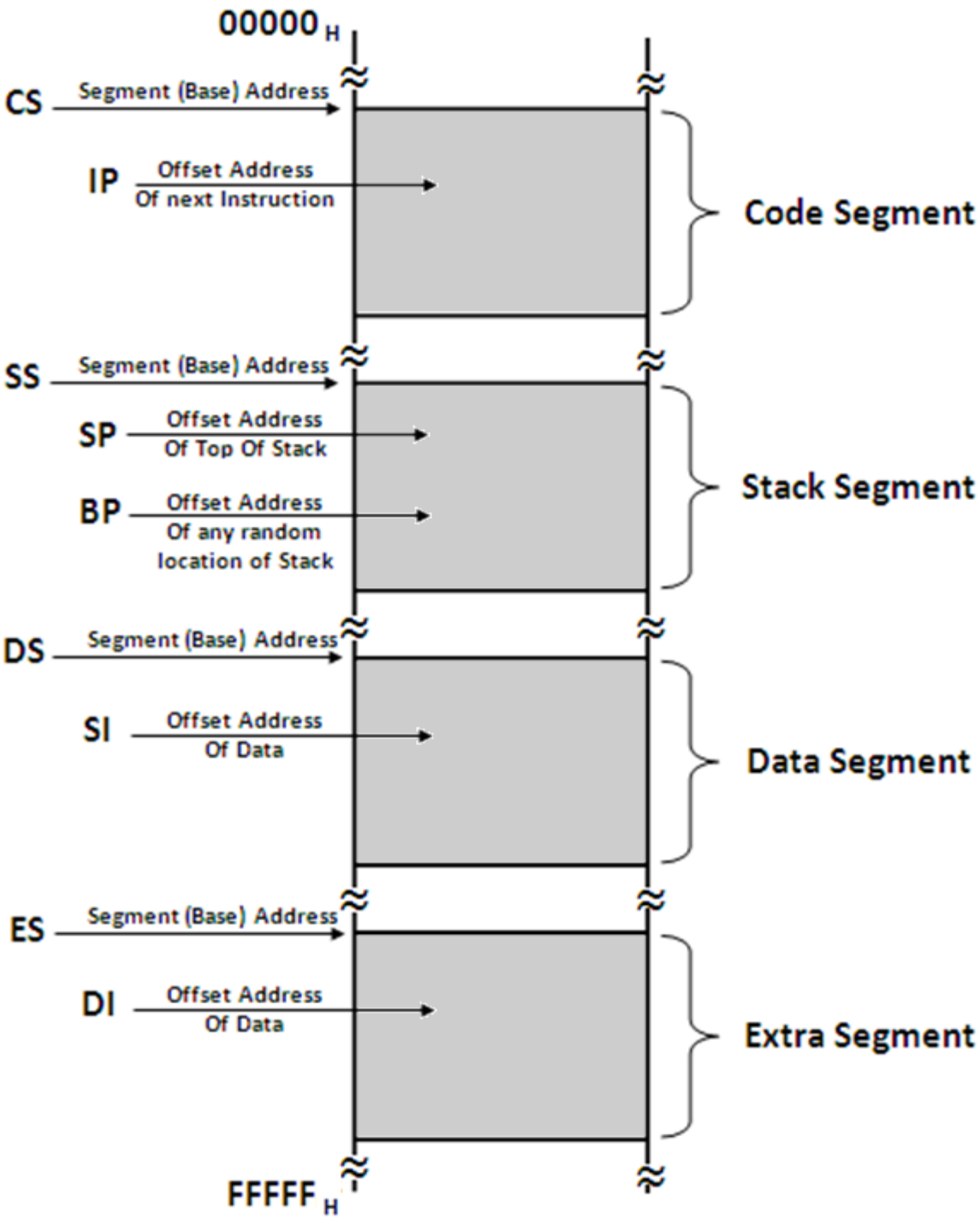
MEMORY SEGMENTATION IN 8086



NEED FOR SEGMENTATION/ CONCEPT OF SEGMENTATION

1) Segmentation means **dividing** the memory into **logically different parts called segments**.
2) 8086 has a **20-bit address bus**, hence it can access 2^{20} Bytes i.e. **1MB** memory.
3) But this also means that **Physical address** will now be **20 bit**.
4) It is **not possible** to work with a **20 bit address** as it is **not** a **byte compatible** number. (20 bits is two and a half bytes).
5) To avoid working with this incompatible number, we **create a virtual model** of the memory.
6) Here the memory is **divided into 4 segments**: Code, Stack Data and Extra.
7) The **max size** of a segment is **64KB** and the **minimum size** is **16 bytes**.
8) Now programmer can access each location with a **VIRTUAL ADDRESS**.
9) The Virtual Address is a **combination** of **Segment Address and Offset Address**.
10) **Segment Address indicates where the segment is located in the memory (base address)**
11) **Offset Address gives the offset of the target location within the segment.**
12) Since both, Segment Address and Offset Address are **16 bits each**, they both are **compatible numbers** and can be easily used by the programmer.
13) Moreover, **Segment Address is given only in the beginning** of the program, to initialize the segment. Thereafter, we **only give offset address**.
14) **Hence we can access 1 MB memory using only a 16 bit offset address for most part of the program. This is the advantage of segmentation.**
15) Moreover, dividing Code, stack and Data into different segments, makes the memory **more organized and prevents accidental overwrites** between them.
16) The **Maximum Size** of a segment is **64KB because offset addresses are of 16 bits**. 2^{16} = 64KB.
17) As max size of a segment is 64KB, programmer can create **multiple Code/Stack/Data segments** till the entire 1 MB is utilized, but **only one of each** type will be **currently active**.
18) The physical address is calculated by the microprocessor, using the formula:

    PHYSICAL ADDRESS = SEGMENT ADDRESS X 10H + OFFSET ADDRESS
19) Ex: if Segment Address = 1234H and Offset Address is 0005H then
    Physical Address = 1234H x 10H + 0005H = 12345H
20) This formula automatically ensures that the **minimum size of a segment is 10H bytes** (10H = 16 Bytes).

Code Segment

This segment is used to hold the **program** to be executed.
Instruction are fetched from the Code Segment.
CS register holds the 16-bit **base** address for this segment.
IP register (Instruction Pointer) holds the 16-bit **offset** address.

Data Segment

This segment is used to hold **general data**.
This segment also holds the **source** operands during **string** operations.
DS register holds the 16-bit **base** address for this segment.
BX register is used to hold the 16-bit **offset** for this segment.
SI register (Source Index) holds the 16-bit **offset** address during String Operations.

Stack Segment

This segment holds the **Stack** memory, which operates in LIFO manner.
SS holds its **Base** address.
SP (Stack Pointer) holds the 16-bit **offset** address of the **Top** of the Stack.
BP (Base Pointer) holds the 16-bit **offset** address during **Random Access**.

Extra Segment

This segment is used to hold **general data**
Additionally, this segment is used as the **destination** during **String Operations**.
ES holds the **Base** Address.
DI holds the **offset** address during string operations.

Advantages of Segmentation:

1) It permits the programmer to access 1MB **using only 16-bit address**.
2) Its **divides** the **memory logically** to store Instructions, Data and Stack separately.

Disadvantage of Segmentation:

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