Module 1

8086 Microprocessor

Memory Segmentation & Banking

Bharati Ingale (bharati_ingale@yahoo.com)

8086 – Programmer's Model of 8086

		ES		Extra Segment
BIU Registers		CS		Code Segment
		SS		Stack Segment
		DS		Data Segment
		IP		Instruction Pointer
	AX	AH	AL	Accumulator
	BX	BH	BL	Base Register
	CX	CH	CL	Count Register
EU Registers	DX	DH	DL	Data Register
		SP		Stack Pointer
		BP		Base Pointer
		SI		Source Index Register
		DI		Destination Index Register
		FLAGS		

8086- Architecture – Memory Segmentation

- **Memory Segmentation means dividing the memory into logical segments.**

8086- Architecture – Memory Segmentation

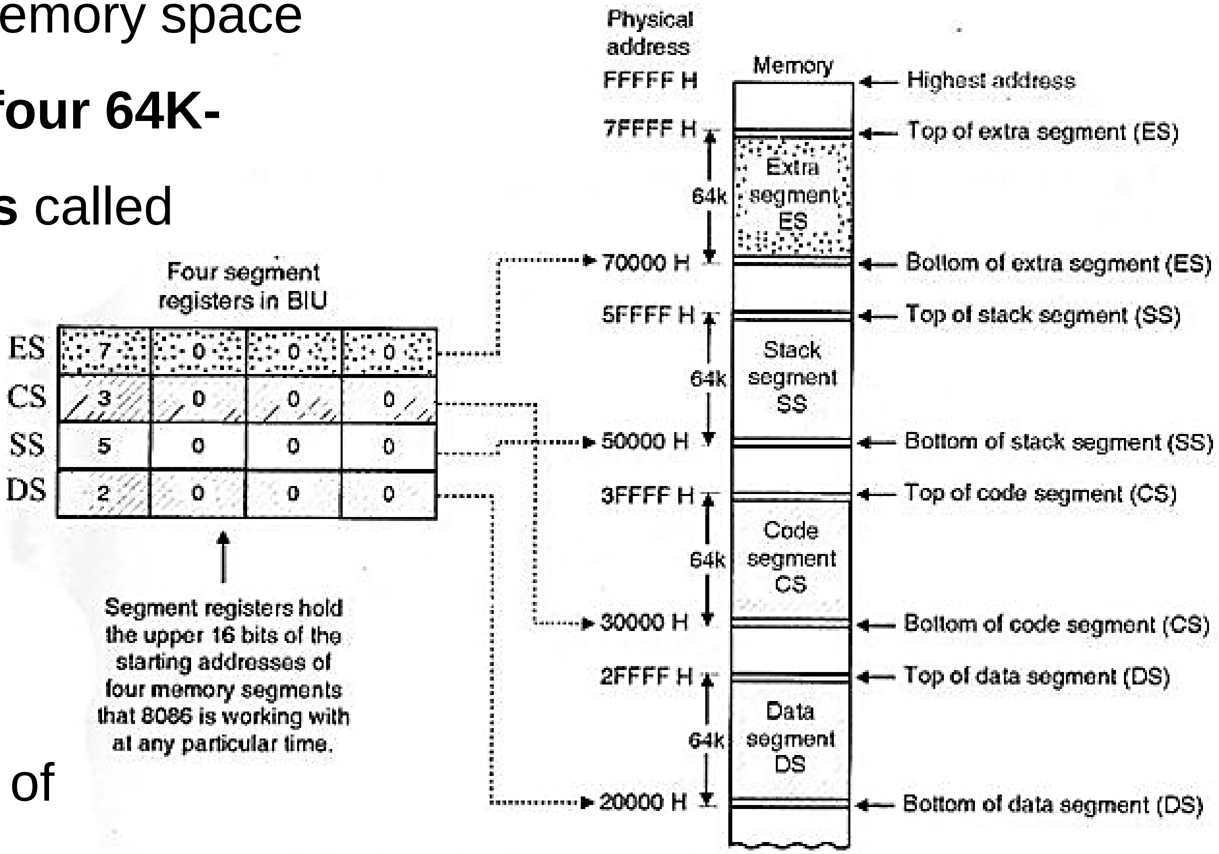
- 8086 has 20-bit address bus. ∴ can **access 2^{20} memory locations i.e. 1 MB memory.**
- Memory is **divided into segments** of max. size of **64 KB** each.
- The programmer can define various segments.
- In 8086 we can access only 4 memory segments at a time.

8086 – Memory Segmentation

- Within the 1 MB of memory space the 8086/88 defines **four 64K-byte memory blocks** called

1. Code segment,
2. Stack segment,
3. Data segment,
4. Extra segment.

- Each of these blocks of memory is used differently by the processor.

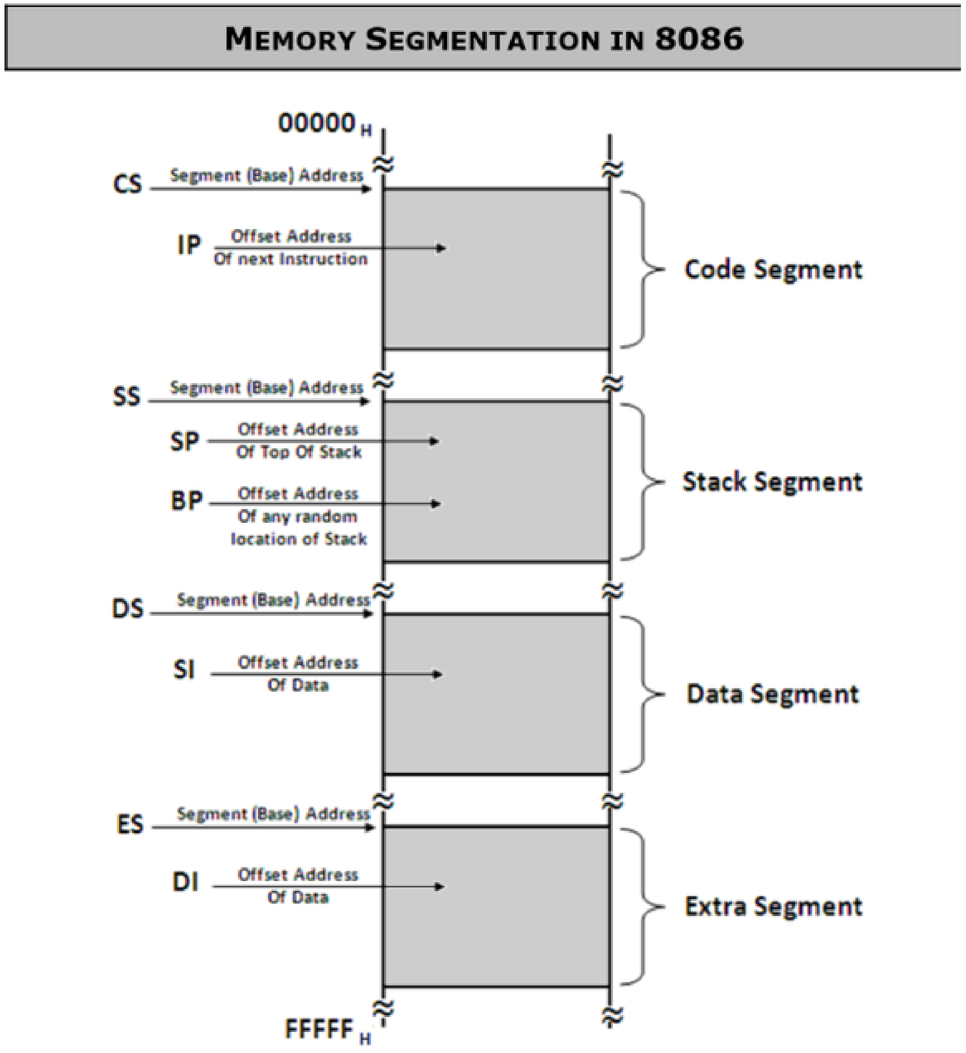


Four segment registers in BIU
ES 7 0 0 0
CS 3 0 0 0
SS 5 0 0 0
DS 2 0 0 0
Segment registers hold the upper 16 bits of the starting addresses of four memory segments that 8086 is working with at any particular time.
Physical address
Memory
FFFFF H
7FFFF H
70000 H
5FFFF H
50000 H
3FFFF H
30000 H
2FFFF H
20000 H
64k
Extra segment ES
Stack segment SS
Code segment CS
Data segment DS
Highest address
Top of extra segment (ES)
Bottom of extra segment (ES)
Top of stack segment (SS)
Bottom of stack segment (SS)
Top of code segment (CS)
Bottom of code segment (CS)
Top of data segment (DS)
Bottom of data segment (DS)

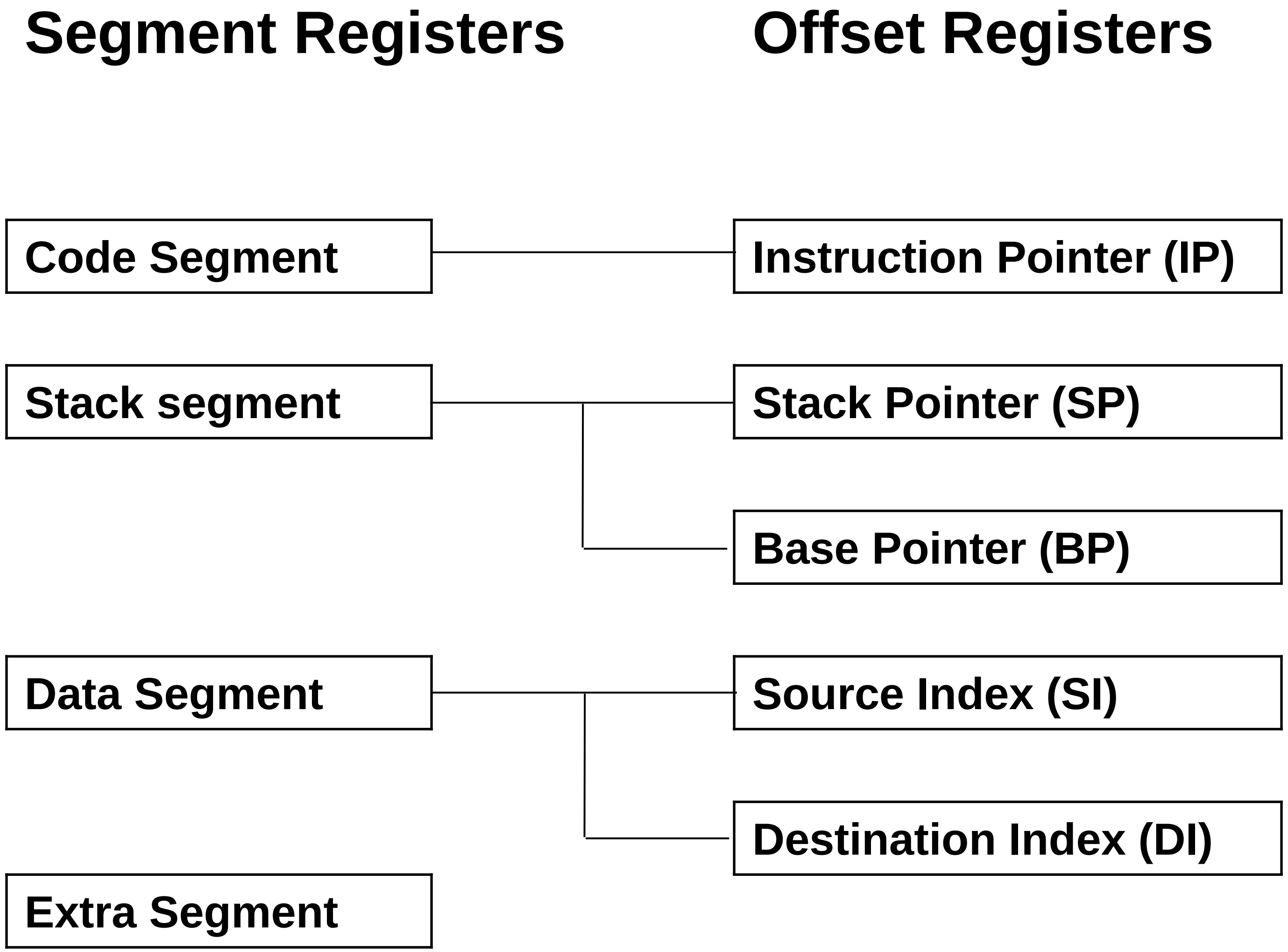
One way of positioning four 64k byte segments within the 1M byte memory space of an 8086

8086 – Memory Segmentation

- 8086 has 4 16 – bit registers
  - CS; DS; SS; ES to hold the base address of the segments.
- 8086 has 16 – bit offset registers
  - IP, SP,BP,SI,DI to hold the offset address for each of the segments.



8086 – Addressing Modes



8086 – Memory Segmentation

- Eg: SP register has offset address for the Stack Segment

  20 bit physical address = SP + (SS * 10H)

- SI holds the offset address for the Data Segment.

  20 bit physical address = SI + (DS * 10H)

8086 – Memory Segmentation

- Code Segment

  i. This segment is used to **hold the program instruction codes** to be executed

  ii. Instruction Fetch operation is performed on CS memory

  iii. **CS** reg holds the **base** address

  iv. **IP** reg holds the 16 bit **offset** address

8086 – Memory Segmentation

- Data Segment**.**

  i. This segment is used to **store general data** for the program

  ii. It also holds **source operands** during string operations.

  iii. DS reg holds the 16- bit **base** address

  iv. BX reg holds the 16 bit **offset** address

  v. SI reg holds the16 bit **offset** address during **String Operations**

8086 – Memory Segmentation

- Stack Segment.

i. This segment is used to **store interrupt and subroutine return addresses.**

ii. It holds the stack memory, which operates in LIFO manner.

iii. **SS** reg holds the 16- bit **base address**

iv. **SP** reg holds the 16 bit **offset** address of the **Top of the Stack**

v. **BP** reg holds the16 bit **offset** address during **Random Access.**

8086 – Memory Segmentation

- Extra Segment

  i. This segment is an **extra data segment** (often used for shared data).

  ii. It is mainly used to hold the **destination** operands during S**tring Operations**

  iii. **ES** reg holds the base address

  iv. **DI** holds the 16 bit **offset** address during String Operations

8086 – Memory Segmentation

- Programs obtain access to **code & data** in the **code segment & data segment** by changing the segment register contents to point to the desired segments.

- All program instructions must be located in main memory – **Code Segment**
  - pointed to by the 16-bit **CS register** with
  - a 16-bit offset in the segment contained in the **16-bit instruction pointer (IP).**

8086 – Memory Segmentation

- The BIU computes the 20-bit physical address internally by the provided logical address (16-bit contents of CS and IP)

- It is done by logically shifting the contents of CS four bits to left and then adding the 16-bit contents of IP.

- In other words, the CS is multiplied by 16_{10} or 10_{16} by the BIU for computing the 20-bit physical address.

Physical Address = Segment Address x 10H+ IP

8086 – Memory Segmentation

Eg:

if [CS] = $348A_{16}$ & [IP] = 4214_{16}, then the 20-bit physical address generated by the BIU is as follows:

```
    Four times logically shifted [CS] to left    =   348A0₁₆
+   [IP] as offset                               =    4214₁₆
--------------------------------------------------------------
20-bit physical address                          =  38AB4₁₆
```

The BIU always inserts four Zeros for the lowest 4-bits of the 20-bit starting address (physical) of a segment.

8086 – Memory Segmentation

- The 20-bit Physical address is often represented as,

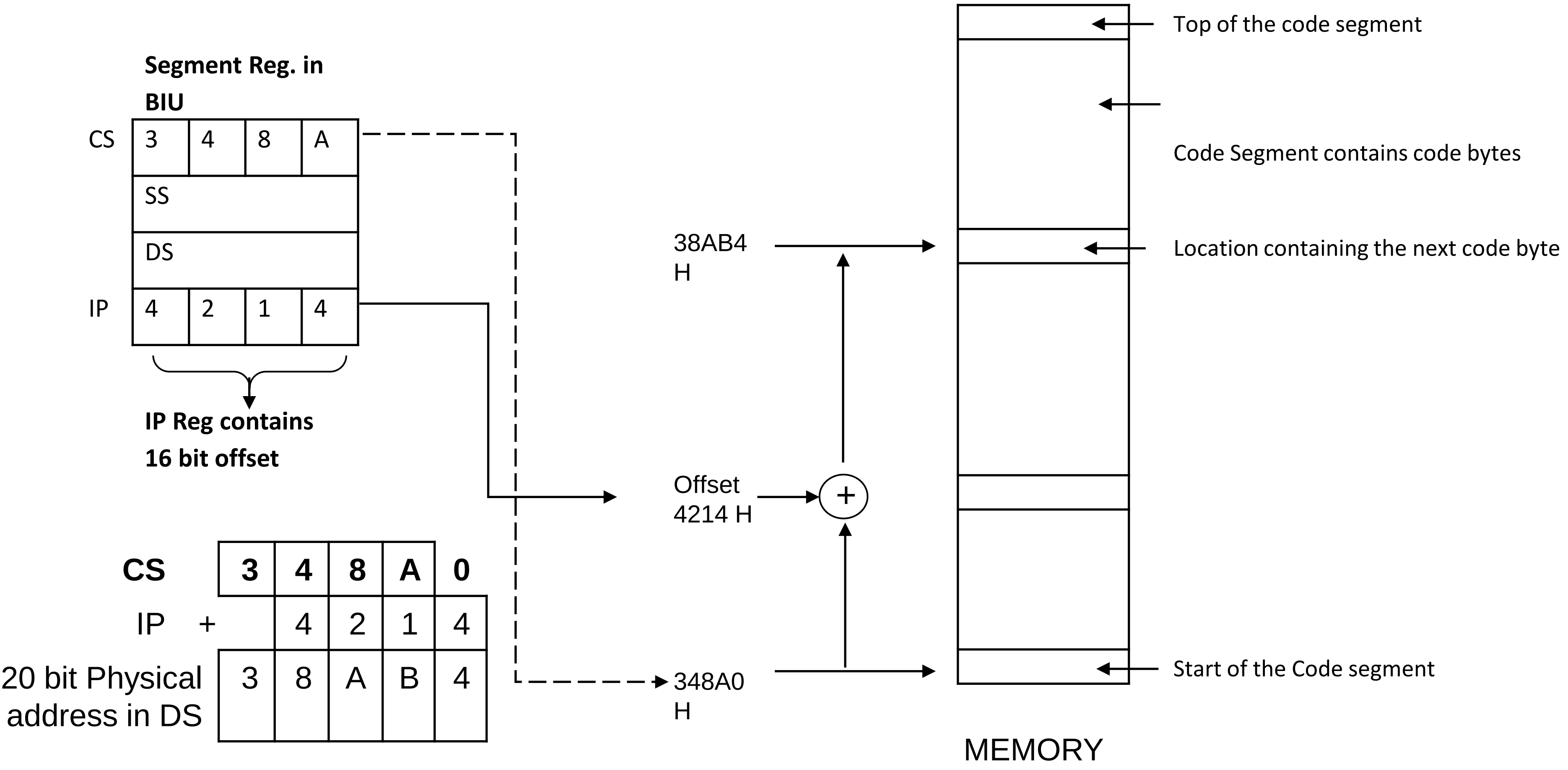
  Segment Base : Offset

  OR

- CS : IP $\longrightarrow$ $[CS] = 348A_{16}$ & $[IP] = 4214_{16}$

CS		**3**	**4**	**8**	**A**	**0**
IP	+		4	2	1	4
20 bit Physical address in DS		3	8	A	B	4

8086- Compuation of Address


Segment Reg. in BIU
CS
3
4
8
A
SS
DS
IP
4
2
1
4
IP Reg contains 16 bit offset
38AB4 H
Offset 4214 H
+
348A0 H
Top of the code segment
Code Segment contains code bytes
Location containing the next code byte
Start of the Code segment
MEMORY

CS	3	4	8	A	0
IP +		4	2	1	4
20 bit Physical address in DS	3	8	A	B	4

8086 – Memory Segmentation

Advantages:

1. Allow the programmer to **access1Mb memory using 16 - bit address.**
2. It **divides memory logically** to store Instructions, Data & Stack separately.
3. Permits a program and/or its data to be put into different areas of memory each time the program is executed.

8086 – Memory Segmentation

Advantages:

4. **Multitasking** becomes easy.
5. Segmentation is very useful for **multi-user environment.**
6. **It provides powerful memory mangement mechanism.**
7. Modular software design

8086 – Memory Segmentation

Disadvantages:

1. Although the total memory is 16* 64KB, at a time only 4* 64 KB memory can be accessed.

2. Any **memory location** needs to be expressed using **2 registers- Base & Offset.**.

8086 – Memory Banks

8086 – Memory Banks

- 8086 has a **16-bit data bus** ∴ it can access **16- bit data in one operation**.
- But memory chips available are normally such that one memory location has **8 bit (1 byte)**.
- 1 Memory locations carries one byte- 8 bits.
- To access 16-bit data it needs to read **2 memory locations**.

8086 – Memory Banks

- If **both memory locations** are consecutive in the **same memory chip** then the **address bus has to contain 2 addresses** at the same time and hence **require double time**. This is impossible.

- Therefore to solve this problem,the memory of 8086 is divided into 2 banks.
- Each bank provides 1byte or 8bits.

8086 – Memory Banks

- One bank contains all even addresses called “**Even Bank**”.
- The other bank contains all odd addresses called “**Odd Bank**”.

8086 – Memory Banks

$$\overbrace{\boxed{\text{512 KB}} \qquad\qquad\qquad \boxed{\text{512 KB}}}^{\textbf{1 MB}}$$

Odd Bank

- Also called as "Higher Bank"
- Address range

  00001H

  00003H

  00005H

  ⋮

  FFFFFH

- Selected when $\overline{\text{BHE}}$ = 0

Even Bank

- Also called as "Lower Bank"
- Address Range

  00000H

  00002H

  00004H

  ⋮

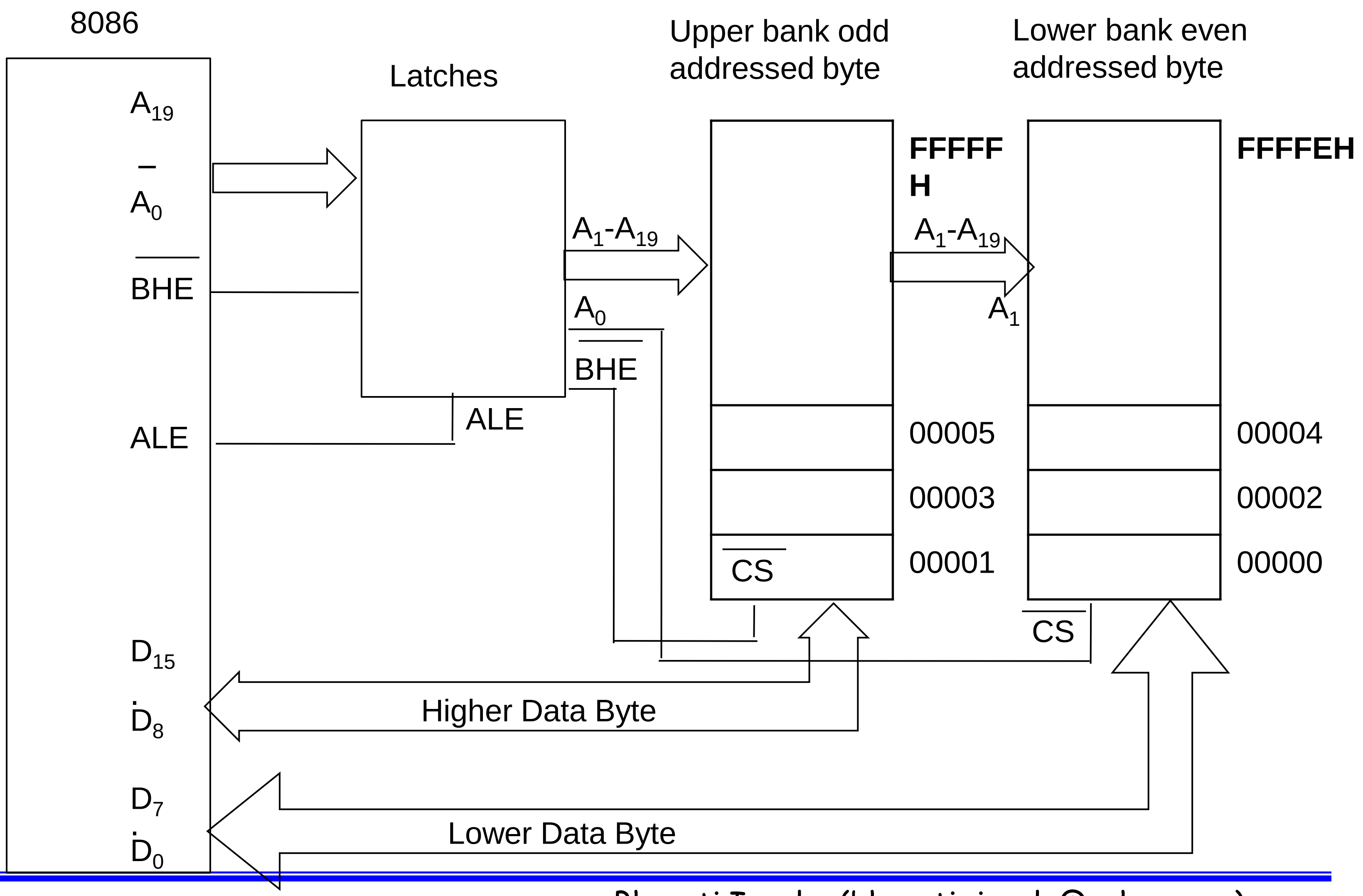
  FFFFFH

- Selected when A0 =0

8086 – Memory Banks

$\overline{BHE}$	A0	Operation
0	0	R/w 16 – bit from both banks
0	1	R/w 8 – bit from higher banks
1	0	R/w 8 – bit from lower banks
1	1	No Operation

8086 – Memory Banks



Thank You