第2章 80x86计算机组织

主讲：许万茹

本章教学内容

- 80X86 CPU的基本组成和寄存器作用
- 标志寄存器各位的含义
- 实模式下存储器的分段、20位物理地址的形成
- 段地址寄存器和偏移地址寄存器之间的隐含规则

主要内容

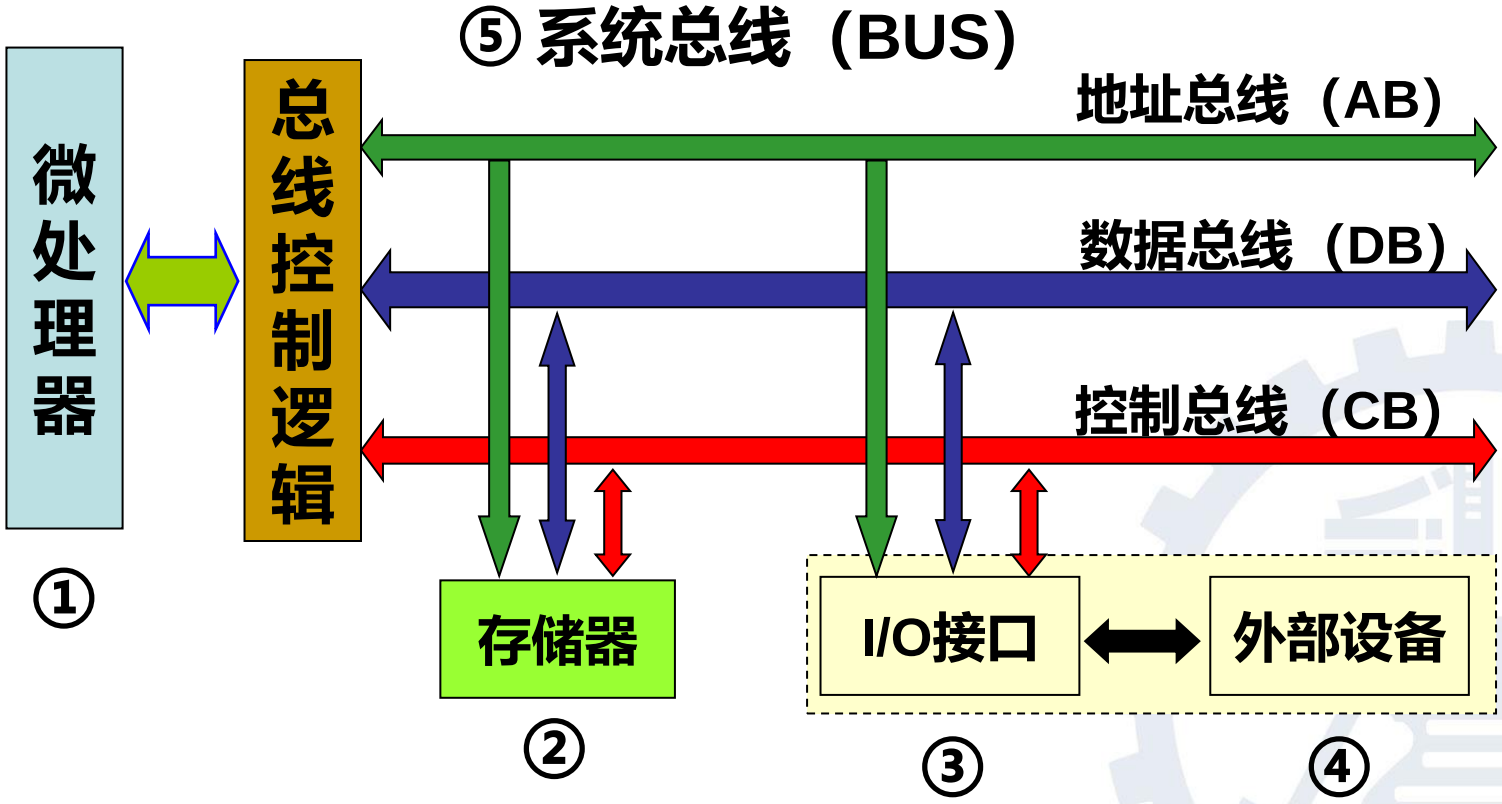
2.1 计算机系统

2.2 80x86微处理器

2.3 存储器地址

2.1 计算机系统

以16位的8086为教学对象



- 微处理器是整个微型计算机的**核心部件和总控制单元**，决定了微机的基本性能；微处理器就是把**运算器、寄存器组和控制器**集成在一个超大规模集成电路芯片上的功能部件，是具有计算机系统运算和控制功能的中央处理单元（Central Processing Unit，CPU）。
- 存储器是**微机存放和记忆程序、数据**的重要部件。通常把位于主机内部用于暂时存放程序和数据的存储器称为**内存**，也称为主存，由只读存储器（ROM）和随机存储器（RAM）两部分组成，属于主板（焊有所有微机芯片的印刷电路板，即PCB）内的存储器，而主板外或主机外部的用于存放大量信息的存储器则称为**外存**。

- 输入/输出接口电路是设置在外部设备（简称外设）与系统总线之间的专门电路，又称I/O接口，用于协调CPU与外设之间的信息和数据的交换。
- 外部设备，即I/O设备，可以分成两个部分，一部分是指直接为计算机本身的顺利运行而服务的，必须与外界及时进行信息与数据交换的输入和输出设备，如键盘、鼠标器、扫描仪、图形或文字识别仪、条码读入器、模/数和数/模转换器等；另一部分是暂时不会影响计算机运行的外设，如打印机、外部存储器（如U盘、光盘等）、开环控制对象（如步进电机）、大屏幕LED显示器、VGA显示器等。

- **系统总线包括数据总线、地址总线和控制总线，简称三总线。**系统总线把CPU、存储器和输入/输出接口电路有机地连接起来，用来传送微机内部各部件间的信息；整个系统总线的工作由**系统总线控制模块**统一安排和指挥。
  - 数据总线传送各类数据，其中包括指令代码、运算数据、中间数据和结果数据；
  - 地址总线上的信息主要是针对存储器具体存储单元或具体外设对象的地址信息，这些地址信息指出了数据的来源或传送的目的地；
  - 控制总线传送CPU对存储器或I/O设备的控制命令和I/O设备对CPU的请求服务的信号。

主要内容

2.1 计算机系统

2.2 8086微处理器

2.3 存储器地址

1. 8086 的基本性能指标

- 16位微处理器
- 16根数据线、20根地址线，可寻址的地址空间达1MB (2^{20}Byte = 1MB)
- 可以和浮点运算器、I/O处理器或其他处理器组成多处理器系统，从而提高系统的数据吞吐能力和数据处理能力。

8086 CPU外部特性

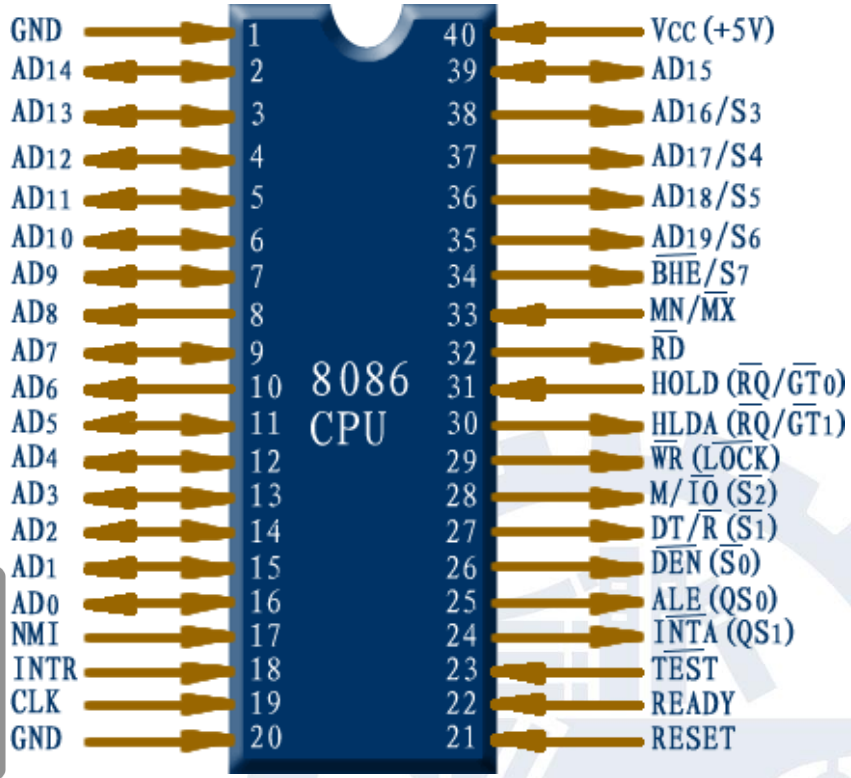
8086 CPU两种工作模式

•**最小模式：**系统中只有8086一个处理器，所有的控制信号都是由8086CPU产生(MN/MX=1)。

•**最大模式：**系统中可包含一个以上的处理器，比如包含协处理器8087。在系统规模比较大的情况下，系统控制信号不是由8086直接产生，而是通过与8086配套的总线控制器等形成(MN/MX=0)。

8086管脚定义

包括数据总线，地址总线和控制总线的定义，以及电源、时钟等引脚的定义



8086 CPU外部特性

8086 CPU管脚定义

- MN/MX＃：最大最小模式控制信号。
- AD15～AD0：地址/数据复用信号
- A19/S6～A16/S3：地址/状态信号选择。
- BHE/S7：允许总线高8位数据传送/状态信号选择
- RD# ：读信号，有效表示当前CPU正在读存储器或IO端口。
- WR# ：写信号，有效表示当前CPU正在写存储器或IO端口。
- M/IO#：区分存储器或IO端口访问信号。
- READY：准备就绪信号。CPU访问的存储器或I／O端口是否准备好传送数据

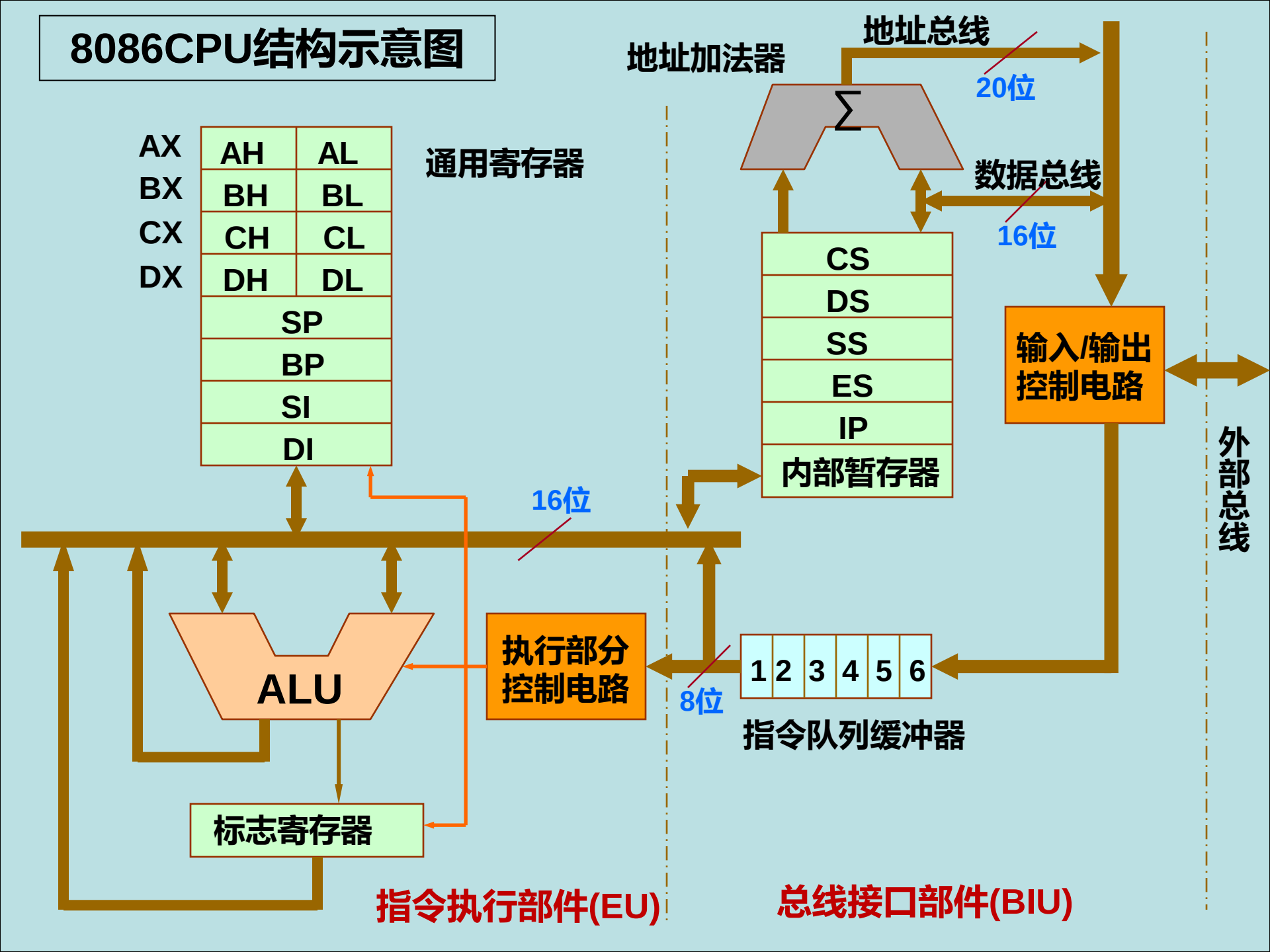
8086 CPU外部特性

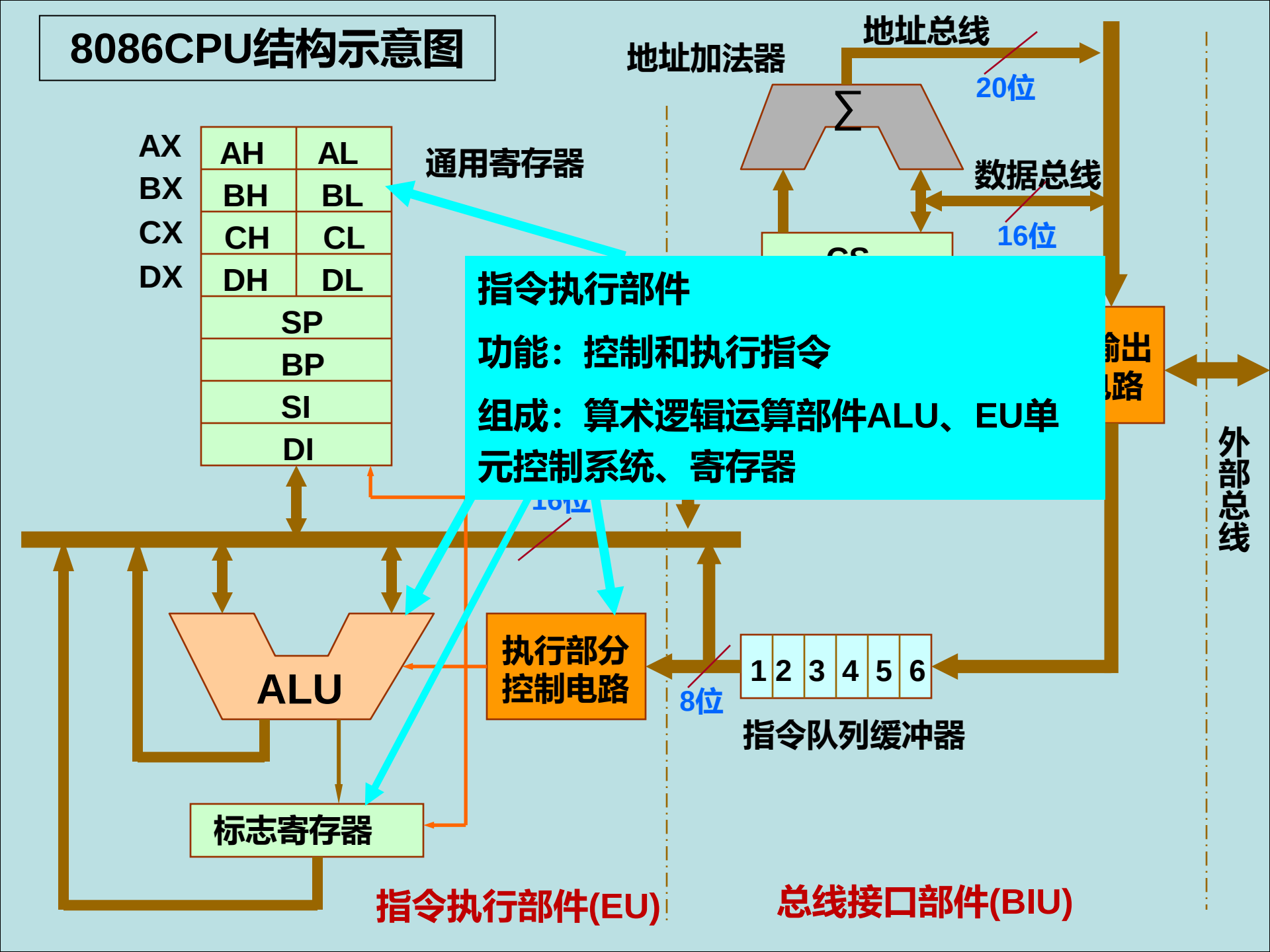
8086 CPU管脚定义

- INTR：外部输入的中断请求信号。
- INTA#：中断响应信号。
- NMI:不可屏蔽中断请求信号。
- ALE：地址锁存使能信号，用来作为地址锁存器的控制信号
- LOCK# ：封锁信号。三态输出，低电平有效。LOCK有效时表示CPU不允许其它总线主控者占用总线。
- HOLD：总线请求信号。由外部输入，高电平有效，表示向CPU请求使用总线
- HLDA：共享总线的处理总线请求响应信号。向外部输出，高电平有效

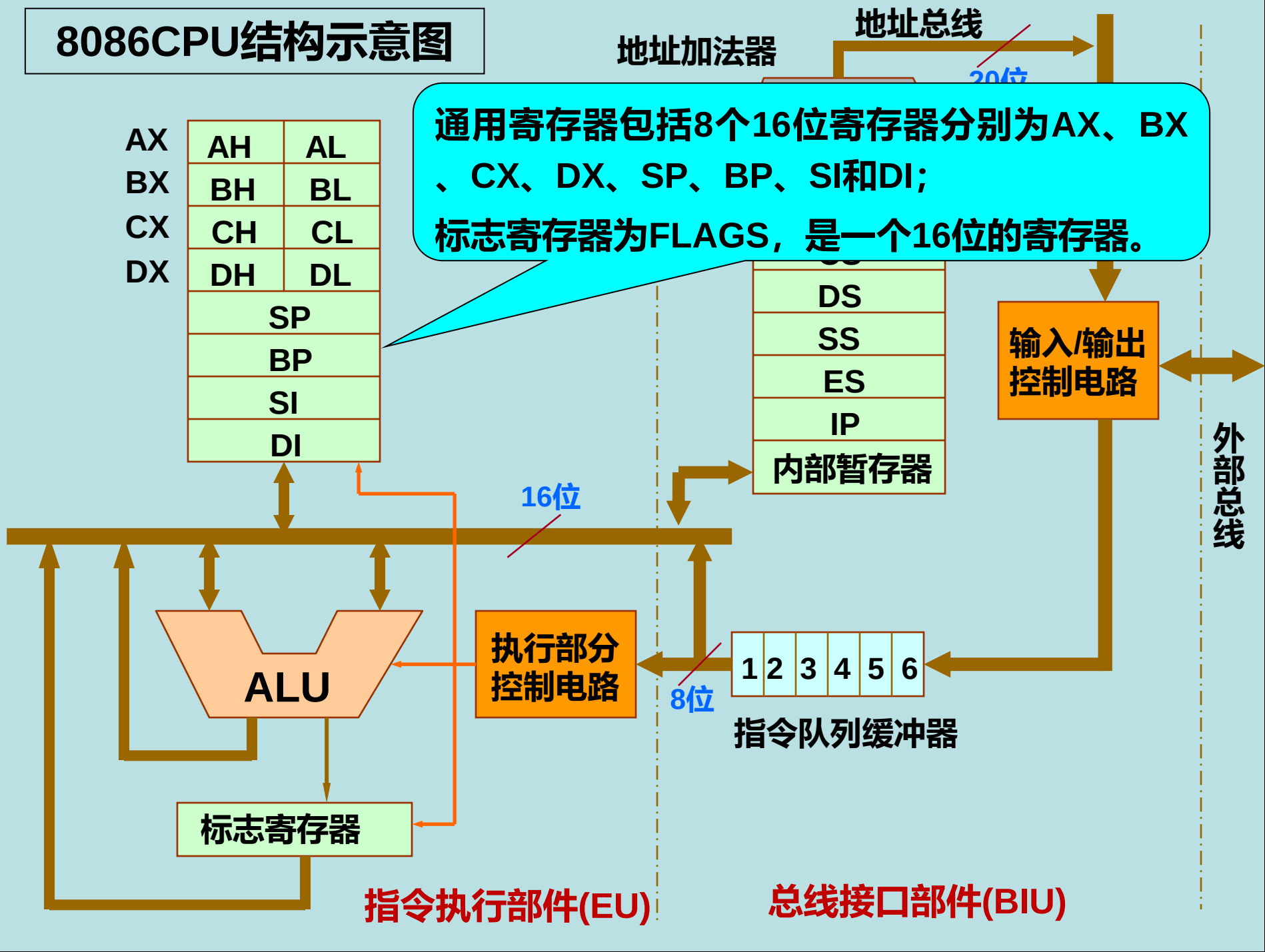
2. 8086 的功能结构

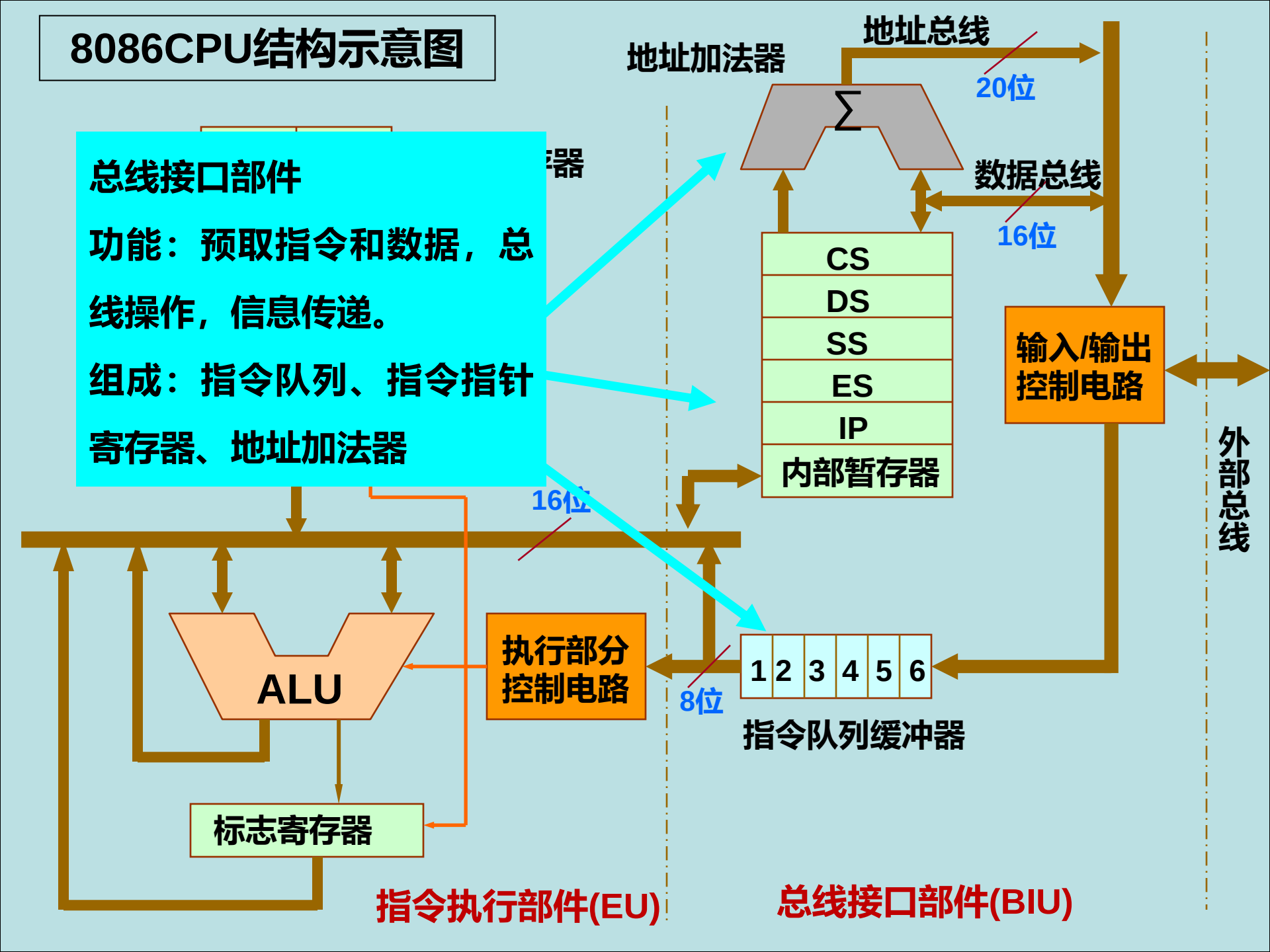
- 指令执行部件(EU)
  - 算术逻辑单元
  - 寄存器（通用和标志）
  - EU控制系统
- 总线接口部件(BIU)
  - 指令队列
  - 指令指针和段寄存器
  - 地址加法器


8086CPU结构示意图
地址加法器
地址总线
20位
数据总线
16位
AX
BX
CX
DX
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
通用寄存器
CS
DS
SS
ES
IP
内部暂存器
输入/输出
控制电路
外部总线
16位
ALU
执行部分
控制电路
1 2 3 4 5 6
8位
指令队列缓冲器
标志寄存器
指令执行部件(EU)
总线接口部件(BIU)


8086CPU结构示意图
地址加法器
地址总线
20位
数据总线
16位
通用寄存器
AX
BX
CX
DX
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
指令执行部件
功能：控制和执行指令
组成：算术逻辑运算部件ALU、EU单元控制系统、寄存器
输出
路
外部总线
16位
ALU
执行部分
控制电路
1 2 3 4 5 6
8位
指令队列缓冲器
标志寄存器
指令执行部件(EU)
总线接口部件(BIU)

8086CPU结构示意图



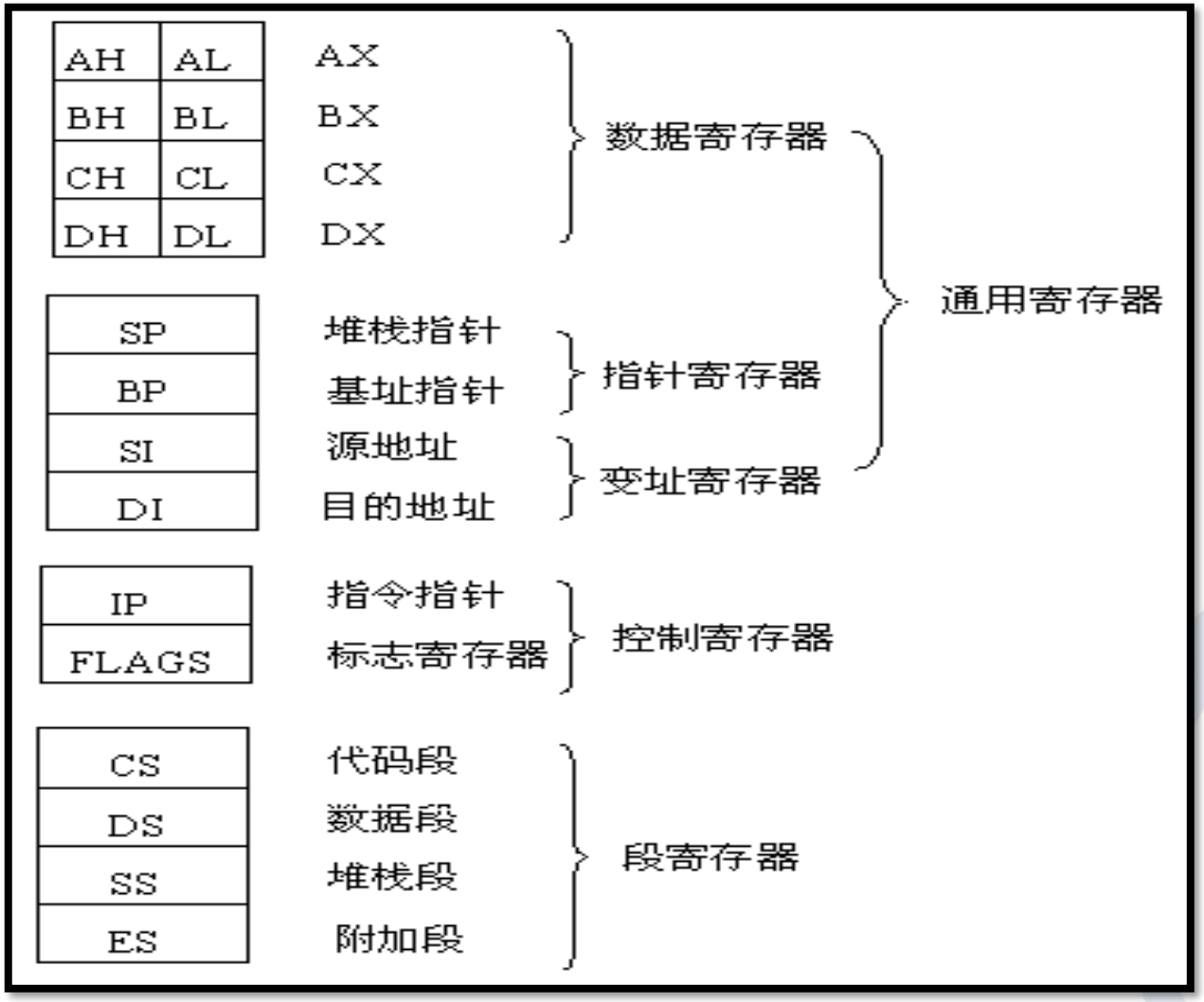

8086CPU结构示意图
地址加法器
地址总线
20位
数据总线
16位
总线接口部件
功能：预取指令和数据，总线操作，信息传递。
组成：指令队列、指令指针寄存器、地址加法器
CS
DS
SS
ES
IP
内部暂存器
输入/输出控制电路
外部总线
16位
ALU
执行部分控制电路
1 2 3 4 5 6
8位
指令队列缓冲器
标志寄存器
指令执行部件(EU)
总线接口部件(BIU)

- 总线接口部件
  - 负责管理与系统总线的接口，负责对存储器和外设访问
  - 包括指令队列、指令指针、段寄存器、地址加法器和总线控制逻辑
- 指令执行部件
  - 包括ALU、通用寄存器、标志寄存器和控制电路
  - 负责指令译码、数据运算和指令执行
- 指令执行的两个主要阶段：取指和执行
  - 取指：从主存取出指令代码进入指令队列
  - 执行：译码指令、并发出有关控制信号实现指令功能

指令预取（Prefetch）

- 8086处理器的指令读取，实际上是指令预取
  - 8086处理器维护着长度为6个字节的指令队列
  - EU单元译码、执行指令，同时BIU单元读取后续指令
  - BIU和EU两个单元相互独立，可以并行操作
- 最简单的指令流水线技术
  - 节省许多取指时间，提高了工作效率

8086CPU 寄存器分组



8086CPU 寄存器分组

累加器。用于算术、逻辑运算以及与外设传送信息等。

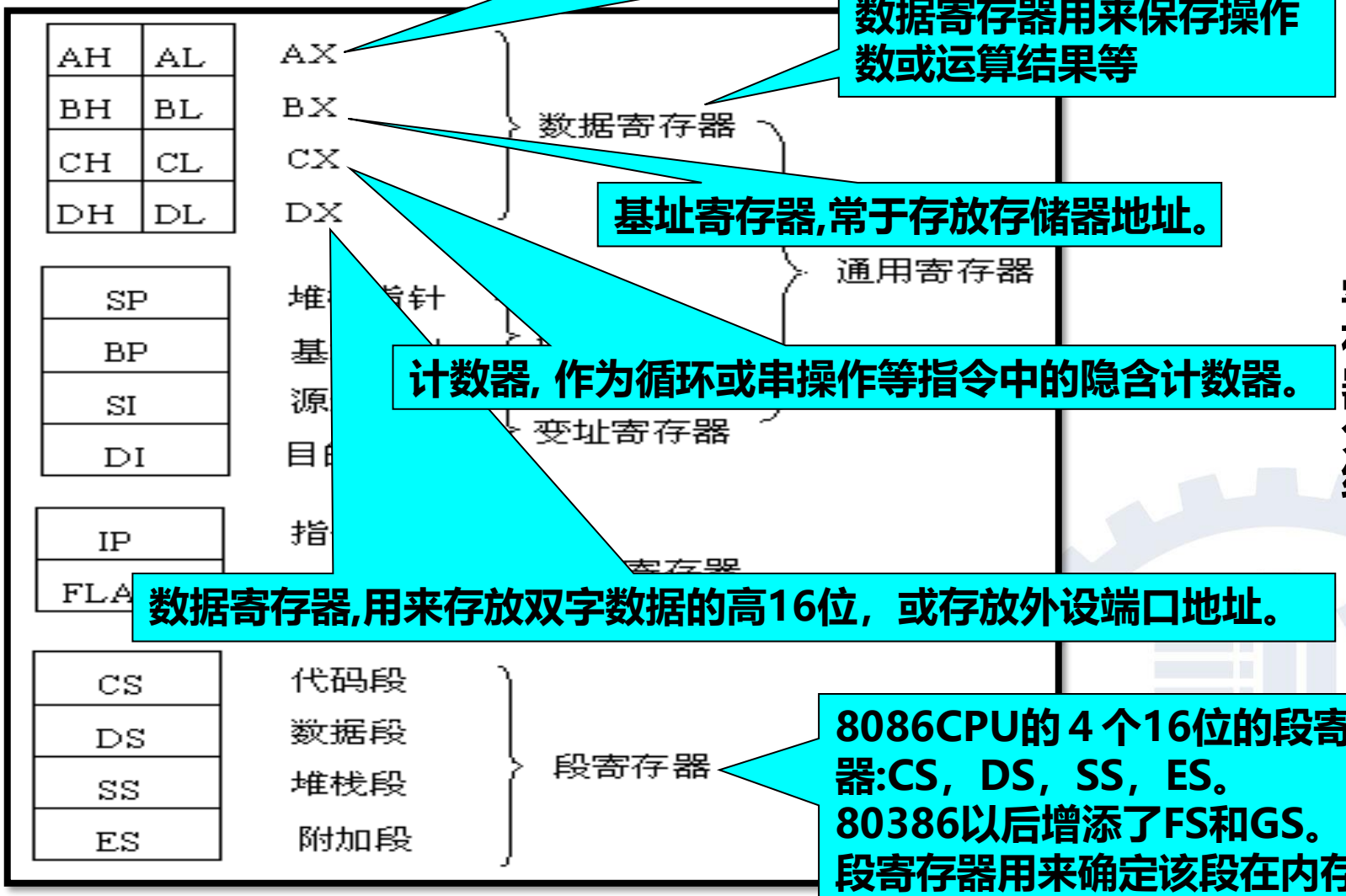
数据寄存器用来保存操作数或运算结果等

基址寄存器,常于存放存储器地址。

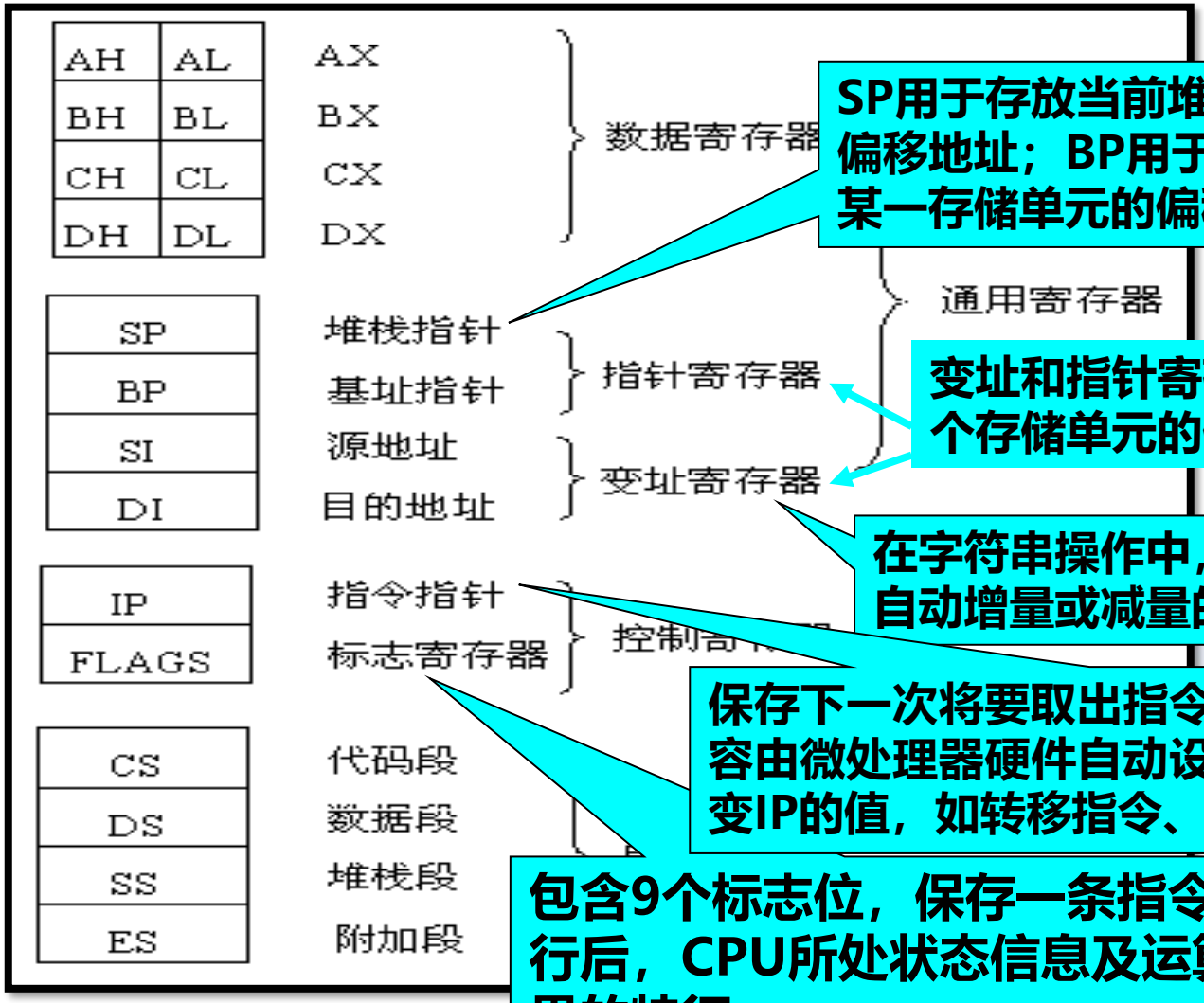
计数器, 作为循环或串操作等指令中的隐含计数器。

数据寄存器,用来存放双字数据的高16位，或存放外设端口地址。

8086CPU的 4 个16位的段寄存器:CS，DS，SS，ES。
80386以后增添了FS和GS。
段寄存器用来确定该段在内存中的起始地址。



8086CPU寄存器分类



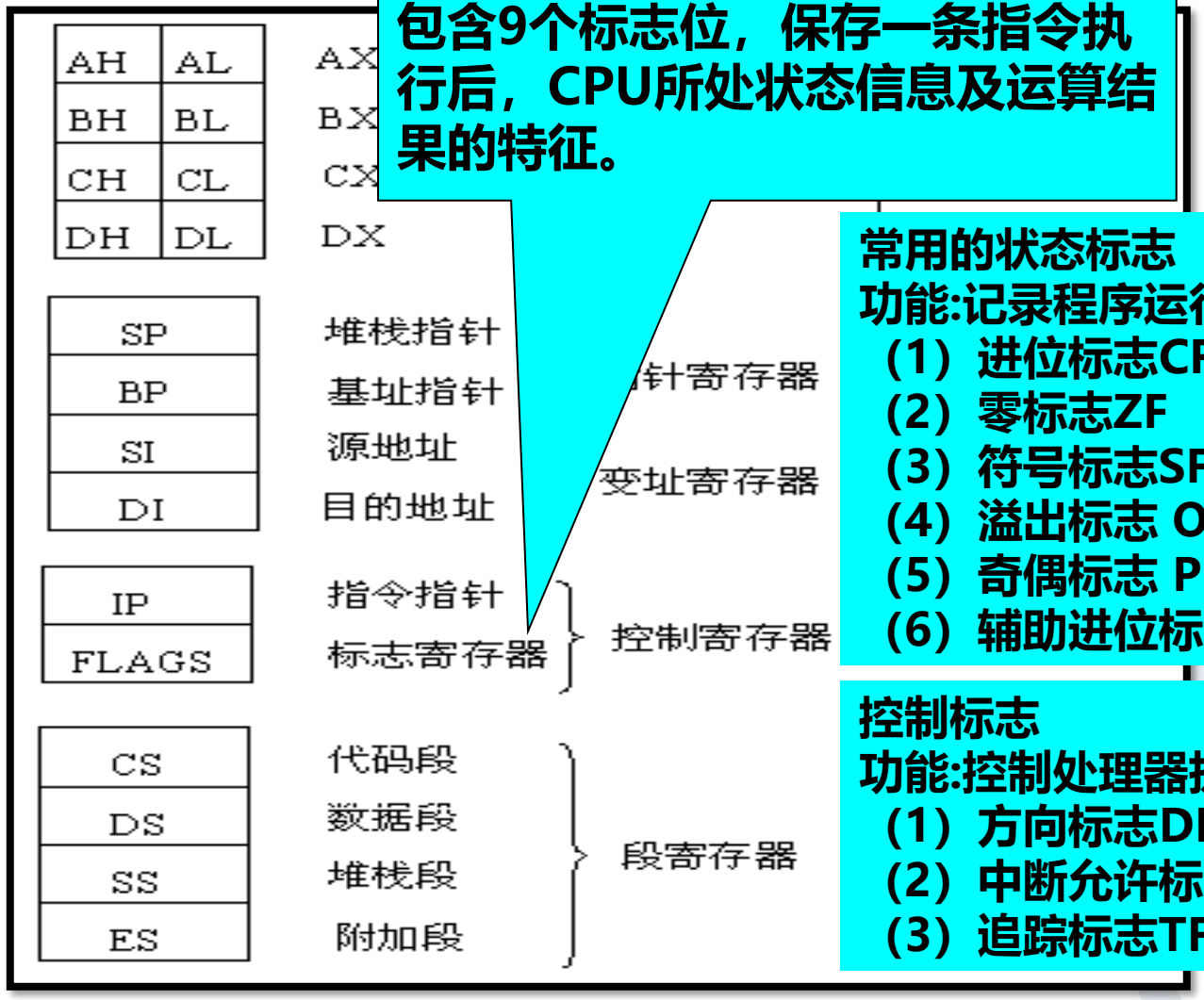
SP用于存放当前堆栈段中栈顶的偏移地址；BP用于存放堆栈段中某一存储单元的偏移地址。

变址和指针寄存器用于存放某个存储单元的偏移地址

在字符串操作中，SI和DI都具有自动增量或减量的功能。

保存下一次将要取出指令的偏移地址，IP的内容由微处理器硬件自动设置，有一些指令可改变IP的值，如转移指令、子程序调用指令等。

包含9个标志位，保存一条指令执行后，CPU所处状态信息及运算结果的特征。



包含9个标志位，保存一条指令执行后，CPU所处状态信息及运算结果的特征。

常用的状态标志
功能:记录程序运行结果的状态
（1）进位标志CF
（2）零标志ZF
（3）符号标志SF
（4）溢出标志 OF
（5）奇偶标志 PF
（6）辅助进位标志AF

控制标志
功能:控制处理器执行指令的方式
（1）方向标志DF
（2）中断允许标志IF
（3）追踪标志TF

8086

通用寄存器的专门用途

寄存器	用　　法
AX	字乘法，字除法，字I/O
AL	字节乘法，字节除法，字节I/O，十进制算术运算
AH	字节乘法，字节除法，DOS中断功能号
BX	存储器指针
CX	串操作或循环控制计数器
CL	移位计数器
DX	字乘法，字除法，间接I/O
SI	存储器指针(串操作中的源指针)
DI	存储器指针(串操作中的目的指针)
BP	存储器指针(存储堆栈指针)
SP	堆栈指针

控制寄存器

1. 指令指针寄存器IP

- 保存将要执行的指令在主存的地址
- 不能由指令直接修改
- 发生程序转移、中断、异常时由处理器自动改变

2. 标志寄存器

- **特别说明：堆栈的两种访问方式**
  - **随机访问方式：**

    用BP寄存器来指示随机访问地址！ 是内存！
  - **先进后出访问方式：**

    用SP寄存器来表示栈顶！ 也是堆栈！

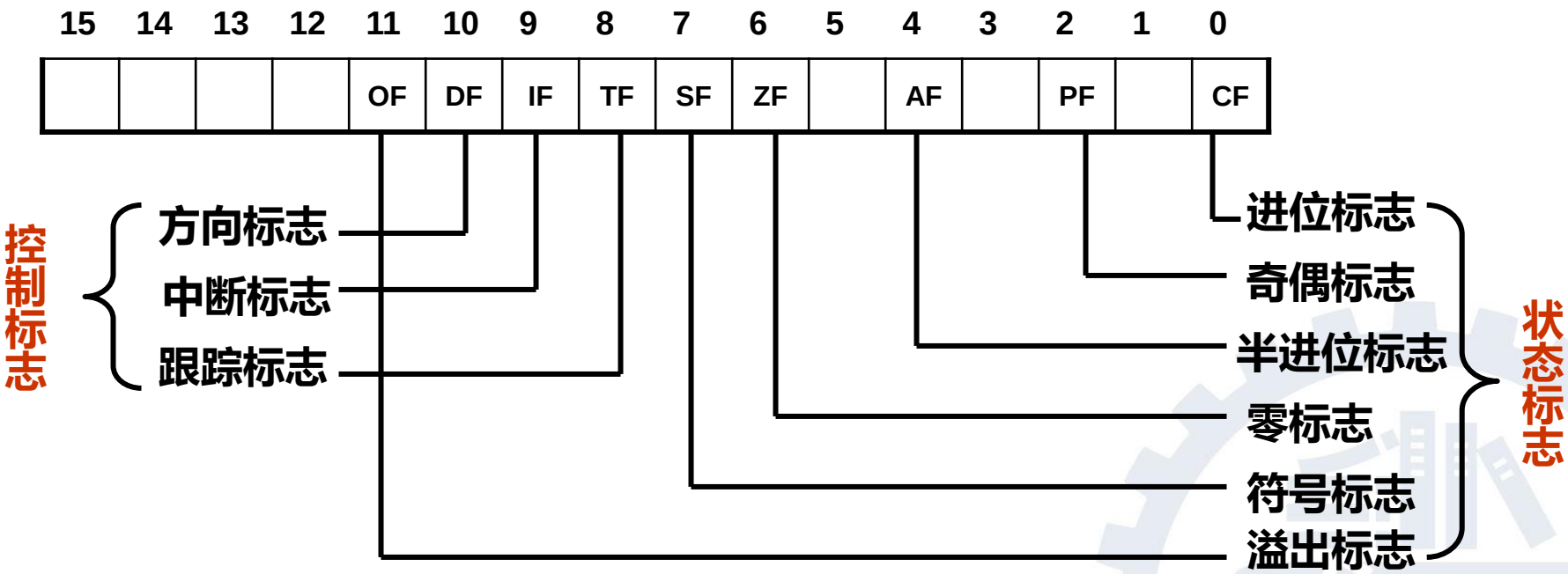
寄存器与存储器的比较：

寄 存 器	存 储 器
在CPU内部	在CPU外部
访问速度快	访问速度慢
容量小，成本高	容量大，成本低
用名字表示	用地址表示
没有地址	地址可用各种方式形成

标志寄存器（FLAGS/PSW）

- FLAGS是一个16位的寄存器，共9个标志，其中6个状态标志，3个控制标志
  - 状态标志反映EU执行算术逻辑运算后的结果特征
  - 控制标志用来控制CPU的工作方式或工作状态

标志寄存器的格式及各位的含义：



1. 状态标志——记录程序运行结果的状态信息

- 最基本的标志，有6个
- 用来记录指令执行结果的辅助信息
- 主要由加减运算和逻辑运算指令设置
- 处理器主要使用其中5个构成各种条件，分支指令判断这些条件实现程序分支

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

进位标志CF（Carry Flag）

- 当加减运算的最高有效位有进位（加法）或借位（减法）时，CF = 1，否则CF = 0

例1：00111010 + 01111100 = 10110110

最高位无进位， CF = 0

例2：10101010 + 01111100 = [1]00100110

最高位有进位，CF=1

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

- 针对**无符号整数**，判断加减结果是否超出表达范围
  - N个二进制位表达无符号整数的范围：

$$0 \sim 2^N - 1$$

  - 8位：0～255
  - 16位：0～65535
  - 32位：$0 \sim 2^{32} - 1$

AF(Auxiliary carry Flag)辅助进位标志，当运算结果的低4位向高4位有进位或借位时，AF=1。该标志与操作数长度无关。

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

溢出标志OF（Overflow Flag）

- **有符号数**加减结果有溢出则OF = 1，否则OF = 0

例1： 00111010 + 01111100 = 10110110

超出表达范围，OF = 1

例2： 10101010 + 01111100 = [1]00100110

未超出表达范围，OF=0

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

- **针对有符号整数，判断加减结果是否超出表达范围**
  - **N个二进制位表达有符号整数的范围：**

$$-2^{N-1} \sim 2^{N-1}-1$$

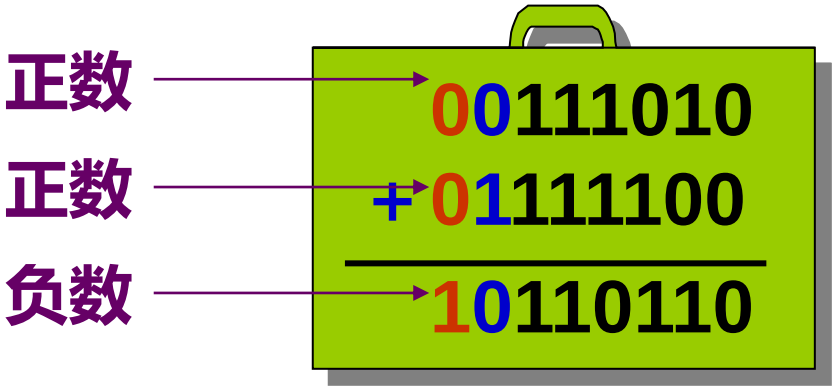
  - **8位：　- 128 ~ + 127**
  - **16位：　- 32768 ~ + 32767**
  - **32位：　$-2^{31} \sim +2^{31}-1$**

- CF反映**无符号整数**运算结果是否超出范围
  - 有进位，加上进位或借位后运算结果仍然正确
- OF反映**有符号整数**运算结果是否超出范围
  - 有溢出，运算结果已经**不正确**
- 处理器按照**无符号整数**求得结果
  - 设置进位标志CF
  - 设置溢出标志OF
- 程序员决定
  - 操作数是**无符号数**，关心**进位**
  - 操作数是**有符号数**，注意**溢出**

进位和溢出的区别

溢出标志的判断

- **处理器硬件判断规则**
  - 最高位和次高位同时有进位或同时无进位，无溢出；
  - 最高位和次高位进位状态不同，有溢出
- **人工判断的简单规则**
  - 只有当两个相同符号数相加（含两个不同符号数相减），而运算结果的符号与原数据符号相反时，产生溢出；其他情况下，不会产生溢出



零标志ZF（Zero Flag）

- 运算结果为0则ZF = 1，否则ZF = 0

例1：00111010 + 01111100 = 10110110

结果不是0，ZF = 0

例2：10000100 + 01111100 = [1]00000000

结果是0，ZF=1

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

符号标志SF（Sign Flag）

- 运算结果最高位为1则SF = 1，否则SF = 0

例1：00111010 + 01111100 = 10110110

最高位为1，SF = 1

例2：10000100 + 01111100 = [1]00000000

最高位为0，SF=0

进位

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

奇偶标志PF（Parity Flag）

- 当运算结果**最低字节**中”1”的个数为零或偶数时，PF = 1；否则PF = 0

例1：00111010 + 01111100 = 10110110

“1”的个数为5个，PF = 0

例2：10000100 + 01111100 = [1]00000000

“1”的个数为0个，PF=1

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

例

二进制加法

10010001B + 01110100B

= 00000101B

```
  10010001
+ 01110100
-----------
1← 00000101
进位
```

CF=1，产生进位　　AF=0，半字节无进位

SF=0，结果为正(最高位为0)　　ZF=0，结果不为0

PF=1，结果偶数个1

OF=0，将两个操作数作为有符号数，相加结果无溢出。

10010001B = -111D，01110100B = 116D

(-111D) + (+116D) = 5D

加法OF的判断：同符号数相加而和的符号与之相反则OF为1(产生溢出)，否则OF为0(不产生溢出)。

练习：下列十六进制数相加会如何影响标志位?

	8000H + 8000H 0000H	C000H + C000H 8000H	4008H + 4008H 8010H	0808H + C000H C808H
CF	1	1	0	0
PF	1	1	0	0
ZF	1	0	0	0
SF	0	1	1	1
OF	1	0	1	0

2. 控制标志——方向标志 DF(Direction Flag)

- 仅用于串操作指令，控制地址的变化方向
  - 设置DF = 0，每次串操作后的存储器地址就自动增加，即从低地址向高地址处理数据串
  - 设置DF = 1，每次串操作后的存储器地址就自动减少，即从高地址向低地址处理数据串
- 执行CLD指令设置DF = 0
- 执行STD指令设置DF = 1

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

- **中断允许标志IF（Interrupt-enable Flag）：** 主要针对外中断中**可屏蔽中断**的开放或禁止。当IF=‘1’时，CPU允许响应可屏蔽中断，中断当前程序，转去执行中断处理程序。IF = ‘0’时，则不允许响应可屏蔽中断。用STI指令使IF置‘1’，用CLI使IF=‘0’。

- **追踪标志TF（Trace Flag）：** 用于单步调试程序。当TF=‘1’时，在执行完一条指令后，产生单步中断。这在DEBUG调试程序状态下，可以使指令单步运行，可逐一检查各寄存器内容，标志状态、存储器的检查或修改等等。TF=‘1’时为调试程序时所用，当程序调试成功后让TF=‘0’，CPU正常工作不产生单步中断。

11	10	9	8	7	6	5	4	3	2	1	0
OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

主要内容

2.1 计算机系统

2.2 8086微处理器

2.3 **存储器地址**

1. 存储单元的地址和内容

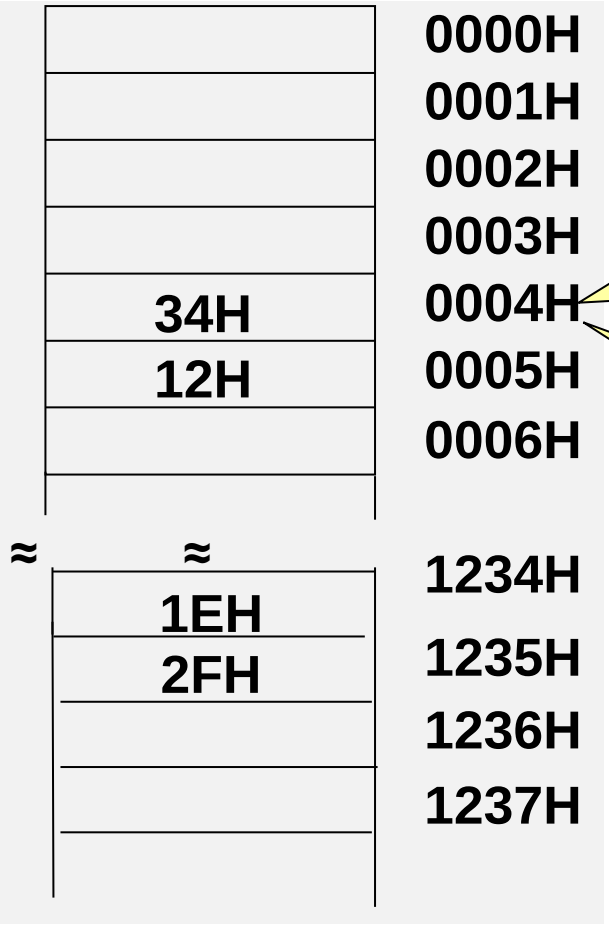
- **存储单元的地址**
  - 将存储单元编号，这个编号就是**存储器地址**
  - 用**十六进制数**来表示地址

例如：存储单元地址1234H

- **存储单元的内容**
  - 存储单元中存放的信息
  - 存储地址加括号”( )”(编程时用[])

例如：X表示某存储单元的地址，则X单元的内容可以表示为 (X)

又如：X单元中存放着Y，而Y是另一个存储单元的地址，则Y单元的内容表示为 (Y)=((X))



4号字节单元存放的信息为34H，表示：(0004H)=34H

4号字单元的内容为1234H，表示为：(0004H)=1234H

字单元由两个字节单元组成，其地址采用它的低地址来表示。

字存入存储器：低位字节存入低地址单元，高位字节存入高地址单元。

例：存储单元的地址

已知：

(0004H) = 5678H

字地址　　字内容

(0004H) = 78H

字节地址　　字节内容

问：

((0004H)) = ?

地址的地址　　字或字节内容

78H	0004H
56H	0005H
34H	0006H
12H	0007H
1EH	5678H
2FH	5679H

存储器的分段

8086的内部寄存器是16位，地址是20位，地址宽度大于字长。显然不能用16位的寄存器一次操作实现对2^{20} = 1M字节单元的寻址。

20 根地址线地址范围 00000H ~ FFFFFH （1MB）

物理地址：每一个存储单元有一个唯一的20位地址

机器字长16位：仅能表示地址范围 0000H ~ FFFFH（64KB）

为此，引入了存储器“分段”的概念，即把1M字节内存空间分成最大64KB的段 - 可由16位寄存器进行寻址。

段地址：表示一个段的开始

偏移地址：在段内相对于段起始地址的偏移值

优点：允许程序在存储器内重定位；

有利于程序和数据的分离。

小段的首地址		小段的尾地址
00000 H	~	0000F H
00010 H	~	0001F H
00020 H	~	0002F H
…		
FFFF0 H	~	FFFFF H

逻辑地址

段地址：偏移地址

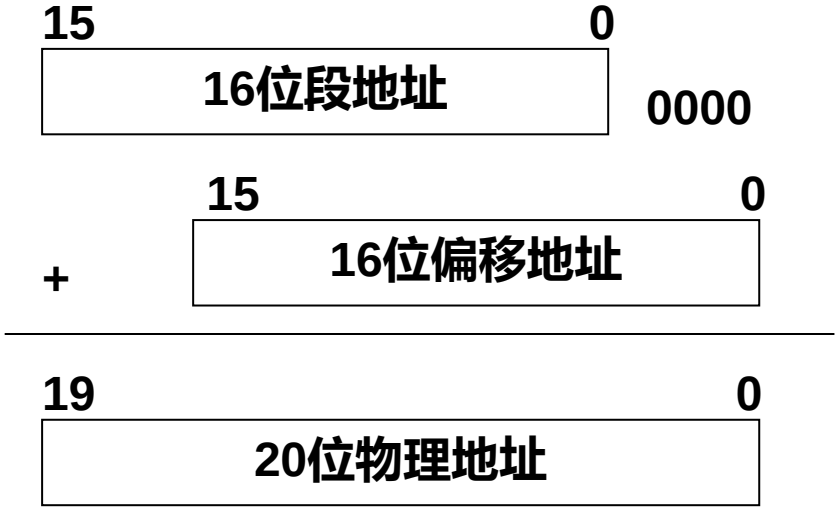
每16个字节为一小段，共有64K个小段

物理地址的形成

物理地址：每一个存储单元有一个唯一的20位地址

表示范围：00000H～FFFFFH 。

物理地址形成：段地址左移4位再加上偏移地址值



访问存储器的实际地址

这里左移的4位是二进制位；如果用十六进制表达地址就是左移一位。
左移4位还可以表达为乘以16，即：段地址×16＋偏移地址。

存储器的逻辑地址与物理地址

逻辑地址 段地址：偏移地址	内容	物理地址
1000：0000H	10011111	10000H
1000：0001H	00100110	10001H
1000：0002H	01001000	10002H
1000：0003H	10000011	10003H
1000：0004H	01011100	10004H
1000：0005H	10100010	10005H

字节内容（10000H）= 9FH;　（10001H）= 26H

字内容　（10000H）= 269FH;　（10001H）= 4826H

逻辑地址与物理地址的转换

例

段地址 = 2354H = 0010 0011 0101 0100 B

偏移地址 = 78DAH = 0111 1000 1101 1010 B

物理地址 = 2354H × 10H + 78DAH = 2AE1AH

0010 0011 0101 0100	2 3 5 4 0
+ 0111 1000 1101 1010	+ 7 8 D A
0010 1010 1110 0001 1010	2 A E 1 A

物理地址 = 段地址×16D+偏移地址 = 段地址×10H+偏移地址

逻辑地址与物理地址的转换

例

设某操作数存放在数据段，DS=250AH，数据所在单元的偏移地址=0204H。则该操作数所在单元的物理地址为：252A4H

物理地址 = 段地址×16D+偏移地址 = 段地址×10H+偏移地址

逻辑地址、物理地址

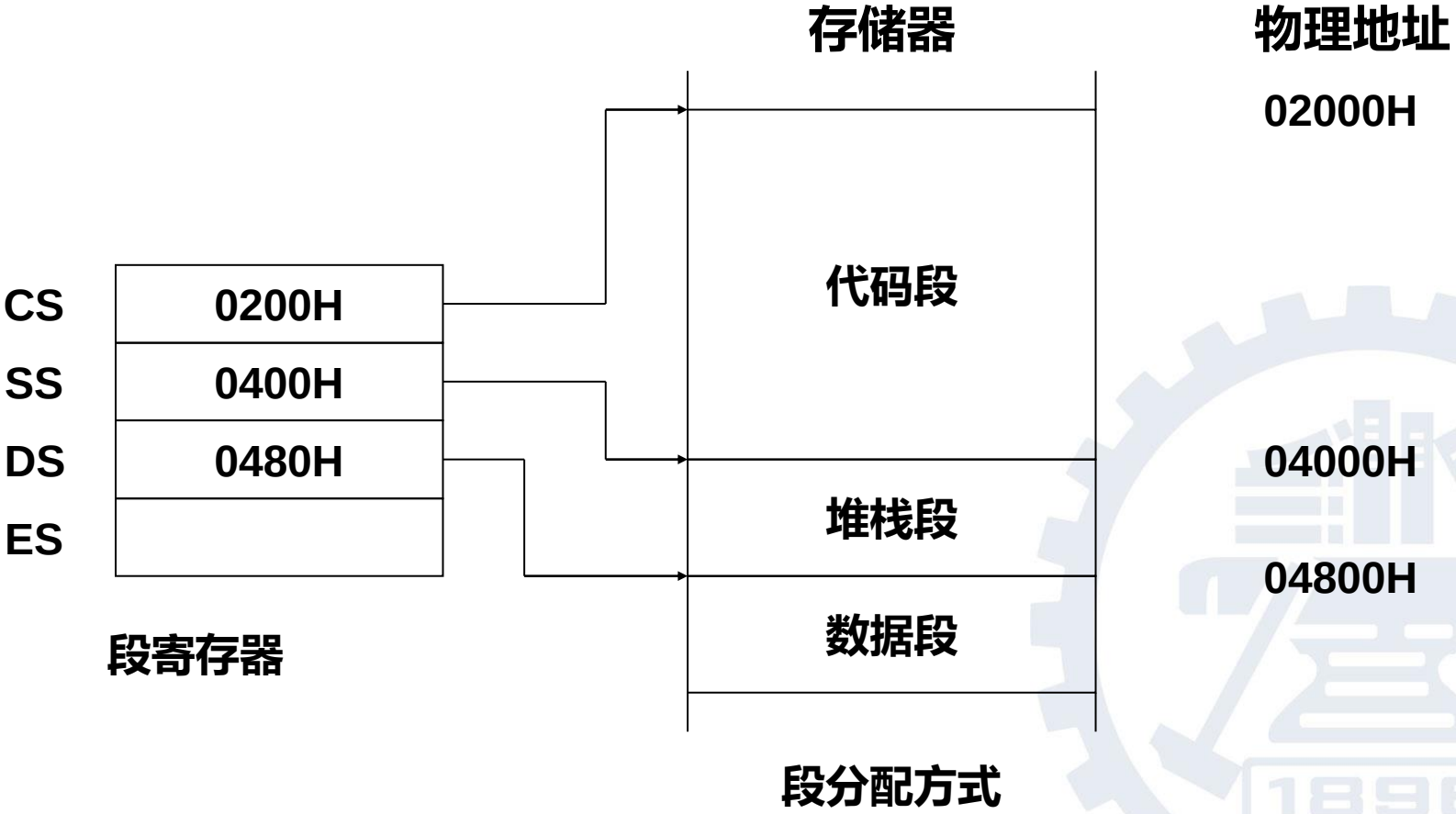
- 逻辑地址的形式：
  指令操作数，指令间的相对地址
- 物理地址：MEM的绝对地址
- 逻辑地址到物理地址的转换过程因CPU ARCH的不同而不同

段地址与段寄存器

- 段是安排相关代码或数据的主存区域
- 段寄存器表明段在主存中的位置
- 4个16位段寄存器：CS DS SS ES
  - 代码段CS主要存放运行的程序。
  - 数据段DS存放运行程序所用的数据。
  - 堆栈段SS定义了堆栈的所在区域。
  - 附加段ES是附加的数据段，它是一个辅助的数据区，在串操作指令中用到目的串必定存放在附加段中。

每个段寄存器可以确定一个段的20位起始地址中的高16位，称为段基址

各段在存储器中的分配是由操作系统负责的。一般情况下，根据需求量分配。注意：只有CS段寄存器是OS分配的，应用程序可读但不可写，其它的段寄存器都可以被修改。



隐含段和偏移寄存器

8086 ~ Pentium微处理机中，段寄存器和偏移寄存器组合有一定规则。

缺省16位段+偏移寻址组合

段	偏　移	主要用途
CS	IP	指令寻址
SS	SP或BP	堆栈寻址
DS	BX,DI,SI或16位数	数据寻址
ES	串指令DI	目标串寻址

注意：当BX/BP与SI/DI组合作为偏移地址时，默认的段寄存器是BX/BP所在段的寄存器，而不是SI/DI所在段的寄存器。

例1：(DS) = 2100H，(BX) = 0500H

物理地址 = 21000H + 0500H = 21500H

例2：(CS) = 3000H，(IP) = 1000H

物理地址 = 30000H + 1000H = 31000H

例3：(SS) = 1000H，(SP) = 0010H

物理地址 = 10000H + 0010H = 10010H

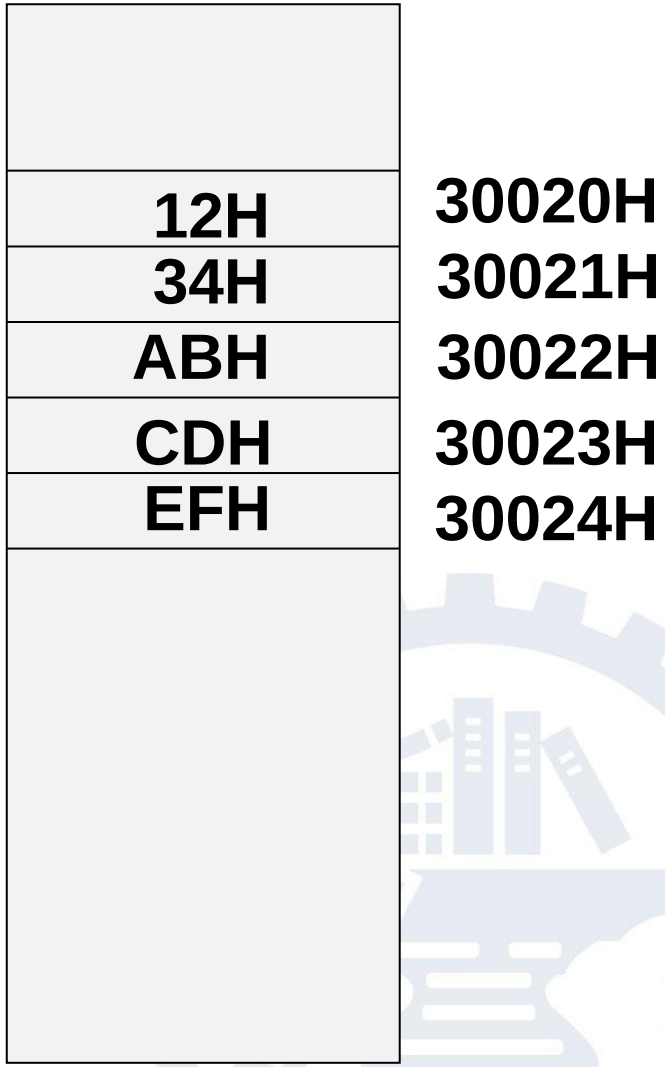
例4：(DS) = 3100H，(SI) = 1000H

物理地址 = 31000H + 1000H = 32000H

作 业

1、有两个16位字1EE5H和2A3CH分别存放在微机存储器000B0H和000B3H单元中，请用图示表示出它们在存储器里的存储情况。

2、微机存储器里存放的信息如右图示，试读出30022H和30024H字节单元的内容，以及30021H和30022H字单元中的内容。



3、在实模式下，段地址和偏移地址为3017H:000AH的存储单元的物理地址是什么？如果段地址和偏移地址是3015H:002AH和3010H:007AH呢?

4、如果在一个程序开始执行之前（CS）=0A7F0H（如果十六进制数的最高位为字母，则应该在其前加1个0），（IP）=2B40H，试问该程序的第1个字的物理地址是多少?