Assembly Programming

第三、四周 计算机组织结构

庞彦

yanpang@gzhu.edu.cn

Fall 2023

80X86微处理器

FAIRCHILD SEMICONDUCTOR®

“叛逆八人帮”(traitorous eight)

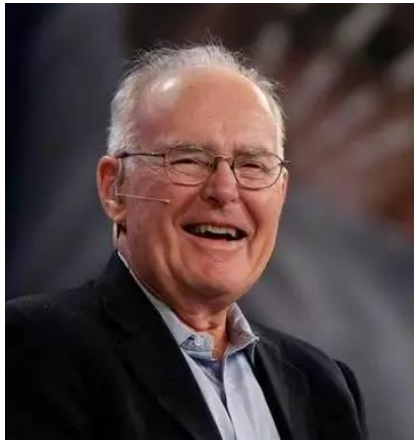
从左到右：摩尔、罗伯茨、克莱纳、诺伊斯、格里尼克、布兰克、赫尔尼、拉斯特

摩尔定律（1964）：

当价格不变时，集成电路上可容纳的晶体管数目，约每隔18个月便会增加一倍，性能也将提升一倍。

罗伯特·诺伊斯

1968



戈登·摩尔

80X86微处理器

Intel 微处理器历史（1）

Intel 4004 (1971)

Intel 8008 (1972)

Intel 8080 (1973)

Intel 8086和8088 (1978)

Intel 80286 (1982)

Intel 80386 (1986)

Intel 80486 (1989)

Intel Pentium (1993)

Intel Pentium Pro (1995)

Intel Pentium II (1997)

Intel Celeron (1998.3)

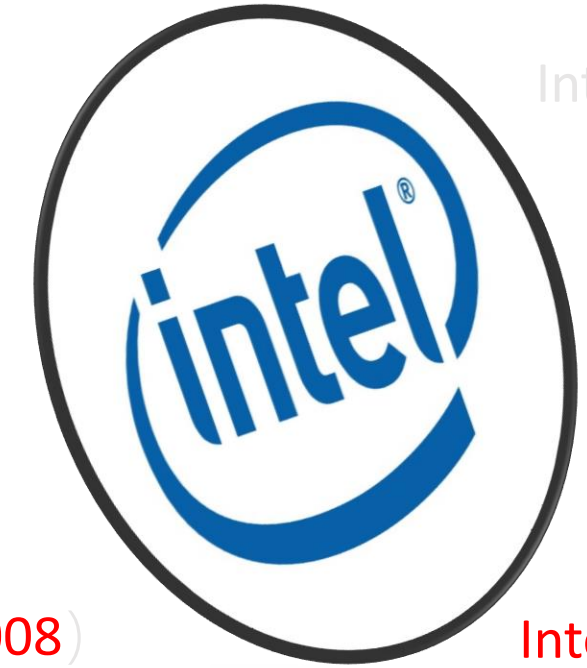
80X86微处理器

Intel 微处理器历史（2）

Intel Celeron 300A (1998.8)

Intel Core i3 (2010)

Intel Pentium III (1999)



Intel Core i5 (2009)

Intel Pentium 4 (2000)

Intel Core i7 (2008)

Intel Pentium M (2003)

Intel Atom（凌动,2008）

Intel Pentium D (2005)

Intel Core 2 Duo(酷睿,2006)

80X86微处理器

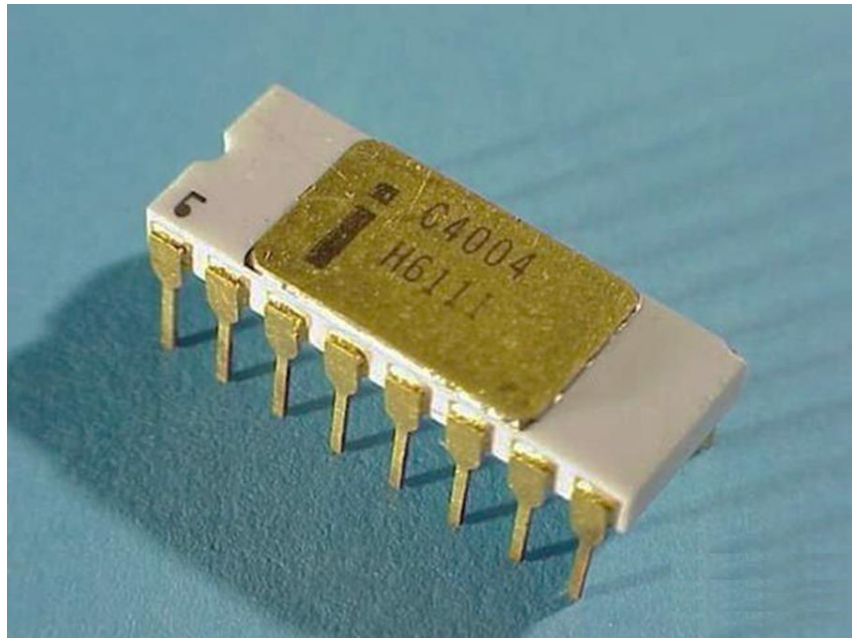
Intel 4004 微处理器

发布时间：英特尔在1971年11月15日向全球市场推出。

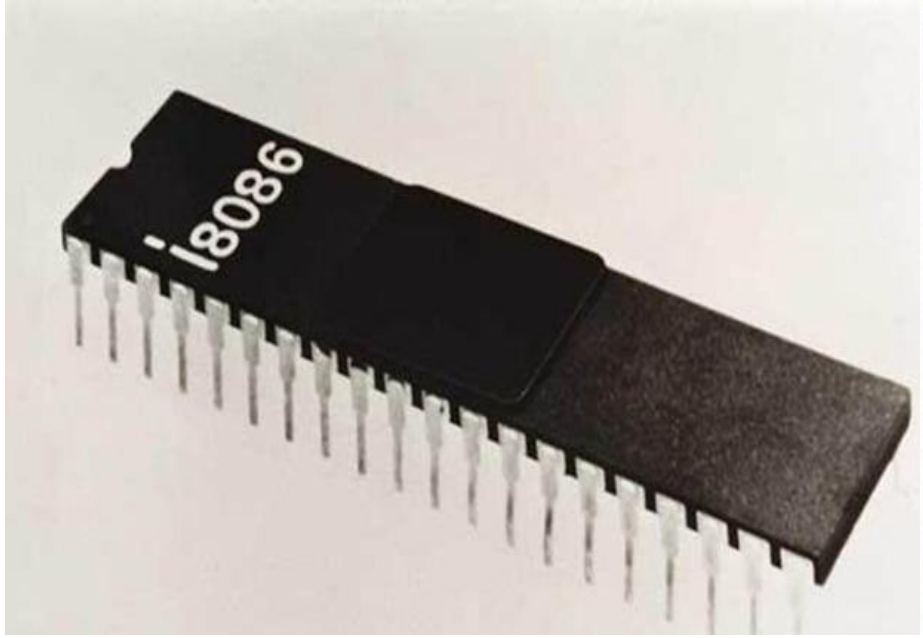
其晶体管数目：约为2250颗。

特征：

- 频率/前端总线 : 108KHZ/ 0.74MHz (4bit)
- 封装/针脚数量 : 陶瓷DIP / 16针
- 核心技术/晶体管数量: 10微米 / 2250



历史意义：世界上第一款商用处理器



发布时间：1978。
晶体管数量：约为29000。
特征：

- 主频：4.77MHZ
- 16位内部数据总线+16位外部数据总线
- 寻址范围1M

意义：8086/8088被广泛应用于IBM PC及一些兼容机，微机时代真正开始。

发布时间：1985年10月17日
晶体管数量：约为27.5万颗。
特征：

- 主频：12.5MHZ
- x86处理器首款32位系统
- 首次采用高速缓存（外置）解决内存速度瓶颈问题
- 运算速度达到了前代产品的数倍
- 寻址范围4GB，并可以管理64TB的虚拟存储空间

发布时间：1993年。
晶体管数量：约为310万颗。
特征：
- 主频：166MHZ（最高200MHZ）
- 内置了16K的一级缓存
- 制造工艺优良，可超频性很好

历史意义：Pentium是Intel首个放弃利用数字来命名的处理器产品，在微架构上取得突破。

发布时间：1998年。
晶体管数量：约为800万颗。
特征：
- 主频：266MHZ
- 开始没有2级缓存，后因性能太差加入了128K或256K的L2缓存
- 经典款：赛扬300A

历史意义：面向低端市场，和AMD竞争。

发布时间：2006年。
晶体管数量：2.91亿个晶体管。
特征：

- 主频：2.6GHZ
- 双核
- 其E6700 2.6GHZ型号比先前推出最强的Intel Pentium D 960，在性能方面提升了40%，省电效率也增加40%

历史意义：酷睿改变了以Pentium命名处理器的传统，以后再也没有奔腾了。

发布时间：2008年11月。
晶体管数量：7.74亿
特征

- 主频：2.66~3.2GHZ
- 64位4核
- 其性能是前代core 2的3倍左右

历史意义：Intel官方正式确认，基于全新Nehalem架构的新一代桌面处理器将沿用“Core”（酷睿） 名称，沿用x86-64指令集，并以Intel Nehalem微架构为基础，取代Intel Core 2系列处理器。

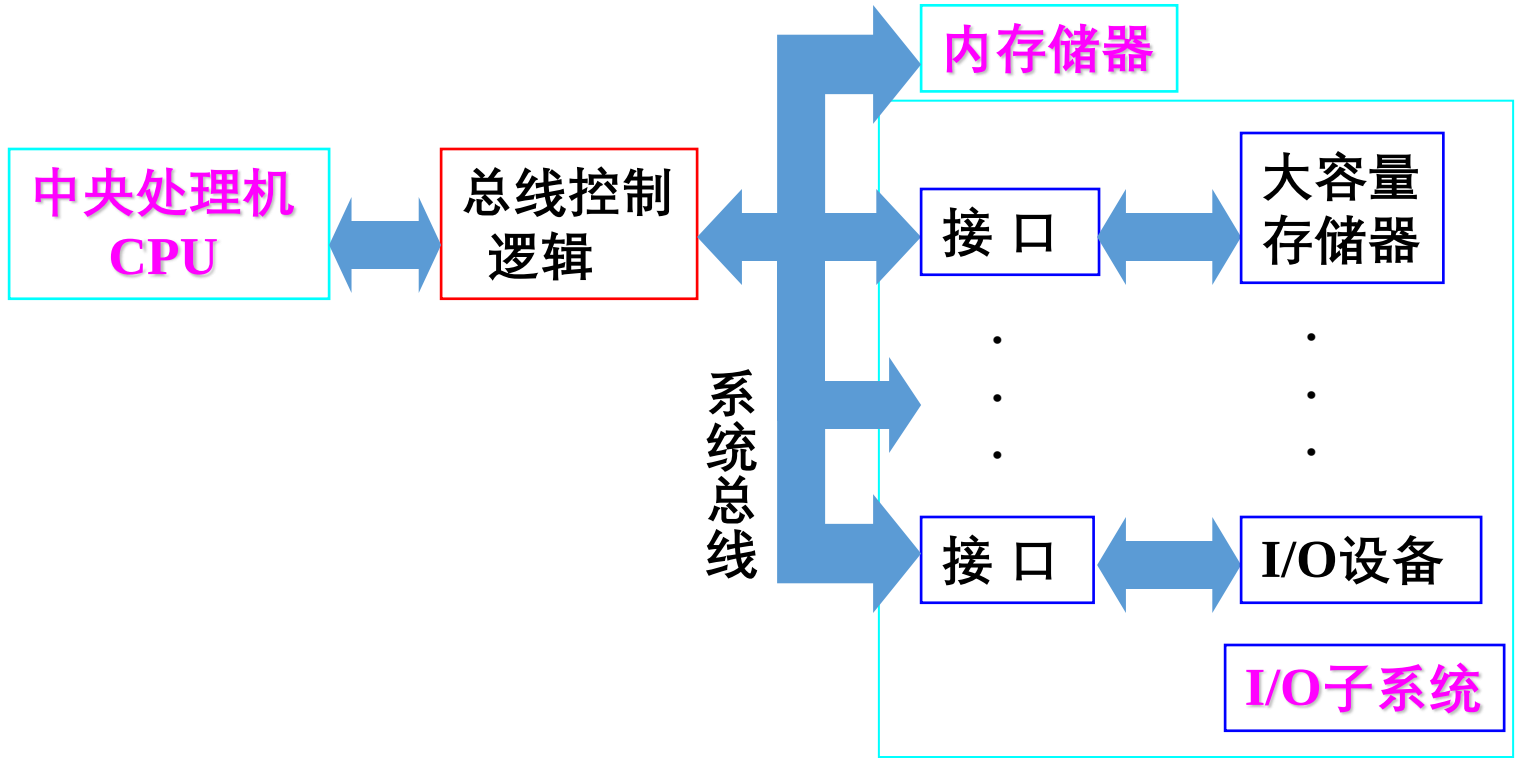
80X86微处理器

型号及年份	字长	主频MHz	数据总线宽度	外部数据总线	地址总线宽度	寻址空间	cache
8086(1978)	16	4.77	16	16	20	1M	无
8088(1979)	16	4.77	16	8	20	1M	无
80286(1982)	16	6~20	16	16	24	16M	无
80386(1986)	32	12.5~33	32	32	32	4G	有(很少)
80486(1989)	32	25~100	32	32	32	4G	8KB
Pentium(1993)	32	60~166	64	64	32	4G	16KB
Pentium Pro(1995)	32	150~200	64	64	36	64G	16KB 256K(二级)
Pentium II(1997)	32	233~333	64	64	36	64G	32KB 512K(二级)

- 汇编语言简介
- 80X86微处理器
- 基于微处理器的计算机系统
  - 硬件
  - 软件
- 中央处理机
- 存储器

基于微处理器的计算机系统

◆ 硬件：三个主要组成部分，用系统总线连接。



◆ **软件：**系统软件 & 用户软件

基于微处理器的计算机系统

典型的微型计算机硬件主要由中央处理机、存储器、系统总线、I/O接口电路和I/O设备组成。

- 中央处理机（CPU）：是微型计算机的核心部件，芯片内集成了运算器、控制器和寄存器组，用来执行程序指令，完成所有的算术和逻辑运算及全机的控制工作。
- 存储器：是微型计算机的重要组成部件，用来存放程序和数据。微型计算机的存储器分为“主存”和“辅存”两类。
  - 主存也称内存，CPU可以通过总线直接存取，微型计算机的主存储器主要都是采用半导体存储器，按照读写方式的不同，分为只读存储器ROM（Read Only Memory）和随机存取存储器RAM（Random Access Memory）两种类型；
  - 辅存也称外存，如磁盘、磁带、光盘等，CPU通过I/O接口对其进行存取，它的容量比内存大很多，但存取信息的速度要比内存慢得多。一般程序（包括数据）是存放在外存中的，只有当运行时，才把它从外存传送到内存的某个区域，再由CPU控制执行。

基于微处理器的计算机系统

- 总线：是指传送信息的一组公共导线，是计算机各功能部件之间进行信息传输的通道。CPU、存储器和I/O接口电路之间通过数据总线、地址总线和控制总线相连，这三组总线统称为系统总线。
  - 数据总线是用来传送数据信息的。该总线是双向总线。数据总线的位数（也称宽度）决定了一次能够传送数据的位数。
  - 地址总线是传送地址信息的。该总线是单向总线，用来输出CPU要访问的内存单元或I/O端口的地址。地址总线位数决定了CPU可以直接寻址的内存空间的大小，对于n条地址总线，可直接寻址的内存范围为2^n。例如，8086的地址总线为20位，可寻址的最大内存空间为2^{20}B，即1MB。
  - 控制总线是用来传送控制信息的。这组信号线比较复杂，有的是微处理器送往存储器和I/O接口的控制信号，如读写控制信号、中断响应信号等；有的是将外界的请求或联络信号送往微处理器，如中断请求信号、准备就绪信号等。
  - 采用标准的总线结构是微型计算机系统的显著特点之一。采用了总线结构后，一个部件只要符合总线标准，就可以连接到采用这种总线标准的系统中，使系统的功能可以很方便地得以扩展。

基于微处理器的计算机系统

- I/O接口电路：由于微机的外部设备种类繁多、工作原理各异，它们不能直接连到微机系统总线上实现与主机通信，必须经过中间电路再与系统相连，这部分电路被称为I/O接口电路。
  - 主要完成数据缓冲、信号变换、以及与CPU联络等工作。
  - 为便于主机访问外设，将I/O接口电路中每个寄存器统一编号，称为I/O端口地址或端口号。80x86的I/O地址空间为64KB，可寻址65536个不同的I/O地址，端口地址的范围是0000H~FFFFH。
- I/O设备：是指微型计算机配备的输入输出设备，也称外围设备（简称外设），是微型计算机必不可少的组成部分。对外设的管理是汇编语言的重要应用之一。



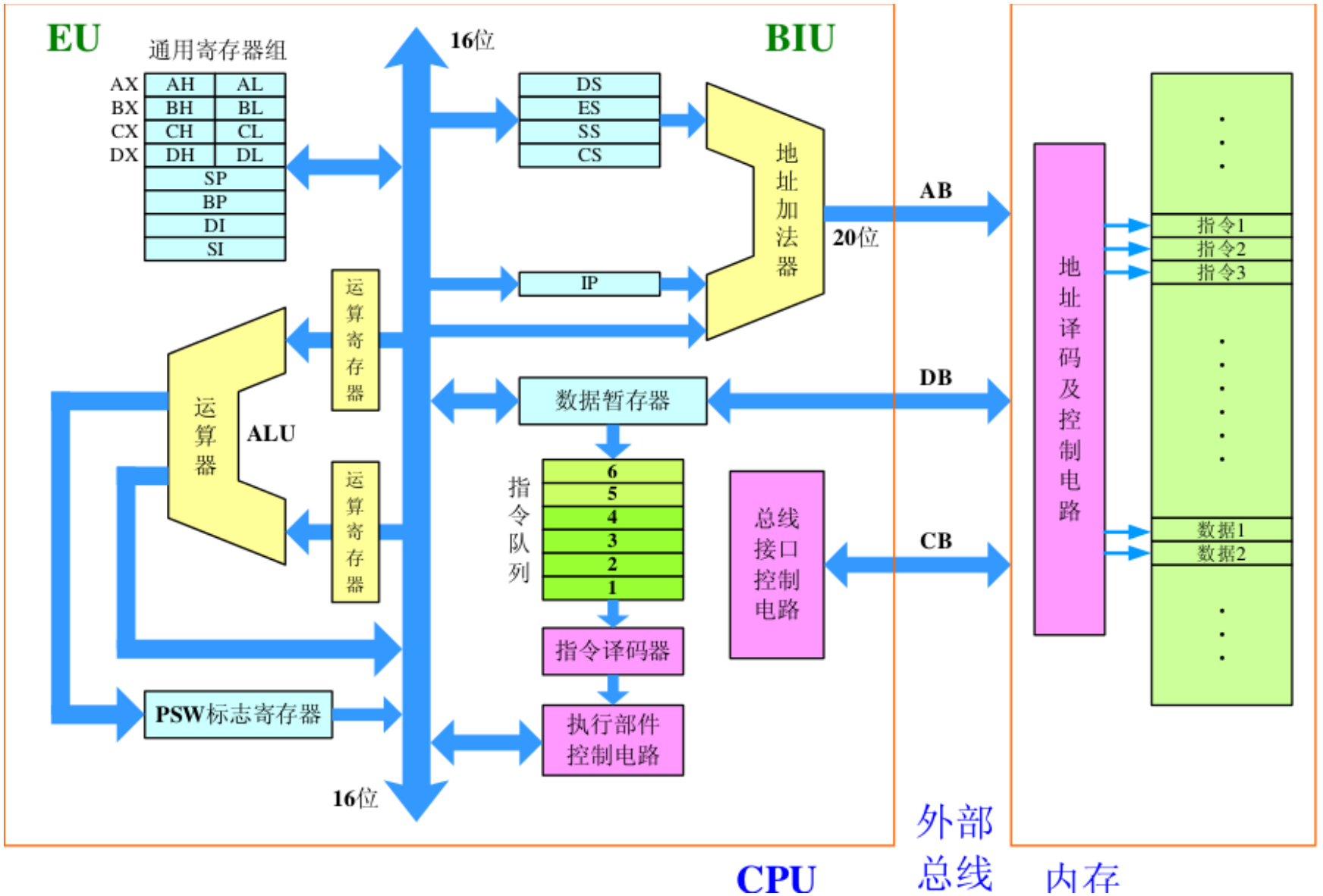

基于微处理器的计算机系统

软件：

- 没有配置软件的计算机，什么工作也不能做，软件是计算机系统的重要组成部分。微型计算机的软件分为系统软件和应用软件两大类。
  - 系统软件是面向所有用户的一类软件，通常包括：操作系统（DOS、Windows、Linux等）、语言翻译程序、诊断调试程序、I/O驱动程序等。系统软件的核心是操作系统，所有应用的程序都是在操作系统构筑的平台上运行的。
  - 应用软件主要是指用户围绕某项应用编写的各种程序。
  - 汇编语言是用来开发系统软件或应用软件的工具之一。

- 汇编语言简介
- 80X86微处理器
- 基于微处理器的计算机系统
  - 硬件
  - 软件
- 中央处理机
- 存储器

中央处理机


EU
通用寄存器组
AX
BX
CX
DX
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
DI
SI
16位
BIU
DS
ES
SS
CS
IP
地址加法器
20位
AB
运算寄存器
运算器
ALU
运算寄存器
数据暂存器
DB
指令队列
6
5
4
3
2
1
指令译码器
执行部件控制电路
PSW标志寄存器
16位
总线接口控制电路
CB
地址译码及控制电路
指令1
指令2
指令3
数据1
数据2
CPU
外部总线
内存

中央处理机

8086CPU从功能上可分为两部分，即总线接口部件BIU(Bus Interface Unit)和执行部件EU(Execution Unit)。

总线接口部件（BIU）：由段寄存器、指令指针寄存器、地址加法器、指令队列和输入输出控制电路等组成。BIU是8086与系统总线的接口，负责CPU与存储器、I / O端口传送数据。

(1)段寄存器：包括4个16位的段寄存器，代码段寄存器CS、数据段寄存器DS、附加段寄存器ES，堆栈段寄存器SS。

(2)16位指令指针寄存器IP：用来存放下一条要执行指令在代码段中的偏移地址。

(3)20位的地址加法器：用来形成20位的物理地址。

(4)6字节的指令队列：用于存放从内存中取来的指令，按照先进先出的方式工作，并按顺序送到EU中执行。其操作遵循下列原则：

a) 每当指令队列缓冲器中存满一条指令后，EU就立即开始执行。

b) 指令队列缓冲器只要有2个空字节时，BIU就会自动把指令取到指令队列中，直到填满为止。

c) 在执行转移、调用或返回指令时，接下去要执行的指令不再是程序中紧接着排列的那条指令了，这样，指令队列中已经装入的指令就没用了，则要清除指令队列缓冲器，并要求BIU从新地址开始取指令填入指令队列缓冲器。

(5)总线接口控制电路：将CPU的内部总线与系统总线相连，是CPU与内存单元或I/O端口交换数据的必经之路。

中央处理机

执行部件EU：由算术逻辑部件(ALU)、通用寄存器、标志寄存器和执行部件控制电路等组成，它负责指令的执行和数据的运算。

（1）通用寄存器：包括4个16位的数据寄存器AX、BX、CX、DX和4个16位的指针与变址寄存器SI、DI与SP、BP。

（2）标志寄存器(FR)：它是一个16位的寄存器，用来反映CPU运算的状态特征和存放控制标志。

（3）算术逻辑部件(ALU)：用来完成8位或16位二进制算术和逻辑运算。

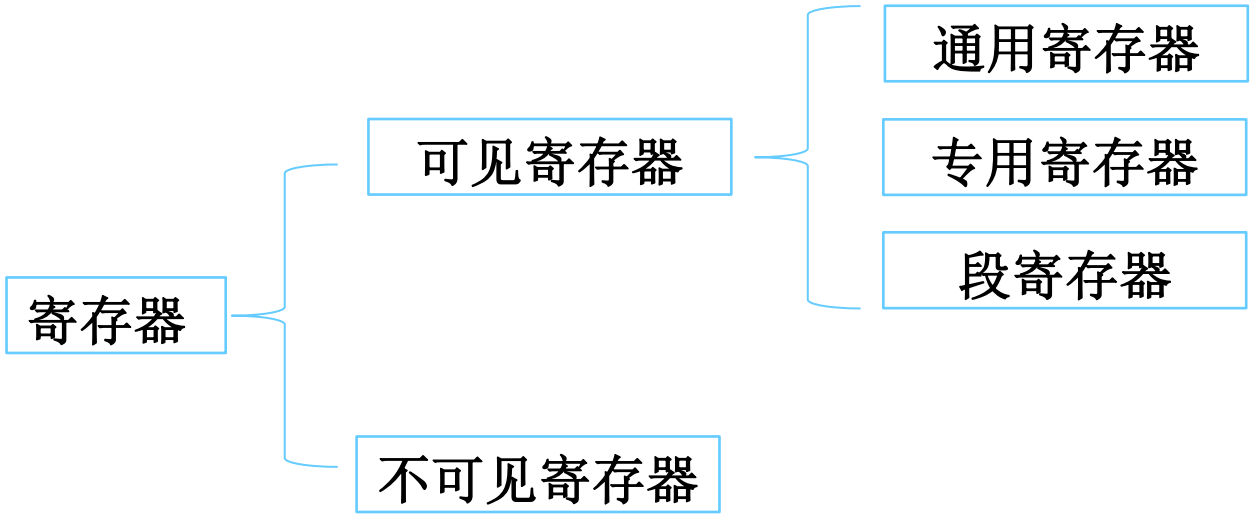
（4）执行部件控制电路：负责从总线接口部件的指令队列缓冲器中取指令，并对指令进行译码，根据指令要求向执行部件内部个部分发出控制命令以完成各条指令的功能。

8086/8088CPU的总线接口部件和执行部件并不是同步工作的，它们相互独立，分别完成各自操作，在执行部件执行指令的同时，总线接口部件可取下面一条或几条指令，总线接口部件和执行部件这种并行操作的特点，可以提高系统的运行速度，从而提高了工作效率。

中央处理机

8086寄存器组

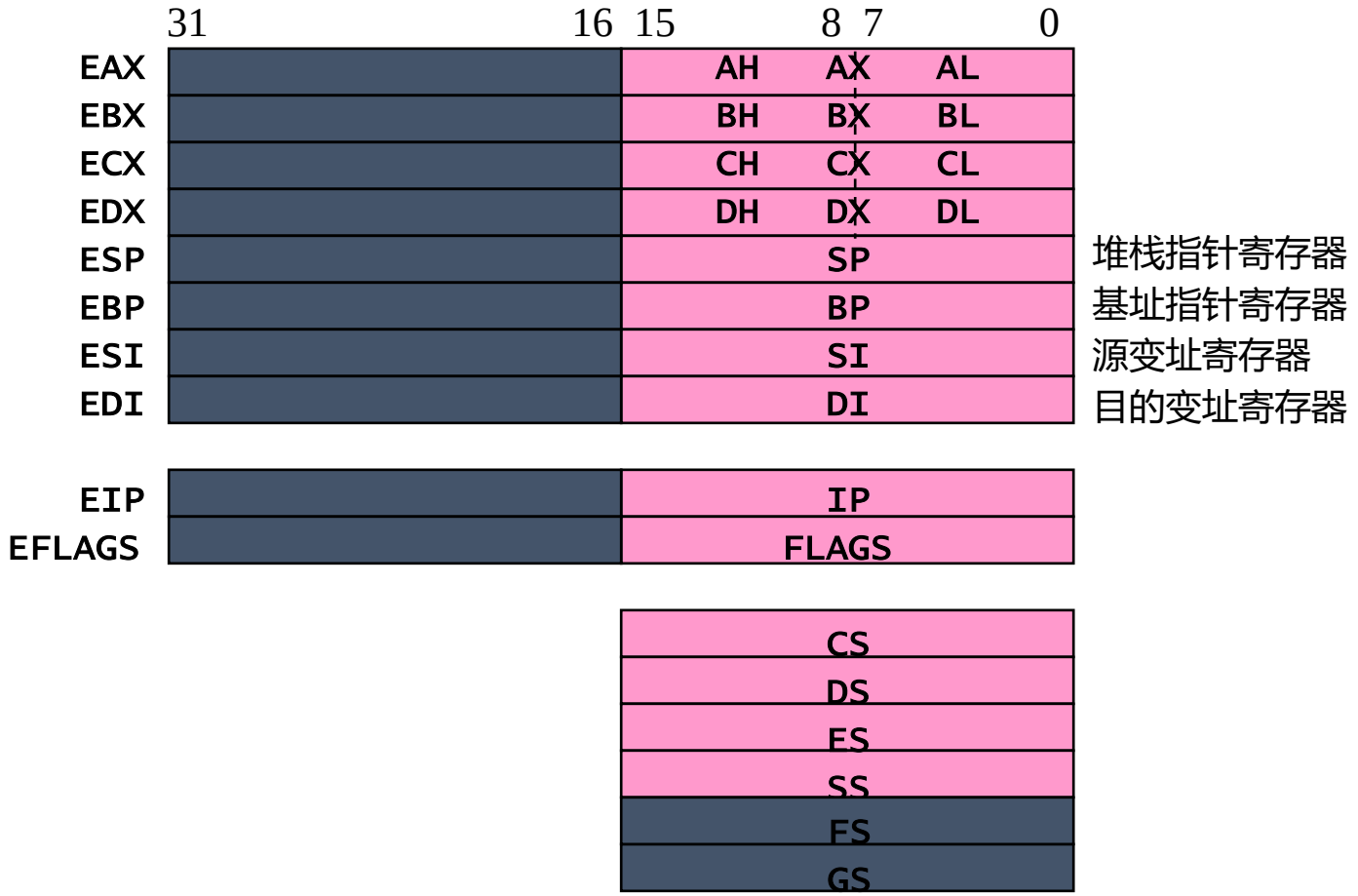
寄存器是CPU内部重要的数据存储资源。



寄存器（可见寄存器）是**汇编程序员能直接使用的硬件资源之一。由于寄存器的存取速度比内存快，所以，在用汇编语言编写程序时，要尽可能充分利用寄存器的存储功能。**

中央处理机

80x86的可见寄存器组



通用寄存器
专用寄存器
段寄存器

8086 /8088 /80286 **寄存器**

中央处理机

1.通用寄存器组

（1）数据寄存器

数据寄存器共有4个：AX、BX、CX、DX，用来保存操作数或运算结果等信息。

- AX(Accumulator)寄存器称为累加器。使用频度最高：用于算术、逻辑运算以及与外设传送信息等。
- BX(Base Register)寄存器称为基址寄存器：常用于存放存储器地址。
- CX(Count Register)寄存器称为计数器：一般作为循环或串操作等指令中的隐含计数器。在位操作中，当移多位时，要用CL来指明移位的位数；
- DX(Data Register)寄存器称为数据寄存器：常用来存放双字数据的高16位，在进行乘、除运算时，它可作为默认的操作数参与运算，亦可存放外设端口地址。

中央处理机

（2） 变址寄存器

寄存器SI和DI称为变址寄存器(Index Register)，主要用于存放某个存储单元的偏移地址。

SI是源变址寄存器，DI是目的变址寄存器

1) SI和DI一般与数据段寄存器DS联用，用来确定数据段中某存储单元的地址。
2) 在字符串操作中。

   SI与DS段寄存器联用，用来确定源操作数的地址。

   DI与ES段寄存器联用，用来确定目的操作数的地址。
3) SI和DI都具有自动增量或减量的功能。
4) 变址寄存器不可分割成8位寄存器。

中央处理机

（3）指针寄存器

寄存器BP和SP称为指针寄存器(Pointer Register)，主要用于存放堆栈内存储单元的偏移量，用它们可实现多种存储器操作数的寻址方式(在下一章有详细介绍)，为以不同的地址形式访问存储单元提供方便。

SP(Stack Pointer)为堆栈指针寄存器，用于存放当前堆栈段中栈顶的偏移地址；
BP (Base Pointer)为基址指针寄存器，用于存放堆栈段中某一存储单元的偏移地址。

2.专用寄存器： IP、FLAGS

（1）指令指针寄存器 IP(Instruction Pointer)

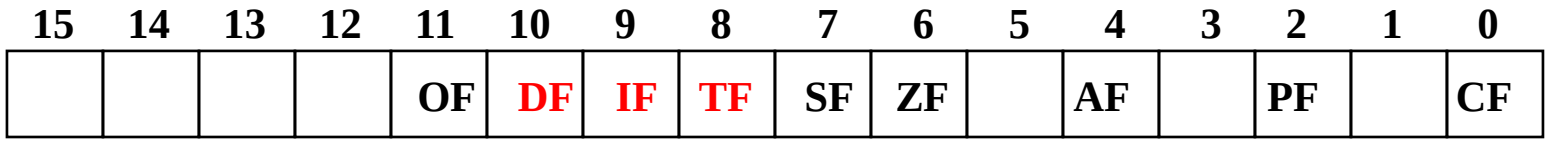
它总是保存下一次将要从主存中取出指令的偏移地址。

IP与CS段寄存器联用，可以确定下一条要取的指令的物理地址，因此IP是很重要的控制寄存器，用于控制程序的执行流程。

在目标程序运行时，IP的内容由微处理器硬件自动设置，程序不能直接访问IP，但一些指令却可改变IP的值，如转移指令、子程序调用指令等。

32位CPU把指令指针扩展到32位，并记作EIP，EIP的低16位与先前CPU中的IP作用相同。

(2) 标志寄存器 (FLAGS / PSW : Program Status Word)



状态标志:
OF 溢出标志
SF 符号标志
ZF 零标志
AF 辅助进位标志
PF 奇偶标志
CF 进位标志

控制标志:
DF 方向标志
I F 中断标志
TF 陷阱标志

注：指令的执行与标志有很大关系。

状态标志：用来记录程序运行结果的状态信息，许多指令的执行都将相应地设置它。

控制标志：可由程序根据需要用指令设置，用于控制处理器执行指令的方式。

中央处理机

进位标志CF（Carry Flag）

当运算结果的最高位（第7或15位）有进位（加法）或借位（减法）时，进位标志置1，即CF = 1；否则CF = 0。

3AH + 7CH =　　CF = ?

AAH + 7CH =　　CF = ?

使用场合：多字节运算，无符号数比较大小和移位操作时，用到该标志。

零标志ZF（Zero Flag）

- 若运算结果为0，则ZF = 1，否则ZF = 0。
- 例如：

3AH + 7CH =　　ZF = ?

84H + 7CH =　　ZF = ?

使用场合：需判断运算结果是否为0。

中央处理机

进位标志CF（Carry Flag）

当运算结果的最高位（第7或15位）有进位（加法）或借位（减法）时，进位标志置1，即CF = 1；否则CF = 0。

3AH + 7CH = B6H，没有进位：CF = 0

AAH + 7CH = （1）26H，有进位：CF = 1

使用场合：多字节运算，无符号数比较大小和移位操作时，用到该标志。

零标志ZF（Zero Flag）

- 若运算结果为0，则ZF = 1，否则ZF = 0。
- 例如：

3AH + 7CH = B6H，结果不是零：ZF = 0

84H + 7CH = （1）00H，结果是零：ZF = 1

使用场合：需判断运算结果是否为0。

中央处理机

符号标志SF（Sign Flag）

运算结果最高位为1，则SF = 1（结果为负）；否则SF = 0 （结果为正）。用于有符号数的运算。

3AH + 7CH = SF = ?

84H + 7CH = SF = ?

奇偶标志PF（Parity Flag）

当运算结果中1的个数为零或偶数时，PF = 1；否则PF = 0。

3AH + 7CH = =

结果中有?个1，是?数：PF = ?

注意：PF标志仅反映最低8位中“1”的个数是偶或奇，即使是进行16位字操作。

中央处理机

符号标志SF（Sign Flag）

运算结果最高位为1，则SF = 1（结果为负）；否则SF = 0 （结果为正）。用于有符号数的运算。

3AH + 7CH = B6H，最高位D7 = 1：SF = 1

84H + 7CH = （1）00H，最高位D7 = 0：SF = 0

奇偶标志PF（Parity Flag）

当运算结果中1的个数为零或偶数时，PF = 1；否则PF = 0。

3AH + 7CH = B6H = 10110110B

结果中有5个1，是奇数：PF = 0

注意：PF标志仅反映最低8位中“1”的个数是偶或奇，即使是进行16位字操作。

中央处理机

溢出标志OF（Overflow Flag）

若算术运算的结果有溢出，则OF=1；否则 OF＝0。

例如：

3AH ＋ 7CH＝　　OF ＝？

AAH ＋ 7CH＝　　OF ＝？

中央处理机

溢出标志OF（Overflow Flag）

若算术运算的结果有溢出，则OF=1；否则 OF = 0。

例如：

3AH + 7CH = B6H，没有溢出：OF = 0

AAH + 7CH = （1）26H，产生溢出：OF = 1

方向标志DF（Direction Flag）

用于串操作指令中，控制地址的变化方向：

- 设置DF = 0，串操作的存储器SI和DI地址自动增加；
- 设置DF = 1，串操作的存储器SI和DI地址自动减少。
- CLD指令复位方向标志：DF = 0
- STD指令置位方向标志：DF = 1

```
例：ADD     AX，BX
    JO / JC ERROR
```

中央处理机

中断允许标志IF（Interrupt-enable Flag）

用于控制外部可屏蔽中断是否可以被处理器响应：

设置IF = 1，则允许中断；

设置IF = 0，则禁止中断。

CLI指令复位中断标志：IF = 0

STI指令置位中断标志：IF = 1

陷阱标志TF（Trap Flag）

用于控制处理器是否进入单步操作方式：

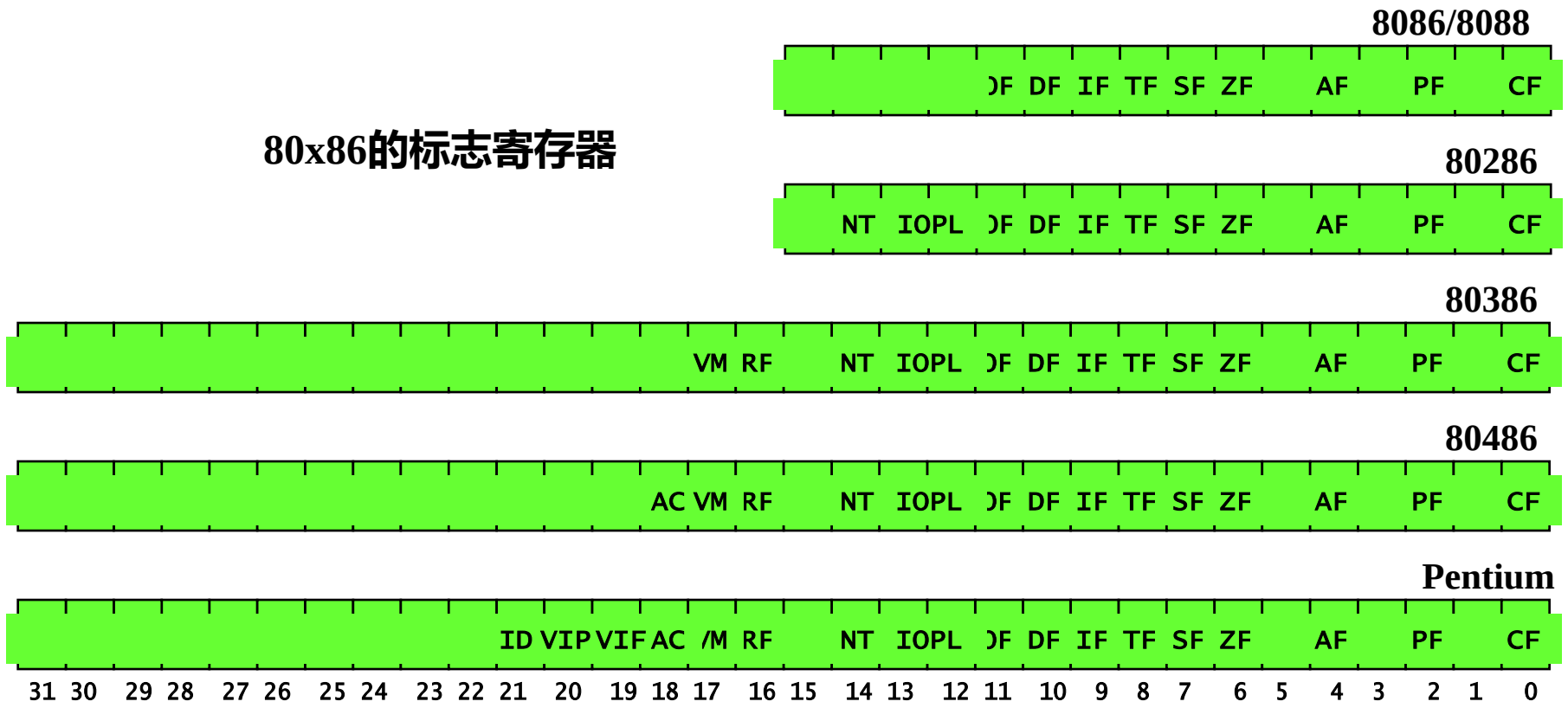
设置TF = 0，处理器正常工作；

设置TF = 1，处理器单步执行指令。

单步执行指令——处理器在每条指令执行结束时，便产生一个编号为1的内部中断。这种内部中断称为单步中断，所以TF也称为单步标志。

利用单步中断可对程序进行逐条指令的调试。

中央处理机



IOPL： I/O特权级
NT： 嵌套任务标志
RF： 重新启动标志
AC： 对准检查方式位
ID： 标识标志

VIP： 虚拟中断未决标志
VIF： 虚拟中断标志
VM： 虚拟8086模式位

中央处理机

3. 段寄存器

8086CPU的４个16位的段寄存器分别称为：代码段寄存器CS，数据段寄存器DS，堆栈段寄存器SS，附加数据段寄存器ES。80386起增加了FS、GS两个段寄存器（附加的数据段）。

段寄存器用来确定该段在内存中的起始地址。段寄存器是根据内存分段的管理模式而设置的。内存单元的物理地址由段寄存器的值和一个偏移量组合而成的，这样可用两个较少位数的值组合成一个可访问较大物理空间的内存地址。

- 处理器利用CS:IP取得下一条要执行的指令
- 处理器利用DS:EA存取数据段中某一存储单元的地址
- 处理器利用SS:SP操作堆栈顶的地址

- 汇编语言简介
- 80X86微处理器
- 基于微处理器的计算机系统
  - 硬件
  - 软件
- 中央处理机
- 存储器

存储器

存储单元的地址和内容

- 存储单元地址：**8086系统中，为了标识和存取每一个存储单元，给每个存储单元规定一个编号，这就是**存储单元地址。
- **存储单元的内容：一个存储单元中存放的信息称为该存储单元的内容。**

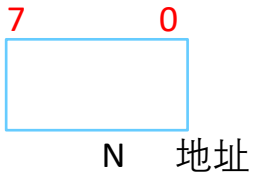
注意：

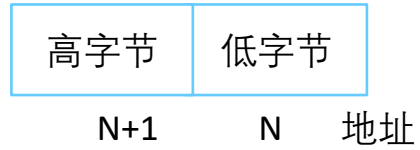
- **存储器以字节（8 bit）为编程单位**
- **每个字节单元都有唯一的地址编码**
- **地址用无符号整数来表示（编程用十六进制表示）**
- **一个字要占用相继的两个字节**
- **低位字节存入低地址，高位字节存入高地址**
- **字单元地址用它的低地址来表示**

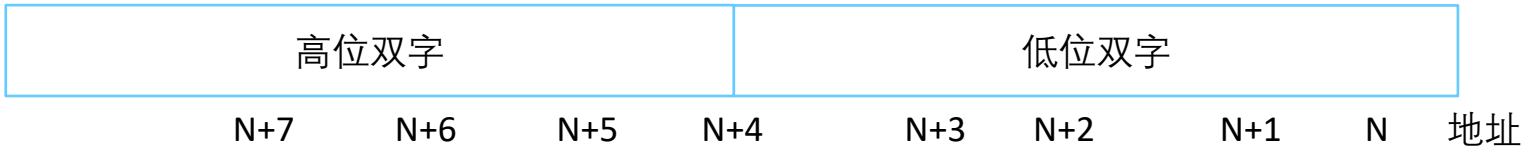
存储器

内存单元的地址和内容的概念及其关系：

- 若设M表示某内存单元的地址，那么M单元的内容可表示为(M)。
- 如果M单元存放着地址信息N，则地址N单元的内容则可表示为(N) = ((M))。

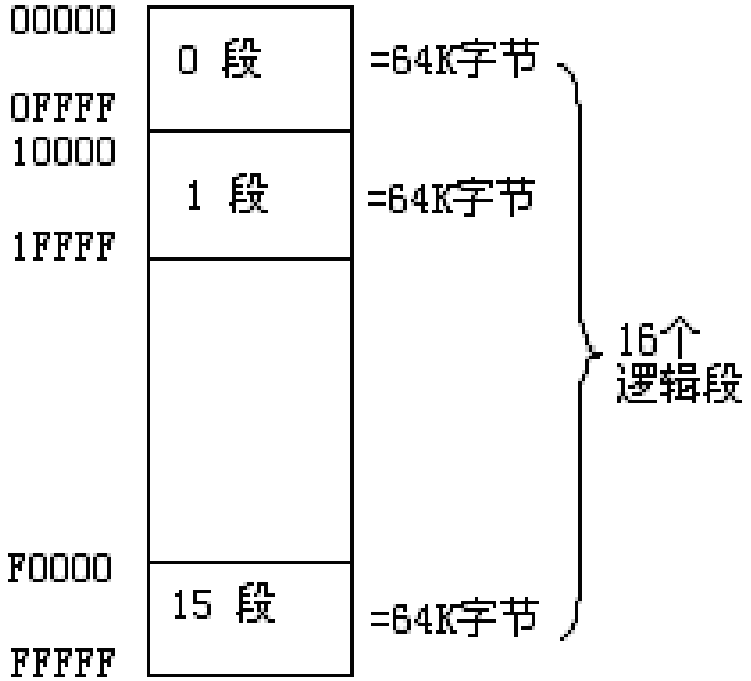






存储器

- 8086/8088CPU的地址线是20位的，它最大可寻址空间为2^{20}=1MB。
- 而8086/8088的内部寄存器是16位寄存器，16位寄存器只能寻址2^{16}=64KB。
- 为了能用16位寄存器来有效地访问1MB的存储空间，8086/8088CPU采用了内存分段的管理模式。
- 8086/8088把1MB存储空间分成若干个逻辑段，每个逻辑段最大为64KB，这样段内地址可以用16位表示。
- 系统的整个存储空间可分为16个互不重叠的逻辑段，如右图所示。



思考：
80286地址总线：24位
80386地址总线：32位
各自的寻址范围是多少？

存储器

段起始地址：段不能起于任意地址，而必须从任一小段的首地址开始。

小段：每16个字节为一小段，共有64K个小段

小段的首地址

00000 H ~ 0000F H

00010 H ~ 0001F H

00020 H ~ 0002F H

…

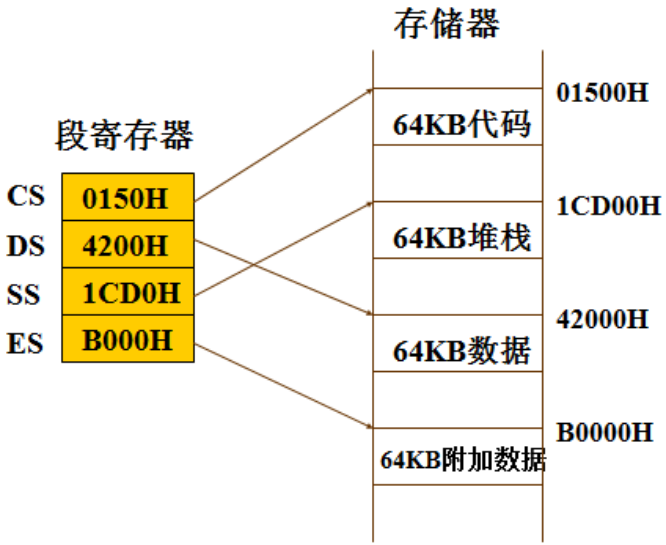
FFFF0 H ~ FFFFF H

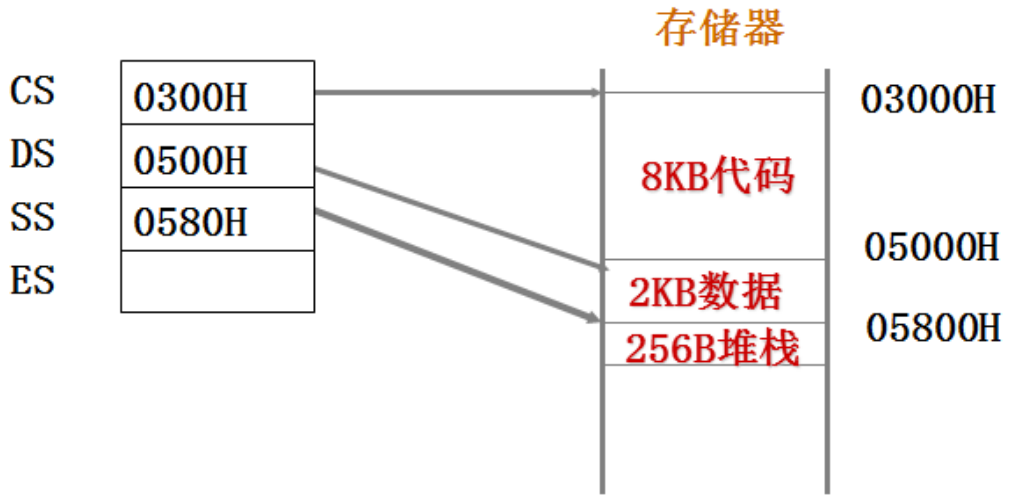
其特征是：在十六进制表示的地址中，最低位为0（即20位地址的低4位为0）。这样，省略低位0，段起始地址可以用16位数据表示，通常被保存在16位的段寄存器中。

段的大小：0-64K 范围内的任意字节。

存储器

- 每个逻辑段最大可以占用64KB存储区。实际上，可以根据实际需要来确定段的大小，它可以是1B、100B或在64KB范围内的任意字节。
- 各段也允许重叠。当然每个存储单元的内容是不允许发生冲突的，所谓重叠只是指每个段区的大小允许根据实际需要来分配，而不一定要占用64KB的空间。
- 一般情况下，各段在存储器中的分配是由操作系统完成的。但是，系统允许程序员在必要时指定所需占用的内存区。





存储器

- 在确定了某个存储单元所属的内存段后，我们也只知道其所处内存位置的范围，还不能确定其具体位置。
- 要想确定内存单元的具体位置，还必须知道该单元离该段地址有多远。
- 我们通常把存储单元的实际地址与其所在段的段地址之间的距离称为段内偏移，也可称为有效地址(EA：Effective Address)或偏移地址(Offset)等。
- 有了段地址和偏移地址，就能唯一地确定某一内存单元在存储器内的具体位置（物理地址，PA：Physical Address）。

存储器

由此可见，存储单元的物理地址分为两部分：段地址和偏移量。计算存储单元物理地址的公式可用“左移4位”和“加”运算来实现。下图是物理地址的计算示意图。

16 位 段 地 址　　0000

\+　　16 位 偏 移 地 址

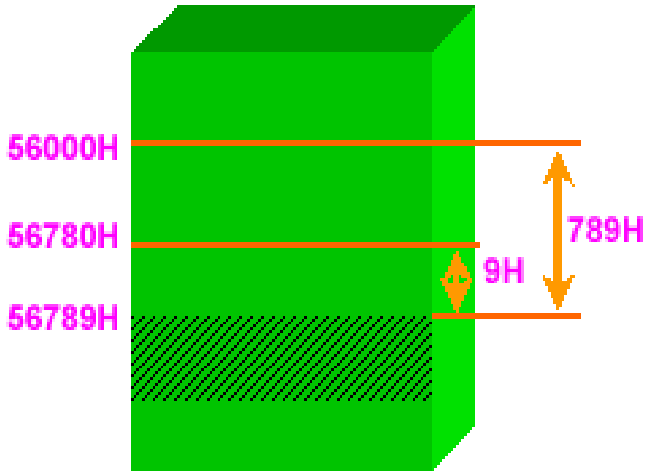
20 位 物 理 地 址

例：(DS) = 2100H, (BX) = 0500H

(PA) = 21000H+0500H

= 21500H

显然：每个存储单元只有唯一的物理地址，但它却可由不同的段地址和不同的偏移地址组成（逻辑地址）。



物理地址和逻辑地址之间的关系

存储器

存储器的逻辑地址与物理地址

逻辑地址		物理地址
段地址：偏移地址		
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

访问两次内存

存储器

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

存储器

- 1MB空间最多能分成多少个段？

- 1MB空间最少能分成多少个段？

存储器

- 1MB空间最多能分成多少个段？
  - 每隔16个存储单元就可以开始一个段
  - 所以1MB最多可以有：

$$2^{20} \div 16 = 2^{16} = 64K \text{ 个段}$$

- 1MB空间最少能分成多少个段？

存储器

- 1MB空间最多能分成多少个段？
  - 每隔16个存储单元就可以开始一个段
  - 所以1MB最多可以有：

$$2^{20} \div 16 = 2^{16} = 64K \text{ 个段}$$

- 1MB空间最少能分成多少个段？
  - 每隔64K个存储单元开始一个段
  - 所以1MB最少可以有：

$$2^{20} \div 2^{16} = 16 \text{ 个段}$$

Q&A