Welcome to the World of Assembly Language Programming

Prepared by
Madhusudan Basak
Lecturer
Department of CSE, BUET

Course Outline

Week No.	Tentative Coursework
Week 1	Class on Chapter 1,2,3
Week 2	Class on Chapter 4,5
Week 3	Class on Chapter 6+ Code Demo + Assigning Offline
Week 4	Evaluation of offline + Online
Week 5	Class on Chapter 7, 9 + Assigning Offline
Week 6	Evaluation of Offline + Online
Week 7	Class on Chapter 8, 17+ Assigning Offline
Week 8	Evaluation of Offline + Online
Week 9	Class on Chapter 10, 11+ Assigning Offline
Week 10	Evaluation of Offline + Online
Week 11	Class on Graphics + Project Allocation
Week 12	Reserved Class
Week 13	Quiz
Week 14	Project Checking + Evaluation by Giving Online

References

- **Assembly Language Programming and Organization of the IBM PC**

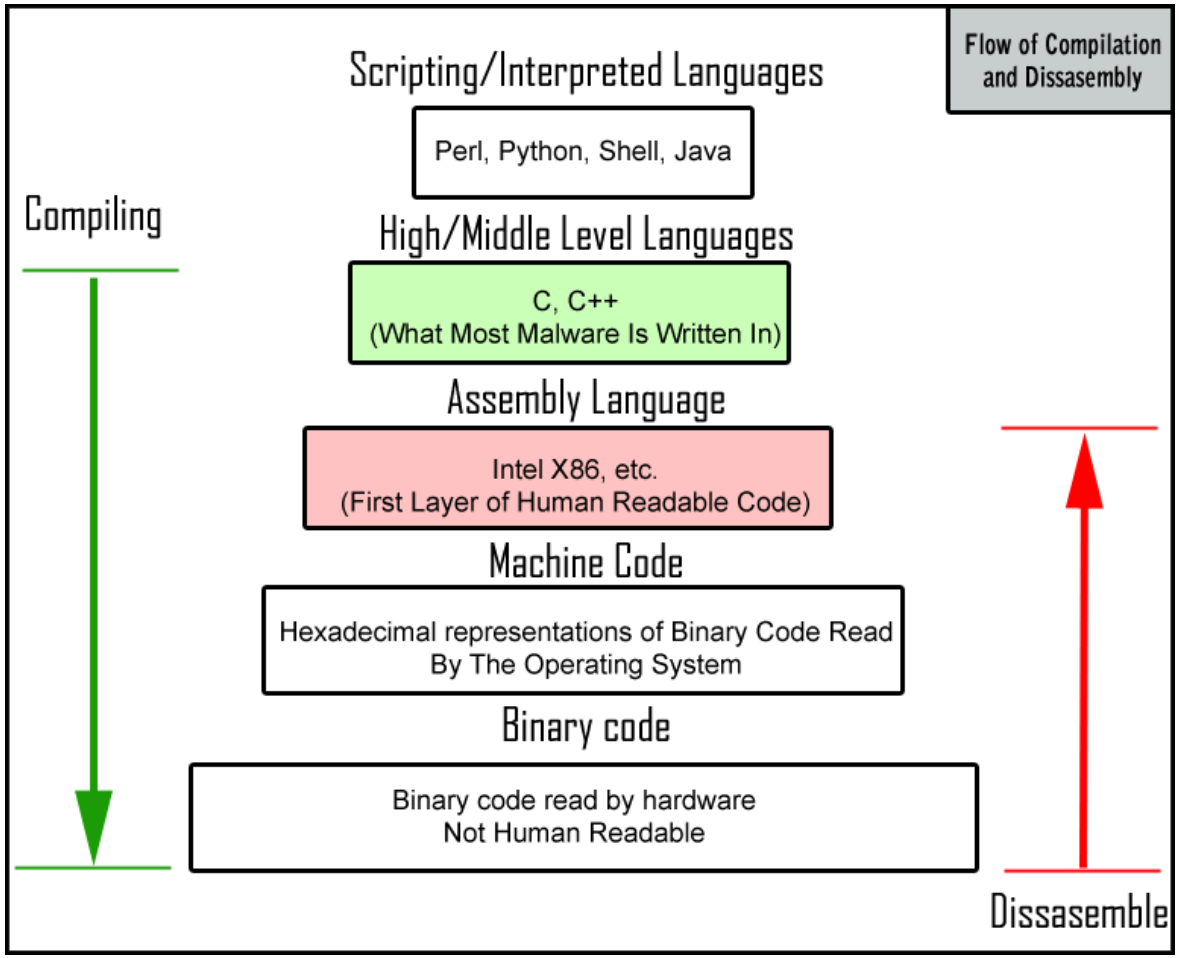
  --- Ytha Yu and Charles Marut

- Assembly Language for the IBM-PC

  --- Kip R. Irvine

The Beginning !

- What is Assembly Language?


Flow of Compilation and Dissasembly
Scripting/Interpreted Languages
Perl, Python, Shell, Java
Compiling
High/Middle Level Languages
C, C++
(What Most Malware Is Written In)
Assembly Language
Intel X86, etc.
(First Layer of Human Readable Code)
Machine Code
Hexadecimal representations of Binary Code Read By The Operating System
Binary code
Binary code read by hardware
Not Human Readable
Dissasemble

About Assembly

- Assembly is human-readable machine code
- Gives you complete control over the system's resources

Why use it ?

- Direct hardware manipulation
- Access to specialized processor instructions
- Performance and efficiency
- Speed optimization
- Control the code behavior

Disadvantages ☹

- Hard and tedious
- Bug-prone
- Non-portable(machine dependent)
- Difficult to debug

A question ?

- In what kind of situations, assembly language will be the only solution?

- An Operating System
- Drivers and communication with custom hardware/electronics.
- An Compiler
- Optimizations

Definition from wiki

- An **assembly language** is a low-level programming language for a computer, or other programmable device, in which there is a very strong (generally one-to-one) correspondence between the language and the architecture's machine code instructions. Each assembly language is specific to a particular computer architecture, in contrast to most high-level programming languages, which are generally portable across multiple architectures, but require interpreting or compiling.

Chapter 1

Microcomputer Systems

The Components of a Microcomputer System

- Typical components
  - System unit
  - Keyboard
  - Display Screen
  - Disk drives
- Functionally three parts
  - CPU
  - Memory circuits
  - I/O circuits

Memory

- Bytes and Words
- Bit position
- Memory operations
- RAM and ROM
- Buses

CPU

- Two main components
  - Execution Unit(EU)
  - Bus Interface Unit(BIU)

I/O Ports

- Serial Ports
- Parallel Ports

Instruction Execution

- Fetch
- Execute

Instruction Execution

- The Process of executing an instruction
  - 00000001 00000110 00000000 00000000

ADD operation

AX Register

Memory Address 00000000

Memory Address 00000000

I/O Devices

- Magnetic Disks
- Keyboard
- Display Monitor
- Printers

Programming Languages

- Machine Language
- Assembly Language
- High-Level Languages

Chapter 2

REPRESENTATION OF NUMBERS AND CHARACTERS

Number Systems

- Binary Number System
- Hexadecimal Number System
- Decimal Number System

Conversion Between Number Systems

- Converting Binary and Hex to Decimal
- Converting Decimal to Binary and Hex
- Conversions between Hex and Binary

Addition and Subtraction

- Hexadecimal Addition Table

How integers are represented in the Computer

- Unsigned integers
- Signed integers
  - One's complement
  - Two's complement
- Decimal Interpretation

Character Representation

- ASCII (American Standard Code for Information Interchange) code
- Normally 128 ASCII Codes
- 32-126 are printable
- 0 to 31 and 127 for control purposes
- For Extended set, there are 256 characters

Chapter 3

ORGANIZATION OF THE IBM PERSONAL COMPUTERS

The Intel 8086 Family of Microprocessors

- The 8086 and 8088 Microprocessors
- The 80186 and 80188 Microprocessors
- The 80286 Microprocessor
- The 80386 and 80386SX Microprocessors
- The 80486 and 80486SX Microprocessors

Organization of the 8086/8088 Microprocessors

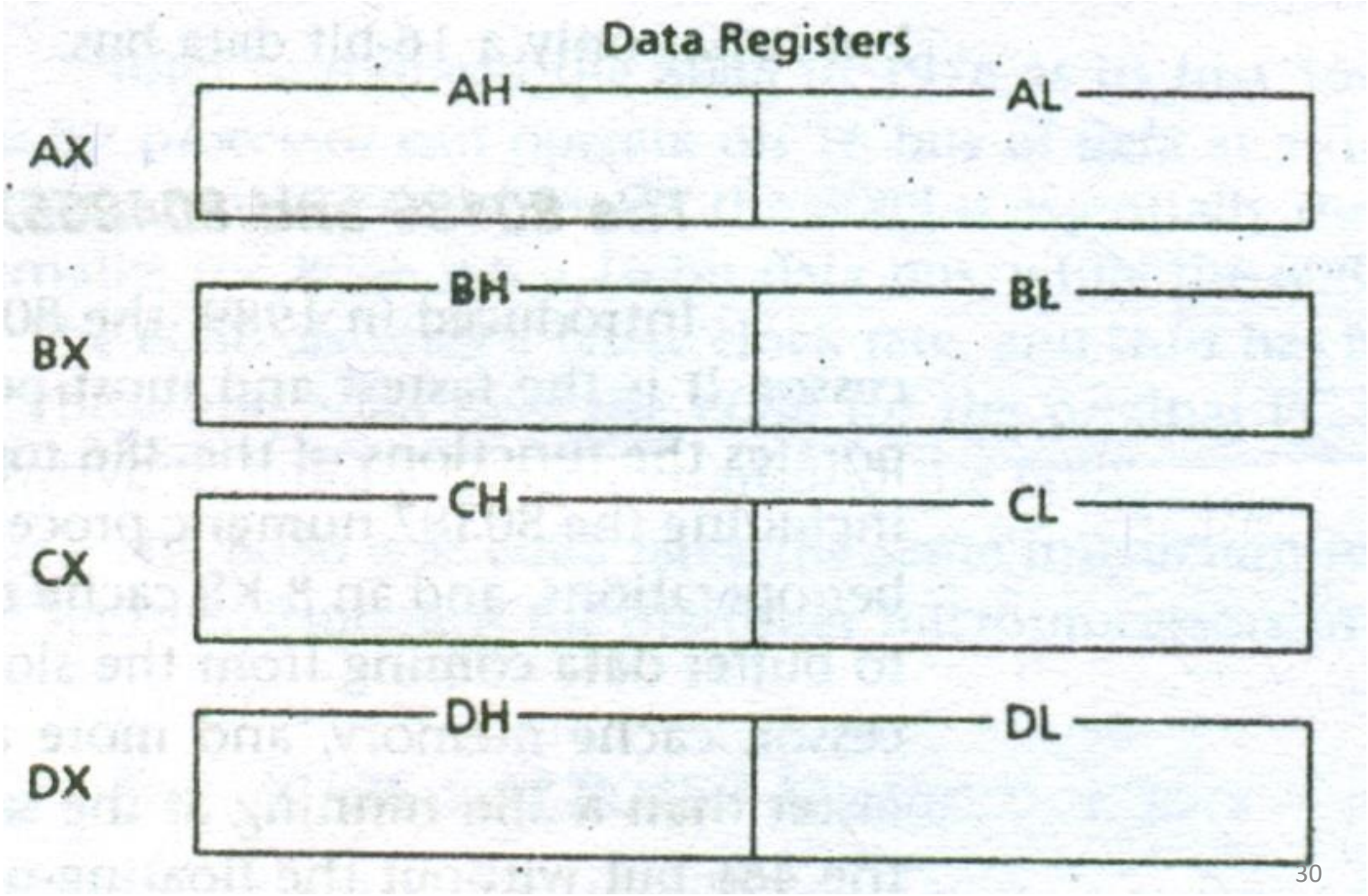
- Registers
  - 3 Types of registers
    - Data Registers
    - Address Registers
    - Status Register

Data Registers

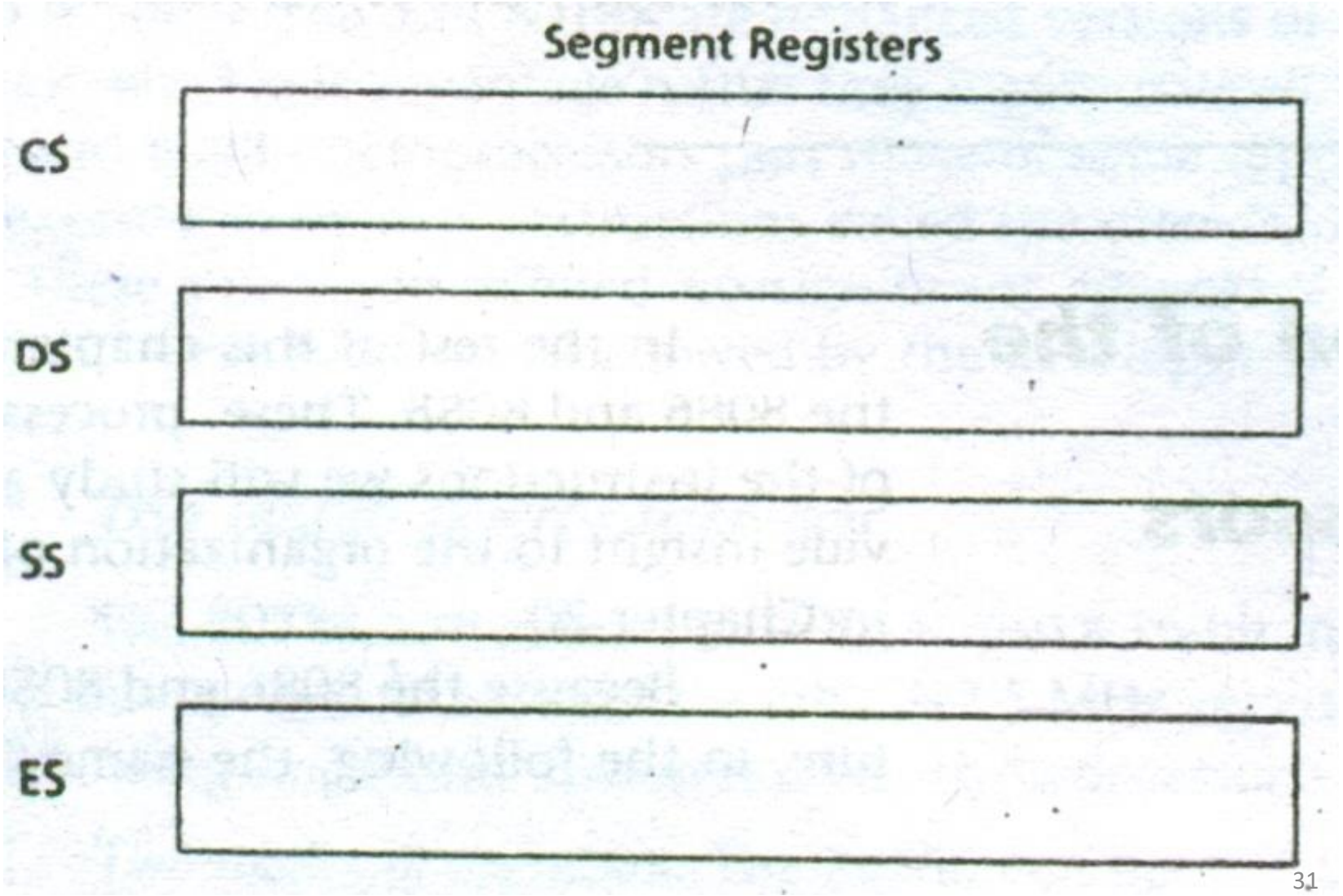
- AX (Accumulator Register)
- BX (Base Register)
- CX (Count Register)
- DX (Data Register)

- Each one has two parts like AX(AH+AL)

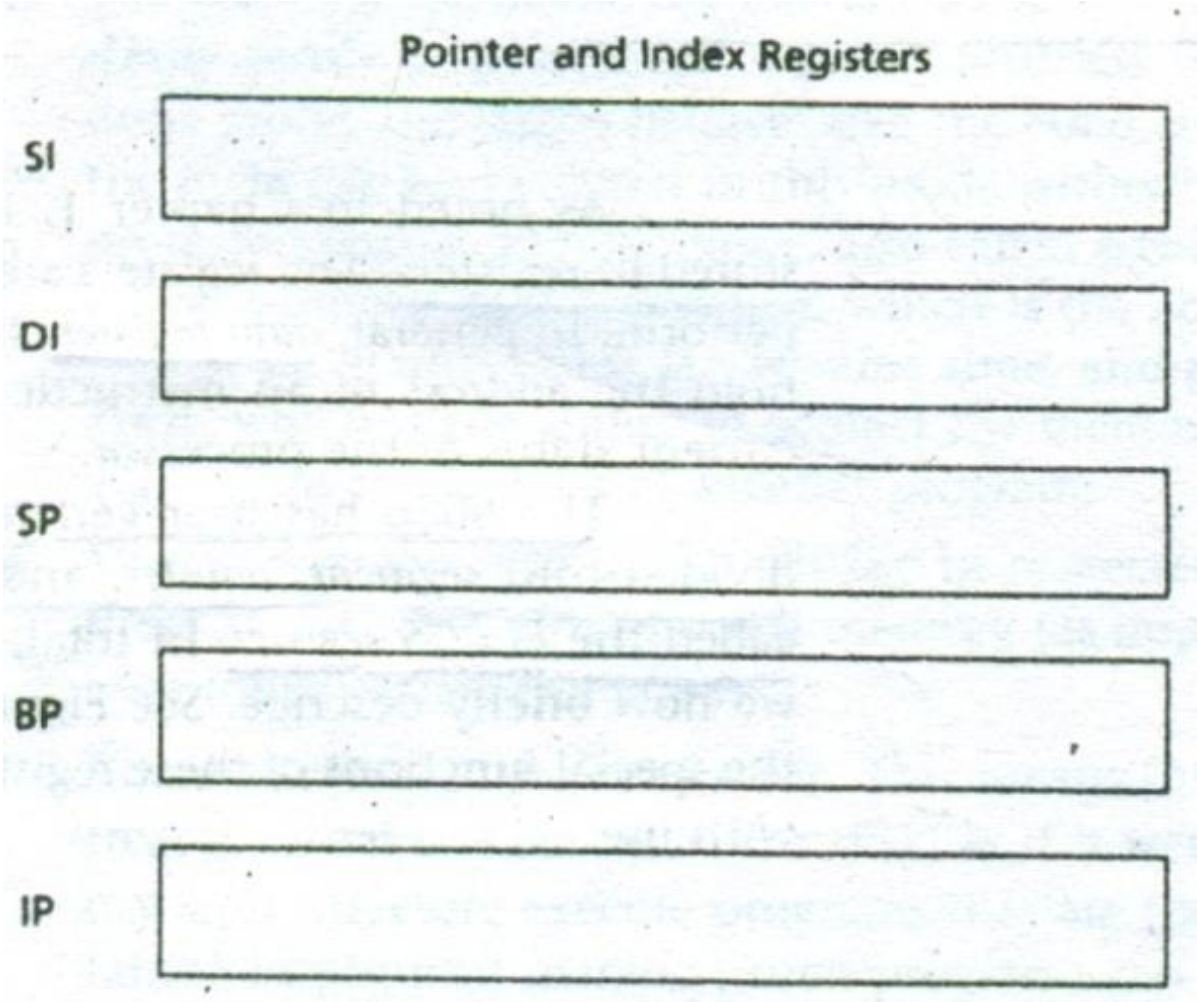
Data Registers



Segment Registers(Address Registers)



Pointer and Index Registers, Instruction pointers (Address Registers)

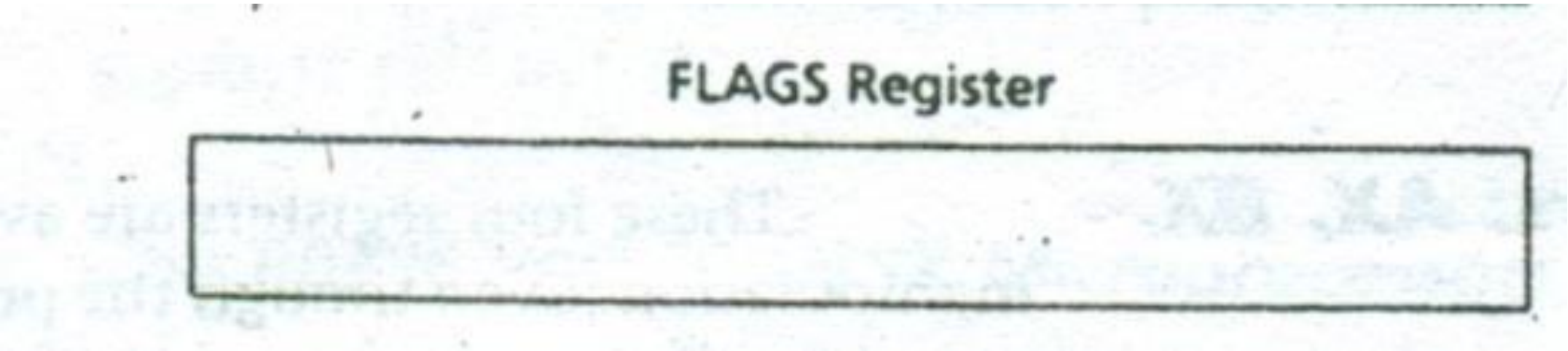


Segment Registers

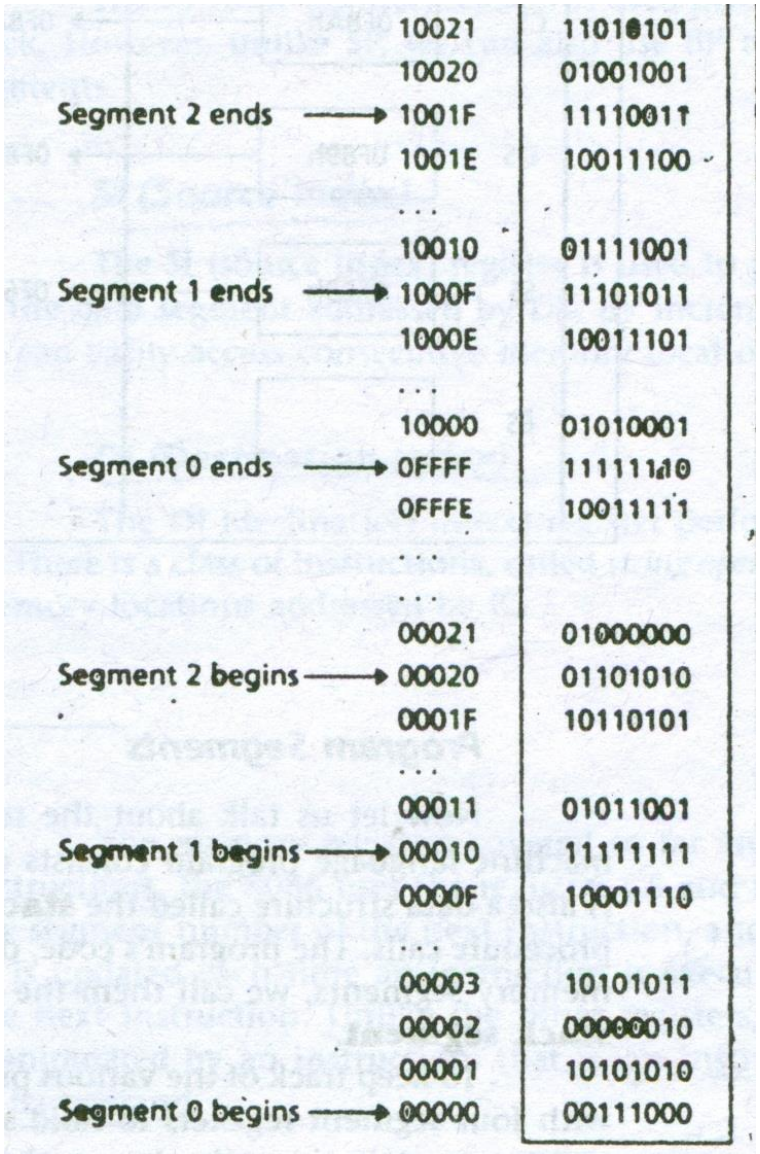
- Memory Segment
- Segment: Offset Address
- Location of Segments

Task	Segment Registers	Pointer and Index Registers
Code	CS	IP
Data	DS	SI
	ES	EI
Stack	SS	SP
		BP

FLAGS Register(Status Register)



Overlapping Scenario



- If you have a segment A4FBh and offset 4872h. Then what is the 20 bit physical address?

A4FB0h

+4872h

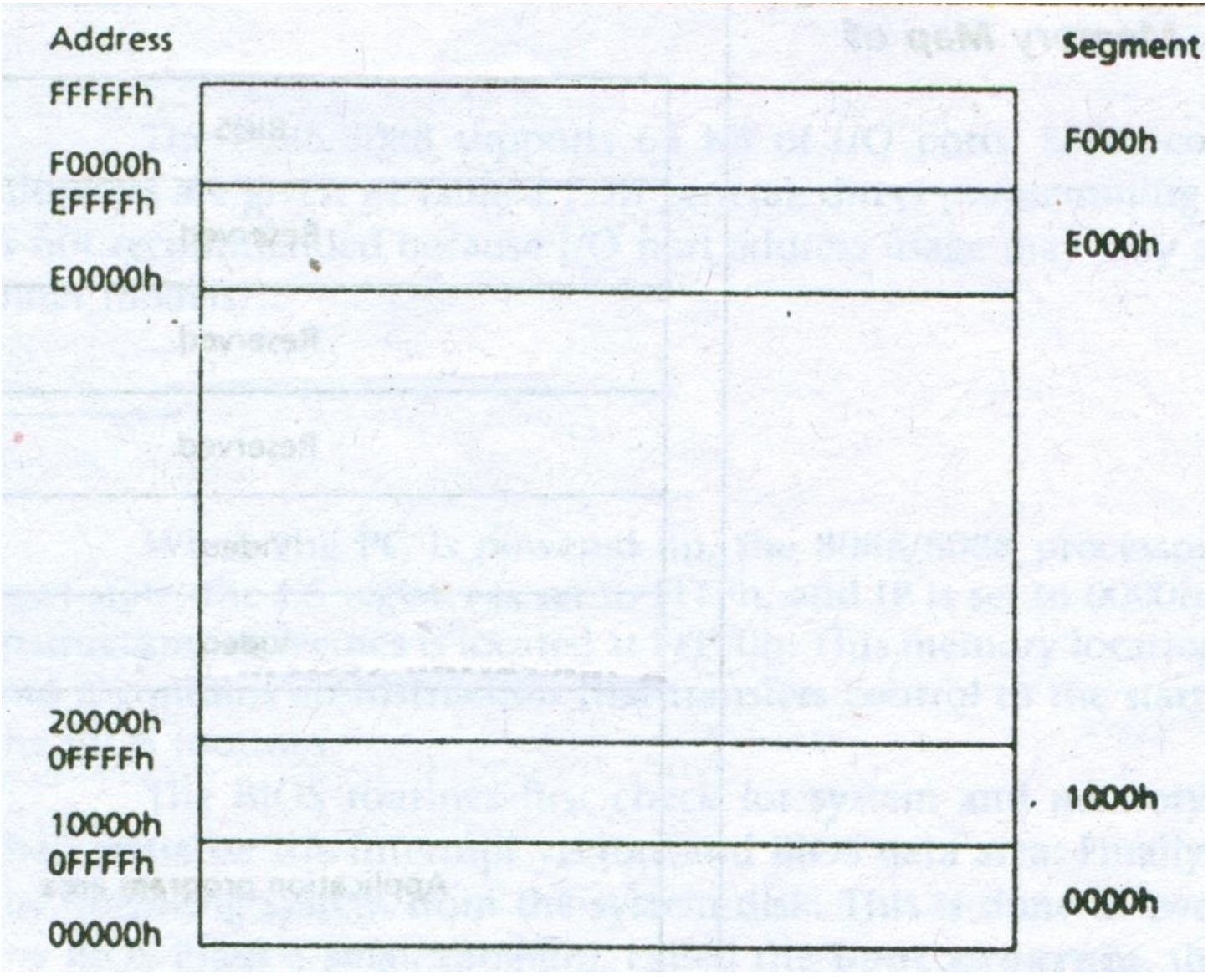
A9822h (20 bit physical address)

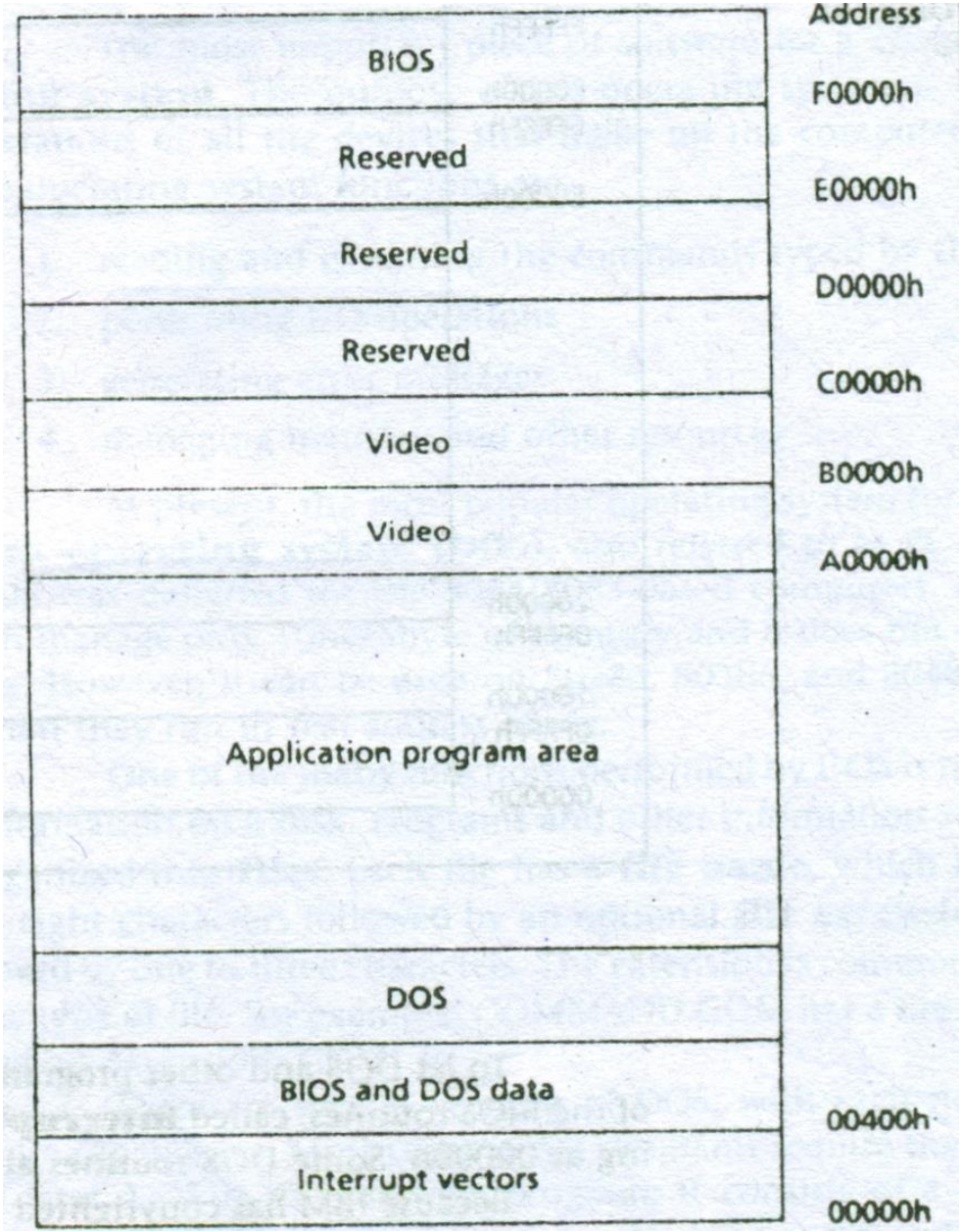
Organization of the PC

- The Operating System
  - DOS (Disk Operating System)
- BIOS (Basic Input/Output System)

Memory Organization of the PC

- 8086/8088 has only 1MB of memory
- Not all memory for application programs
- Interrupt Vector, Video Display Memory etc are needed
- IBM fixed all the positions and allowed all to live happily


Address
Segment
FFFFFh
F0000h
F000h
EFFFFh
E0000h
E000h
20000h
0FFFFh
10000h
1000h
0FFFFh
00000h
0000h


Address
BIOS
F0000h
Reserved
E0000h
Reserved
D0000h
Reserved
C0000h
Video
B0000h
Video
A0000h
Application program area
DOS
BIOS and DOS data
00400h
Interrupt vectors
00000h

I/O ports addresses

- Interrupt controller (20h-21h)
- Keyboard controller (60h-63h)
- etc...

Start-up Operation

- Boot program


Thank you!