CT215-2
Software Organization

Objective

- **Operating System**
- **Boot Process**
- **Input-Output Interface**
- **System Program Loader**
- **Stack**

Features of the Operating System

The operating system manages the hardware resources

- **file management - maintain directories and files**
- **input/output - send or request data by means of interrupts**
- **program loading - placing the program to be executed from disk in the memory and initialize**
- **memory management - allocates space for the program and its data**
- **interrupt handling - allows user's programs attached to the interrupt system to perform special functions**

Organization of the OS

Three major components of MS-DOS (PC-DOS)

IO.SYS (IBMBIO.COM)

- **Initialization functions at boot up time**
- **I/O functions & device drivers**

MSDOS.SYS (IBMDOS.COM)

- **System kernel: file management, memory management, and I/O**

COMMAND.COM

- **command interpreter (shell or interface)**

The Boot Process

Cold boot:

Processor enters

- **a reset state**
- **clear all memory locations to zero**
- **parity check of memory**
- **set the starting address CS:IP at address FFFF0H (the entry point to BIOS in ROM)**

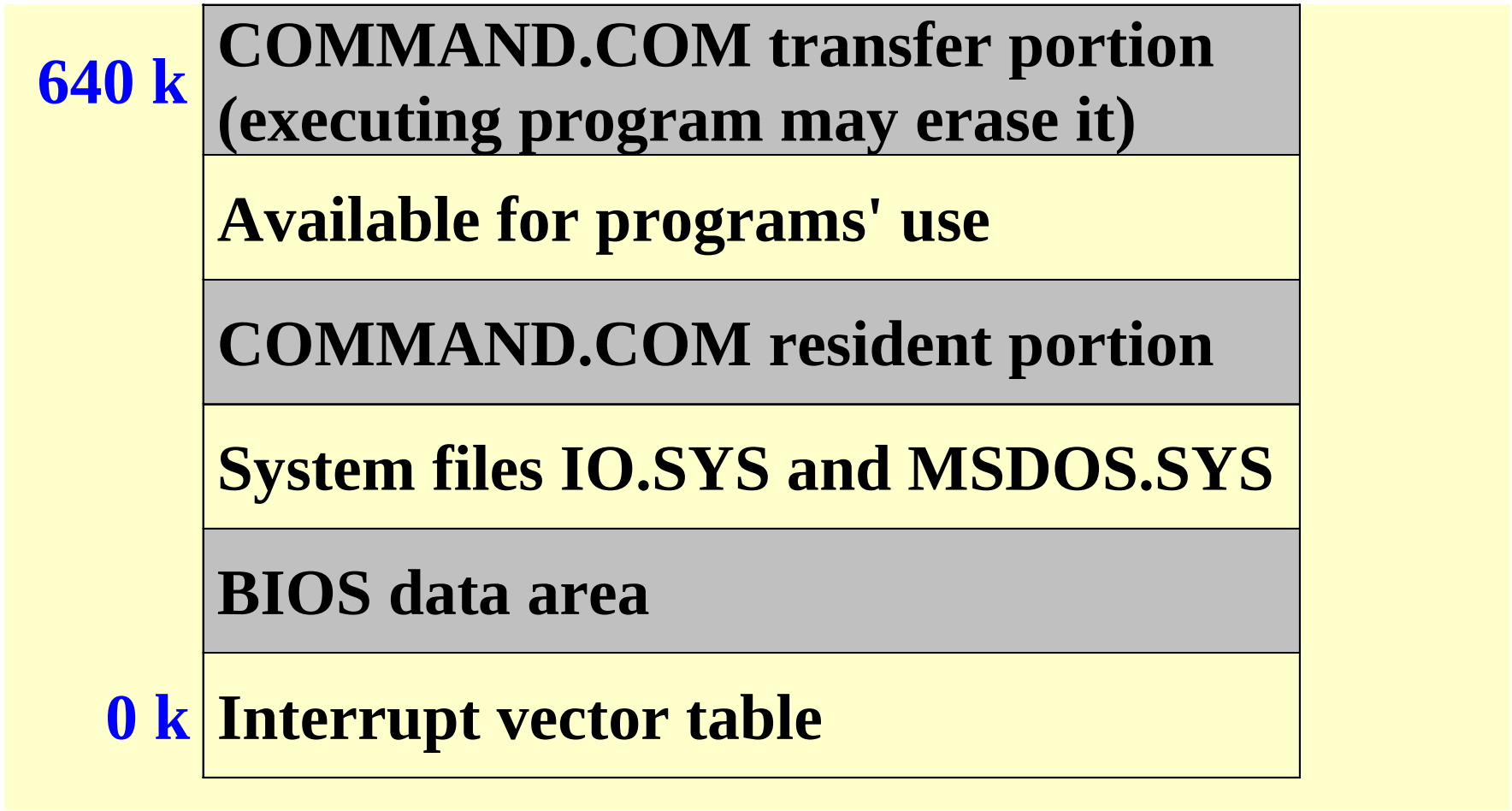
The Boot Process

Cold boot:

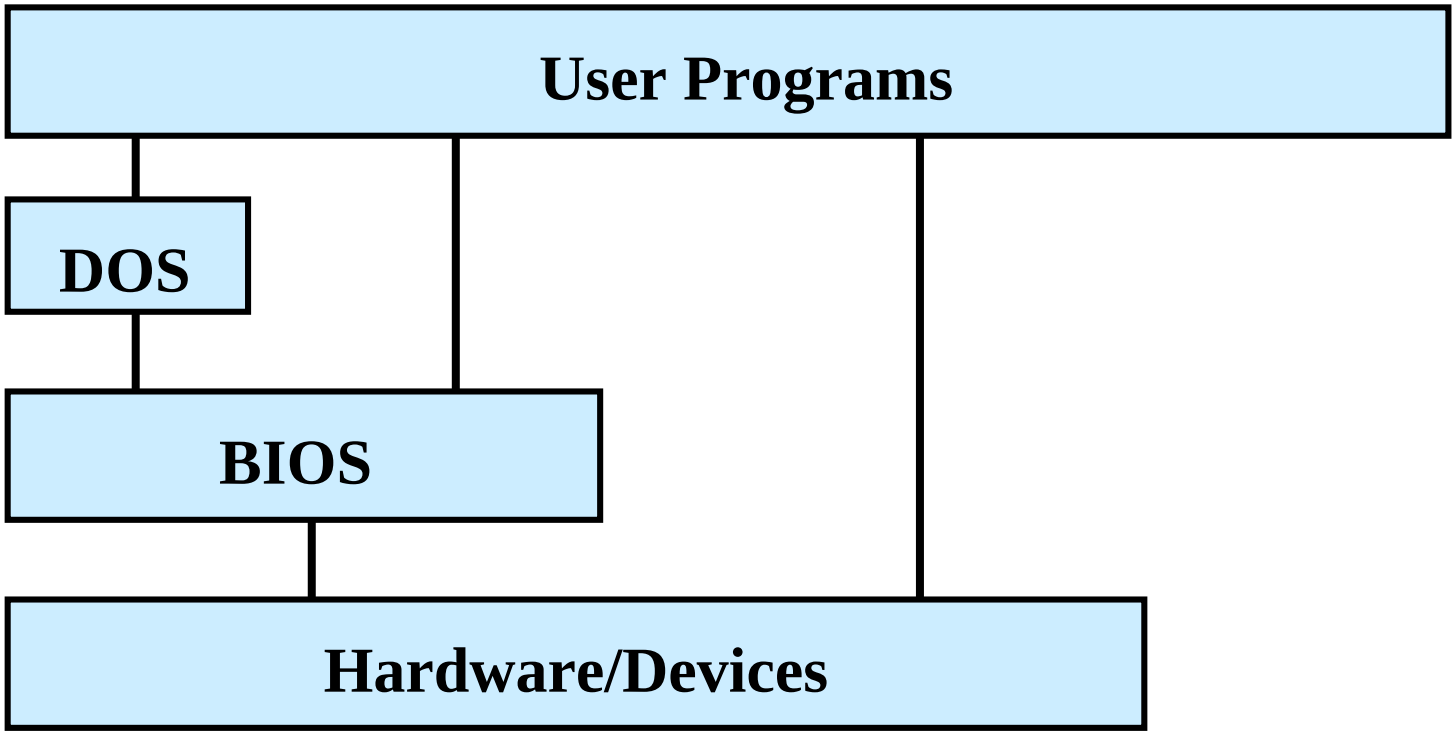
BIOS

- **checks various I/O ports to identify and initialized devices attached to the computer**
- **creates 2 data areas - Interrupt vector table and BIOS data area**
- **accesses the Bootstrap loader from disk that loads system files (IO.SYS, MSDOS.SYS, COMMAND.COM) into memory**

The Boot Process



Input-output Interface



Machine language Programs

.COM vs .EXE programs

.COM

- **1 segment that includes code, data, and stack**
- **used for small programs (64KB max)**
- **utility or resident program**

.EXE

- **all other programs**
- **separate code, data, and stack segments**

System program loader

- access .EXE file from disk
- construct a 256-byte (100H) program segment prefix (PSP)
- store the program in memory after PSP
- load address of PSP in DS and ES
- set CS and IP registers -- IP generally initialized 0
- set SS and SP registers
- transfer control to execute the instruction at CS:IP

The Stack

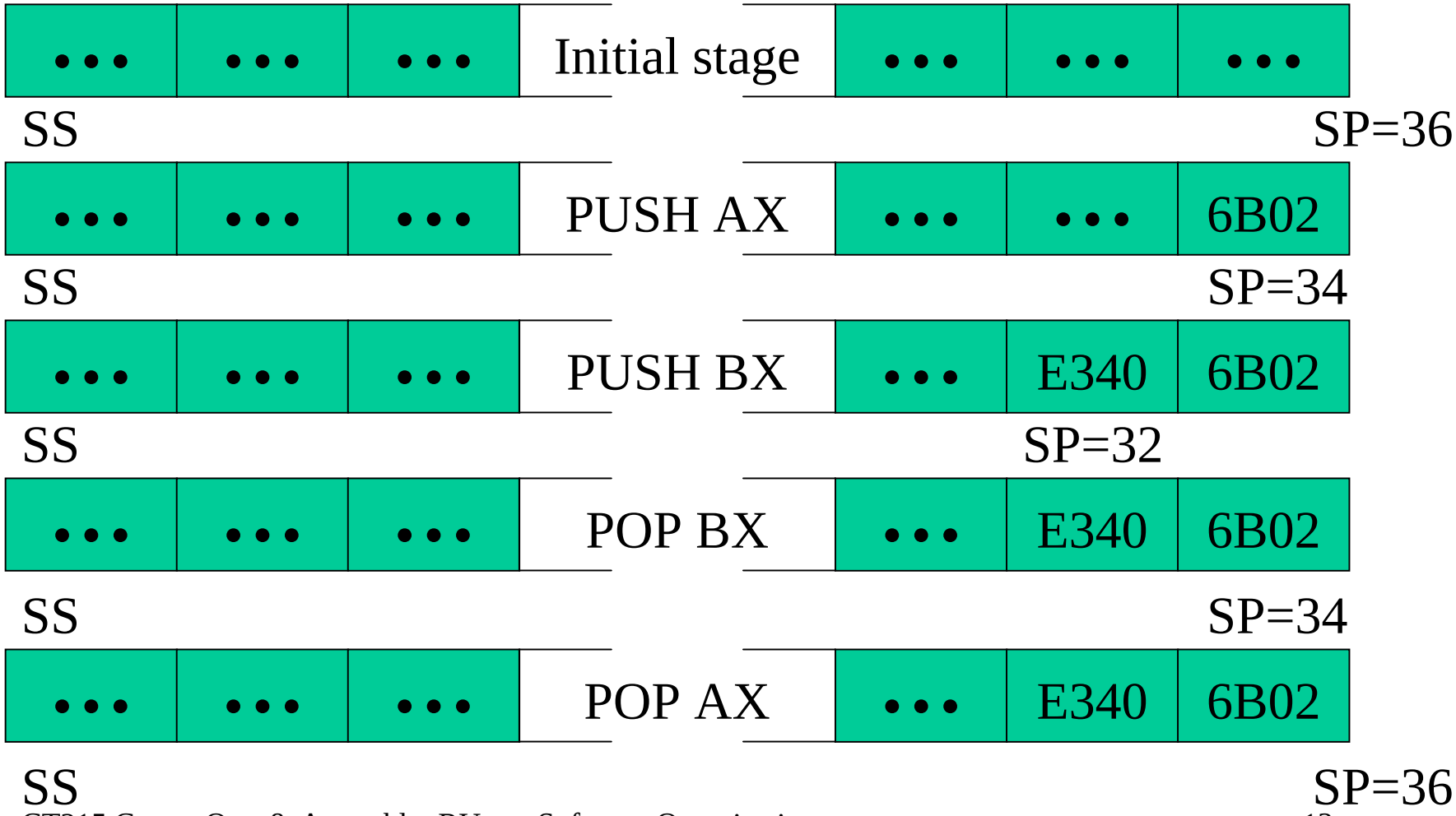
- needed for both .COM and .EXE programs
- temporary storage of addresses for data items
- the program loader defines the stack for .COM program
- user must specify a stack for .EXE program
- each data item is one word (2 bytes)
- SS register contains beginning address of the stack
- SP register contains the value that points to the byte past the end of the stack -- stack size

Stack Operations

PUSH and POP instructions

- modify the contents of the SP register
- store and retrieve data on the stack
- PUSH -- decrement SP by 2
- POP -- increment SP by 2
- PUSHF and POPF -- save and restore the status flags register
- PUSHA and POPA -- save and restore the contents of all general-purpose registers (386 & later)

Example


• • •
• • •
• • •
Initial stage
• • •
• • •
• • •
SS
SP=36
• • •
• • •
• • •
PUSH AX
• • •
• • •
6B02
SS
SP=34
• • •
• • •
• • •
PUSH BX
• • •
E340
6B02
SS
SP=32
• • •
• • •
• • •
POP BX
• • •
E340
6B02
SS
SP=34
• • •
• • •
• • •
POP AX
• • •
E340
6B02
SS
SP=36

Addressing Instructions and Data

An instruction consists of

- at least one operation -- ADD, MOVE, AND
- zero, one or more operands to reference the data
- Generally, the first operand is the destination

example:

```
MOV  AX,25      ; Immediate operand
MOV  BX,AX      ; register to register
MOV  BX,[AX]    ; indexed addressing
```

Addressing Instructions and Data

- CS register contains the address of the beginning of a program's code segment which contains instructions
- DS register contains the address of the beginning of a program's data segment which contains data that instructions reference
- IP register indicates the offset address of the current instruction in the code segment to be executed
- Instruction operand indicates an offset address in the data segment to be referenced

Example

The program loader loads .EXE program into memory at 5BE0H (05BE[0]H)

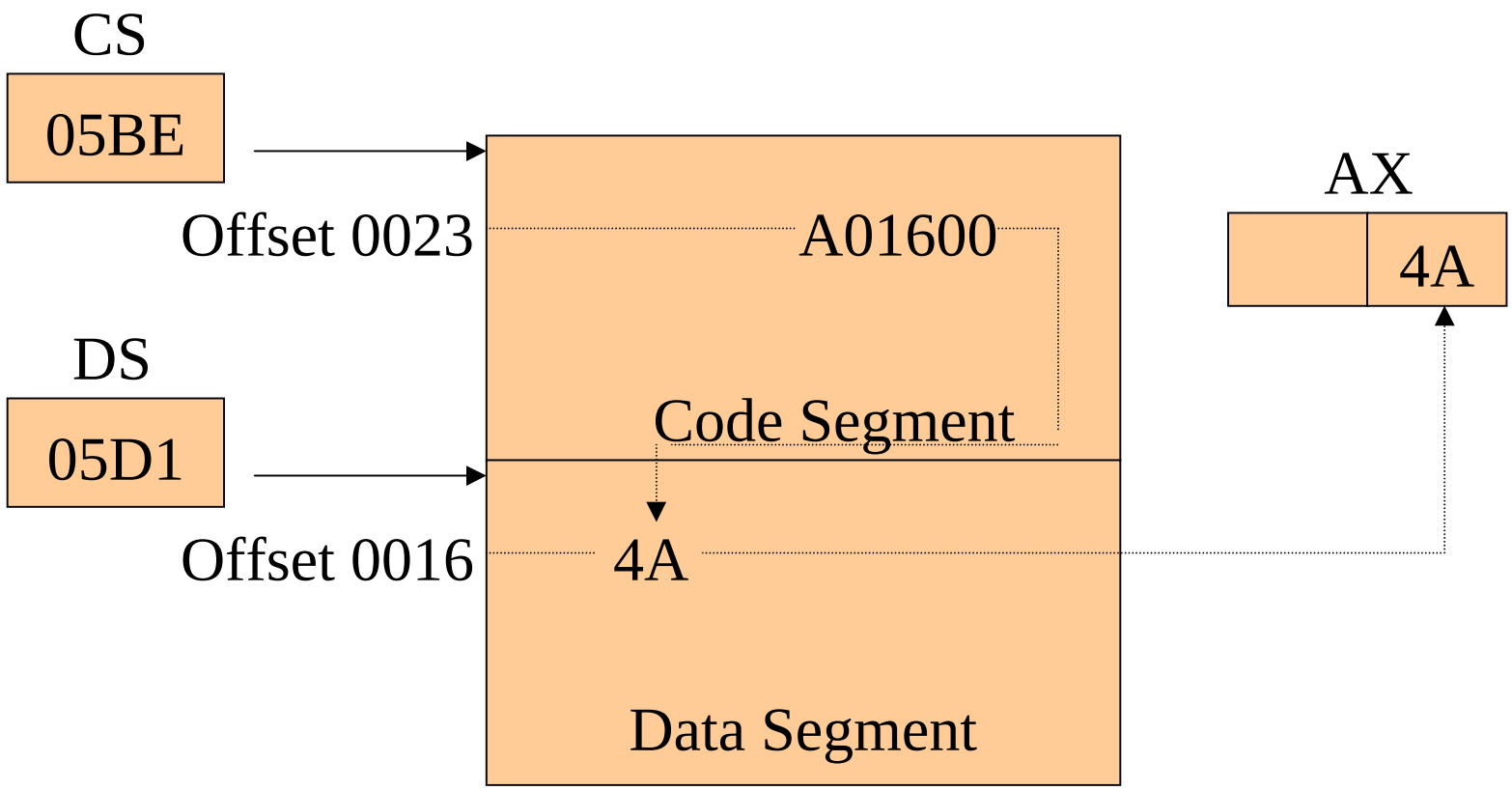
Segment address in CS	5BE0H
Offset address in IP	+0023H
Address of next instruction	5C03H

Assume that the instruction in 05C03H copies the contents of one byte at offset 0016 in memory to AL

```
A01600   MOV AL,[0016] ; indexed addressing
```

DS segment address	5D10H
Segment offset	+0016H
Address of data item	5D26H

Example


CS
05BE
Offset 0023
DS
05D1
Offset 0016
A01600
Code Segment
4A
Data Segment
AX
4A

Addressing More Than One Byte

```
MOV AX,35F3H
MOV [1500],AX
```

These instructions copy the contents of AX (i.e. 35F3H) into 2 memory locations starting at address DS:1500.

Contents of bytes:	F3	35
Offset in data segment:	1500	1501

Addressing Data in Memory

Assume DS = 3146[0]H

=> DS:1500 = 32960H

Memory

Instruction Operands

```
X1 DW   0       ; Define X1 as a word
...
MOV CX,X1       ; Move X1 to CX
MOV CX,25       ; Move 25 to CX
MOV CX,DX       ; Move contents of DX to CX
MOV CX,[DX]     ; Move contents of location addressed
                  by DX
```

the actual address of the [DX] is DS:DX -- DS +DX