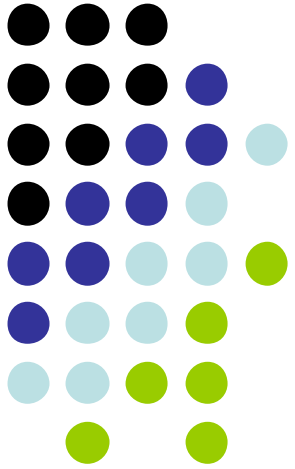
西南交通大学
Southwest Jiaotong University

第二章

8086微处理器及其系统结构



本章学习目的

1.掌握微处理器的编程结构;

2.掌握标志寄存器F各位的含义、作用及DEBUG运行后标志位的表示方法;

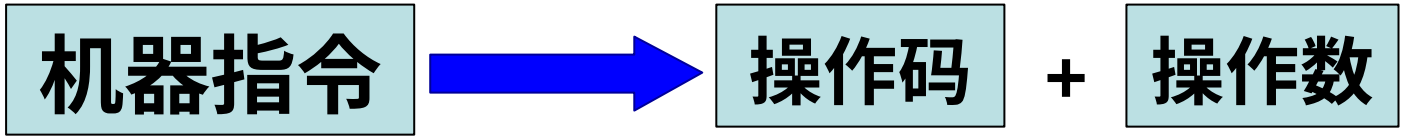
3.掌握8086微处理器的主要引脚功能;

4.掌握8086系统的存储器组织结构及堆栈的活动情况;

5.掌握8086的时钟及其发生电路,记住系统复位后各寄存器的状态。

第一节　8086微处理器内部结构

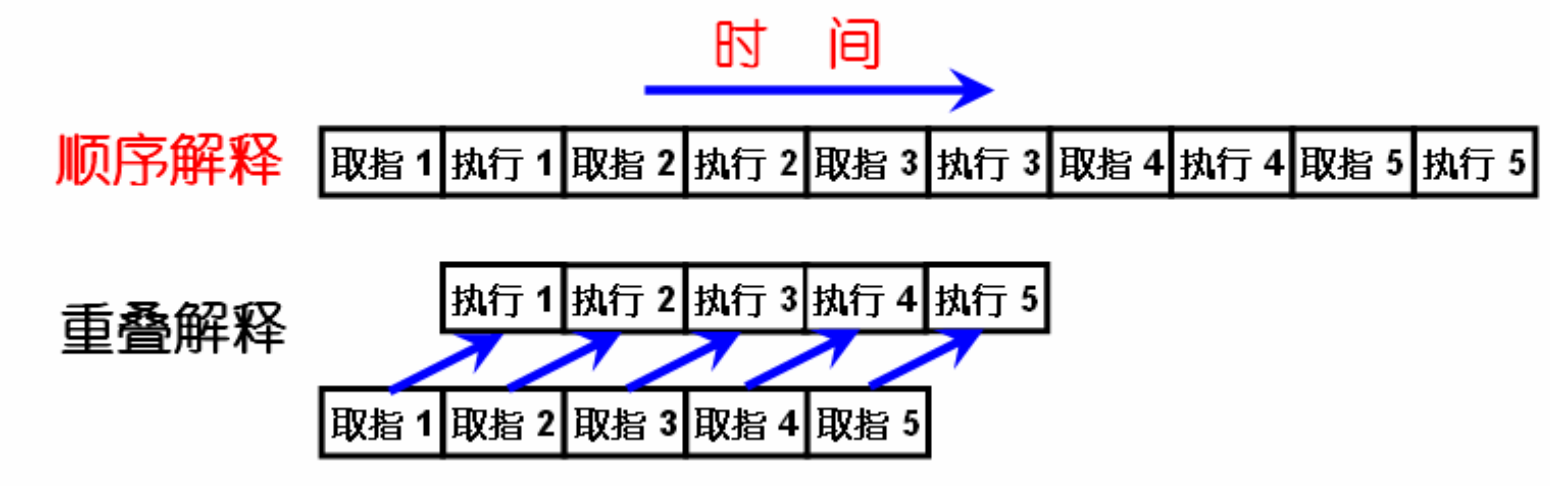
一、指令和程序的解释方式



若干条指令构成程序

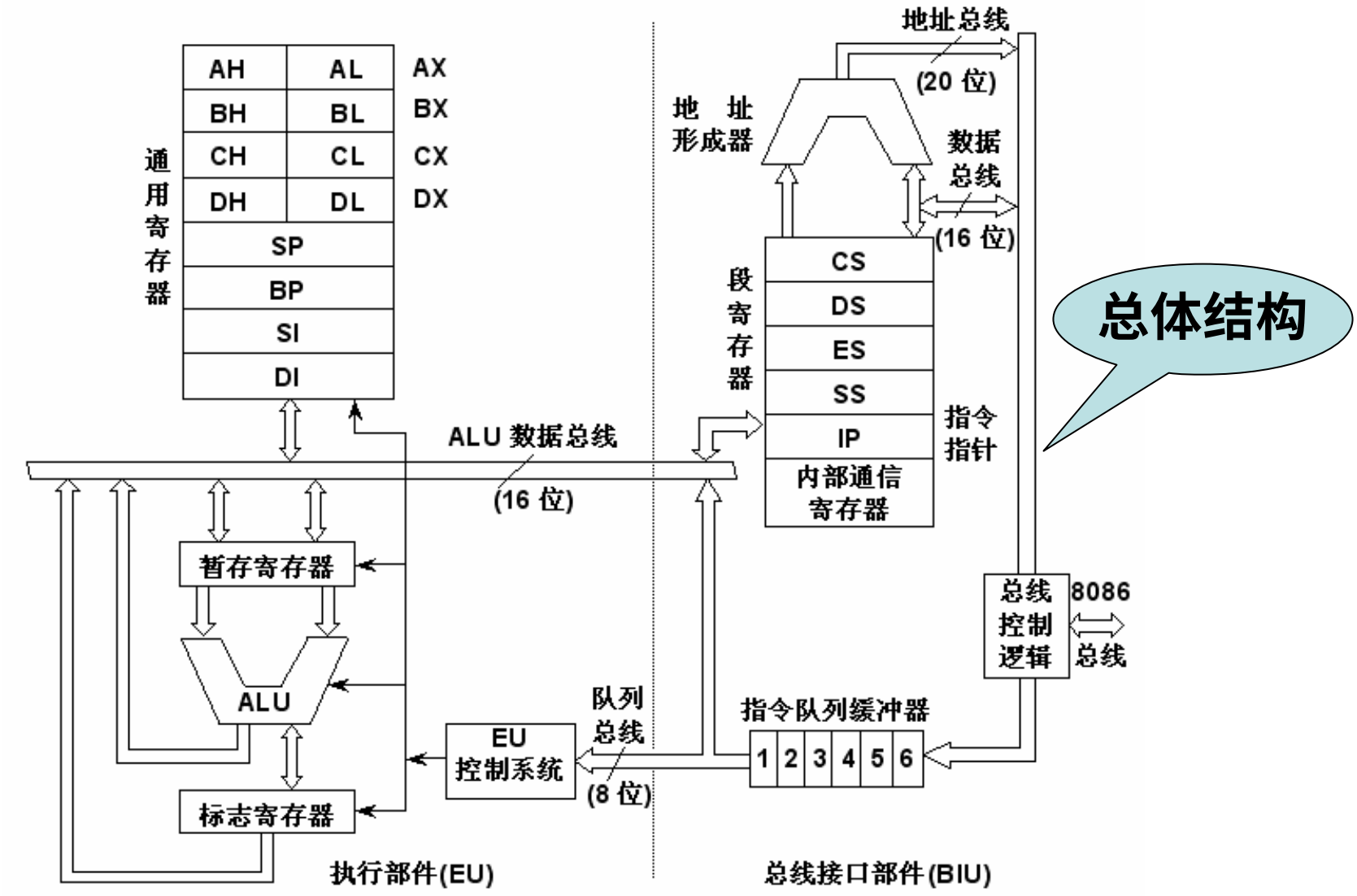
在微机中,微处理器解释一条指令的完整步骤可以归纳为如下两个阶段:

- **取指** 从内存中取出指令，明确指令规定的功能；
- **执行** 分析指令要求实现的功能，读取所需要的操作数，执行指令规定的操作，并保存执行结果。



8位机顺序解释 **VS** ***8086重叠解释***

二、8086的内部结构



8086/8088 CPU的内部是由两个独立的工作部件构成,分别是总线接口部件**BIU**(Bus Interface Unit)和执行部件**EU**(Execution Unit)。图中虚线右半部分是BIU,左半部分是EU。两者并行操作,提高了CPU的运行效率。

下面分别介绍**BIU**和**EU**两者的功能：

总线接口部件BIU

▲功能：实现8086CPU与存储器和外部设备之间的数据传送。

▲任务：

(1)从内存取指令®指令队列缓冲器;

(2) 取数据到运行单元进行运算并将运算结果送到目的单元(内存或外设端口);

(3)*重叠实现取指令与执行指令！*

指令队列缓冲器

▲结构：指令队列缓冲器为6字节(8088只有4字节)容量的FIFO。

设立指令队列缓冲器的目的是为了实现重叠解释指令。

▲工作原理：

①缓冲器中只要有一条指令，EU就开始执行；

②缓冲器中只要有两个字节为空，BIU便自动执行取指操作；

③当EU执行完转移、调用和返回等切换程序流程的指令时，缓冲器中原来的内容将被清除，BIU从内存中新的位置开始重新预取指令填入队列中。

地址发生器

▲组成：地址加法器、指令指针IP和段寄存器。

▲设立地址加法器的目的：8086 CPU地址总线有20位,而内部地址寄存器都是16位,需要加法器形成20位地址。

▲工作原理：段寄存器内容×16 + 偏移地址(IP或指令中的操作数地址给出)。

执行部件EU

▲功能：执行指令并暂存运算结果

▲结构：

(1)16位算术逻辑单元ALU;

(2)16位标志寄存器F;

(3)数据暂存寄存器(与编程无关,即不对用户开放)

(4)通用寄存器组：

AX、BX、CX、DX---数据寄存器

SP、BP ---指针寄存器

SI、DI ---变址寄存器

(5)EU控制电路：内部电路,不对用户开放

8086 CPU的编程模型

程序设计者所看到(或程序可见)的8086CPU即为该CPU的编程模型。包括三个部分：

编程模型
- 寄存器组
- 存储器
- I/O资源

三、8086的寄存器结构

1. 通用寄存器

通用寄存器共有8个,又可分为两组。

(1)数据寄存器

数据寄存器可以用于存放8位或16位的二进制操作数,这些操作数可以是参加操作的原始数据、运算得到的中间结果,也可以是操作数的地址。大多数算术和逻辑运算指令都可以使用这些寄存器。

在8086微处理器中,数据寄存器有4个,累加器AX(Accumulator)、基址寄存器BX(Base)、计数寄存器CX(Count)、数据寄存器DX(Data)。

每个16位数据寄存器可分为高8位(AH、BH、CH和DH)和低8位(AL、BL、CL和DL),可分别寻址,独立操作。

(2)指针寄存器和变址寄存器

指针寄存器和变址寄存器一般用来存放地址偏移量，用于堆栈操作和变址运算中计算操作数的有效地址。

指针寄存器指的是堆栈指针寄存器**SP**(Stack Pointer)和基址指针寄存器**BP**(Base Pointer)，其中**SP**用来指示堆栈顶部单元的位置，实现堆栈操作，而**BP**用来存放在现行堆栈段中的一个数据区的基地址。

变址寄存器包括源变址寄存器**SI**(Source Index)和目的变址寄存器**DI**(Destination Index)，分别用来存放源操作数和目的操作数的偏移地址。

在某些指令中,以上各通用寄存器具有特定的用途,称为寄存器的隐含用法。

寄存器	操作
AX	1）在字乘／字除指令中作累加器； 2）在字I/O操作时作数据寄存器。
AH	1）在字节乘／字节除指令中存放结果； 2）在LAHF中作目的寄存器。
AL	1）在字节乘／字节除指令中作累加器； 2）在字节I/O操作时作数据寄存器； 3）BCD、ASCII码数据运算时作累加器； 4）在XLAT指令中作累加器。
BX	1）间接寻址时作地址寄存器和基址寄存器； 2）在XLAT指令中作基址寄存器。
CX	1）串操作时作串长计数器； 2）循环操作时作循环次数计数器。
CL	移位操作时作移位次数计数器。
DX	1）字乘／字除指令中作辅助寄存器； 2）I/O指令间接寻址时作端口地址寄存器。
SP	堆栈操作时作堆栈指针。
SI	1）间接寻址时作地址寄存器和变址寄存器； 2）串操作时作源变址寄存器。
DI	1）间接寻址时作地址寄存器和变址寄存器； 2）串操作时作目的变址寄存器。

2. 指令指针IP(Instruction Pointer)

指令指针IP是一个16位专用寄存器,程序运行时,它始终指向EU要执行的下一条指令所在单元。当EU执行本条指令时,IP中的内容自动增量,以指向下一条指令所在的内存单元。

需要注意的是,程序中不能随意对IP进行存取操作,只有转移指令、子程序调用和返回指令以及中断处理时才能对IP的内容进行修改和设置。

3. 段寄存器

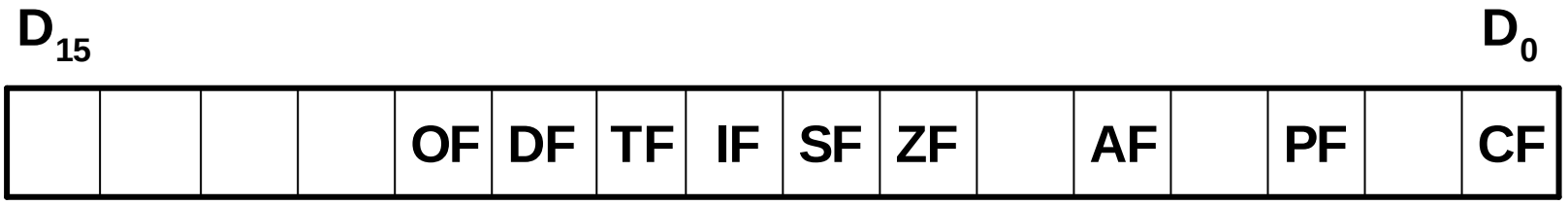
8086系统对内存实行分段管理，在BIU中有4个16位的段寄存器，专门用于存放各段在内存中的起始地址(段基址)。这4个段寄存器是：

①代码段寄存器CS(Code Segment)，指向当前代码段；
②数据段寄存器DS(Data Segment)，指向当前数据段；
③附加段寄存器ES(Extra Segment)，指向当前附加段；
④堆栈段寄存器SS(Stack Segment)，指向当前堆栈段。

1M字节的地址是分段寻址的，分段是个动态概念。段基址与具体程序无关。

4. 标志寄存器F(Flag Register)

标志寄存器位于EU中,它是一个16位的寄存器,但实际上只用了其中9位。



根据各位的作用和含义,各标志位又分为状态标志位和控制标志位两大类。

(1)状态标志位

用于反映EU执行算术或逻辑运算以后的结果特征,指令执行时由EU根据指令的执行结果设置这些位的状态,并进一步影响或控制某些后继指令(如条件转移指令、循环指令等)的执行。不同指令对各状态标志位的影响是不同的。

状态标志位共有6位,它们分别是:

- **进位标志CF**(Carry Flag) 反映算术运算后最高位是否出现进位或借位,有则为“1”,无则为“0”。
- **辅助进位标志AF**(Auxiliary Flag) 反映运算结果的**低4位**是否向高位产生进位或借位,有则为“1”,无则为“0”。
- **奇偶标志PF**(Parity Flag) 反映指令执行结果**低8位**数据中1的个数是否为偶数,若是则该位置“1”,否则置“0”。

- **零标志ZF**(Zero Flag) 反映运算结果是否为零，若是,则该位置“1”,否则置“0”。

- **符号标志SF**(Sign Flag) 反映运算结果最高位的状态,并与运算结果最高位状态相同。表明了本次运算的结果是正还是负。

- **溢出标志OF**(Overflow Flag) 反映带符号数进行算 术运算后是否有溢出,有则为“1”,无则为“0”。

(2)控制标志位

用于对CPU的某些操作实施控制,并可在程序中用相应的指令来设置其状态。3位控制标志分别是:

- **方向标志DF**(Direction Flag) 用于控制串操作指令执行时的步进方向,该位为“1”,则串操作指令按地址递减的顺序对串进行操作,否则按地址递增的顺序进行操作。

- **中断允许标志IF**(Interrupt Flag) 指示是否允许系统响应外部的可屏蔽中断请求,该位为“1”,表示允许(开中断),否则表示禁止(关中断)

- **陷阱标志TF**(Trace Flag) 当该位为“1”时,CPU每执行完一条指令便自动产生一个内部中断,并转去执行一个中断服务程序,可以借助该中断服务程序来检查每条指令的执行情况,称为“单步工作方式”,常用于程序的调试。

DEBUG中标志的表示

- NV等价于 OF=0，　　OV等价于OF=1;
- UP等价于 DF=0，　　DN等价于DF=1;
- DI 等价于 IF =0，　　EI 等价于 IF =1;
- PL等价于 SF=0，　　NG等价于SF=1;
- NZ等价于 ZF=0，　　ZR等价于ZF=1;
- NA等价于 AF=0，　　AC等价于AF=1;
- PO等价于 PF=0，　　PE等价于 PF=1;
- NC等价于 CF=0，　　CY等价于CF=1。

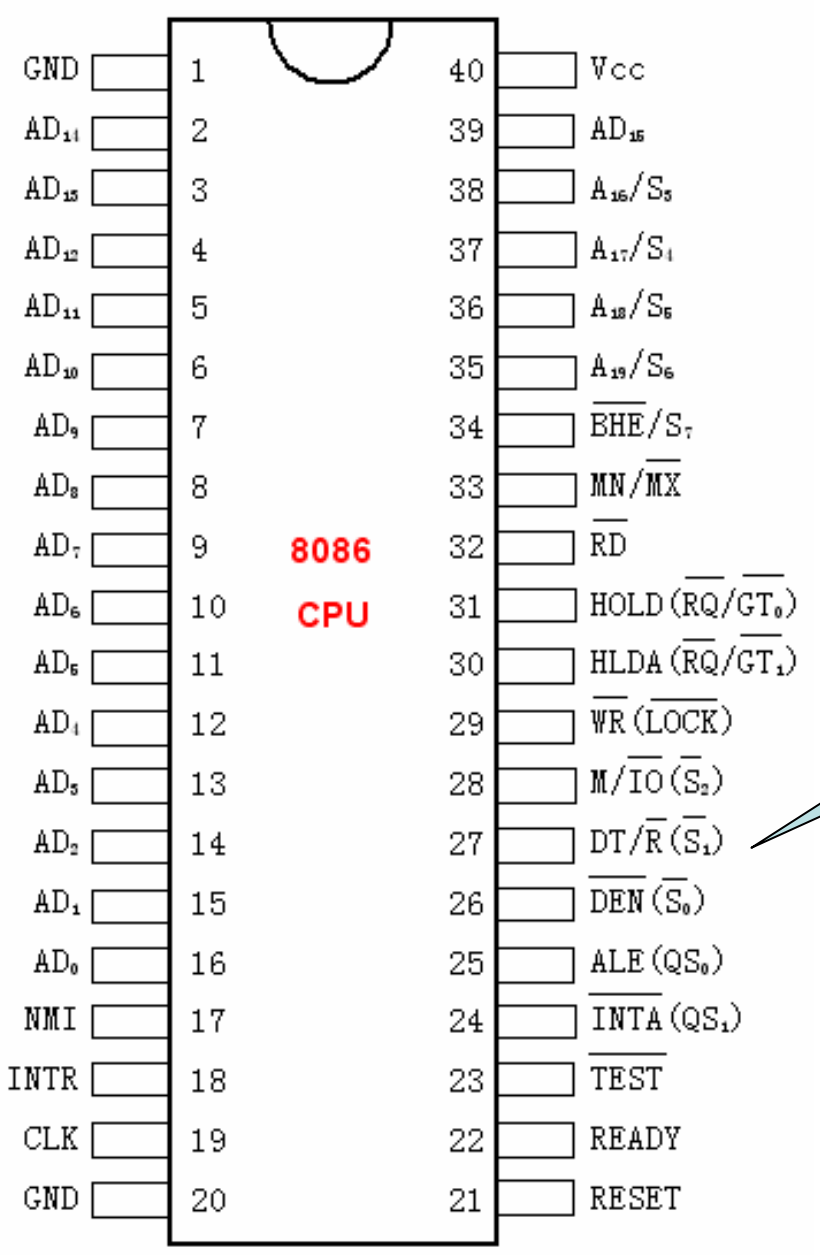
四、8088与8086在内部结构上的比较

Intel 8088是准16位CPU,其内部采用16位结构,与8086基本相同,外部数据总线宽度为8位。8088内部也包括两大功能部件,其中EU与8086完全一样,只是BIU略有区别:

① 8086指令队列是6字节长,而8088指令队列只有4字节长;

② 对于8086微处理器,当指令队列缓冲器中有2个字节变空时,BIU便自动执行取指操作;而对于8088,只要指令队列缓冲器中有1个字节变空便自动执行取指操作。

第二节　8086微处理器引脚功能



采用40引脚的DIP(双列直插)封装

8086是Intel公司的第三代微处理器——16位微处理器。它具有如下特点：

① 采用**分时复用**技术，在不同时刻通过相同引脚传送不同的信息，从而减少了引脚数量。

② 在两种不同的工作方式下，部分引脚具有两种不同的功能定义。

一、引脚功能

8086的40条引脚信号按功能分为4部分，即地址总线、数据总线、控制总线以及其它(时钟和电源)引脚。

1. 地址总线和数据总线

数据总线用来在CPU和内存(或I/O设备)之间传递数据,地址总线用于传送 CPU产生的内存单元(或I/O端口)的地址。前者为16位双向三态总线,后者为20位输出三态总线,其中:

① AD_{15}～AD_0为**地址/数据分时复用总线**。在每个总线周期的开始,用作地址总线的低16位,以给出内存单元(或I/O端口)的地址。其它时间作为数据总线,用于传输数据。

② A_{19}～A_{16}/S_6～S_3为地址/状态分时复用总线。在每个总线周期开始,用作地址总线的高4位,以给出内存单元的高4位地址;在I/O操作时,这4位置“0”。在总线周期的其它时间,这4条信号线指示CPU的状态信息,其中,S_6恒为低电平;S_5反映标志寄存器中IF的当前值;

S_4和S_3表示正在使用哪个段寄存器。

S_4	S_3	状态(所使用的寄存器)
0	0	ES
0	1	SS
1	0	CS
1	1	DS

③ $\overline{BHE}/S_7$为地址高允许/状态分时复用引脚。在总线周期的开始，作为数据总线高半部分允许信号，当$\overline{BHE}/S_7$为低电平时，将读/写的8位数据与AD_{15}～AD_8接通，该信号与地址总线的最低位A_0配合以决定传送的数据字中高字节是否有效，传送的是16位还是8位数据。在总线周期的其它时间，该引脚输出备用状态信号S_7。

2. 控制总线

控制总线是传送控制信号的一组信号线，其中有些是输出线，用来传送CPU送至其它部件的控制命令；有些是输入线，由外部向CPU输入状态及请求信号（复位、中断请求等）。

8086的控制总线中有一条MN/$\overline{MX}$线，即最小/最大方式控制线，用于决定8086的工作方式。当其接+5V时，8086处于最小方式；当其接地时，8086处于最大方式。

以下几条不受MN/$\overline{MX}$影响的控制线:

① 读控制信号$\overline{RD}$,三态,输出。

② 准备就绪信号READY,输入。

③ 可屏蔽中断请求信号INTR,输入。

④ 非屏蔽中断请求信号NMI,输入。当该引脚上产生从低电平到高电平的正跳变时,表示外部向CPU发出非屏蔽中断请求。

⑤ 等待测试控制信号$\overline{TEST}$,输入。

⑥ 复位信号RESET,输入。当其为高电平时,系统进入复位状态。

3. 其它信号

① 时钟信号CLK，输入，该信号为8086提供基本的定时脉冲。

② 电源Vcc，输入。要求接+5 V±10%。

③ 地线GND。

二、8086的工作方式及信号定义

8086有最小和最大两种基本的工作方式，最小方式一般用于构成一个小型的单处理机系统，而最大方式一般用于多处理机系统。

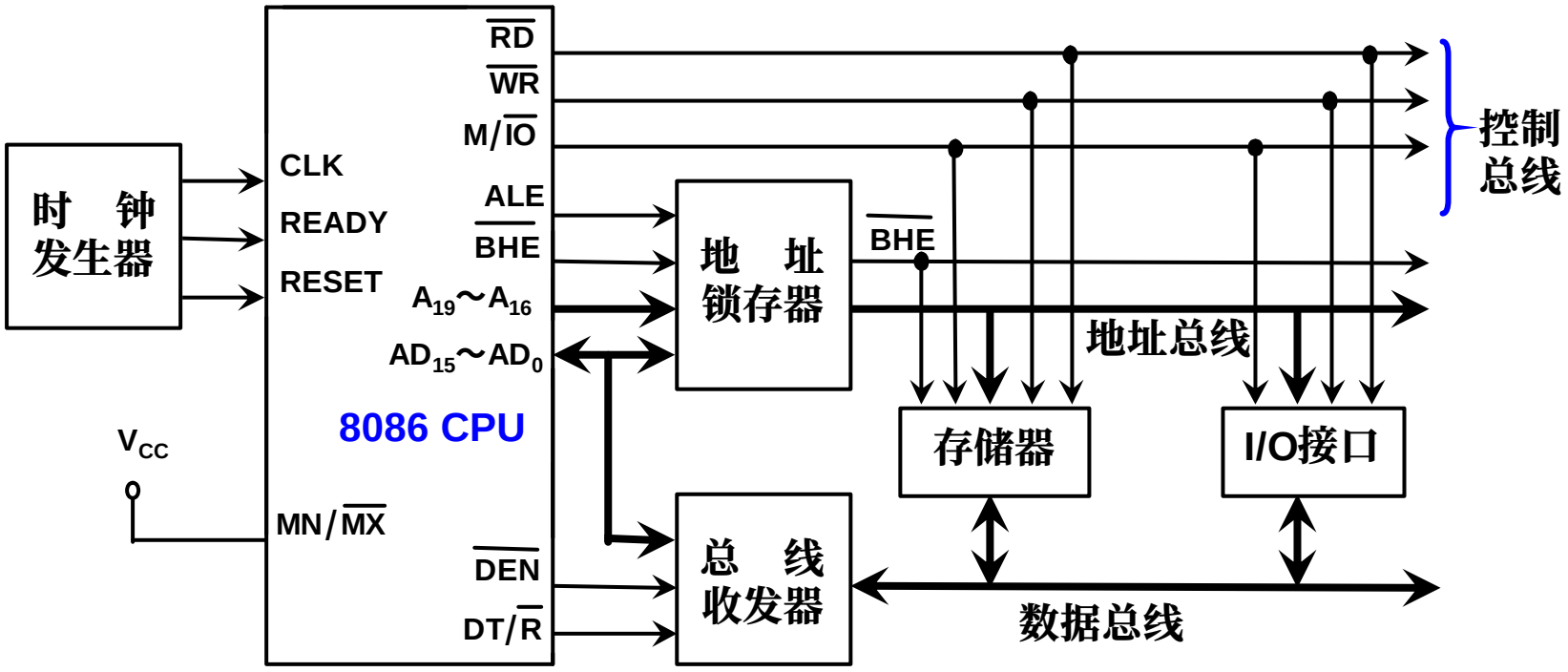
在最小方式下，8086微处理器提供系统所需的全部控制信号；

在最大方式下，由总线控制器8288提供系统的总线控制信号。

- M/$\overline{IO}$为存储器/输入输出控制信号，输出，三态。用于指示当前CPU是访问存储器(M/$\overline{IO}$为高电平)还是I/O端口(M/$\overline{IO}$为低电平)。
- DT/$\overline{R}$为数据发送/接收信号，输出，三态。用于指示CPU是进行读操作(DT/$\overline{R}$为低电平)还是写操作(DT/$\overline{R}$为高电平)。
- $\overline{DEN}$为数据允许信号，输出，三态。在CPU访问存储器或I/O端口的总线周期的后一段时间，该信号有效，用作系统中总线收发器的允许控制信号。

- $\overline{WR}$　写控制信号,输出,三态。用于指示 CPU在进行对存储器或I/O进行写操作。
- ALE　为地址锁存允许信号(输出)。8086在总线周期的开始通过地址总线输出地址的同时,通过该引脚输出一个正脉冲,其下降沿用于将地址信息打入外部的地址锁存器中。
- $\overline{INTA}$　为中断响应信号(输出,三态)。当8086响应来自INTR引脚的可屏蔽中断请求时,通过该引脚输出连续的两个负脉冲。

三、最小方式下的基本配置



- 时钟发生器为8086(8088)以及其它外设芯片提供所需的时钟信号。常用的时钟发生器/驱动器芯片为**8284**。
- 常用的地址锁存器有Intel 8282、Intel8283、74LS273、**74LS373**等。对8086系统来说,由于要锁存的信息共有21位,所以至少需要3片锁存器芯片,每片锁存8位信息。
- 总线收发器用来对AD_{15}～AD_0上的数据进行缓冲和驱动,常用的总线收发器还有**74LS245**、Intel 8287、**8286**等。

四、8088与8086在引脚和系统配置上的区别

① 8088的**地址/数据复用总线**只有8条，即AD_7～AD_0，而A_{15}～A_8为单一功能的地址总线。

② 8088的存储器/输入输出控制线为 IO/$\overline{M}$，当其为高电平时表示访问I/O端口；为低电平时表示访问存储器。

③ 8088无$\overline{BHE}$/S_7信号。

在系统配置上，8088同样有最大模式和最小模式两种工作方式，可以构成单处理机系统，也可构成多处理机系统。只需要一片数据总线收发器用于形成8位系统数据总线。在小规模系统中，如只需要用到16根地址总线，地址锁存器也只需要2片。

第三节 8086系统的存储器组织

一、存储器的分段管理

为什么要分段管理?

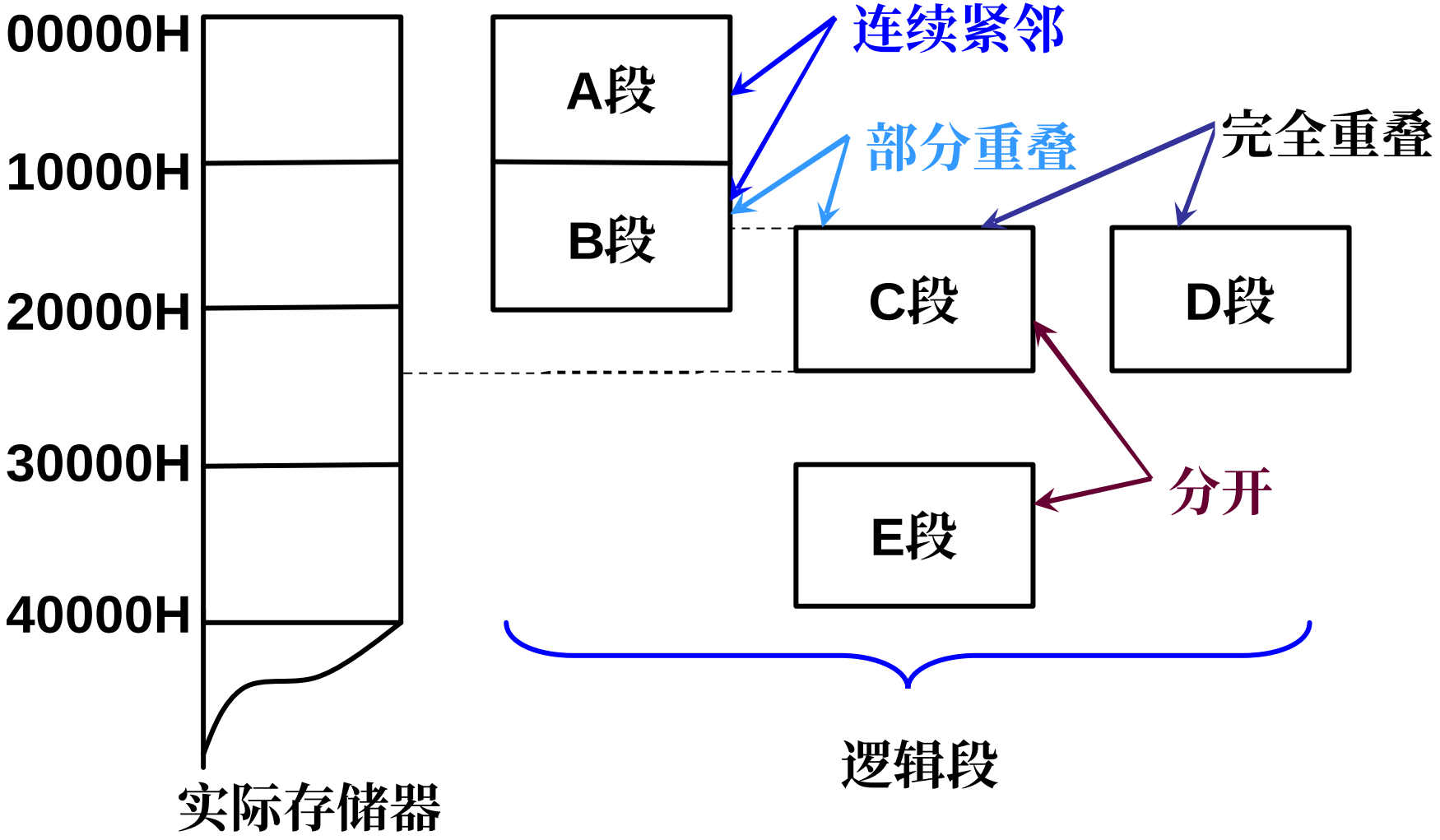
8086微处理器内部数据通路和寄存器均为16位,内部ALU只能进行16位数据的运算,在程序中也只能使用16位地址,寻址范围局限在$2^{16}=65536$(64 K)字节,为了能寻址1 M字节地址空间,必须对内存实行分段管理。

分段方法

8086程序将1 M字节的存储空间视为一组存储段，各段的功能由具体用途而定，分为**代码段**、**堆栈段**、**数据段**和**附加段**。

一个存储段是存储器的一个逻辑单位，每个段由连续的若干存储单元构成（最多64 K个），并且都是存储器中独立的、可分别寻址的单位。

各段在空间上可以完全重叠、**部分重叠**，**可以连续紧邻**，**也可分开**。


00000H
10000H
20000H
30000H
40000H
实际存储器
A段
B段
C段
D段
E段
连续紧邻
部分重叠
完全重叠
分开
逻辑段

每个段第一个字节的位置称为“段起始地址”,段起始地址是一个能被16整除的数,即二进制数的最后4位必须是0。段起始地址中的高16位(段基值)存放于相应的段寄存器中。

2. 物理地址与逻辑地址

物理地址又称**实际地址PA**(Physical Address),是CPU和存储器进行数据交换时所采用的地址。

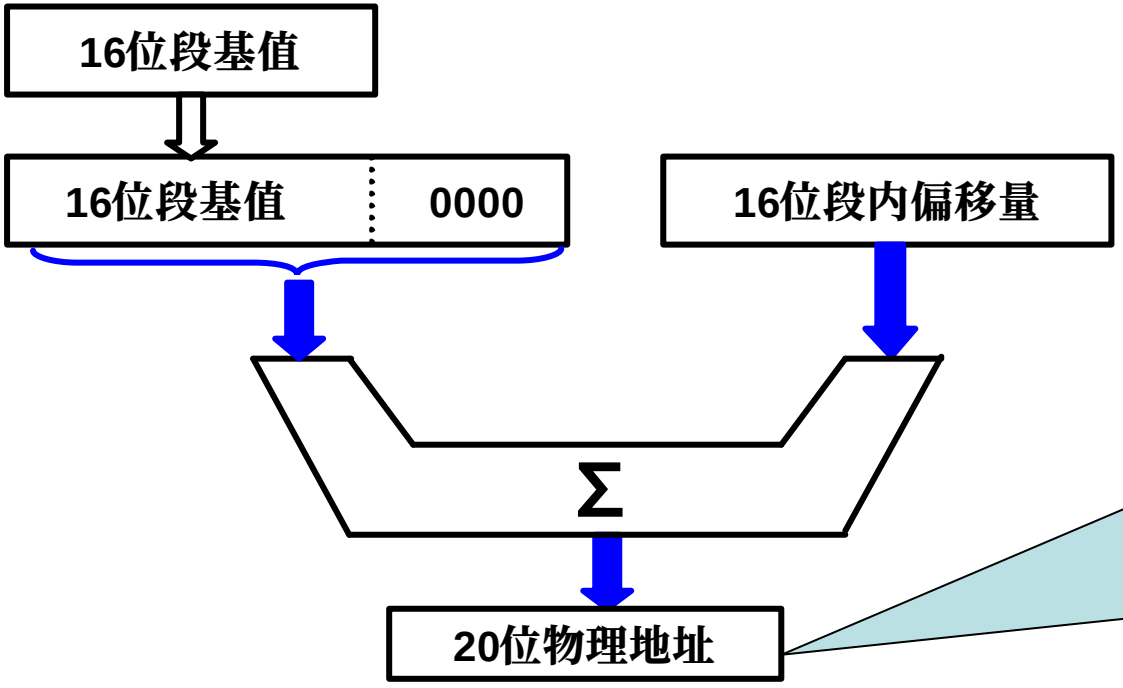
逻辑地址是程序员在程序中使用的地址。每个内存单元的逻辑地址由段基值和段内偏移量两部分构成,其中段基值确定需要访问的单元在哪一个段,段内偏移量确定需要访问的单元相对于该段起始单元的距离。**段基值**和**段内偏移量**都是16位二进制无符号数,合称为**32位地址指针**。由这两部分就可唯一确定一个内存单元。

3. 物理地址的形成方法

给出需要访问的内存单元的逻辑地址后，必须首先由CPU中的BIU将其转换为物理地址，才能通过系统地址总线送至存储器。

怎么确定要访问的单元?

由于段基值代表的是需要访问的单元所在段的起始单元地址的高16位，而偏移地址代表需要访问的单元与段起始单元的距离，所以在形成指定单元的20位物理地址时，只需将段基值左移4位再加上16位的偏移地址即可。



例如，假设某内存单元的逻辑地址为1234H:5678H，则该单元的物理地址为：

PA=1234H×16+5678H=179B8H。

在程序执行过程中，需要访问内存时，必须提供其逻辑地址。各种访问内存单元的操作中，逻辑地址的来源可归纳下表所示。

EA（Effective Address）为有效地址，相当于段内偏移量。

操作类型	隐含的段基值	可替换的段基址	偏移地址
取指令	CS	无	IP
堆栈操作	SS	无	SP
BP用作基址寄存器	SS	CS、DS、ES	EA
通用数据读写	DS	CS、SS、ES	EA
字符串操作(源地址)	DS	CS、SS、ES	SI
字符串操作(目的地址)	ES	CS、SS、DS	DI

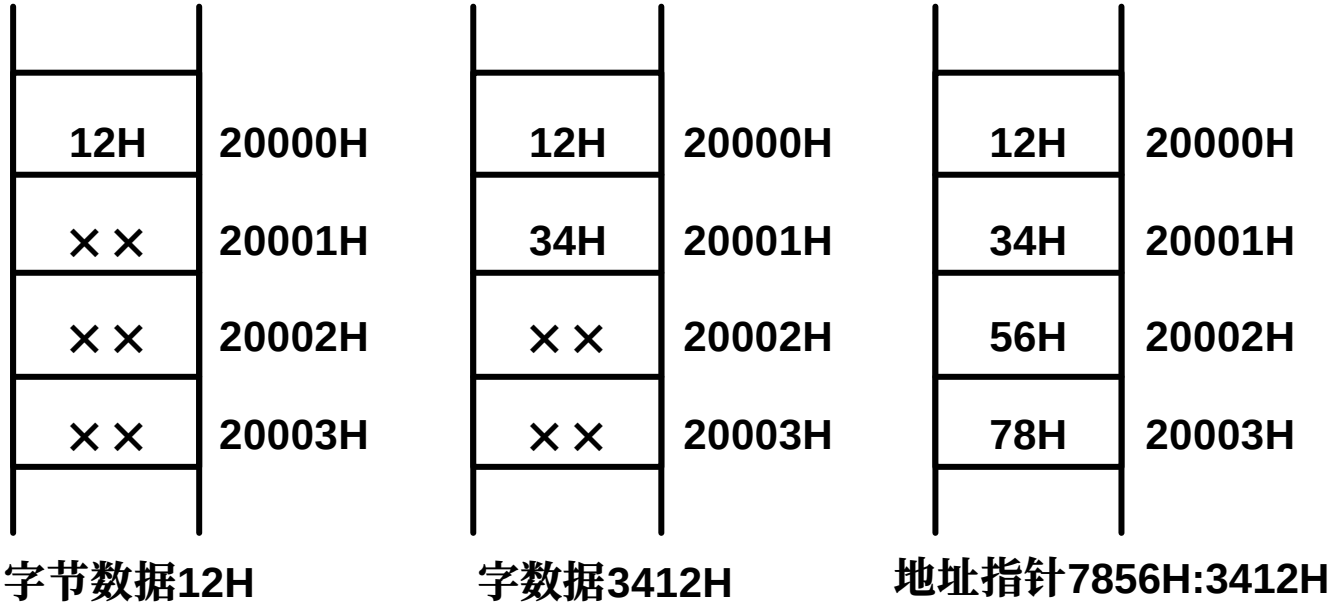
二、存储器组织

8086系统地址总线有20条，可以寻址1 M(2^{20})字节的内存空间，内存空间都按字节组织，每个内存单元存储一个字节的数据，并具有一个唯一的20位地址编号，称为字节地址。如果存放16位的字数据，则需要占用连续的两个单元，其中高字节存放在高地址，低字节存放在低地址，并以该单元地址作为该字数据的字地址。

在某些情况下，8086系统还需要在内存单元中存储32位的地址指针，此时，32位地址指针中的段基值和偏移地址作为两个16位字数据分别存入连续的4个内存单元，其中，偏移地址存入地址较小的单元，段基值存入地址较大的单元。

内存中数据格式

数据类型包括：
字节数据：8 位二进制数;
字 数 据：16位二进制数;
双字数据：32位二进制数;
地址指针：32位逻辑地址。



规则字与非规则字

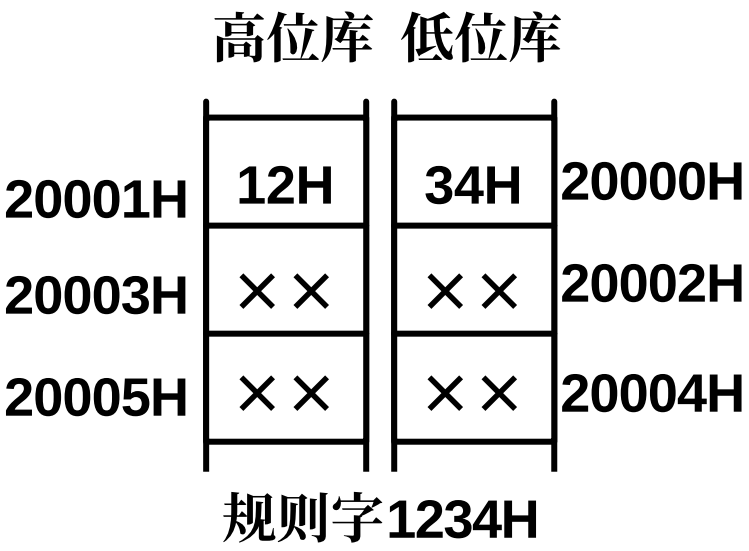
▲8086系统中将1 M字节的内存单元均分为两个存储分体，每个分体包括512 K个单元，分别称为**高位库**和**低位库**。**低位库**中的所有单元地址为**偶数**，**高位库**中的所有单元地址为**奇数**。每个分体的数据线分别与系统数据总线的**高8位**和**低8位**相连。

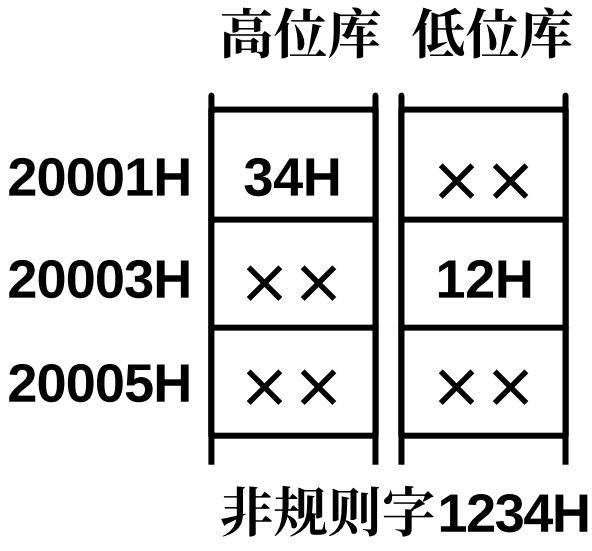
▲8086CPU读写16位字数据时，总是从低位库开始。

▲定义从低位库地址开始存放的字数据为**规则字**；即从高位库开始存放字数据为**非规则字**。

▲CPU读写内存中的规则字只需**一个**总线周期，而读写非规则字则需要**两个**总线周期。

▲为了不影响程序的执行速度，字数据在内存中应尽量按规则字格式存放。


高位库 低位库
20001H
12H
34H
20000H
20003H
××
××
20002H
20005H
××
××
20004H
规则字1234H


高位库 低位库
20001H
34H
××
20003H
××
12H
20005H
××
××
非规则字1234H

三、堆栈

▲堆栈是按照“后进先出”原则进行读/写访问的一段由“高到低”生成的特殊内存区域,用于暂存数据。

▲用作堆栈的区域称为堆栈段最多包含64 K个单元。堆栈段在内存中的位置由堆栈段寄存器SS和堆栈指针SP来指示。SS中存放堆栈段的首地址,SP中存放栈顶单元的偏移地址。

▲ 8086CPU规定堆栈操作总是按字进行。

▲入栈和出栈操作遵循“先进后出”(LIFO)的原则。

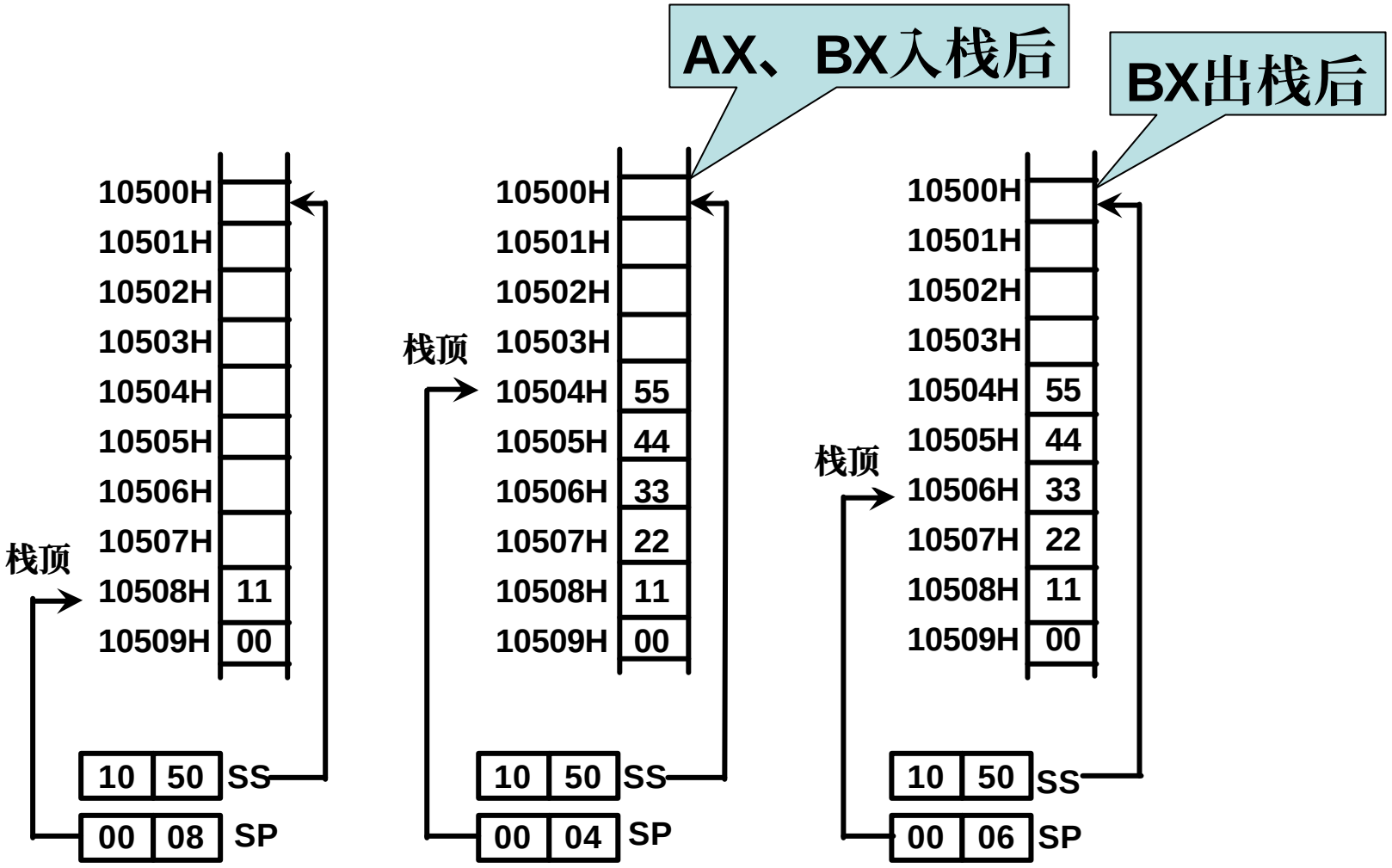
▲ 堆栈指针SP——堆栈的操作工具。

堆栈操作

- 在堆栈操作中,SP的内容随着入出栈操作的进行而不断变化,使得堆栈栈顶是“浮动”的。
- “后进先出(LIFO)”原则。
- 压栈(PUSH)与弹出(POP)。
- 8086CPU栈顶单元数据总是有效(即所谓的“实栈顶”)。即SP指向的存储单元已存放有信息。
- 8086CPU堆栈指针SP压栈时“负向”调整(–2),弹出时+2。
- 堆栈段与栈底地址。

堆栈操作举例

假设现在需要将AX＝2233H和BX＝4455H依次推入堆栈。执行入栈操作前的情况是SS＝1050H，SP＝0008H，因此堆栈段首地址为10500H，当前栈顶单元地址为SS×16 + SP＝10508H。在完成上述操作后执行出栈操作，将当前栈顶单元的内容弹出到BX中。操作情况如下图所示。


AX、BX入栈后
BX出栈后
栈顶
10500H
10501H
10502H
10503H
10504H
10505H
10506H
10507H
10508H
10509H
11
00
55
44
33
22
11
00
10 50 SS
00 08 SP
00 04 SP
00 06 SP

I/O资源

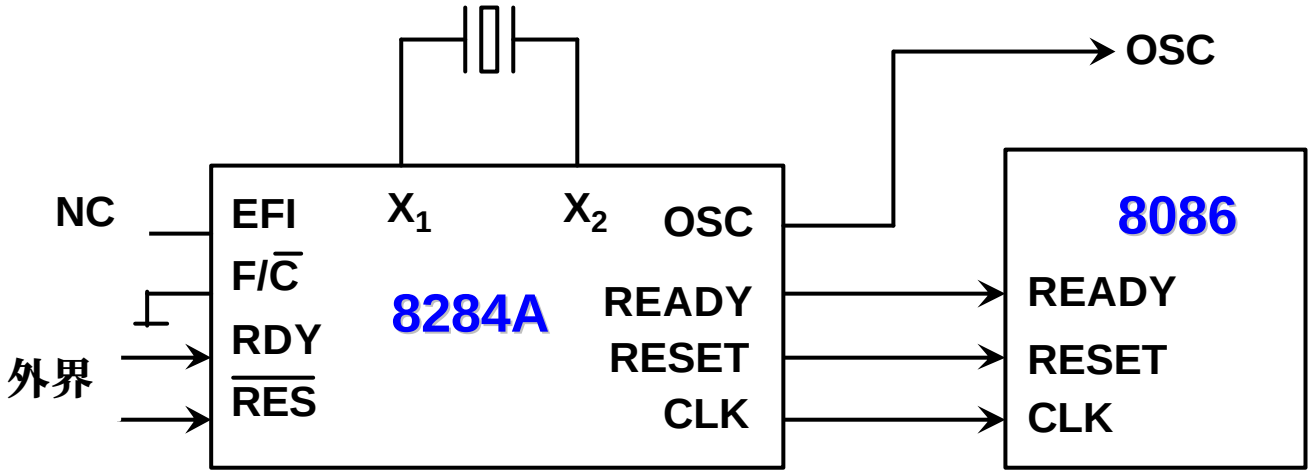
- 8086CPU采用独立编址的I/O端口。
- 有独立于存储器的操作信号。
- I/O空间2^{16}，有16条I/O端口地址线。

总线周期简介

几个术语

- **时钟周期**——时钟信号振荡周期;
- **总线周期**——CPU通过总线读/写一次数据所需时间;
- **指令周期**——CPU执行一条指令所需时间。

时钟及时钟信号发生器



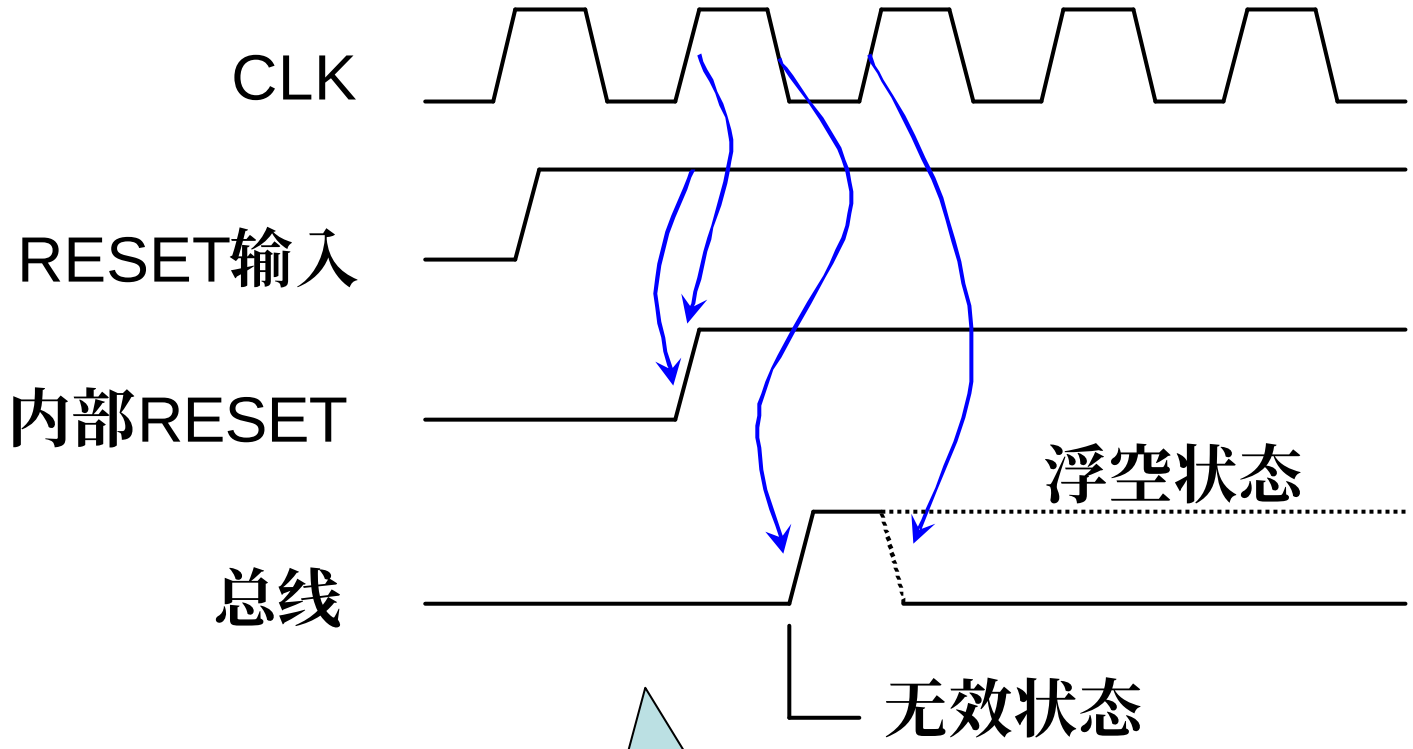
系统的复位

8086 CPU的RESET引脚用于启动或复位系统。

当8086在RESET引脚上检测到一个脉冲的正跳变时，就停止正在进行的所有操作，处于初始化状态，直到RESET信号变为低电平。

寄存器	状 态
标志寄存器F	清 除
IP	0000H
CS	FFFFH
DS	0000H
SS	0000H
ES	0000H
指令队列缓冲器	清 除

复位操作时序



8086的复位操作时序