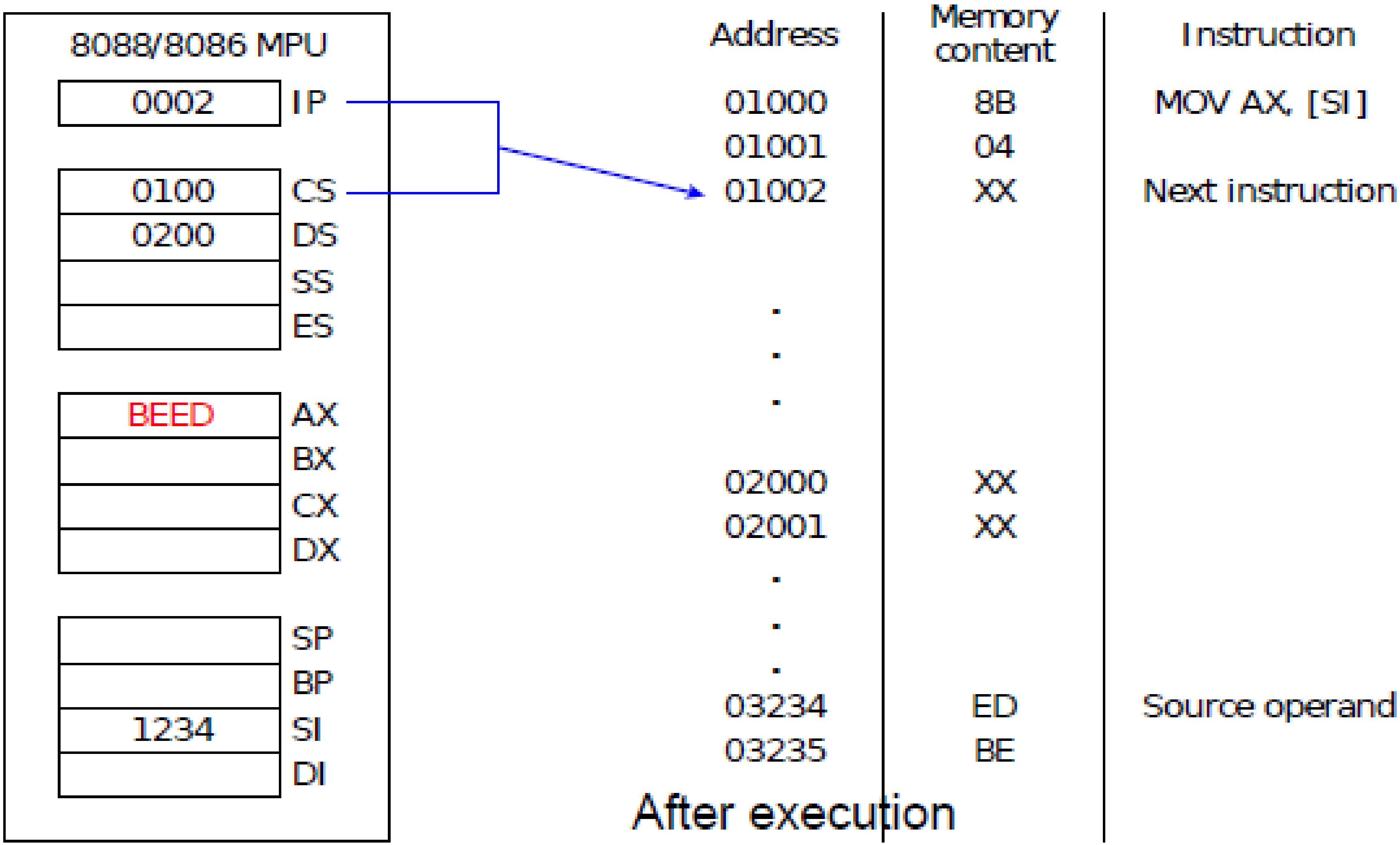
0000	IP
0100	CS
0200	DS
	SS
	ES
XXXX	AX
	BX
	CX
	DX
	SP
	BP
1234	SI
	DI

PA = 02000 H + 1234 H
= 03234 H

Address	Memory content	Instruction
01000	8B	MOV AX, [SI]
01001	04	
01002	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
03234	ED	Source operand
03235	BE	

Before execution

Register Indirect Addressing Mode – 4/4



8088/8086 MPU

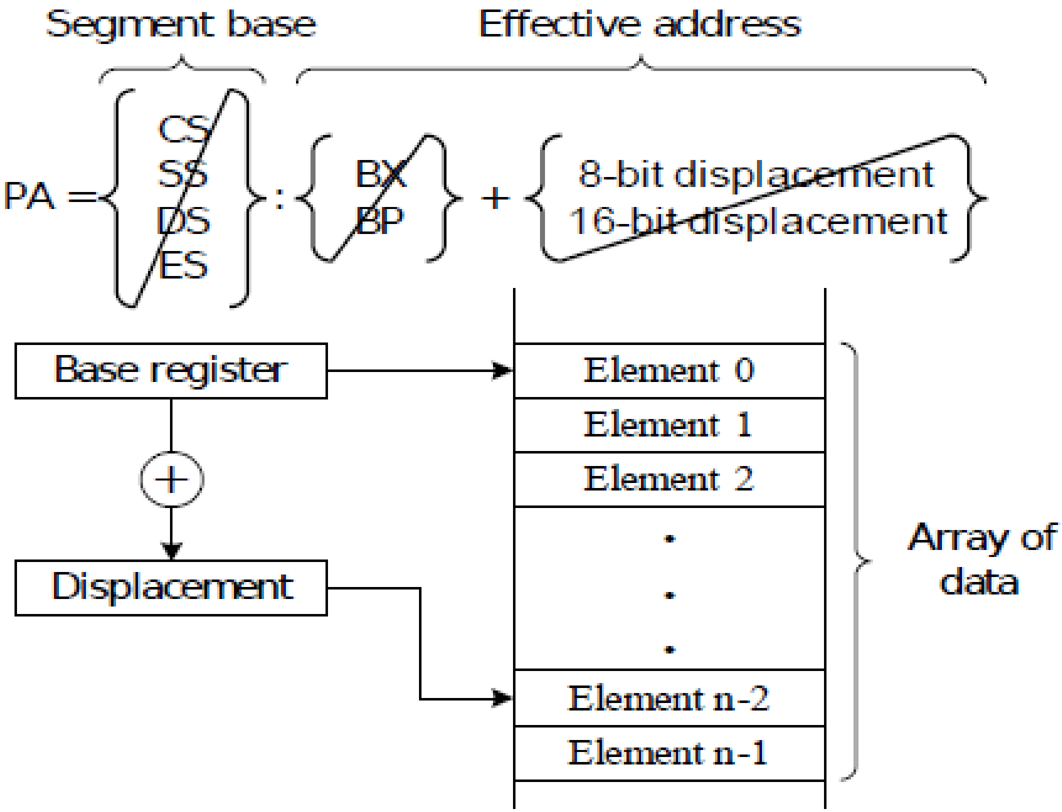
0002	IP
0100	CS
0200	DS
	SS
	ES
BEED	AX
	BX
	CX
	DX
	SP
	BP
1234	SI
	DI

Address	Memory content	Instruction
01000	8B	MOV AX, [SI]
01001	04	
01002	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
03234	ED	Source operand
03235	BE	

After execution

Based Addressing Mode – 1/8

- In this addressing mode, the **EA is obtained by adding a direct or indirect displacement to the contents of either BX or BP register**



Based Addressing Mode – 2/8

- The value in the base register defines the beginning of a data structure (e.g. array) in memory, and the displacement selects an element of data within the structure

- To access a different element in the array, the programmer simply changes the value of the displacement

Based Addressing Mode – 3/8

- To access the same element in another similar array, the programmer can change the value in the base register so that it points to the beginning of the new array

```
MOV [BX]+1234H, AL
```

- This instruction uses base register BX and direct displacement 1234H to derive the destination operand

Based Addressing Mode – 4/8

- The based addressing mode is implemented by specifying the base register in brackets followed by a + sign and direct displacement

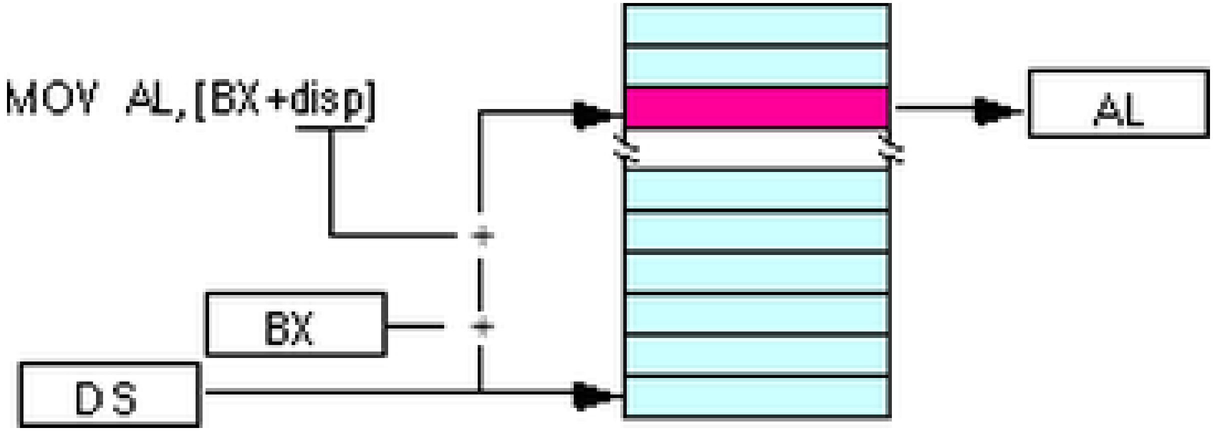
- The microprocessor calculates the physical address of the destination operand from the contents of DS, BX and the direct displacement, and then moves the contents of AL into memory location

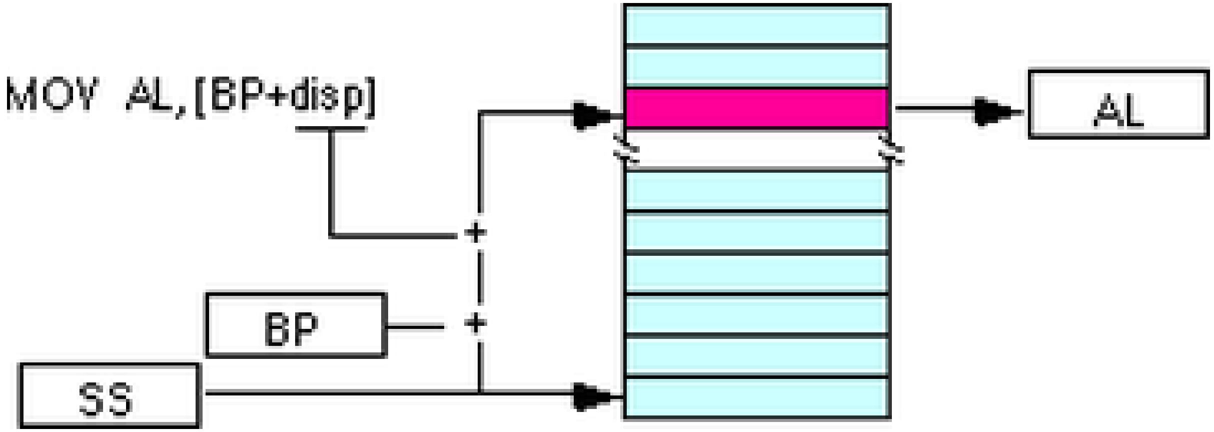
- **The default segment register for physical address calculation is DS register**, but it can be changed to other segment register with the segment override prefix

Based Addressing Mode – 5/8

- If BP is used instead of BX, the calculation of the physical address is performed using the contents of the stack segment (SS) register instead of DS register

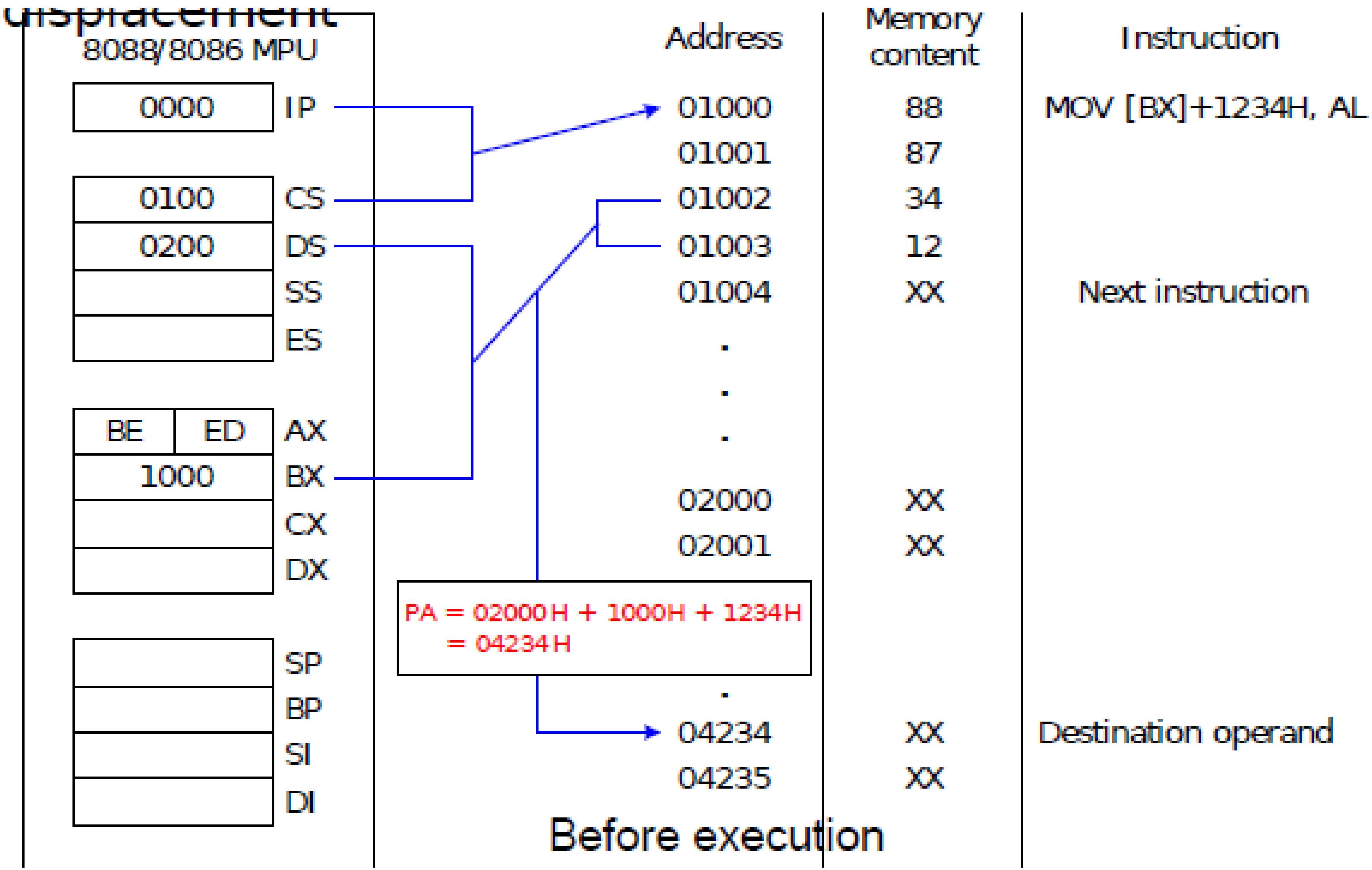
Based Addressing Mode – 6/8





Based Addressing Mode – 7/8

displacement



8088/8086 MPU

0000	IP
0100	CS
0200	DS
	SS
	ES
BE ED	AX
1000	BX
	CX
	DX
	SP
	BP
	SI
	DI

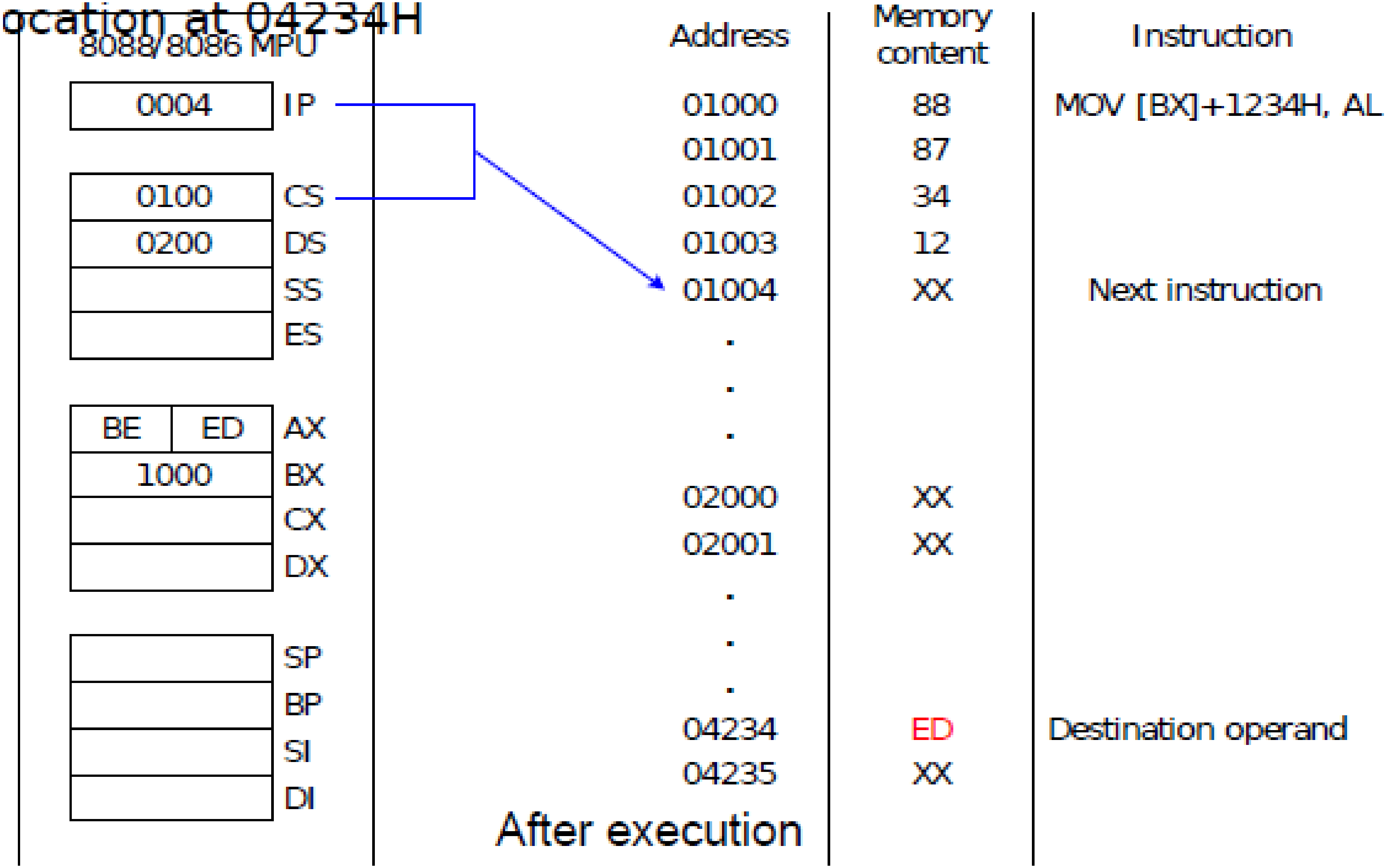
Address	Memory content	Instruction
01000	88	MOV [BX]+1234H, AL
01001	87	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
04234	XX	Destination operand
04235	XX	

PA = 02000H + 1000H + 1234H
= 04234H

Before execution

Based Addressing Mode – 8/8

ocation at 04234H



8088/8086 MPU

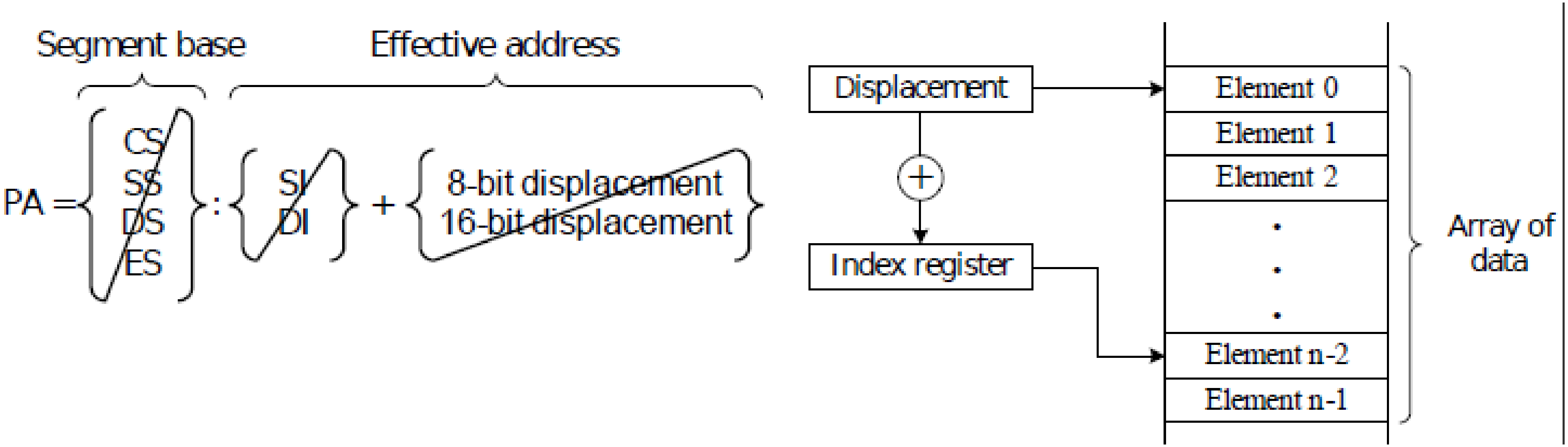
0004		IP
0100		CS
0200		DS
		SS
		ES
BE	ED	AX
1000		BX
		CX
		DX
		SP
		BP
		SI
		DI

Address	Memory content	Instruction
01000	88	MOV [BX]+1234H, AL
01001	87	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
04234	ED	Destination operand
04235	XX	

After execution

Indexed Addressing Mode – 1/4

- Indexed addressing mode **uses the value of the displacement as a pointer to the starting point of an array of data in memory and the contents of the specified register as an index that selects the specific element in the array to be accessed**

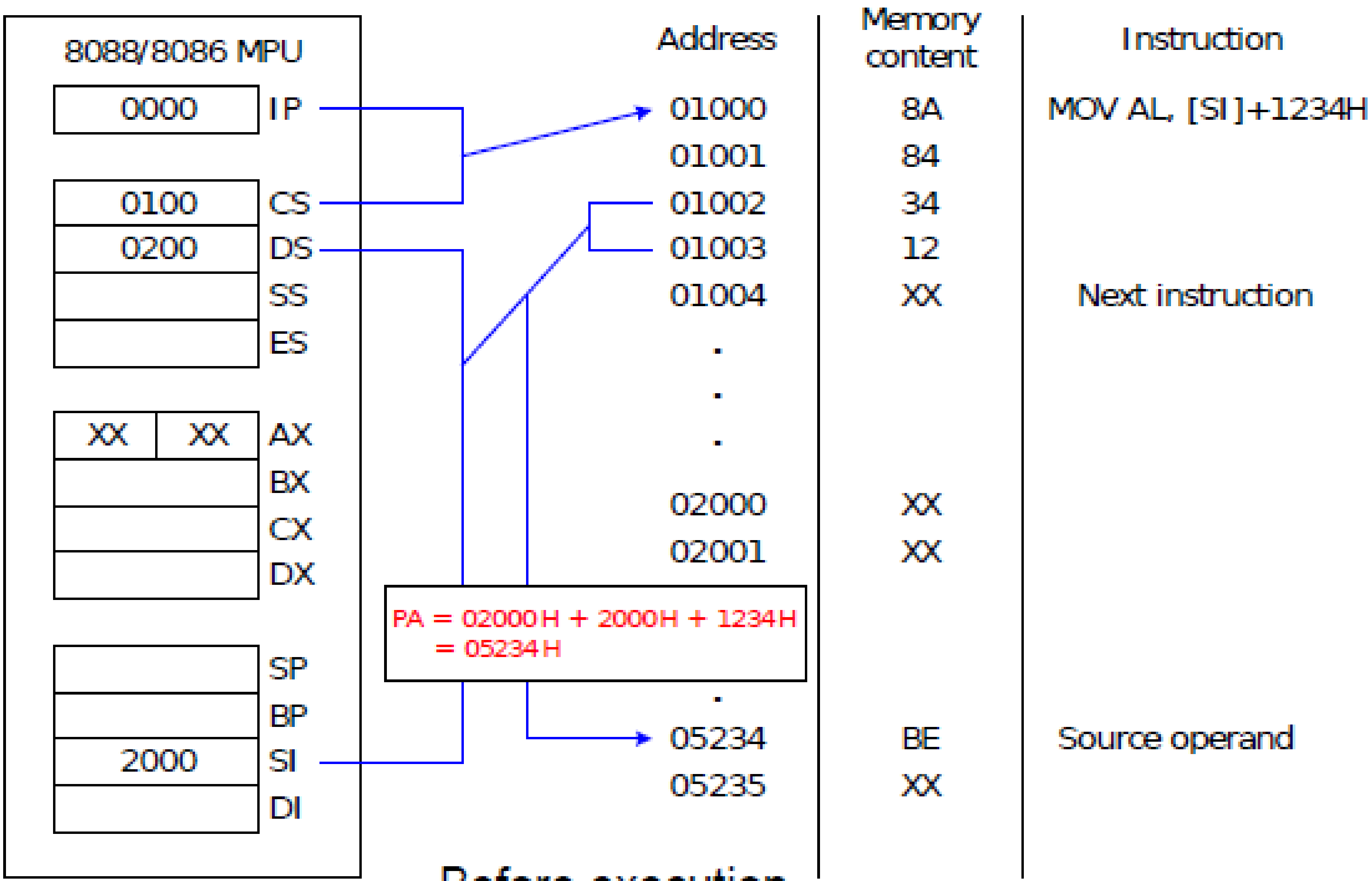


Indexed Addressing Mode – 2/4

MOV AL, [SI]+1234H

- The physical address of the source operand is first calculated from the contents of DS, SI and the direct displacement
- The data byte stored at the address is then moved to AL

Indexed Addressing Mode – 3/4

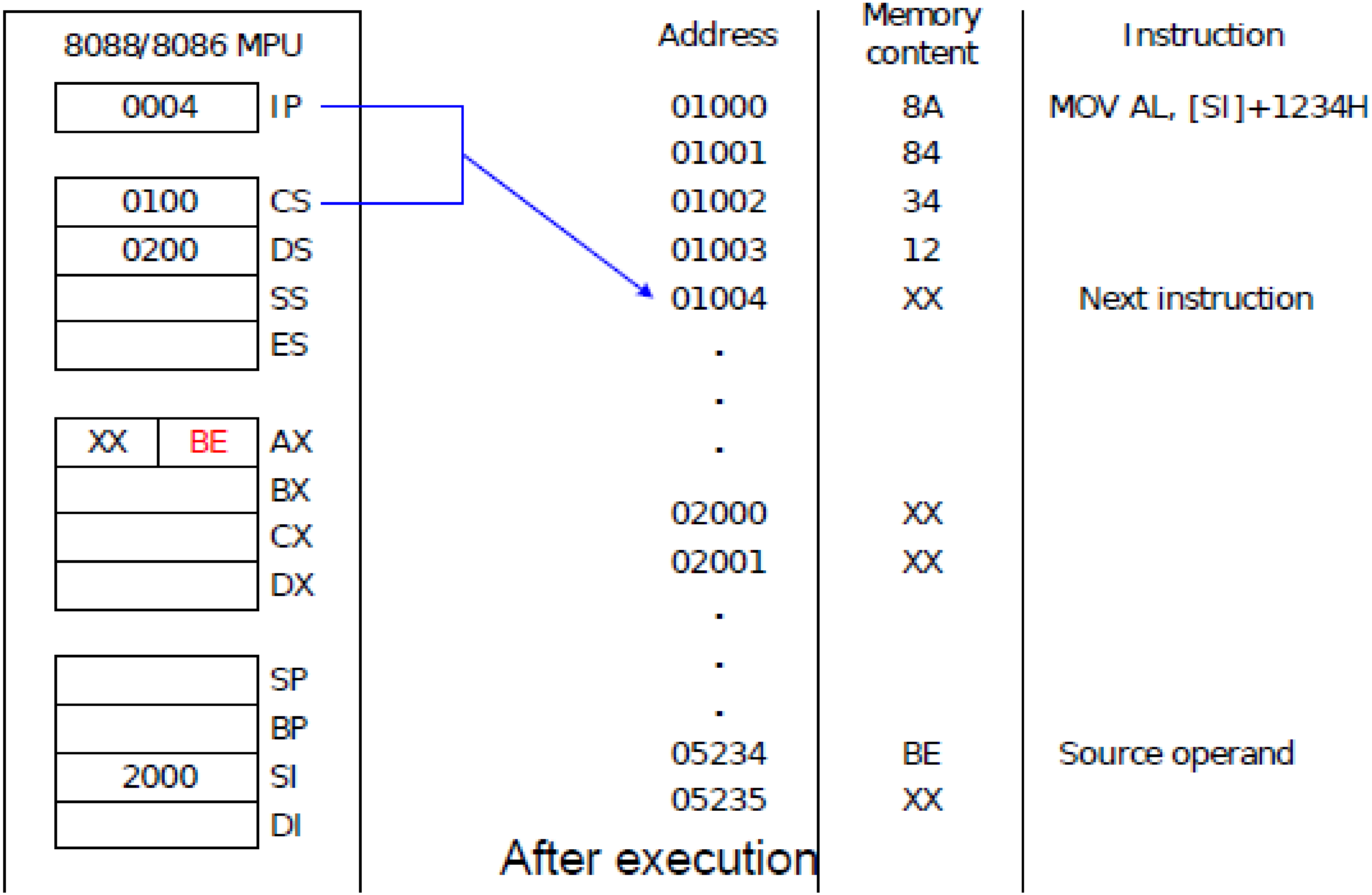


Address	Memory content	Instruction
01000	8A	MOV AL, [SI]+1234H
01001	84	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
05234	BE	Source operand
05235	XX	

PA = 02000 H + 2000 H + 1234 H
= 05234 H

Before execution

Indexed Addressing Mode – 4/4



8088/8086 MPU

0004		IP
0100		CS
0200		DS
		SS
		ES
XX	BE	AX
		BX
		CX
		DX
		SP
		BP
2000		SI
		DI

Address	Memory content	Instruction
01000	8A	MOV AL, [SI]+1234H
01001	84	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
05234	BE	Source operand
05235	XX	

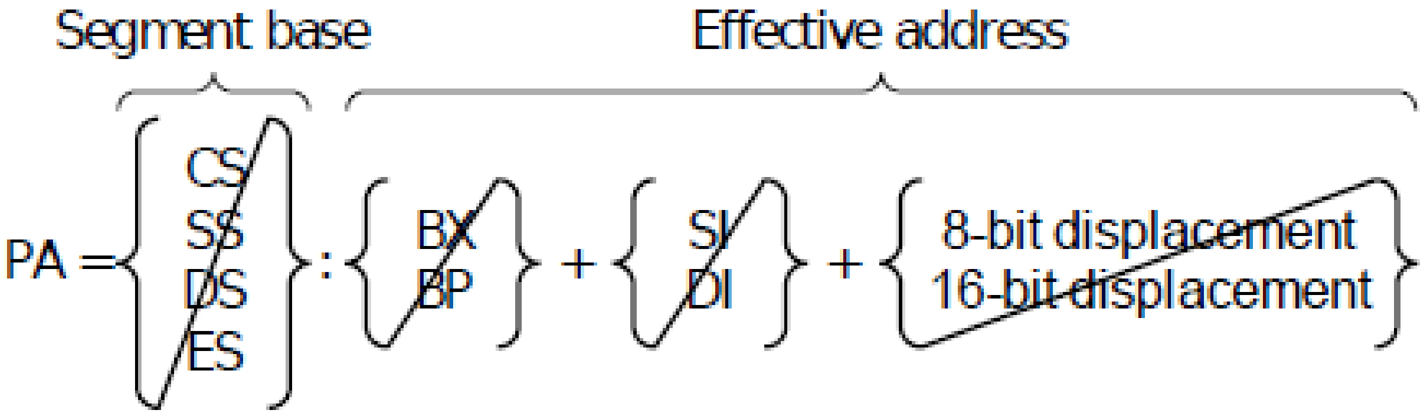
After execution

Based-Indexed Addressing Mode – 1/7

- This addressing mode is a combination of the based addressing mode and the indexed addressing mode

- The effective address is formed by three elements: **base register, index register and a displacement**

Based-Indexed Addressing Mode – 2/7



MOV AH, [BX][SI]+1234H

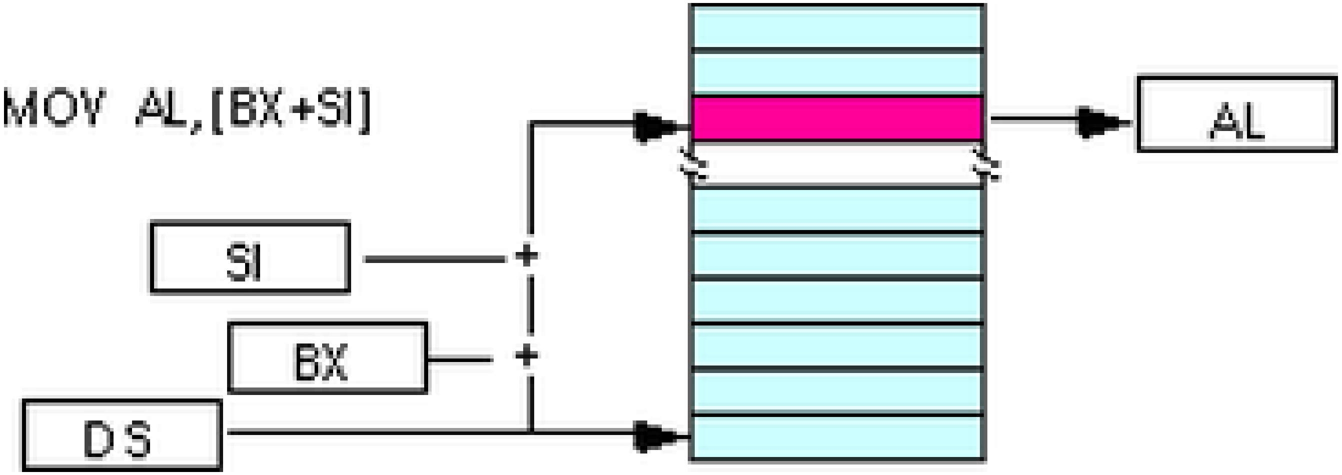
- Before the source operand can be moved to the destination, the 8086 calculates its physical address
- The effective address is formed by BX and SI registers with 16-bit displacement

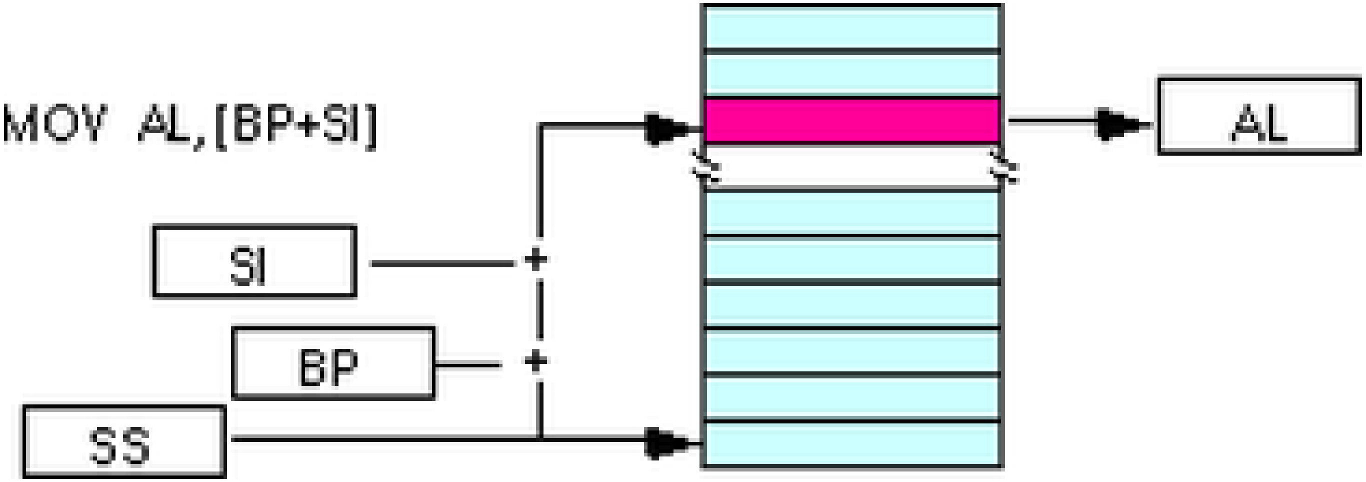
Based-Indexed Addressing Mode – 3/7

- The physical address is then computed from the current contents of DS register and the effective address

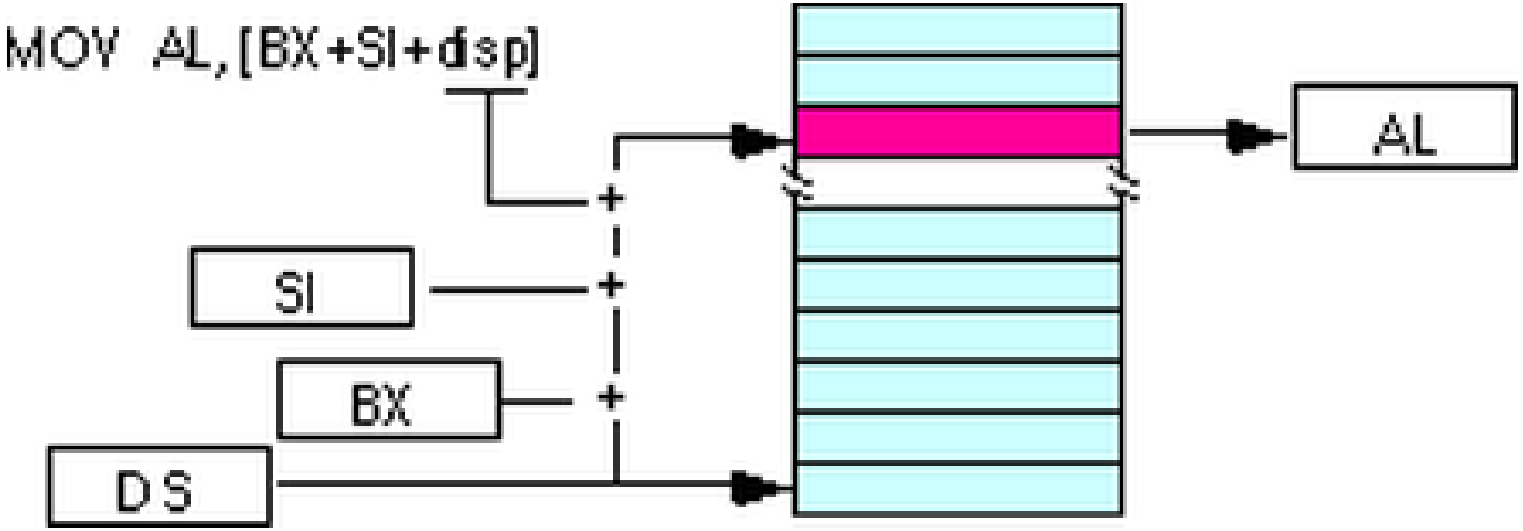
- Execution of the instruction moves the value stored at the physical address into AH

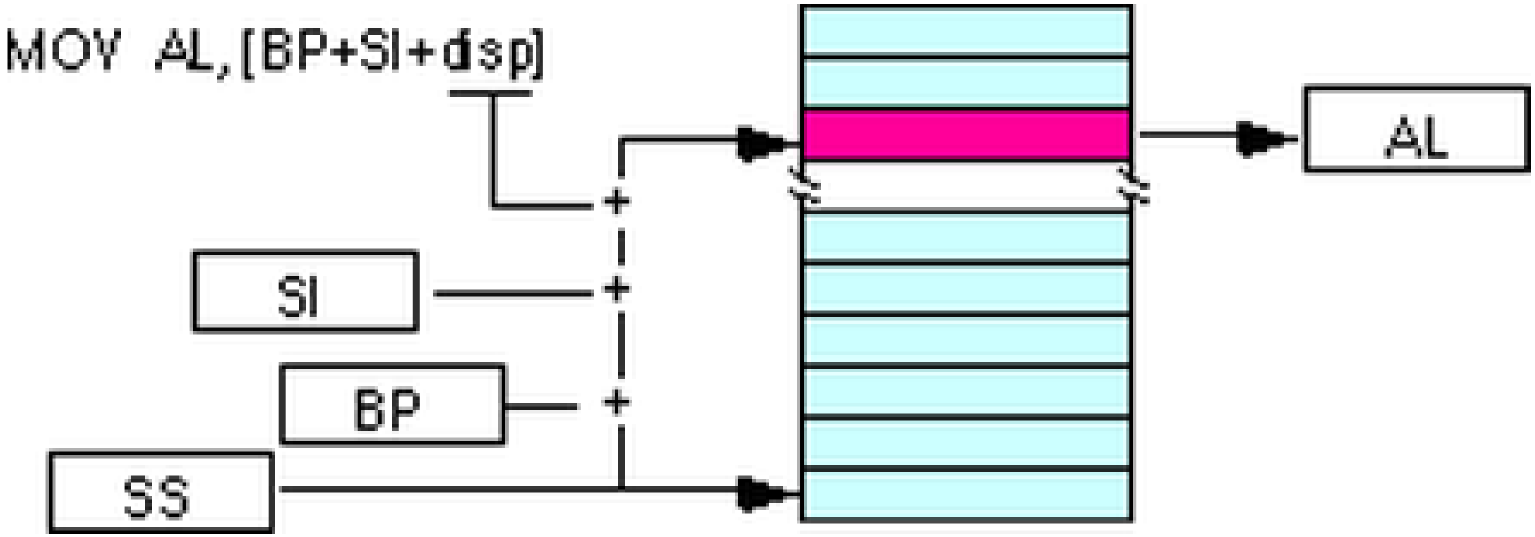
Based-Indexed Addressing Mode – 4/7



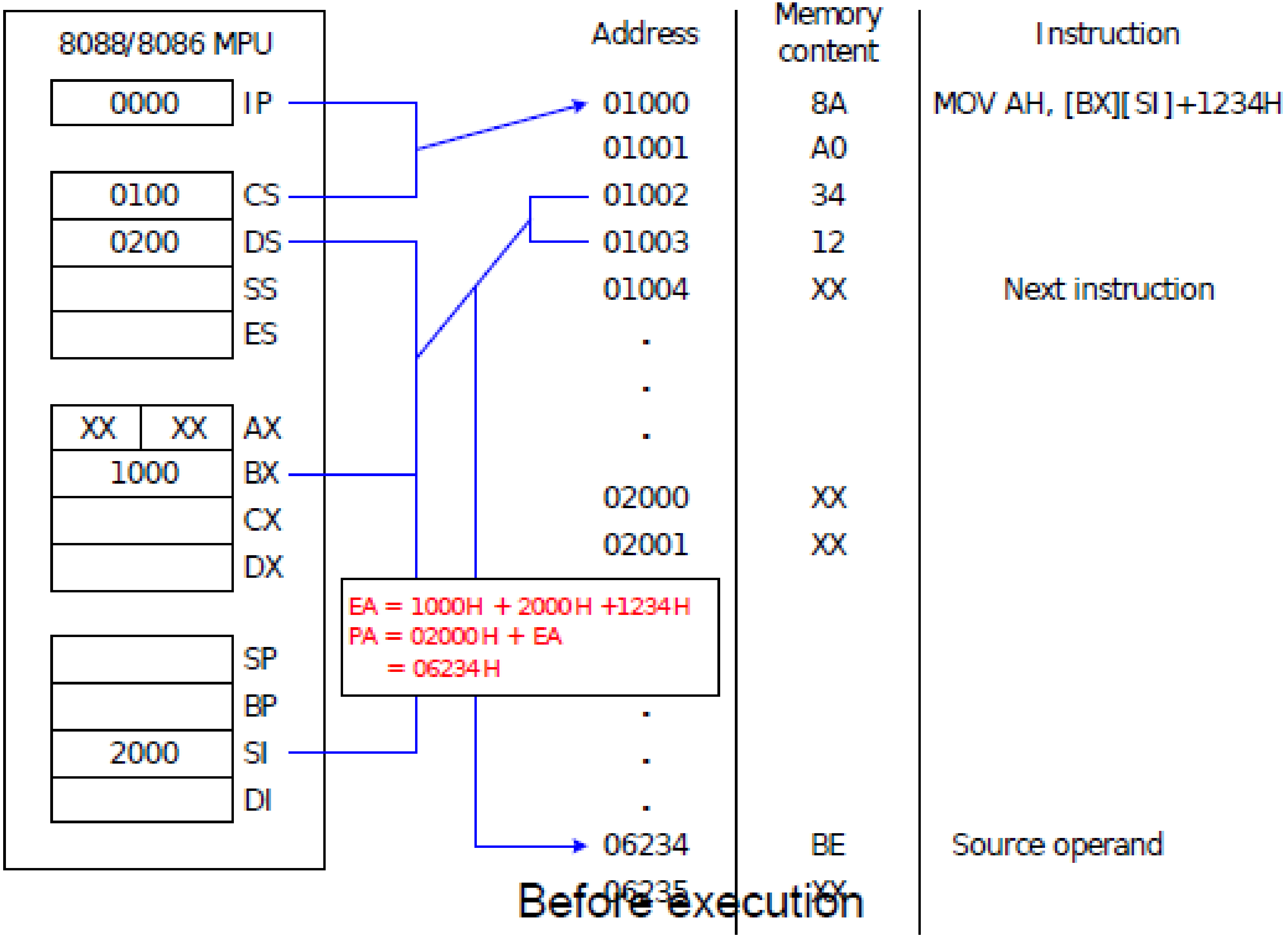


Based-Indexed Addressing Mode – 5/7





Based-Indexed Addressing Mode – 6/7



8088/8086 MPU

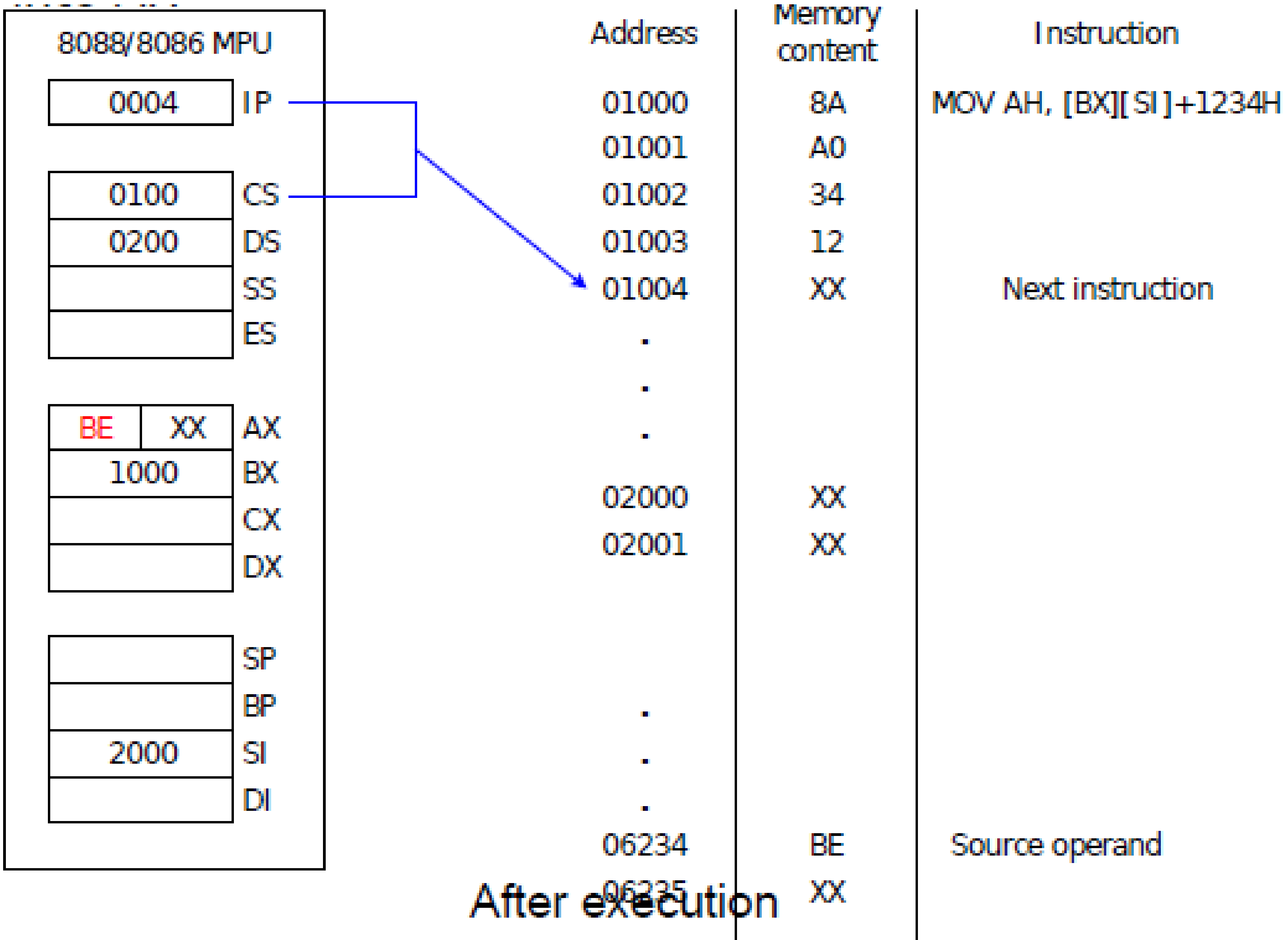
0000	IP
0100	CS
0200	DS
	SS
	ES
XX XX	AX
1000	BX
	CX
	DX
	SP
	BP
2000	SI
	DI

Address	Memory content	Instruction
01000	8A	MOV AH, [BX][SI]+1234H
01001	A0	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
06234	BE	Source operand
06235	XX	

EA = 1000H + 2000H +1234H
PA = 02000H + EA
= 06234H

Before execution

Based-Indexed Addressing Mode – 7/7



8088/8086 MPU

0004	IP
0100	CS
0200	DS
	SS
	ES
BE XX	AX
1000	BX
	CX
	DX
	SP
	BP
2000	SI
	DI

Address	Memory content	Instruction
01000	8A	MOV AH, [BX][SI]+1234H
01001	A0	
01002	34	
01003	12	
01004	XX	Next instruction
.		
.		
.		
02000	XX	
02001	XX	
.		
.		
.		
06234	BE	Source operand
06235	XX	

After execution

Summary of Addressing modes

Addressing mode	Operand	Default segment
Register	Reg	None
Immediate	Data	None
Direct	[offset]	DS
Register indirect	[BX] [SI] [DI]	DS DS DS
Based	[BX]+disp [BP]+disp	DS SS
Index	[DI]+disp [SI]+disp	DS DS
Based indexed	[BX][SI or DI]+disp [BP][SI or DI]+disp	DS SS

Addressing mode	***Example***	***Remarks***
Immediate	MOV AX, 1234H	Data=1234H
Register	MOV AX, BX	AX contains the data
Direct	MOV AX, [1234H]	Data disp.=1234H
Register indirect	MOV AX, [BX]	Data disp.=(BX)
Register indirect	MOV AX, CS: [BX]	Segment override: Segment base=(CS) Date disp.=(BX)
Based	MOV AX, 12H[BX]	Data disp.=12H+(BX)
Indexed	MOV AX, 34H[SI]	Data disp.=34H+(SI)
Based and Indexed	MOV AX, 56H[SI] [BX]	Data disp.=56H + (SI) + (BX)

Fig: Addressing modes of 8088 (‘(…)’ implies ‘contents of’)