第3章 32位Pentium微处理器

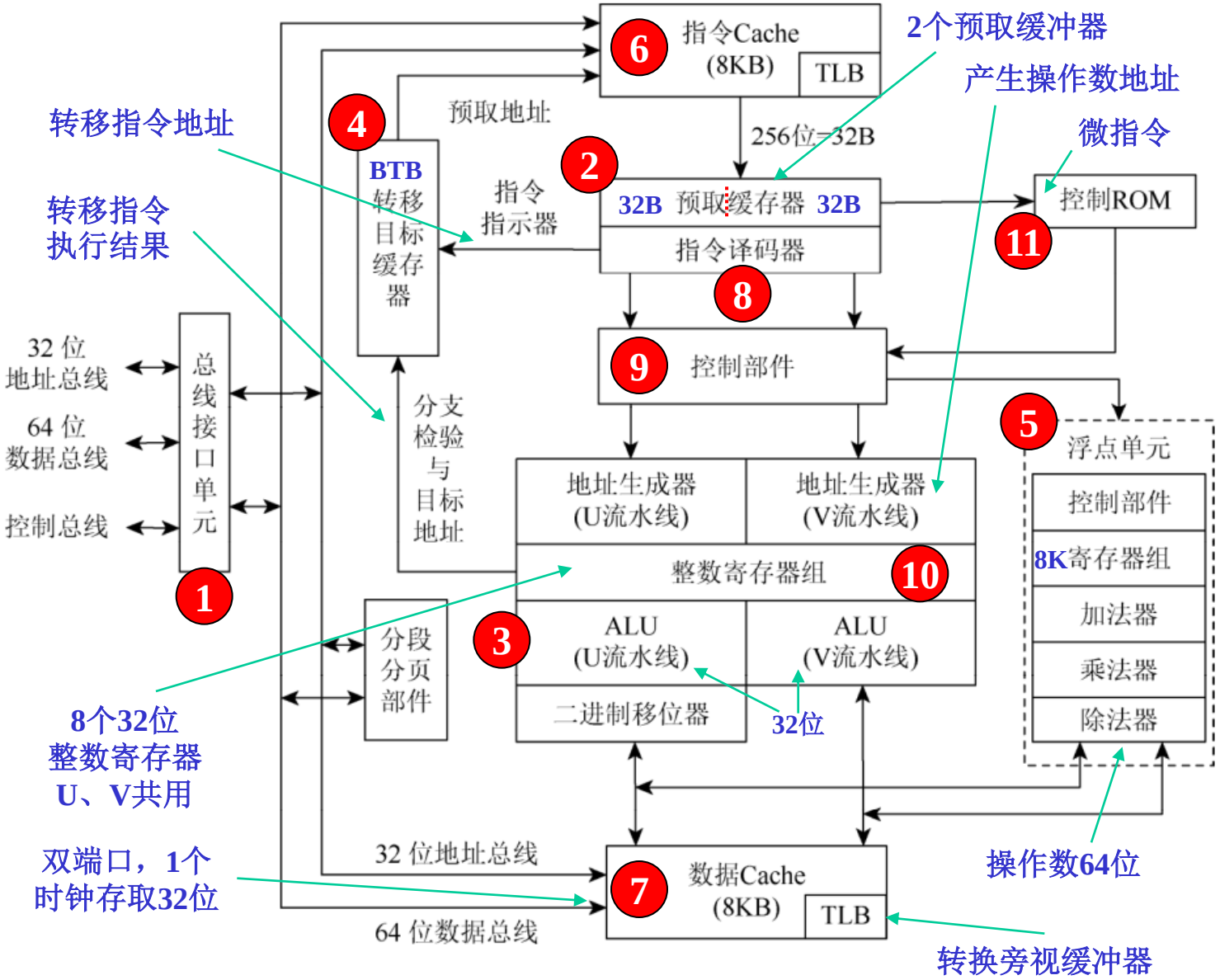


图 3.1.1 Pentium 微处理器的内部结构

结构特性：

1. 超标量流水线，超标量为2
2. 分立Cache，指令数据各8K
3. 重新设计的浮点运算部件，8级流水
4. 动态转移预测
5. 内部32位，外部DB64位，属于32位机
6. 4种工作方式：实地址、保护虚地址、虚拟8086、系统管理

1. 总线接口部件
2. 预取缓冲存储器
3. 整数运算部件
4. 转移目标缓冲器
5. 浮点运算部件
6. 代码Cache
7. 数据Cache
8. 指令译码部件
9. 控制部件
10. 整数及浮点数寄存器组
11. 控制ROM部件

3.1.1 总线接口单元

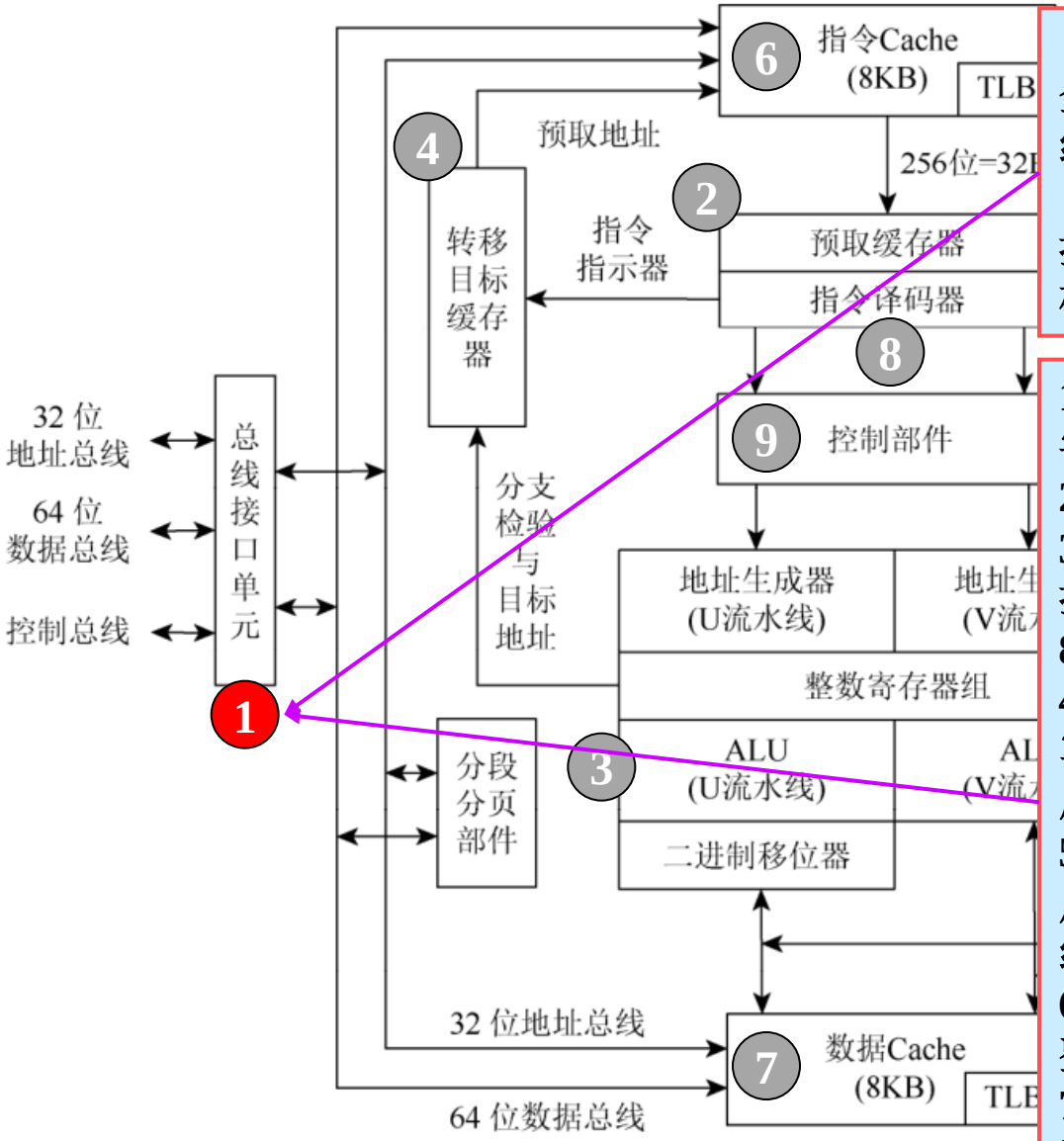


图 3.1.1 Pentium 微处理器的内部结构

总线接口部件包括三总线的控制：全部总线控制信号、独立的32位地址总线、独立的64位数据总线。

总线接口部件根据优先级高低协调数据的传送、指令的预取等操作，在处理机的内部部件和外部系统间提供控制。

1. 地址收发器和驱动器：驱动A31～A3、字节允许信号BE7～BE0。
2. 数据总线收发器：控制D63～D60。
3. 总线宽度控制：用2个输入端控制数据总线的宽度，4种：64位、32位、16位、8位。
4. 写缓冲：配备一个暂时存储器，存放写到主存中的4个32位的数据，起缓冲作用。
5. 总线周期和总线控制：支持多种总线周期控制，如成组传送、单个传送、总线仲裁、中断等。
6. 奇偶校验的生成和控制：实现数据完整性检测和错误检测。
7. Cache控制：实现对Cache操作的控制和一致性检查。

3.1.2 流水线操作

- 流水线设计是将一个顺序的处理过程分解为若干个子过程，每个子处理过程称为一个段，完成一个具体的功能，并产生一个中间结果。
- 指令流水线机制：在冯.诺伊曼计算机体系结构中，一条指令的执行被分为几个步骤，每个步骤由流水线的相应段来完成，就构成了一条指令的流水线。

1）取指令：从Cache或主存取指令

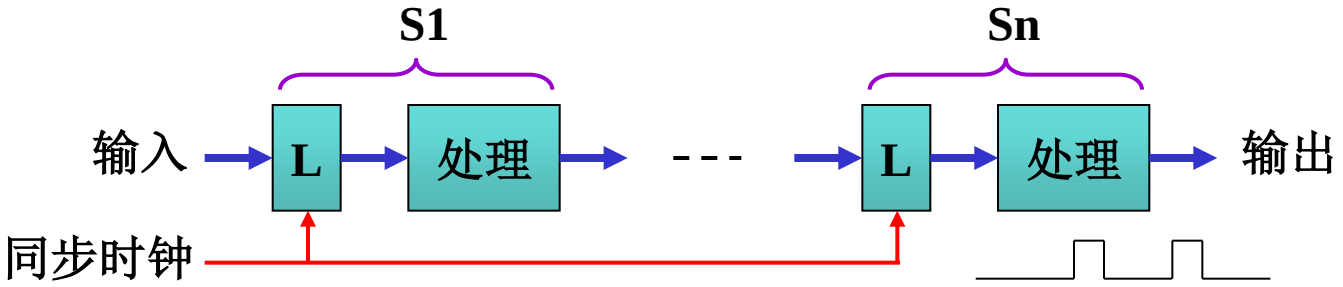
2）译码：确定操作类型

3）生成操作数地址：若需要存储器操作数，则确定操作数地址，由Cache或主存取数

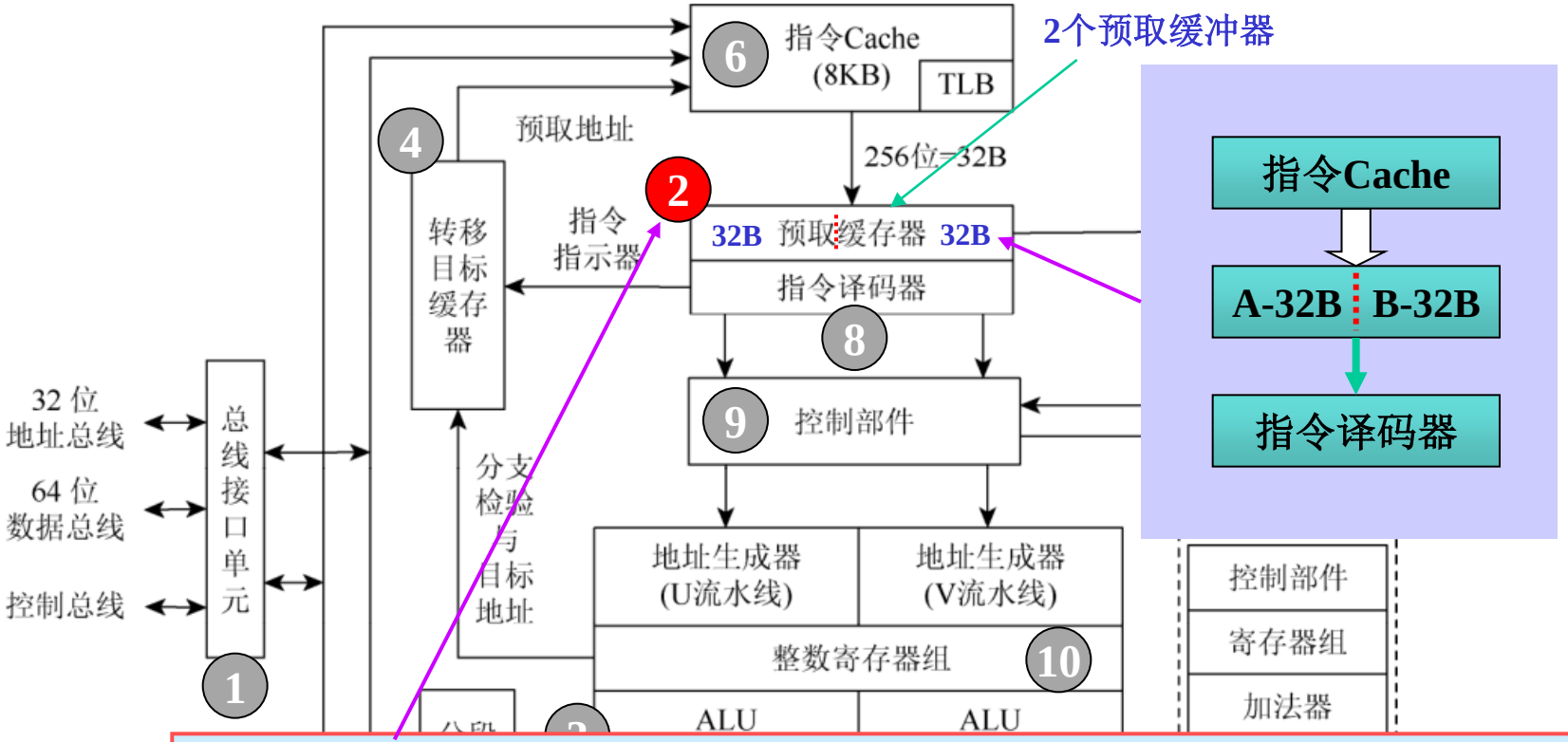
4）完成操作

5）写回：将结果写回寄存器或主存

同步流水线基本结构



指令预取



预取缓冲器作用：预取指令。

预取缓冲器结构：2个部分，各32个字节，但只能有一个是工作状态，顺序取和转移取交替，由路径指针转换。

预取时机：总线部件不执行总线周期时，就执行一个取指令周期。

预取缓冲器的优先权最低，预取操作不允许影响当前指令的执行。

图 3.1.1　Pentium 微处理器的内部结构

超标量技术

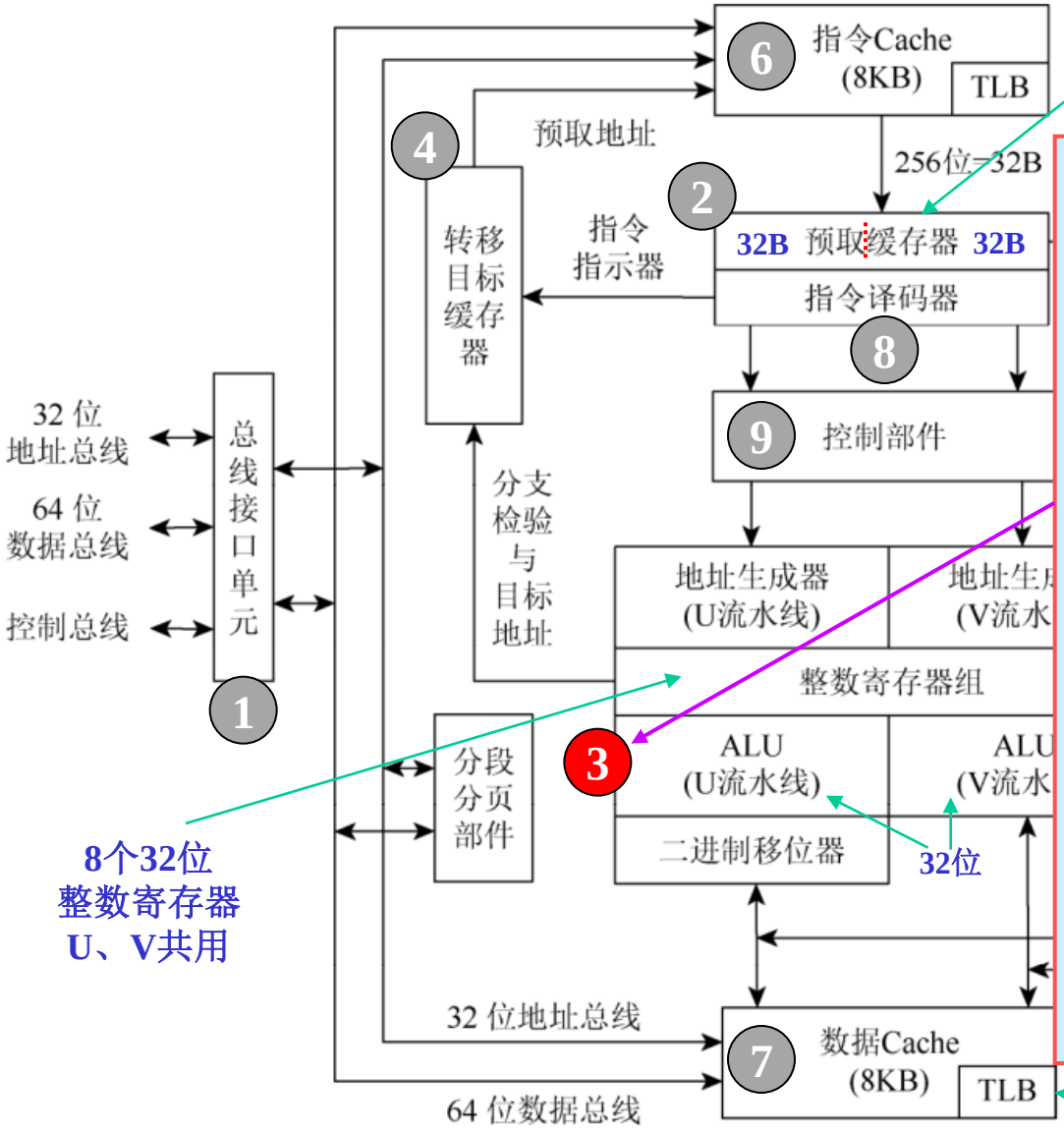


图 3.1.1 Pentium 微处理器的内部结构

1）超标量：CPU中有多条指令流水线。

2）Pentium中有U、V两条流水线，超标量为2。

3）超标量目的：改善标量指令的执行性能，与向量处理器相对比，由于多数应用都是标量运算，所以超标量技术代表了高性能处理器的发展方向。

4）改善指令流水线的不同途径

（1）超流水线

（2）超标量流水线

超标量执行

PF：Prefetch Stage，指令预取，由指令Cache取指令，指令长度可变，存入一个预取缓冲器，2个32B部件。

D1：Decode Stage 1，指令译码，确认操作数和寻址方式，完成指令配对检查。结合BTB进行转移指令搜索，确定下次指令预取地址。

D2：Decode Stage 2，操作数地址生成，计算存储器操作数地址。

EX：Execution Stage，指令执行，运行ALU操作，完成数据Cache访问，验证非条件转移指令分支预测的正确性

WB：Writeback Stage，写回，用计算结果修改目标，验证条件转移指令分支预测的正确性。

每条Pentium流水线由5段组成

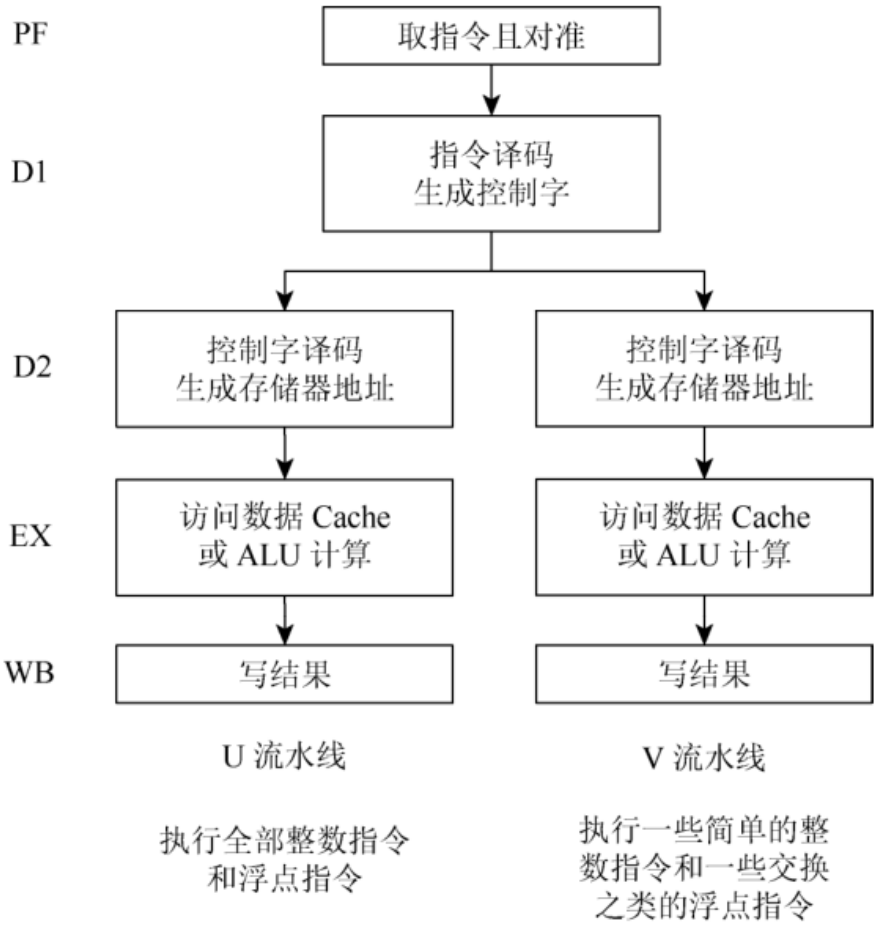


图 3.1.2　流水线与超标量执行

整数流水线

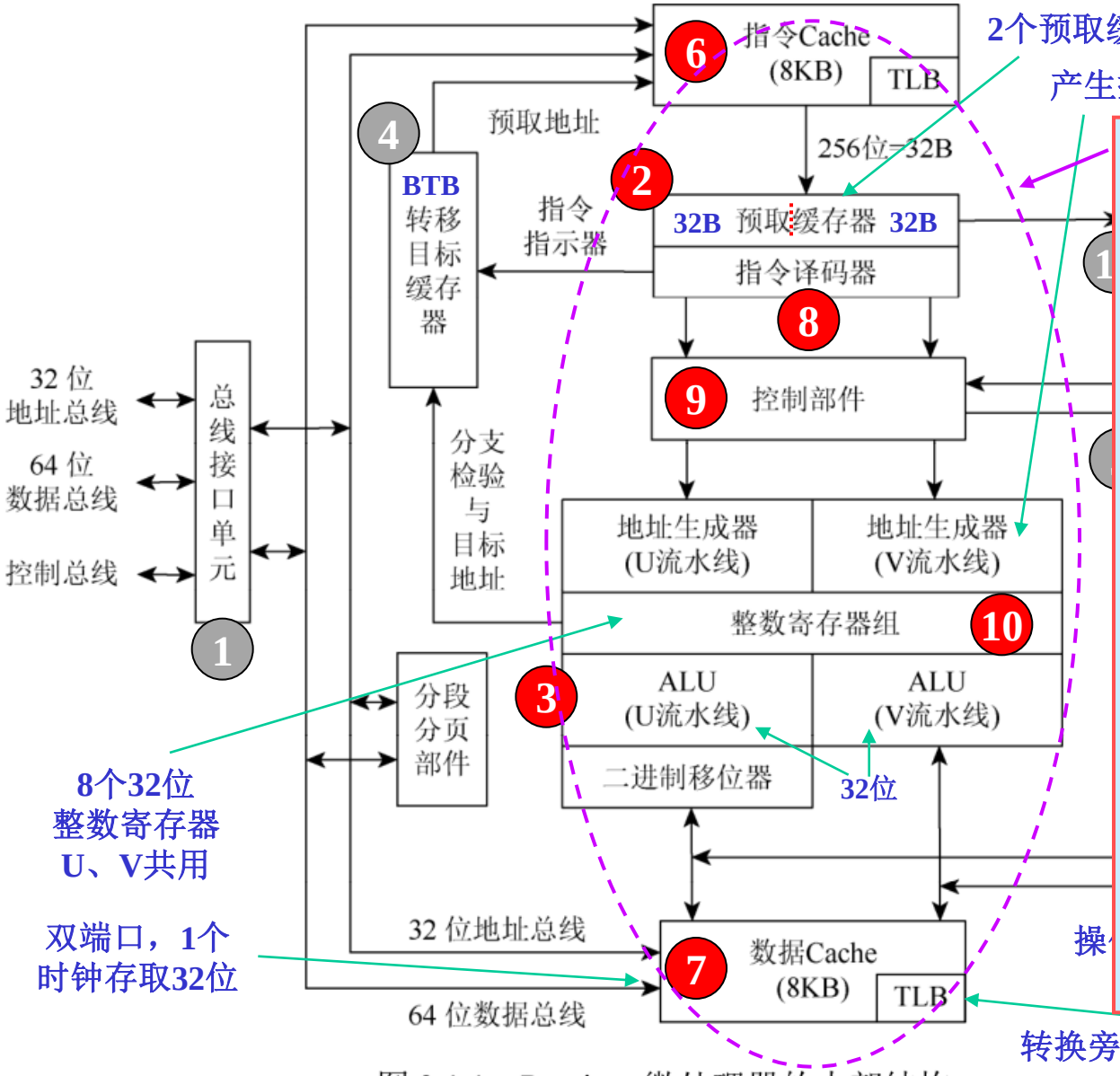


图 3.1.1 Pentium 微处理器的内部结构

1）Pentium中有U、V 2条整数流水线，5级；1条浮点流水线，8级。

U：执行全部整数指令和浮点指令

V：执行简单整数指令和交换类浮点指令

2）U、V 并行执行

3）2个指令预取缓冲器，各32B，指令译码器还负责指令配对检查

4）每条流水线有自己的ALU、地址生成电路、Cache接口

Pentium流水线操作与结构对应关系

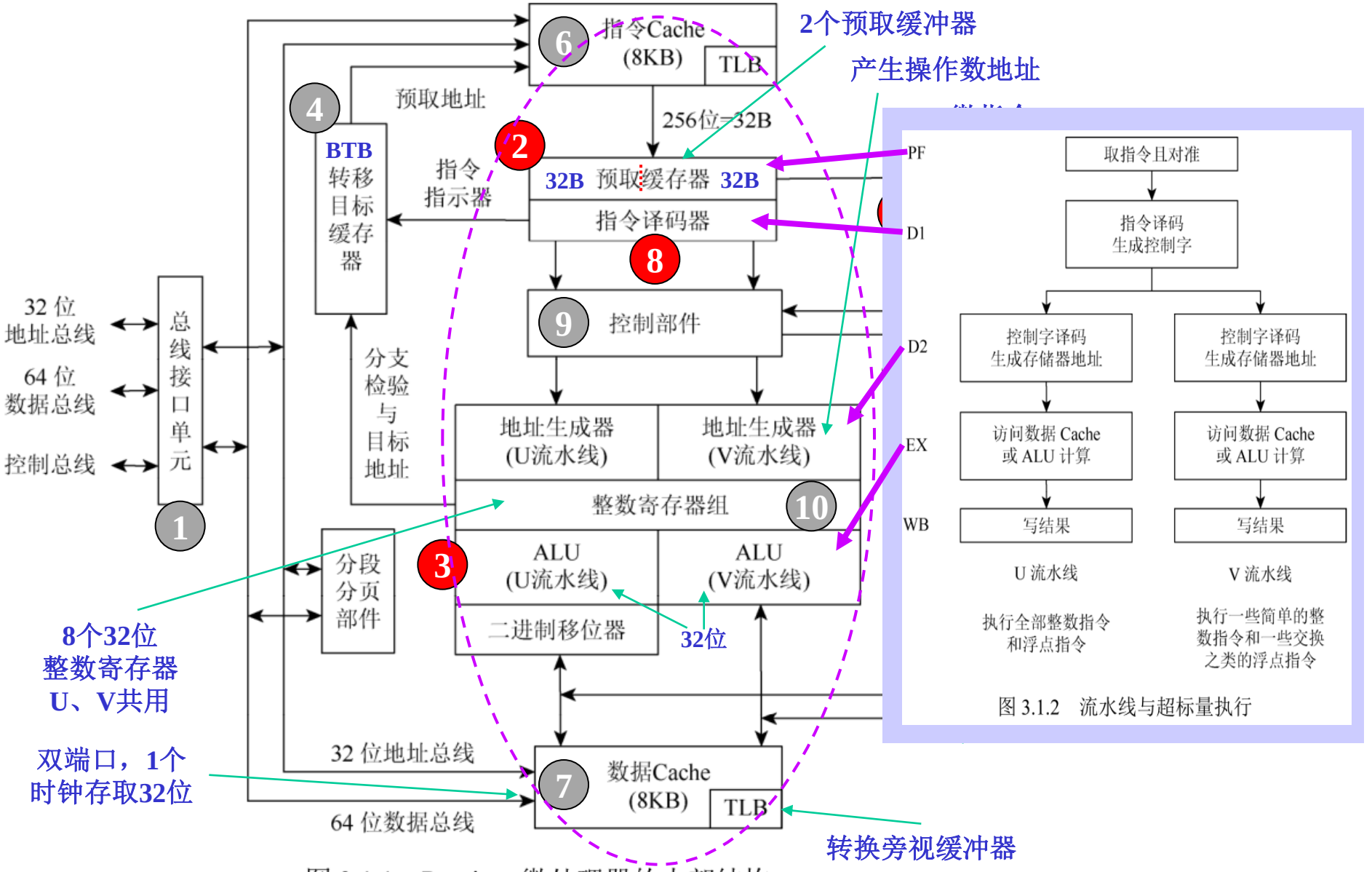


图 3.1.2　流水线与超标量执行

图 3.1.1　Pentium 微处理器的内部结构

转移（分支）预测技术

- 分支预测技术是流水线的突出问题。
- 分支预测：遇到分支指令时（如JMP、CALL、中断等指令），指令预取单元能比较准确地判定是否转移取指。
- 8086的预取指令队列只能顺序取，不能预测。

静态预测：依据转移指令类型来预测处理。有的转移指令总预测转移，有的转移指令则总预测不转移。比如，向前转移指令预测转移，向后转移指令预测不转移，方法简单，准确率不高，辅助方法。

动态预测：根据一条转移指令过去的行为（历史状态）预测将来的去向。算法复杂，准确率较高。Pentium采用动态分支预测，准确率85%。

- 动态预测实现的依据是统计规律：

1）分支指令转向机会不等（条件转移指令，转移为60%）

2）多数分支发生在循环程序段

分支预测

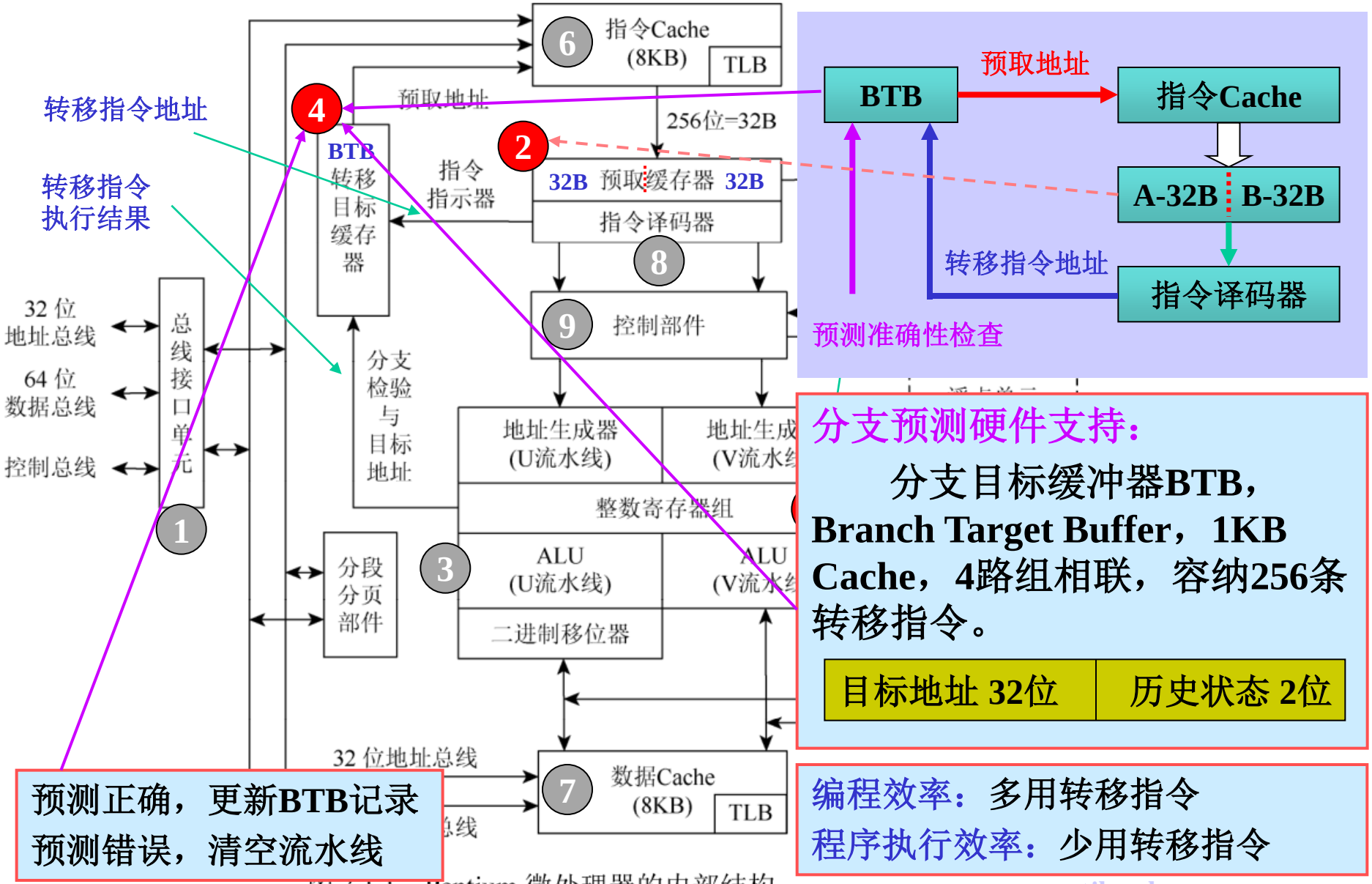


图 3.1.1 Pentium 微处理器的内部结构

分支预测硬件支持：

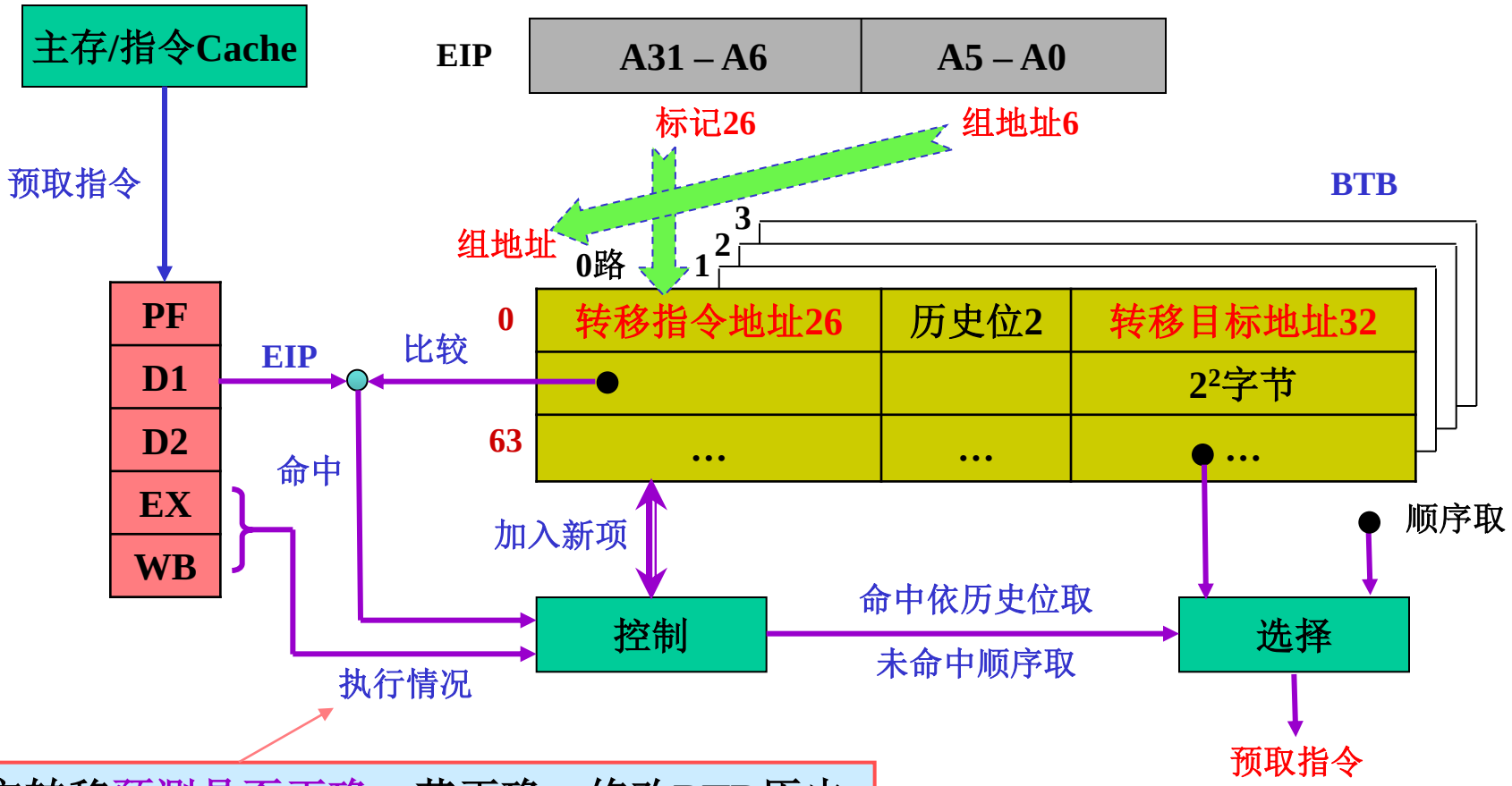
分支目标缓冲器BTB，Branch Target Buffer，1KB Cache，4路组相联，容纳256条转移指令。

目标地址 32位	历史状态 2位

预测正确，更新BTB记录
预测错误，清空流水线

编程效率：多用转移指令
程序执行效率：少用转移指令

流水线、BTB结构与转移预测



判定转移预测是否正确：若正确，修改BTB历史位；不正确，清空流水线，取正确路径。
EX：非条件转移检验，　WB：条件转移检验

BTB：　4路组相联Cache，1KB
数据容量= $2^6 \times 2^2 \times 2^2 = 2^{10}$

组×字节（转移目标地址）×路

指令译码：指令配对规则

- 1）配对的2条指令必须是“简单”指令：硬件化的，不需要任何微代码控制，在1个时钟周期内执行。

  MOV REG，REG / MEM / IMM

  MOV MEM，REG / IMM

  ALU REG，REG / MEM / IMM

  ALU MEM，REG / IMM

  INC REG / MEM

  DEC REG / MEM

  PUSH REG / MEM

  POP REG

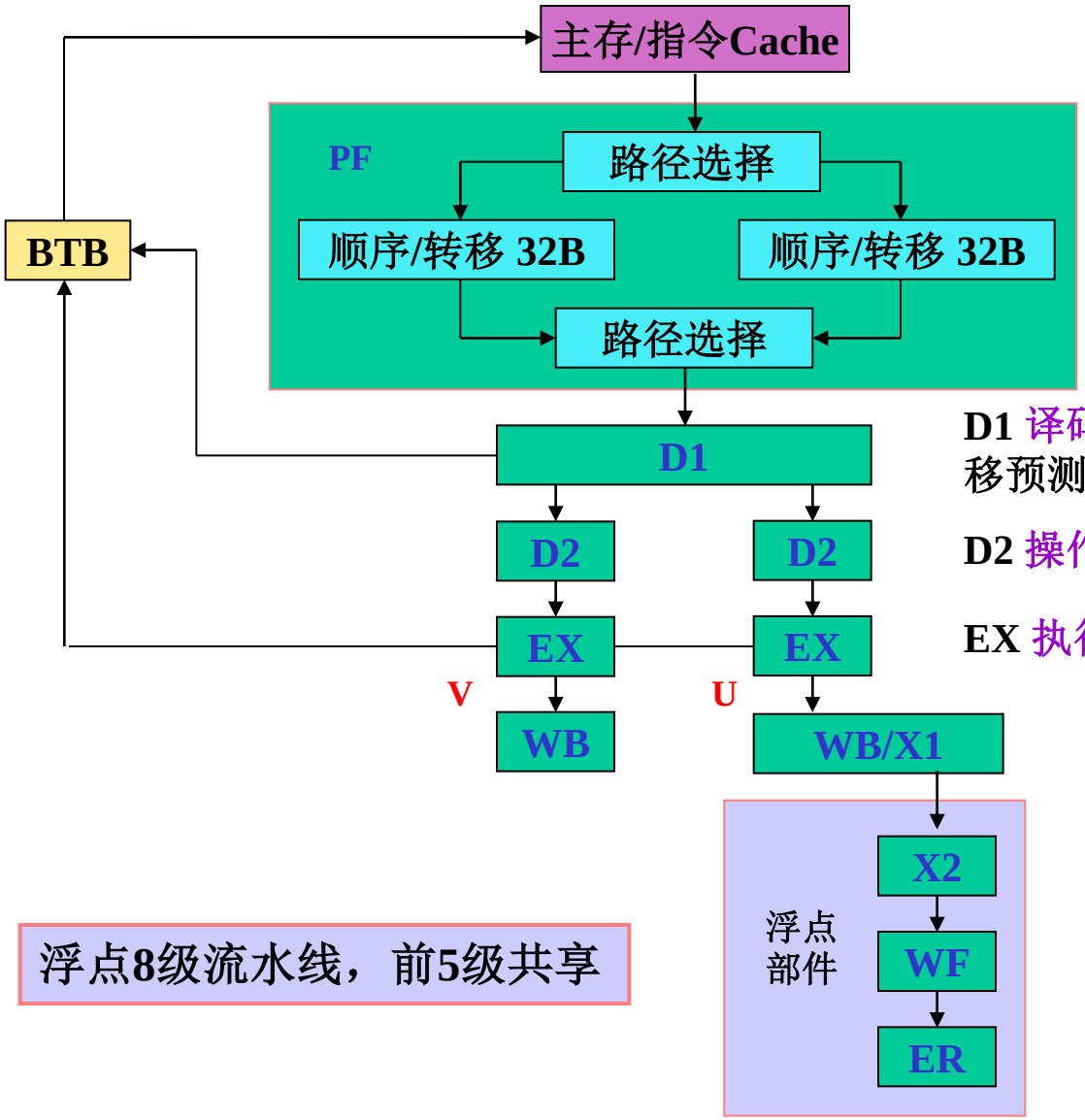
  LEA REG，MEM

  近转移的JMP / CALL / JCC

  NOP

- 2）2条指令之间不得存在“写后读”或“写后写”这样的寄存器相关性。
- 3）1条指令不能同时既包含立即数又包含位移量。
- 4）带前缀的指令只能出现在U流水线中。
- 此外，条件分支转移和无条件分支转移指令只有当作为配对中的第二条指令时才可以配对。它们不可以与下一条顺序指令进行配对。

浮点流水线



PF 预取指令，FIFO结构，2选1，一次取32B

D1 译码，指令配对，2条指令并行译码，转移预测

D2 操作数地址计算

EX 执行，ALU操作，Cache存取，预测验证

WB/X1 保存结果/一次执行，将数据转成实型格式，存入寄存器堆

X2 二次执行，实际执行级

WF 写回，写浮点数，结果写回80位浮点寄存器堆

ER 错误报告，处理可能出现的错误

浮点8级流水线，前5级共享

3.1.3 Cache

- 1. Cache：弥补主存速度，在CPU与主存之间设置的高速、小容量的存储器，构成Cache-主存存储层次，速度是Cache的，容量是主存的。
- 2. 解决CPU与主存速度的问题，3种方法：

  （1）在CPU中增设寄存器。

  （2）采用多体交叉并行存储器。

  （3）采用Cache存储器。
- 主存速度不够→引出Cache，主存容量不够→引出虚拟存储器。
- 3. 程序具有局部性（各存储层次构成的主要依据）

  （1）时间上的局部性：最近的未来要用到的信息可能是现在正在使用的信息，因为程序存在循环。

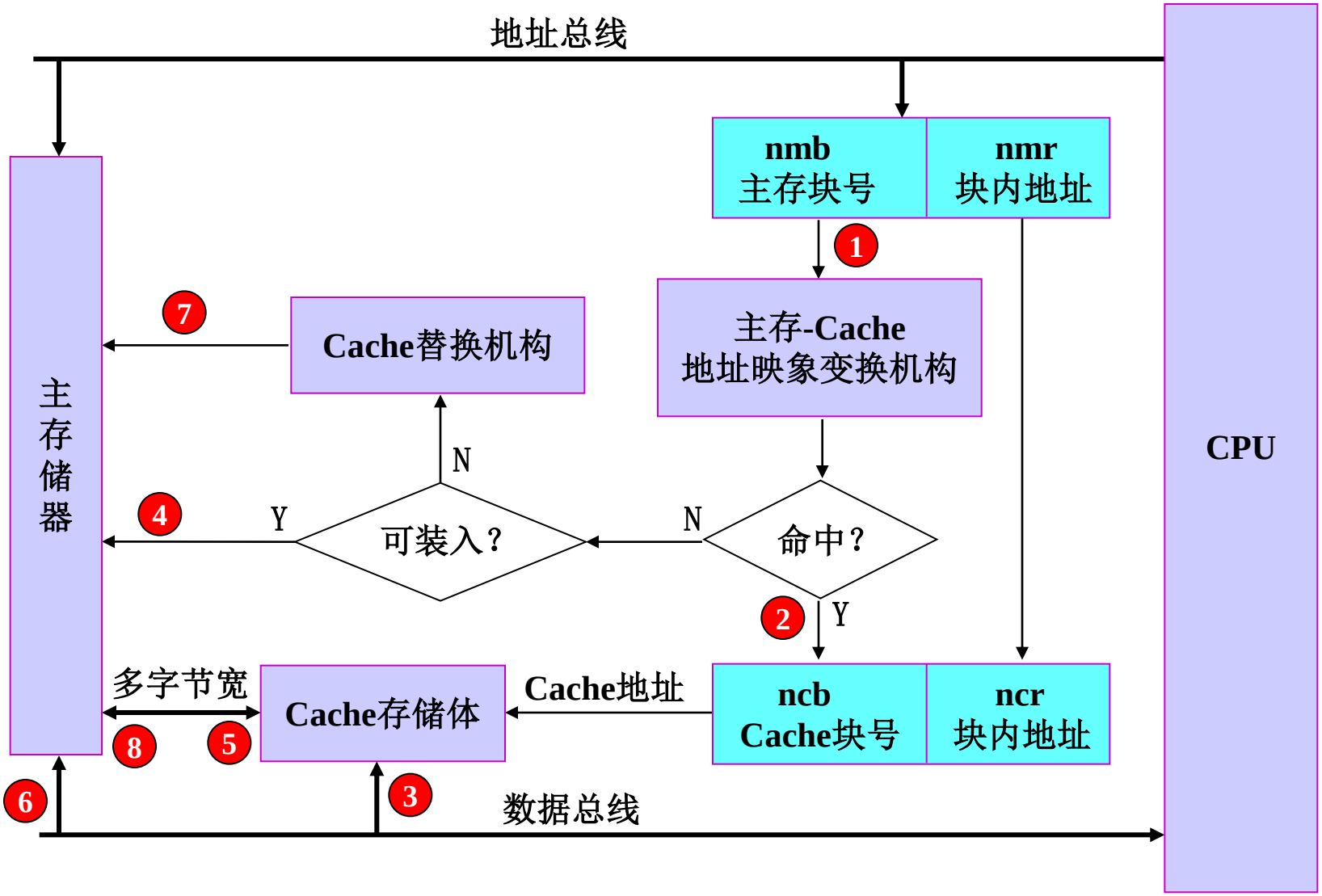
  （2）空间上的局部性：最近的未来要用到的信息可能与现在使用的信息在程序空间上是邻近的，因为指令按顺序存放。
- 4. 命中率：

  H=命中次数/总访问次数

Cache 及主存地址的组成

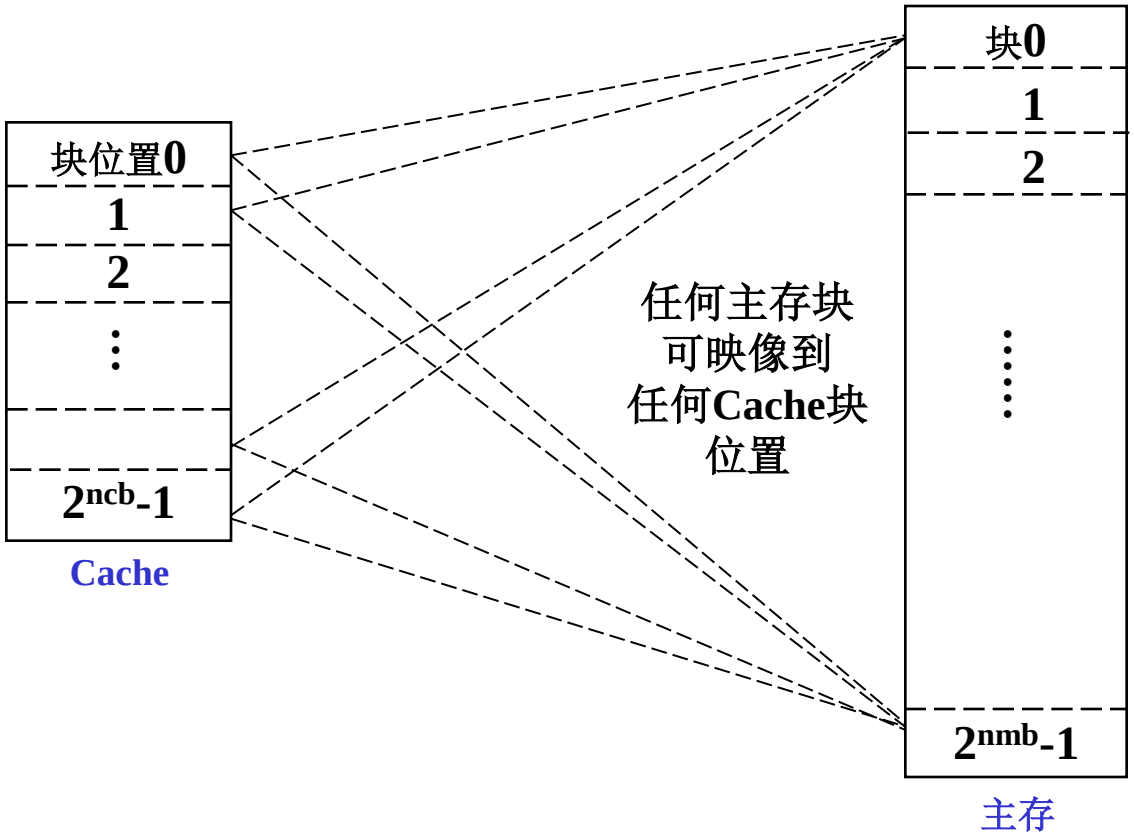
- Cache和主存等分成容量相同的块，每块由4、16、32等字节组成，并且把块有序地编号。
- Cache基本结构：Cache存储体、地址映象变换机构、替换机构。
- Cache存储体：Cache存储信息的主体。
- 地址映像：将每个主存块按什么规则装入Cache中。
- 地址变换：将主存地址变换成Cache地址。
- 块冲突：主存块要进入Cache中的位置已被其他主存块占用，要用替换算法。
- 替换机构：发生块冲突时，根据替换算法完成Cache块替换。
- Cache操作由硬件完成，对用户透明。
- 在进行地址变换、替换时，都是以块为单位进行调度。

图 3.1.4 Cache 的基本结构及工作过程



地址映像 - 全相联映像

- 地址映像：主存中任意一块都可映像装入到Cache中任意一块位置。
- 优点：灵活，块冲突率低，只有在Cache中的块全部装满后才会出现冲突，Cache利用率高。



全相联映像地址变换

- 地址变换：硬件实现。目录表，相联比较。
- 缺点：地址变换机构复杂，成本高。

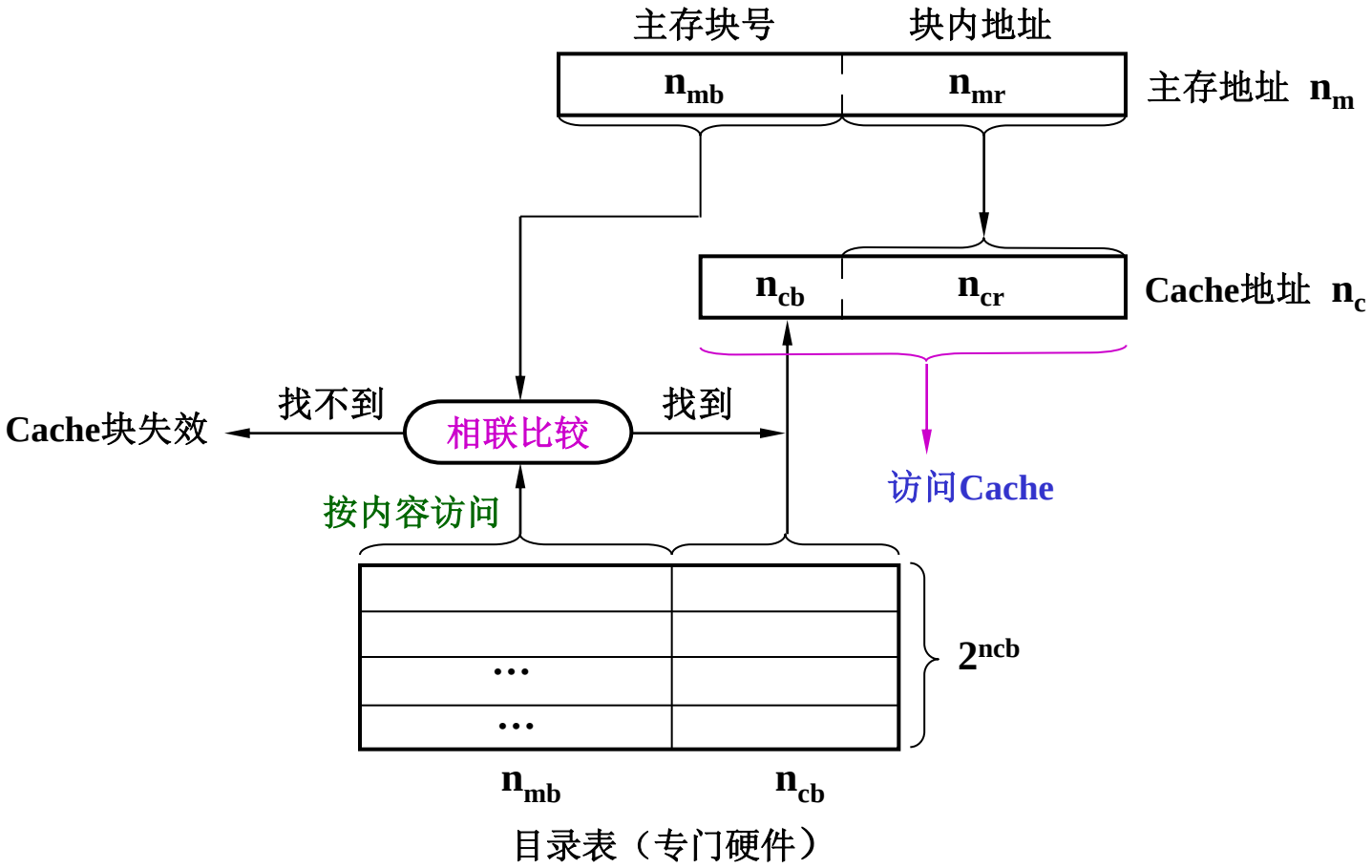
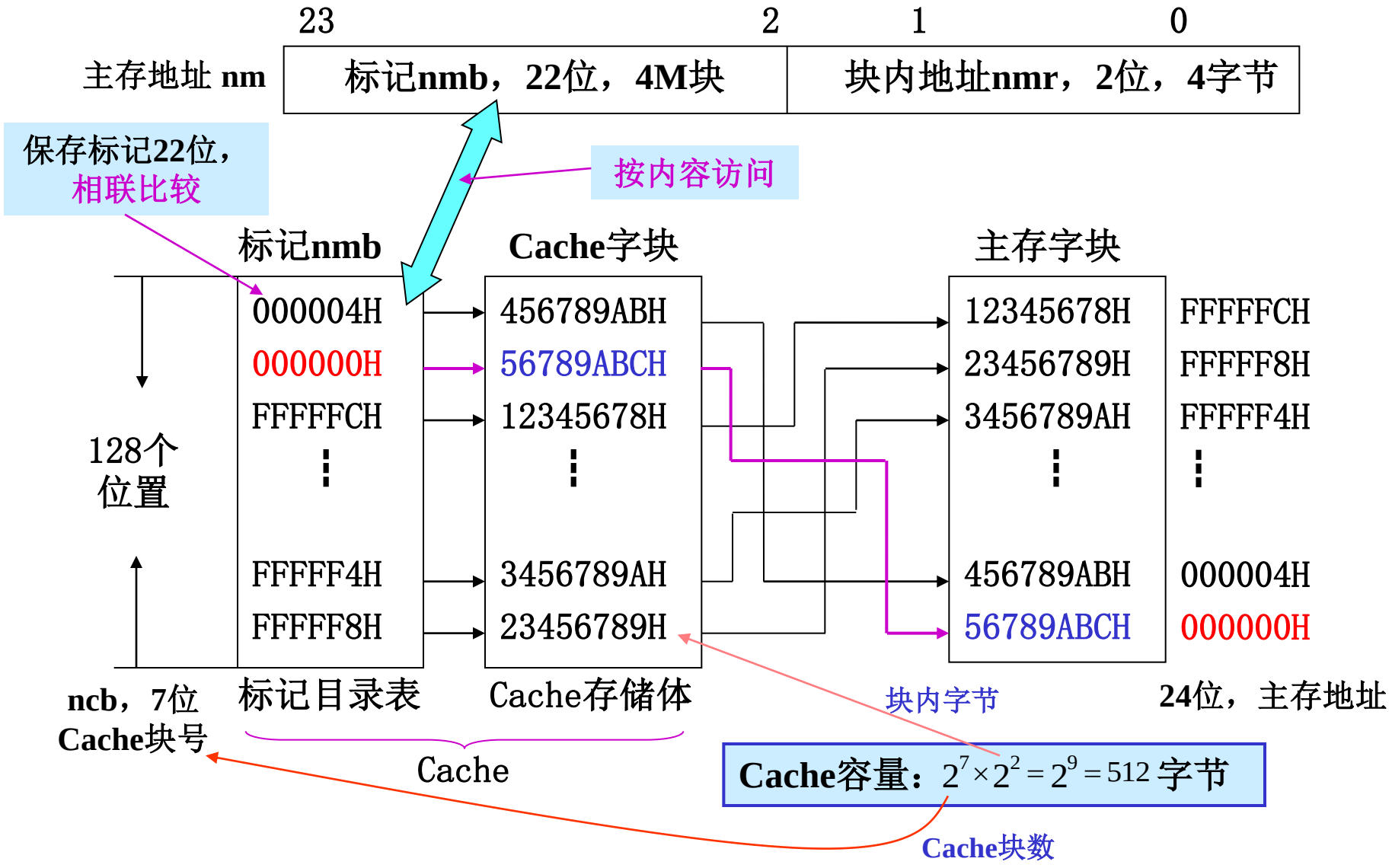
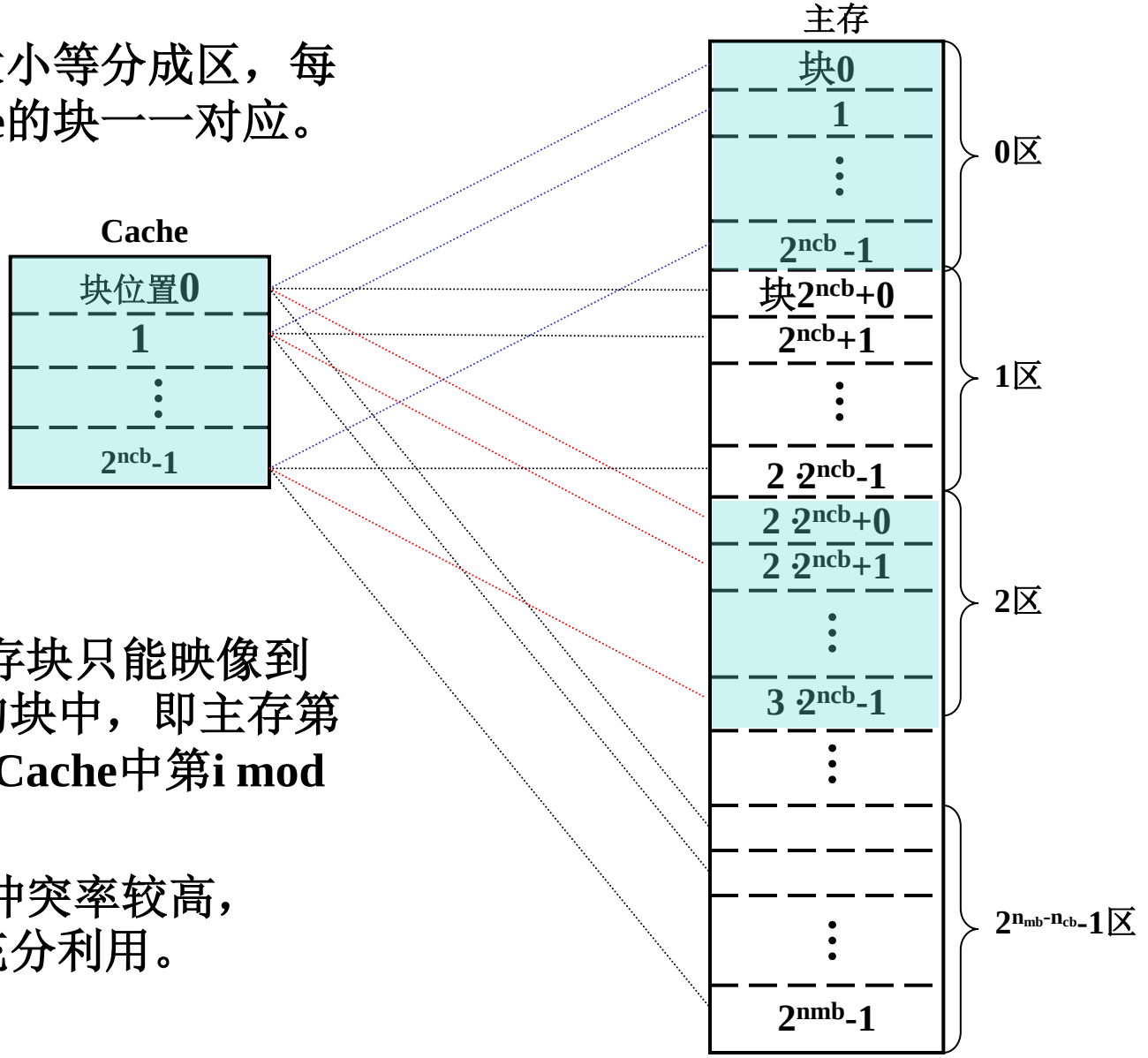


图 3.1.5 全相联映像示意图



地址映像 - 直接映像

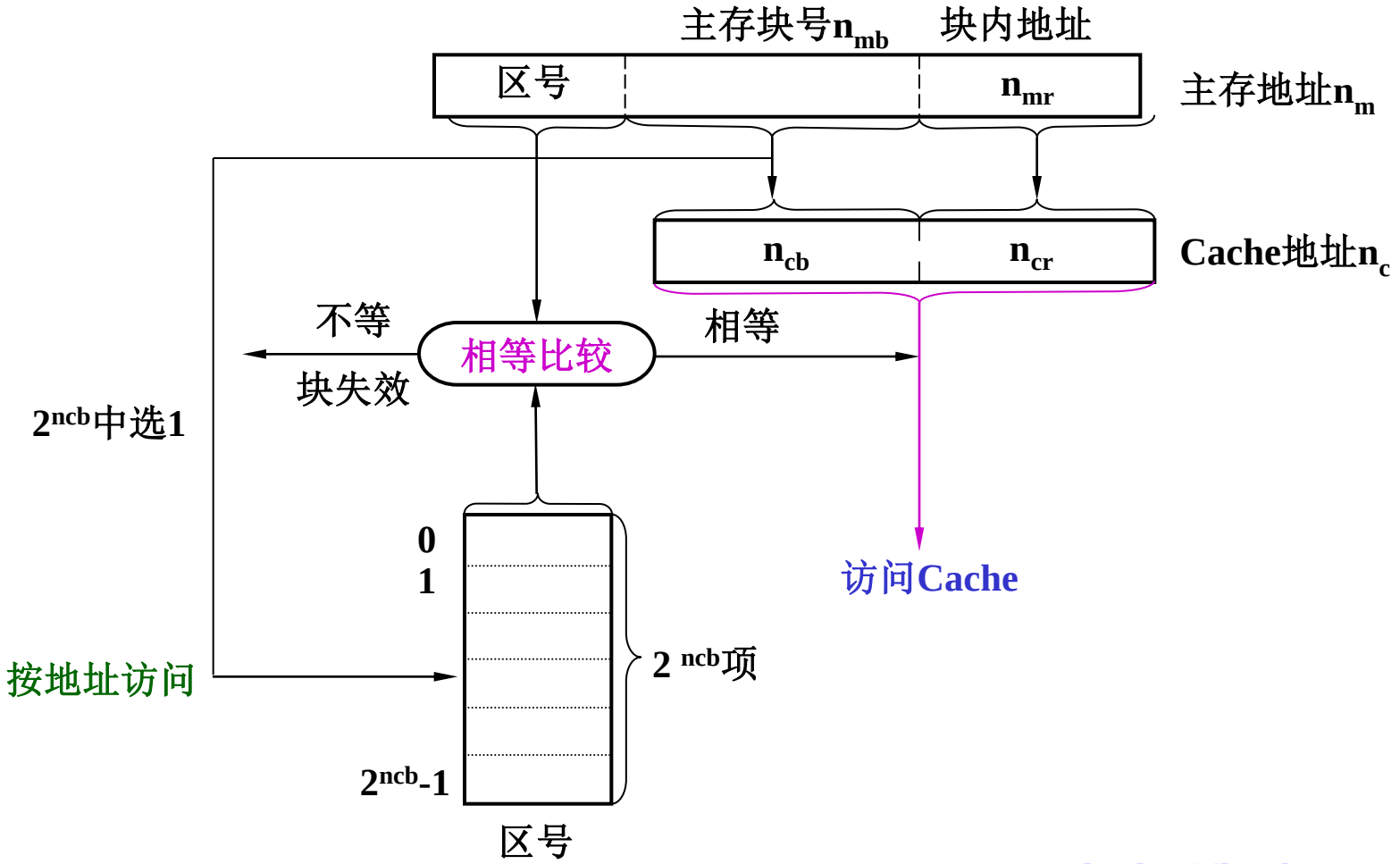
- 主存空间按Cache大小等分成区，每区内的各块与Cache的块一一对应。



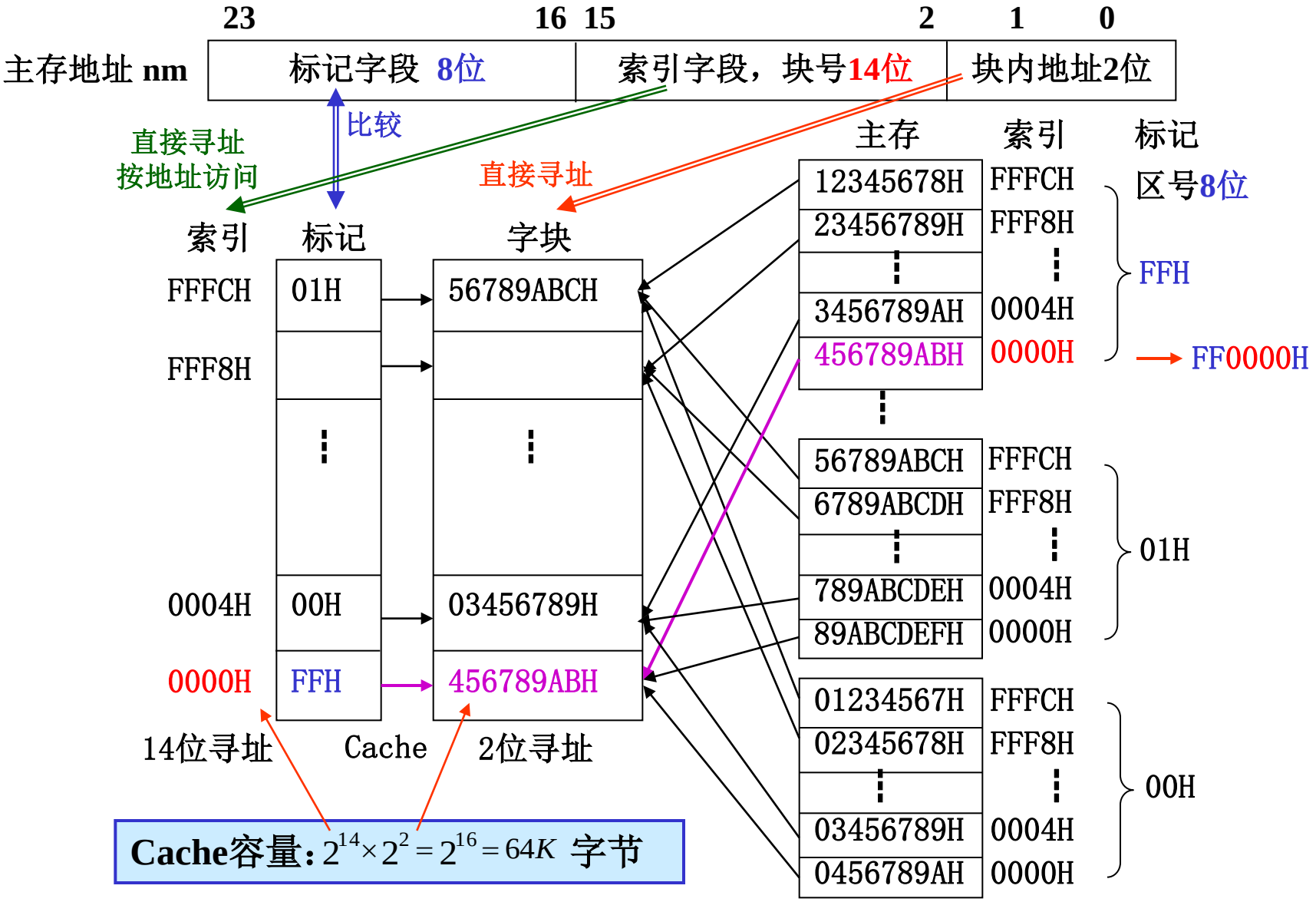
- 地址映像：一个主存块只能映像到Cache中唯一指定的块中，即主存第i块只能唯一映像到Cache中第i mod 2^{ncb}块位置上。
- 缺点：不灵活，块冲突率较高，Cache空间得不到充分利用。

直接映像地址变换

- 地址变换：硬件实现。主存地址中直接产生Cache地址，标志表中比较区号，标志表存储器按地址访问。
- 优点：地址变换简单、速度快，节省硬件。



直接映像示意图


23
16 15
2 1 0
主存地址 nm
标记字段 8位
索引字段，块号14位
块内地址2位
比较
直接寻址
按地址访问
直接寻址
索引
标记
字块
FFFCH
FFF8H
0004H
0000H
01H
00H
FFH
56789ABCH
03456789H
456789ABH
14位寻址
Cache
2位寻址
Cache容量：$2^{14}\times 2^{2}=2^{16}=64K$ 字节
主存
索引
标记
12345678H
23456789H
3456789AH
456789ABH
FFFCH
FFF8H
0004H
0000H
区号8位
FFH
FF0000H
56789ABCH
6789ABCDH
789ABCDEH
89ABCDEFH
FFFCH
FFF8H
0004H
0000H
01H
01234567H
02345678H
03456789H
0456789AH
FFFCH
FFF8H
0004H
0000H
00H

地址映像 - N路组相联映像

地址映像：路内直接映像，路间全相联映像

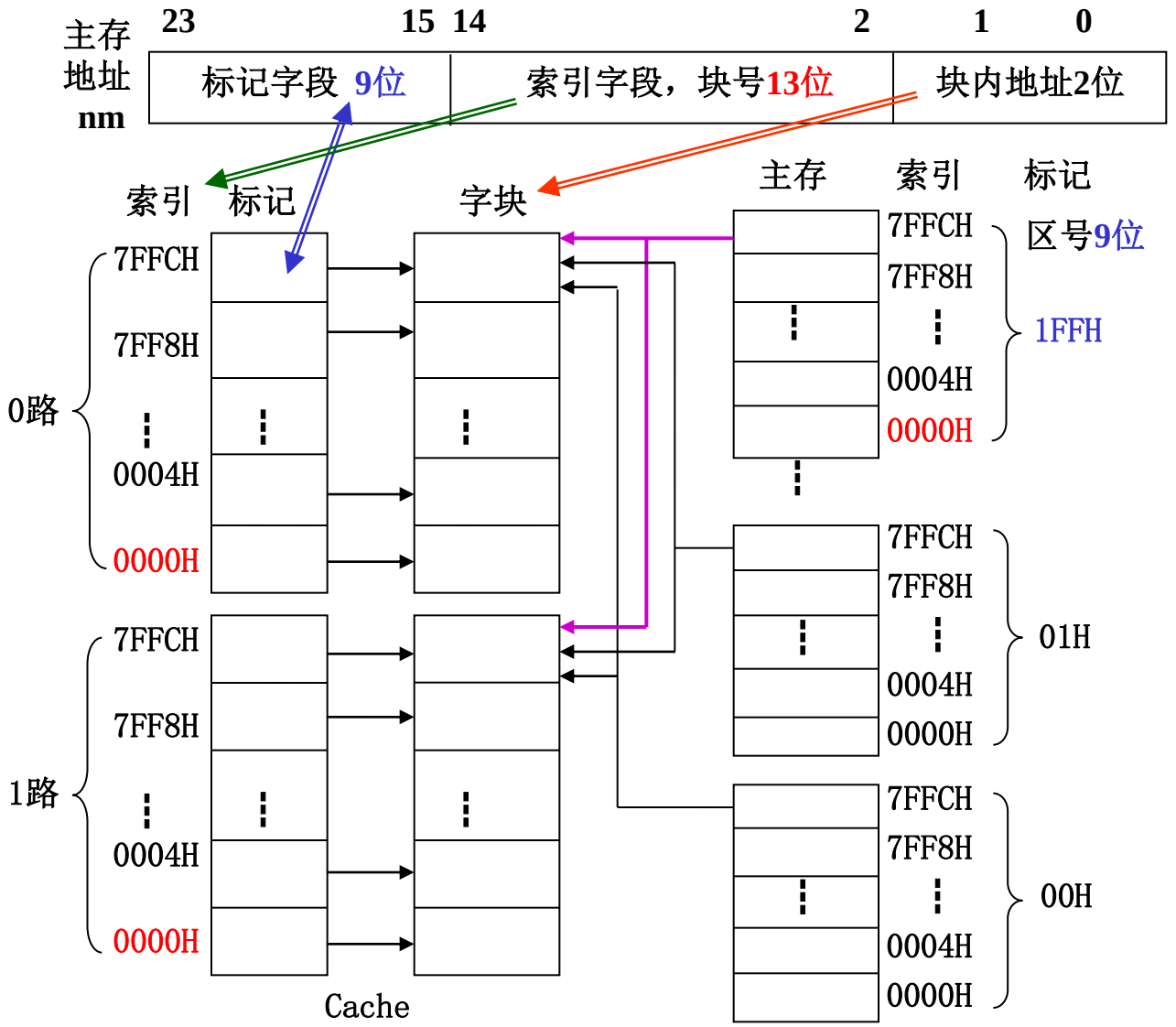
地址变换：主存地址中直接产生Cache地址（2路对应地址相同），同时比较各路对应块标记

特点：是直接映像与全相联映像的折中方案，若路数为1，则为直接映像，若路内块为1，则为全相联映像

每一路容量：

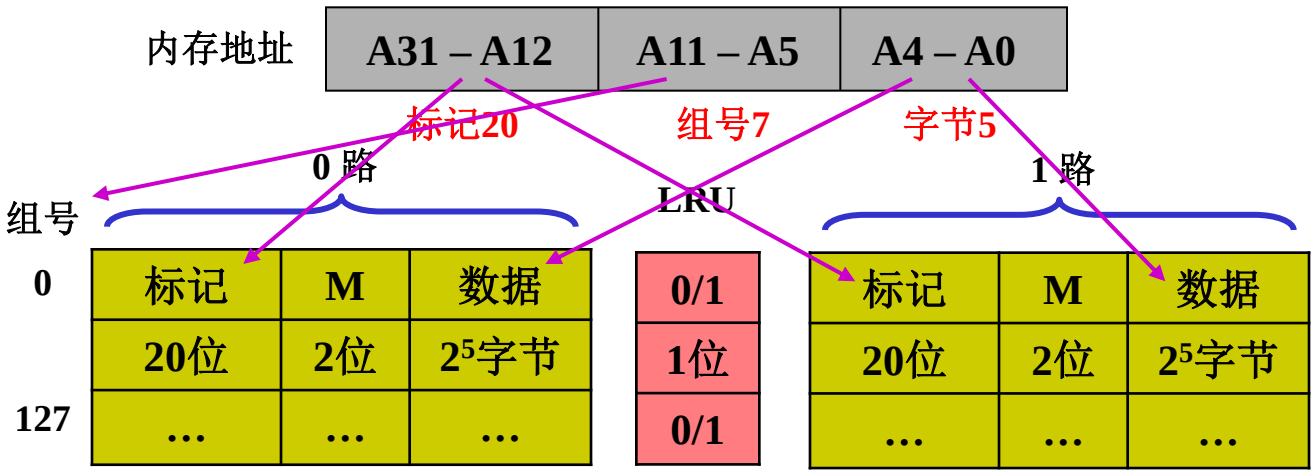
$$2^{13}\times 2^{2}=2^{15}=32K$$

2路容量为64K字节



Pentium片内Cache的结构

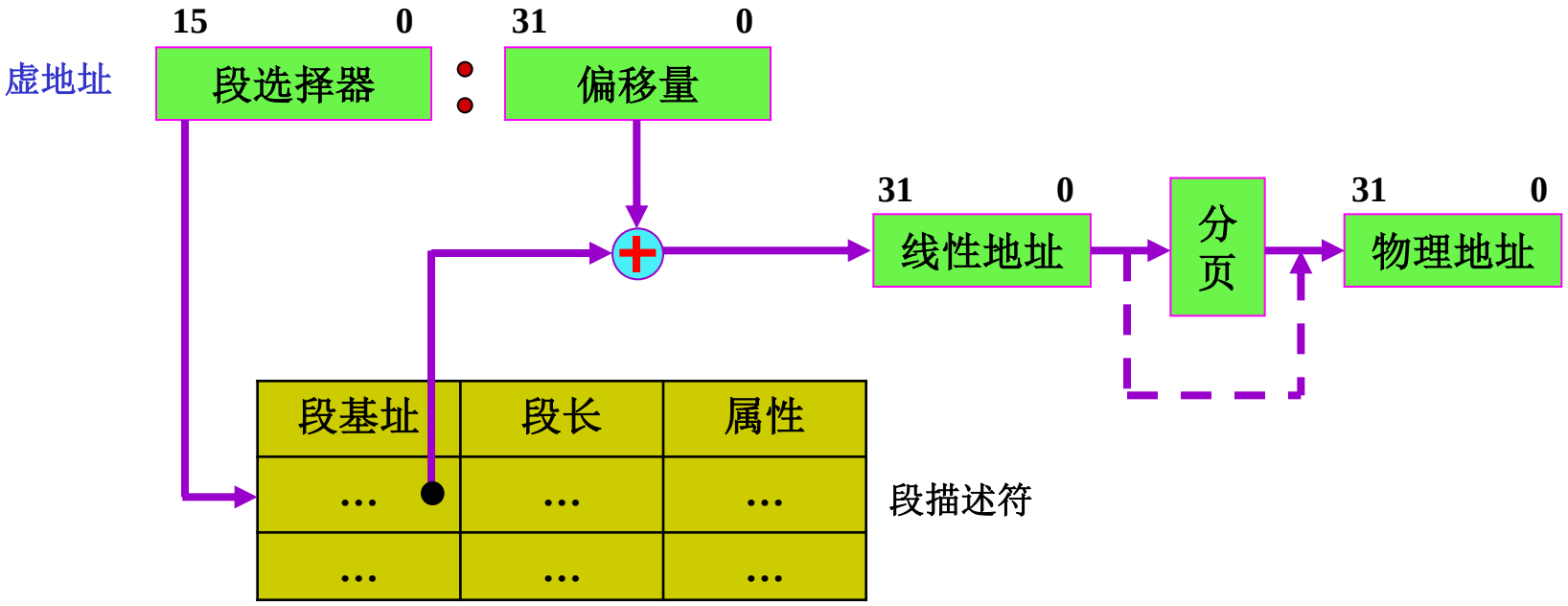
1）片内Cache，分立，指令8K，数据8K。指令跟踪结果：取指63%，取数25%，写数12%。
2）指令Cache：只读，单端口，256位（32字节），向PF提供指令代码。
3）数据Cache：读写，双端口，每端口32位（4字节），向U、V提供数据，支持二元访问，出现冲突时，U优先，可组成64位端口与浮点部件连接。
4）TLB：旁视缓冲器，用于在保护方式下快速将线性地址转换为物理地址。
5）数据Cache结构：8KB，2路组相联，128组，每组2行（即2路），每行32字节。
6）LRU：近期最少使用替换算法。例如规定，0路拷贝新数据为1， 1路拷贝新数据为0，则替换时，LRU=0，换0路，LRU=1，换1路。
7）MESI协议：Cache一致性协议，表明Cache与主存的一致性，M修改，E专有，S共享，I无效



数据Cache容量
= 组 × 字节 × 路
= $2^7 \times 2^5 \times 2^1 = 2^{13} = 8K$

3.1.4 分段与分页部件

1）段：被保护的独立的存储地址空间。
2）分段目的：在各应用程序间实行强制性的隔离，以便调用恢复。
3）分段部件功能：将逻辑地址转换成线性地址。
4）Pentium处理器采用二级分页管理机制，配有转换旁视缓冲器TLB。Pentium微处理器将4 KB定义为一页。
5）分页部件功能：将线性地址转换成物理地址。



3.2 Pentium 微处理器编程结构

- **Pentium微处理器的寄存器组包括5部分：**

基本结构寄存器：16个，其中通用寄存器8、专用寄存器2、段寄存器6

系统级寄存器：9个，其中系统4、控制5

调试寄存器：8个

模型专用寄存器：20个

浮点寄存器：18个

3.2.1 基本结构寄存器

E：扩展，Extended

标志寄存器EFLAGS

VM：虚拟8086方式，保护方式下，VM=1，CPU工作在虚拟8086模式。

AC：对齐检查，AC=1，当访问地址出现对齐错误时，产生异常中断17。

VIF：虚拟中断标志，在虚拟方式下，VIF是IF的虚拟映像。

VIP：虚拟中断挂起，指示虚拟中断是否挂起，为虚拟方式提供中断信息

ID：标识标志，ID=1，表示该CPU支持CPUID指令，该指令提供厂商、产品系列等信息。

RF：恢复标志，RF=1，即使遇到断点或调试故障，也不产生异常中断1（单步），成功执行指令时，RF自动清0。

NT：嵌套任务，保护方式下，NT=1，当前执行的任务嵌套在另一个任务中，NT=0，无嵌套。

IOPL：I/O特权级，保护方式下，为I/O寻址操作设置最小保护级，0-3，0为最高优先级，CPL≥IOPL（CPL数值上小于等于IOPL）时，I/O指令才能执行，否则产生异常。只有特权级为0级的程序才能修改IOPL。

与8086相同

保留	ID	VIP	VIF	AC	VM	RF	0	NT	IOPL	OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF
31…22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

段寄存器

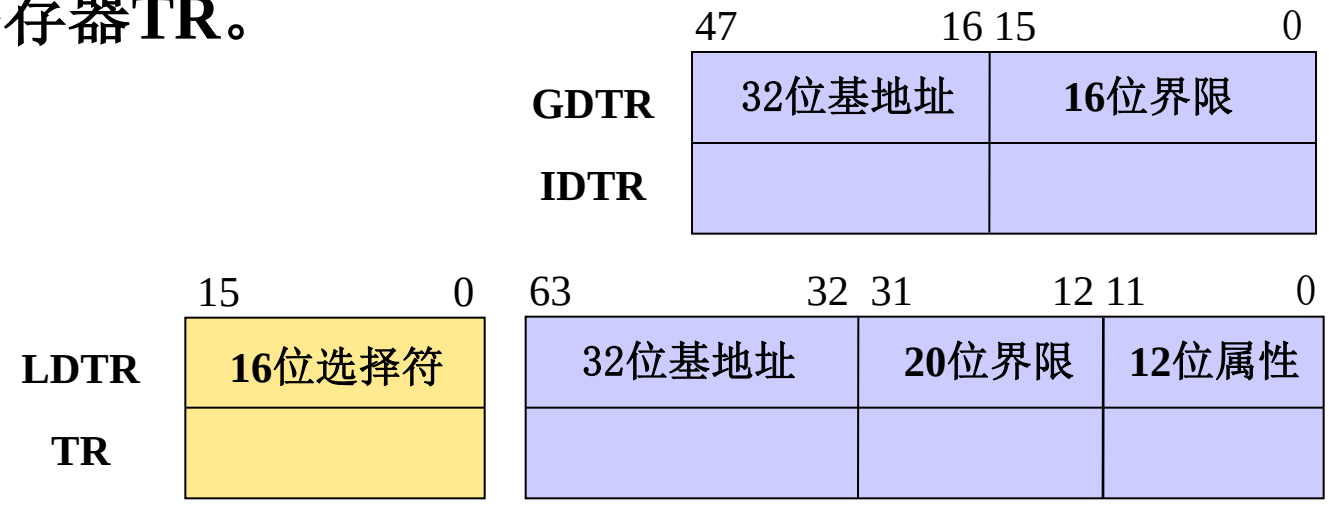
1）Pentium有6个段寄存器，每个都是16位长，CS是代码段寄存器，DS是数据段寄存器， ES、FS、GS是附加段寄存器，SS是堆栈段寄存器。
2）每一个段寄存器都有一个与之相对应的段描述符高速缓存器，用来描述一个段的段基地址、段限（段的范围、长度）和段的属性。
3）段寄存器是程序员可见的，而段描述符寄存器对程序员是透明的。

段寄存器	段选择符16位	段描述符64位		
		段基址32位	段限20位	属性12位
CS				
DS				
SS				
ES				
FS				
GS				

图 3.2.3　Pentium 微处理器段寄存器

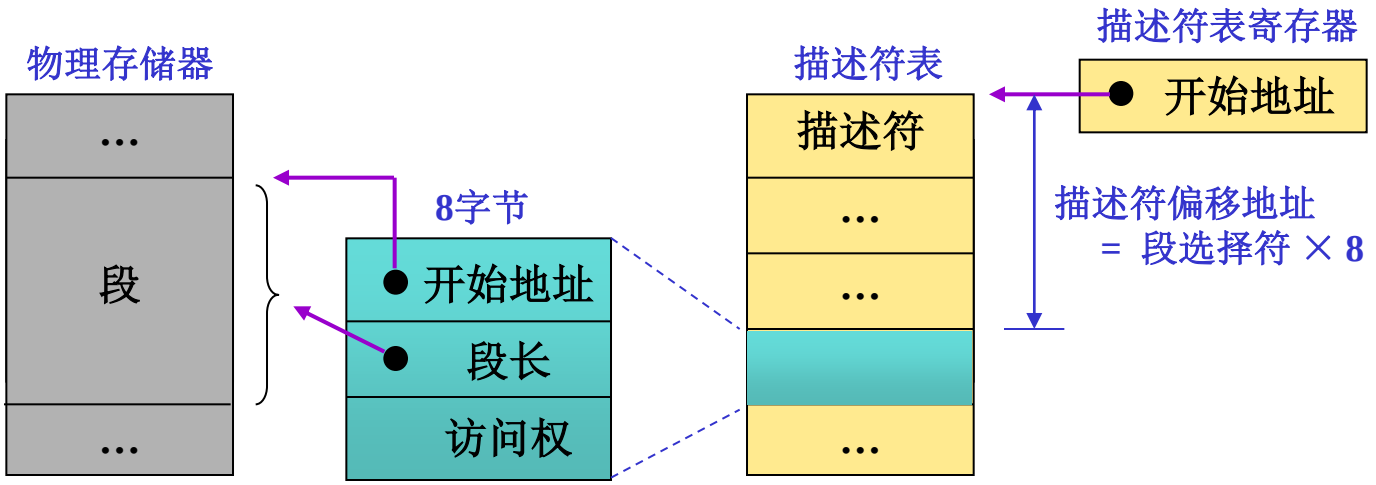
3.2.2 系统级寄存器

- 系统级寄存器包括4个系统地址寄存器和5个控制寄存器。
- 1. 系统地址寄存器
- 由于系统存储管理的需要，配备了4个系统地址寄存器，用于控制分段存储器管理中数据结构的位置。
- 4个系统地址寄存器：全局描述符表寄存器GDTR、中断描述符表寄存器IDTR、局部描述符表寄存器LDTR、任务状态寄存器TR。



段描述符

1）以段为单位为程序分配内存，段的信息是重要的。
2）段描述符：用于描述段的信息，8字节组成。
3）段描述符表：各个段的描述符作为一种数据结构，像数组似的组合在一起，就构成了描述符表。
4）描述符表的开始地址放在描述符表寄存器中，该首地址再加上描述符表偏移量，就会得到相应的描述符。



段描述符表

- **GDT：global descriptor table，全局描述符表，操作系统使用的段描述符和各项任务共用的段描述符放在一起组成的表。整个系统一个。**
- **LDT：local descriptor table，局部描述符表，某项任务专用的所有的各种段描述符放在一起构成的表。每个任务各有一个。**
- **IDT：中断描述符表，一种特殊的表，它把每个中断向量与描述符联系在一起，描述符包含中断服务程序入口地址和特性，整个系统一个。**
- **对应3个描述符表，就有3个描述符表寄存器，用于保存描述符表的开始地址：**
  - **GDTR：保存GDT开始地址和段长，48位。**
  - **LDTR：保存LDT开始地址和段长、属性，16+64位。**
  - **IDTR：保存IDT开始地址和段长，48位。**
- **任务寄存器TR：16+64位，保护方式下使用，保存当前正在执行任务的任务状态段的信息（包括段选择符、基地址、段长、属性），任务切换时，用它自动保存和恢复机器状态。CPU运行程序装入16位的段选择符，由段选择符选择64位的段描述符，装入高速缓存。**

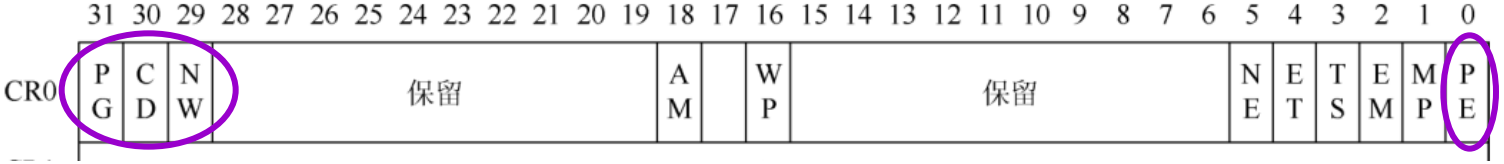
2. 控制寄存器

- **Pentium微处理器由于控制管理的需要，配备了CR0-CR4控制寄存器。**
- **作用：保存全局性机器状态。**
- **CR0：机器状态**
- **CR1：保留**
- **CR2：页故障线性地址寄存器**
- **CR3：页目录基地址**
- **CR4：设置Pentium扩展功能**

图 3.2.5　控制寄存器

CR0

- **PE：保护方式允许，PE=1 保护方式，PE=0 实地址方式。**
- **MP：监视协处理器，MP=1 有协处理器，MP=0 无，用于运行286、386程序。**
- **EM：仿真协处理器，EM=1 由软件仿真协处理器，EM=0 不仿真**
- **TS：任务切换，TS=1 出现任务切换，TS=0 无。**
- **ET：扩展类型，用于386，ET=1 配387，ET=0 配287，Pentium始终置1。**
- **NE：数值异常，NE=1 由16号中断处理浮点故障，NE=0 由外部中断处理。**
- **WP：写保护，WP=1 用户级只读页禁止写入，WP=0 允许。**
- **AM：对齐屏蔽，AM=1 屏蔽对准错误，AM=0 Eflag中的AC位有效。**
- **NW：不透明写，NW=1不透明写，只写Cache，不写主存。 NW=0 透明写，写直达，既写Cache，又写主存。**
- **CD：禁止Cache。CD=1 禁止片内Cache填充。CD=0 允许。**
- **PG：允许分页，PG=1 允许分页，PG=0 禁止。**



CR3

- **CR3：保存页目录表的基地址，PG=1 有效，高20位有效，页目录表按页对齐（4KB）。**
- **PWT：页面写直达，对外部Cache，PWT=1 外部Cache写直达，PWT=0 外部Cache写回。**
- **PCD：页面Cache禁止，对外部Cache，PCD=1 禁止外部Cache，PCD=0 允许。**

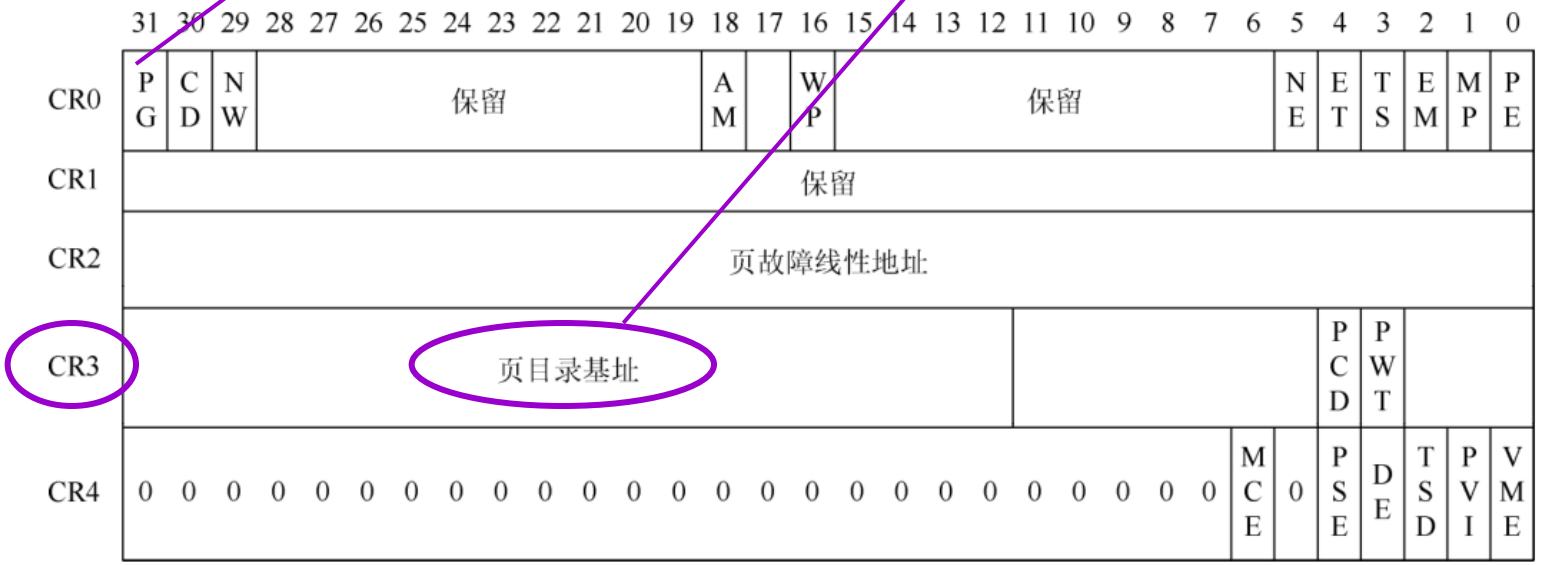


图 3.2.5　控制寄存器

3.3 Pentium 微处理器外部结构

- Pentium 微处理器的主要引脚及其功能：
- 1. 地址线及控制
- 2. 数据线及控制
- 3. 总线周期控制
- 4. Cache 控制
- 5. 系统控制
- 6. 总线控制

1. 地址线及控制

- $\overline{\text{A20M}}$：A20以上地址线屏蔽信号，与ISA总线兼容，仿真8086CPU 1M空间。
- A31～A3：地址总线，寻址4GB。内部Cache，地址双向。

  A31～A3突发传送时不变。

  A2-A0在CPU中被编码，形成$\overline{\text{BE7}}$-$\overline{\text{BE0}}$的输出，用于字节选择。
- $\overline{\text{ADS}}$：地址选通，有效时表示CPU启动1个总线周期，T1有效。
- AP：地址偶校验，为存储和I/O传送提供地址偶校验位。
- $\overline{\text{APCHK}}$：地址校验检查，当检查到地址校验错时该信号变为逻辑0。

2. 数据线及控制

- $\overline{\text{BE7}}$～$\overline{\text{BE0}}$：字节允许，对应每个字节，由地址A2-A0产生。
- D63～D0：数据总线。数据读写时，按字节偶校验。
- DP7～DP0：数据偶校验，每一位依次对应数据线上一个字节的偶校验。
- $\overline{\text{PCHK}}$：奇偶校验检查，有效表明读数据时偶校验出错。
- $\overline{\text{PEN}}$：奇偶校验允许，如果该信号为逻辑0，只要读校验出错，CPU就会自动执行异常处理。若内存配置不带奇偶校验位，应使$\overline{\text{PEN}}$ =1无效，否则引起异常中断。

3. 总线周期控制

- **D/$\overline{C}$**：数据/控制。
- **M/$\overline{IO}$**：存储器/IO。
- **W/$\overline{R}$**：写/读。
- **$\overline{BRDY}$**：突发就绪，类似Ready，结束一个总线周期，此时外设准备好，将进入总线流水，可用于确定是否插入等待状态。
- **$\overline{NA}$**：下一地址有效，用于支持地址流水线操作。采样到$\overline{NA}$有效的两个时钟周期后，CPU可送出新地址。
- **$\overline{LOCK}$**：总线锁定，LOCK前缀使该信号有效，用来指示现行总线周期不能被打断。
- **SCYC**：分割周期，地址指针未对准，需用2个总线周期。

4. Cache控制

- $\overline{\text{Cache}}$：Cache控制，指示目前处于Cache周期。
- PCD：页面Cache禁止，反映CR3的PCD位状态，对外部Cache，有效时禁止。
- PWT：页面写直达，反映CR3的PWT位状态，对外部Cache。
- $\overline{\text{FLUSH}}$：Cache擦除，数据Cache中修改过的行写回操作，并使相应Cache行无效，对内部Cache。
- $\overline{\text{KEN}}$：Cache允许，有效时，指示进入突发读周期，将外部数据复制到内部Cache。
- WB/$\overline{\text{WT}}$：写回/写直达，对内部Cache。
- $\overline{\text{EADS}}$：外部地址选通，指示地址总线由外部部件驱动。
- $\overline{\text{HIT}}$：Cache命中。
- $\overline{\text{HITM}}$：命中的Cache行被修改，用于在已修改过的Cache行写回到存储器之前禁止其他单元访问这些数据。
- INV：无效请求，使内部Cache行变为无效。
- AHOLD：地址保持请求，和$\overline{\text{EADS}}$一起，用于Cache监听，保持Cache一致性，在DMA，存储器发生变化，若这两个信号有效，则Cache与存储器相同地址的内容无效。
- $\overline{\text{EWBE}}$：外部写缓冲器空，若$\overline{\text{EWBE}}$=1无效，则外部Cache E（互斥）和M（修改）的行禁止写入。

5. 系统控制

- **CLK**：时钟。
- **RESET**：复位，冷复位，1）实地址方式；2）A31-A20为FFFH（12个），3）CS=F000H，EIP=0000FFF0H；4）清0 Cache和浮点寄存器；5）清0相关寄存器。
- **INIT**：初始化，热复位，与RESET类似，不清0 Cache、浮点寄存器。

RESET和INIT的区别：
1）RESET立即复位。
2）INIT当前指令结束后复位，且不清0 Cache、浮点寄存器。

表 3.3.1　复位后部分寄存器的值

寄存器	RESET 值
EAX	0
EDX	0500××××H
EBX、ECX、ESP、EBP、ESI、EDI	0
EFLAGS	2
EIP	0000FFF0H
CS	F000H
DS、ES、SS、FS、GS	0
GDTR、TR	0
CR0	60000010H
CR2、CR3、CR4	0

6. 总线控制

- **HOLD**：总线保持请求，请求Pentium停止总线驱动。完成当前总线周期后让出总线，用HLDA响应。
- **HLDA**：总线保持响应，指示Pentium让出总线控制。
- $\overline{\text{BREQ}}$：总线请求，输出，指示Pentium内部已产生了一个总线请求。CPU向其他总线主控设备说明，它正在占用总线。
- **BOFF**：总线释放，总线脱离，强制，当前时钟周期后，立即让出总线。
- **INTR**：可屏蔽中断请求。
- **NMI**：非屏蔽中断请求。

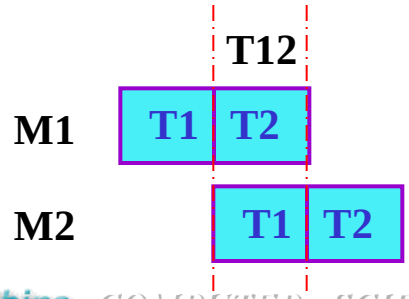
3.4 Pentium 微处理器的总线周期

- 多种数据传送周期：
  - 单数据传送
  - 突发式数据传送
  - 非缓存式（属于单数据）
  - 缓存式 （属于突发式）
  - 非流水线式
  - 流水线式

- 1个总线周期通常由多个时钟周期组成。
- 1个时钟周期对应1个总线状态。每个状态占用1个完整的CLK时钟。
- Pentium有6种状态，每种状态采用Inter统一的缩写符表示，基本总线周期占用2个总线状态。
- 令总线周期为M，第i个总线周期为Mi。

6 种总线状态

- **Ti：空闲状态，不执行总线周期，处于等待、就绪状态，BOFF、RESET信号进入Ti。**
- **T1：总线状态1，地址、状态有效，ADS有效，外部电路可以利用ADS将地址和状态送入锁存器，它是总线周期的第1个时钟周期。**
- **T2：总线状态2，数据有效，它是总线周期的第2个或更后一些的时钟周期，此时CPU对BRDY采样，为低就绪，为高等待。**
- **T12：流水线状态1，两个总线周期部分重叠，M1的T2与M2的T1重叠。**
- **T2p：流水线状态2，两个总线周期部分重叠，M1的T2与M2的T2重叠。若T12已经完成，而M1仍未结束，就会在T12之后形成T2p，一般出现在存储器或外设较慢的情况。**
- **TD：休止状态，读写转换的过渡状态，出现在T12之后，插入在读写操作之中，用于调整，此时，数据无效，CPU不对BRDY采样。奔腾在流水线访问期间使用TD，非流水线的读写转换通常用Ti描述。**



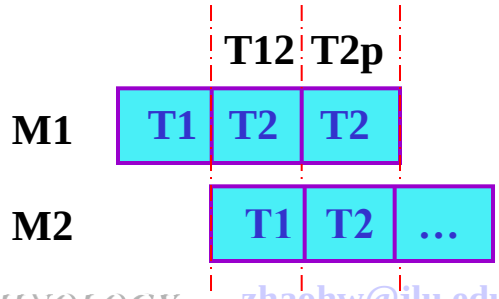


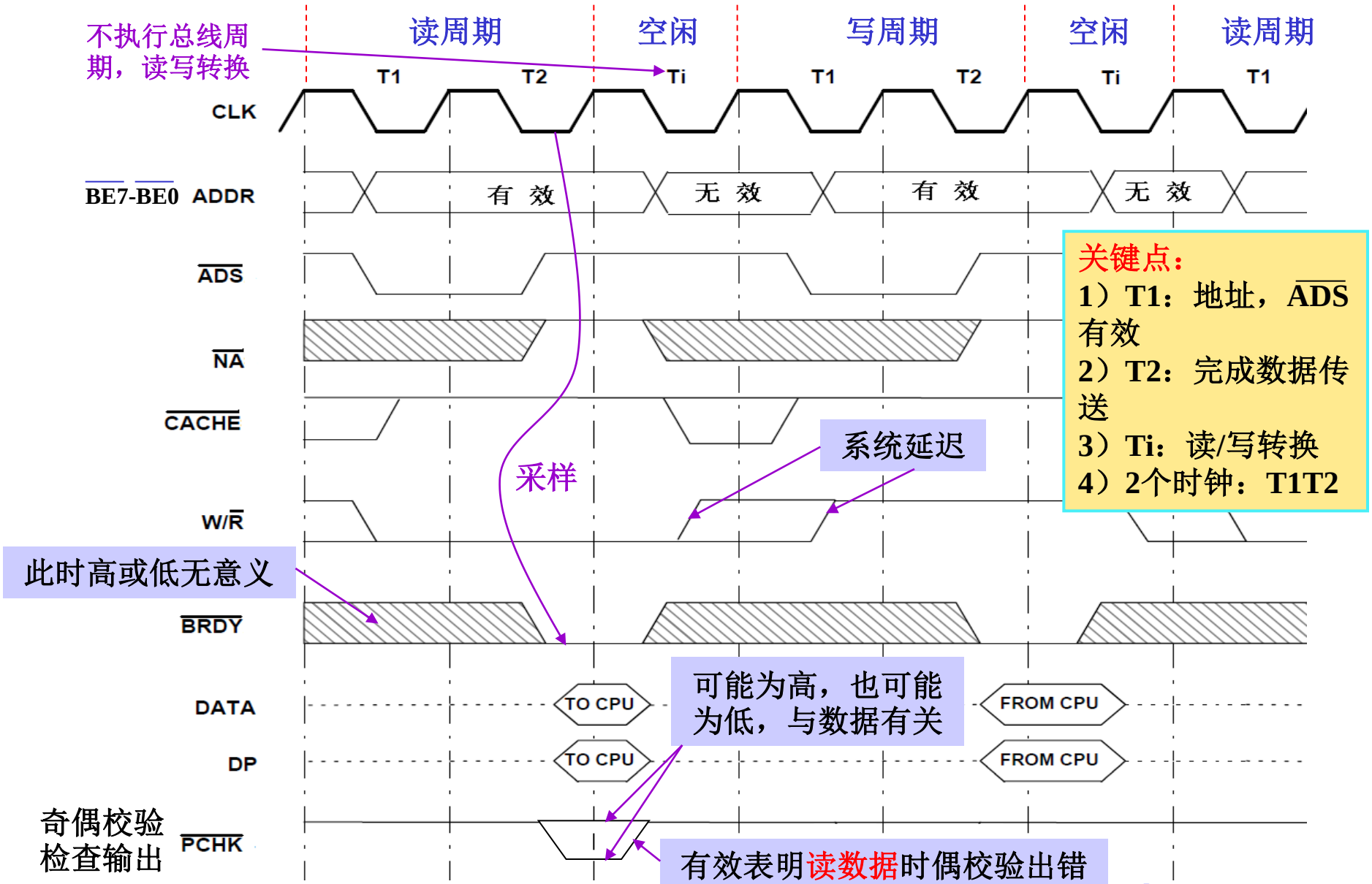
表3.4.1 总线周期类型

输入，Cache允许，控制读操作，Cache充填

	总线周期类型						
	M/$\overline{IO}$	D/$\overline{C}$	W/$\overline{R}$	$\overline{Cache}$	$\overline{KEN}$	总线周期	传送次数
I/O操作	0	0	0	1	x	中断响应周期（两个锁定周期）	每周期传1次
	0	0	1	1	x	特殊总线周期	1
	0	1	0	1	x	I/O读，小于等于32位，非缓冲	1
	0	1	1	1	x	I/O写，小于等于32位，非缓冲	1
代码读	1	0	0	1	x	代码读，64位，非缓冲	1
	1	0	0	x	1	代码读，64位，非缓冲	1
	1	0	0	0	0	代码读，256位，猝发式数据行填充	4
	1	0	1	x	x	保留	n/a
M读	1	1	0	1	x	存储器读，小于等于64位，非缓冲	1
	1	1	0	x	1	存储器读，小于等于64位，非缓冲	1
	1	1	0	0	0	存储器读，256位，猝发式数据行填充	4
M写	1	1	1	1	x	存储器写，小于等于64位，非缓冲	1
	1	1	1	0	x	256位，突发式写回	4

I/O可处理4个字节

3.4.1 非流水线式读写总线周期



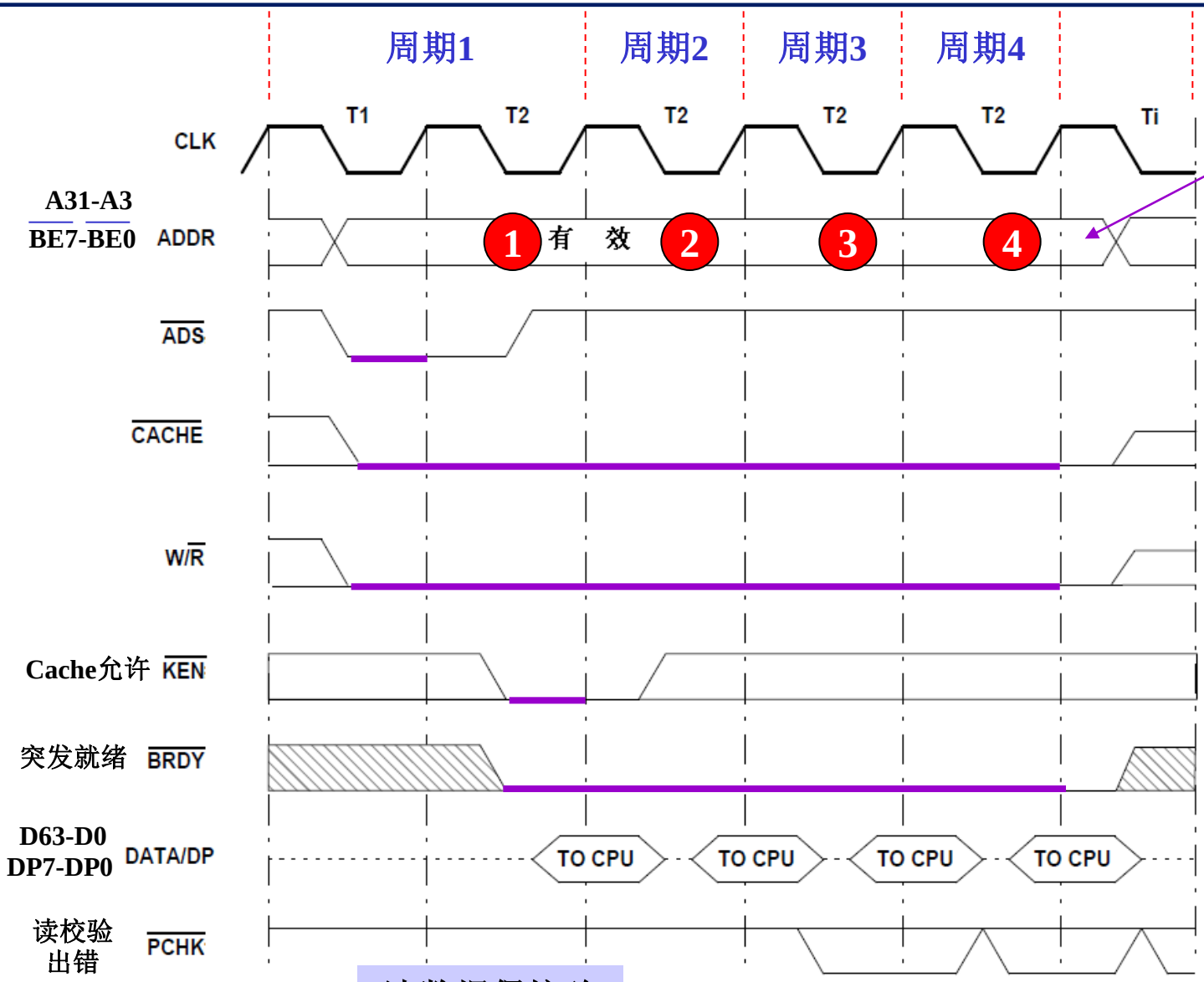
3.4.2 突发式读写总线周期

- **Pentium微处理器有三类突发式总线周期：**

  - 代码读突发式行填充
  - 数据读突发式行填充
  - 突发式写回

- **对Cache操作，每周期256位（一个Cache行为32字节）。**
- **每次传64位（8个字节）。**
- **连续传4次。**

图 3.4.2 突发式读总线周期



该地址一直有效，是四倍双字地址，外部电路必须将地址递增，指出后续3个双字地址。

关键点：

1）整个周期有效信号：Cache，地址，W/R。

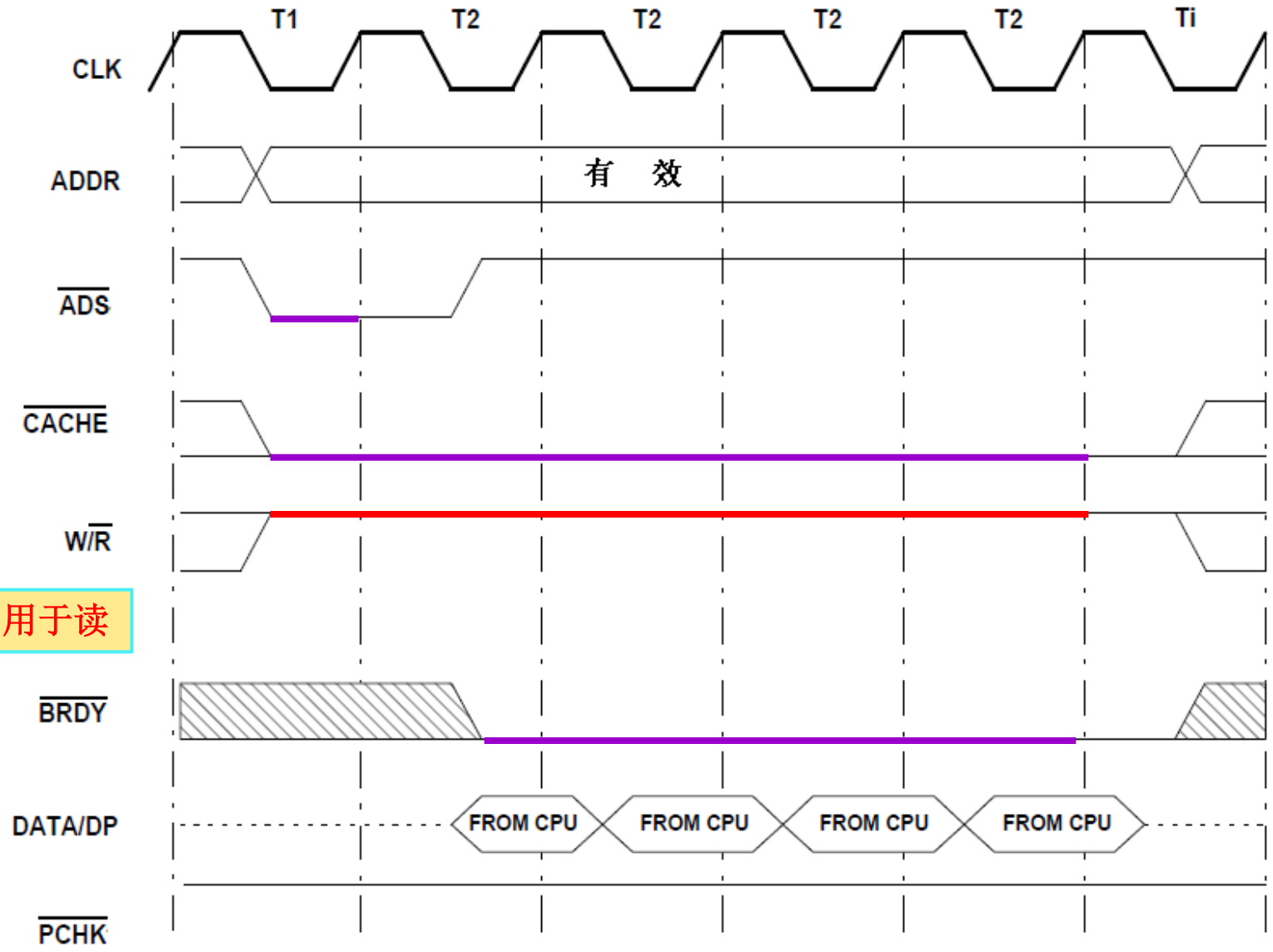
2）KEN在第1个T2有效，用KEN指示开始突发式周期。

3）突发就绪BRDY在所有T2有效。

4）5个时钟：T1T2T2T2T2

读数据偶校验

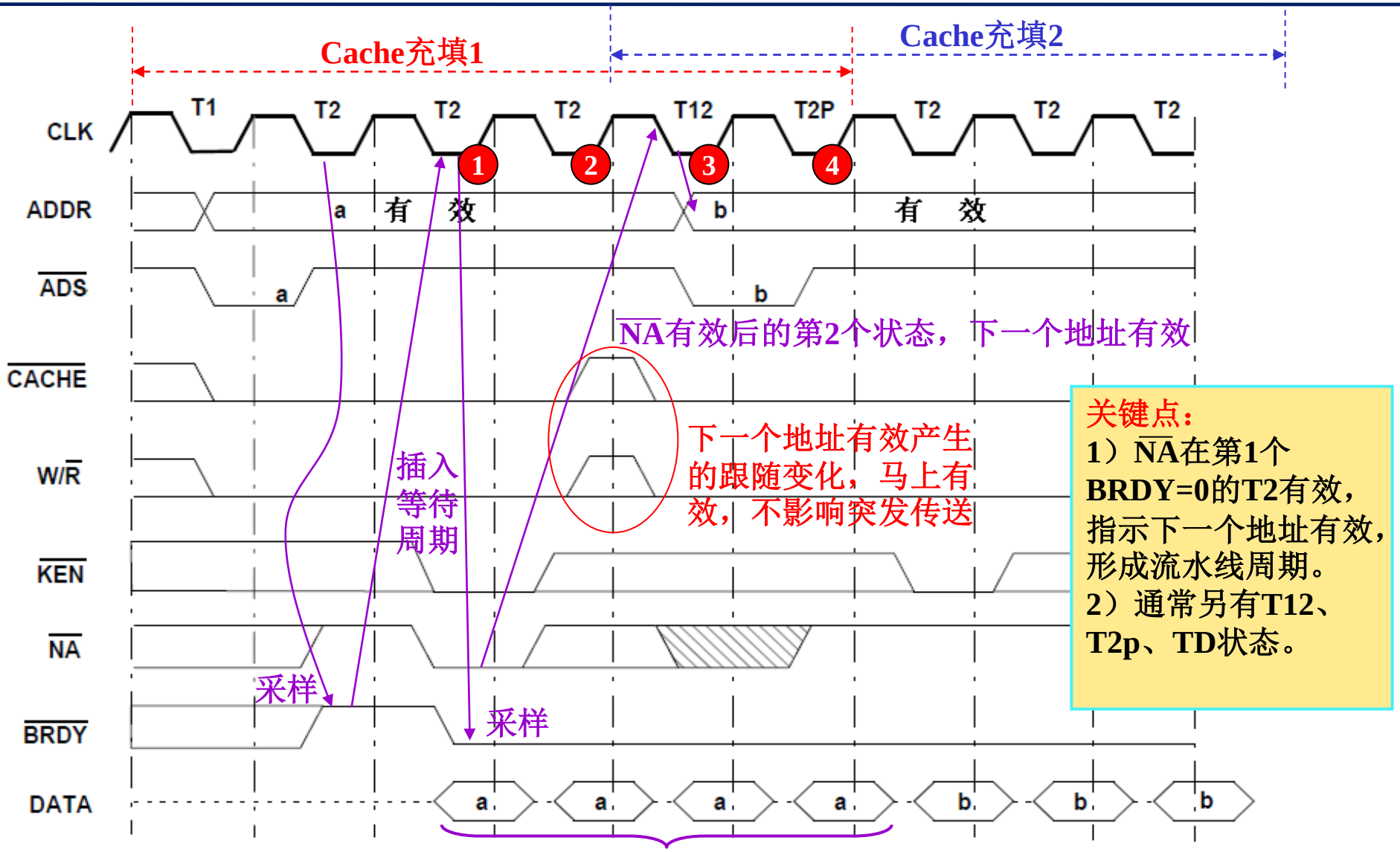
图 3.4.3 突发式写总线周期



3.4.3 流水线式读写总线周期

- 用“下一个地址”$\overline{\text{NA}}$信号形成流水线周期。
- 单数据传送周期和突发式周期都可以是流水线式的。
- 流水线周期中，数据传送逻辑和地址解码逻辑并行工作，即当前总线周期结束前，也就是正在传送数据时，地址逻辑已经开始接收下一个总线周期的地址。

图 3.4.4 流水线式突发式读总线周期



关键点：
1）NA在第1个BRDY=0的T2有效，指示下一个地址有效，形成流水线周期。
2）通常另有T12、T2p、TD状态。

4次传送，每次8个字节，共32字节，32×8=256位

3.5 Pentium微处理器的操作模式

- **Pentium的操作模式：**

{ 实地址模式
{ 保护虚地址模式 { 一般保护模式
虚拟8086模式

3.5.1 实地址模式

- 简称实地址方式，Pentium与8086兼容，基本体系结构相同。
- 能有效地使用8086所没有的寻址方式、32位寄存器和大部分指令。
- 存储空间为1M字节。不允许分页，所以线性地址就是物理地址。
- 物理地址=段地址×16+偏移地址
- 保留2个物理存储空间：

  0000:0000H～0000:03FFH：中断向量区，每个中断向量占用4个字节

  FFFF:0000H～FFFF:000FH：系统初始化区。
- 在实地址方式下，可以把Pentium设置为保护方式。

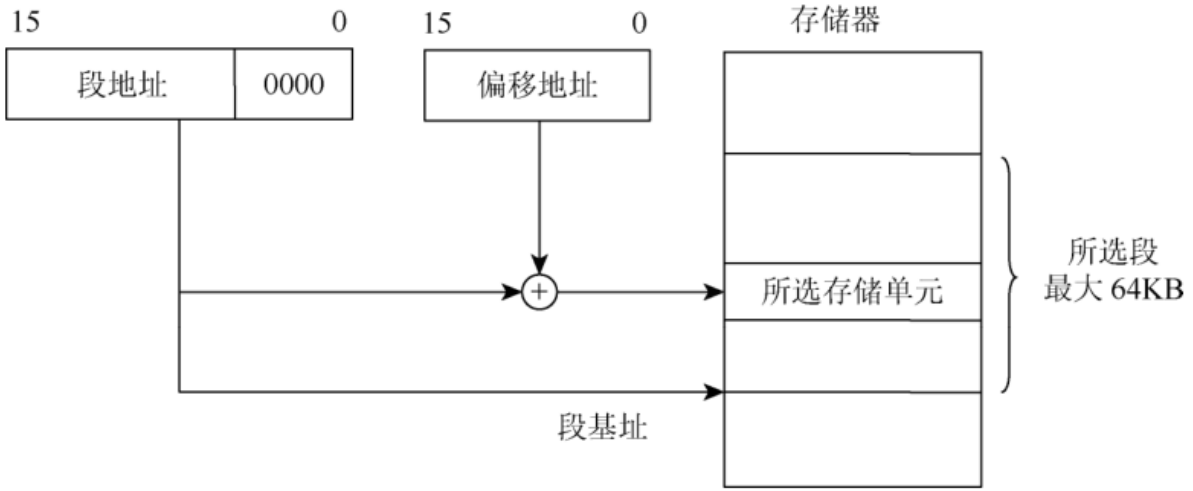


图 3.5.1　实地址模式存储器寻址

3.5.2 保护虚拟地址模式

- 简称保护方式 。4GB物理空间，64TB逻辑空间 。
- 在保护方式下，逻辑地址由选择符和偏移地址两部分组成，选择符存放在段寄存器中，但它不能直接表示段基地址，而由操作系统通过一定的方法取得段基地址，再和偏移地址相加，从而求得所选存储单元的物理地址。

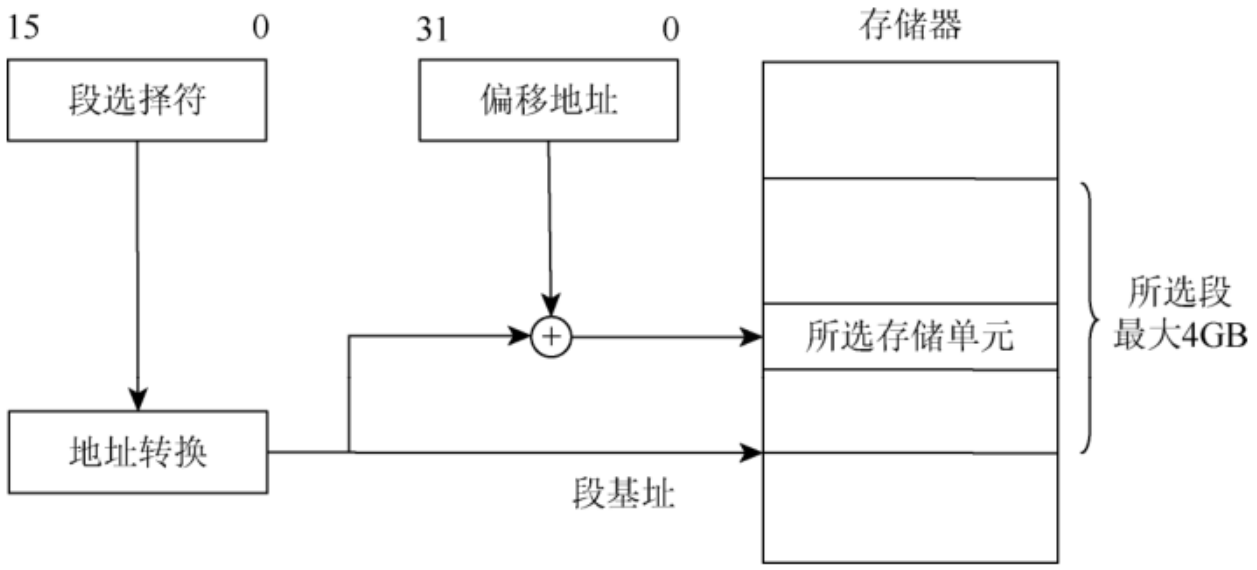


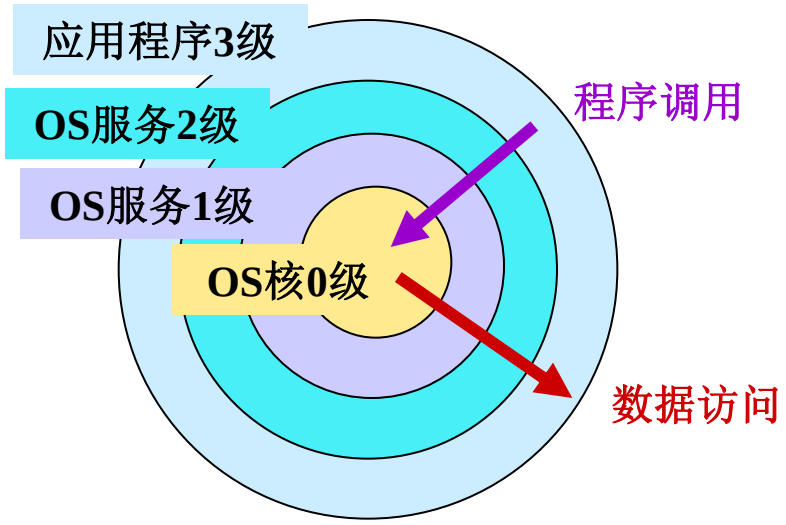
图 3.5.2　保护模式存储器寻址

环保护方式：特权级保护

- **Pentium微处理器有多种保护方式，其中最突出的是环保护方式。环保护是在用户程序与用户程序之间、用户程序与操作系统之间实行隔离，通过特权级实现。**
- **特权级：4级，0～3（高～低），0级分配给操作系统核心。**
- **规则：**

  高特权级可以访问低特权级的数据（大于等于）

  低特权级可以调用高特权级的程序（小于等于）



虚拟8086方式

- Pentium微处理器允许在实方式和保护方式下执行8086的应用程序。
- 虚拟8086方式相当于程序运行在实地址方式。
- 通过使用分页功能，可以把虚拟8086方式下的1MB地址空间映象到Pentium微处理器的4GB的物理空间中的任何位置。
- 虚拟8086方式与实地址方式的不同：

  1）虚拟8086方式是一个程序的运行方式。

  2）实地址方式是处理器的工作方式。

3.6 Pentium 微处理器的存储器组织

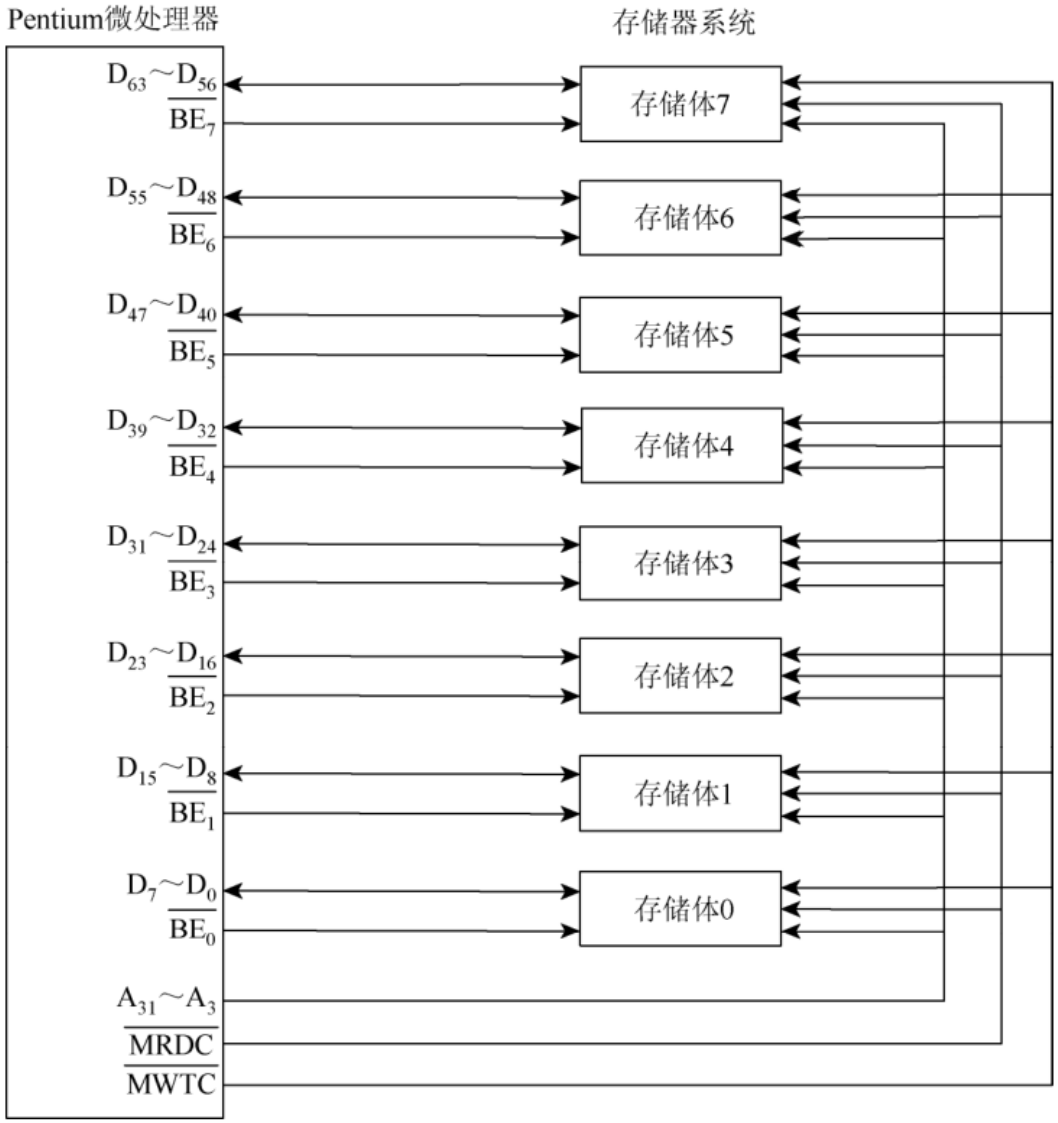


图 3.6.2　Pentium 存储器系统的 8 个存储体结构示意图

Pentium 微处理器具有 64 位的外部数据线，其要求能对存储器任意地址的字节、字、双字和四倍字进行访问。

图 3.6.2 描述了 Pentium 微处理器的存储器组织及其 8 个存储体。访问时也有数据对准问题，不对准数据的访问会增加数据读写的总线周期。

结 束