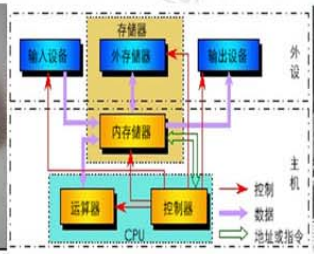
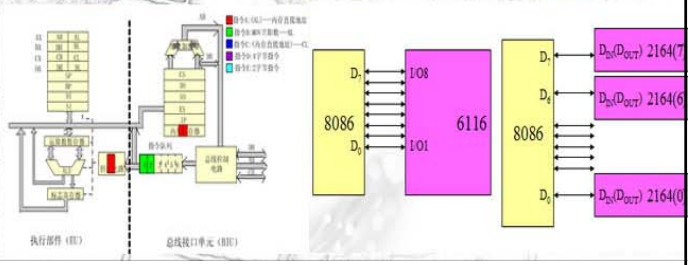
微型计算机原理及其应用

第二章　80x86微处理器





合肥工业大学计算机与信息学院　2012-02

第二章 80x86微处理器

2.1 微处理器的基本结构

1. 算术逻辑单元ALU
2. 控制器
3. 总线与总线缓冲器
4. 寄存器阵列

3

2.1.1 算术逻辑单元ALU

- 数学问题的求解可分解为算术和逻辑运算实现。
  - 在算术运算中，若符号数采用补码表示，则减法可用加法实现；乘除法可通过多次的加减和移位实现。
  - 在逻辑运算中，只要具备“与”、“或”、“非”、“异或”等功能的部件就能实现各种复杂的逻辑运算。
  - 所以，在不考虑数据信息表示方式的情况下，计算机只要具备加法、“与”、“或”、“非”等运算和移位操作功能，就能实现各种算术运算和逻辑运算。
- 算术逻辑单元(Arithmetic Logic Unit，ALU)
  - 是一个对二进制数进行算术和逻辑运算的部件。

4

2.1.1 算术逻辑单元ALU

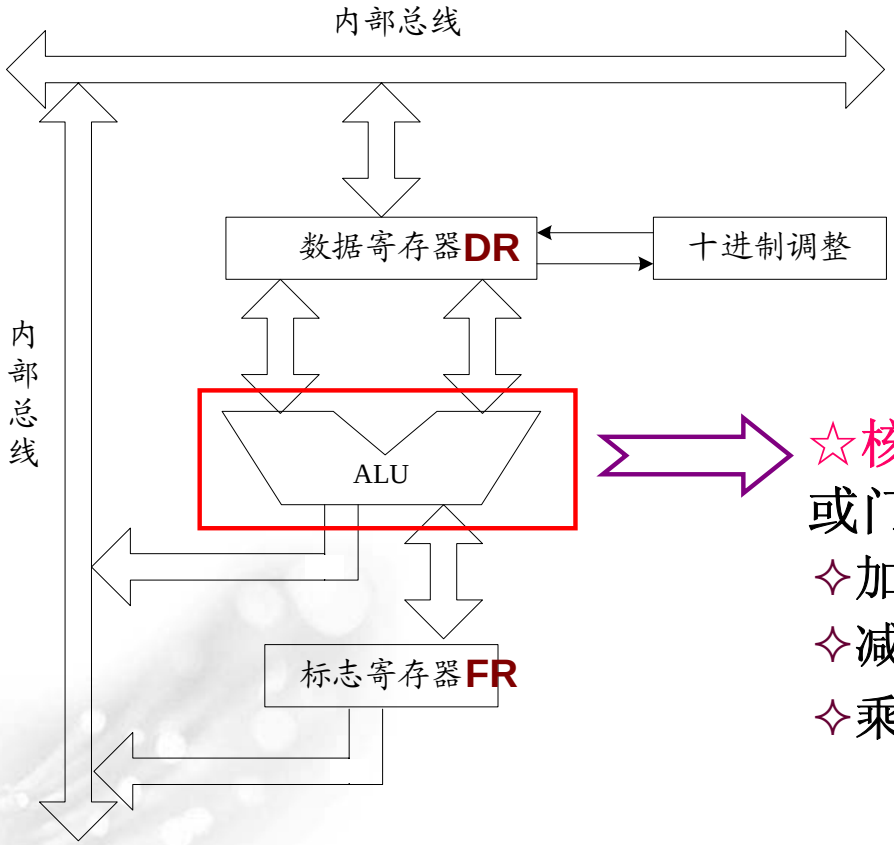
ALU的主要功能

- 硬件实现基本运算
  - 加、减、求补、与、或、非、异或、移位、BCD码运算的十进制调整等。
- 乘除法运算
  - 中低档的8位微处理器：乘除法运算是通过软件编程实现的，它由加、减、移位功能组合完成。
  - 高档的8位微处理器和16位以上的微处理器：专用的乘除法指令，其乘除法运算功能也由硬件电路来完成。
- 浮点运算
  - 在8位或16位微处理器中，所有数都采用定点数表示，浮点数由两个定点数组成，浮点运算采用软件编程实现。
  - 高性能微处理器中集成了专门的浮点处理器，并有专门的浮点运算指令。

5

2.1.1 算术逻辑单元ALU

ALU的构成与工作原理



✧数据经内部总线进入DR

✧DR的待运算数据和FR的进位标志(CF)输入ALU

✧结果送DB或DR，同时将运算结果的状态送FR保存。

☆核心：加法器（与门+或门电路）

✧加法运算

✧减法运算：补码表示➔加法

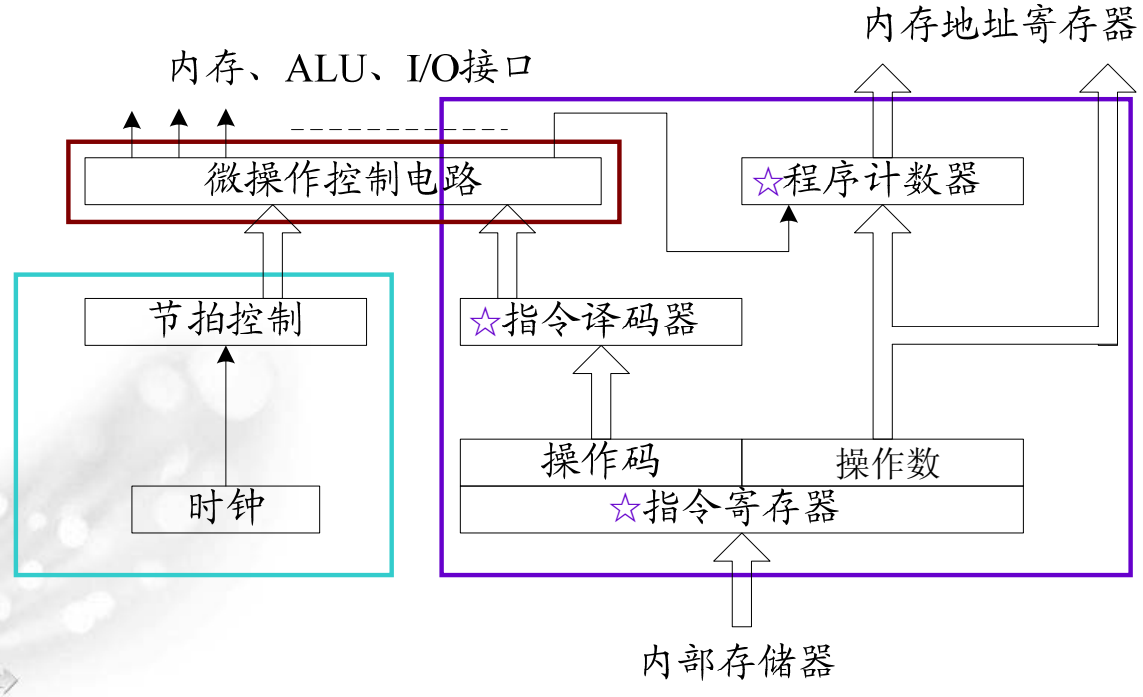
✧乘除运算：用移位操作实现

2.1 微处理器的基本结构

1. 算术逻辑单元ALU
2. 控制器
3. 总线与总线缓冲器
4. 寄存器阵列

2.1.2 控制器

- 任务：取指令、指令译码和指令执行。
  - 操作命令的发布、程序和原始数据的输入、CPU内部信息处理、结果输出、外设与主机之间的信息交换等。
- 至少包括指令部件、时序部件和微操作控制电路。

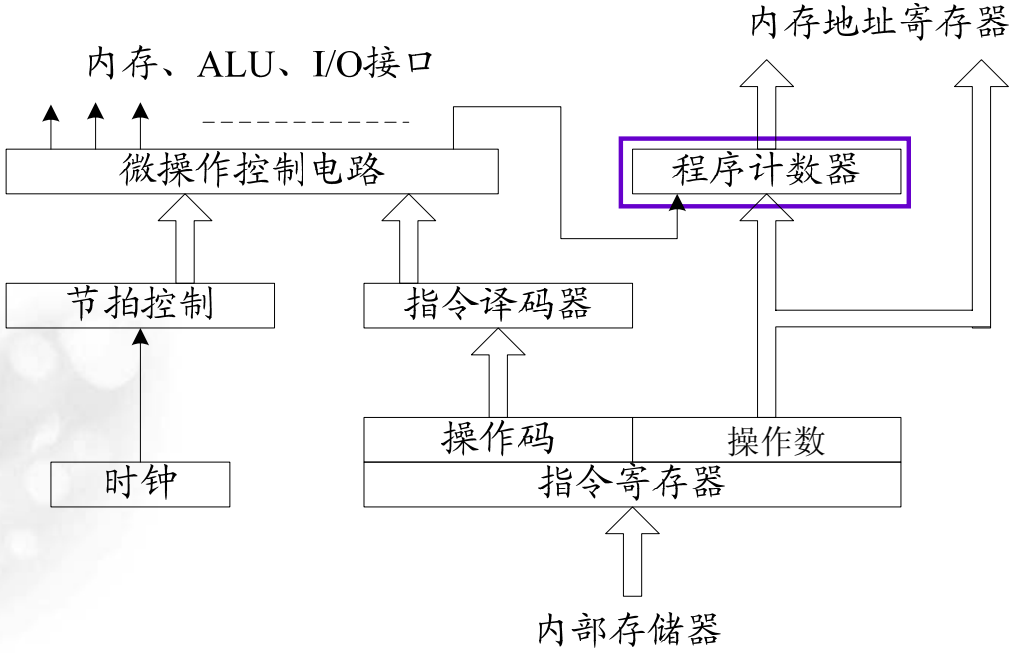


2.1.2 控制器

指令部件

程序计数器PC

- 工作原理：计算机运行程序时，**PC**指针首先指向要执行程序的首地址，开始取指令操作，取出指令后，**PC**指针的内容就自动增加，指向下一条指令的首地址。



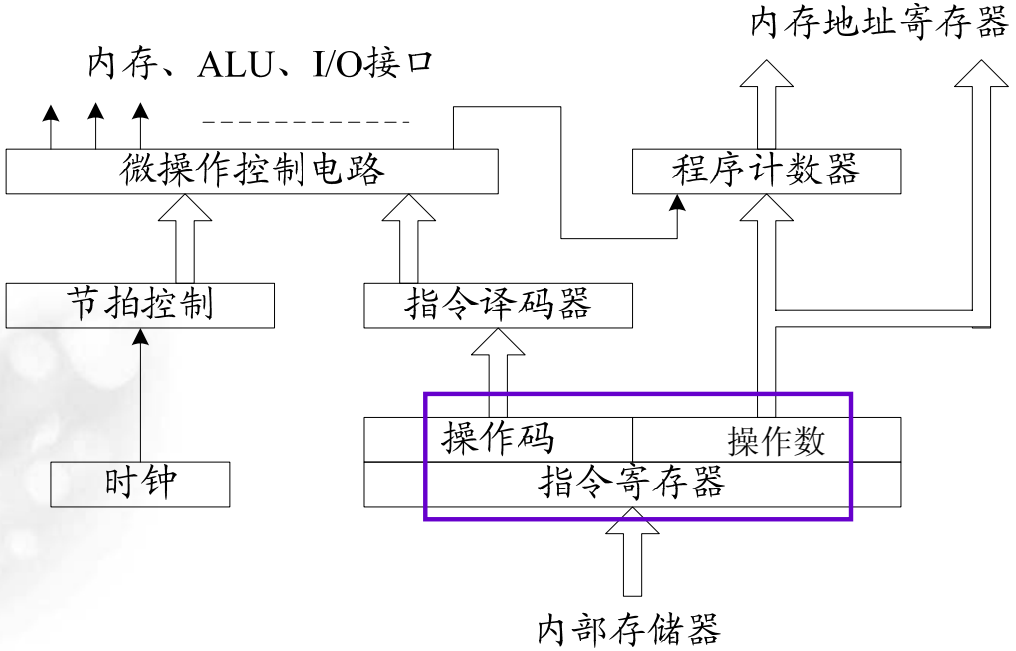
9

2.1.2 控制器

指令部件（续）

指令寄存器IR

- 存放当前执行指令内容，包括操作码和操作数两部分。
  - 操作码通过指令译码器译码，得到相关的操作命令；
  - 操作数则表示指令执行时需要的数据或存放数据的地址。

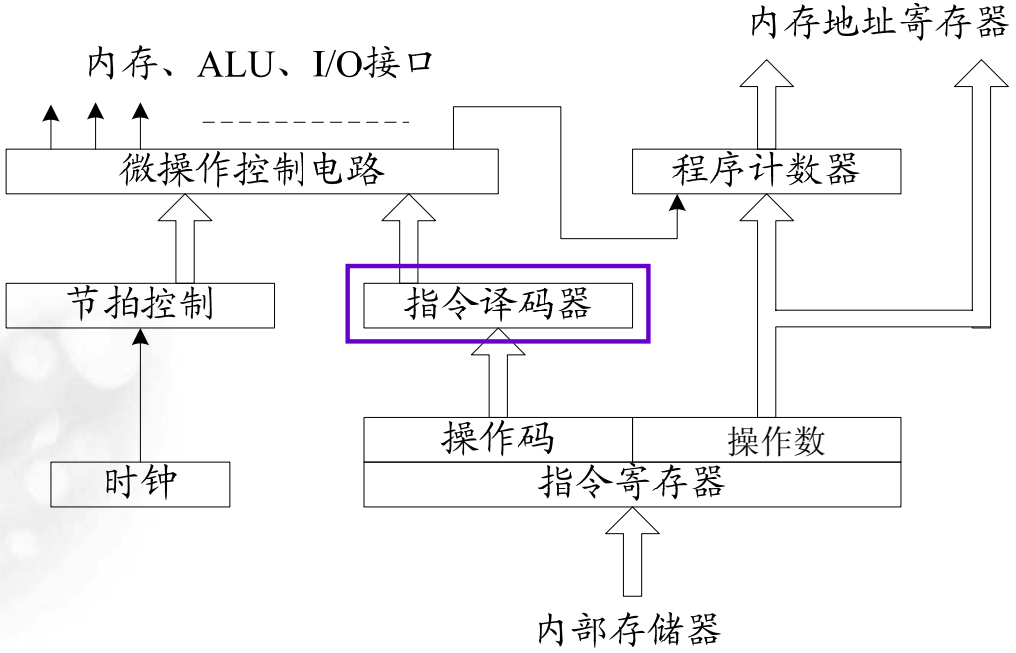


2.1.2 控制器

指令部件（续）

指令译码器ID

- 操作码经指令译码器译码后，产生相应操作和控制电位。
- **8位操作码经译码后，可对应2^8=256种特定操作，16位操作码经译码后，对应2^{16}=65536种特定操作。**



2.1.2 控制器

时序部件——产生各部件所需的定时信号

- 时钟系统
  - 包括脉冲源和时钟启停逻辑两大部分。
    - 脉冲源用来产生具有一定频率和宽度的脉冲信号(主脉冲)，通常使用外接石英晶体振荡器产生。
      - ☆概念：两个相邻脉冲的时间间隔称为时钟周期(或T状态)，它是CPU操作的最小时间单位。
    - 时钟启停逻辑用作控制启停主脉冲信号的开关，按指令和控制台的要求，可准确地开启或关闭时钟脉冲序列。
- 脉冲分配器
  - 产生计算机各部分所需的能按一定顺序逐个出现的节拍电位，以控制和协调计算机各部分有节奏地动作。

12

2.1.2 控制器

微操作控制——产生各部件所需的控制信号

☆什么是微操作？

✧微型计算机执行一条指令，通常是把一条指令分成若干个基本动作（微操作），并在节拍和脉冲信号指挥下完成一个微操作。

➢ 实现方式1：组合逻辑控制

✧以逻辑代数为工具，对实现每条指令所需的操作进行分析和归纳，得到一些简单的逻辑表达式，并根据这些表达式组成逻辑电路，以实现对计算机的控制。

✧优点：控制速度快，高速计算机较多采用这种方式。

2.1.2 控制器

微操作控制——产生各部件所需的控制信号（续）

- 实现方式2：微程序控制
  - 利用程序存储控制的原理，采用微程序完成机器指令系统中每一条指令功能的控制方法。
    - 实现过程：将每条指令都看成由若干条微指令组成的微程序，把这些微程序存放在只读存储器ROM中。计算机工作时，逐条取出微指令，执行一段微程序，便实现了这条机器指令的控制功能。
  - 优点：便于设计，便于检查、修改和更换指令。
  - 缺点：速度较慢。
- 实现方式3：可编程序逻辑阵列控制
  - 通过程序设计来执行特定逻辑功能的组合逻辑结构。
  - 优点：具有上述两种控制的优点。

14

2.1 微处理器的基本结构

1. 算术逻辑单元ALU
2. 控制器
3. 总线与总线缓冲器
4. 寄存器阵列

15

2.1.3 总线与总线缓冲器

片内总线

- 在微处理器内部各单元之间传送信息的总线。
  - 单总线结构
    - 微处理器内部各部件(如累加器、ALU、寄存器组与FR)都挂在内部总线上。
    - 在同一时刻数据总线上只能传送一个操作数，输入数据和输出数据不能同时出现在数据总线上。
  - 双总线结构
    - 内部总线分为输入总线与输出总线。
    - 通过内部输入总线将数据从各个寄存器送到ALU，而ALU的输出则通过内部输出总线送至各寄存器。
  - 多总线结构

16

2.1.3 总线与总线缓冲器

片外总线

- 在微处理器与各外部部件之间传送信息的总线。
  - 数据总线DB
  - 地址总线AB
  - 控制总线CB
- 与单总线结构相似，总线上的数据只能分时进行传送。当系统运行时各个部件均挂在总线上。
  - 在同一时刻只能允许一个部件向总线发送数据。
  - 可以允许多个部件同时接收数据。

2.1.3 总线与总线缓冲器

总线缓冲器

- 缓冲器的作用
  - 是数据（信息）与总线连接的一个缓冲部件。当其处于低阻状态时，表示部件与总线连通；处于高阻状态时，表示部件与总线断开。
- 缓冲器的类型
  - 当部件只需向总线发送数据，可采用单向三态缓冲器；
  - 若部件既需向总线发送数据，又需从总线上接收数据，则可采用双向三态缓冲器。
- CPU内部数据总线与外部数据总线间都有数据总线缓冲器，用来隔离CPU内部数据总线和外部数据总线。

18

2.1 微处理器的基本结构

1. 算术逻辑单元ALU
2. 控制器
3. 总线与总线缓冲器
4. 寄存器阵列

19

2.1.4 寄存器阵列

- 存放待处理数据的寄存器
  - 累加器(Accumulator)
    - CPU中至少有一个累加器，为运算和处理提供数据。
    - 除了一个主累加器外，其它几个通用寄存器也可作累加器使用。
  - 通用寄存器组
    - 是在数据处理过程中可被指定为不同用途的一组寄存器。通常用来临时存放数据和地址。
    - 优点：CPU可以直接处理这些数据和地址，大大减少了访问存储器的次数，提高了运算速度。

20

2.1.4 寄存器阵列

- 存放地址码的寄存器
  - 程序计数器PC
    - 用来存放现行指令的地址。
    - 取出现行指令后，**PC**就自动加**1(**除转移指令外**)**，以指向下一指令的地址。
    - 不能供编程使用。
  - 堆栈指针SP
    - 用来指示内存**(RAM)**中堆栈栈顶的地址。
    - 每次压入或弹出一个数据时，它能自动修改**SP**的值，以便保证**SP**始终指向栈顶。

21

2.1.4 寄存器阵列

存放控制信息的寄存器

- 指令寄存器IR
  - 用于存放指令的代码（机器码），一直保存到指令译码器译码完成为止。
- 标志寄存器FR
  - 用来保存**ALU**运行结果的标志。
  - **FR**的内容可以用专门的指令来测试，进而判断程序是否转移。大多数算术、逻辑操作都会影响一到几个标志位。
  - 通常状态标志有以下几个：
    - 进位**(CF)**标志
    - 辅助进位**(AF)**标志
    - 奇偶校验**(PF)**标志
    - 零**(ZF)**标志
    - 符号**(SF)**标志
    - 溢出**(OF)**标志

22

2.1.4 寄存器阵列

总线缓冲寄存器

- 数据总线缓冲寄存器DBUF
  - 是**CPU**内部数据总线和外部数据总线之间起缓冲作用的三态双向缓冲器，可以有效避免总线冲突。
- 地址总线缓冲寄存器ABUF
  - 是具有三态控制的单向缓冲器，用于在**CPU**内部地址总线与外部地址总线之间起缓冲作用。

23

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

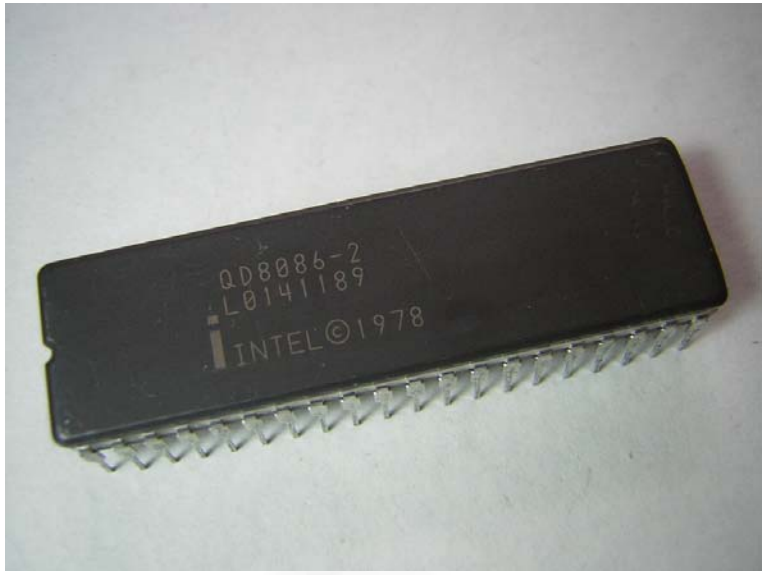
2.7 80486微处理器*

2.8 Pentium处理器*

24

2.2 Intel 8086微处理器

8086/8088微处理器



- **8086/8088微处理器是Intel公司推出的第三代CPU芯片，它们的内部结构基本相同，都采用16位结构进行操作及存储器寻址，但外部性能有所差异，两种处理器都封装在相同的40脚双列直插芯片中。**

25

2.2 Intel 8086微处理器

1. 8086的寄存器结构
2. 8086 CPU的编程结构
3. 8086 CPU的引脚及其功能

2.2.1 8086的寄存器结构

8086CPU内有4组寄存器

2.2.1 8086的寄存器结构

通用寄存器

- 8086/8088有4个16位的数据寄存器(AX、BX、CX、DX)，可以存放16位的操作数，也可分为8个8位的寄存器(AL、AH；BL、BH；CL、CH；DL、DH)来使用。
- AX称为累加器，BX称为基址寄存器，CX称为计数寄存器，DX称为数据寄存器。

寄存器	用　　途
AX	字乘法，字除法，字I/O
AL	字节乘，字节除，字节I/O，十进制算术运算
AH	字节乘，字节除
BX	查表（表格的开始地址）
CX	串操作，循环操作的计数器
CL	移位或循环次数
DX	字节乘，字节除，间接I/O

28

2.2.1 8086的寄存器结构

- 指针和变址寄存器
  - 16位的指针寄存器
    - ✧SP是堆栈指针寄存器，由它和堆栈段寄存器SS一起来确定堆栈在内存中的位置。
    - ✧BP是基址指针寄存器，通常用于存放基地址。
  - 16位的变址寄存器
    - ✧SI是源变址寄存器
    - ✧DI是目的变址寄存器

    } 用于访问数据存储器中的数据。

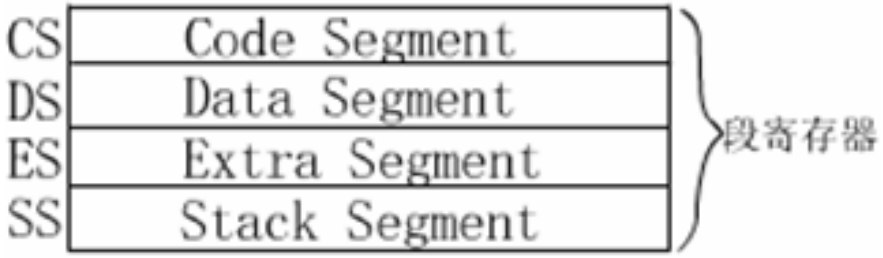
SP	Stack Pointer	} 指针及变址寄存器
BP	Base Pointer	
SI	Source Index	
DI	Destination Index	

29

2.2.1 8086的寄存器结构

段寄存器

- 共有4个16位段寄存器：
  - 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



- 段寄存器的内容与有效的地址偏移量一起，可确定内存的物理地址。通常CS划定并控制程序区，DS和ES控制数据区，SS控制堆栈区。

30

2.2.1 8086的寄存器结构

指令指针寄存器IP

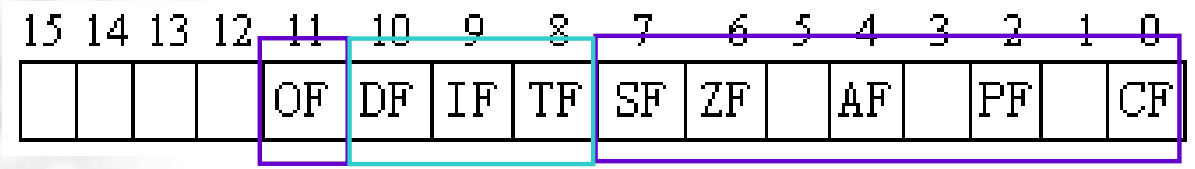
IP Instruction Pointer 指令指针

- 用来控制CPU的指令执行顺序，它和代码段寄存器CS一起可以确定当前所要取的指令的内存地址。
  - 顺序执行程序时，CPU每取一个指令字节，IP自动加1，指向下一个要读取的字节。
  - 当IP单独改变时，会发生段内的程序转移；当CS和IP同时改变时，会产生段间的程序转移。

2.2.1 8086的寄存器结构

标志寄存器FR (Flag Register)

- 16位寄存器，它的内容即程序状态字(PSW, Program Status Word)，用来存放CPU在工作过程中的状态。
- 程序状态字共有9个标志位，包括6个状态标志位和3个控制标志位。
  - 状态标志：反映**ALU**执行算术、逻辑运算等操作后的一些特征。它们可作为后续操作（转移等）的判断标志。
  - 控制标志：反映了人对微机系统工作方式的可控制性。它们可通过指令人为设置，用以对**CPU**的某种特定功能起控制作用**(**如中断屏蔽等**)**。



32

2.2.1 8086的寄存器结构

程序状态字——状态标志位

- CF (Carry Flag)进位标志位。【PSW_0】
  - 当指令执行的结果（8位或16位）在最高位产生了进位（如加法）或借位（如减法），则CF=1，反之CF=0。
- PF (Parity Flag)奇偶标志位。【PSW_2】
  - 当运算结果的低8位中“1”的个数为偶数，则PF=1，为奇数，PF=0。
- AF (Auxiliary Carry Flag)辅助进位标志位。【PSW_4】
  - 当一个8位（或16位）数的D3位向D4位有进位（如加法运算）或借位（如减法运算），则AF=1，反之AF=0。
  - 举例：

```
    1101   1000
  +  1010   1110
  ------------------
   1←1000←0110        AF＝1，CF＝1，PF=0
```


33

2.2.1 8086的寄存器结构

程序状态字——状态标志位（续）

- ZF (Zero Flag)零标志位。【PSW_6】
  - 当运算结果为全零，则ZF=1，反之ZF=0。
- SF (Sign Flag)符号标志位。【PSW_7】
  - 当运算结果为负数，即结果的最高位为1，则SF=1，反之SF=0。（SF与运算结果的最高位相同）
- OF (Overflow Flag)溢出标志位。【PSW_{11}】
  - 当运算结果超出了8位或16位带符号数所能表达的范围，则OF=1，反之OF=0。

OF=1
- 8位字节运算：结果大于+127或小于-128
- 16位字运算：结果大于+32767或小于-32768

34

2.2.1 8086的寄存器结构

程序状态字——状态标志位（续）

- 举例：字节运算67H+69H后，CF和OF的值?
  - 分析：
    67H　　01100111
    + 69H　　01101001

    11010000　=(0D0H=+208>+127)
  - 结果：CF=0，OF=1
- 举例：字节运算0A6H+0FAH后，CF和OF的值?
  - 分析：
    0A6H(-90)　10100110
    + 0FAH(-6)　11111010
    ---
    1 10100000　=(0A0H=-96>-128)
  - 结果：CF=1，OF=0
- 结论：进位与溢出的含义不同。进位主要反映不带符号数（纯数值）运算结果的状态，而溢出主要反映带符号数运算结果的状态。

35

2.2.1 8086的寄存器结构

程序状态字——控制标志位

- TF (Trap Flag)陷阱/单步/跟踪标志位。【PSW_8】
  - 若该位置1，将使8086/8088进入单步工作方式，通常用于程序的调试；置0，则正常执行程序。
- IF (Interrupt-enable Flag)中断允许标志位。【PSW_9】
  - 若该位置1，则微处理器可以响应外部可屏蔽中断（开中断）；置0，则不响应外部可屏蔽中断（关中断）。
- DF (Direction Flag)方向标志位。【PSW_{10}】
  - 若该位置1，则串操作指令的地址修改为自动减量方向（从高地址向低地址递减）；置0，则为自动增量方向（从低地址向高地址递增）。

36

2.2 Intel 8086微处理器

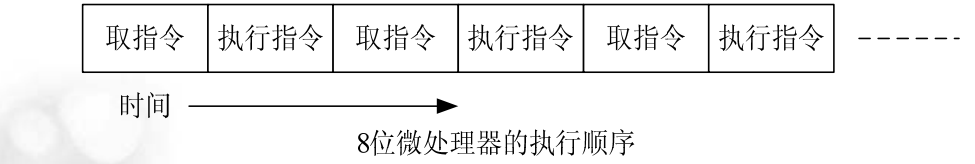
1. 8086的寄存器结构
2. 8086 CPU的编程结构
3. 8086 CPU的引脚及其功能

37

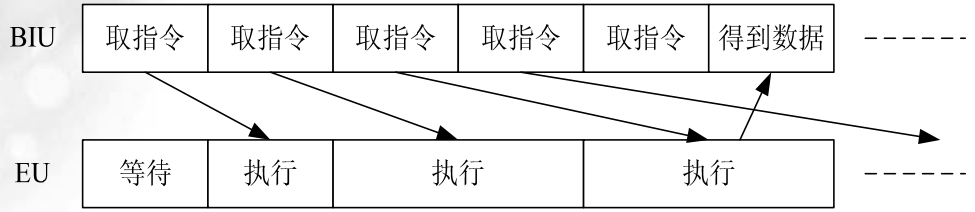
2.2.2 8086CPU的编程结构

编程结构

- 指从编程者角度看到的结构，亦可称为功能结构。
- 从功能上看，8086CPU可分为两部分：
  - 总线接口部件BIU ——负责指令和操作数读及结果写。
  - 执行部件EU——负责指令的执行。
- 两个部件独立地进行操作（即并行工作），使得取指令、分析指令和执行指令可以并行操作，提高了CPU的工作效率，加快了指令的执行速度。



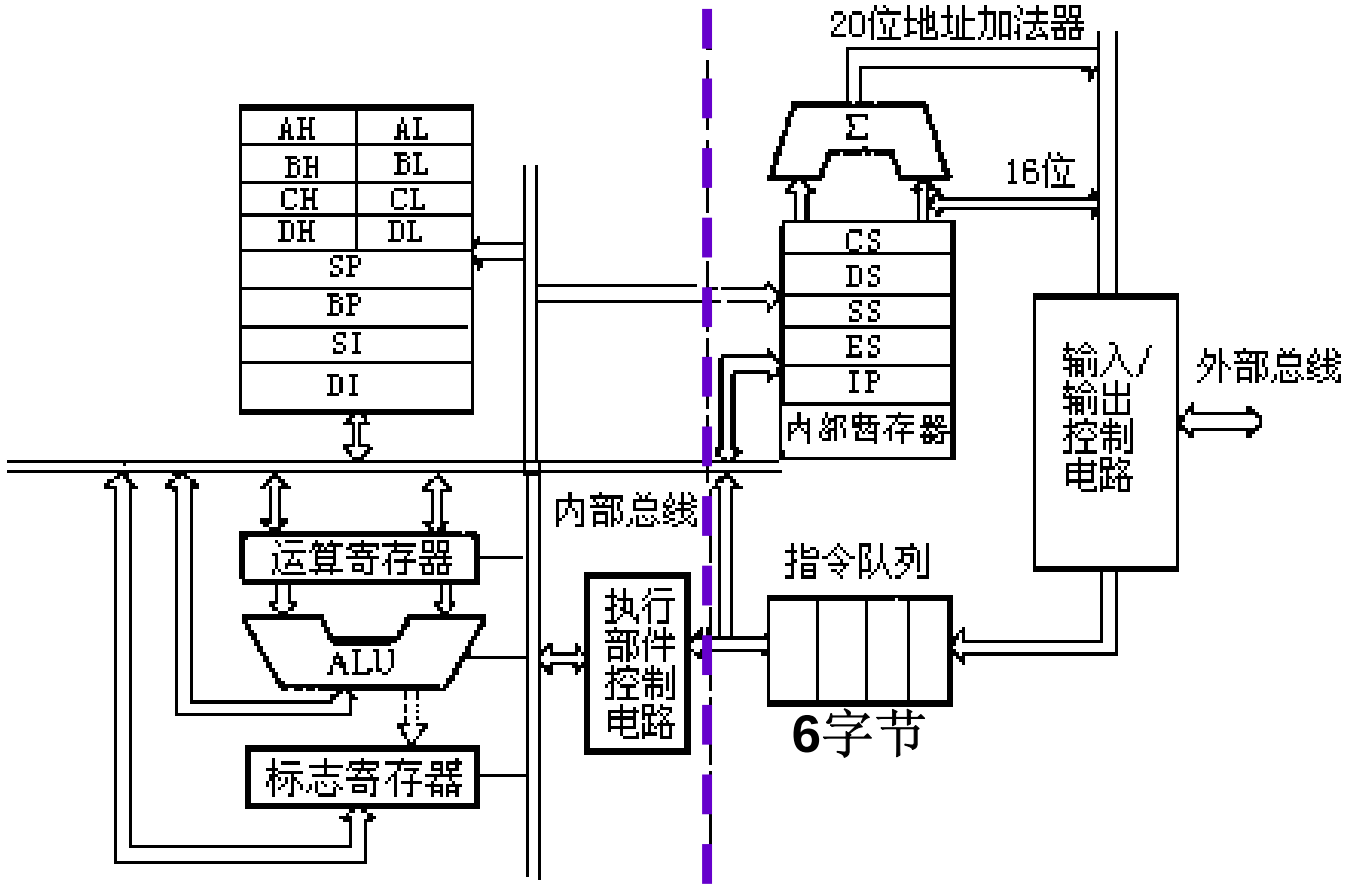
8位微处理器的执行顺序



8086的执行顺序

38

2.2.2 8086CPU的编程结构



执行单元EU
Execution Unit

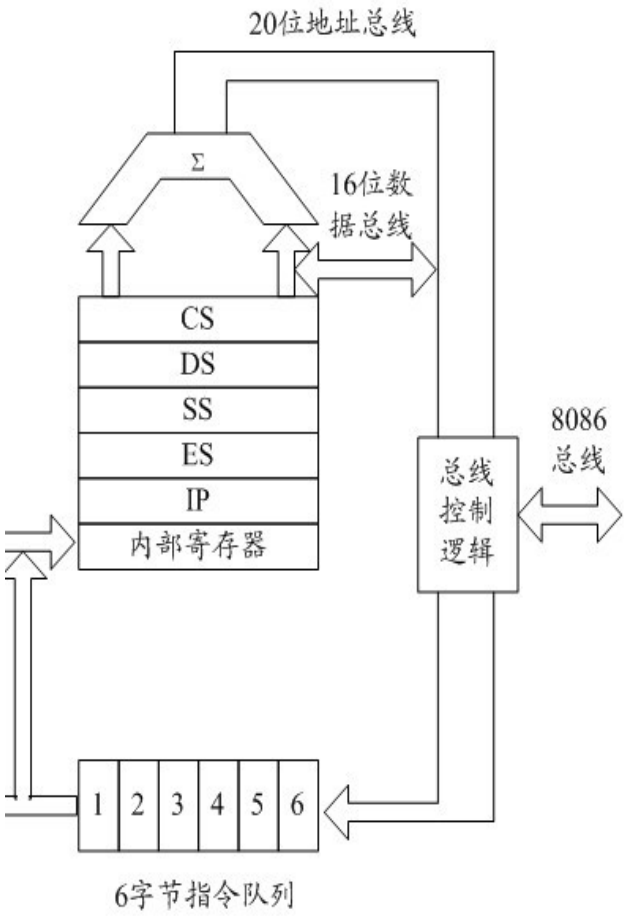
总线接口部件BIU
Bus Interface Unit

39

2.2.2 8086CPU的编程结构

总线接口部件BIU

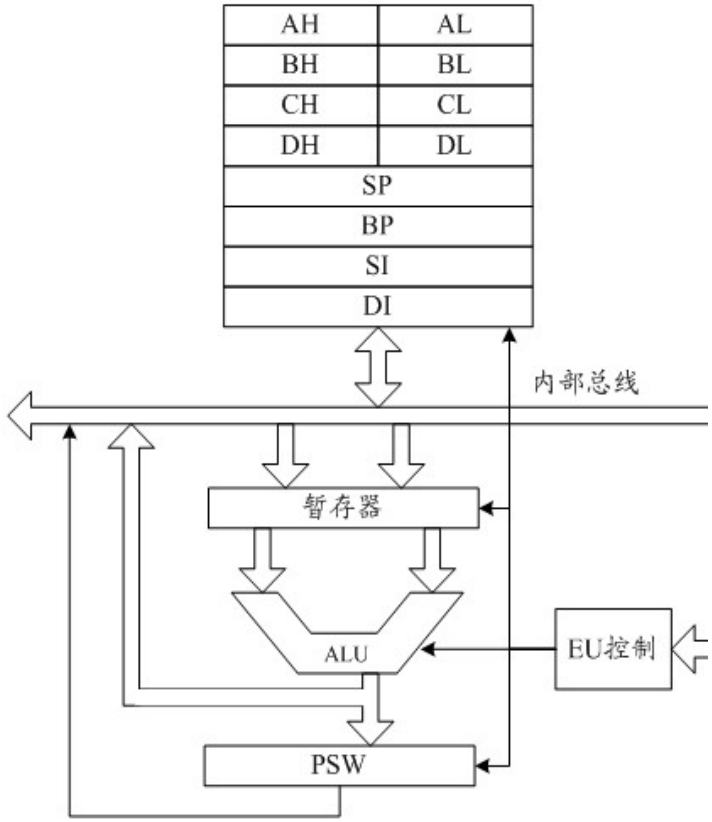
- 组成
  - 16位段寄存器(DS、CS、ES、SS)；
  - 16位指令指针寄存器IP；
  - 20位地址加法器(用来产生20位地址)；
  - 6字节(8088为4字节)指令队列缓冲器；
  - 总线控制逻辑。
- 功能
  - 负责从内存中取指令；
  - 送入指令队列；
  - 实现CPU与存储器和I/O接口之间的数据传送。



2.2.2 8086CPU的编程结构

执行部件EU

- 组成
  - ALU(算术逻辑单元);
  - 数据寄存器(AX、BX、CX、DX);
  - 指针和变址寄存器(BP、SP、SI、DI);
  - 标志寄存器(PSW);
  - EU控制系统。
- 功能
  - 负责分析指令和执行指令。



41

2.2.2 8086CPU的编程结构

BIU和EU的动作协调原则（流水线技术）

- 当指令队列中有两个或两个以上空字节，且EU未向BIU申请读写存储器或I/O口时，BIU就会自动地顺序预取后续指令到指令队列（先入先出队列）。
  - 队列输入指针：在一个字节的代码装入队列输入端后，自动调整队列输入指针，以指向下一个字节单元。
  - 队列输出指针：EU从队列输出端取指令，在其取走一个字节的代码后，自动调整队列输出端指针，使其指向队列中下一个可用字节单元。
- 当指令队列已满，且EU又没有总线访问请求时，BIU便进入空闲状态。

42

2.2.2 8086CPU的编程结构

BIU和EU的动作协调原则（流水线技术）（续）

- 当EU准备执行一条指令时，首先从BIU部件的指令队列前端取出指令代码，然后用几个时钟周期对指令进行译码并执行指令。
  - 在此过程中，若需访问存储器或I/O口，则EU会请求BIU进入总线周期，完成读写存储器或I/O口的操作。
  - 若此时BIU正好处于空闲状态，则立即响应EU的请求；若BIU正将某个指令字节取到指令队列中，则BIU将首先完成这个取指令的总线周期，然后再去响应EU的请求。
- 在执行转移指令、调用指令和返回指令时，由于待执行指令的顺序发生了变化，则指令队列中已经装入的字节被自动消除，BIU会接着往指令队列装入转向的另一程序段中的指令代码。

动画演示

43

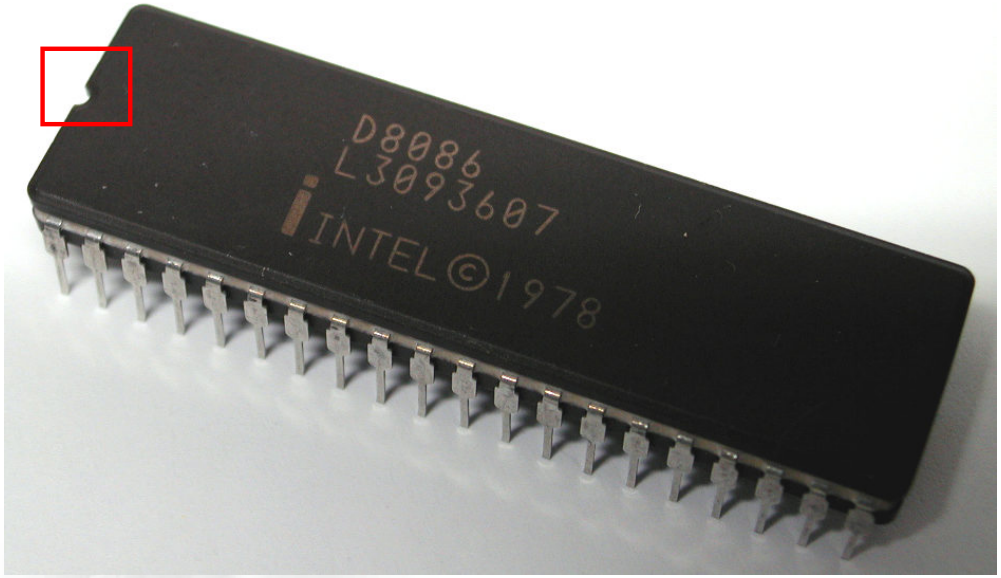
2.2 Intel 8086微处理器

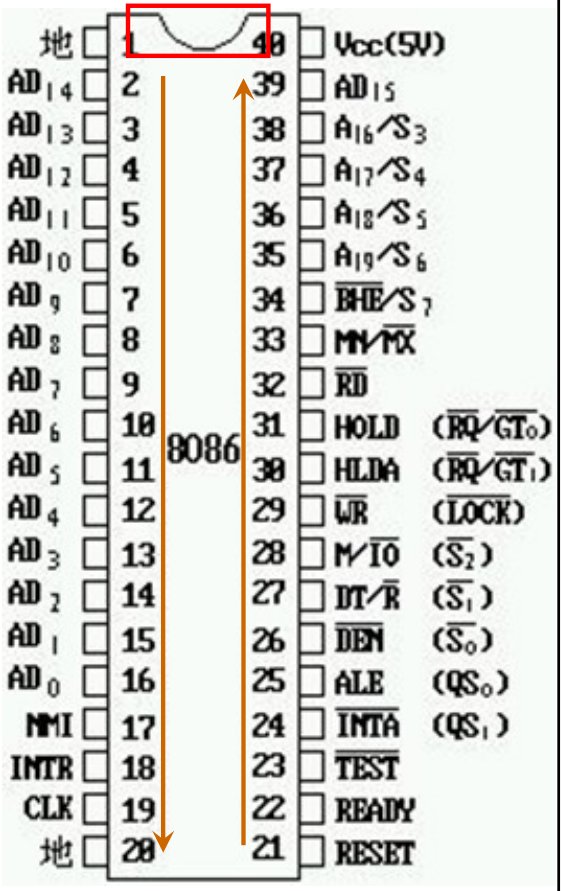
1. 8086的寄存器结构
2. 8086 CPU的编程结构
3. 8086 CPU的引脚及其功能

44

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚结构





☆40根引脚，双列直插式封装。

45

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能

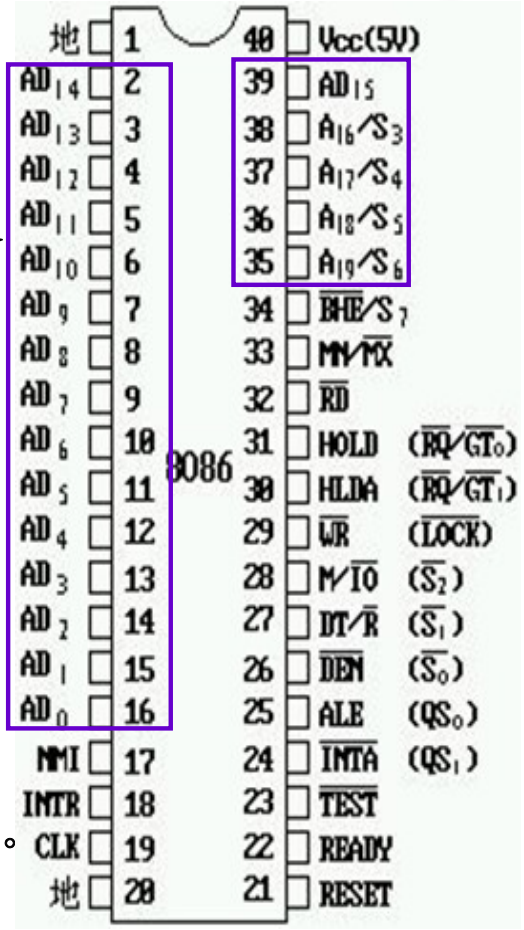
- VCC(40)、GND(1、20)：供电电源的正负引脚。
  - ✧采用单一的**5V**电源供电，具有两个接地引脚。
- CLK(19)：时钟信号输入引脚。
  - ✧为**CPU**和总线控制器提供定时时钟信号，该信号为非对称方波信号，占空比约为**33%**，即**1/3**周期为高电平，**2/3**周期为低电平。
  - ✧从该引脚输入的时钟信号频率为**5MHz**。（**8086**的主频）

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- AD_{15}～AD_0(2～16, 39)：地址/数据复用信号输入/输出引脚。
  - ✧分时输出低16位地址信号及输入/输出16位数据信号。
- A_{19}/S_6～A_{16}/S_3(35～38)：地址/状态复用信号输出引脚。
  - ✧分时输出地址的高4位及状态信息。
    - S_6为0表示8086当前与总线连通；
    - S_5为1表示8086可响应可屏蔽中断；
    - S_4、S_3共有四个组合状态，用以指明当前使用的段寄存器。

00—ES，01—SS，10—CS，11—DS。



☆什么是分时与复用？

47

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- RESET(21)：复位信号输入引脚，高电平有效。
  - ✧至少维持4个时钟周期，才有复位效果。
  - ✧复位后，CPU结束当前操作，并对FR、IP、DS、SS、ES及指令队列执行清零操作，将CS设置为0FFFFH。
- READY(22)：“准备好”状态信号输入引脚，高电平有效。
  - ✧接收来自内存单元或I/O口发来的“准备好”状态信号，表明存储器或I/O口已经准备好进行读写操作。
  - ✧该信号是协调CPU与存储器或I/O口之间进行信息传送的联络信号。

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

配合

- $\overline{RD}$(32)：读控制信号输出引脚，低电平有效。
  - ✧用以指明要执行一个对存储器或I/O口的读操作。

配合

- M/$\overline{IO}$(28)：存储器或I/O口选择信号输出引脚。
  - ✧是CPU区分进行存储器访问还是I/O口访问的输出控制信号。
- $\overline{WR}$(29)：写控制信号输出引脚，低电平有效。
  - ✧用以指明要执行一个对存储器或I/O口的写操作。

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- $\overline{DEN}$(26)：数据允许信号输出引脚，低电平有效。
  - 用以指明CPU当前准备发送或接收数据。
  - 为数据总线收发器提供控制信号。
- DT/$\overline{R}$(27)：数据收发控制信号输出引脚。
  - CPU通过该引脚发出控制数据传送方向的控制信号。
  - 当该信号为高电平时，表示数据由CPU经数据总线收发器输出，否则数据传送方向相反。

2.2.3 8086CPU的引脚及其功能

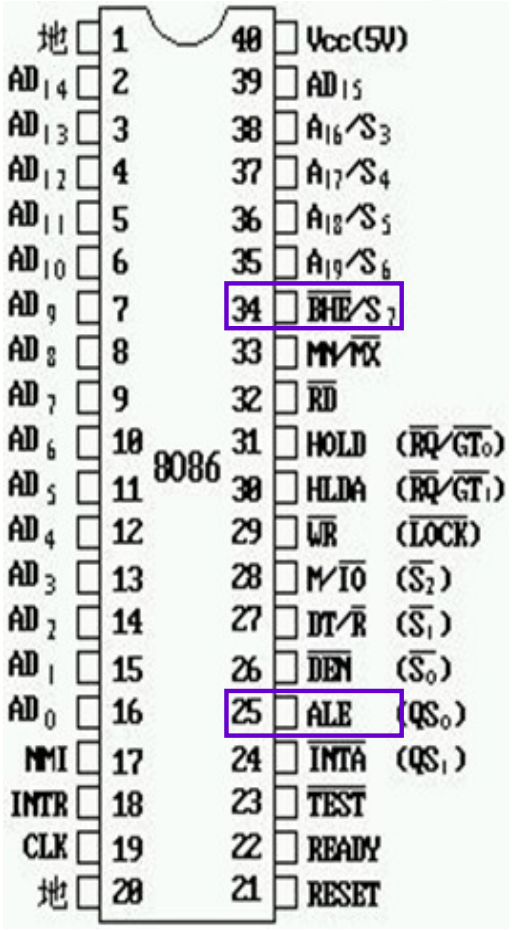
8086CPU的引脚功能（续）

- NMI(17)、INTR(18)：中断请求信号输入引脚，高电平有效。
  - 前者引入非屏蔽中断请求，后者引入可屏蔽中断请求。
- $\overline{\text{INTA}}$(24)：中断响应信号输出引脚，低电平有效。
  - 该引脚是CPU响应中断请求后，向中断源发出的认可信号。用以通知中断源，以便其提供中断类型码。
  - 该信号为两个连续的负脉冲。

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- ALE(25)：地址锁存允许信号输出引脚，高电平有效。
  - CPU通过该引脚发出地址锁存允许信号，使地址锁存器对当前地址/数据复用总线上的地址信息进行锁存。
- $\overline{BHE}/S_7$(34)：高8位数据允许/状态复用信号输出引脚。
  - 该引脚分时输出有效信号，表示高8位数据线D_{15}~D_8上的数据有效和S_7状态信号。
  - S_7未定义任何实际意义。故该引脚实质上是低电平有效。

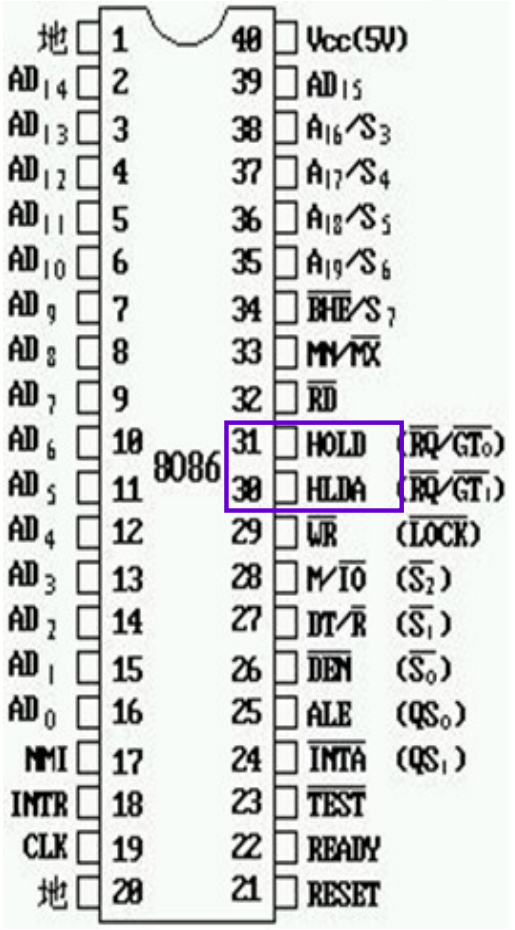


52

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- HOLD(31)：总线保持请求信号输入引脚，高电平有效。
  - ✧引入系统中其它总线部件向CPU发来的总线请求信号。
- HLDA(30)：总线保持响应信号输出引脚，高电平有效。
  - ✧表示CPU认可其它总线部件提出的总线占用请求，并准备让出总线控制权。



2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

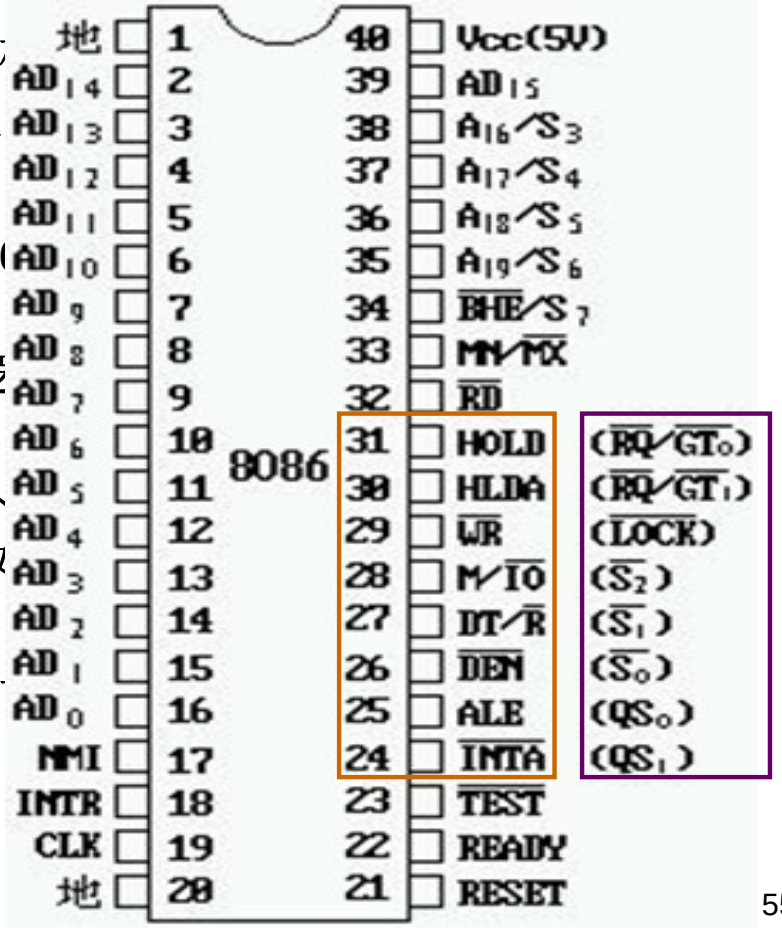
- $\overline{TEST}$(23)：测试信号输入引脚，低电平有效。
  - CPU执行WAIT指令后，处于等待状态，仅当该引脚有低电平输入时，系统脱离等待状态，继续执行被暂停的指令。
- MN/$\overline{MX}$(33)：最小/最大模式设置信号输入引脚。
  - 该输入引脚的电平决定了CPU工作在最小模式还是最大模式。
    - 接+5V时，CPU工作于最小模式。
    - 接地时，CPU工作于最大模式。

？？

2.2.3 8086CPU的引脚及其功能

8086CPU的工作模式

- 为了尽可能适应各种各
  CPU芯片时，允许其工
  - ✧最小模式
    - 系统中只有一个808
      CPU产生。
    - 系统中的总线控制逻
  - ✧最大模式
    - 系统中至少包含两个
      8086CPU，其它微处
      器工作。
- 最大模式下和最小模式



55

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- QS_1、QS_0(24、25)：指令队列状态信号输出引脚。
  - 两者的组合给出了前一个T状态中指令队列的状态，以便于外部跟踪CPU内部指令队列的动作。

QS_1	QS_0	性　能
0	0	无操作
0	1	取走了指令队列的第一个字节代码
1	0	队列为空
1	1	除第一个字节外，还取走了后续字节中的代码

最大模式

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- $\overline{S_0}$、$\overline{S_1}$、$\overline{S_2}$(26、27、28)：总线周期状态信号输出引脚，低电平有效。
  - 三个信号的组合，表示当前总线周期中数据传输过程的类型。供总线控制器使用。

S_0	S_1	S_2	性　能
0	0	0	中断响应
0	0	1	读I/O端口
0	1	0	写I/O端口
0	1	1	暂停
1	0	0	取指令
1	0	1	读存储器
1	1	0	写存储器
1	1	1	无作用

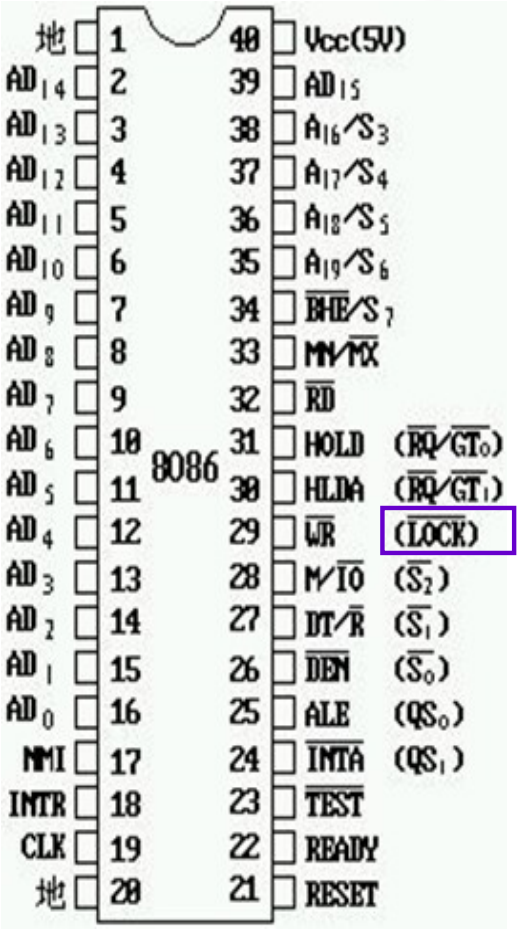
最大模式

2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

- $\overline{\text{LOCK}}$(29)：总线封锁信号输出引脚，低电平有效。
  - ✧该引脚输出低电平时，系统中其它总线部件就不能占用系统总线。
  - ✧信号由指令前缀**LOCK**产生，指令执行完毕后，便撤消信号。
  - ✧在**8086**的**2**个中断响应脉冲之间，该信号也自动变为有效的低电平，以防止其它总线部件在中断响应过程中占有总线，从而使一个完整的中断响应过程被破坏。

最大模式

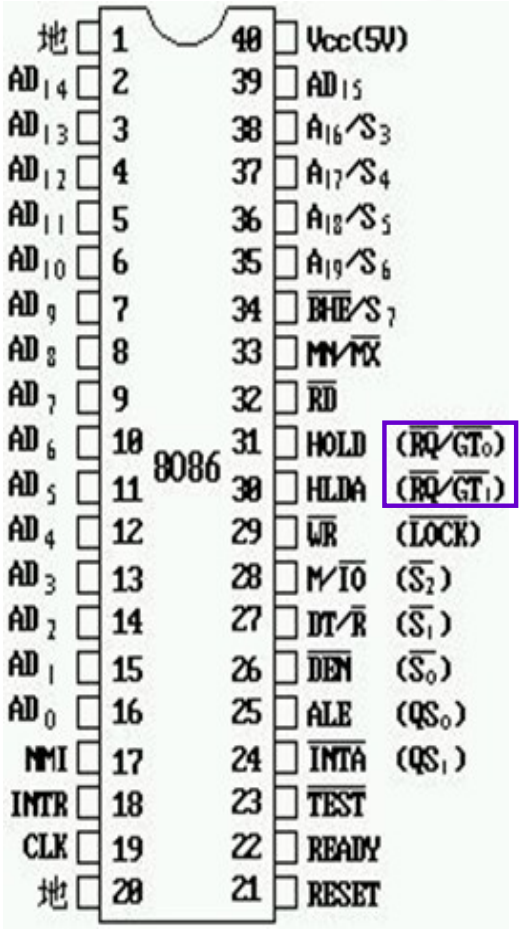


2.2.3 8086CPU的引脚及其功能

8086CPU的引脚功能（续）

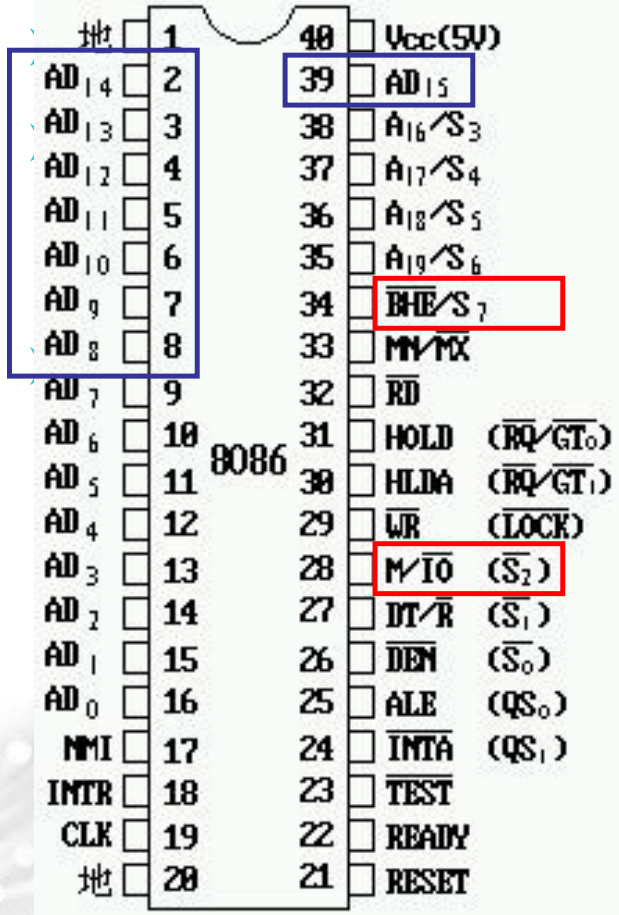
- $\overline{RQ}/\overline{GT_0}$、$\overline{RQ}/\overline{GT_1}$(31、30)：总线请求信号输入/总线允许信号输出引脚。低电平有效。
  - 这两个引脚可供CPU以外的两个处理器使用，用于发出使用总线的请求信号和接收CPU对总线请求信号的应答。
  - 两个引脚都是双向的，请求与应答信号在同一引脚上分时传输，方向相反。
  - 31脚比30脚的优先级高。

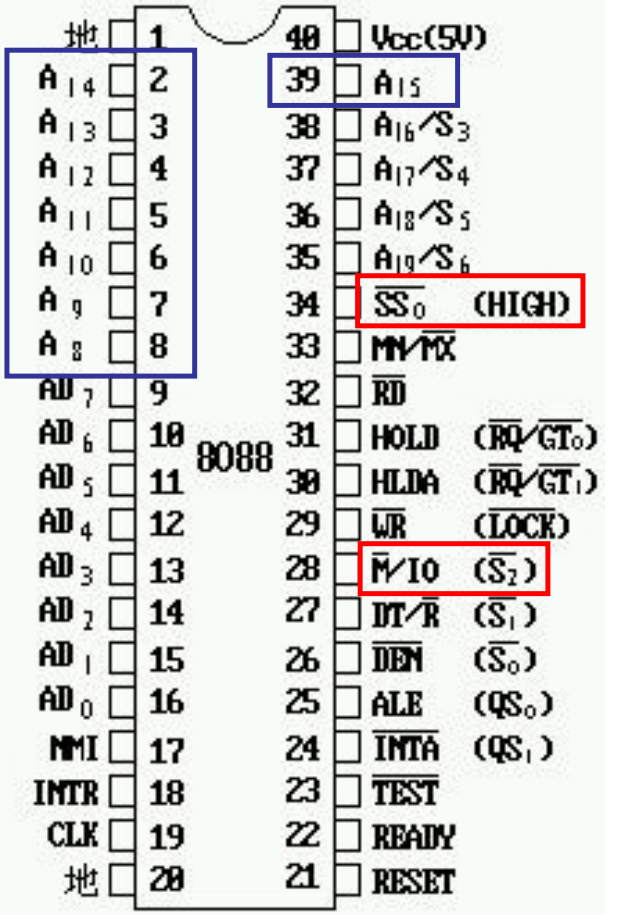
最大模式



2.2.3 8086CPU的引脚及其功能

- 8088与8086CPU的区别





60

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

61

2.3 8086中的程序状态字和堆栈

1. 程序状态字
2. 堆栈

62

2.3.1 程序状态字

- 参见标志寄存器FR

PSW：Program Status Word

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

63

2.3 8086中的程序状态字和堆栈

1. 程序状态字
2. 堆栈

2.3.2 堆栈

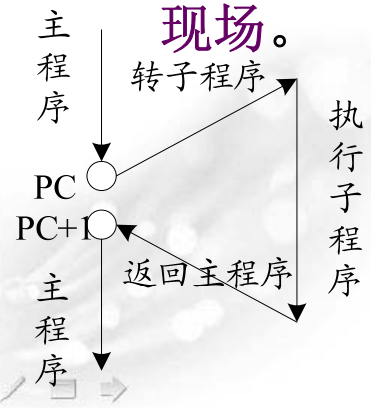
- 定义
  - 所谓堆栈就是一个按照后进先出(LIFO：Last In First Out)的原则存取数据的存储器空间。
- 分类
  - 硬件堆栈：早期的微处理器常常利用一组内部寄存器作为堆栈，称为硬件堆栈。
    - ✧特点：工作速度较快，但受寄存器数目限制，深度有限。
  - 软件堆栈：把内存的一个区域作为堆栈，称为软件堆栈。目前的微型机一般都采用这种方式。
    - ✧特点：容量可以很大。但需要堆栈指针SP来指示堆栈在内存的什么位置。
    - ✧8086/8088规定：SP始终指向堆栈的顶部，即始终指向最后压入堆栈的数据所在的单元。

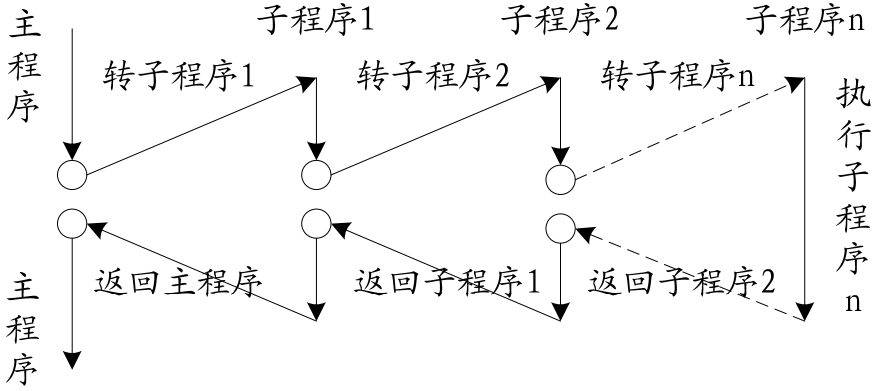
65

2.3.2 堆栈

用途

- 调用子程序或转向中断服务程序时，保存关键信息，并在程序返回时恢复这些信息。
- 关键信息包括：
  - 返回后要执行指令的地址（即断点）。
  - 主程序和子程序或中断服务程序中都用到的寄存器、标志位等。
- 关键信息入栈称为保护现场，关键信息出栈称为恢复现场。





2.3.2 堆栈

表示

- 堆栈段寄存器**SS**：标识现行堆栈段的段基地址。
- 堆栈指针**SP**：标识现行堆栈段内的偏移地址。
  - 随着数据压入堆栈，SP的值减小。——向下生成（8086）
  - SP的初值设定：MOV SP，data。
  - 堆栈的最大容量即为SP的初值。

实现

- 存放数据（入栈）：PUSH操作
  - **step1：sp-2 →sp**（注意方向）
  - **step2：**存放数据
- 读取数据（出栈）：POP操作
  - **step1：**读取数据
  - **step2：sp+2 →sp**（注意方向）

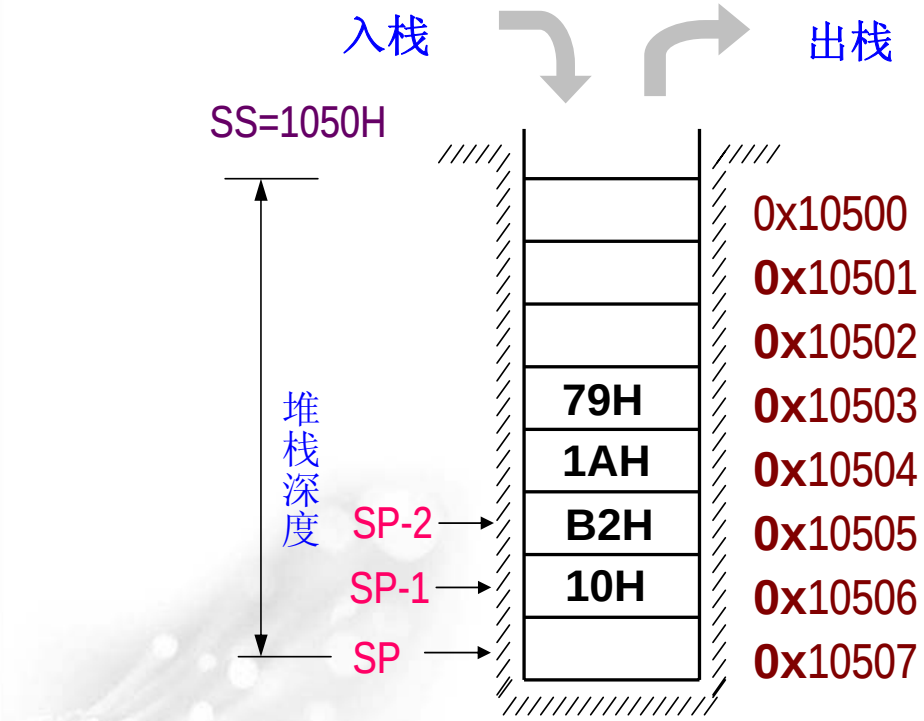
☆关键：**SP**始终指向有效数据

67

2.3.2 堆栈

举例

- 堆栈存取方式：先入后出FILO（=后入先出LIFO）。



MOV SP, #7

以字word为单位

☆思考：初值是奇数还是偶数好？

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

69

2.4 8086系统的组成

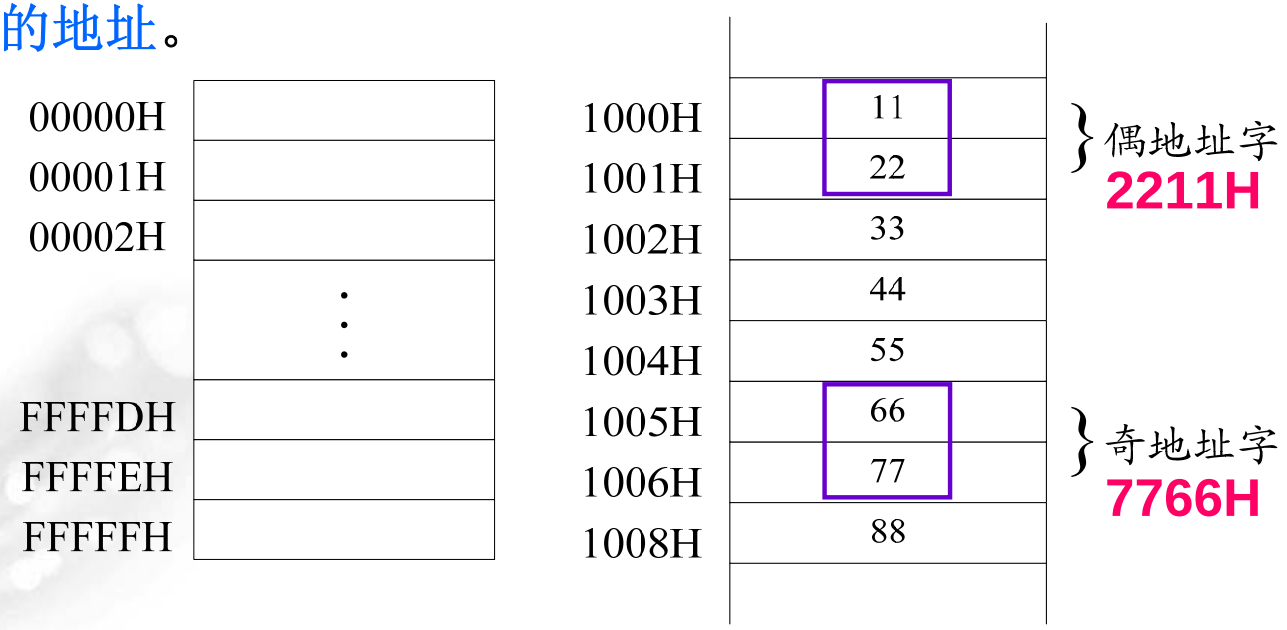
1. 存储器组织与存储器分段
2. 输入/输出结构
3. 总线接口结构
4. 8086的两种组态

70

2.4.1 存储器组织与存储器分段

存储器组织

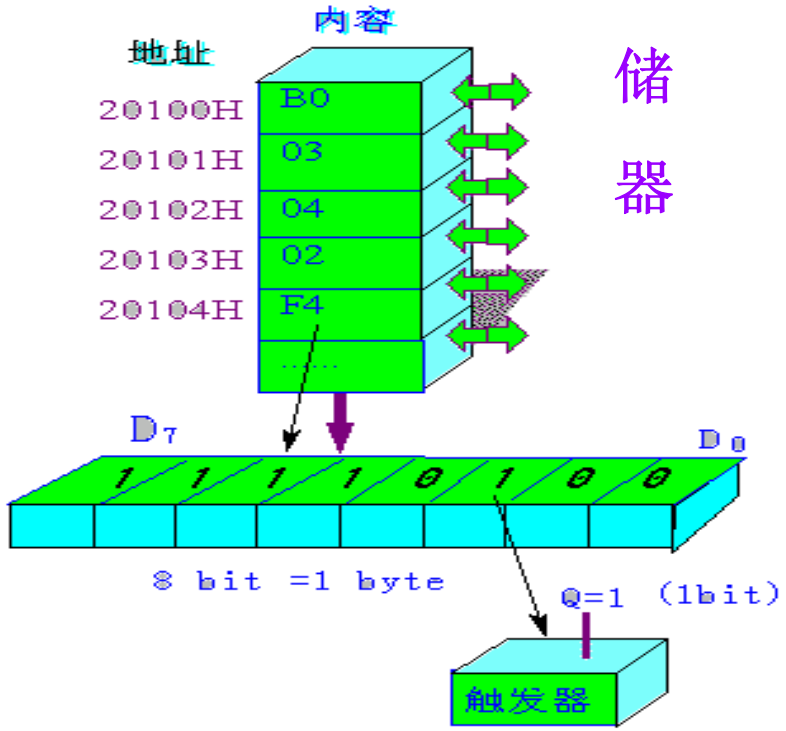
- 8086的AB有20根地址线，直接寻址能力为2^{20}＝1MB，地址00000H到FFFFFH。
- 任何两个相邻的字节单元可以存放一个16位的字(Word)，且两个地址中数值较小的那个地址作为该字的地址。



71

2.4.1 存储器组织与存储器分段

存储器组织（续）



20100H字节内容

= B0H

20100H字内容

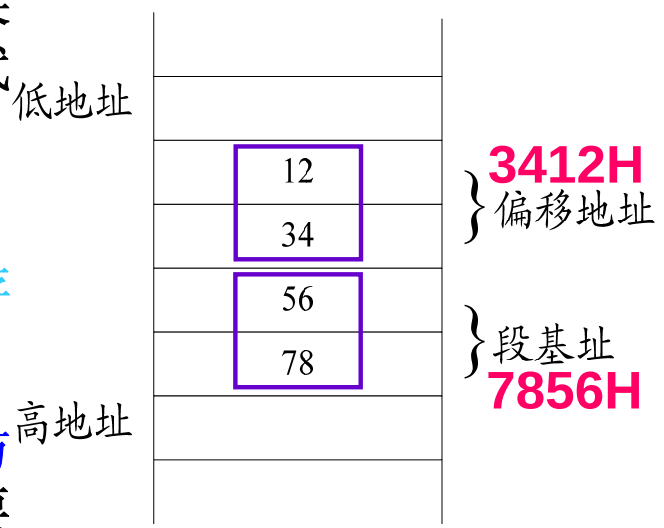
= 03B0H

72

2.4.1 存储器组织与存储器分段

存储器组织（续）

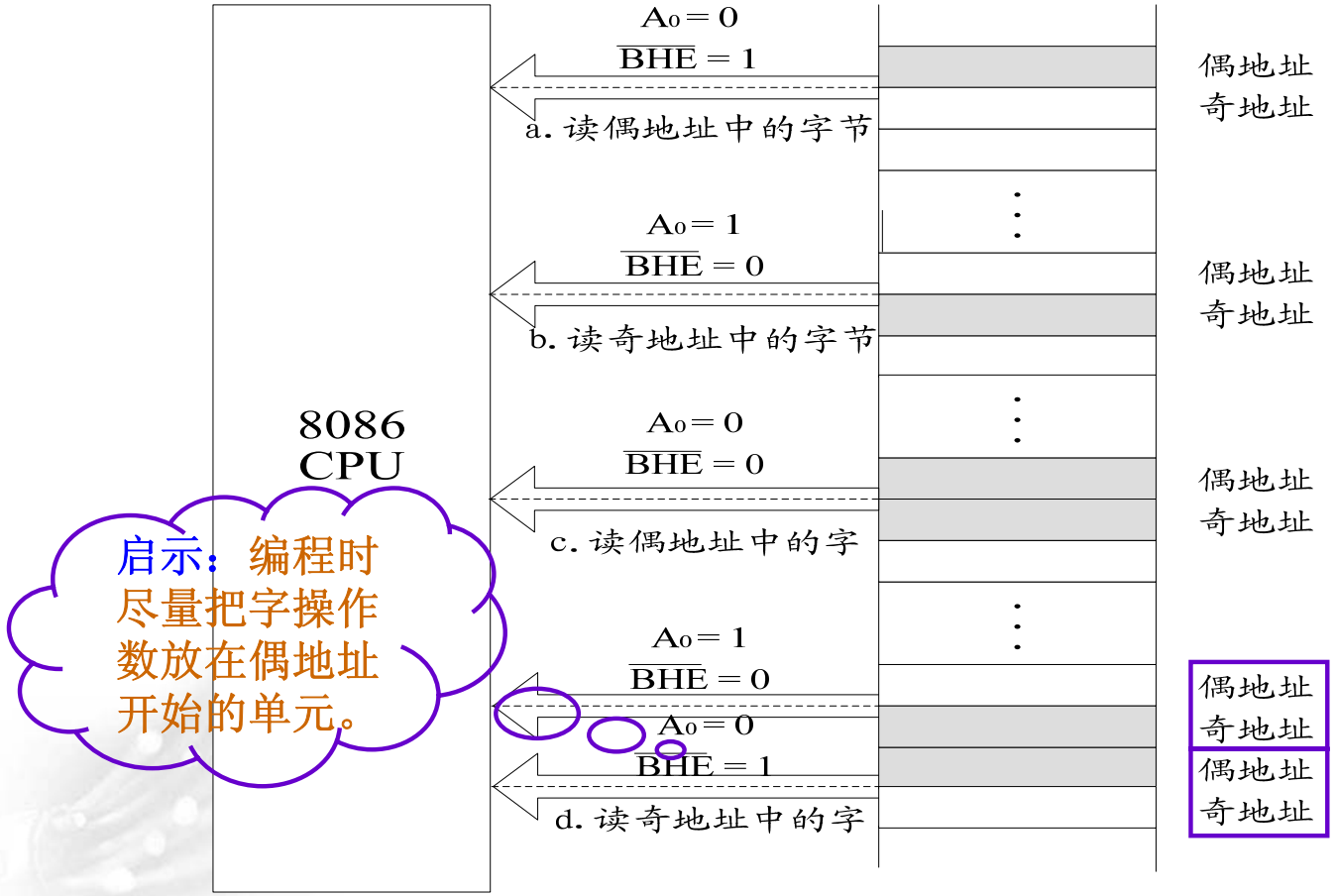
- 对于指针，一般是按双字进行存放。指针的低位字代表在某段中的偏移量，而高位字则代表该段的段基址。
- 8086不论是读还是写存储器，每次总是16位，并且第一个存储单元的地址一定是偶地址。
- 读写奇地址的字，必须两次访问存储器，分别地取它所需要的那半个字，并进行合并后形成所需的字。（由CPU自动完成）



73

2.4.1 存储器组织与存储器分段

存储器组织（续）



74

2.4.1 存储器组织与存储器分段

存储器分段

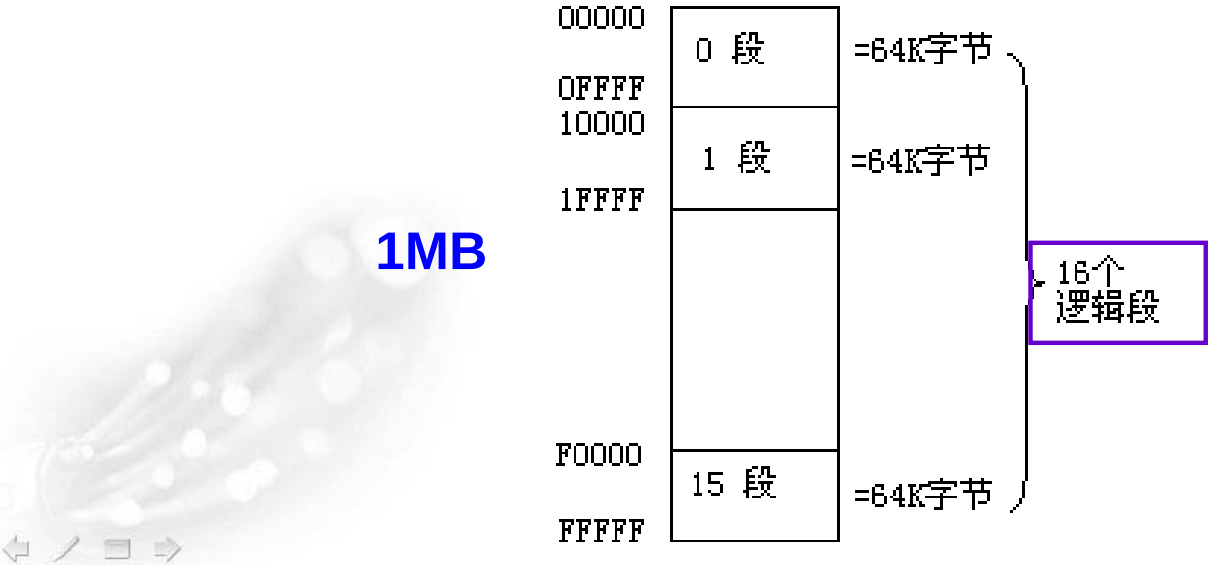
- 8086的AB有20根地址线，意味着存储器每个存储单元的地址由20位二进制数构成。
- 8086内部用来存放地址信息的寄存器只有16位。

出现矛盾 → 解决办法 → 存储器逻辑上分段

2.4.1 存储器组织与存储器分段

存储器分段（续）

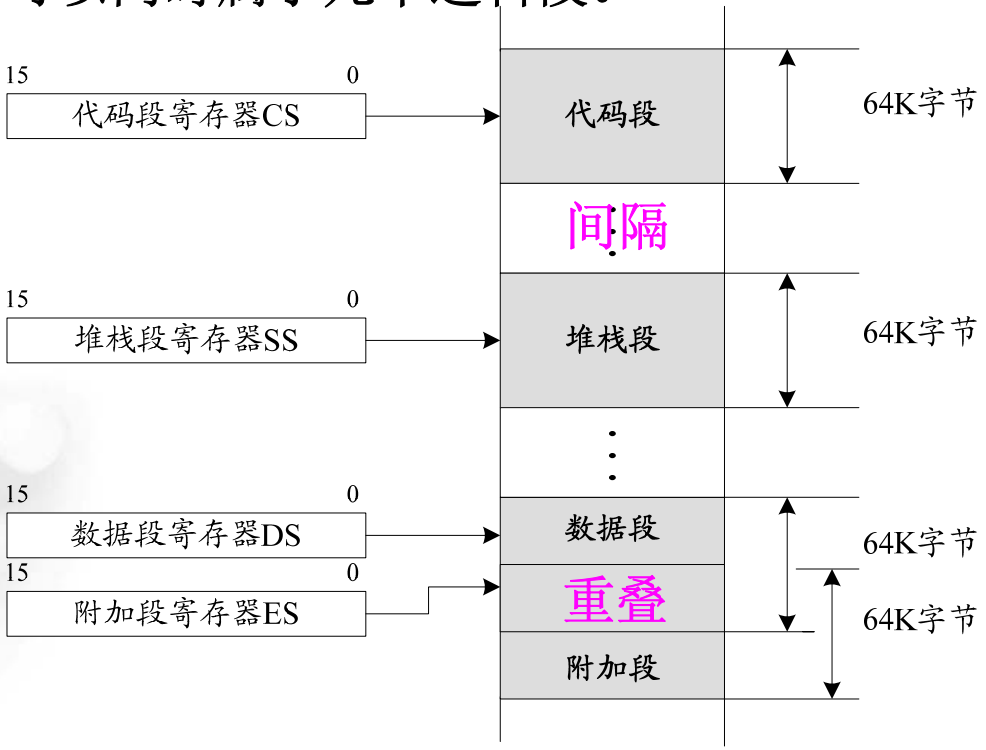
- 分析：16位二进制地址可寻址范围是64KB，而1MB的存储空间可以在逻辑上分为16个段，每段大小是64KB。
- 解决：
  - 用段地址（也称为段基址）给每个段编号，每个段内的地址单元用偏移地址编号。



2.4.1 存储器组织与存储器分段

存储器分段（续）

- 存储空间的分段不是唯一的。段之间允许互相重叠。对于一个具体的存储单元来说，它可以属于一个逻辑段，也可以同时属于几个逻辑段。



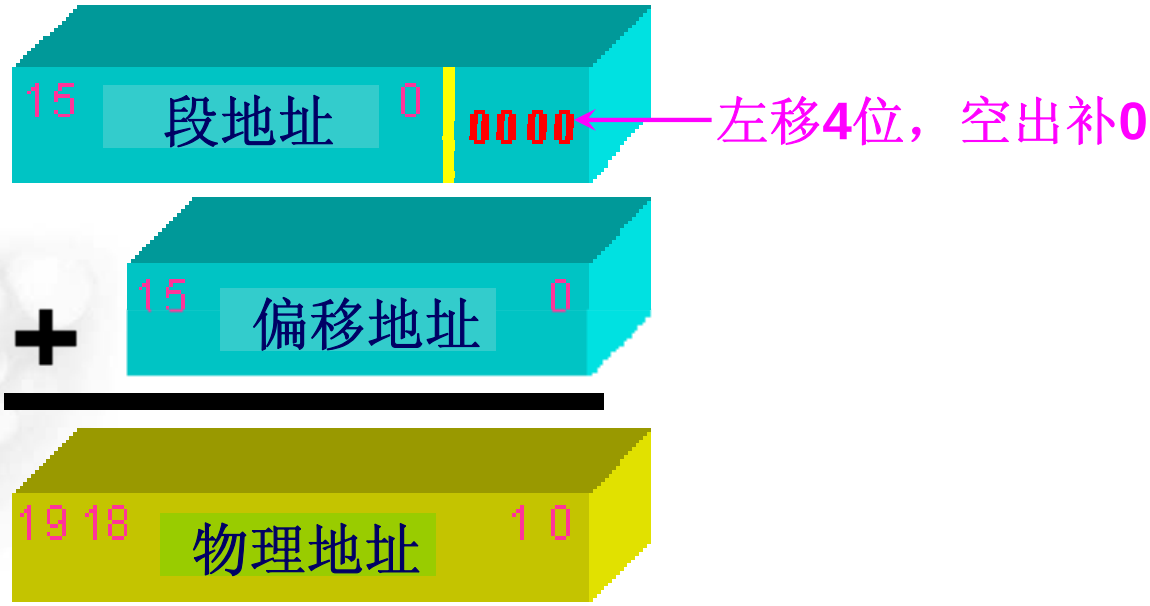
2.4.1 存储器组织与存储器分段

- 与地址有关的几个概念
  - 物理地址：20位，一个存储单元的实际地址。物理地址与存储单元是一一对应关系。(20202H)
  - 逻辑地址：由段地址和偏移地址组成，是指令中引用的形式地址。一个逻辑地址只能对应一个物理地址，而一个物理地址可以对应多个逻辑地址。(2000:0202H、2010:0102H、……)
    - 段地址：16位，存储单元所在逻辑段的编号。通常存放在对应的段寄存器中。(2000H)
    - 偏移地址：16位，存储单元在逻辑段内相对于该段第一个存储单元的距离。(0202H)

78

2.4.1 存储器组织与存储器分段

- 物理地址的计算方法
  - 20位物理地址 = 段地址×16 + 偏移地址
    - ✧取指令操作：**CS** ×16 + **IP**
    - ✧堆栈操作：**SS** ×16 + **SP**
    - ✧数据存储器操作：**DS/ES** ×16 + 偏移地址



79

2.4.1 存储器组织与存储器分段

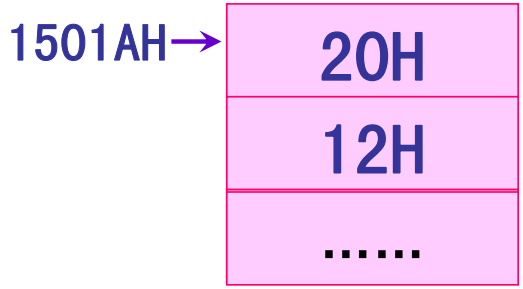
举例

在如图所示的堆栈中，若SS=1000H，SP=501AH，则对应存储单元的字节内容和字内容分别是什么？

解：物理地址=1000H×16 + 501AH

=1501AH

```
  1 0 0 0 0
+     5 0 1 A
-------------
  1 5 0 1 A
```



1501AH字节单元的内容为20H

1501AH字单元的内容为1220H

80

举例（续）

- 若CS=2000H，则其最大的寻址空间可达多少？

解：代码段寄存器CS与IP寄存器对应，

而IP的取值范围：0000H～FFFFH

因此，最小物理地址=2000H×16 + 0000H

=20000H

最大物理地址=2000H×16 + FFFFH

=2FFFFH

81

2.4.1 存储器组织与存储器分段

讨论

- 存储器中，逻辑段的最大和最小空间分别为多少？

最大段为64K

最小段为16字节

↓

段地址=FFFFH

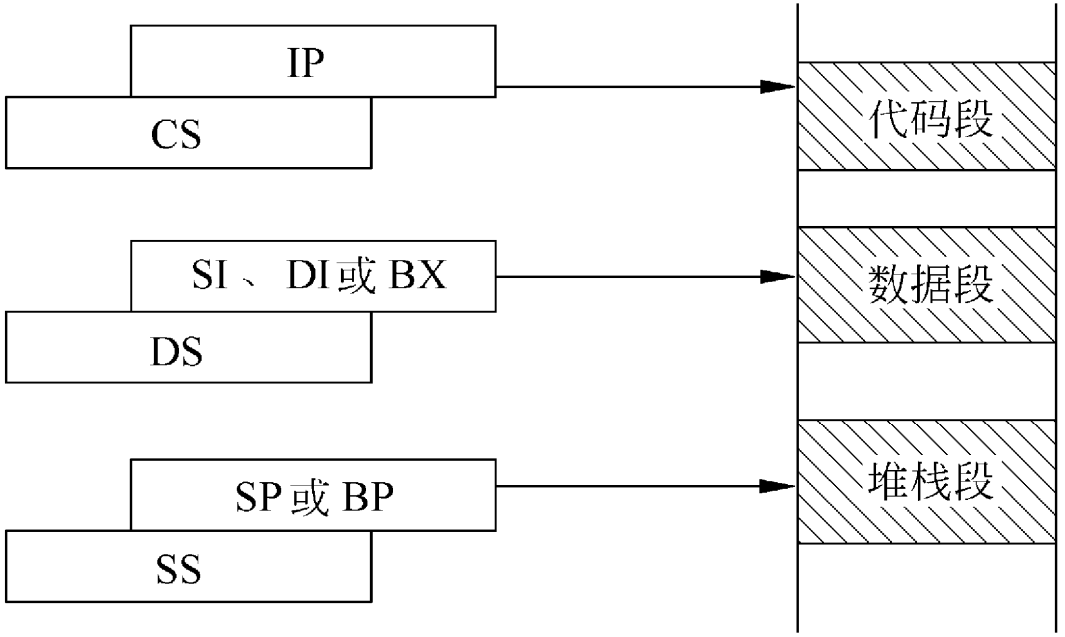
↓

偏移地址=0000H~000FH

82

2.4.1 存储器组织与存储器分段

- 段寄存器与提供偏移地址寄存器的对应关系



83

2.4.1 存储器组织与存储器分段

特殊的内存区域

- 中断矢量区：00000H—003FFH共1K字节，用以存放256种中断类型的中断矢量，每个中断矢量占用4个字节，共256×4=1024=1K。
- 显示缓冲区：B0000H—B0F9FH约4000（25×80×2）字节，是单色显示器的显示缓冲区，存放文本方式下，所显示字符的ASCII码及属性码；B8000H—BBF3FH约16K字节，是彩色显示器的显示缓冲区，存放图形方式下，屏幕显示象素的代码。
- 启动区：FFFF0H—FFFFFH共16个单元，用以存放一条无条件转移指令的代码，转移到系统的初始化部分。

84

2.4 8086系统的组成

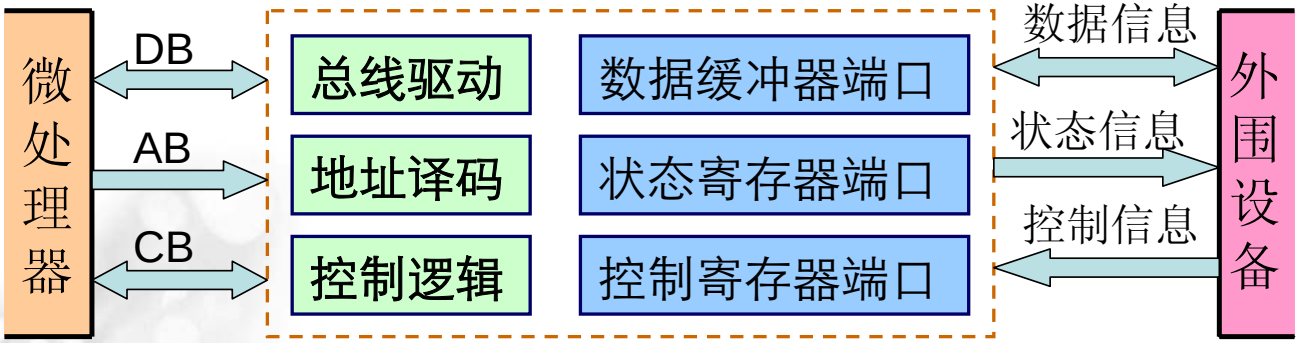
1. 存储器组织与存储器分段
2. 输入/输出结构
3. 总线接口结构
4. 8086的两种组态

85

2.4.2 输入/输出结构

什么是计算机的输入/输出？

- 计算机与相连的外围设备进行数据交换的过程称为输入/输出（Input/Output， I/O）。
- I/O过程的响应时间影响整个微机系统的工作效率。
  - 改进：将I/O过程的管理从CPU中分离出来，交由专门的功能电路完成。——I/O(接)口



I/O接口电路的典型结构

2.4.2 输入/输出结构

I/O接口的访问

- 8086微处理器具有专用 I/O访问指令，用来完成读端口信息和写端口信息的操作。
  - 8086系统可以访问8位或16位端口地址。
  - 若微处理器用直接寻址方式寻外设端口，端口地址用8位，可寻址2^8＝256个端口。
  - 若微处理器用DX间接寻址外设的端口时，端口地址用16位时，可寻址2^{16}＝64 K个端口。

87

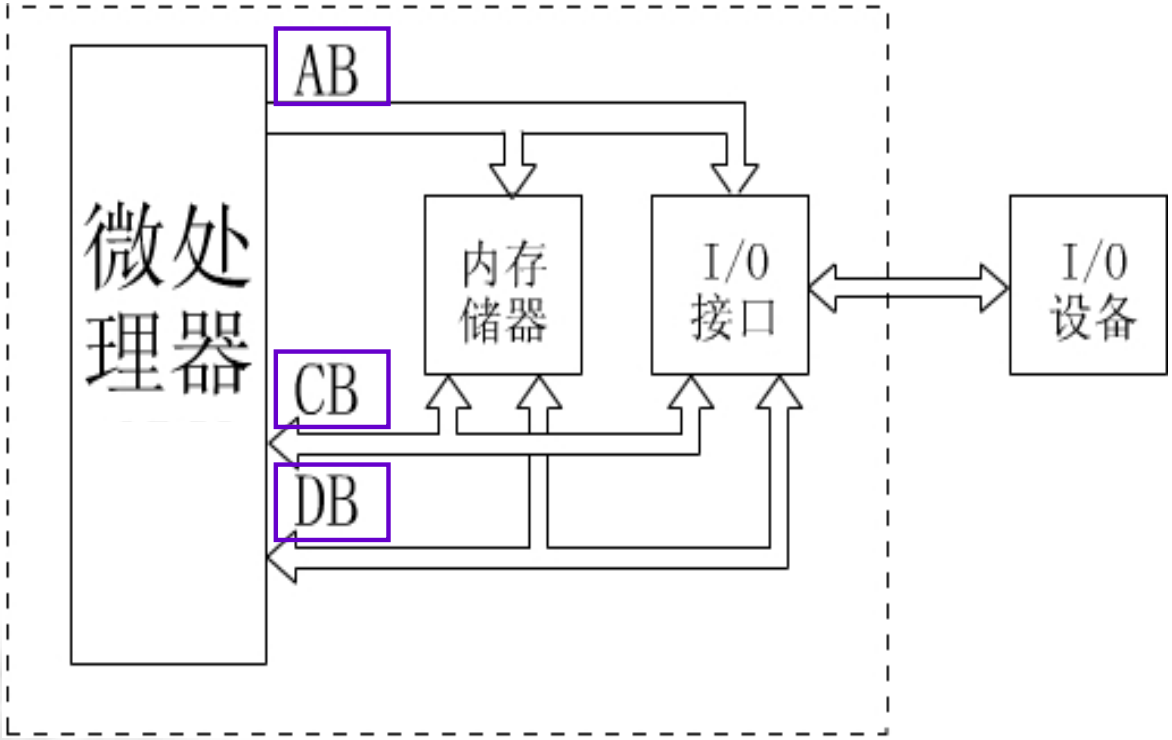
2.4 8086系统的组成

1. 存储器组织与存储器分段
2. 输入/输出结构
3. 总线接口结构
4. 8086的两种组态

88

2.4.3 总线接口结构

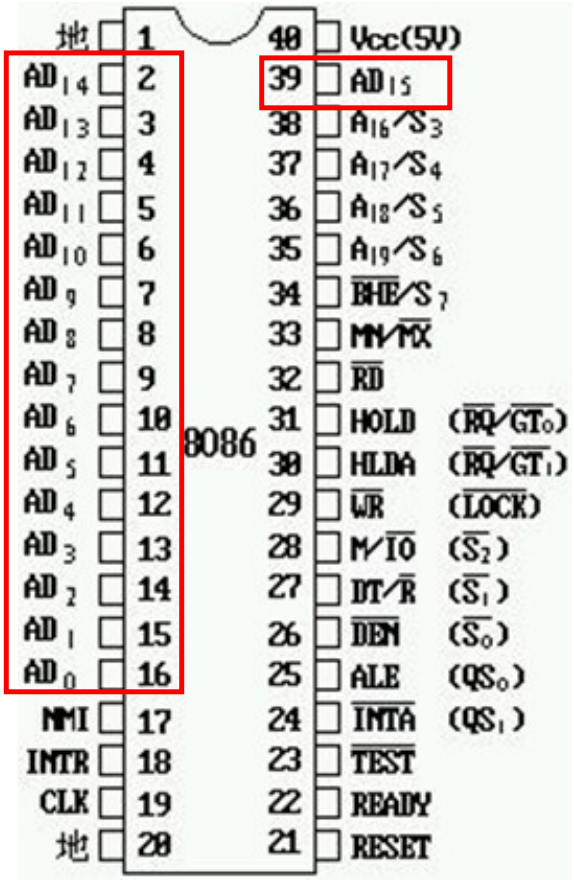
- 回顾：微机的三总线结构



89

2.4.3 总线接口结构

- 缘由
  - 受8086微处理器引脚数目的限制，数据总线DB与地址总线AB只能按分时复用设计。
  - 导致数据和地址信息冲突问题。
- 作用
  - 需要将复用的信号予以区分和保持，确保读写操作正常进行。
- 组成
  - 地址锁存器
  - 双向总线驱动器



2.4.3 总线接口结构

地址锁存器

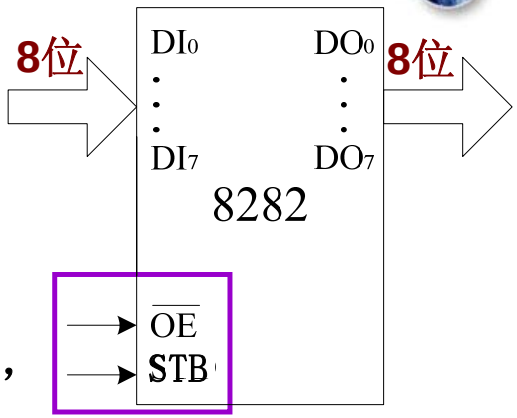
- 本质是一个暂存器，用于暂存复用总线上的地址信息。
- 工作原理
  - 微处理器与存储器或I/O口交换信息时，首先发送确定交互位置的地址信息，同时在ALE引脚上送出地址锁存有效信号。
  - ALE信号通知地址锁存器把20位地址信息及$\overline{\text{BHE}}$引脚的状态进行锁存。
  - 接着开始数据信息传送。

91

2.4.3 总线接口结构

地址锁存器（续）

- 常用地址锁存器——8282芯片。
  - ✧8位双向锁存器，20条引脚。
  - ✧选通信号STB：输入引脚。
    - 高电平时，输入、输出信号线直通，不执行锁存。
    - 由高变低，且满足持续时间要求时，对输入信号线的内容进行锁存。
    - 接CPU的ALE引脚。
  - 输出允许信号$\overline{OE}$：输入引脚，低电平有效。
    - 低电平时，输出缓冲器与输出信号线连通。
    - 高电平时，输出缓冲器处于高阻态，与输出信号线断开。
    - 通常接地。



92

2.4.3 总线接口结构

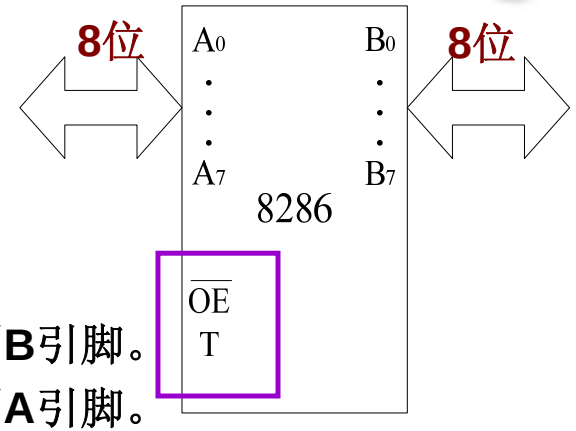
- 双向总线驱动器
  - 作用
    - ✧当系统中存储器和I/O接口数目较多时，若它们同时接收数据，则存在超过微处理器数据总线带负载能力的可能，导致数据传送不可靠。
    - ✧连接外围设备的电缆是电容性负载，线缆上存在的分布电容同样会对数据传送造成的不利影响。
    - ✧因此，需要使用总线驱动器。
  - ✧特点
    - ✧信息在微处理器与存储器或I/O接口之间的传送是双向的，故要求总线驱动器是双向的。
    - ✧又称总线收发器。

93

2.4.3 总线接口结构

双向总线驱动器（续）

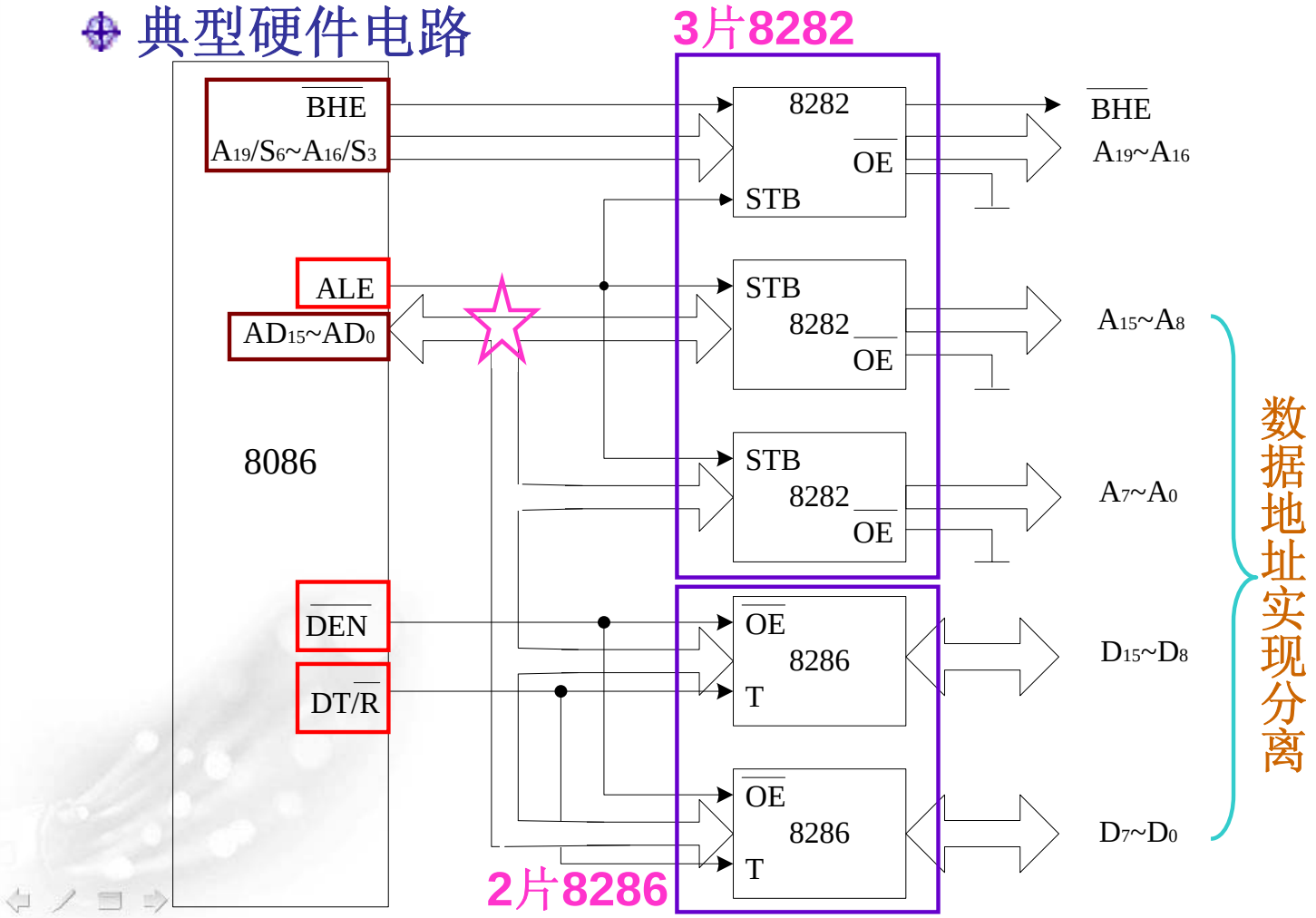
- 常用总线驱动器——8286芯片。
  - ✧8位收发器。
  - ✧收发方向控制信号T：输入引脚。
    - 高电平时，数据从A引脚传送到B引脚。
    - 低电平时，数据从B引脚传送到A引脚。
  - ✧输出允许信号$\overline{OE}$：输入引脚，低电平有效。
    - 低电平时，A引脚与B引脚连通。
    - 高电平时，处于高阻态， A引脚与B引脚断开。



94

2.4.3 总线接口结构

典型硬件电路



95

2.4 8086系统的组成

1. 存储器组织与存储器分段
2. 输入/输出结构
3. 总线接口结构
4. 8086的两种组态

96

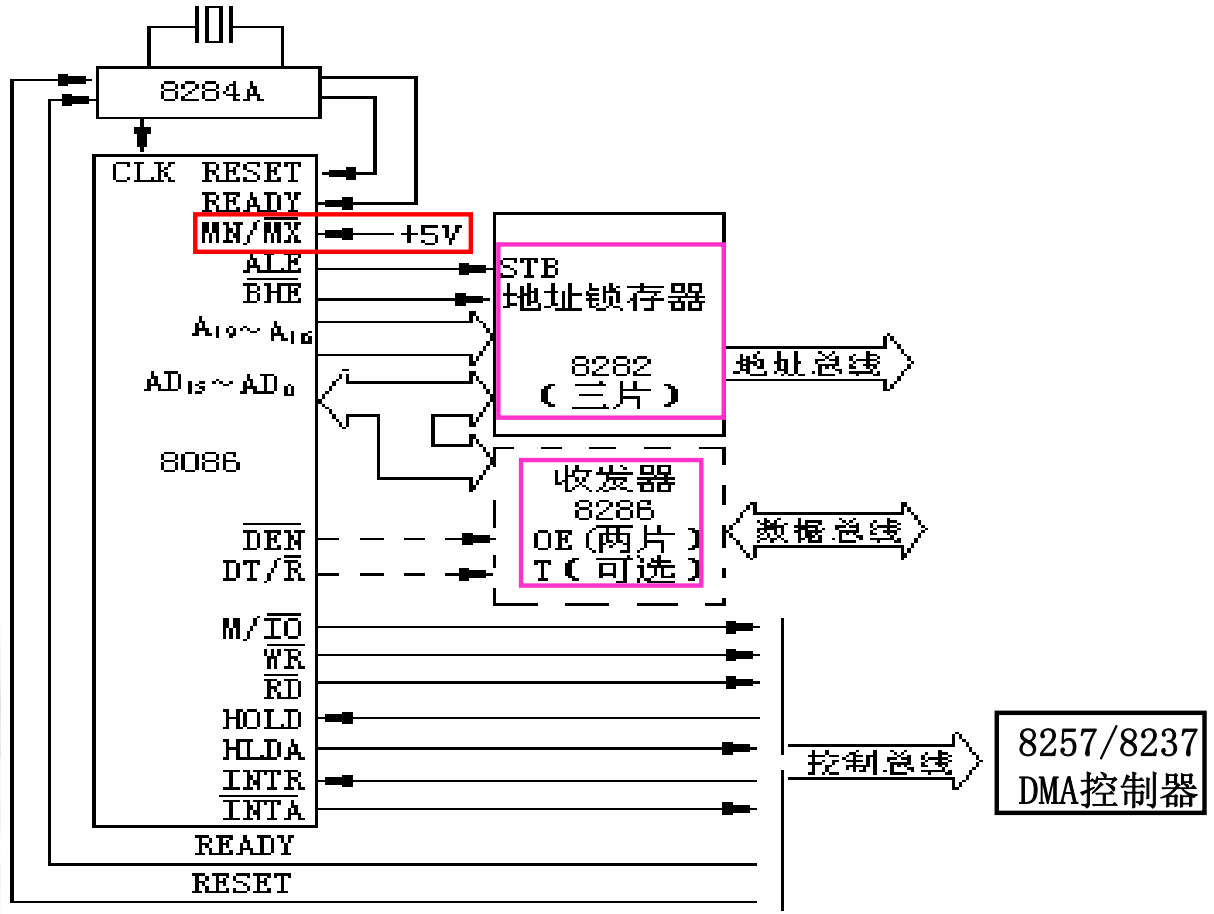
2.4.4 8086的两种组态

- 回顾
  - 8086微处理器的最小模式和最大模式。
    - 最小模式：单处理器；最大模式：多处理器。
    - 由MN/$\overline{MX}$引脚的电平决定。
- 组态
  - 两种工作模式下，微处理器与各功能器件的连接方式。

97

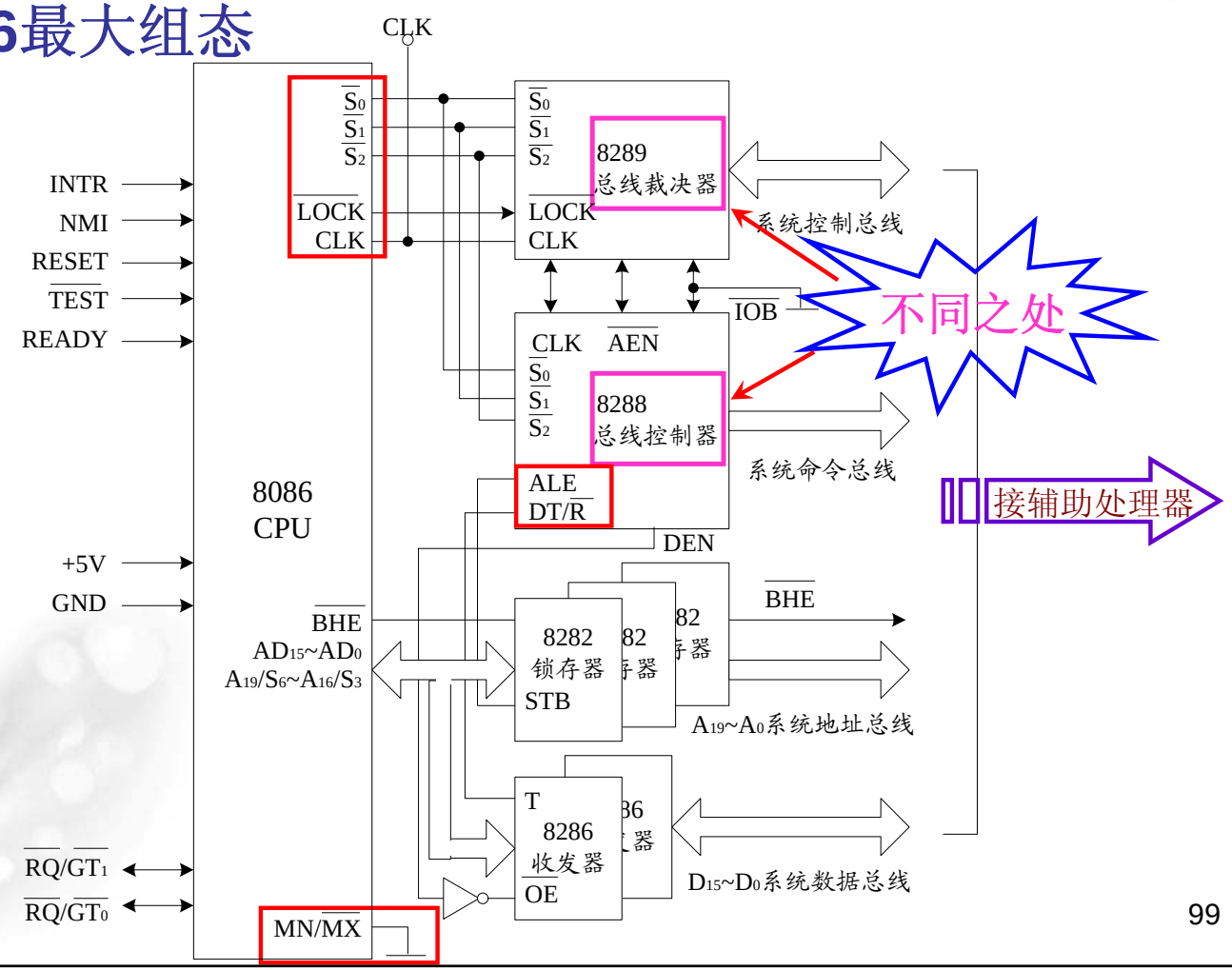
2.4.4 8086的两种组态

8086最小组态



2.4.4 8086的两种组态

8086最大组态

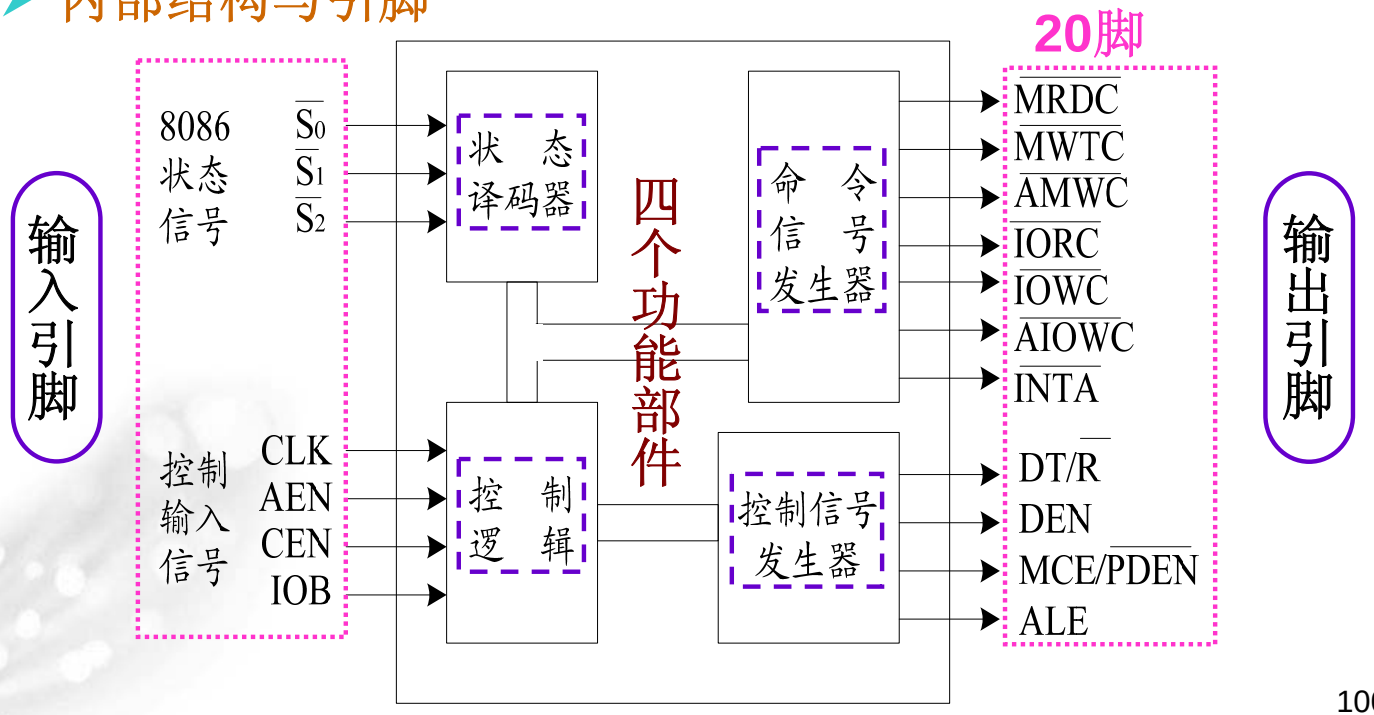


99

2.4.4 8086的两种组态

8288总线控制器

- 用途：产生满足最大模式下总线访问要求的各种定时命令和控制信号。
- 内部结构与引脚



2.4.4 8086的两种组态

8288总线控制器（续）

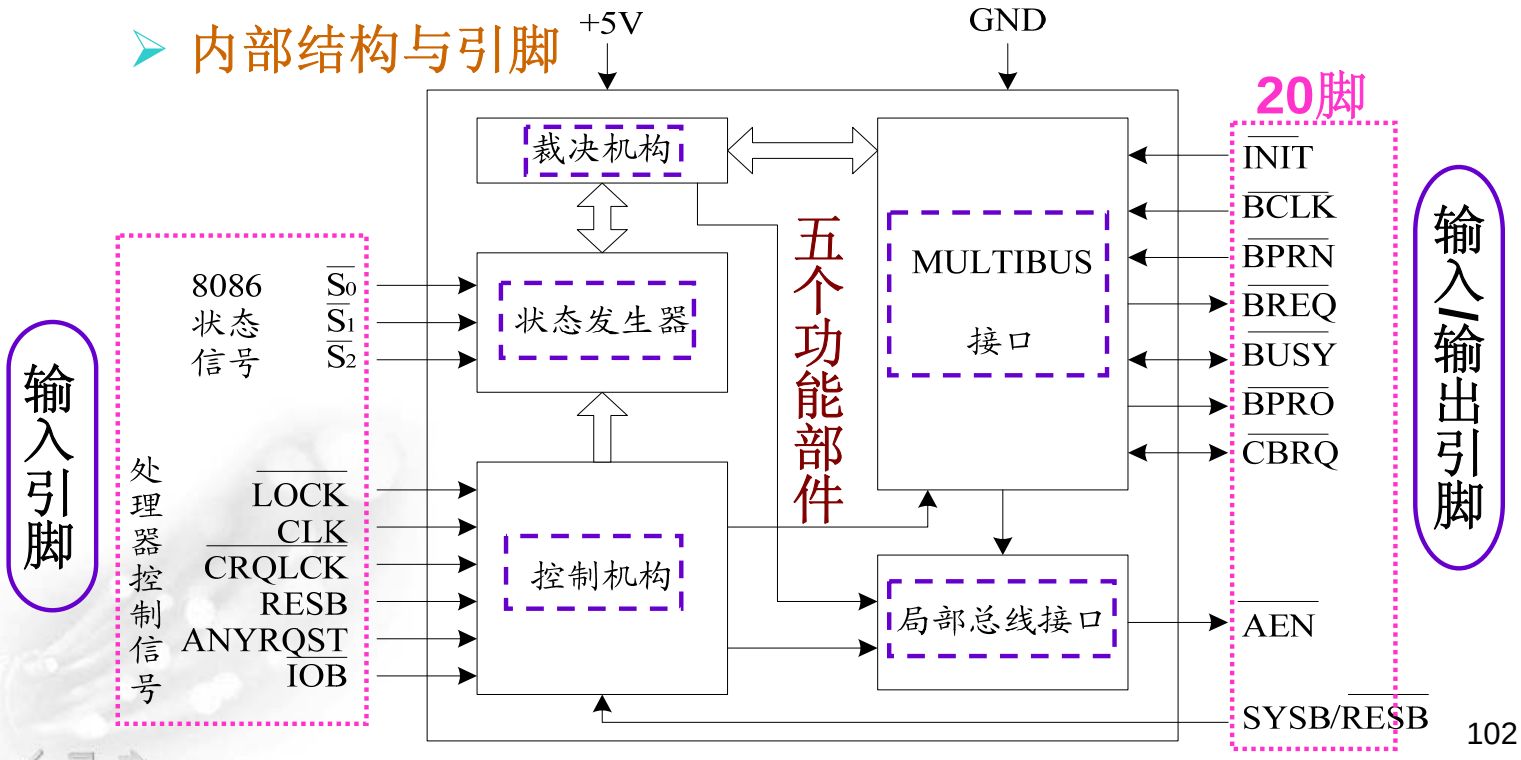
- 工作原理：对输入引脚上引入的8086三位总线状态码$S_2S_1S_0$进行译码，产生7个控制命令。

输入状态 $\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	CPU总线周期	8288输出命令
0	0	0	中断响应	$\overline{INTA}$
0	0	1	读I/O端口	$\overline{IORC}$
0	1	0	写I/O端口	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	停机	无
1	0	0	取指令	$\overline{MRDC}$
0	0	1	读存储器	$\overline{MRDC}$
1	1	0	写存储器	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	无	无

2.4.4 8086的两种组态

8289总线裁决器

- 用途：协调系统总线上各处理器(总线主设备)的动作，以避免多个总线主设备之间的竞争问题 。
- 内部结构与引脚



2.4.4 8086的两种组态

8289总线裁决器（续）

- 工作原理：对来自8086、8087、8089处理器的输入状态码$S_2S_1S_0$进行译码，从而产生多总线命令信号。
  - $\overline{\text{INIT}}$：初始化信号，用来复位所有的总线裁决器。
  - $\overline{\text{BCLK}}$：多总线主设备的系统总线时钟信号，系统总线上的所有接口信号都与BCLK同步。
  - $\overline{\text{BPRN}}$：总线优先权输入信号。当它有效时，其裁决器就知道它可在下一个BCLK的下降沿获得系统总线。
  - $\overline{\text{BREQ}}$：总线请求信号。在并行优先权判别方式中，总线裁决器通过激活其BREQ信号，来请求使用多个主设备的系统总线。
  - $\overline{\text{BPRO}}$：总线优先权输出信号。在串行优先权判别方式中，BPRO总是接到下一个优先权较低的裁决器的BPRN端，即把所有总线裁决器连成一个菊链。

103

2.4.4 8086的两种组态

8289总线裁决器（续）

- $\overline{\text{BUSY}}$：总线忙信号。当系统总线可供使用时(处于空闲状态)，发出总线请求的裁决器中优先权最高者将取得总线，并将BUSY置为低电平，以阻止其他裁决器使用该总线。一旦此总线裁决器完成总线操作，它就允许BUSY变为高电平(不忙)，从而使另一个总线裁决器可以获得系统总线。
- $\overline{\text{CBRQ}}$：公共总线请求信号。当作为输入信号使用时，该信号告诉裁决器是否存在其他优先级较低的裁决器在请求使用系统总线。系统总线上的所有8289总线裁决器的CBRQ引脚都被连在一起。

104

2.4.4 8086的两种组态

补充：协处理器

- 与8086/8088CPU配合工作的协处理器有两类：
  - 数值协处理器8087
    - 能实现多种类型的数值运算，如高精度的整型和浮点型数值运算，超越函数（三角函数、对数函数）的计算等。
  - 输入/输出协处理器8089
    - 有一套专门用于输入/输出操作的指令系统，可以直接为输入/输出设备服务，使主处理器不再承担这类工作。明显提高了主处理器的效率，尤其是在输入/输出操作比较频繁的系统中。

105

2.4.4 8086的两种组态

- 补充：协处理器
  - **$\overline{\text{BUSY}}$**：总线忙信号。当系统总线可供使用时(处于空闲状态)，发出总线请求的裁决器中优先权最高者将取得总线，并将**BUSY**置为低电平，以阻止其他裁决器使用该总线。一旦此总线裁决器完成总线操作，它就允许**BUSY**变为高电平(不忙)，从而使另一个总线裁决器可以获得系统总线。
  - **$\overline{\text{CBRQ}}$**：公共总线请求信号。当作为输入信号使用时，该信号告诉裁决器是否存在其他优先级较低的裁决器在请求使用系统总线。系统总线上的所有**8289**总线裁决器的**CBRQ**引脚都被连在一起。

106

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

107

2.5 8086系统时钟和总线周期

1. 系统时钟
2. 总线周期

2.5.1 系统时钟

回顾：8086微处理器的CLK引脚

- 作用
  - 引入系统时钟信号，为**8086**微处理器的内部和外部操作提供同步的时间基准信号。
- 信号类型
  - 时钟信号频率为**5MHz**非对称方波信号。
- 问题
  - 方波信号是怎么来的？

⇩

来源于一种称为时钟发生器的功能器件。

109

2.5.1 系统时钟

8284A时钟发生器

用途

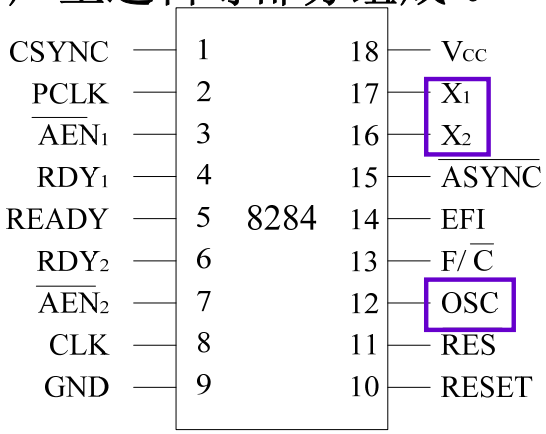
- 为8086微处理器及其它外设芯片提供所需的时钟信号。
- 提供起同步作用的READY(准备好)信号。
- 提供系统复位信号。

内部结构

- 由晶体振荡器、三分频器、多总线准备好(READY)信号控制逻辑、复位(RESET)信号产生逻辑等部分组成 。

引脚功能

- X_1和X_2：接一个15MHz或24MHz的晶体。
- OSC：系统时钟信号输出。

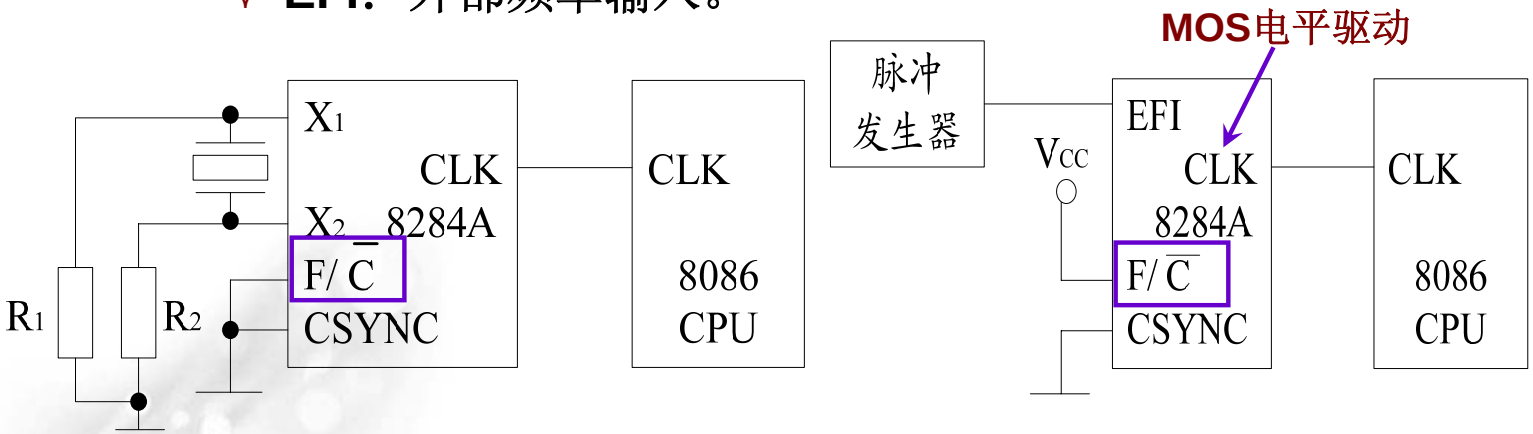


2.5.1 系统时钟

8284A时钟发生器（续）

引脚功能

- **F/$\overline{C}$**：三分频计数器信号源选择输入。
  - **F/$\overline{C}$=0**时，把晶体振荡器作为三分频器的时钟输入；
  - **F/$\overline{C}$=1**时，把**EFI**作为三分频器的时钟输入。
- **EFI**：外部频率输入。



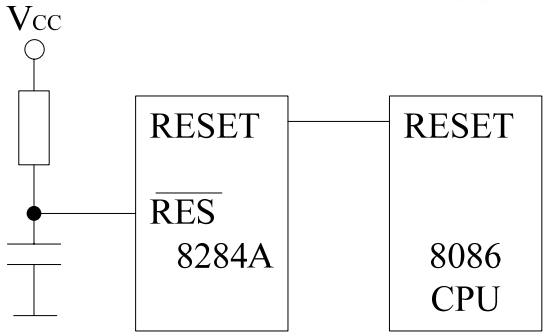
晶体振荡器作信号源　　外部频率作信号源

2.5.1 系统时钟

8284A时钟发生器（续）

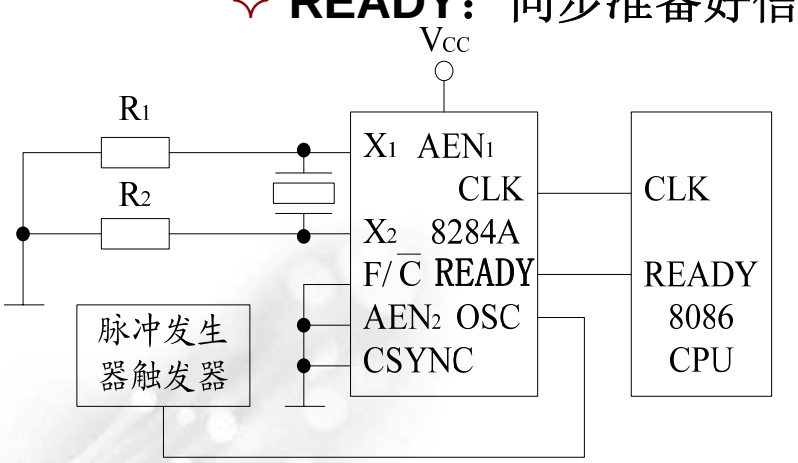
引脚功能

- **RESET**：复位定时信号输出。

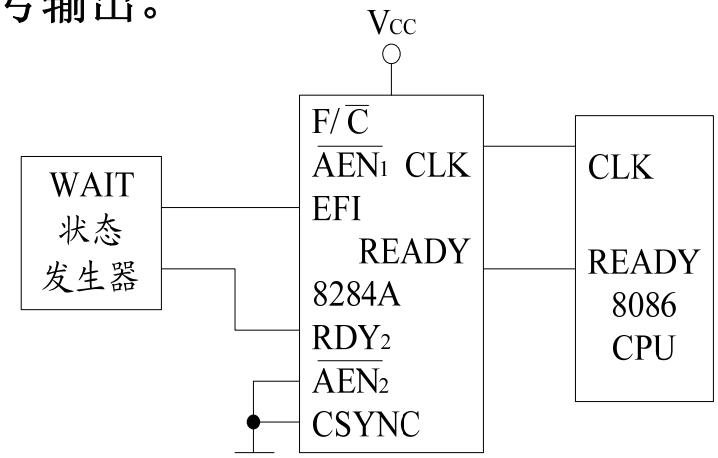


上电复位

- **READY**：同步准备好信号输出。



用晶体震荡同步



用EFI外部频率同步

2.5 8086系统时钟和总线周期

1. 系统时钟
2. 总线周期

2.5.2 总线周期

与周期相关的基本概念

时钟周期

- 是微处理器执行操作的基本时间单位，由CLK引脚引入的系统时钟信号的频率（主频）确定。又称为T状态。
- 如8086的主频为5MHz，则它的时钟周期为200ns。

总线周期

- 微处理器与存储器或外设进行一次数据传送所需的时间。
- 一个总线周期是由若干个T状态组成的。8086的基本总线周期是由4个T状态（T_1、T_2、T_3、T_4）组成的。

等待周期T_W

- 在一个总线周期的T_3和T_4之间插入的一或多个时间间隔。
- 由READY引脚上的信号决定。
- 以时钟周期为单位，即T_W=T。

114

2.5.2 总线周期

与周期相关的基本概念（续）

空闲周期

- 二个总线周期之间的时间间隔。此时总线处在空闲状态。
- 以时钟周期为单位，即T_I=T。

指令周期

- 一条指令从其机器代码被从内存单元中取出，到其所规定的操作执行完毕所用的时间。

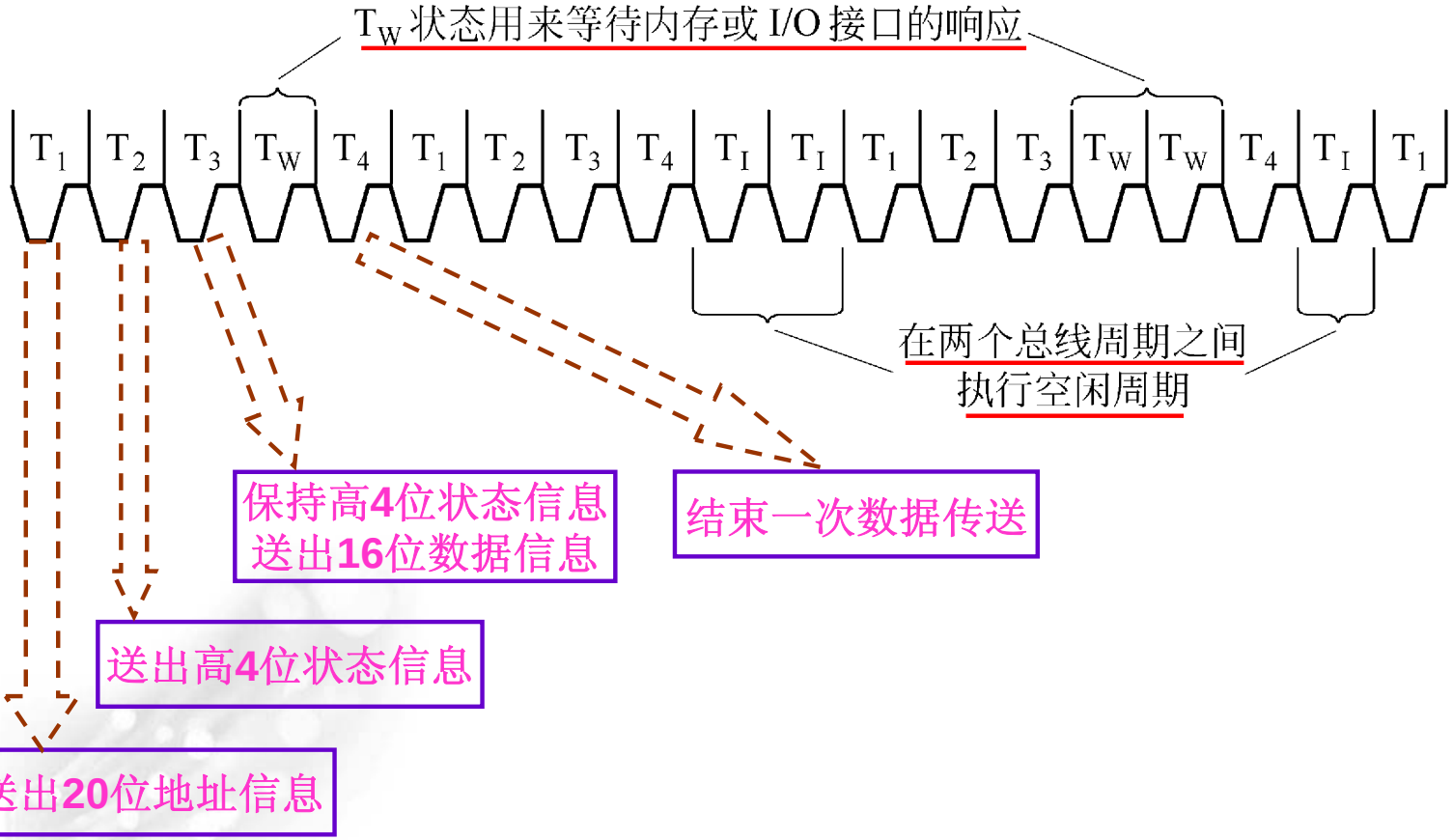
☆几个周期之间的数学关系

- 时钟周期(T)是基本时间单位；
- 一个等待周期T_W=T；
- 一个空闲周期T_I=T；
- 一个总线周期通常由四个T组成，分别称为T_1 T_2 T_3 T_4；
- 一个指令周期由一到几个总线周期组成。

115

2.5.2 总线周期

- 典型的8086总线周期序列

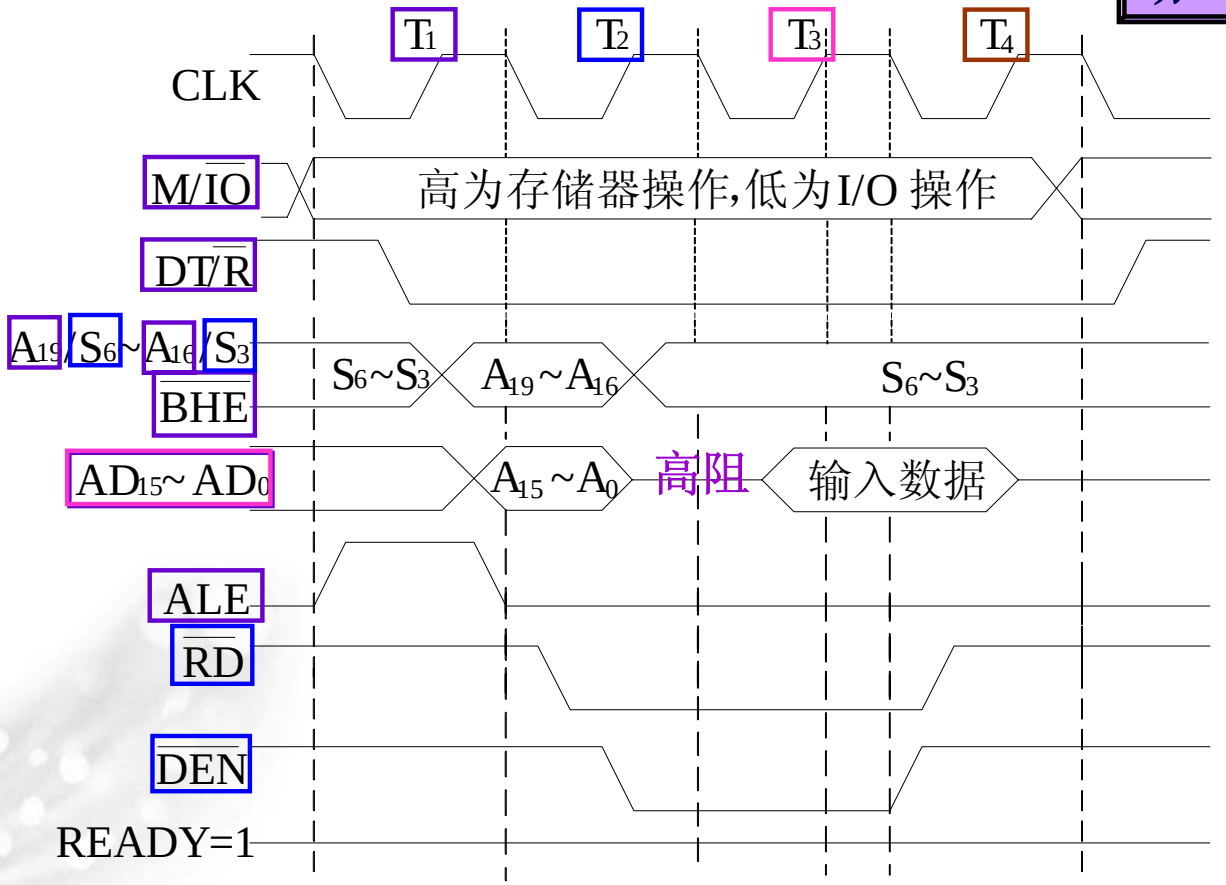


116

2.5.2 总线周期

- 读总线周期的具体过程——最小模式

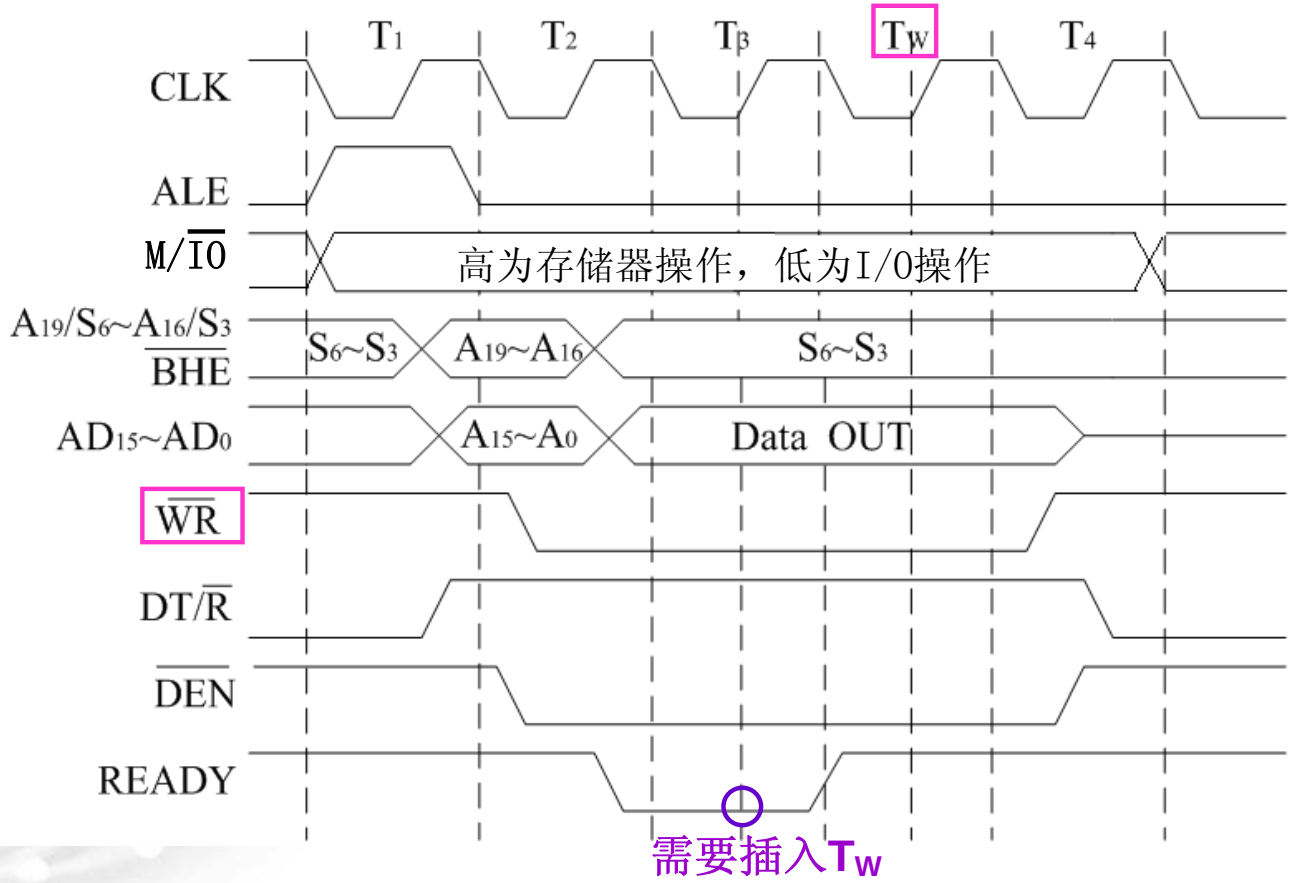
动画演示



注：假设双方速度匹配，否则T3后自动插入Tw，直至READY=1。

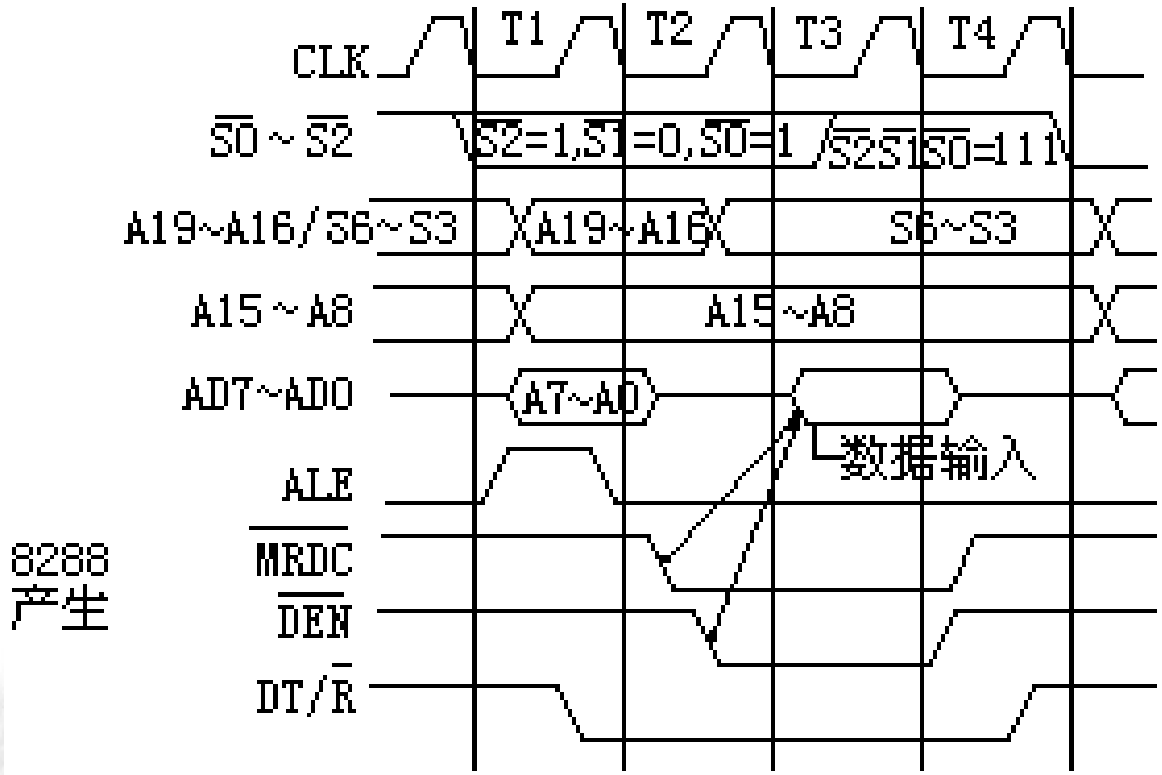
2.5.2 总线周期

- 写总线周期的具体过程——最小模式



2.5.2 总线周期

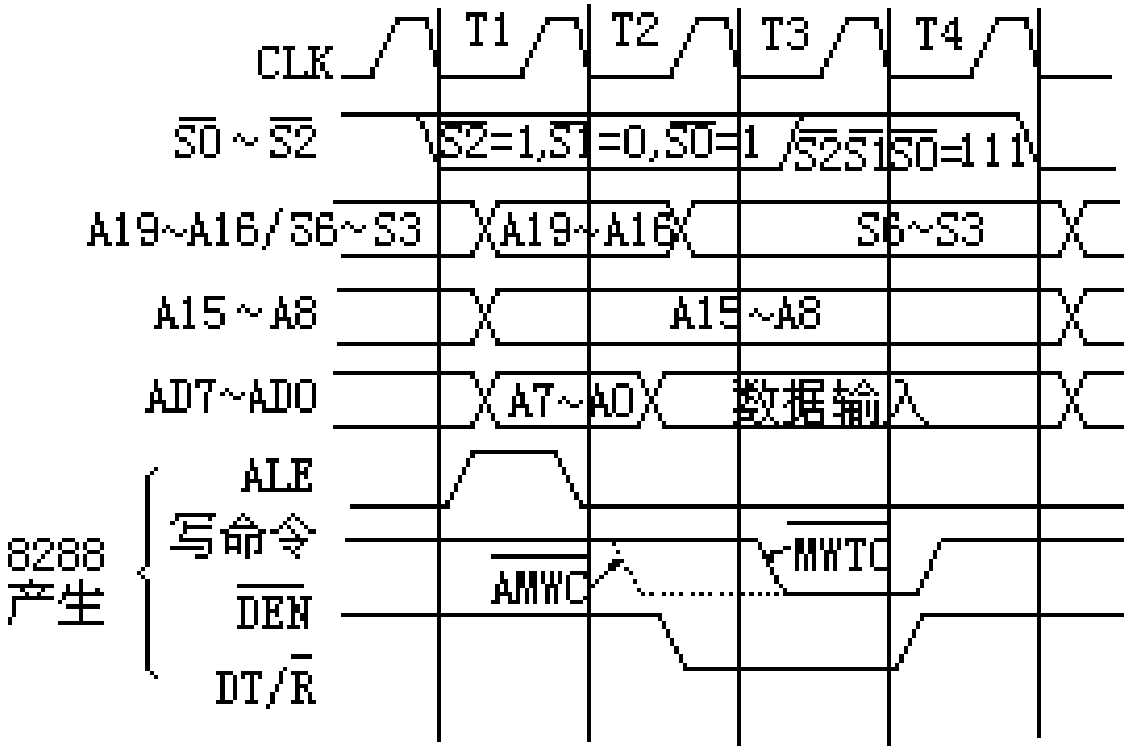
- 读总线周期的具体过程——最大模式（*）



119

2.5.2 总线周期

- 写总线周期的具体过程——最大模式（*）

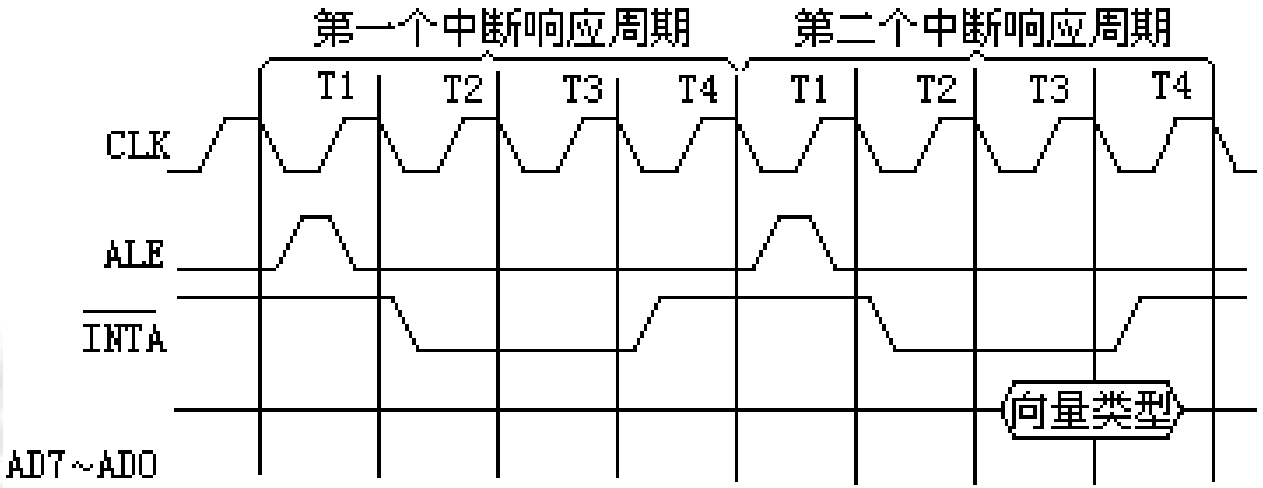


120

2.5.2 总线周期

时序

- 定义：微型计算机执行各种操作的时间顺序。
- 作用：直观地展现了微机系统工作过程中各类信号之间的顺序关系；有助于设计高质量的微机应用系统。
- 表现形式：时序图。



第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

122

2.6 80386微处理器

1. 80386微处理器的主要特性
2. 80386内部基本结构
3. 80386内部寄存器
4. 80386处理器引脚信号
5. 80386工作模式

2.6.1 80386微处理器的主要特性

- 封装
  - 集成了27.5万个晶体管，对外引脚132条，采用陶瓷网格阵列(PGA)封装。
- 寻址能力
  - 采用32位结构，其内部功能部件、数据线和地址线均为32位，故能寻址的物理空间为2^{32}＝4 GByte。
- 存储器管理
  - 片内集成存储器管理部件MMU，支持虚拟存储技术和特权保护技术。
    - 虚拟存储器空间可达64TB。
    - 保护机构采用四级特权层。

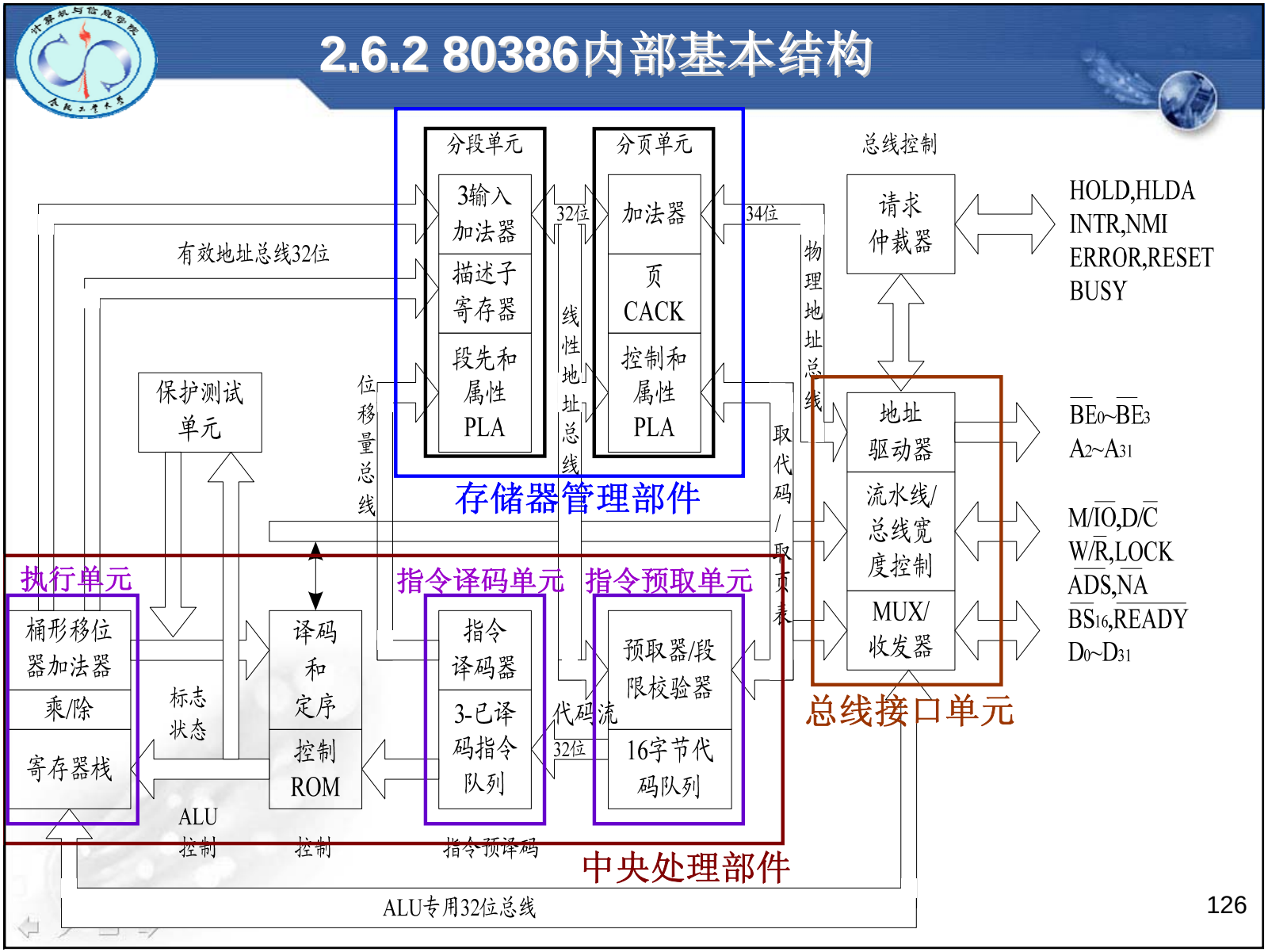
124

2.6 80386微处理器

1. 80386微处理器的主要特性
2. 80386内部基本结构
3. 80386内部寄存器
4. 80386处理器引脚信号
5. 80386工作模式

125

2.6.2 80386内部基本结构



126

2.6.2 80386内部基本结构

总线接口部件BIU

- 提供中央处理部件和系统其他部件之间的高速接口。
- 用于产生访问存储器和I/O口所必须的地址、数据和命令信号。
- 响应取指令、取数据和分页部件产生的请求，并按优先权进行排队。
- 由于总线数据传送与总线地址形成可同时进行，所以总线周期只用2个时钟。

127

2.6.2 80386内部基本结构

中央处理部件CPU

指令预取单元(IPU：Instruction Prefetch Unit)

- 负责从存储器取出指令。有一个16字节的指令队列。
- 把预取总线周期通过分页部件发给总线接口。
- 每当预取队列不满或发生控制转移时，就向BIU发一个取指请求。
- 指令预取的优先级别低于数据传送等总线操作。

指令译码单元(IDU：Instruction Decode Unit)

- 从指令预取单元之中取出指令，进行译码。
- 译码后的可执行指令放入3条已译码队列中，以备执行部件执行。
- 当已译码队列中有空间时，就从预取队列中取出指令并译码。

2.6.2 80386内部基本结构

中央处理部件CPU（续）

执行单元(EU：Execution Unit)

- 包括8个32位的寄存器组、32位算术逻辑单元ALU、一个64位桶形移位寄存器和一个乘法/除法器。
- 桶形移位寄存器用来有效地实现移位、循环移位和位操作。可以在一个时钟周期内实现64位同时移位，也可对任何一种数据类型移任意个位数。
- 桶形移位器与ALU并行操作，可加速乘法、除法、位操作、移位和循环移位操作。

129

2.6.2 80386内部基本结构

- 存储器管理部件MMU
  - 分段单元(SU：Segmentation Unit)
    - 应执行部件的请求，把逻辑地址转换成线性地址。
    - 同时执行总线周期分段的违章检验工作。
    - 可以实现任务之间的隔离以及指令和数据区的再定位。
  - 分页单元(PU：Paging Unt)
    - 把由分段单元或代码预取单元产生的线性地址转换成物理地址，并且要检验访问是否与页属性相符合。
    - 为加快地址转换速度，内设有一个页描述符高速缓冲存储器(TLB)，可存储32项页描述符，在地址转换期间大部分情况不需要到内存中查页目录表和页表。
    - 对于TLB没有命中的地址转换，设有硬件查表功能，从而使因查表引起的速度下降明显好转。

2.6 80386微处理器

1. 80386微处理器的主要特性
2. 80386内部基本结构
3. 80386内部寄存器
4. 80386处理器引脚信号
5. 80386工作模式

2.6.3 80386内部寄存器

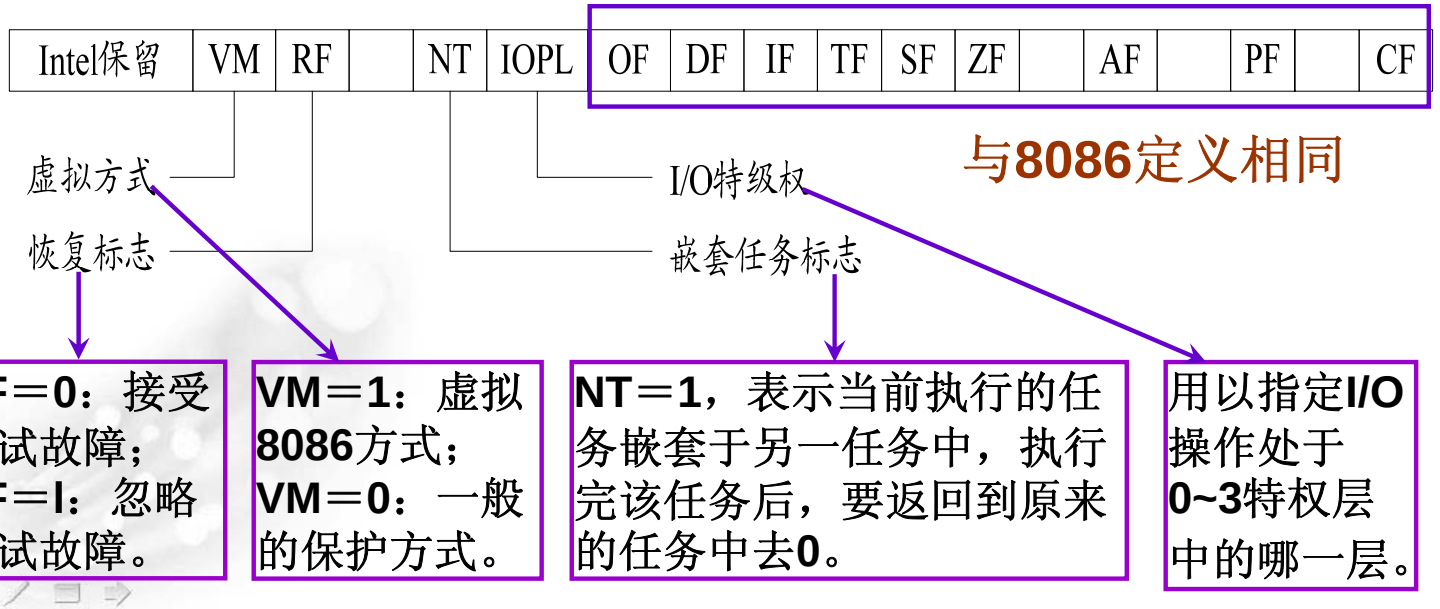
通用寄存器

- **80386有8个32位的通用寄存器。**
- 和**8086**通用寄存器相同，但位数为**32**位。为表示区别，需要在原来的寄存器名字前加一字符**E**，即：**EAX，EBX，ECX，EDX，ESP，EBP，ESI，EDI。**
- 支持**8**位和**16**位操作，用法和**8086**相同。

2.6.3 80386内部寄存器

指令指针和标志寄存器

- 80386的指令指针EIP是一个32位寄存器，与80386的32位地址线对应，是8086的IP的扩充。
- 80386的标志寄存器EFLAGS也是一个32位寄存器，其中定义了15位。



2.6.3 80386内部寄存器

控制寄存器

- **80386有4个32位控制寄存器CR_0，CR_1，CR_2和CR_3。**
- 用于控制选择**80386**的操作环境(实地址方式、保护方式、分页保护环境)、协处理器的使用控制以及作为线性地址的故障页保护和页目录表的基址保护。

段寄存器

- **6个段寄存器：CS，SS，DS，ES，FS和GS。增加了FS与GS，用于减轻对DS段和ES段的压力。、**
- 每一个段寄存器都有一个关联的段描述符寄存器，用来描述一个段的段基地址、段限和段的属性。
- 硬件自动根据段寄存器中的值去索引，从段描述符表中取出一个描述符，装入段描述符寄存器中，并以其中的段基地址作为线性地址计算中的一个元素。

134

2.6.3 80386内部寄存器

系统地址寄存器

- 80386有4个系统地址寄存器，用来保护操作系统需要的保护信息和地址转换表信息，定义目前正在执行任务的环境、地址空间和中断向量空间。
  - **GDTR：** 48位全局描述符表寄存器，用于保存全局描述符表32位线性基地址和16位全局描述符表界限。
  - **IDTR：** 48位中断描述符表寄存器，用于保存中断描述符表32位线性基地址和16位中断描述符表界限。
  - **LDTR：** 16位局部描述符表寄存器，用于保存局部描述符表段的选择器。
  - **TR：** 16位任务状态寄存器，用于保存任务状态段(TSS)的16位选择器。

135

2.6.3 80386内部寄存器

调试寄存器

- 80386设有8个32位调试寄存器DR_0~DR_7，为调试提供了硬件支持。
  - DR_0~DR_3为保存4个线性断点地址寄存器。
  - DR4、DR5为备用寄存器。
  - DR6为调试状态寄存器，通过该寄存器的标志可以检测异常并进入异常处理程序或禁止进入异常处理程序。
  - DR7为调试控制寄存器，用来规定断点字段的长度、断点访问类型、“允许”断点和“允许”所选择的调试条件。

2.6.3 80386内部寄存器

测试寄存器

- 80386设置了8个32位的测试寄存器TR_0 ~ TR_7，
  - TR_0 ~ TR_5是Intel公司保留的。
  - TR6是测试控制寄存器，TR7是测试状态寄存器，保存测试结果的状态。用于控制对TLB中的RAM和CAM相连存储器的测试

137

2.6 80386微处理器

1. 80386微处理器的主要特性
2. 80386内部基本结构
3. 80386内部寄存器
4. 80386处理器引脚信号
5. 80386工作模式

2.6.4 80386处理器引脚信号

- **CLK2**：两倍时钟信号输入引脚。
  - 该信号与80384时钟信号同步输入，在80386内部二分频后产生指令执行时钟CLK。每个CLK由两个CLK2时钟周期组成，分别称其为相1和相2。
- **D_0 ~ D_{31}**：数据总线信号引脚，双向三态。
  - 一次可传送8、16、32位数据。
- **A_2 ~ A_{31}**：地址总线信号输出引脚，三态。
  - 和BE_0 ~ BE_3相结合可起到32位地址作用。
- **BE_0 ~ BE_3**：字节选通信号输出引脚。
  - 每条线控制选通一个字节，分别对应选通D_0 ~ D_7、D_8 ~ D_{15}、D_{16} ~ D_{23}与D_{24} ~ D_{31}，相当于分为4个存储体。
  - 与A_2 ~ A_{31}结合可寻址2^{32}＝4 GB个单元。

2.6.4 80386处理器引脚信号

- **W/R**：读/写控制信号输出引脚。
- **D/C**：数据/控制信号输出引脚。
  - 表示是数据传送周期还是控制周期。
- **M/IO**：存储器与I/O选择信号输出引脚。
- **LOCK**：总线锁定信号输出引脚。
- **ADS**：地址状态信号输出引脚，三态。
  - 表示总线周期中地址信号有效。
- **NA**：下一地址请求信号输入引脚。
  - 允许地址流水线操作，即当前周期发下一总线周期地址的状态信号。

140

2.6.4 80386处理器引脚信号

- **BS_{16}：** 总线宽度为16的信号输入引脚。
- **READY：** 准备就绪信号输入引脚。
  - 表示当前总线周期已完成。
- **HOLD：** 总线请求保持信号输入引脚。
- **HLDA：** 总线响应保持信号输出引脚。

141

2.6.4 80386处理器引脚信号

- **PEREQ**：处理器扩展请求信号输入引脚。
  - 表示80387要求80386控制它们与存储器之间的信息传送。
- **BUSY**：协处理器忙信号输入引脚。
- **ERROR**：协处理器出错信号输入引脚。
- **NMI**：不可屏蔽中断请求信号输入引脚。
- **INTR**：可屏蔽中断请求信号输入引脚。
- **RESET**：复位信号输入引脚。

142

2.6 80386微处理器

1. 80386微处理器的主要特性
2. 80386内部基本结构
3. 80386内部寄存器
4. 80386处理器引脚信号
5. 80386工作模式

143

2.6.5 80386工作模式

实地址模式

- 80386加电系统复位后，工作于该模式。
- 只有地址线A_2~A_{19}和BE_0~BE_3是有效的，而A_{20}~A_{31}总是低电平。因此寻址空间为1MB 。
- 保留了两个固定的专用存储区域。
  - 中断向量表区：**00000H～003FFH**，在**1K**字节存储空间保留**256**个中断服务程序的入口地址，每个入口地址占用**4**个字节，这与**8086**一样。
  - 系统初始化区：**FFFFFFF0H～FFFFFFFFH**，存放**ROM**引导程序。

144

2.6.5 80386工作模式

保护虚地址模式

- 线性地址空间增加到**4 GB**，虚拟存储器地址空间可达**64 TB**。
- 提供了复杂的存储管理和硬件辅助的保护机构，以及支持多任务操作系统的特别优化的指令。
- 虚拟地址空间
  - 由磁盘等外部存储器支持实现，它与微处理器的存储器管理部件、与**48**位和**16**位的描述符表寄存器有直接关系。
  - 程序可以存放在磁盘存储器上，但执行时必须加载到物理存储器上。这就存在**46**位虚拟地址变换成**32**位物理地址的问题。

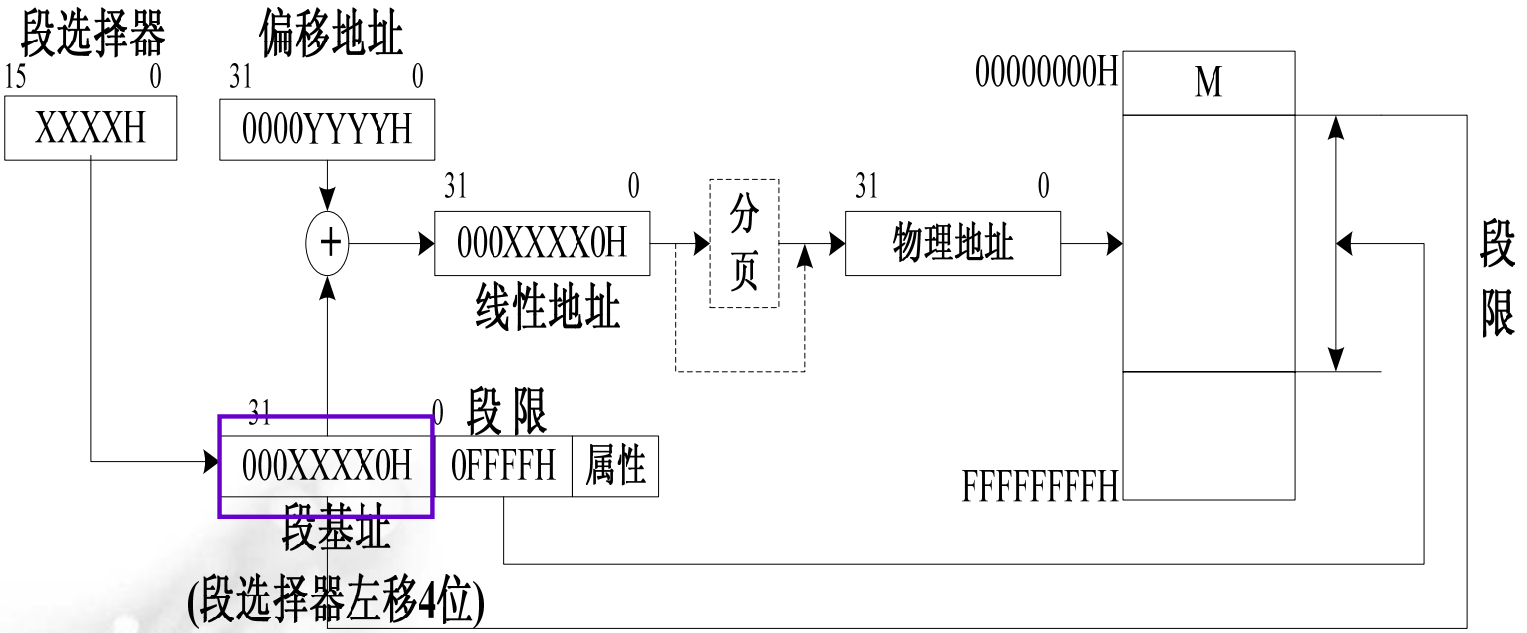
145

2.6.5 80386工作模式

- 保护虚地址模式下的地址变换
  - 32位的段基地址存放在一个段描述符表中，
  - 16位段寄存器的内容作为选择器，即作为段描述符表的索引，可以从表中取出相应的段描述符（包括32位段基地址、段界限和访问权等）。
  - 在进行地址转换时，由段选择器左移4位得到描述符寄存器中的32位段基地址，与对应的32位偏移地址相加得到线性地址，再通过分页机构进行变换，得到物理地址。如果不分页，线性地址就等于物理地址。

2.6.5 80386工作模式

- 保护虚地址模式下的地址变换（续）



2.6.5 80386工作模式

段描述符表

段选择器

✧在保护虚地址模式下，段寄存器就成为一个段选择器，它由3个字段组成。

15 3	2	1 0
索　引	TI	RPL

✧索引字段：给出要选择的段描述符表中的位置，又称描述符偏移地址。

✧TI：描述符表指示器。TI＝0，表示指向全局描述符表GDT；TI＝1，表示指向局部描述符表LDT。

✧RPL：选择器特权级。定义当前请求的特权层级别，有4级特权级0~3，0级最高，3级最低。

148

2.6.5 80386工作模式

段描述符表（续）

段描述符

- 用段选择器从段描述符表中所选择的对象称为段描述符。
- 段描述符包含了一个存储分段的所有信息，包括段的线性基地址和此段的界限(大小)和段的一些属性。
- 段的属性包括：
  - 段的保护等级。
  - 读、写、执行特权和保护特权级别。
  - 操作数的默认长度 (16或32位)。
  - 段的类型。
  - 段的粒度（granularity），即段的长度的基本单位。

149

2.6.5 80386工作模式

段描述符表（续）

段描述符

由8个字节构成，其中段基地址共32位(4个字节)，段界限字段是20位，还有12位定义了段的属性信息。

31					16	15					0
段基地址(15～0)						段界限(15～0)					
段基地址(31～24)	G	D	0	AVL	段界限(19~16)	P	DPL	S	TYPE	A	段基地址(23～16)

- A：访问位；
- TYPE：段类型；
- S：段描述符；
- DPL：描述符特权级；
- P：存在位；
- G：粒度位；
- D：缺省操作数的大小(仅用于代码段描述符)；
- AVL：用户或OS可以使用。

150

2.6.5 80386工作模式

段描述符表（续）

段描述符

- ✧段基地址字段规定了存储分段在线性地址空间中的基地址。
- ✧段界限字段指定了该存储分段的长度。
- ✧粒度位**G＝0**时，表示段长是以字节为单位，最大地址空间**1MB**；**G＝1**时，表示段长是以页为单位，最大地址空间是**4 GB**。

31					16	15					0
段基地址(15～0)						段界限(15～0)					
段基地址(31～24)	G	D	0	AVL	段界限(19~16)	P	DPL	S	TYPE	A	段基地址(23～16)

2.6.5 80386工作模式

段描述符表（续）

段描述符

- **D位**指示了操作数和有效地址的缺省长度。**D=1**，使用**32**位操作数和**32**位寻址方式；**D=0**，使用**16**位操作数和**16**位寻址方式。
- **P位**表示该存储分段是否在内存。**P＝l**，表示该分段已在内存；**P＝0**，表示该分段在外存硬盘交换区。

31					16	15					0
段基地址(15～0)						段界限(15～0)					
段基地址(31～24)	G	D	0	AVL	段界限(19~16)	P	DPL	S	TYPE	A	段基地址(23～16)

2.6.5 80386工作模式

段描述符表（续）

段描述符

✧S、TYPE和A位有三种组合方式：数据段或堆栈段(S＝1且TYPE＝0)；代码段(S＝l且TYPE＝1)；系统段(S＝0)。

✧系统段描述操作系统的系统表、任务和门的信息；非系统段就是代码和数据段。

31					16	15					0
段基地址(15～0)						段界限(15～0)					
段基地址(31～24)	G	D	0	AVL	段界限(19~16)	P	DPL	S	TYPE	A	段基地址(23～16)

153

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

154

2.7 80486微处理器

- 封装
  - 对外引脚**168**条，采用**CHMOSI**艺**PGA**封装，在一单片上集成了**120**万个晶体管 。
- 寻址能力
  - 与80386相同，能寻址的物理空间为2^{32}＝4GB。
- 特性
  - 可以满足对图形用户接口（**GUI**）、多媒体和数字图像等方面的应用要求。

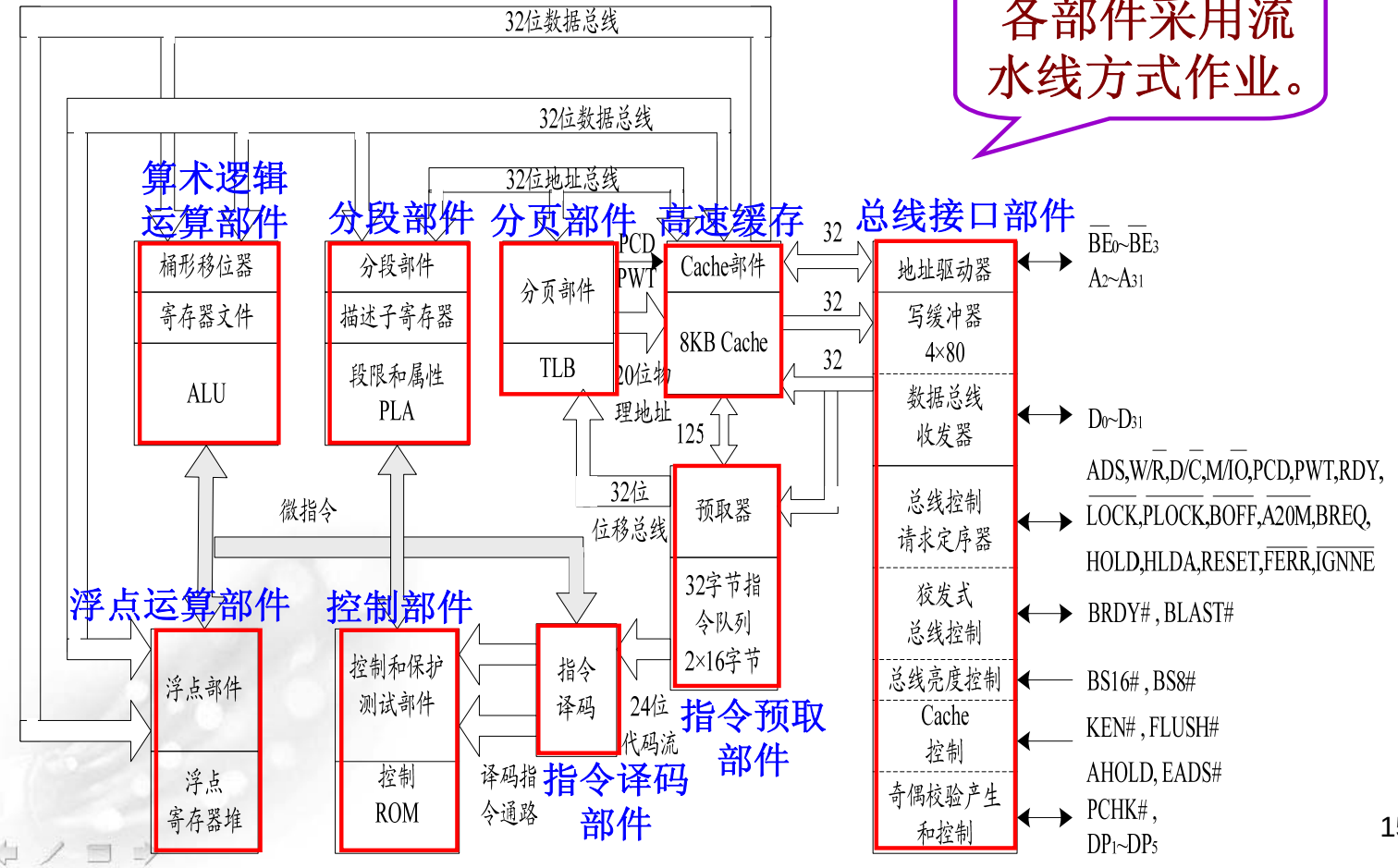
155

2.7 80486微处理器

1. 80486内部结构
2. 80486 CPU的特点
3. 80486 CPU主要引脚信号

2.7.1 80486内部结构

九大功能部件



各部件采用流水线方式作业。

157

2.7 80486微处理器

1. 80486内部结构
2. 80486 CPU的特点
3. 80486 CPU主要引脚信号

158

2.7.2 80486 CPU的特点

- 采用精简指令集计算机(RISC：Reduced Instruction Set Computer)技术。
  - 频繁使用的基本指令由以前的微代码控制，改为用布线逻辑直接控制，极大提高了指令译码和执行速度。
  - 如寄存器间的加减、逻辑运算指令，1个时钟周期即可执行，而80386需要2～4个时钟周期。
- 采用突发式总线(Burst Bus)技术。
  - 2-1-1-1突发总线周期：在5个时钟周期内取16个字节的信息，2个时钟周期用于地址和第一个双字，接着3个时钟取3个顺序的双字。
  - 效率比最快的标准总线周期高60%。

159

2.7.2 80486 CPU的特点

- 含有8 KB/16KB的数据和指令混合型高速缓存器(Cache)。
  - 当微处理器访问存储器时，若所需的指令和数据驻留在Cache中，就称作“命中”。
  - 命中的指令和数据可从Cache中直接取出，不必再从内存中读取。
  - 常用指令和数据在Cache的命中率可以高达90%，即大部分信息都可很快地直接从片内Cache中提供。
  - 若信息在Cache中未命中，微处理器再从内存中去读取指令或数据，如此可以减少系统的等待时间。

2.7.2 80486 CPU的特点

- 集成了增强型80387浮点运算器(FPU) 。
  - FPU在微处理器芯片内部，提高了浮点部件与其它内部单元的接口效率，且它们之间的通信是同步的。
  - Cache与FPU之间有两条32位总线，它们可作为一条64位总线使用，一次可完成双精度数据的传送。
- 采用一种新的系统管理方式(SMM)。
  - 监视微处理器、各种控制器、其它部件的不工作时间。
  - 若不工作时间超过预置值，微处理器就使用系统管理方式，有选择地关掉不工作的子系统。
  - Intel和Microsoft联合开发的先进功率管理(APM)软件专门用来实现硬件功率管理。

2.7 80486微处理器

1. 80486内部结构
2. 80486 CPU的特点
3. 80486 CPU主要引脚信号

162

2.7.3 80486 CPU主要引脚信号

- A_2 ~ A_{31}：地址信号。
- A20M：地址位20屏蔽。
- D_0 ~ D_{31}：数据信号。
- BE_0 ~ BE_3：字节允许信号，分别对应D_0 ~ D_7、D_8 ~ D_{15}、D_{16} ~ D_{23}与D_{24} ~ D_{31}4个字节。
- BS_8和BS_{16}：8位或16位数据总线宽度控制信号，BS_{16}或BS_8有效时，选择16位或8位数据总线。
- PD_0 ~ PD_3：数据奇偶校验。
- PCHK：奇偶校验错误。

163

2.7.3 80486 CPU主要引脚信号

- **ADS**：地址状态信号。
- **M/IO**：内存或I/O选择。
- **D/C**：数据与代码选择。
- **W/R**：写/读信号。
- **RDY**：非突发数据就绪信号。
- **BRDY**：突发数据准备就绪信号输入。
- **BLAST**：突发结束输出，用来终止高速缓存的行填充或其他多数据周期的传送。
- **KEN**：内部cache允许信号。

2.7.3 80486 CPU主要引脚信号

- **PCD**：页高速缓存禁止输出信号。PCD＝I时，禁止以页为单位的Cache操作。
- **PWT**：页通写输出，是以页为单位的写操作方式控制信号。PWT＝1表示写操作命中时既要写Cache，也要写内存。
- **EADS**：外部地址，指明在80486地址引脚上已放置了某一有效地址，使80486去读取外部总线地址，并与Cache登记的地址比较检验，如相等则使Cache中相应区域内容失效。
- **AHOLD**：地址总线保持(内部)信号。
- **FLUSH**：cache刷新(内部)信号。

165

2.7.3 80486 CPU主要引脚信号

- **FERR**：浮点错误信号。
- **IGNNE**：忽略浮点错误信号。
- **LOCK**：总线锁定信号。
- **PLOCK**：总线伪锁定信号。
- **BREQ**：总线请求输出，每当总线周期请求在内部挂起时，**80486**就发出这个信号。
- **BOFF**：总线屏蔽输入，它有效时强制**80486**在下一时钟浮空其总线。

166

2.7.3 80486 CPU主要引脚信号

- **HOLD**：总线请求保持信号。
- **HLDA**：总线响应信号。
- **NMI**：不可屏蔽中断请求信号。
- **INTR**：可屏蔽中断请求信号。
- **RESET**：系统复位信号。

2.7.2 80486 CPU的特点

- 采用精简指令集计算机(RISC：Reduced Instruction Set Computer)技术。
  - 频繁使用的基本指令由以前的微代码控制，改为用布线逻辑直接控制，极大提高了指令译码和执行速度。
  - 如寄存器间的加减、逻辑运算指令，1个时钟周期即可执行，而80386需要2～4个时钟周期。
- 采用突发式总线(Burst Bus)技术。
  - 2-1-1-1突发总线周期：在5个时钟周期内取16个字节的信息，2个时钟周期用于地址和第一个双字，接着3个时钟取3个顺序的双字。
  - 效率比最快的标准总线周期高60%。

168

第二章 80x86微处理器

2.1 微处理器的基本结构

2.2 Intel8086微处理器

2.3 8086中的程序状态字和堆栈

2.4 8086系统的组成

2.5 8086系统时钟和总线周期

2.6 80386微处理器*

2.7 80486微处理器*

2.8 Pentium处理器*

169

2.8 Pentium微处理器

1. Pentium处理器的特点
2. Pentium处理器内部框图与信号功能
3. 80486与Pentium总线之间的主要区别

2.8.1 Pentium处理器的特点

- 允许一次执行两条指令。
  - 两个平行的**EU**可增加每个时钟周期执行的指令数。
  - 有三条执行流水线：浮点流水线和两条独立的整数流水线(即**U**和**V**管道)。这种能一次同时执行多条指令的体系结构称为超标量体系结构。
- 外部数据总线宽度增加到**64**位。
  - 借助突发读、写周期能达到**528 MB/s**的最高带宽。
- 有两个独立的**8 KB**高速缓存**(cache)**，
  - 将指令和数据信息分开，允许两个**cache**同时存取，因而使得传输效率更高。

171

2.8.1 Pentium处理器的特点

- 对数据Cache增加了回写能力。
  - 新数据写到高速缓存时，推迟对主存储器的修改。
    - 80486的单高速缓存是以通写方式工作的。在写后虽然数据仍存在高速缓存中，但它要立刻将数据也写到主存中，高速缓存中的数据与主存中的数据总是相同的。
    - 回写式优点
      - 只有必需时才进行主存操作，延迟期间处理器可去进行其它计算；
      - 减少了连接高速缓存和主存的总线的使用时间。当两个或多个处理器试图通过一条总线访问存储器时，减少每个处理器使用总线的时间就非常重要。

172

2.8.1 Pentium处理器的特点

- 利用转移预测部件(BPU)预测程序转移的发生。
  - 使用一个小型的1KB Cache(转移目标缓冲器BTB)来预测转移。
    - BTB记录以前发生的转移，通过跟踪程序，下一次预取到相同的转移要执行时，BTB利用它的信息在执行之前去预测结果：发生还是不发生转移。
    - 当它作出的预测正确时，就可无延迟地执行转移指令，从而避免了因发生程序转移使执行流水线停顿。
  - 程序局部性原则(即程序通常访问最近访问过的存储单元的邻近单元)，使得转移预测部件在10次预测中有9次是正确的，从而让执行流水线性能提高25%。

173

2.8.1 Pentium处理器的特点

- 使用了一种新型的浮点指令部件。
  - 最常用的浮点操作(加、乘和除) 用硬件实现，即增加了专门执行这些指令的电路。
    - 大多数浮点指令都可在一个时钟周期内执行，这比运行在66 MHz的80486DX2的浮点性能提高了4倍。
  - 对于一些先进的大型程序是很需要的。
    - 例如，三维动画的设计软件包和建模软件包，以前只能在工作站上运行。

2.8.1 Pentium处理器的特点

- Pentium的两大技术流派。
  - Pentium Pro(高能奔腾)
  - MMX Pentium(多能奔腾)：在Pentium中追加多媒体扩展功能。
- Pentium II：在Pentium Pro中追加MMX功能。

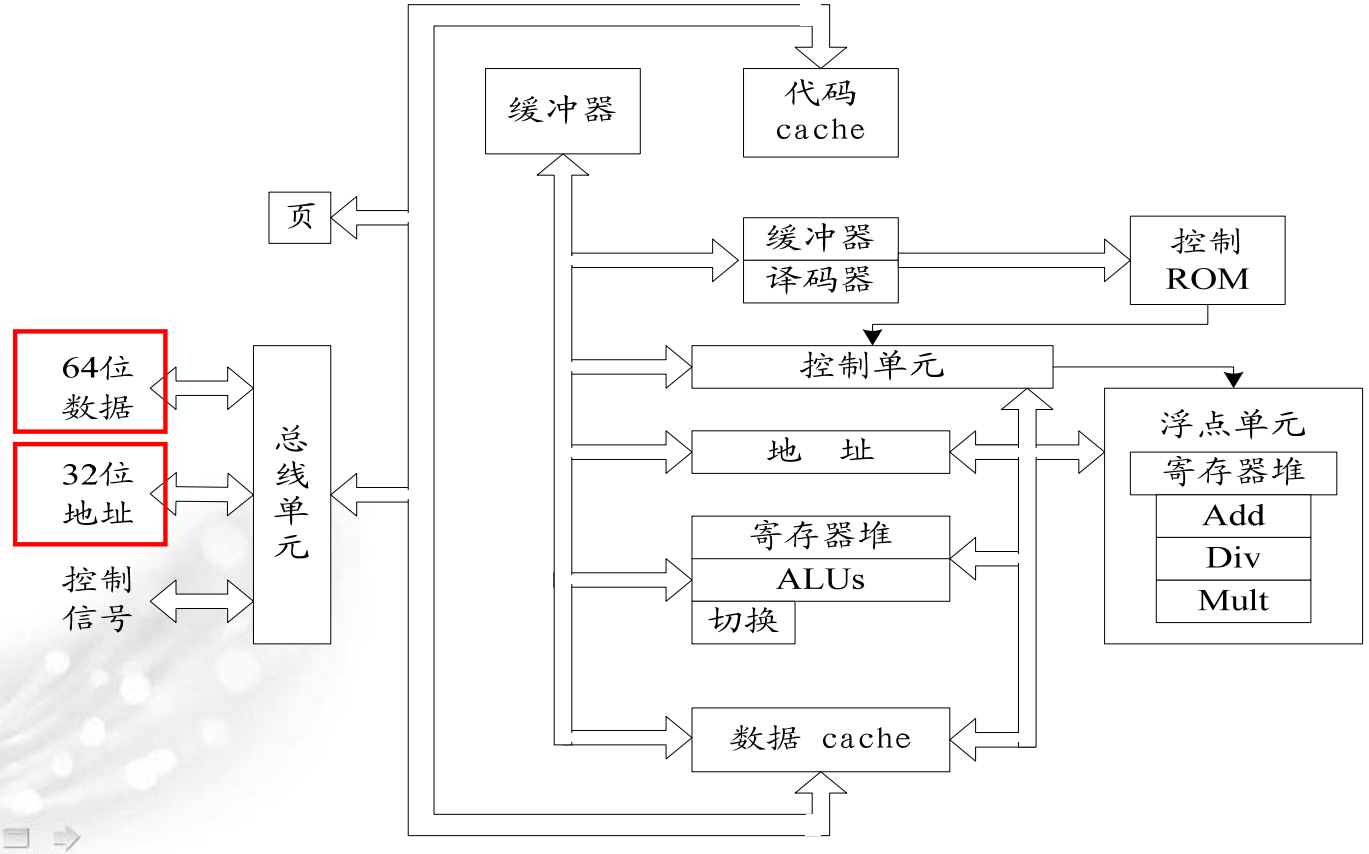
175

2.8 Pentium微处理器

1. Pentium处理器的特点
2. Pentium处理器内部框图与信号功能
3. 80486与Pentium总线之间的主要区别

2.8.2 Pentium处理器内部框图与信号功能

- 对外有237条引脚，采用PGA封装，集成了310万个晶体管 。



2.8.2 Pentium处理器内部框图与信号功能

- **A_3 ~ A_{31}：** 双向地址线。
  - 作为输出时，它与字节允许信号BE_0 ~ BE_7组成地址总线，可寻址4GB物理内存空间与64 KBI/O地址空间。
  - 作为输入时，驱动地址总线回到处理器执行询问周期。
- **ADS：** 地址选通，低电平有效。
  - ADS有效时，表示处理器驱动一新的有效总线周期，且：A_3 ~ A_{31}、AP、BE_0 ~ BE_7、LOCK、M/IO、W/R、D/C、SCYC、PWT和PCD信号被驱动为有效电平。
- **BE_0 ~ BE_7：** 字节允许信号输出，低电平有效。
  - 与地址线一起用来提供内存与I/O口的物理地址。分别对应数据线的D_0 ~ D_7，…， D_{56} ~ D_{63}共8个字节。

178

2.8.2 Pentium处理器内部框图与信号功能

- **A20M：** 地址 20屏蔽 。
  - 用于取消8086的1MB地址空间限制。当这个异步输入为低电平时，在执行内部Cache查找或驱动内存总线周期之前，处理器屏蔽物理地址位20。
- **AP：** 地址线的双向地址奇偶信号。
  - A_{31}~ A_5有一地址奇偶位。在奇偶地址判定时，A_4和A_3未使用。
- **AHOLD：** 地址保持。
  - 悬浮地址总线，驱动处理器的询问周期；允许其它总线主控设备用地址询问周期来驱动处理器的地址总线。

179

2.8.2 Pentium处理器内部框图与信号功能

- **APCHK**：地址奇偶校验输出，低电平有效。
  - 当处理器在询问周期检测到A_{31}~ A_5上的奇偶错误时，该信号在EADS有效两个时钟后出现。
- **M/IO**：存储器与I/O选择。
  - M/IO＝1，表示访问存储器；M/IO＝0，表示访I/O。
- **W/R**：写与读信号 。
  - 高电平为写周期，低电平为读周期。
- **D/C**：数据与代码选择。
  - D＝1，输出表示存取数据；D/C＝0，输出表示代码。
- **D_0 ~ D_{63}**：64位数据信号。
  - D_0 ~ D_7和D_{56} ~ D_{63}分别表示最低和最高字节。

2.8.2 Pentium处理器内部框图与信号功能

- **$DP_0 \sim DP_7$：** 数据奇偶校验信号 。
  - 数据总线上的$D_0 \sim D_7$，…， $D_{56} \sim D_{63}$ 8个字节分别对应$DP_0 \sim DP_7$校验信号。
- **PCHK：** 数据奇偶校验信号。
  - 低电平表示数据读奇偶校验结果。只表示请求的有效的数据字节的奇偶状态。
- **PEN：** 奇偶允许信号，低电平有效。
  - 用以确定在读周期出现数据奇偶校验错误时是否产生机器检查异常。
- **NMI：** 不可屏蔽中断请求信号。
- **……**

注：其它引脚说明略。详细内容可参考教材或专业资料。

2.8 Pentium微处理器

1. Pentium处理器的特点
2. Pentium处理器内部框图与信号功能
3. 80486与Pentium总线之间的主要区别

2.8.3 80486与Pentium总线的主要区别

- Pentium处理器：64位数据总线，80486：32位数据总线。Pentium比80486有更多的字节允许位(BE_0 ~ BE_7)和数据奇偶位(DP_0 ~ DP_7)。
- Pentium处理器具有同时驱动两个总线周期的能力。它采样Cache有效输入信号KEN只要一次，而80486需采样KEN两次。在Pentium中由Cache信号和地址驱动突发长度信息，而80486用BLAST信号控制突发长度。
- Pentium处理器一个总线周期产生8字节的写操作，且不使用PLOCK信号。它不改变地址低位和突发时的字节允许。

183

2.8.3 80486与Pentium总线的主要区别

- Pentium处理器需要写回和在线填充运行突发周期。不支持非Cache周期和非突发Cache周期。
- Pentium处理器使用的引脚CACHE、HIT、HITM、INVT和WB/WT支持写回方式Cache。不支持80486以BS_8和BS_{16}管理的动态总线调整。
- Pentium处理器在连接的LOCKED周期和SCYC引脚之间提供一个空闲时钟表示在锁存操作时的分离周期。Pentium处理器非Cache代码预取是8字节，不是16字节。

184

2.8.3 80486与Pentium总线的主要区别

- Pentium处理器增加INIT引脚用于初始化；增加TDI、TDO、TMS、TRST和TCK信号实现边界扫描测试；增加APCHK、BUSCHK、PEN和AP信号增强数据奇偶校验、地址奇偶错误校验等。
- Pentium处理器有IU、IV和IBT信号和一个跳转跟踪信息周期，以执行跟踪。
- Pentium处理器设有SMI和SMIACK信号用于系统管理方式；设有BP_3、BP_2和BP_1/PM_0信号用于执行性能监视和外部断点；设有FRCMC和IERR引脚支持功能冗余检查，而80486没有这些引脚。

185

本章要点

- 8086微处理器的结构。
  - 8086微处理器的结构特点。
  - 8086微处理器的寄存器结构。
- 8086微处理器的引脚功能和相关知识。
  - 8086总线分时复用的特点。
  - 8086系统中的存储器分段与物理地址的形成。
  - 8086两种工作方式——最小方式与最大方式的区别。
- 8086微处理器的总线时序。
  - 几种周期概念的区别及联系。
  - 8086在最小模式下的读总线周期时序图。

186