Intel 8086 Microprocessor
Memory Partition and Registers

Department of Computer Science & Engineering
BRAC University.

Course ID: CSE341
Course Title: Microprocessors

Lecture References:

- **Book:**
  - *Microprocessors and Interfacing: Programming and Hardware, Chapter # 2,* **Author:** Douglas V. Hall
  - *The 8086/8088 Family: Design, Programming, And Interfacing, Chapter # 2,* **Author:** John Uffenbeck.

Microprocessor

- **Intel 8086**
  - The microprocessor 8086 can be considered to be the basic processor for the Intel X86 family from 1978.
  - It has a 20-bit address bus along with 16-bit data bus.
  - With the knowledge of 8086 16-bit processor, one can study the further versions of this processor 80286, 80406 and Pentium.

Memory Partitioning for 8086 Processor

- **The 8086 processor assign a 20-bit physical address to its memory locations.**
  - **2^{20} → 1 Mbyte -> 1,048,576 bytes**
  - **20 bits → 5 hex digits**
  - **First addresses: 00000, 00001,…,0000A,…FFFFF.**
  - **But Registers are 16-bits (4 Hex digits) and can address only 2^{16} = 64 KBytes. 0000,0001,0002,……FFFF**

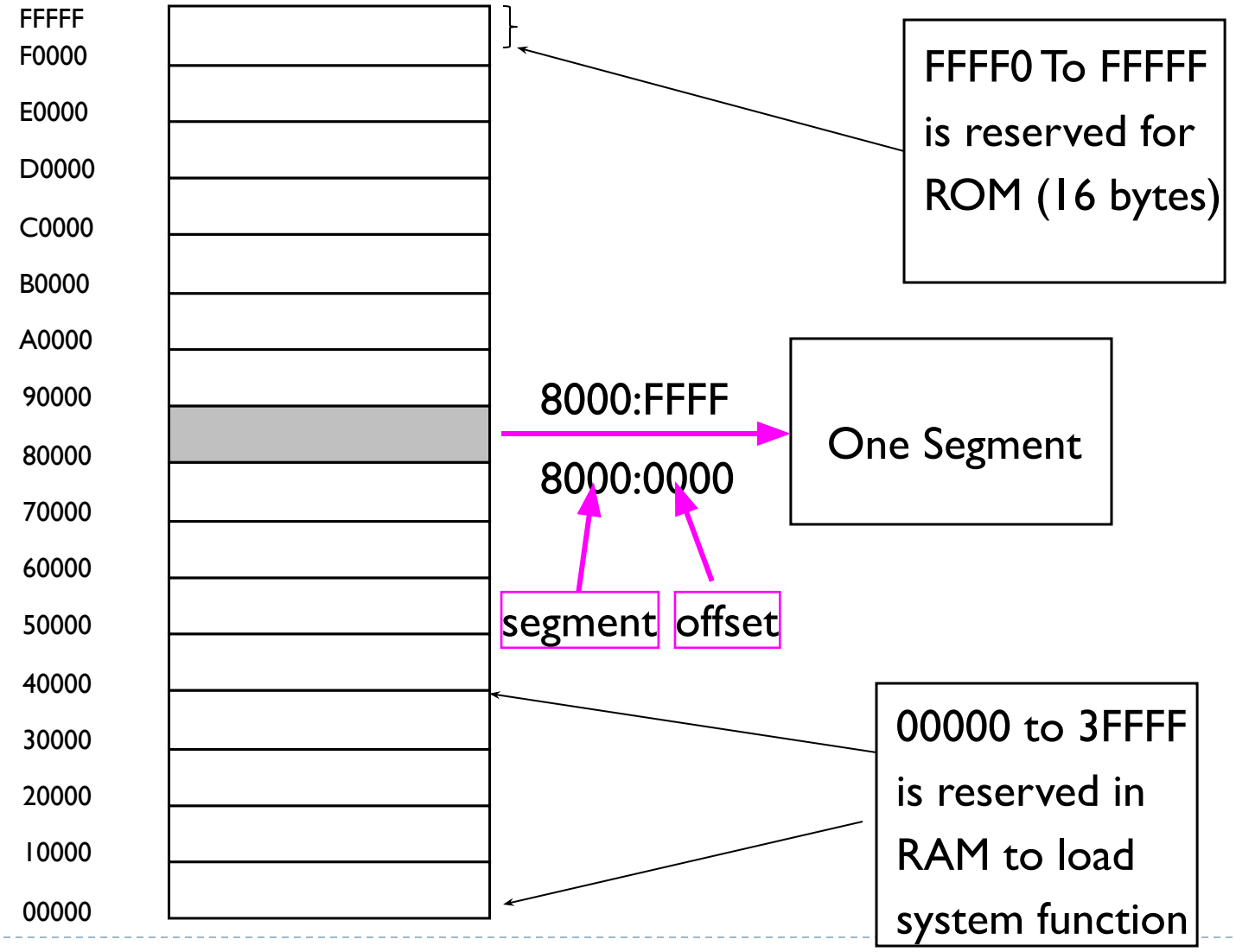
Suppose 20 bits → 5 hex digits: 38A41H

3	8	A	4	1
0011	1000	1010	0100	0001

Here, 0011 1000 1010 0100 0001 = 20 bits = 5 Hex digits

- **Partition the memory into segments**
  - Total of (16), 64 Kbytes segments might be positioned within the 1 Mbyte address space of an 8086 processor.

Memory Segment: ***Address Space***



Memory Segmentation

- **Segmentation** is the process in which the main memory of the computer is logically divided into different segments and each segment has its own base address.
- It is basically used to enhance the speed of execution of the computer system, so that the processor is able to fetch and execute the data from the memory easily and fast.
- A segment is a logical unit of memory that may be up to 64 kilobytes long. Each segment is made up of contiguous memory locations.

Memory Segment: ***Address Space***

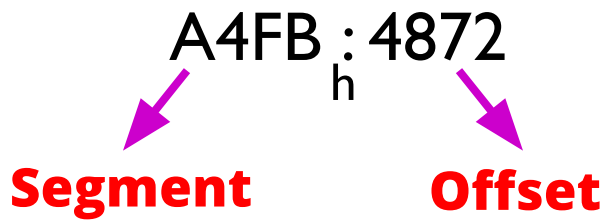
- ***Memory segment*** is a block of 2^{16} (64) KBytes consecutive memory bytes.

- Each segment is identified by a 16-bit number called **segment number**, starting with 0000 up to FFFFh. Segment registers hold segment number.
- Within a segment, a memory location is specified by giving an **offset** (16-bit) = It is the number of bytes from the beginning of the segment (0→ FFFFh).
- The basic difference between **Logical** and **physical address** is that **Logical address** is generated by CPU in perspective of a program whereas the **physical address** is a location that exists in the memory unit.

Memory Segment: ***Address Space***

- A memory location may be specified by a ***segment number*** and ***offset*** (logical address).

  Example :



- **Offset:** is the distance from the beginning to a particular location in the **segment.**

- **Segment Number:** defines the starting of the segment within the memory space.

Segmented Memory Address

Highest Address: A4FB:FFFF = A4FB0
FFFF
B4FAF

A9822 (20 bits)

Offset 4872 (16 bits)

Base address: A4FB
(16 bits)

A4FB0 (20 bits)

Lowest Address: A4FB:0000 = A4FB0

4Hex digits = 4 x 4 = 16 digits
so lowest value can be 0000 and highest value can be FFFF

Segmented Memory Address

- Start location of the segment must be 20 bits the absolute address is obtained by appending a hexadecimal zero (0H = 0000) to the segment number, i.e., **multiplying by 16(10_h).**
  - Adds 4 Nibble bits at the lower portion of each 16-bit address.

Starting location Address = Segment number X 10_h = A4FB0

- So, the **Physical Memory Address** is equal to:

Physical Address = Segment number X 10_h + Offset

- Physical **Address** for **A4FB : 4872**

```
                A4FB0 h
Here, A4FB0  =1010 0100 1111 1011 0000 (16+4=20bits)
        4872 =      0100 1000 0111 0010 (16 bits)
        A9822 =1010 1001 1000 0010 0010 (20 bits)
```

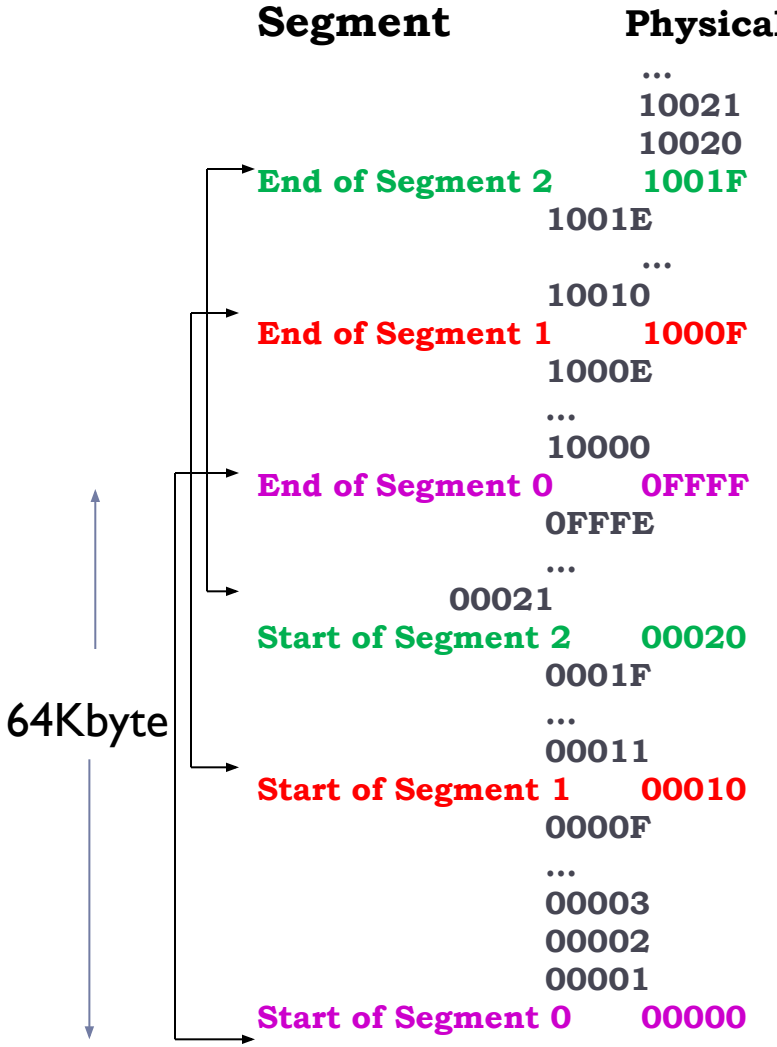
+ 4872 h

A9822 h (20 Bits)

Memory Segmentation

- **Types Of Segmentation –**
- **Overlapping Segment –** A segment starts at a particular address and its maximum size can go up to 64kilobytes. But if another segment starts along with this 64kilobytes location of the first segment, then the two are said to be *Overlapping Segment*.
- **Non-Overlapped Segment –** A segment starts at a particular address and its maximum size can go up to 64kilobytes. But if another segment starts before or after this 64kilobytes location of the first segment, then the two segments are said to be *Non-Overlapped Segment*.

Physical Location of Segments



0001F physical address has Different logical address

For Segment 0,

Base address = 0000

offset = 001F

For Segment 1,

Base address = 0001

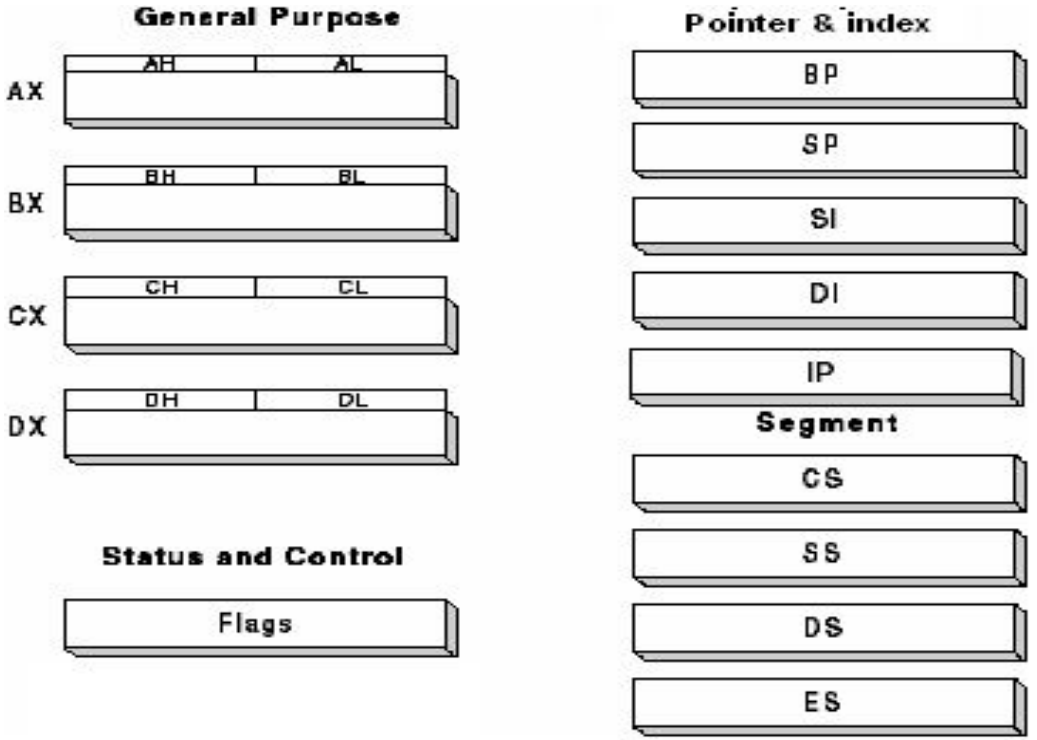
Offset = 000F

Registers:

- **Registers:**
  - **Information is stored in registers**
  - **Registers are classified according to the functions they perform**
  - **All Processors have internal registers some are visible and some are not visible to the programmers.**
  - **Internal Registers are used as temporary storage for operands, and if the operand is already in memory, it takes less time for execution of the associated instruction.**

8086 Register Categories

- **Data registers (4)**: General data registers hold data for an operation (AX, BX, CX, DX).
- **Address registers (9)**: (Segment, Pointer and Index registers) hold the address of an instruction or data.
- **Status register (1)**: FLAG register keeps the current states of the processor.



8086 Registers and Memory

Number of Registers: 14, each of that16-bit registers

Memory Size: 1M Bytes

Registers of the 8086/80286 by Category

Category	Bits	Register Names
General	16	AX,BX,CX,DX
	8	AH,AL,BH,BL,CH,CL,DH,DL
Pointer	16	SP (Stack Pointer), Base Pointer (BP)
Index	16	SI (Source Index), DI (Destination Index)
Segment	16	CS(Code Segment) DS (Data Segment) SS (Stack Segment) ES (Extra Segment)
Instruction	16	IP (Instruction Pointer)
Flag	16	FR (Flag Register)

Memory Segment and Segment Registers

- Note that, 8086 does not work with the whole 1MB memory at a given time, it works only with four 64KB segments within the whole memory, namely
  - **Code segment CS: holds segment number of the code segment.**
  - **Data Segment DS: holds segment number of the data segment.**
  - **Extra Segment ES: extra segment : holds alternate segment number of the data segment.**
  - **Stack Segment SS: holds segment number of the stack segment and used when sub-program executes.**
- Codes , data , and stack are loaded into different memory segments (registers).

General Data Registers

- These are 16-bit registers and can also be used as two 8 bit registers: ***low and high bytes*** can be accessed separately
- **AX (Accumulator)**
  - Can be partitioned into AH, AL
  - Most efficient register for arithmetic, logic operations and data transfer: the use of AX generates the shortest machine code.
  - In multiplication and division operations, one of the numbers involved must be in AL or AX
- **BX (Base)**
  - The base address register (offset)
- **CX (Counter)**
  - Counter for looping operations: loop counter, in REP instruction, and in the shift and rotate bits
- **DX (Data)**
  - Used in multiply and divide, also used in I/O operations

Pointer and Index Registers

- **Used for offset of data, often used as pointers. Unlike segment registers, they can be used in arithmetic and other operations.**
- **SP (Stack Pointer):**
  - Used with SS for accessing the stack segment.
  - Holds **Offset** address relative to SS
  - Always points to word (byte at even address)
  - An empty stack will had SP = FFFEh
- **BP (Base Pointer):**
  - Used with SS to access data on the stack. However, unlike SP, BP can be used to access data in other segments.
  - Primarily used to access parameters passed via the stack
  - Holds **Offset** address relative to SS

Pointer and Index Registers

- **SI (Source Index):**
  - Source of string operations. Used with DS (or ES).
  - Can be used for pointer addressing of data with effective address (EA)
  - Used as source in some string processing instructions
  - Offset address relative to DS
- **DI (Destination Index):**
  - Destination of string operation. Used with ES (or DS).
  - Can be used for pointer addressing of data
  - Used as destination in some string processing instructions
  - Offset address relative to ES
- **IP (Instruction pointer):**
  - Points to the next instruction.
  - Offset address relative to CS

Memory Segmentation

- **Rules of the Segmentation:** Segmentation process follows some rules:
- The starting address of a segment should be such that it can be evenly divided by 16.
- Minimum size of a segment can be 16 bytes and the maximum can be 64 kB.

Segment	Offset Registers	Function
CS	IP	Address of the next instruction
DS	BX, DI, SI	Address of data
SS	SP, BP	Addresses in the stack
ES	BX, DI, SI	Address of destination data (for string instructions)

Memory Segmentation

- **Advantages of the Segmentation** The main advantages of segmentation are as follows:
  - It provides a powerful memory management mechanism.
  - Data related or stack related operations can be performed in different segments.
  - Code related operation can be done in separate code segments.
  - It allows to processes to easily share data.
  - It allows to extend the address ability of the processor, i.e. segmentation allows the use of 16 bit registers to give an addressing capability of 1 Megabytes. Without segmentation, it would require 20 bit registers.
  - It is possible to enhance the memory size of code, data or stack segments beyond 64 KB by allotting more than one segment for each area.

Task

- Q1. Suppose, segment no = 1111 H, offset = 1332 H, calculate the physical address?

Q2. Suppose, physical address = 33330, offset = 0020, calculate the segment no?

Thank You !!