计算机组成原理

顾崇林
计算机科学与技术学院
guchonglin@hit.edu.cn

第四章 存储器

4.1存储器概述

一、存储器分类

1. 按存储介质分类

(1) 半导体存储器　　TTL 、MOS　　易失

(2) 磁表面存储器　　磁头、载磁体

(3) 磁芯存储器　　硬磁材料、环状元件

(4) 光盘存储器　　激光、磁光材料

非易失

2. 按存取方式分类

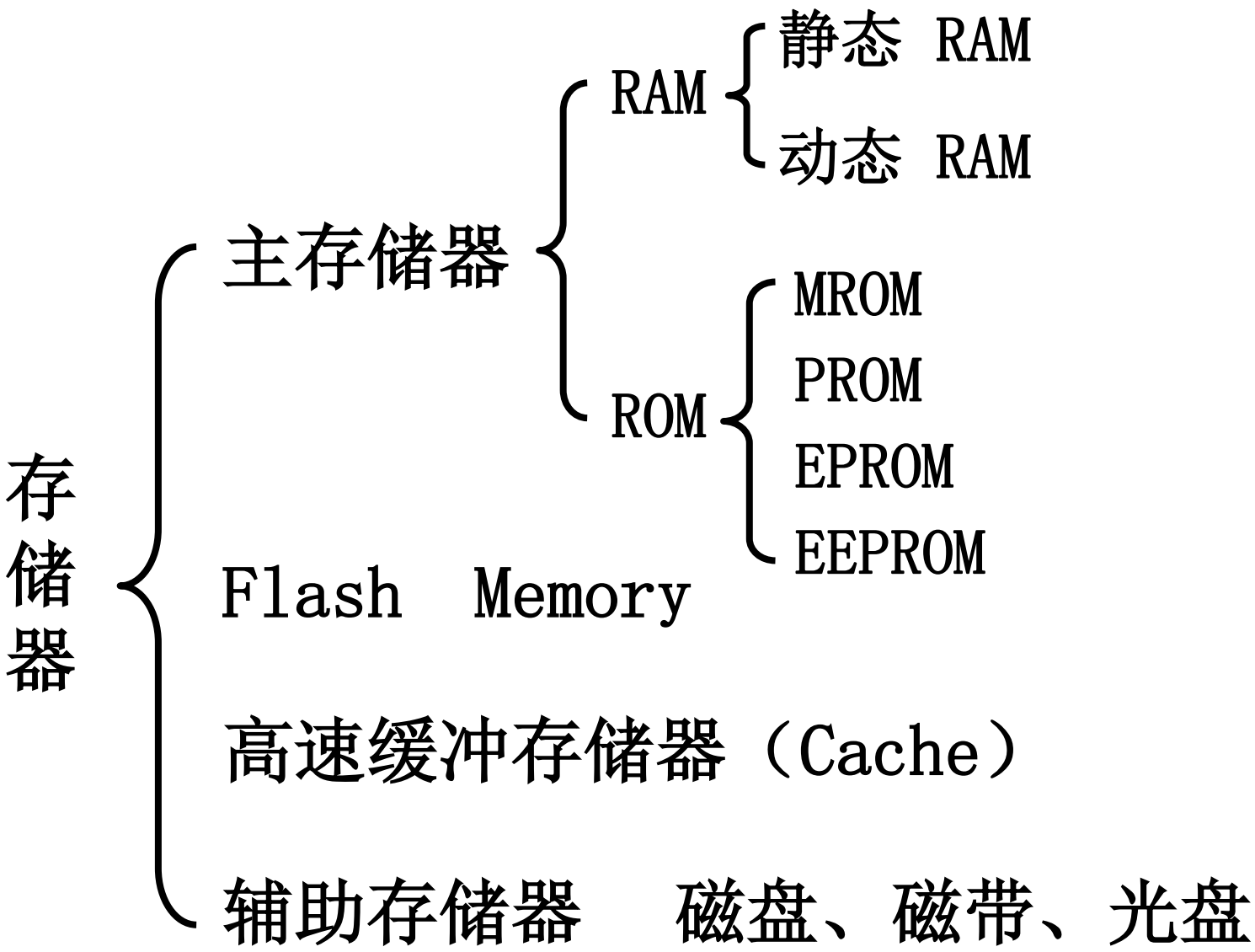
(1) 存取时间与物理地址无关（随机访问）

- 随机存储器　在程序的执行过程中 可 读 可 写
- 只读存储器　在程序的执行过程中 只 读

(2) 存取时间与物理地址有关（串行访问）

- 顺序存取存储器　　磁带
- 直接存取存储器　　磁盘

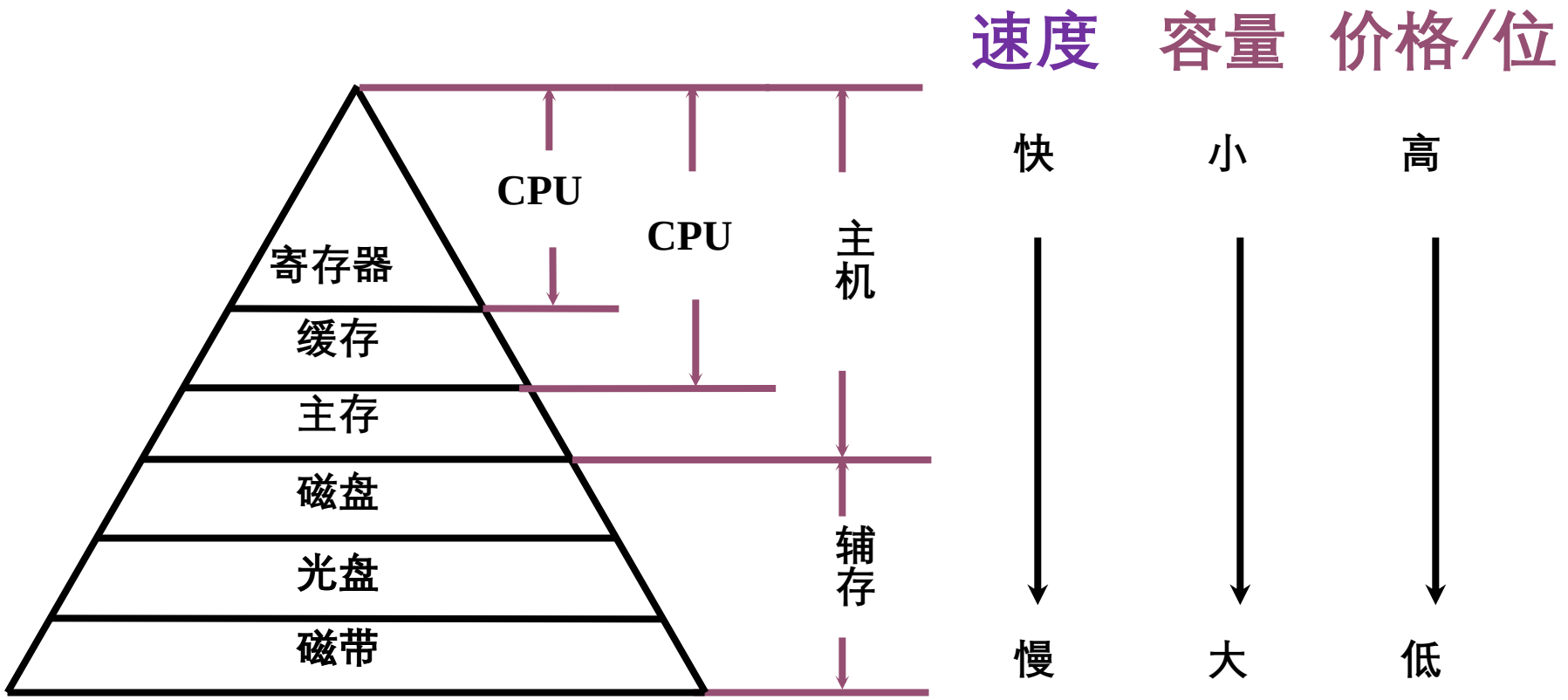
3. 按在计算机中的作用分类



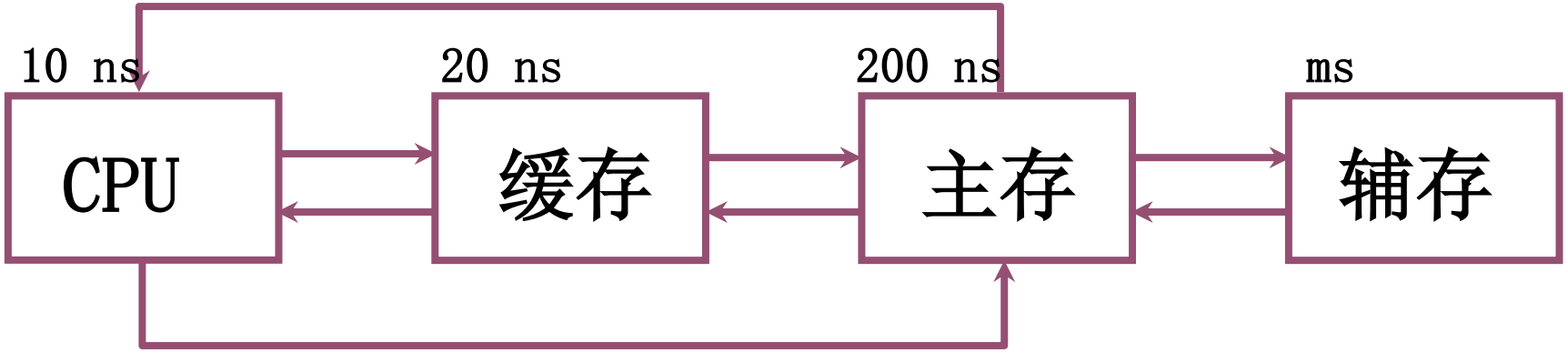
4.1存储器概述

- 二、存储器的层次结构

1. 存储器三个主要特性的关系



2. 缓存－主存层次和主存－辅存层次



（速度）　　（容量）

缓存－主存　　主存－辅存

主存储器	虚拟存储器
实地址	虚地址
物理地址	逻辑地址

第四章 存储器

4.1 存储器概述

4.2 主存储器

4.3 高速缓冲存储器

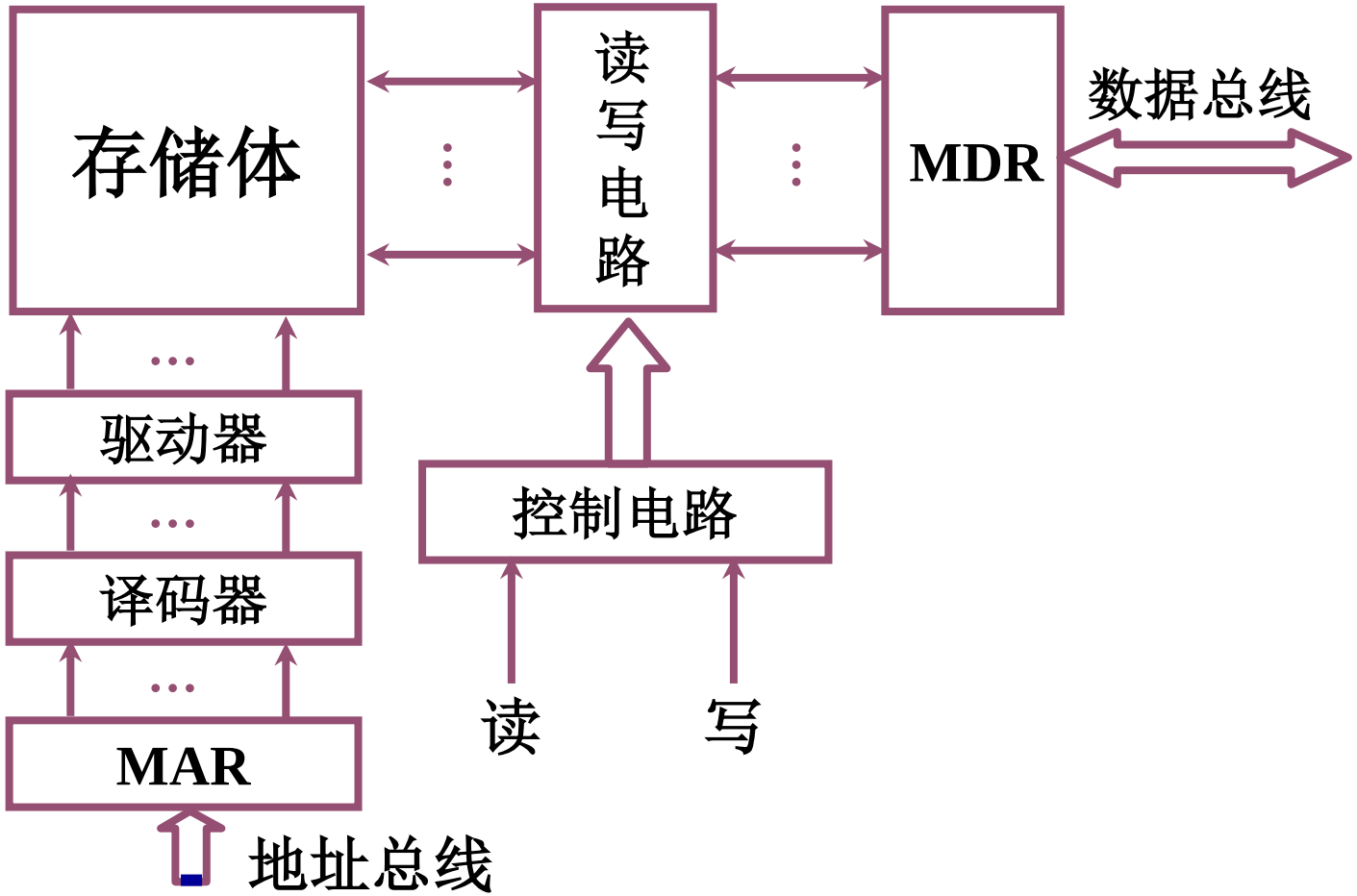
4.4 虚拟存储器

4.5 辅助存储器

4.2主存储器

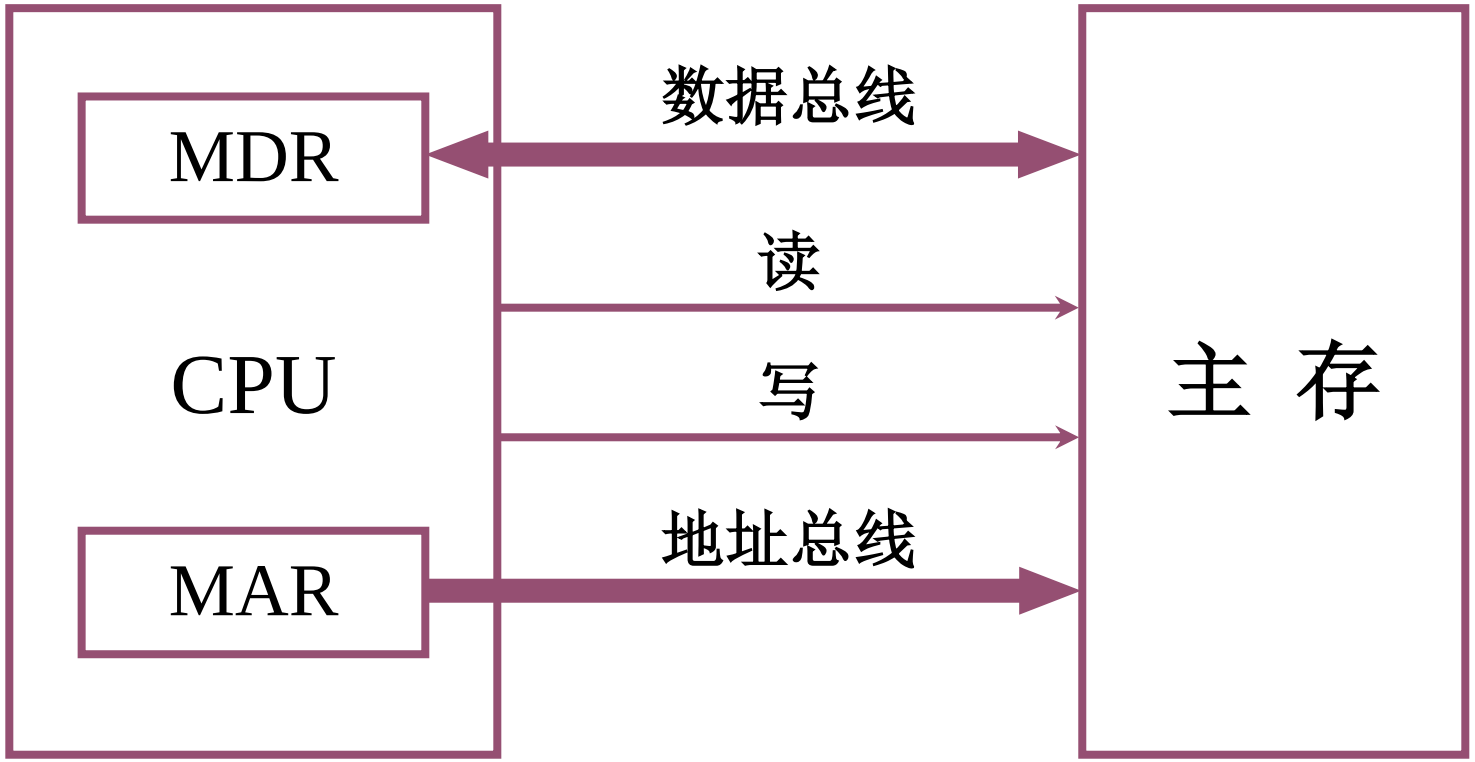
- 一、概述

1. 主存的基本组成



4. 2主存储器

2. 主存的和 CPU 的联系



4. 2主存储器

3. 主存中存储单元地址的分配

高位字节 地址为字地址

字地址	字节地址			
0	0	1	2	3
4	4	5	6	7
8	8	9	10	11

低位字节 地址为字地址

字地址	字节地址	
0	1	0
2	3	2
4	5	4

设地址线 24 根　　按 字节 寻址　2^{24} = 16 M

若字长为 16 位　　按 字 寻址　8 M

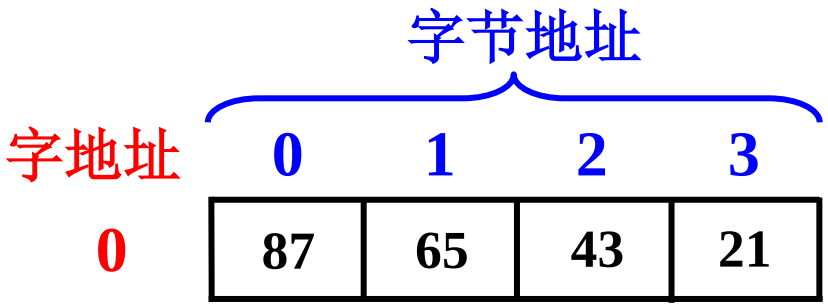
若字长为 32 位　　按 字 寻址　4 M

4.2主存储器

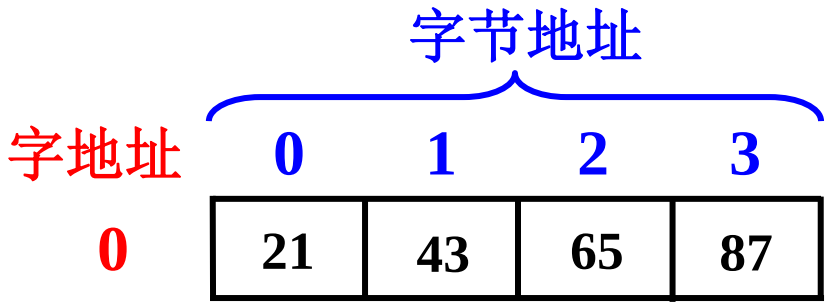
3. 主存中存储单元地址的分配

例：给定一个十六进制数87654321H，其存储格式分别是：

1）**大端（大尾）模式**：字数据的高字节存储在低地址中。而字数据的低字节则存放在高地址中。

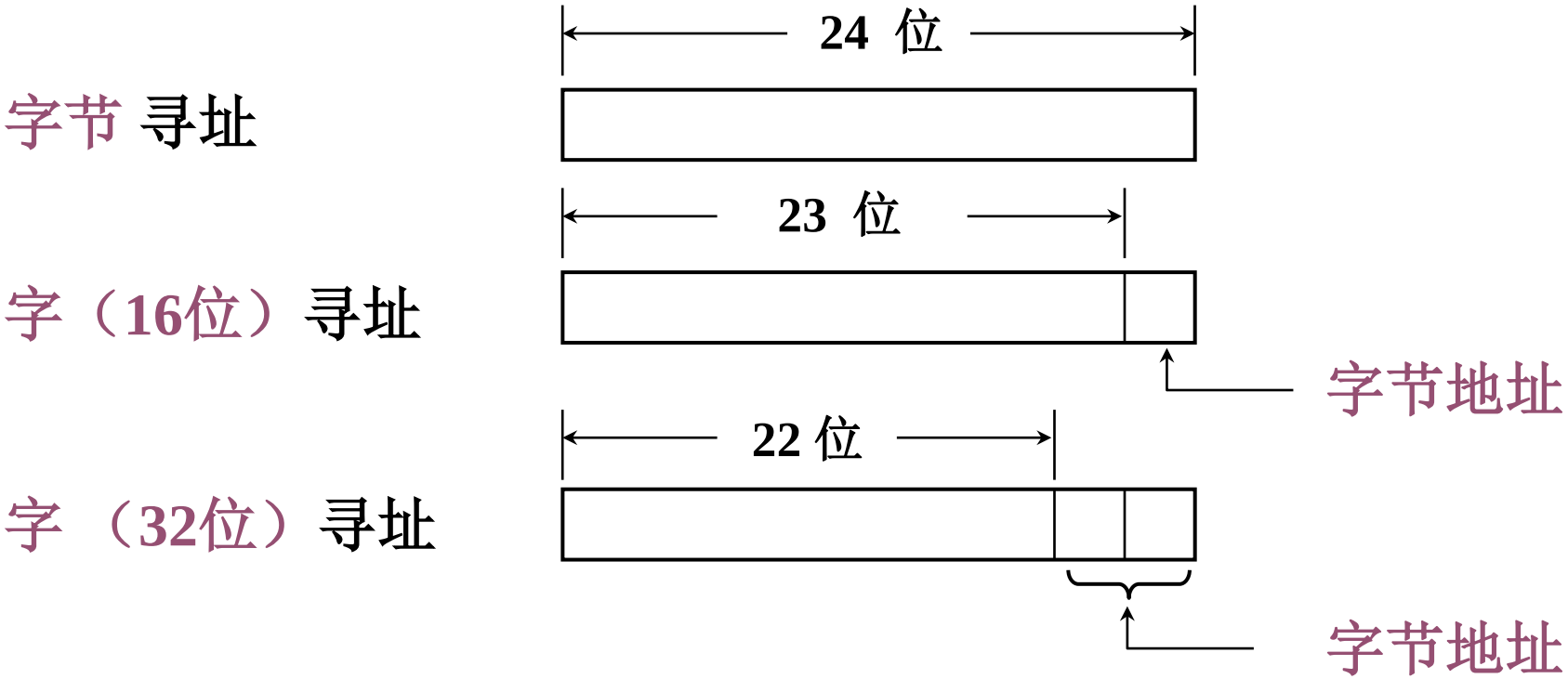


2）**小端（小尾）模式**：字数据的高字节存储在高地址中。而字数据的低字节则存放在低地址中。



如 16 MB （2^{27}位）的存储器

	寻址范围	容量
按 字节 寻址	2^{24}= 16 M	$2^{24} \times 2^3 = 2^{27}$位
按 字（16位）寻址	2^{23}= 8 M	$2^{23} \times 2^4 = 2^{27}$位
按 字 （32位）寻址	2^{22}= 4 M	$2^{22} \times 2^5 = 2^{27}$位



4. 主存的技术指标

(1) 存储容量 主存 存放二进制代码的总位数

(2) 存储速度

- 存取时间 存储器的 访问时间

  读出时间 写入时间

- 存取周期 连续两次独立的存储器操作

  （读或写）所需的 最小间隔时间

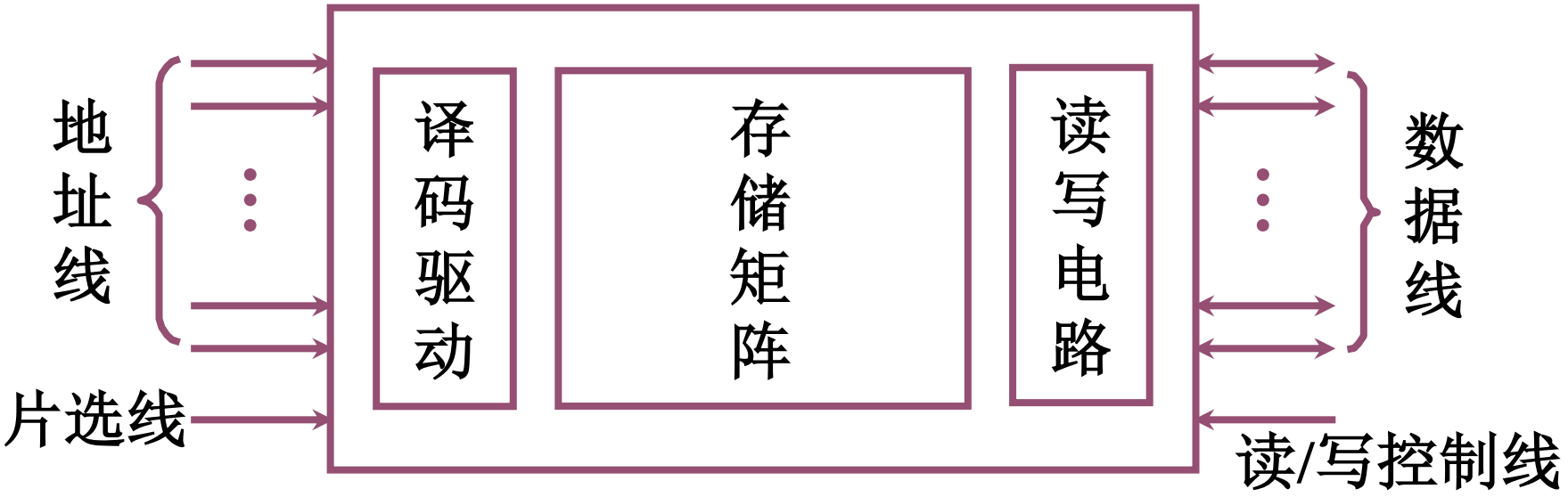
  读周期 写周期

(3) 存储器的带宽 单位时间内存储器存取的信息量(位/秒)

4.2主存储器

二、半导体存储芯片简介

1. 半导体存储芯片的基本结构

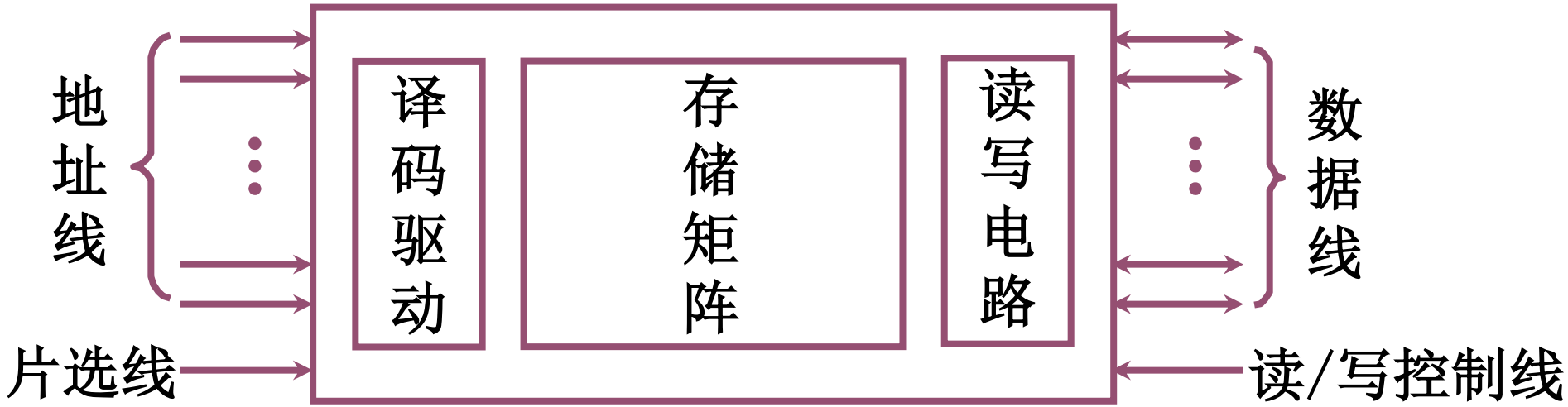


地址线（单向）	数据线（双向）	芯片容量
10	4	1K×4位
14	1	16K×1位
13	8	8K×8位

4.2主存储器

二、半导体存储芯片简介

1. 半导体存储芯片的基本结构



片选线 $\overline{CS}$ $\overline{CE}$

读/写控制线 $\overline{WE}$（低电平写 高电平读）

$\overline{OE}$（允许读）$\overline{WE}$（允许写）

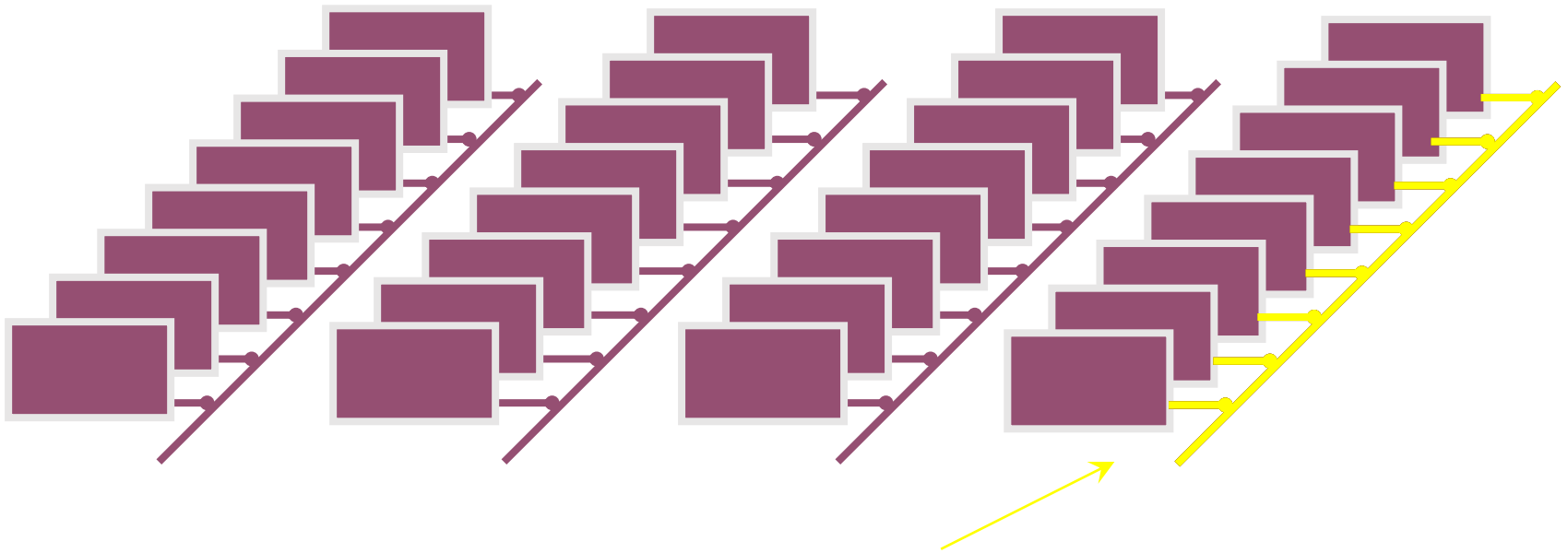
4.2主存储器

- 存储芯片片选线的作用

用 16K × 1位 的存储芯片组成 64K × 8位 的存储器

32片

8片 16K × 1位　8片 16K × 1位　8片 16K × 1位　8片 16K × 1位

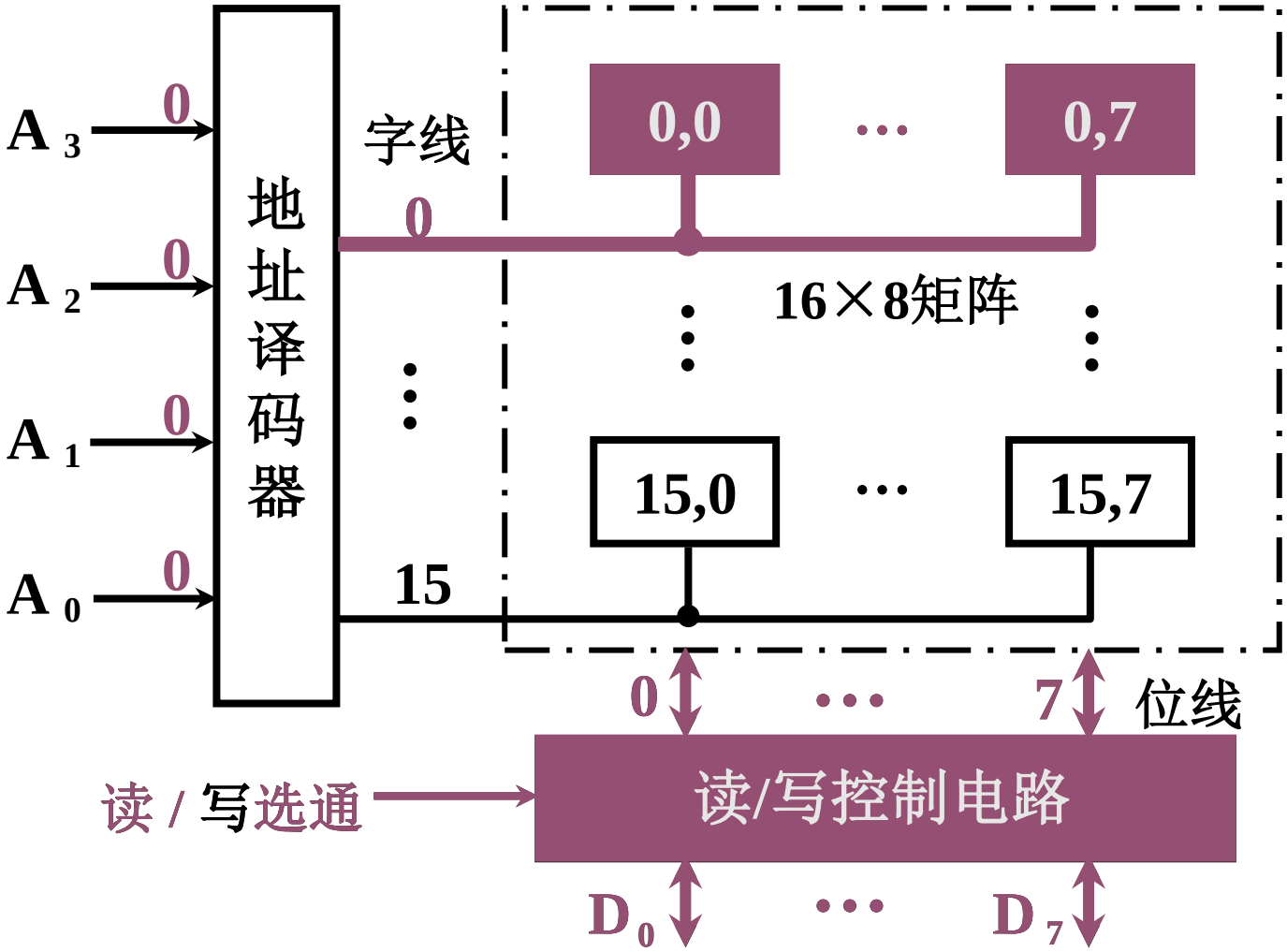


当地址为 65 535 时，此 8 片的片选有效

4. 2主存储器

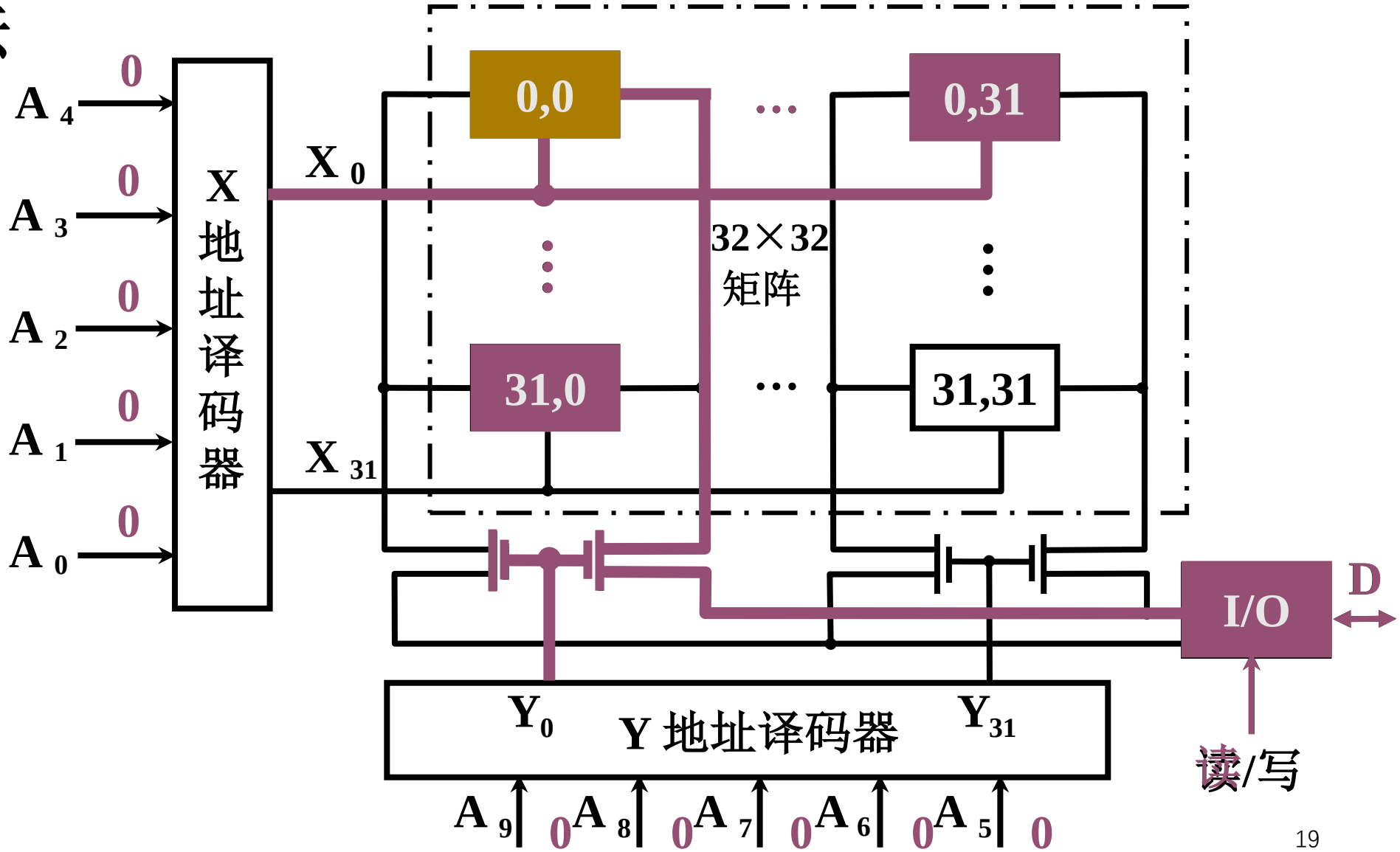
2. 半导体存储芯片的译码驱动方式

(1) 线选法



4. 2主存储器

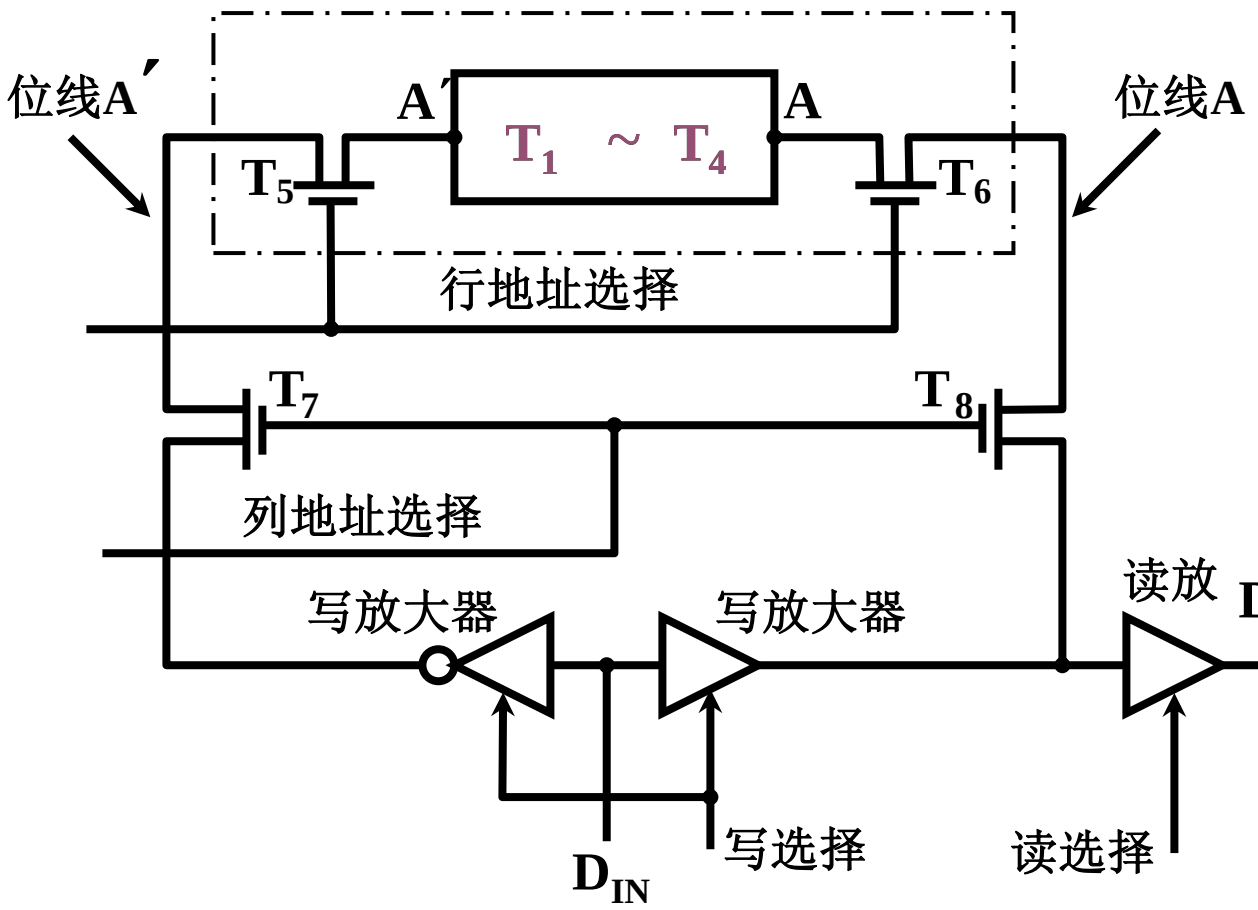
(2) 重合法


A 4
A 3
A 2
A 1
A 0
0
0
0
0
0
X 地址译码器
X 0
X 31
0,0
0,31
31,0
31,31
32×32 矩阵
I/O
D
读/写
Y 0
Y 地址译码器
Y 31
A 9
A 8
A 7
A 6
A 5
0
0
0
0
0

三、随机存取存储器（ RAM ）

1. 静态 RAM (SRAM)

(1) 静态 RAM 基本电路



$T_1 \sim T_4$ 触发器

T_5、T_6 行开关

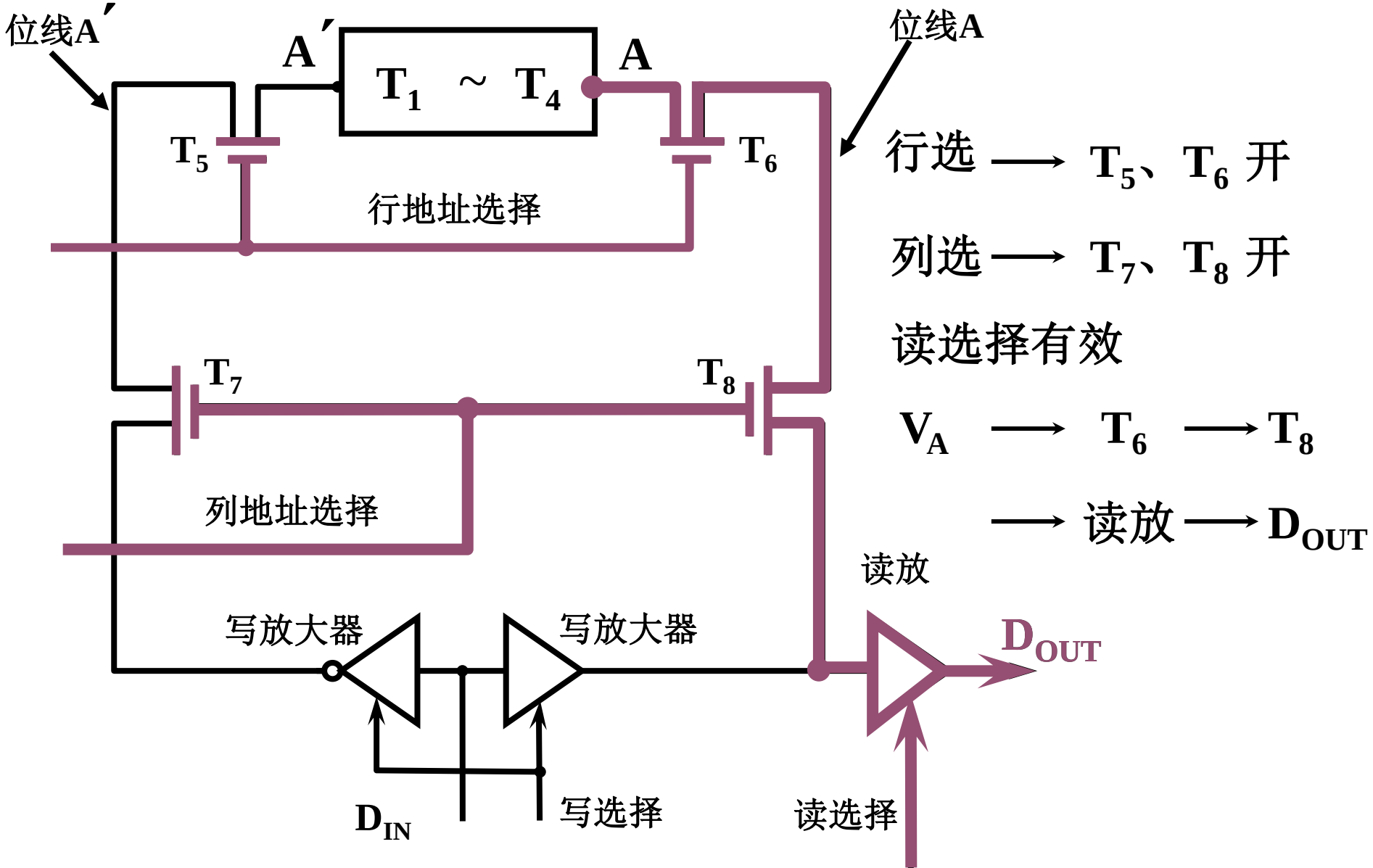
T_7、T_8 列开关

T_7、T_8 一列共用

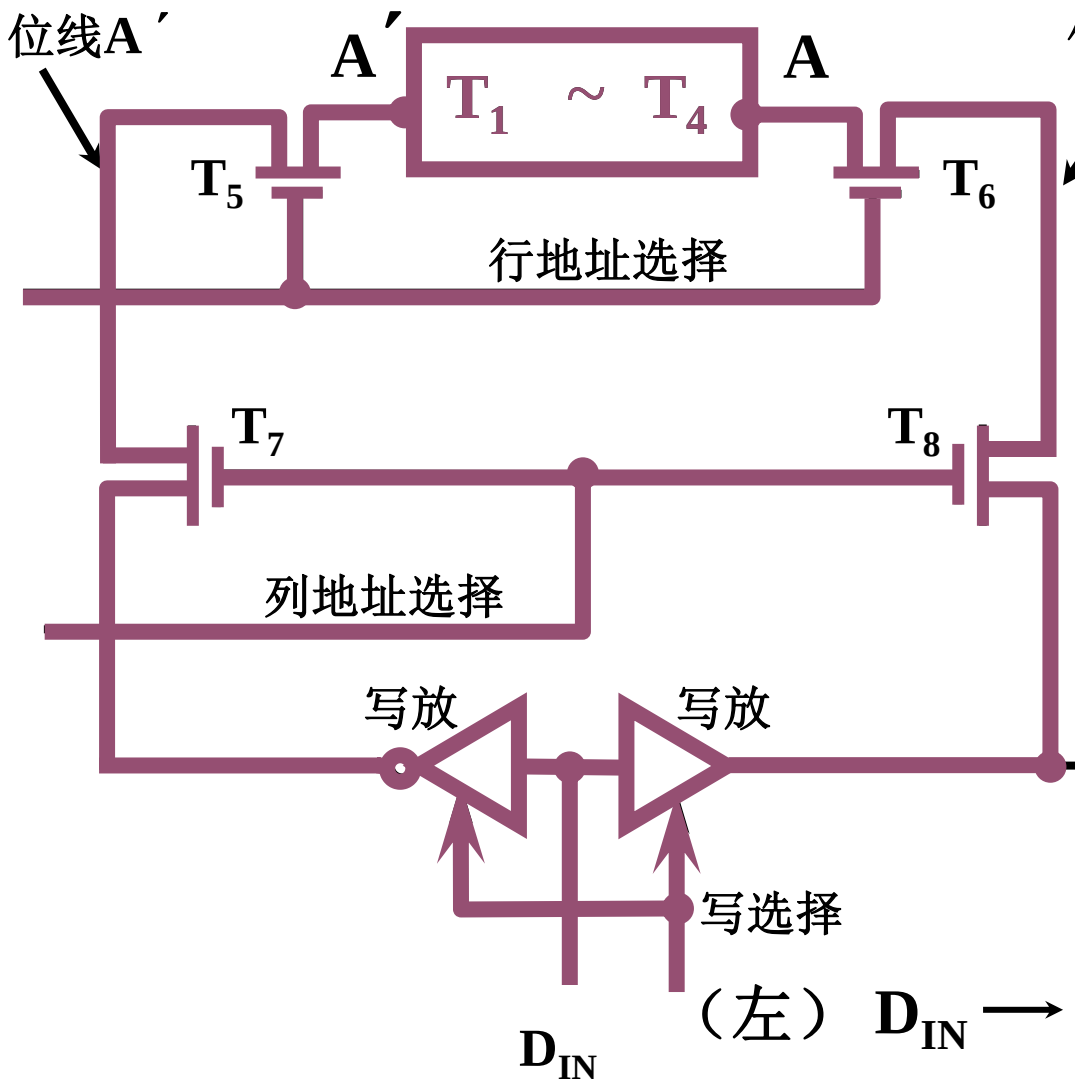
A 触发器原端

A′ 触发器非端

① 静态 RAM 基本电路的 读 操作



② 静态 RAM 基本电路的 写 操作



行选 → T_5、T_6 开

列选 → T_7、T_8 开

写选择有效

D_{IN} → 两个写放

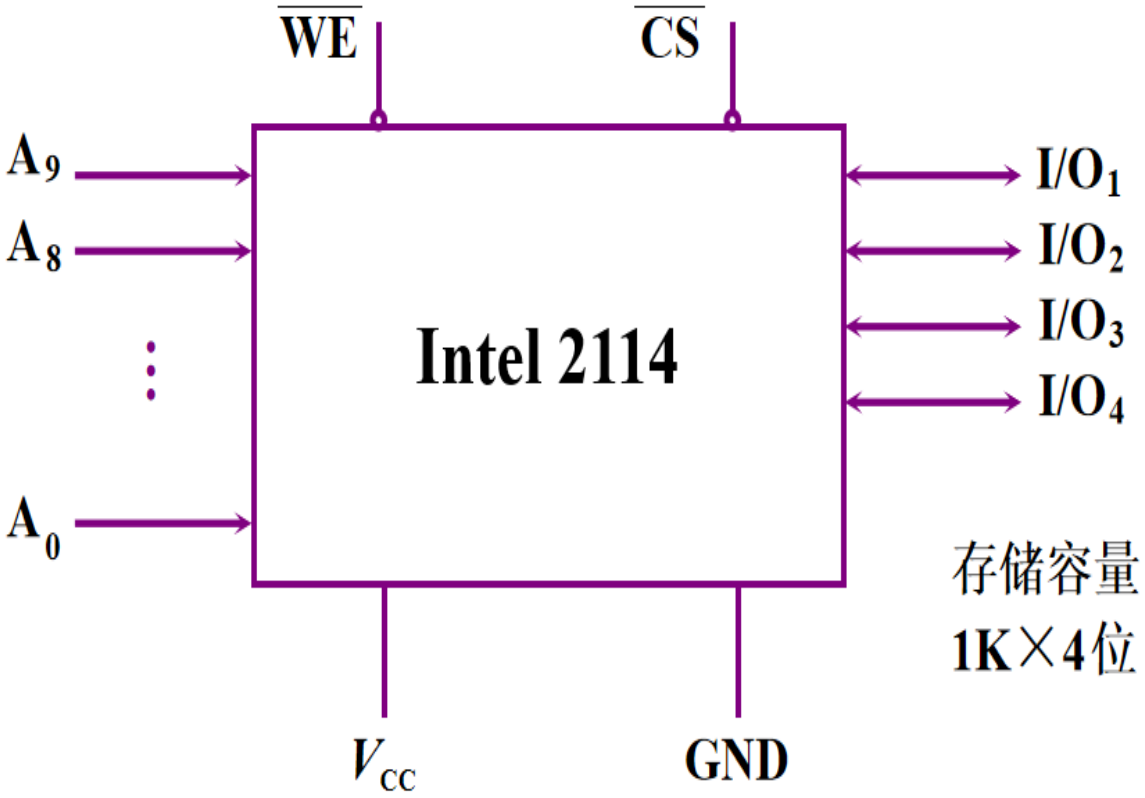
（左）D_{IN} → 反相 → T_7 → T_5 → A′

（右）D_{IN} → T_8 → T_6 → A

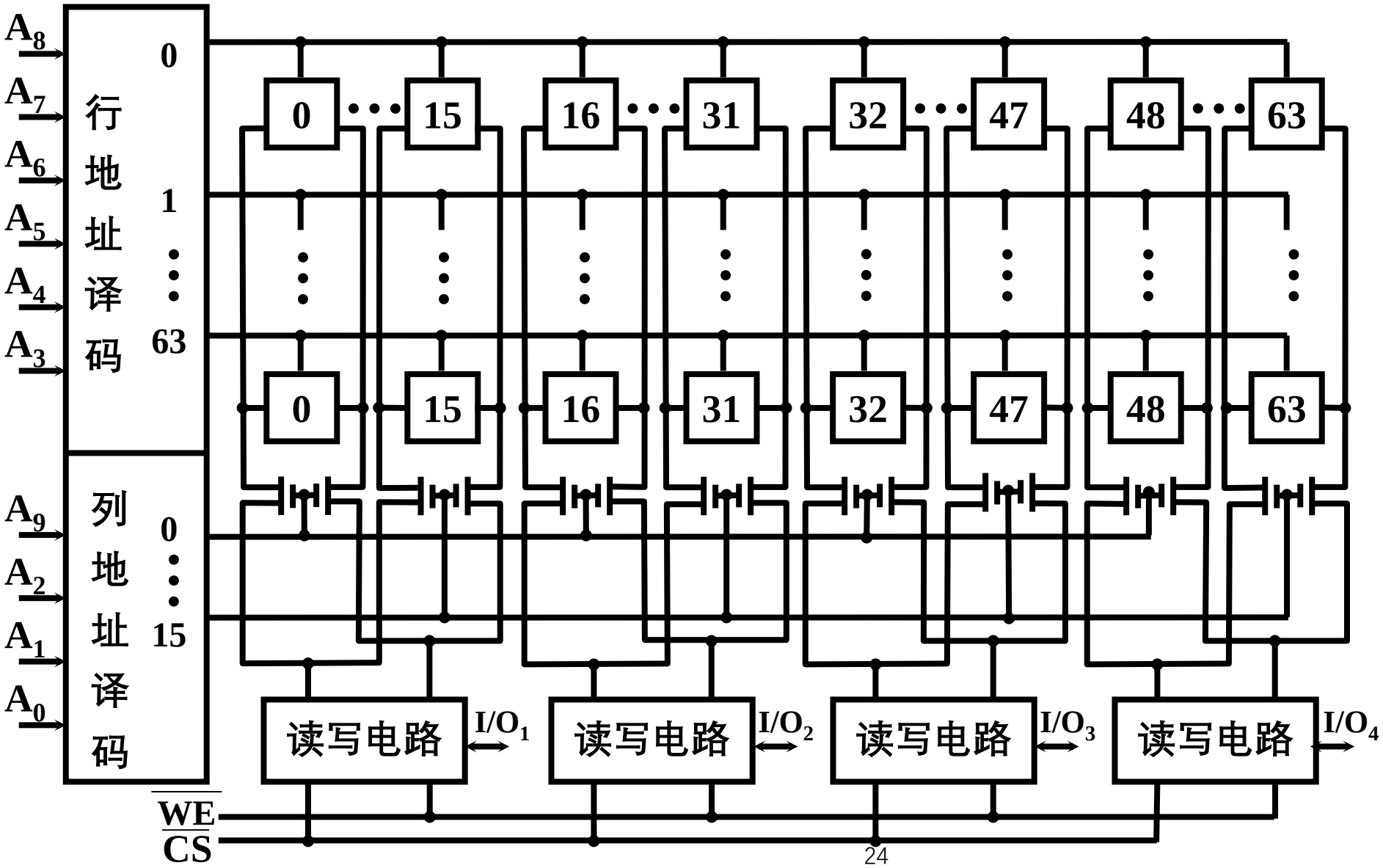
4. 2主存储器

(2) 静态RAM芯片举例

① Intel 2114 外特性

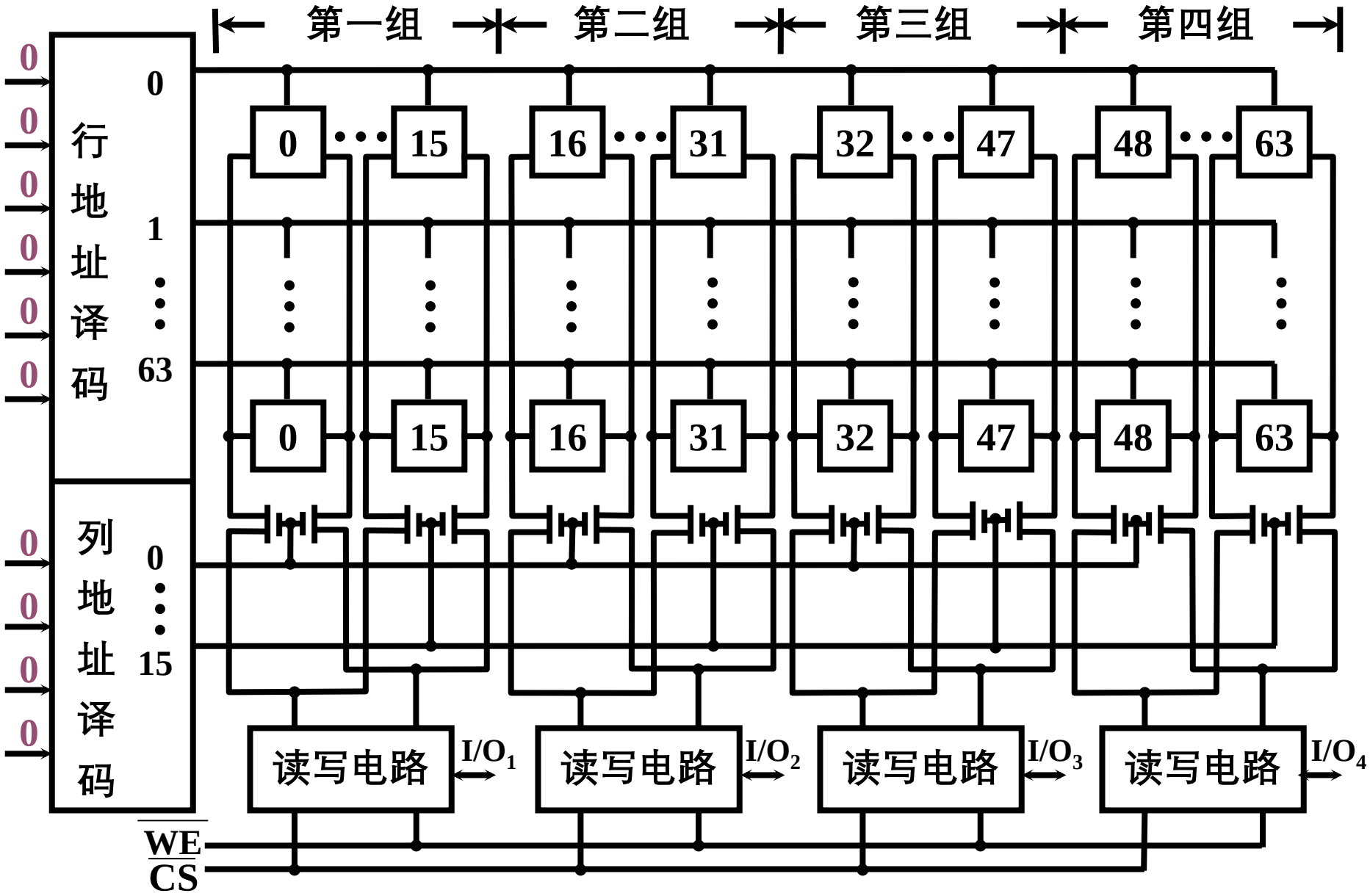


Intel 2114 RAM 矩阵 (64 × 64) 读

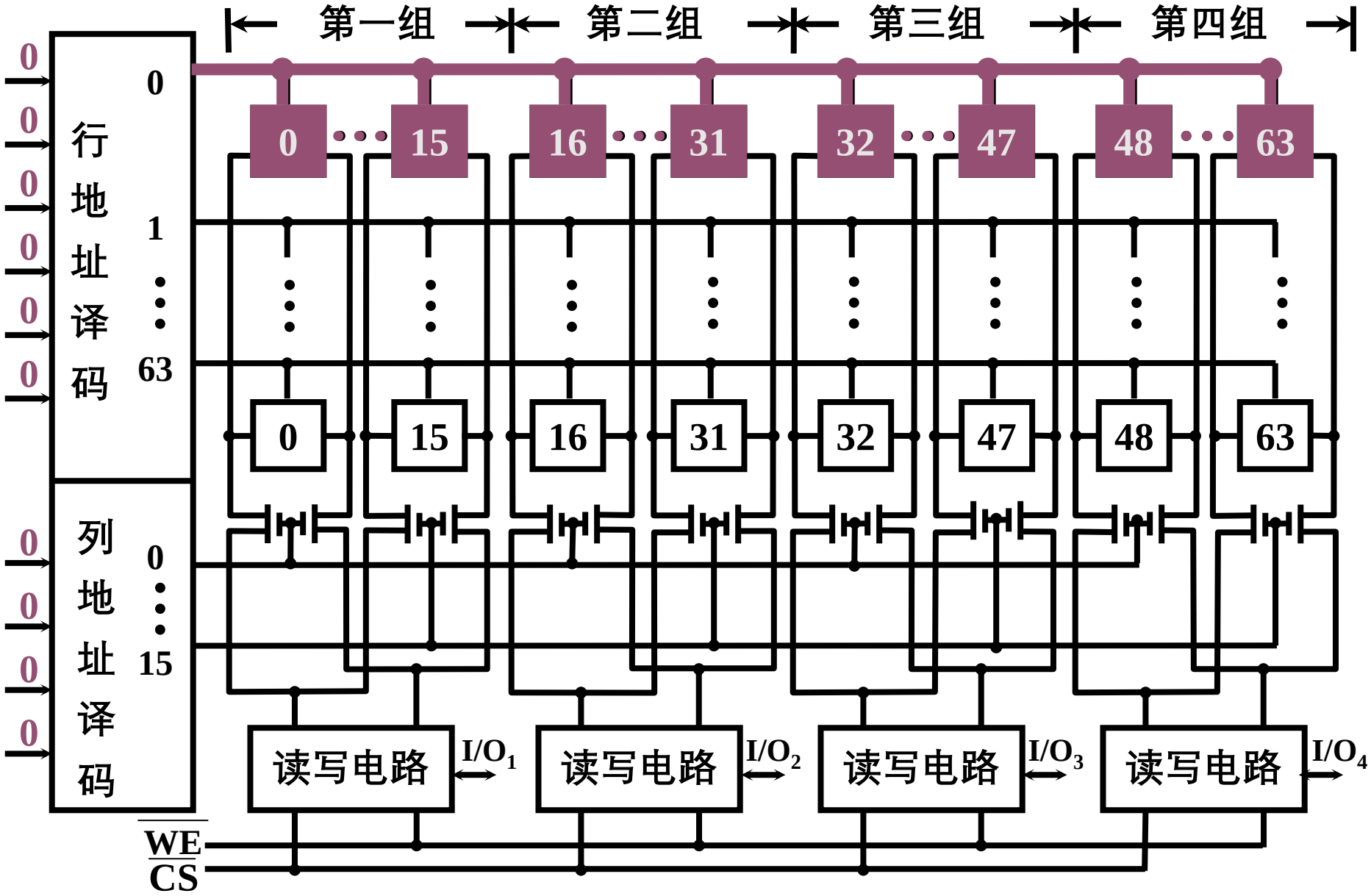


24

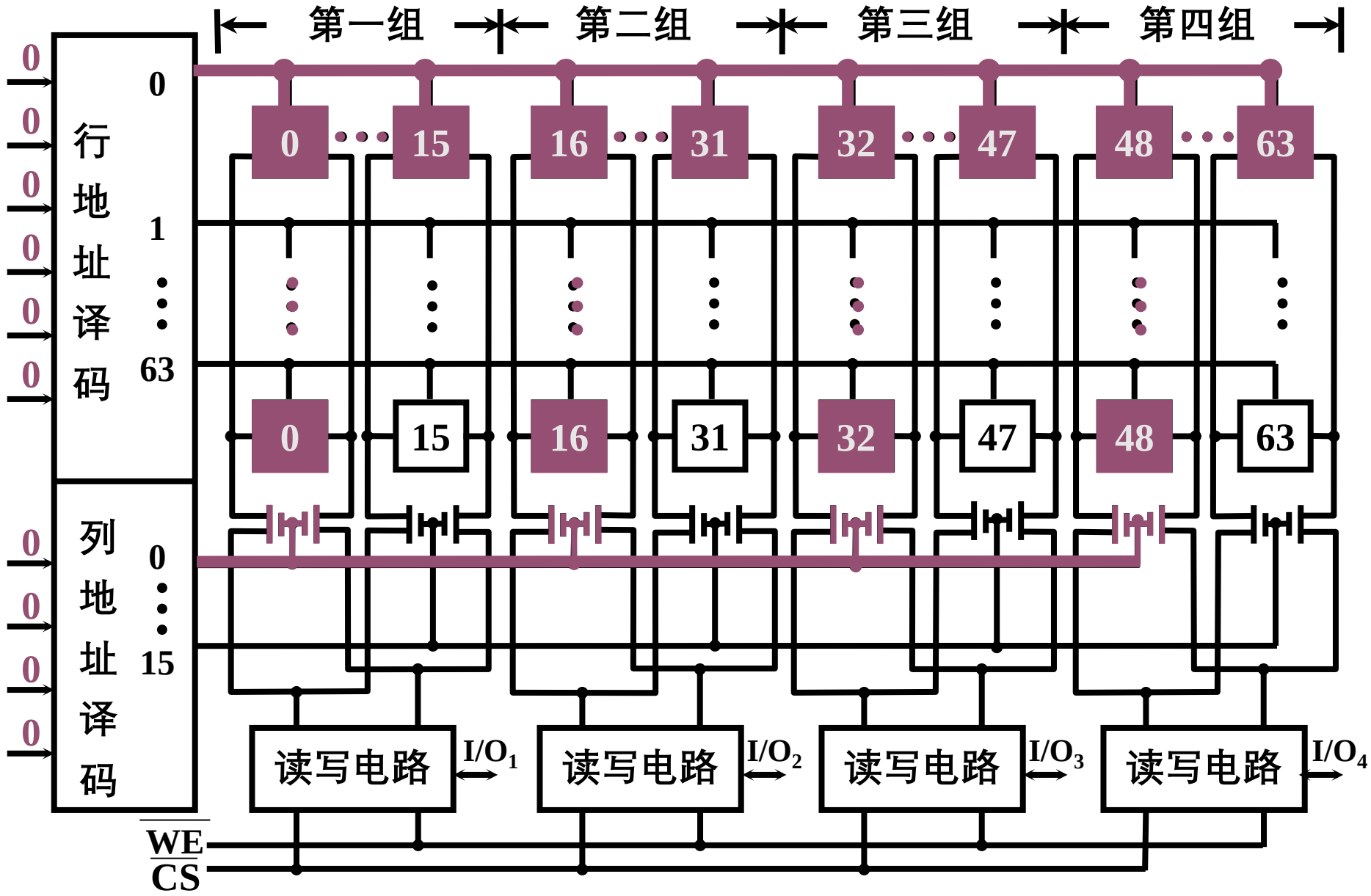
Intel 2114 RAM 矩阵 (64 × 64) 读



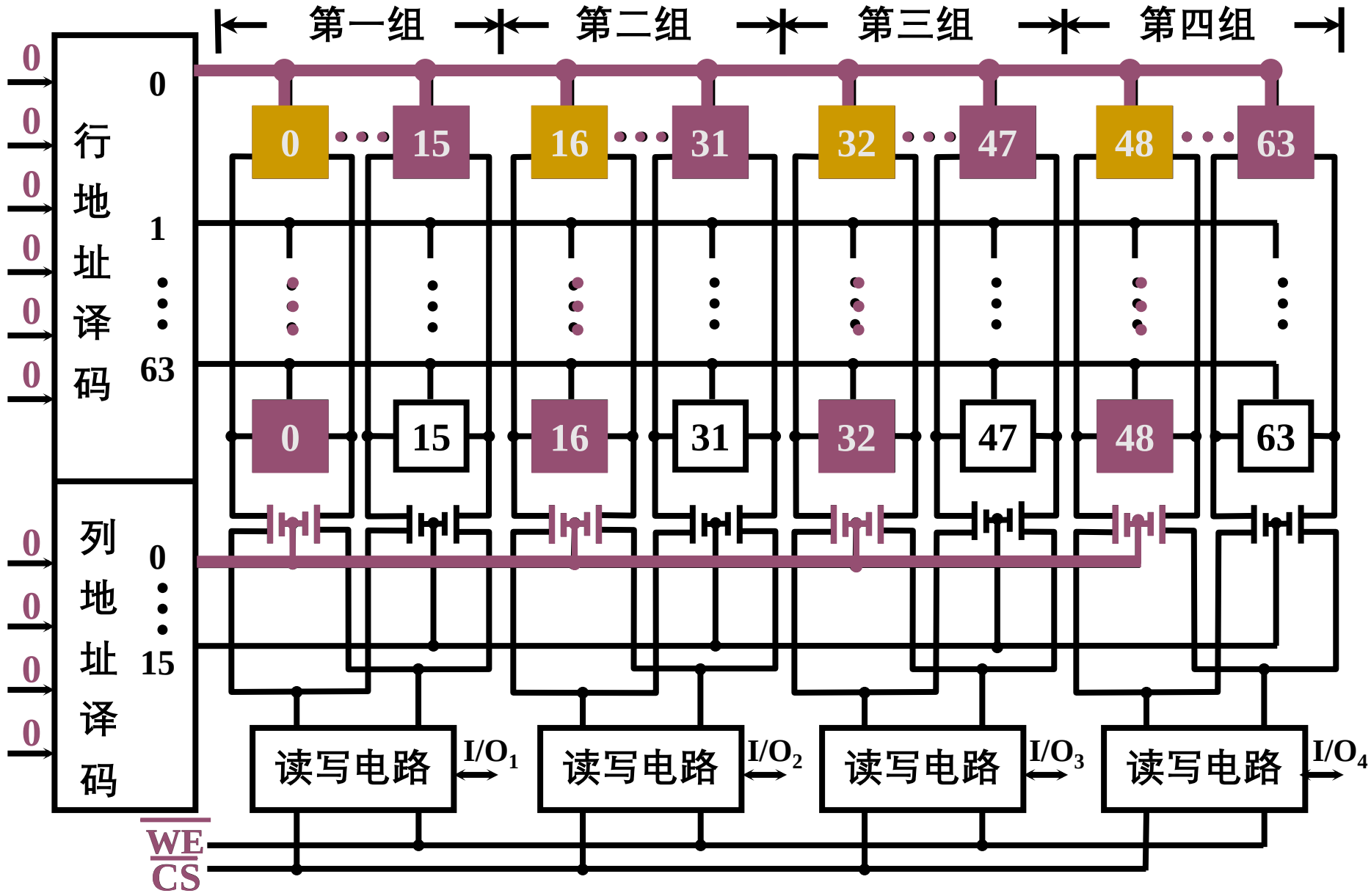
Intel 2114 RAM 矩阵 (64 × 64) 读



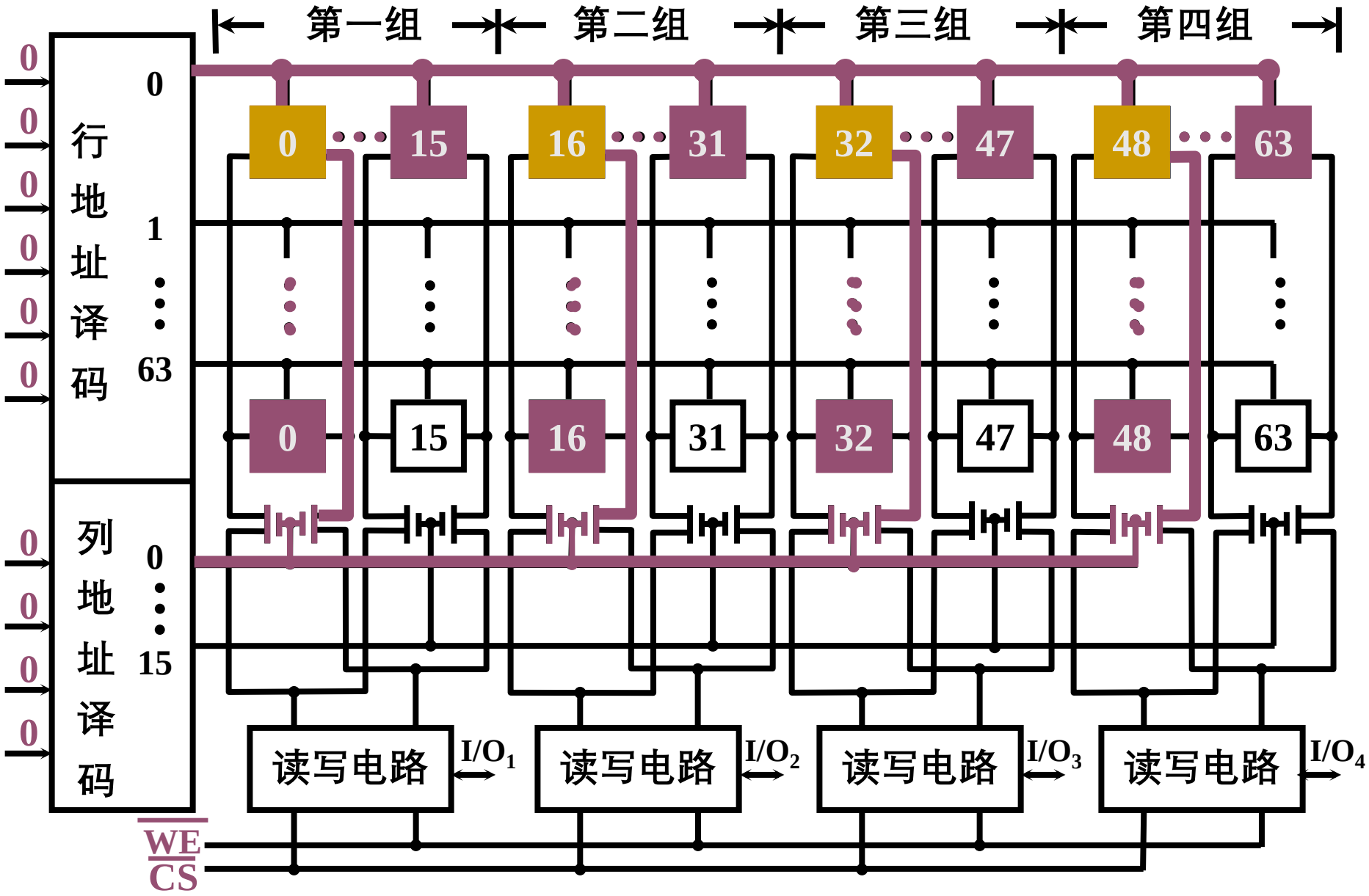
Intel 2114 RAM 矩阵 (64 × 64) 读



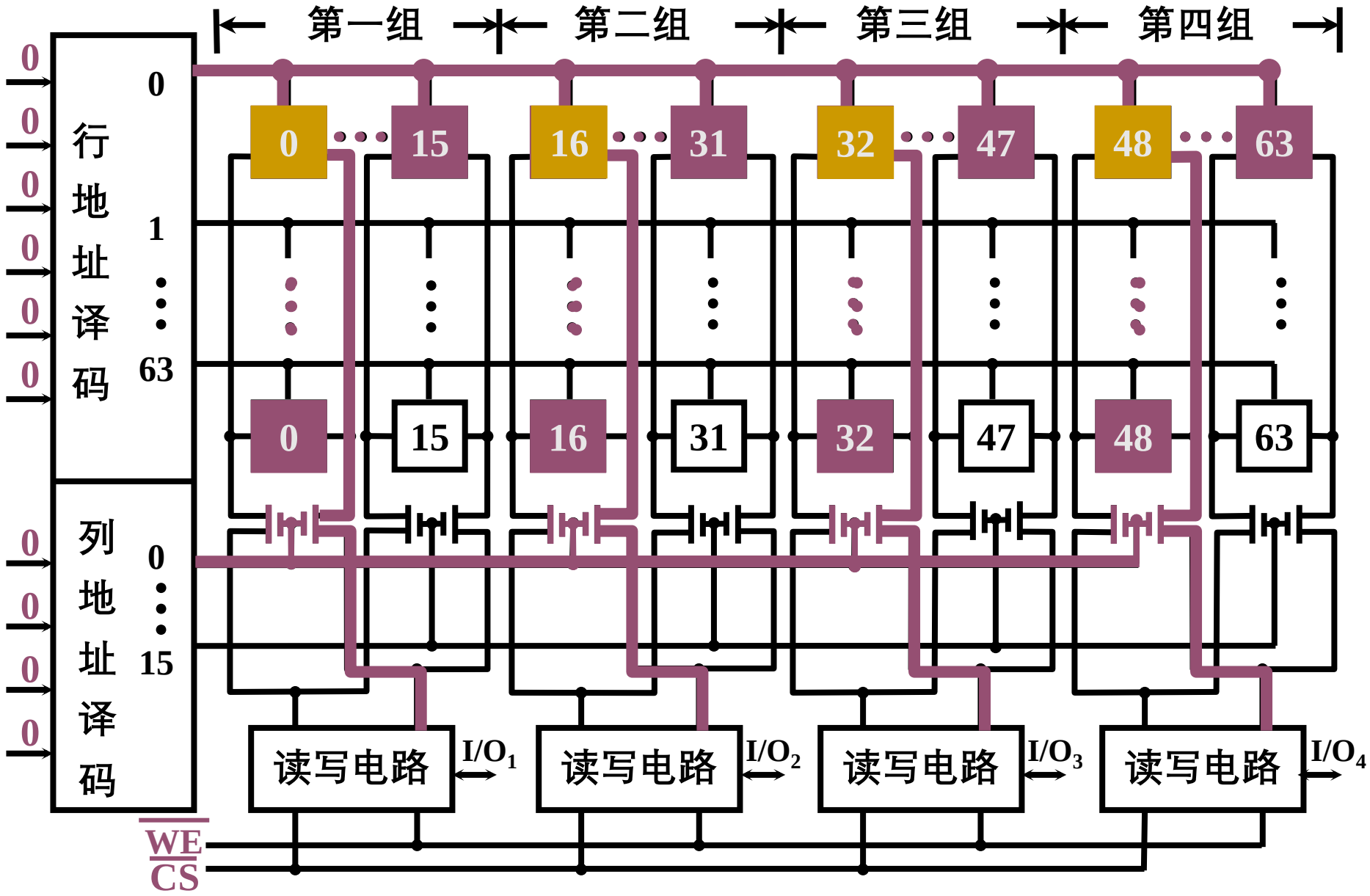
Intel 2114 RAM 矩阵 (64 × 64) 读



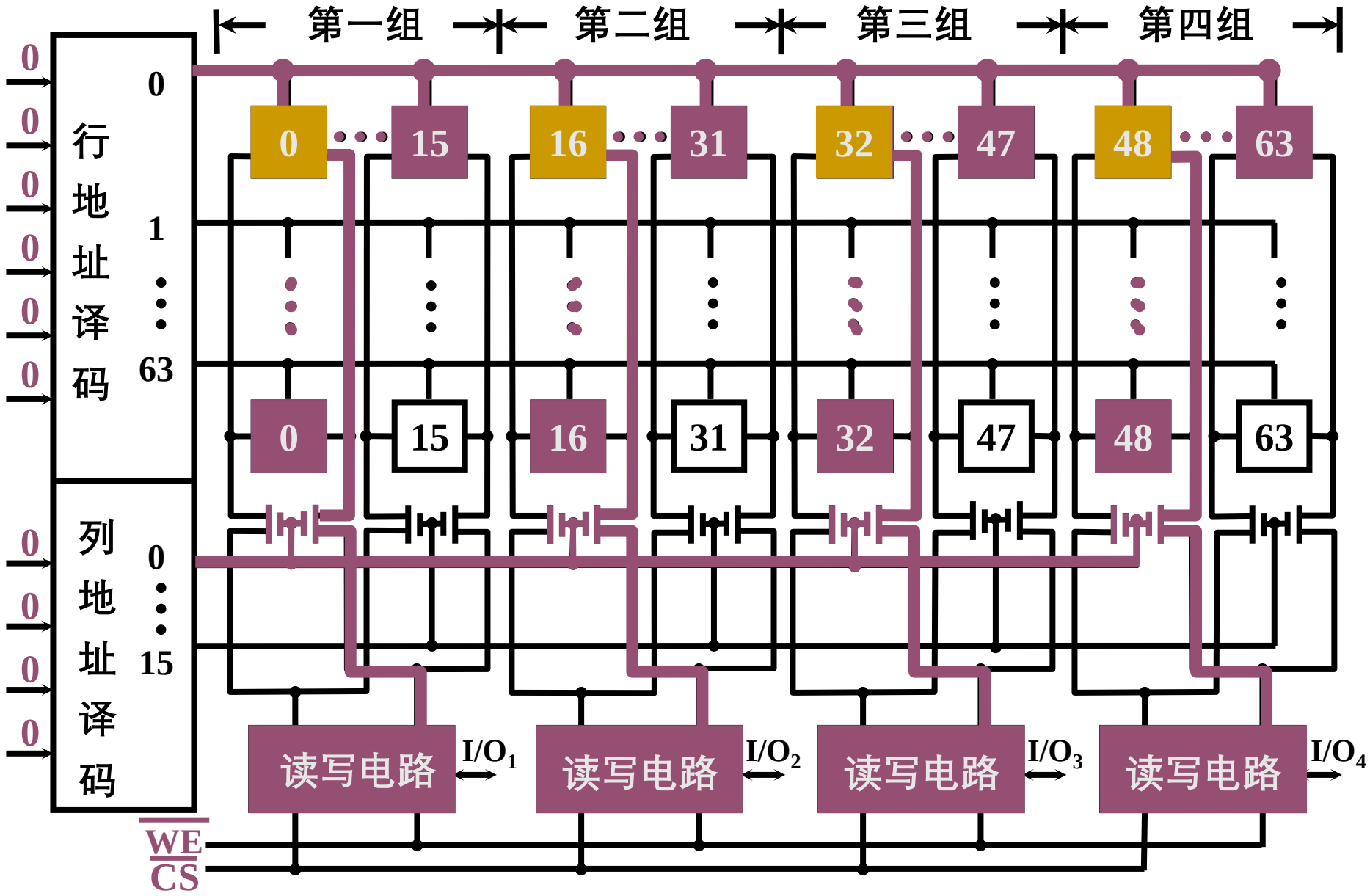
Intel 2114 RAM 矩阵 (64 × 64) 读



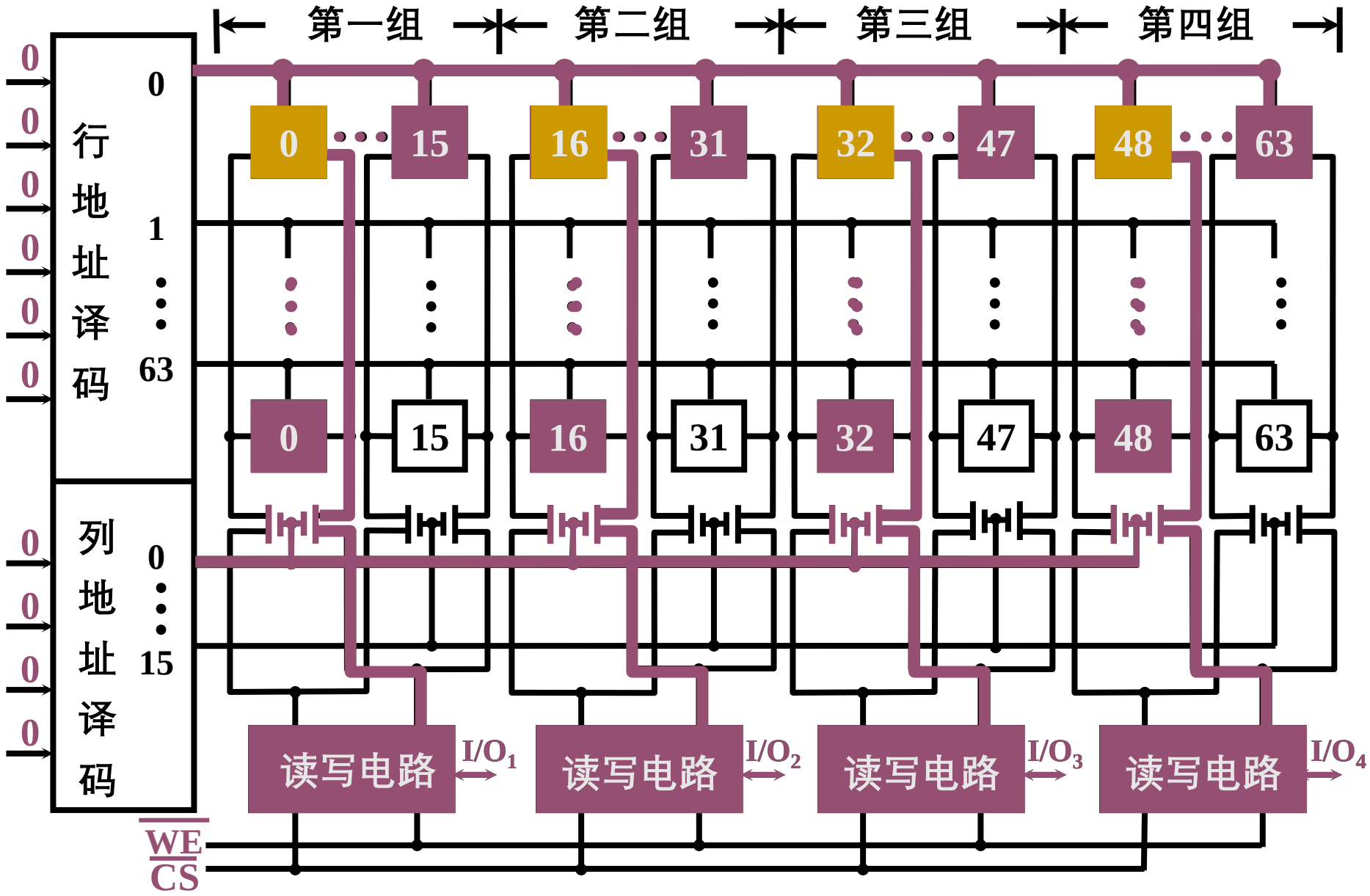
Intel 2114 RAM 矩阵 (64 × 64) 读



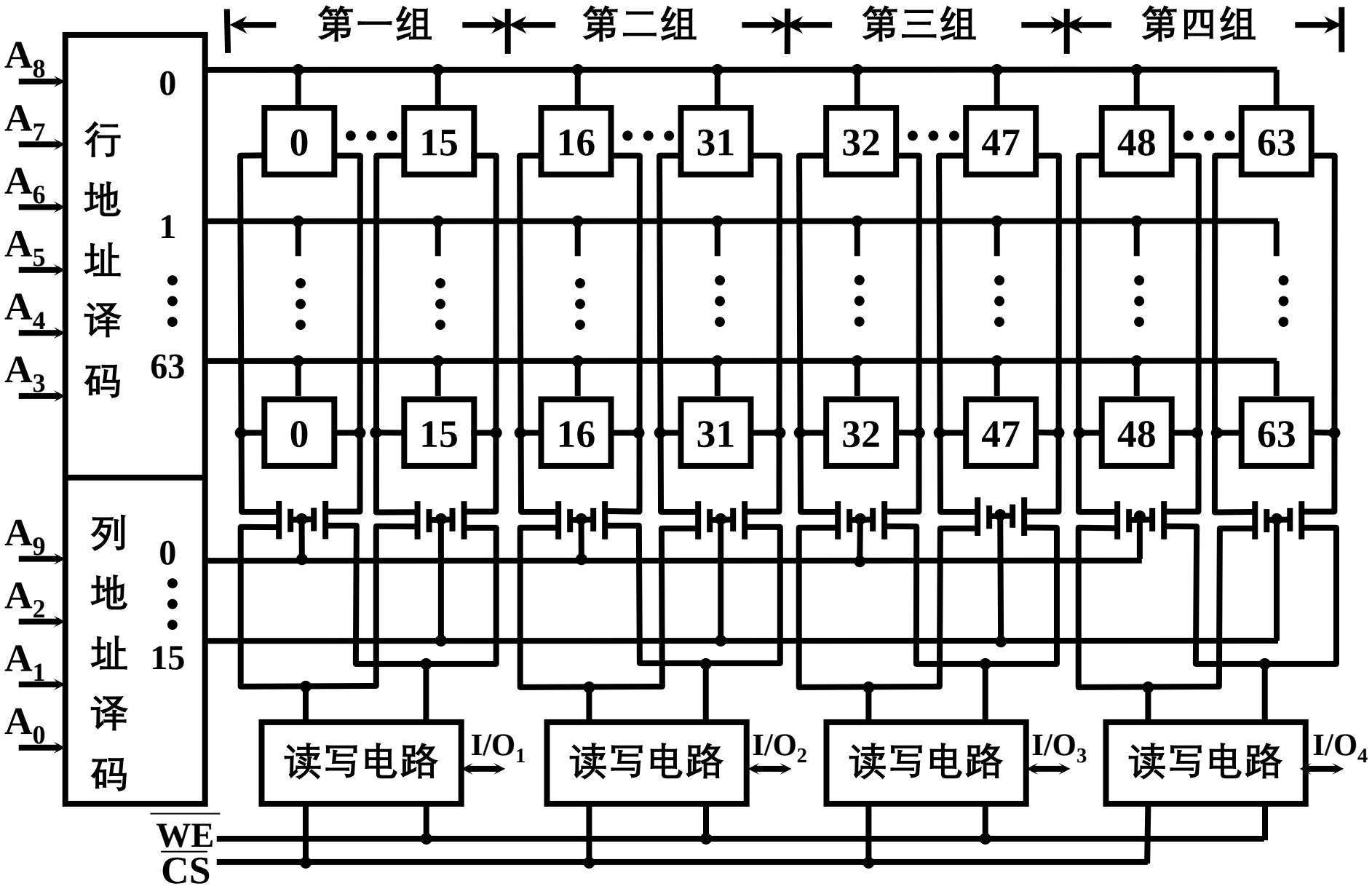
Intel 2114 RAM 矩阵 (64 × 64) 读



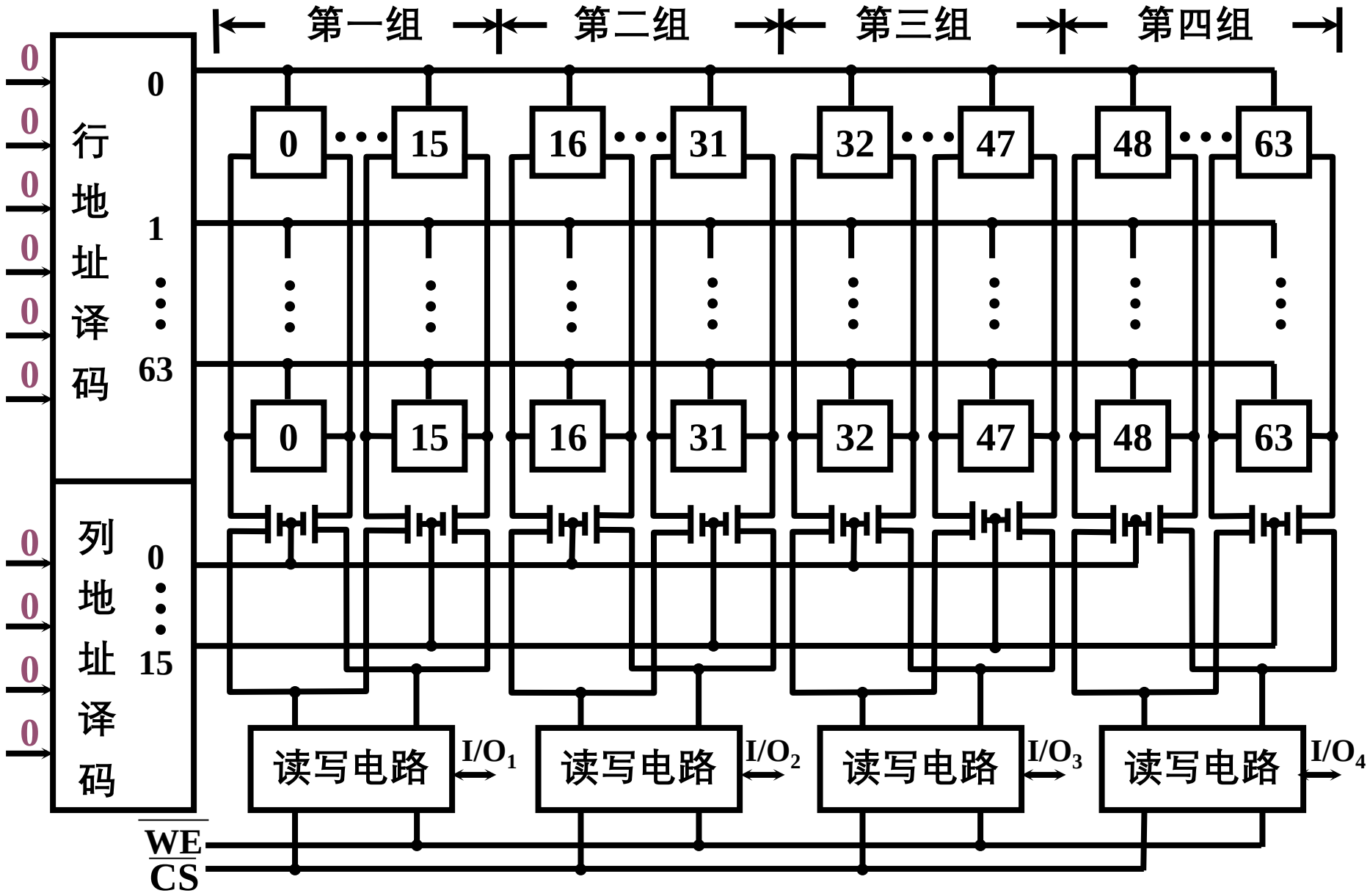
Intel 2114 RAM 矩阵 (64 × 64) 读



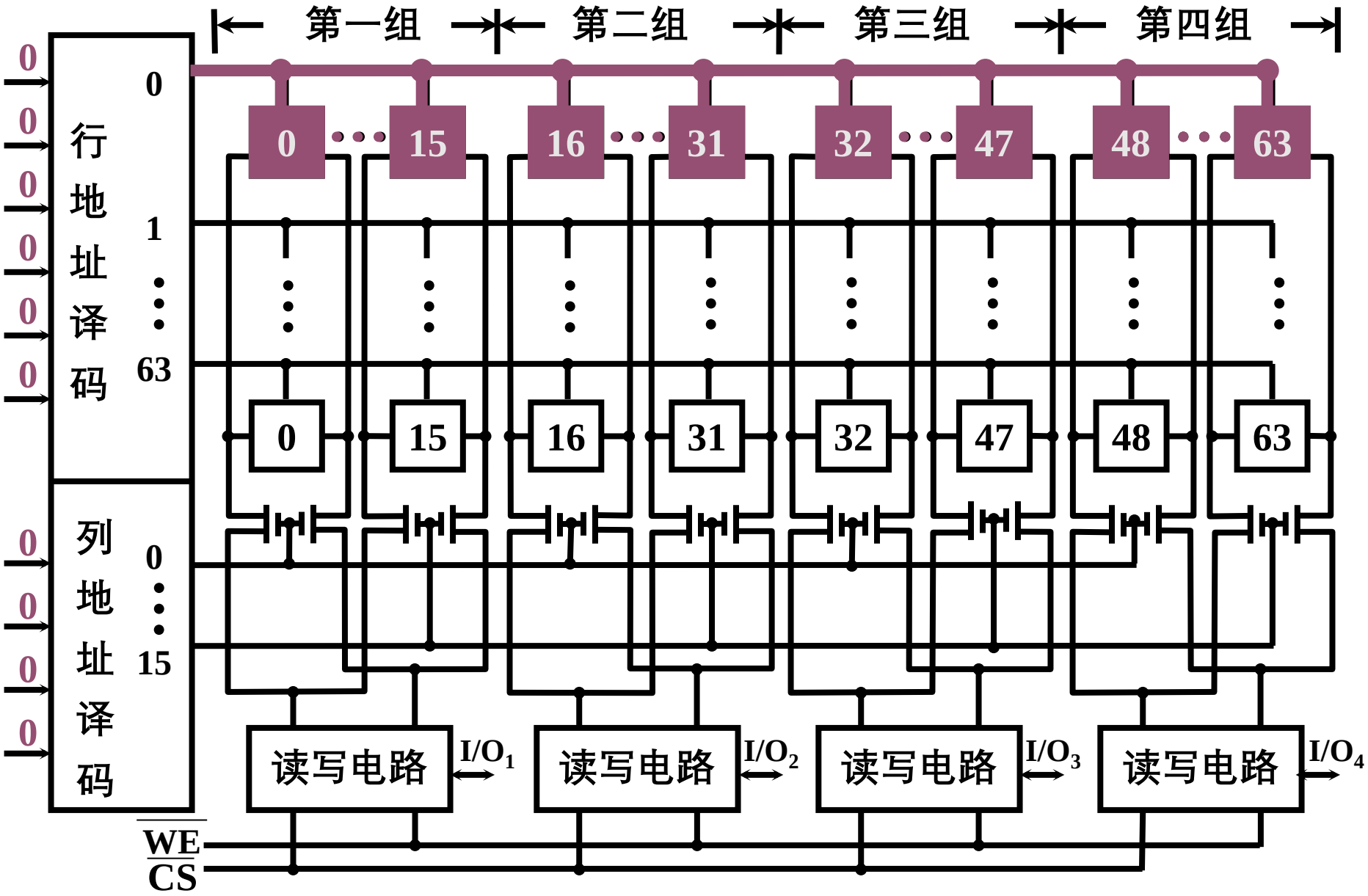
Intel 2114 RAM 矩阵 (64 × 64) 写



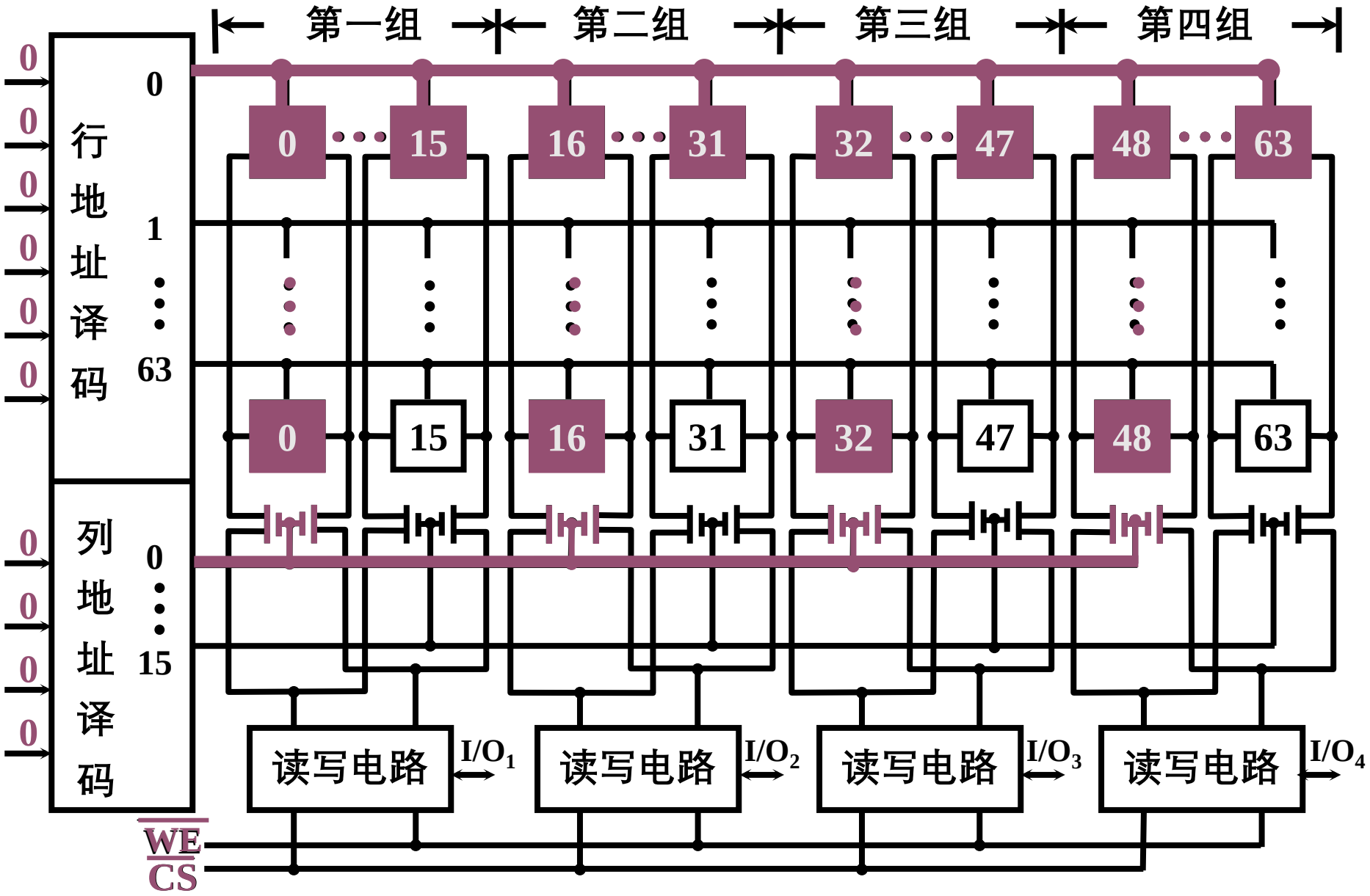
Intel 2114 RAM 矩阵 (64 × 64) 写



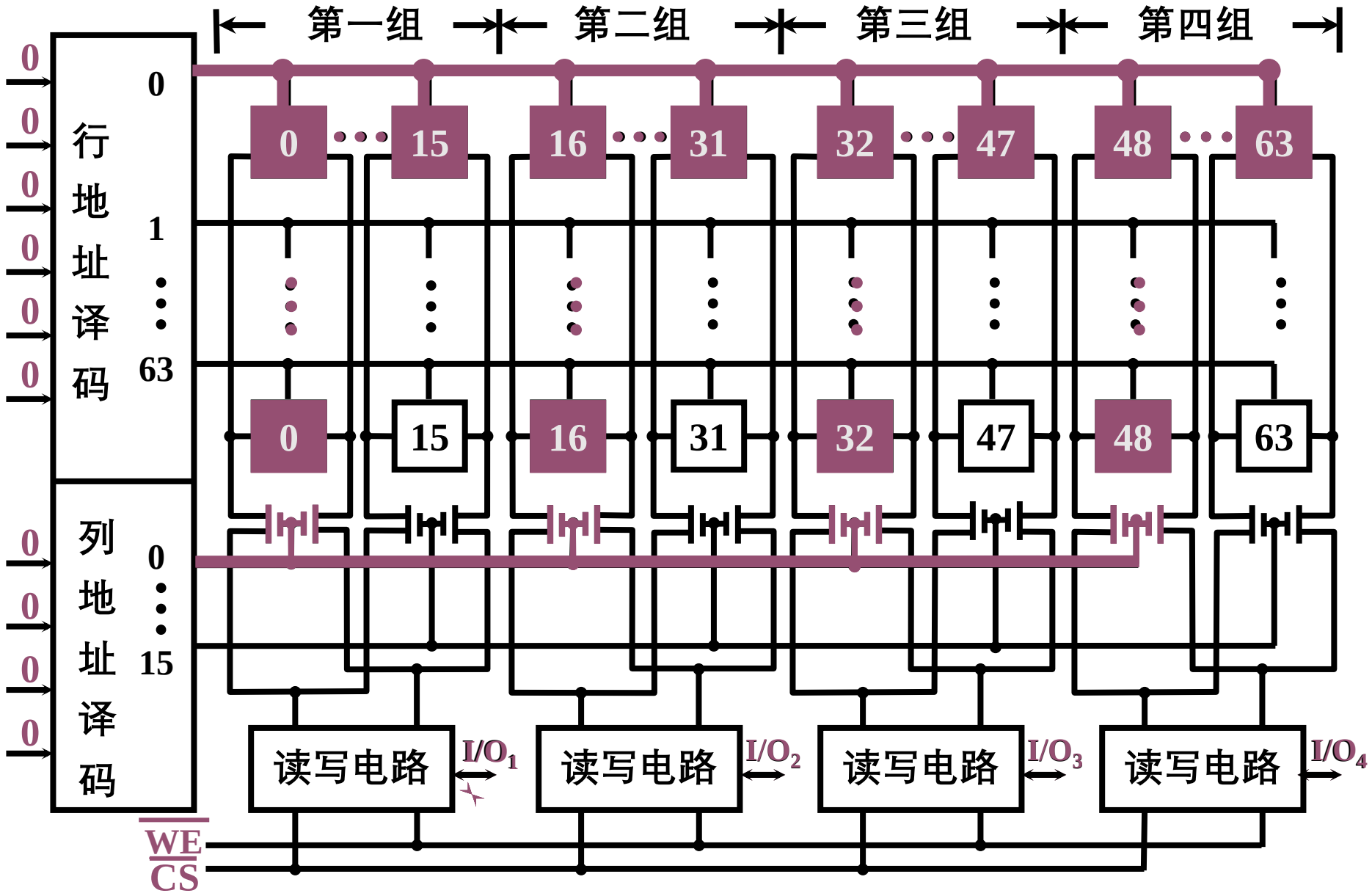
Intel 2114 RAM 矩阵 (64 × 64) 写



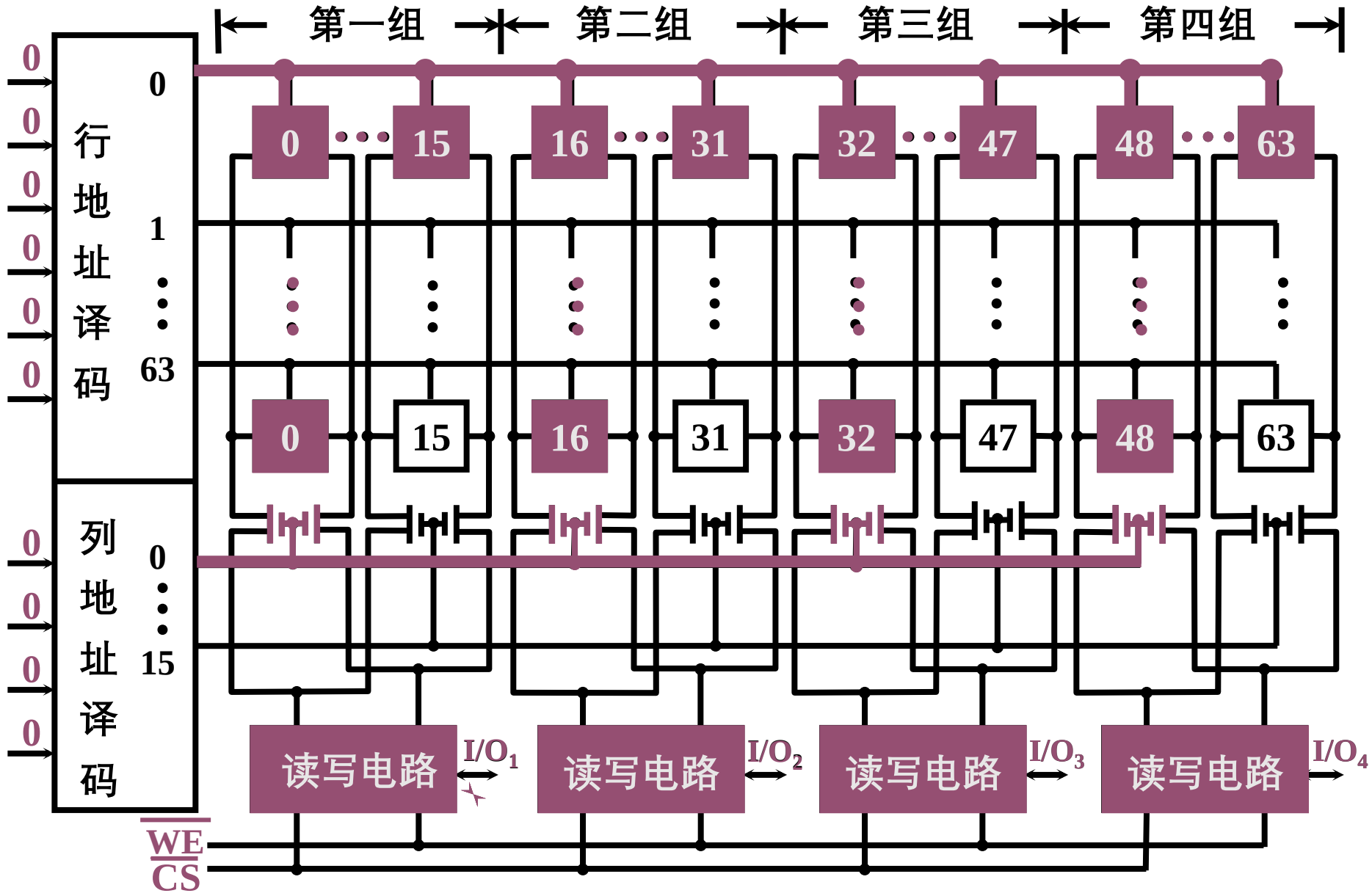
Intel 2114 RAM 矩阵 (64 × 64) 写



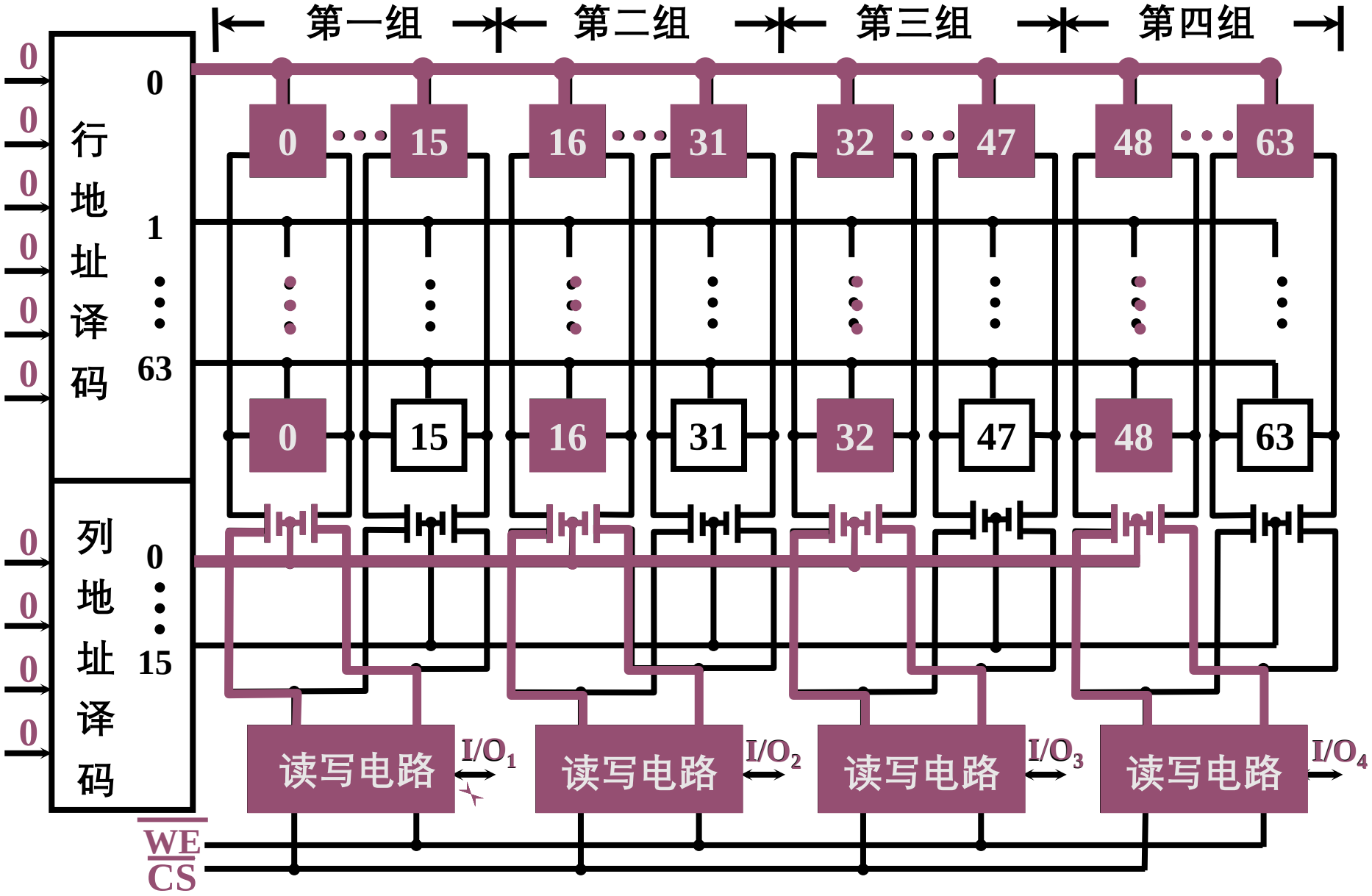
Intel 2114 RAM 矩阵 (64 × 64) 写



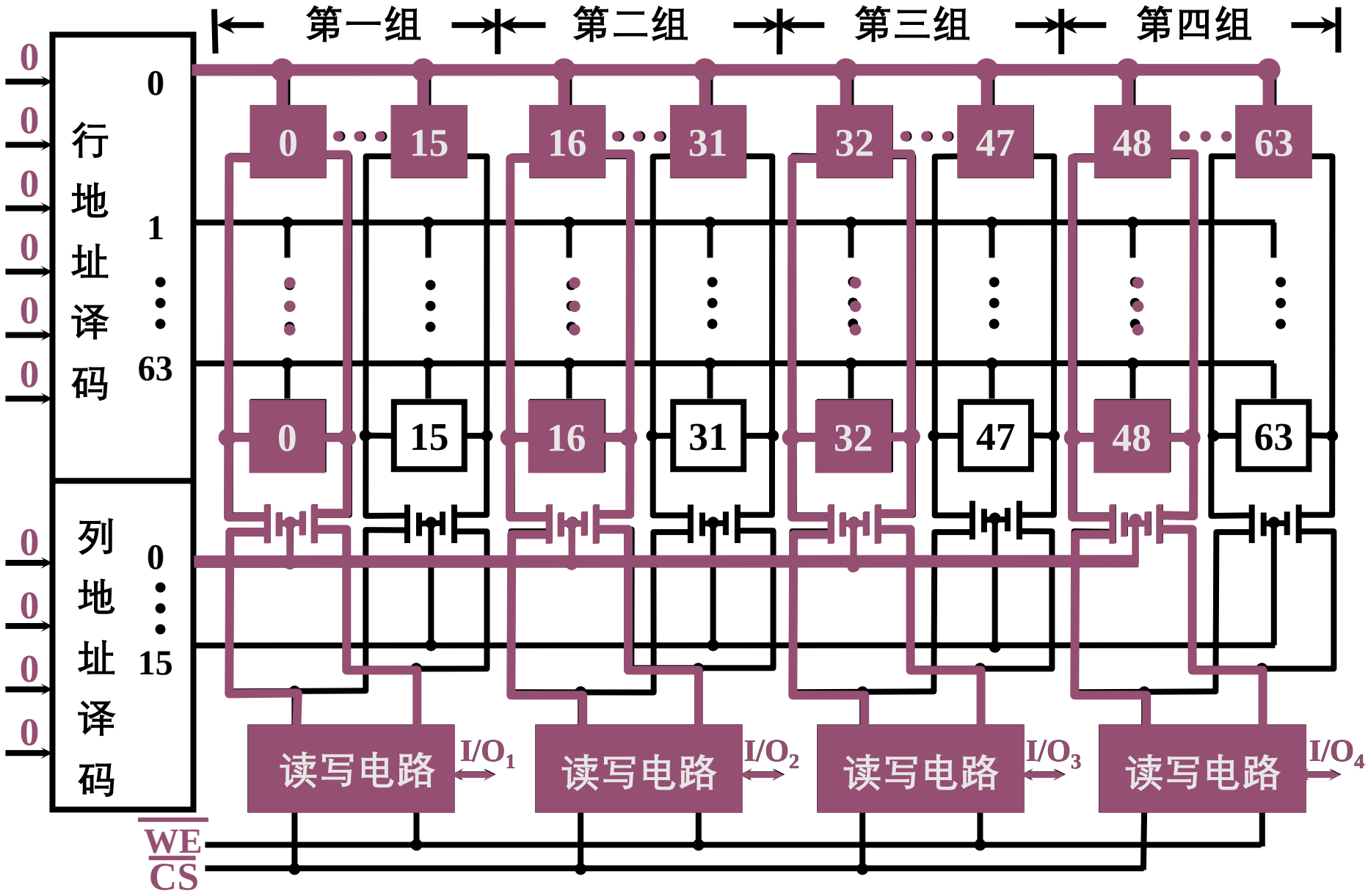
Intel 2114 RAM 矩阵 (64 × 64) 写



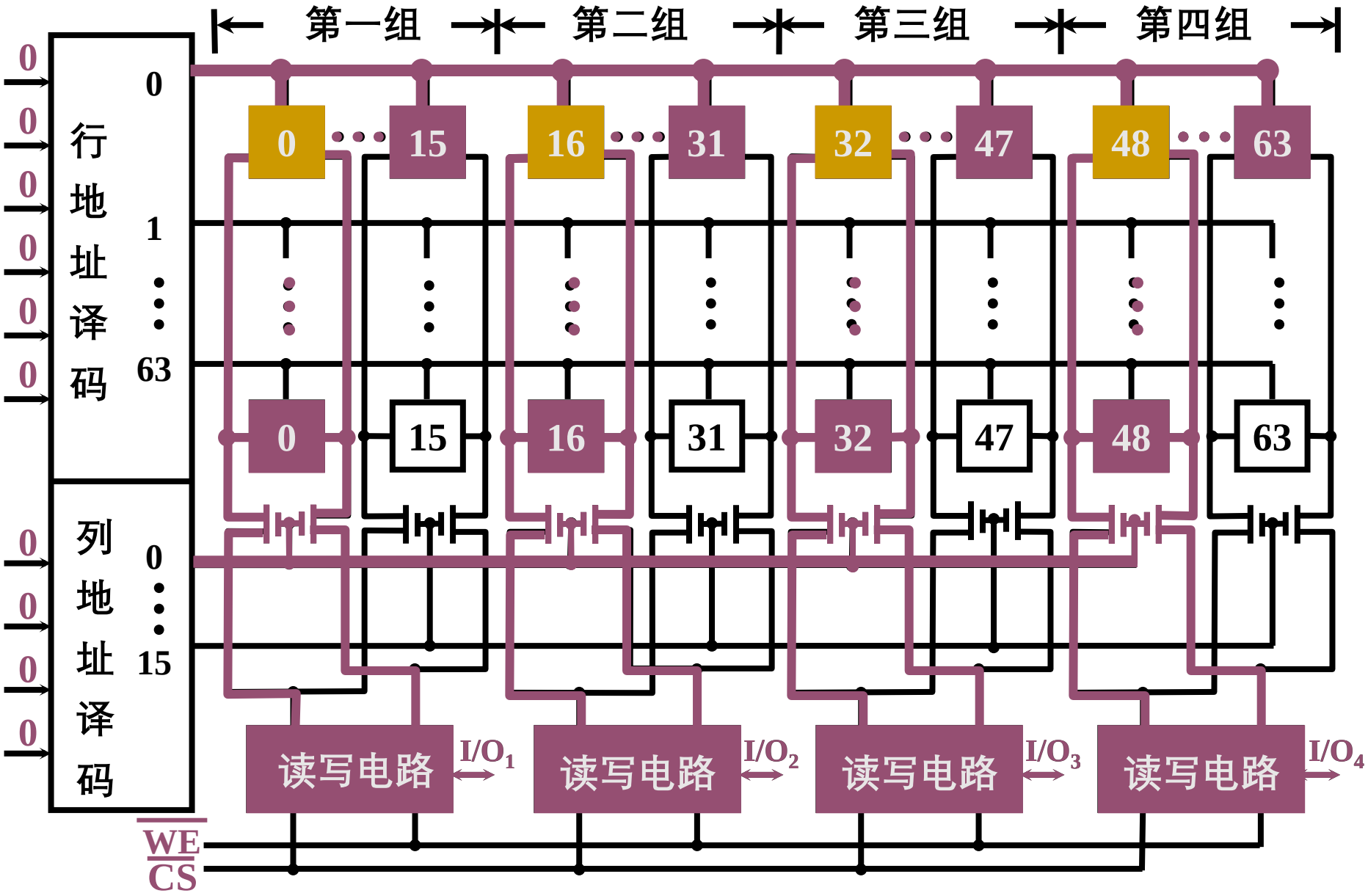
Intel 2114 RAM 矩阵 (64 × 64) 写



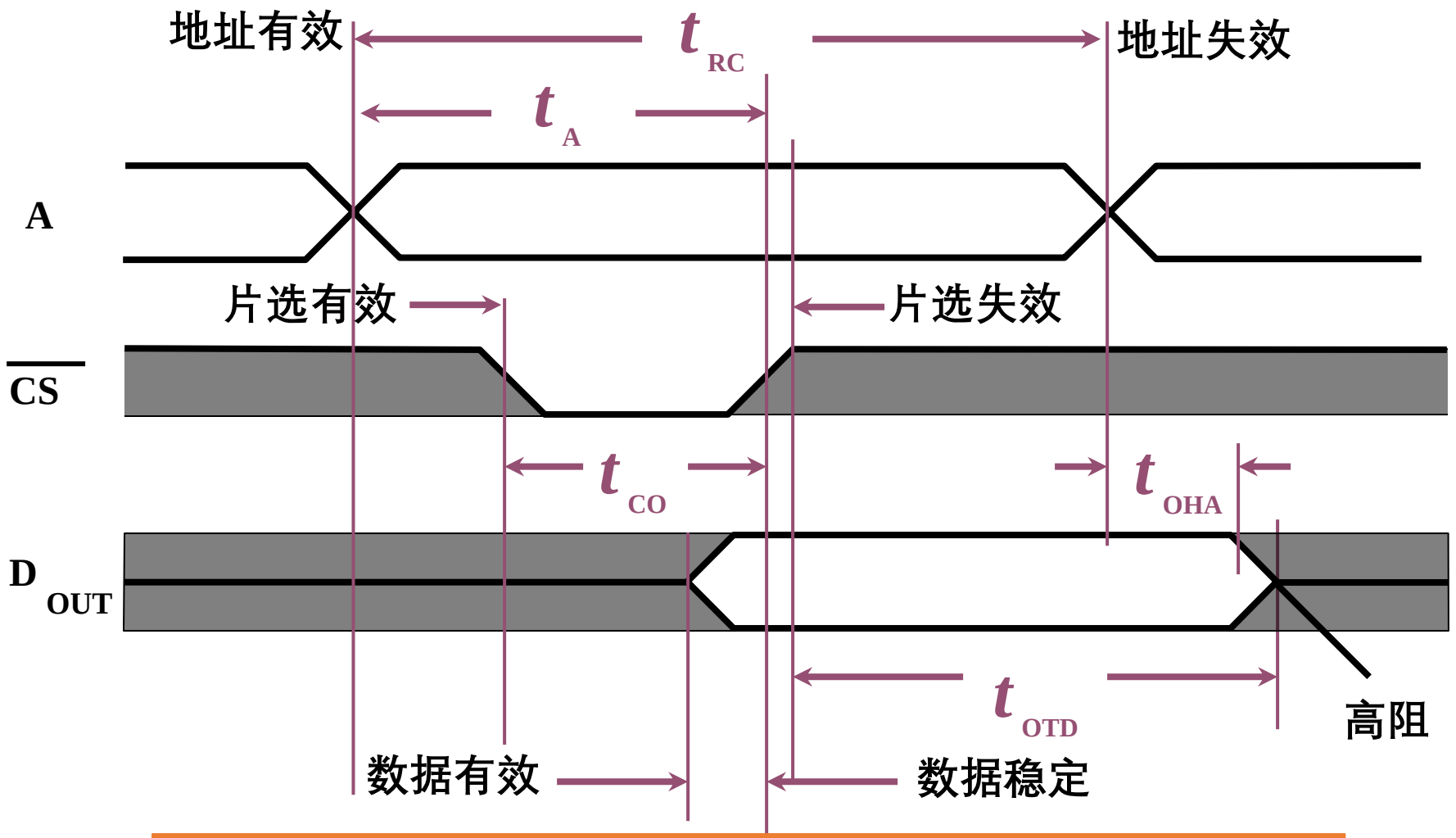
Intel 2114 RAM 矩阵 (64 × 64) 写



Intel 2114 RAM 矩阵 (64 × 64) 写

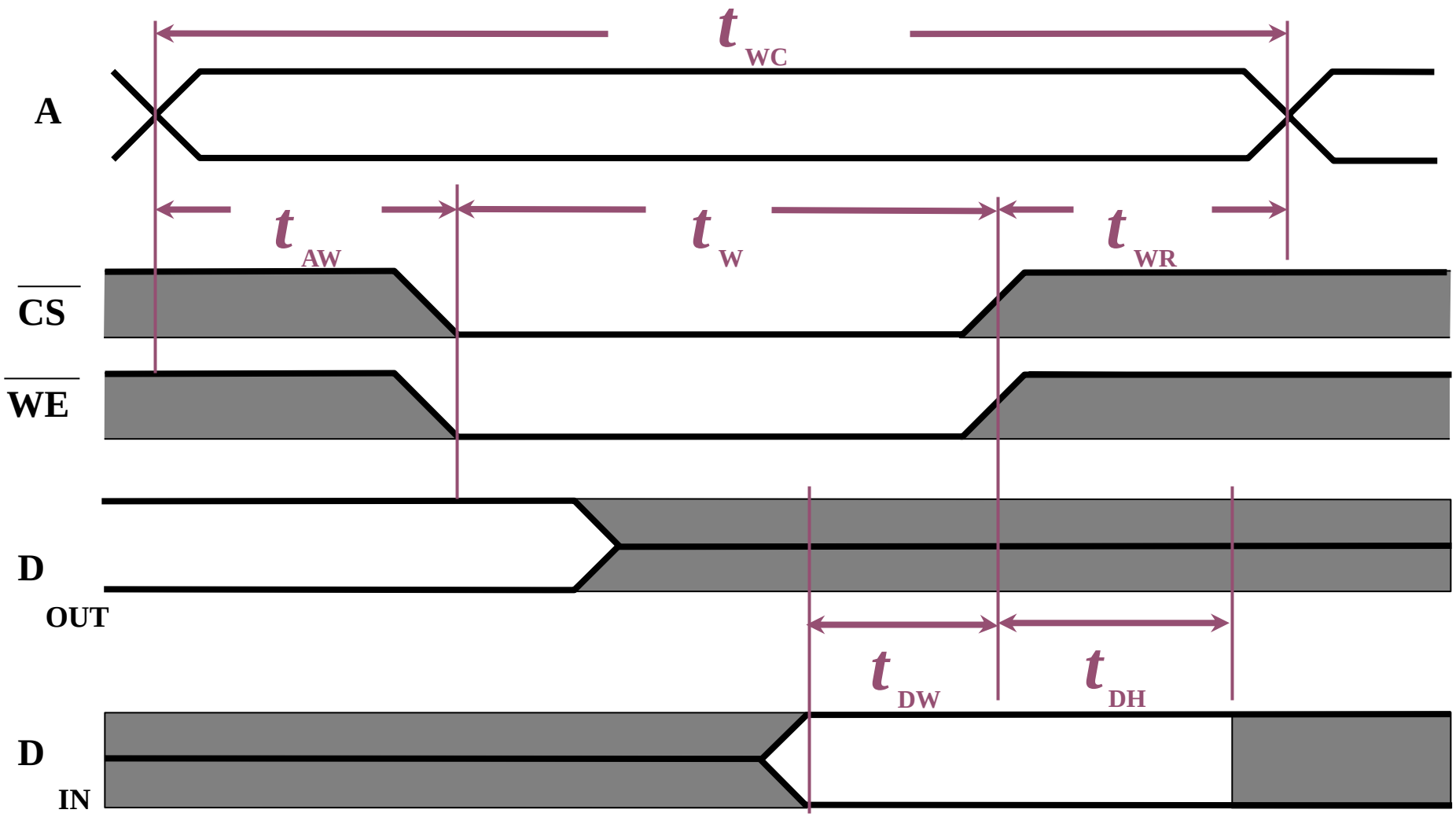


静态 RAM（2114） 读 时序



t_{OHA} 地址失效后的 数据维持时间

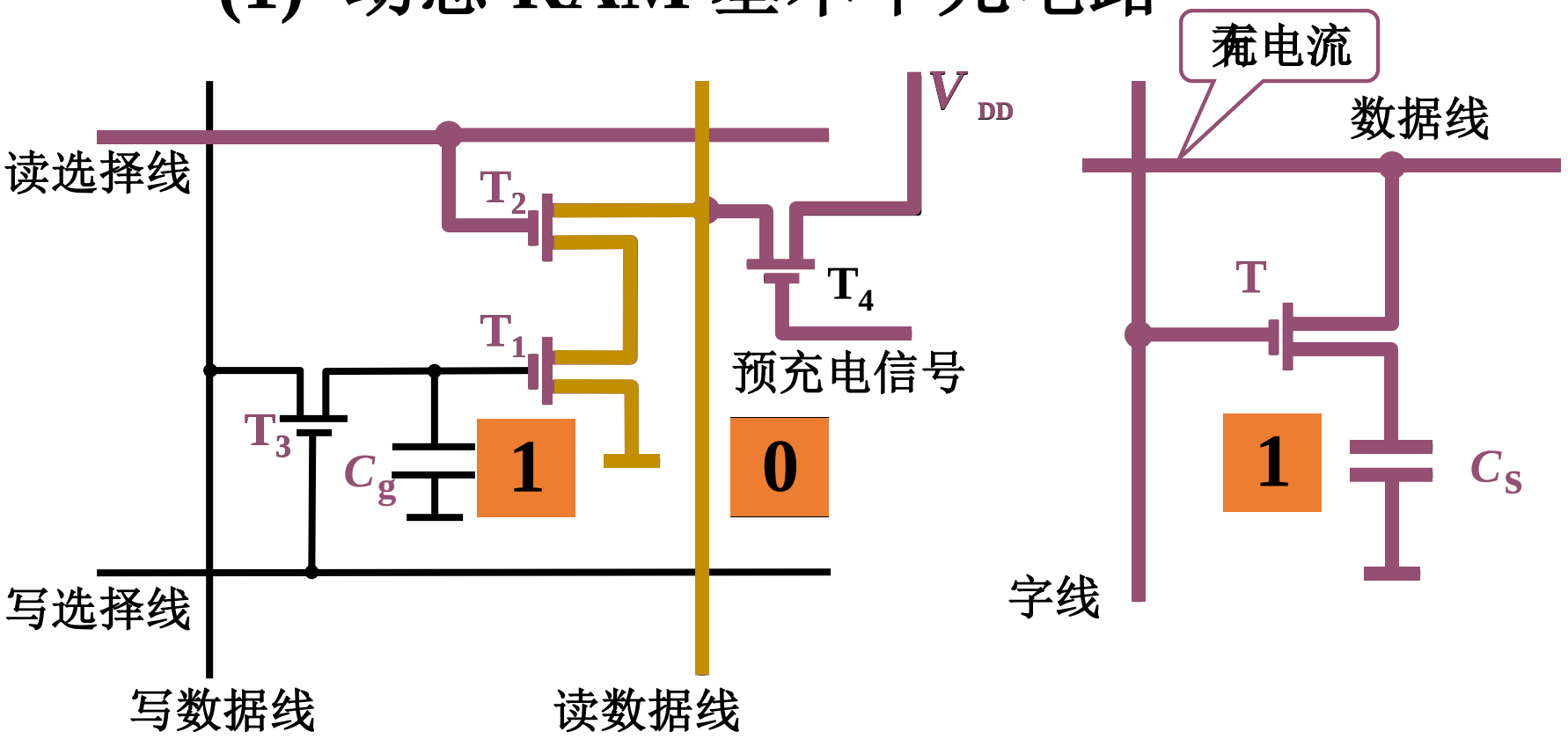
静态 RAM（2114） 写 时序



t_{DH} $\overline{WE}$失效后的数据维持时间

2. 动态 RAM (DRAM)

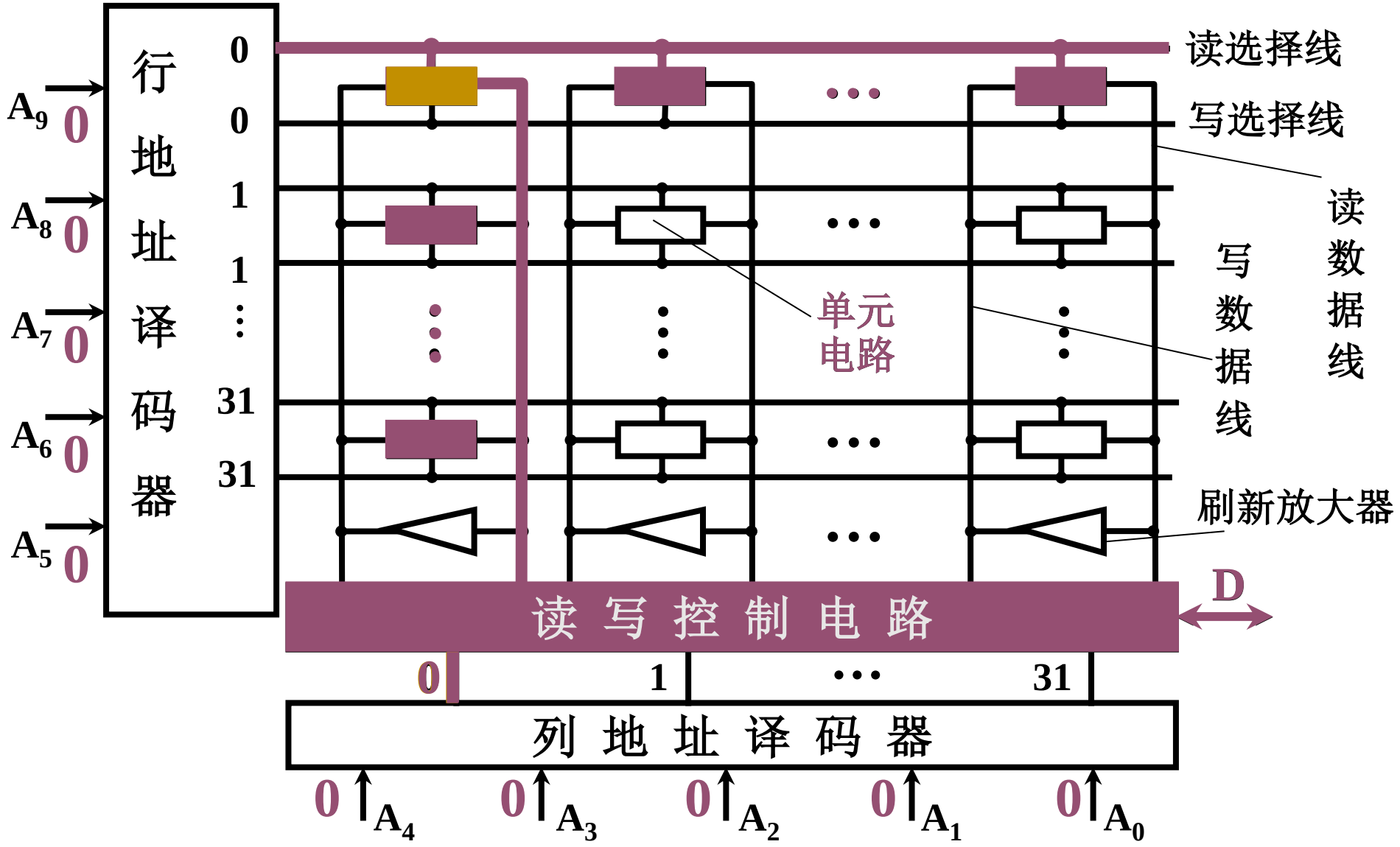
(1) 动态 RAM 基本单元电路



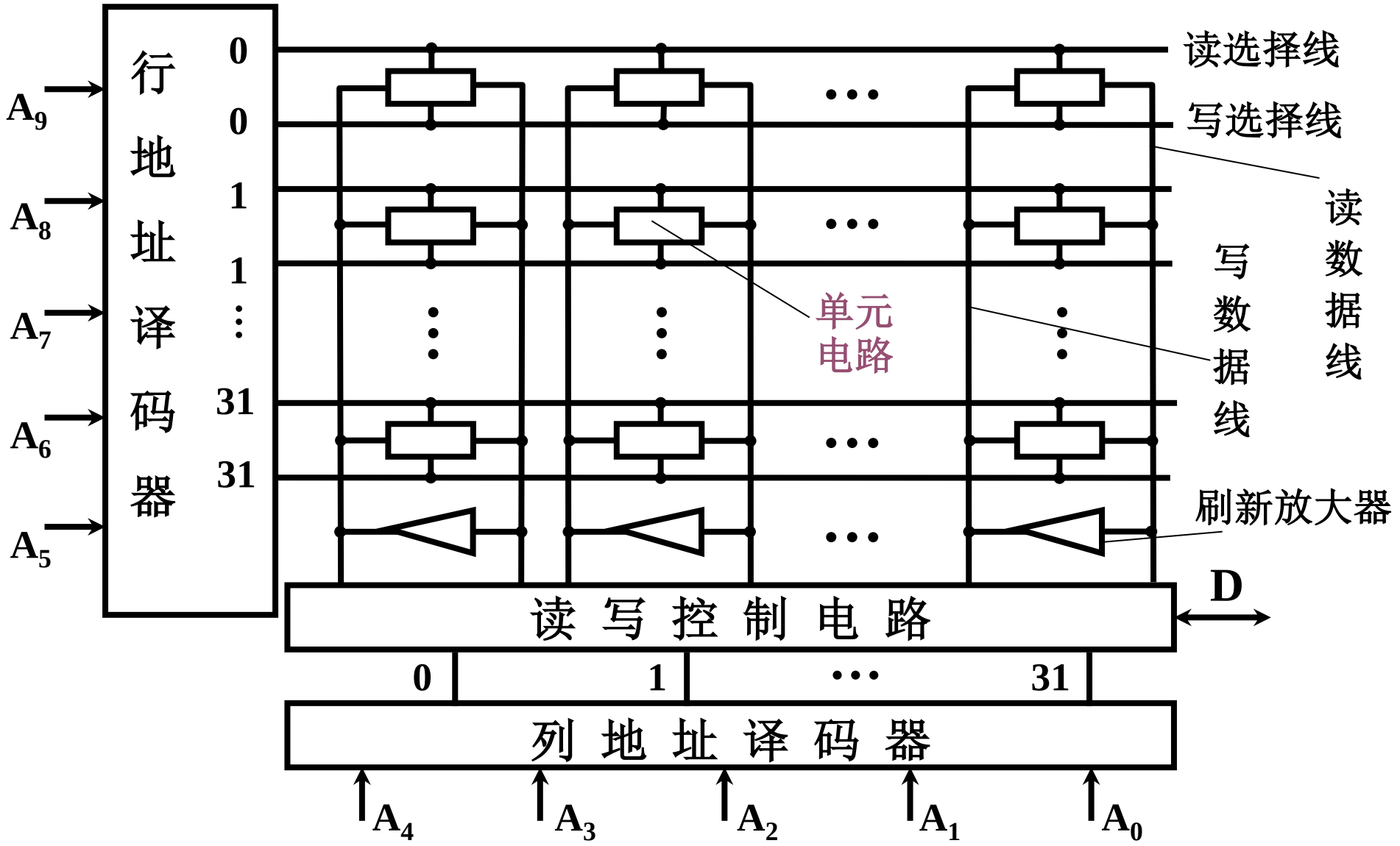
读出与原存信息相反　　读出时数据线有电流 为 “1”

写入与输入信息相同　　写入时 C_S 充电 为 “1” 放电 为 “0”

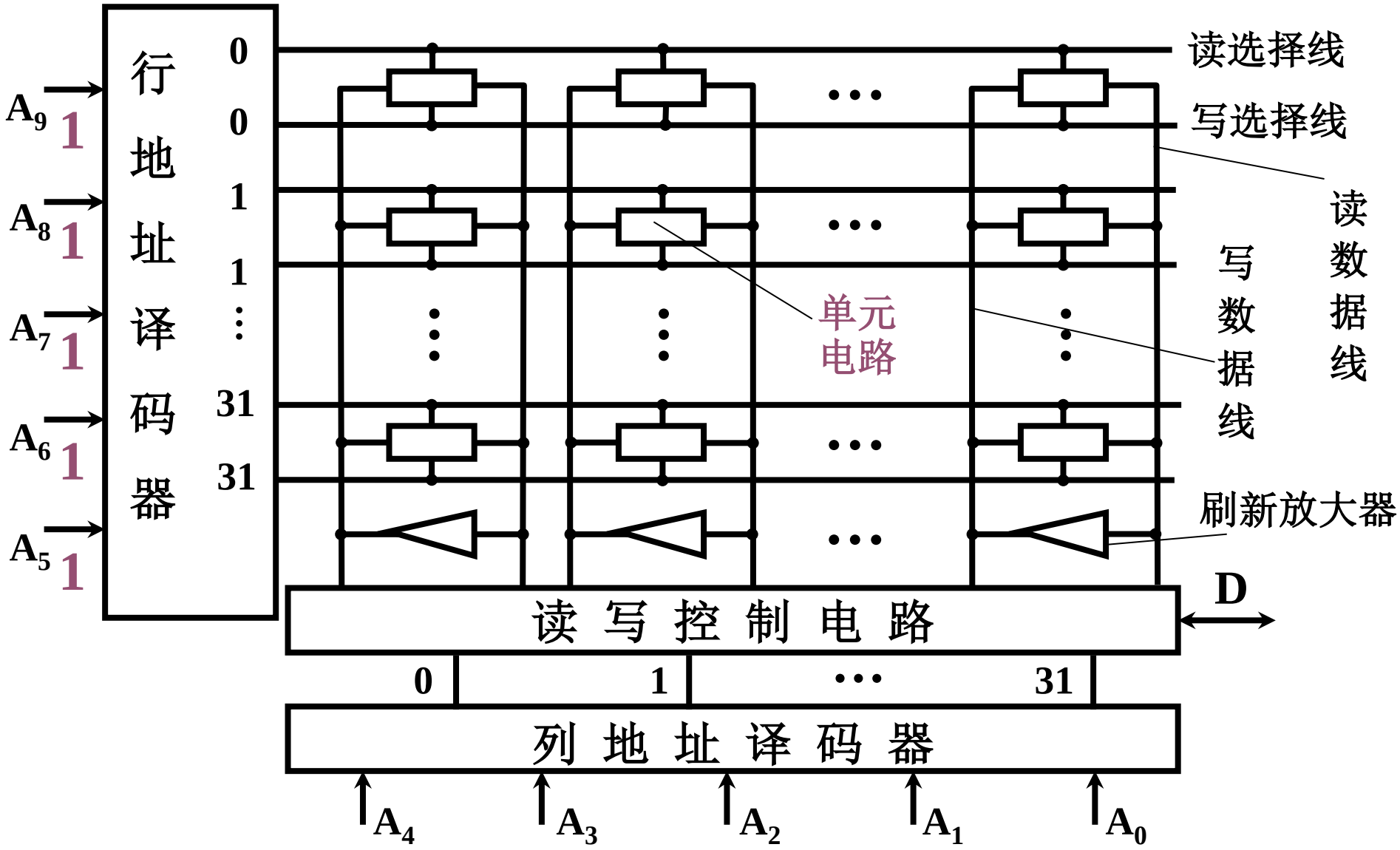
（2）动态RAM举例 ①三管动态RAM芯片 (Intel 1103) 读



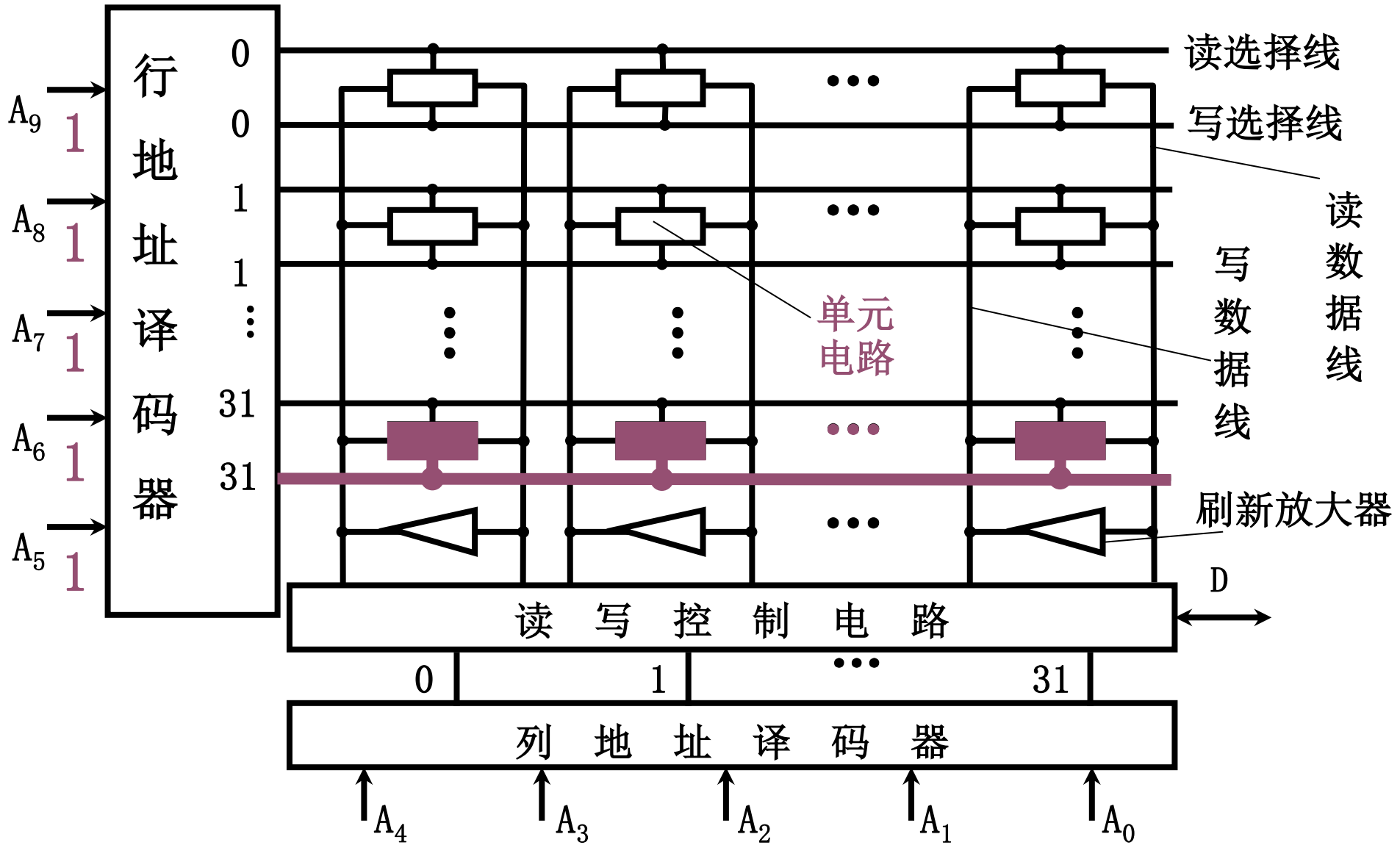
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



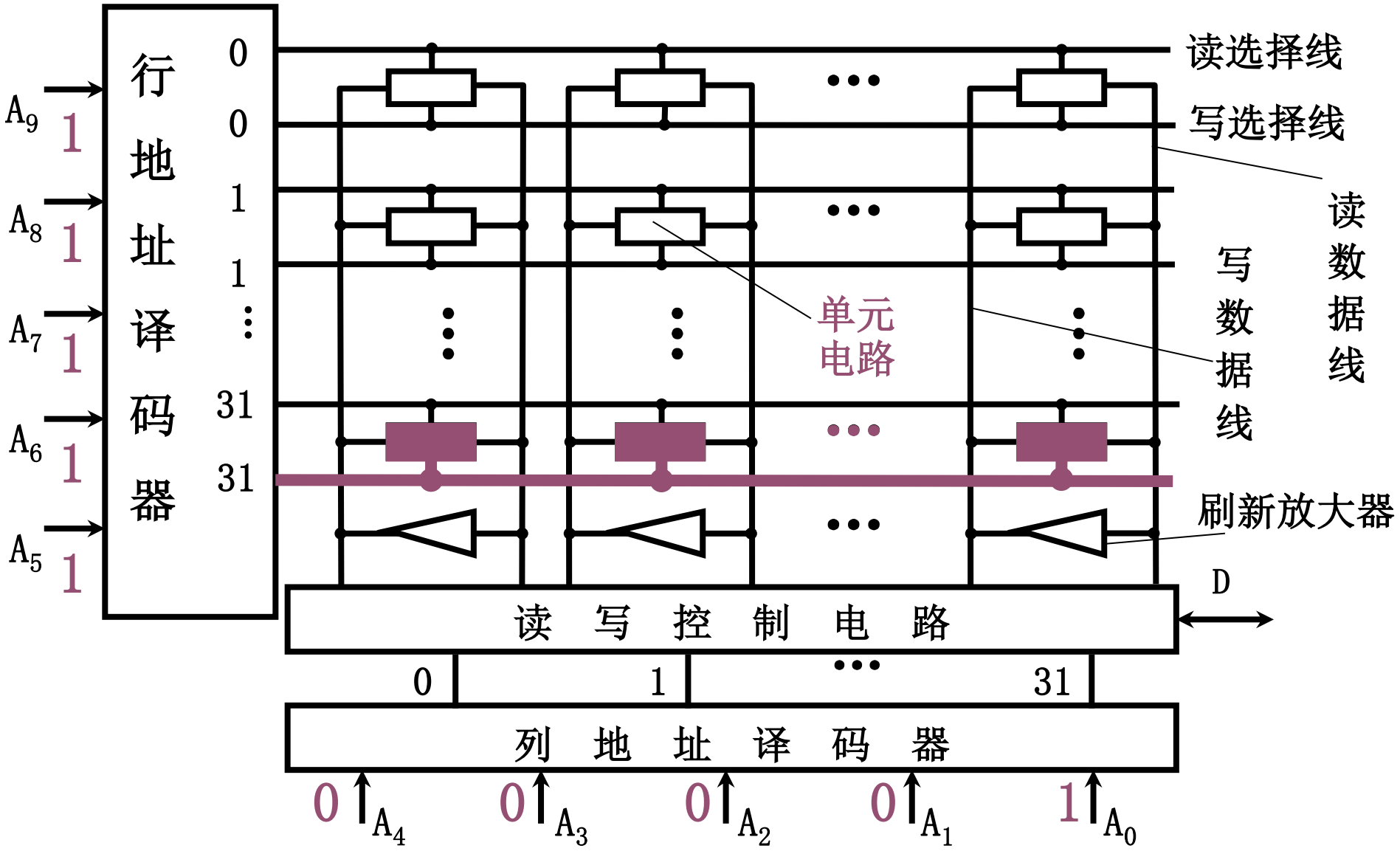
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



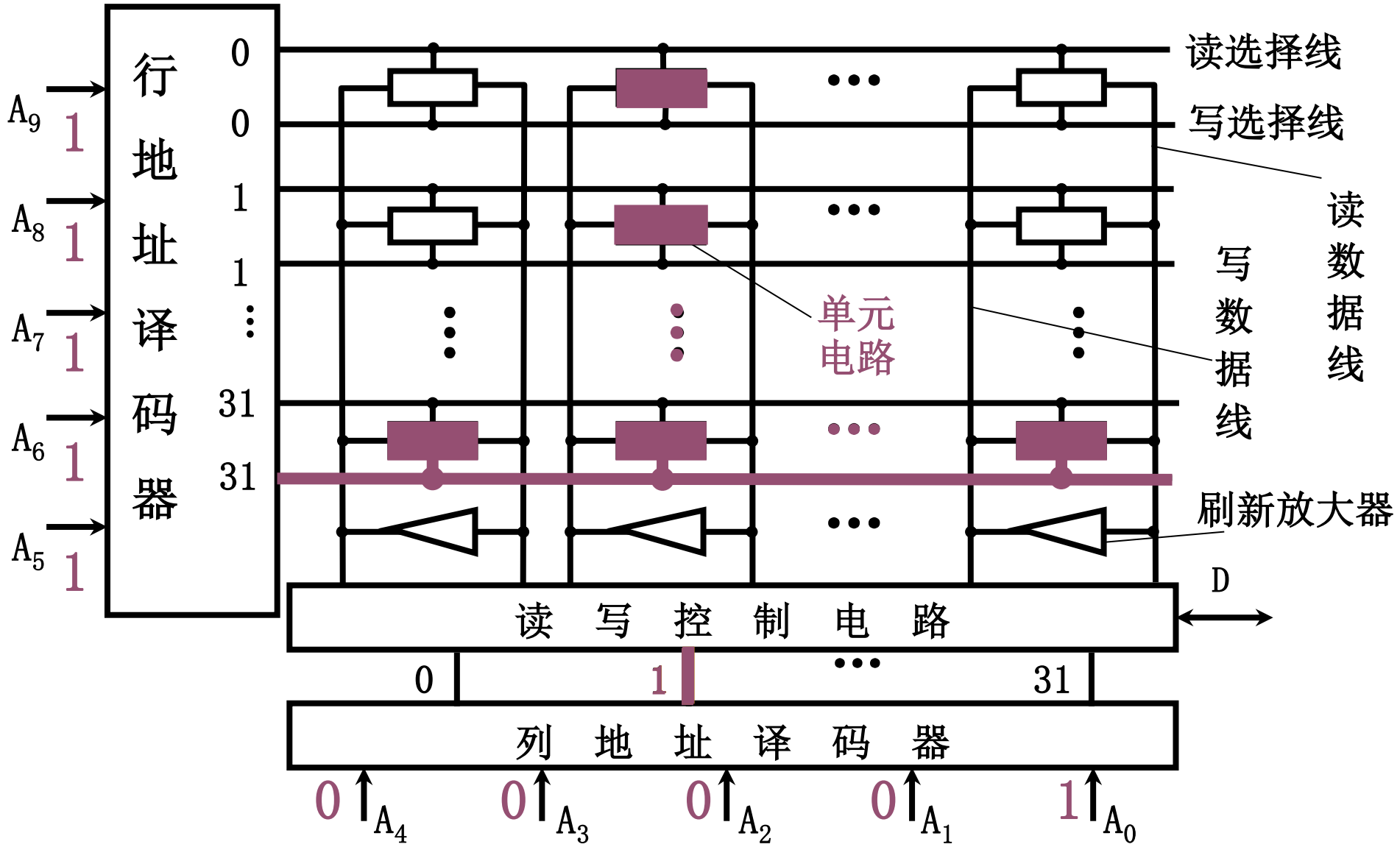
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



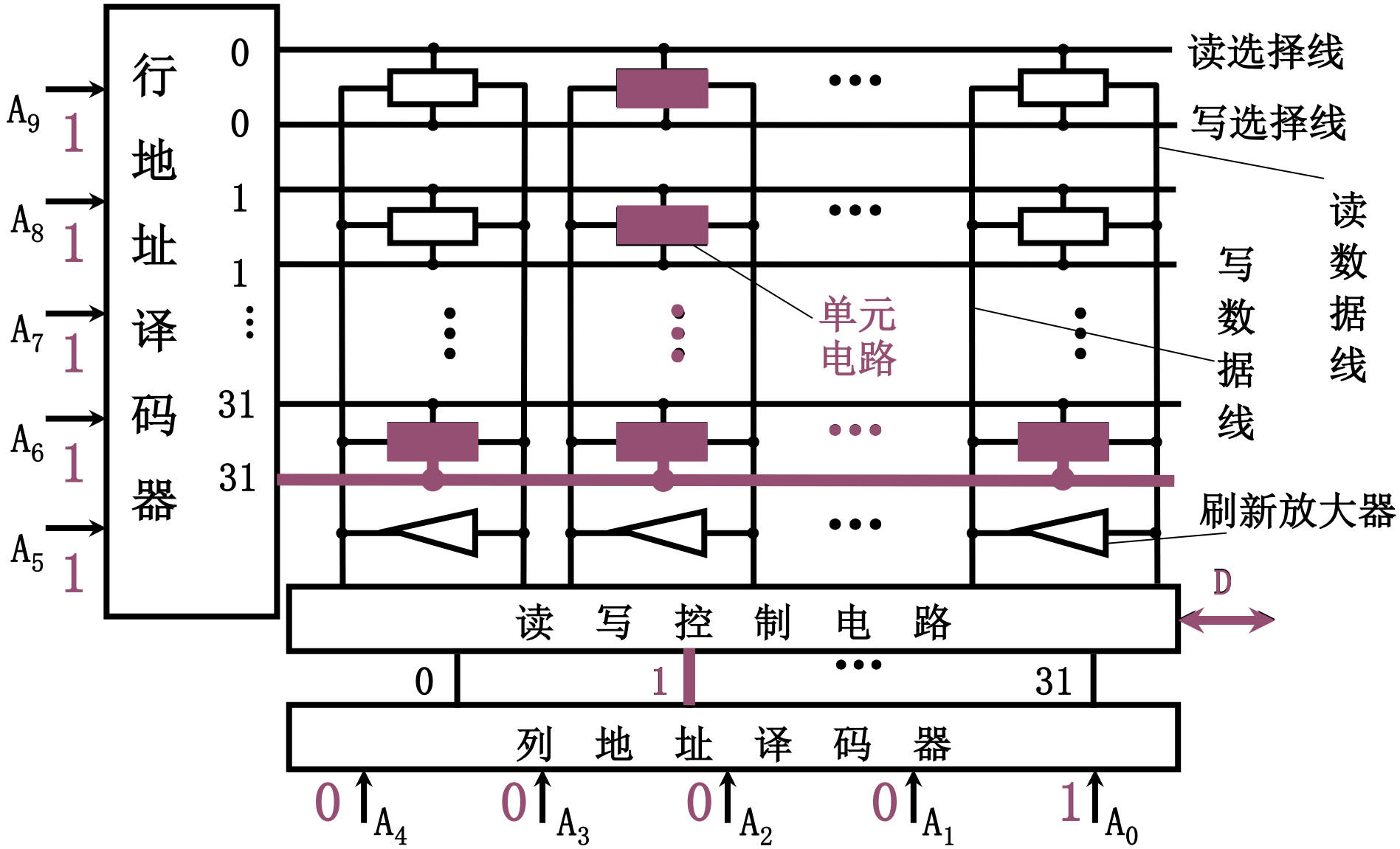
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



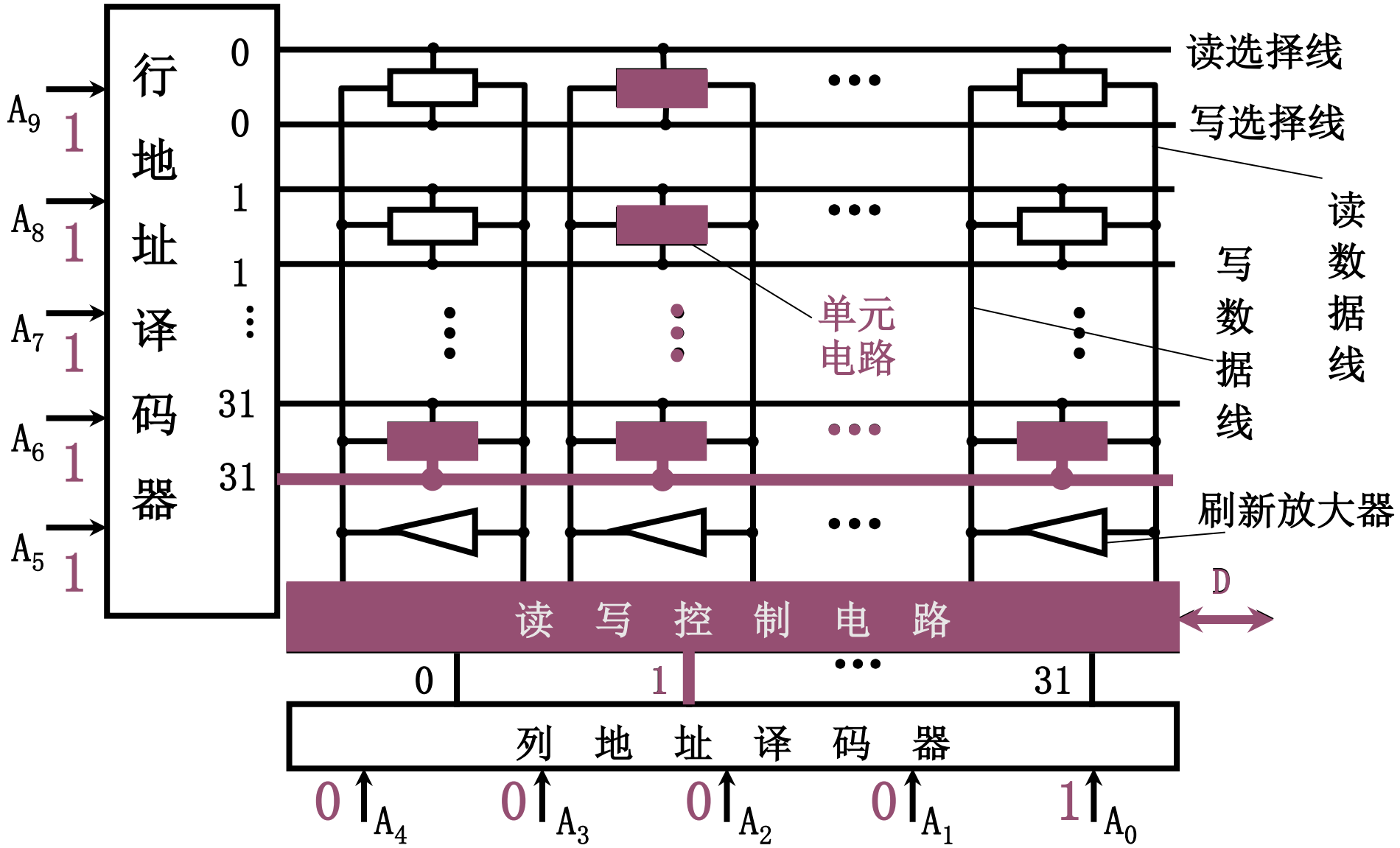
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



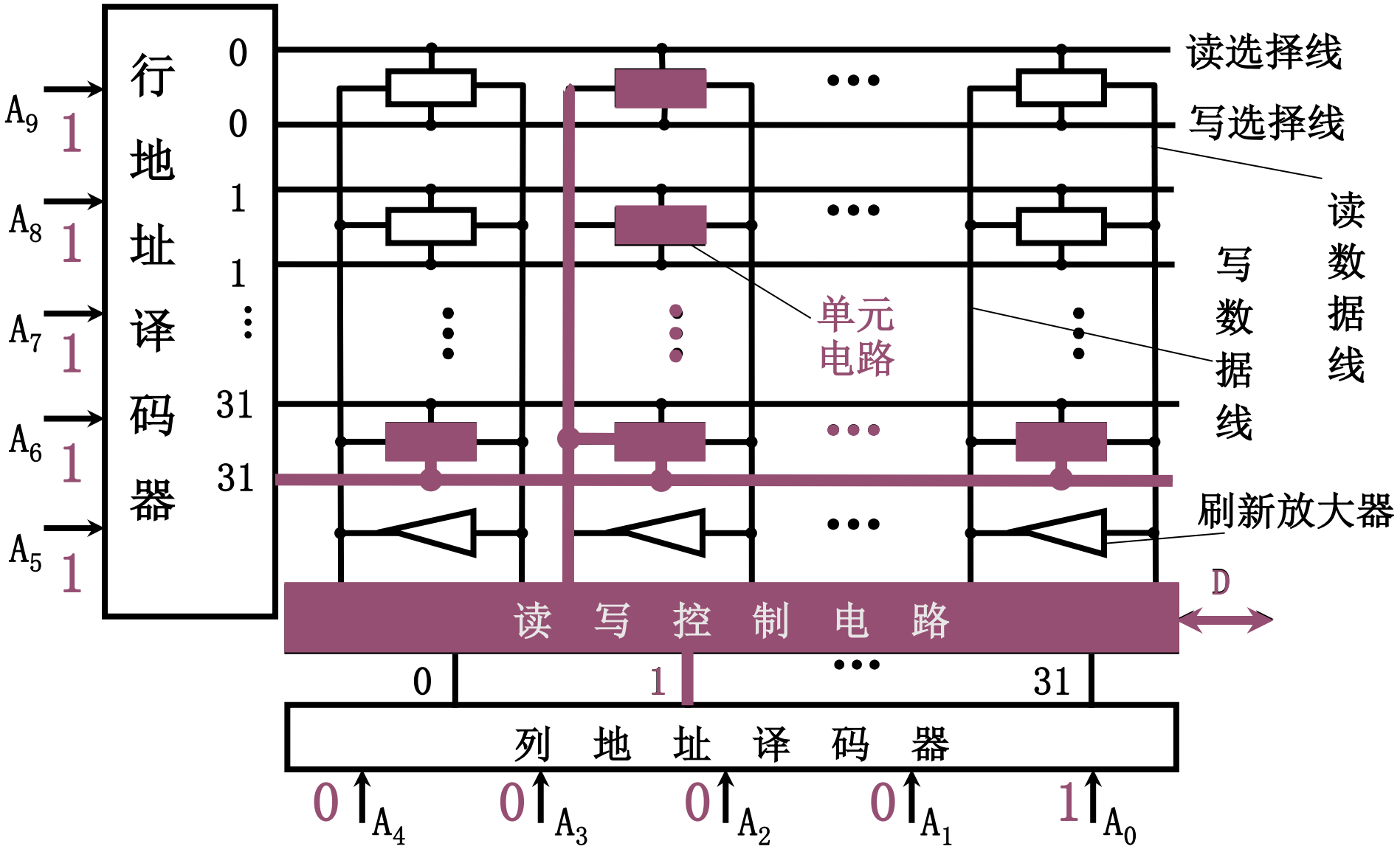
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



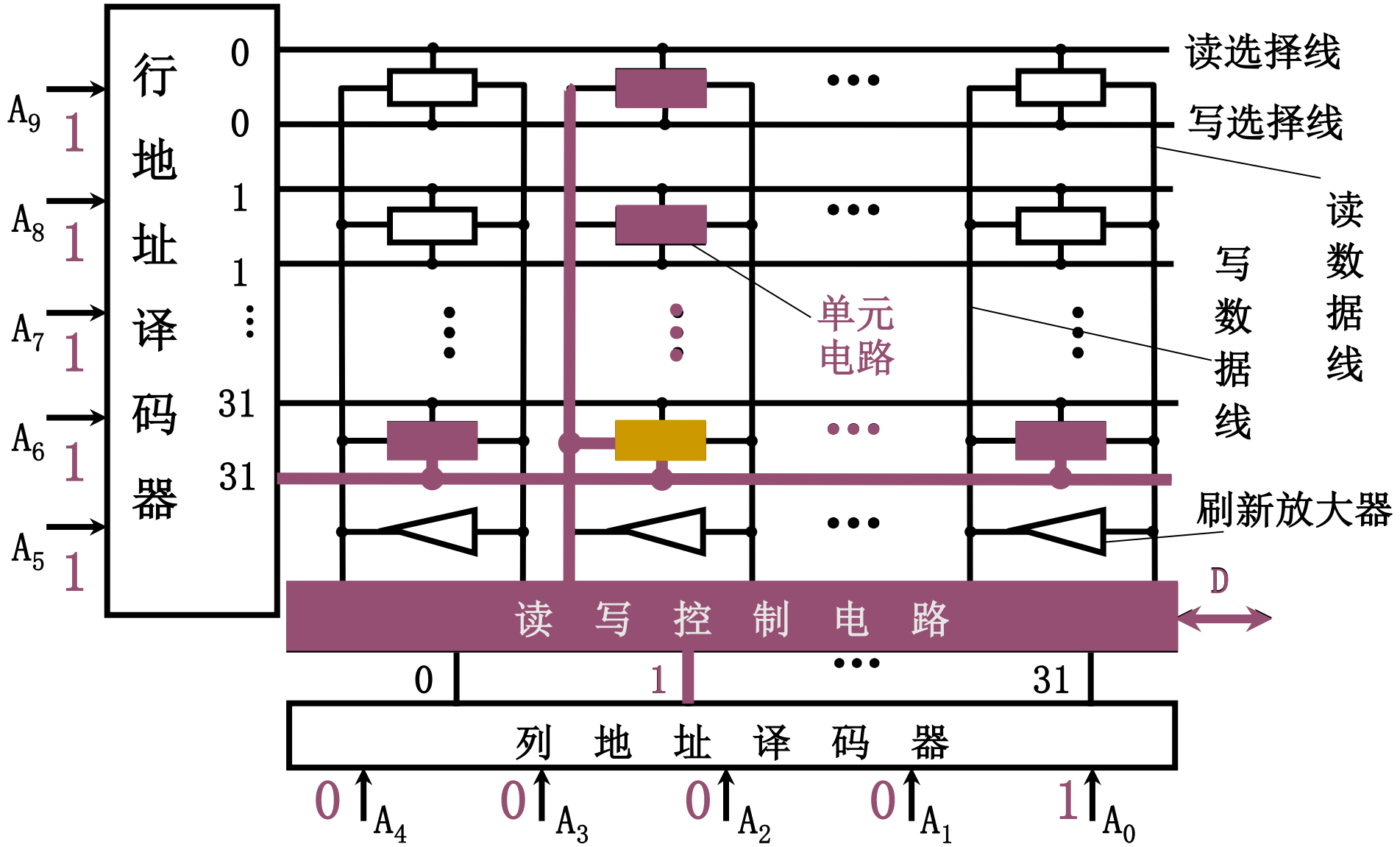
（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



（2）动态RAM举例 ②三管动态RAM芯片 (Intel 1103) 写



（3）动态RAM时序

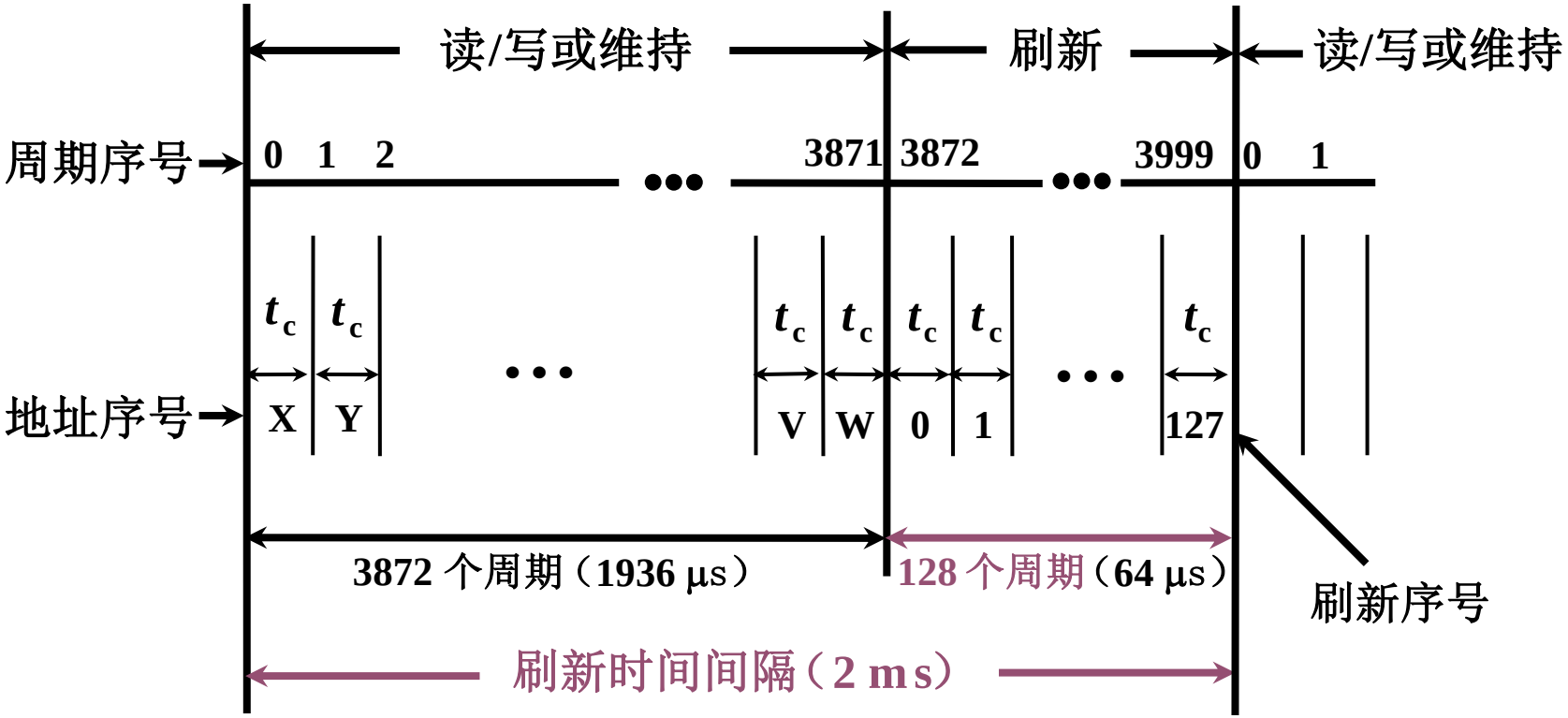
行、列地址分开传送

读时序	写时序
行地址 $\overline{\text{RAS}}$ 有效	行地址 $\overline{\text{RAS}}$ 有效
写允许 $\overline{\text{WE}}$ 有效(高)	写允许 $\overline{\text{WE}}$ 有效(低)
列地址 $\overline{\text{CAS}}$ 有效	数据 D_{IN} 有效
数据 D_{OUT} 有效	列地址 $\overline{\text{CAS}}$ 有效

动态RAM刷新

刷新与行地址有关

① 集中刷新 （存取周期为0.5 μs ）



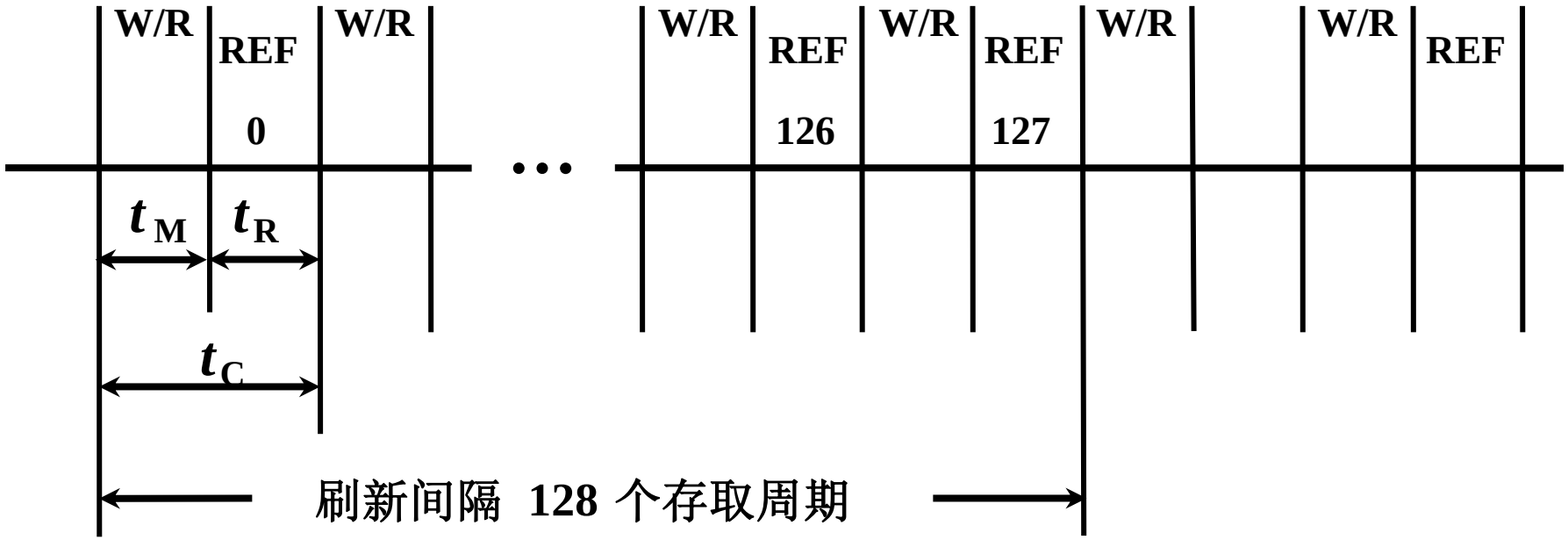
“死区”为 0.5 μs ×128 = 64 μs

“死时间率”为 128/4 000 ×100% = 3.2%

动态RAM刷新

② 分散刷新（存取周期为1μs）

以 128 ×128 矩阵为例



$t_C = t_M + t_R$　　无“死区”

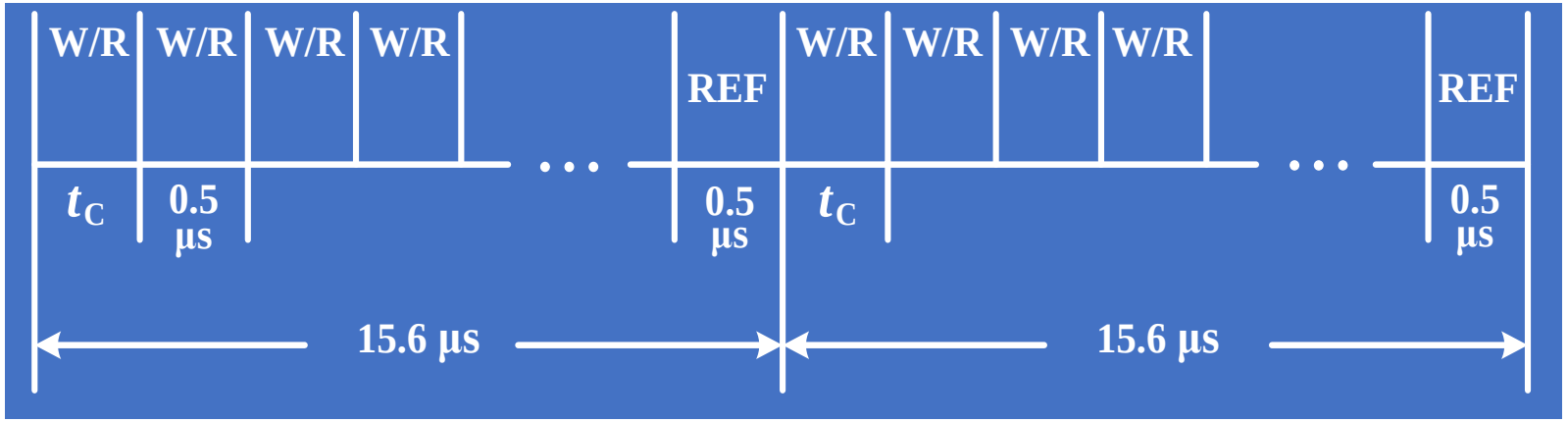
读写　刷新　　(存取周期为 0.5 μs + 0.5 μs)

动态RAM刷新

③ 分散刷新与集中刷新相结合（异步刷新）

对于 128 ×128 的存储芯片（存取周期为 0.5 μs ）

若每隔 15.6 μs 刷新一行



每行每隔 2 ms 刷新一次　　“死区”为 0.5 μs

将刷新安排在指令译码阶段，不会出现“死区”

动态RAM和静态RAM的比较

主存

缓存

	DRAM	SRAM
存储原理	电容	触发器
集成度	高	低
芯片引脚	少	多
功耗	小	大
价格	低	高
速度	慢	快
刷新	有	无

四、只读存储器（ROM）

- 早期的只读存储器——在厂家就写好了内容
- 改进1——用户可以自己写——一次性
- 改进2——可以多次写——要能对信息进行擦除
- 改进3——电可擦写——特定设备
- 改进4——电可擦写——直接连接到计算机上

4.2主存储器

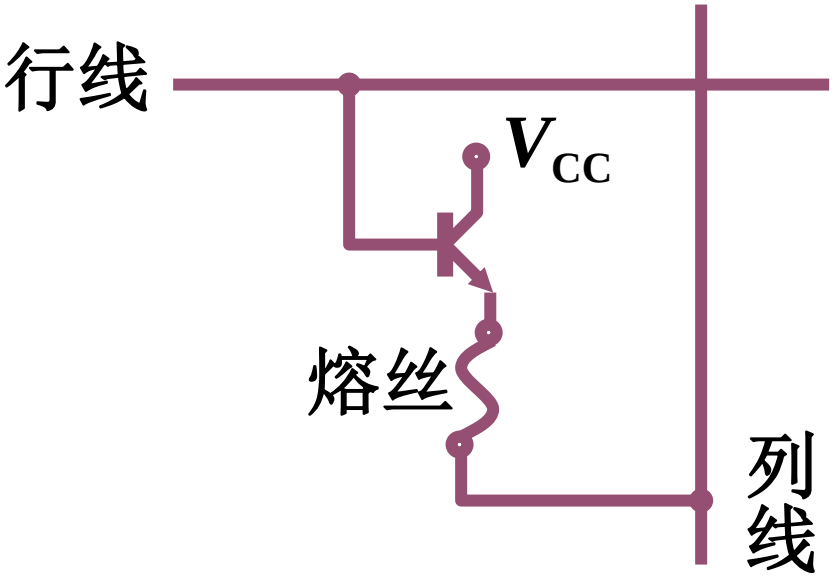
四、只读存储器（ROM）

1. 掩模 ROM (MROM)

行列选择线交叉处有 MOS 管为“1”

行列选择线交叉处无 MOS 管为“0”

2. PROM (一次性编程)



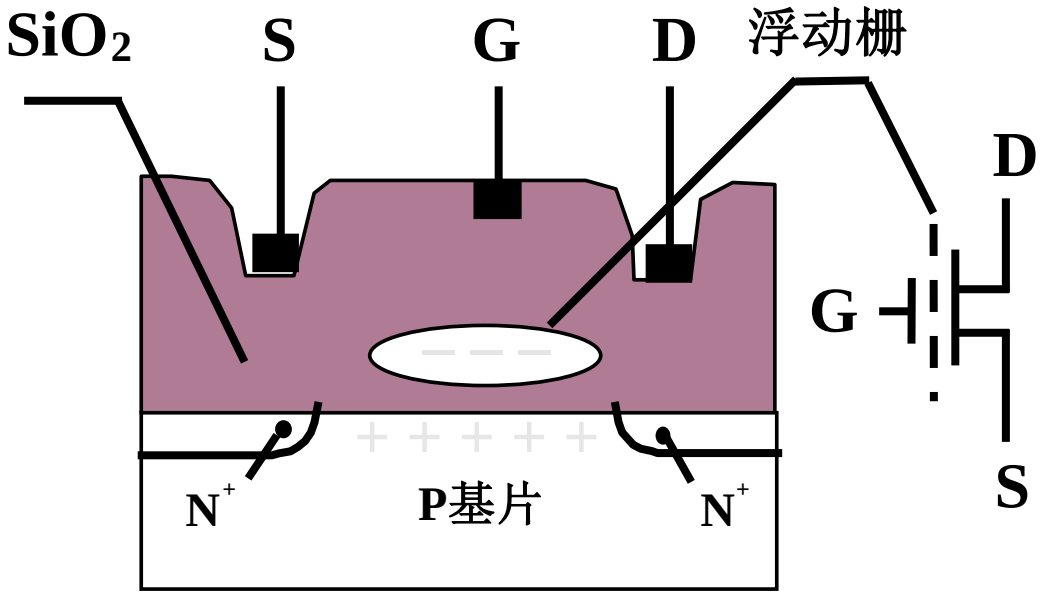
熔丝断　　为“0”

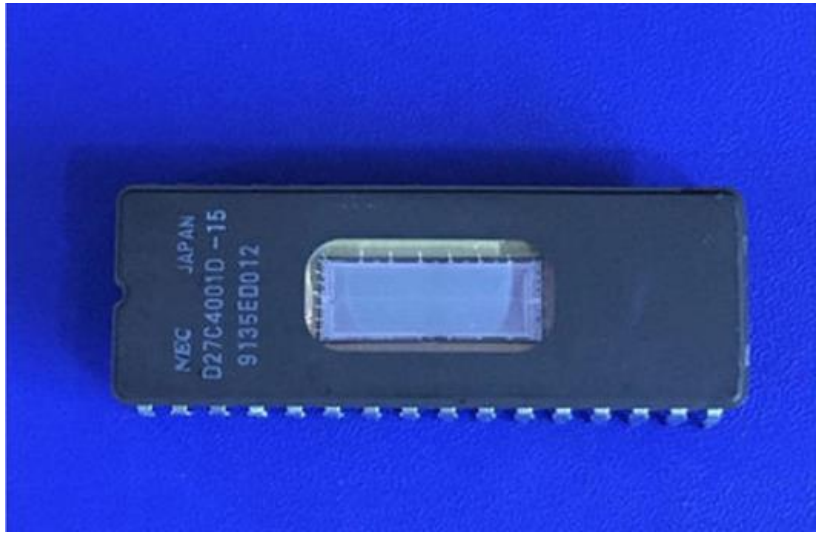
熔丝未断　为“1”

4. 2主存储器

3. EPROM (多次性编程)

例子：N型沟道浮动栅 MOS 电路





G 栅极

S 源

D 漏

紫外线全部擦洗

D 端加正电压　　形成浮动栅　　S 与 D 不导通为 “0”

D 端不加正电压　　不形成浮动栅　　S 与 D 导通为 “1”

4. 2主存储器

4. EEPROM (多次性编程)

电可擦写

局部擦写

全部擦写

5. Flash Memory (闪速型存储器)

EPROM 价格便宜 集成度高

EEPROM 电可擦洗重写

比 **EEPROM**快 具备 **RAM** 功能

五、存储器与 CPU 的连接

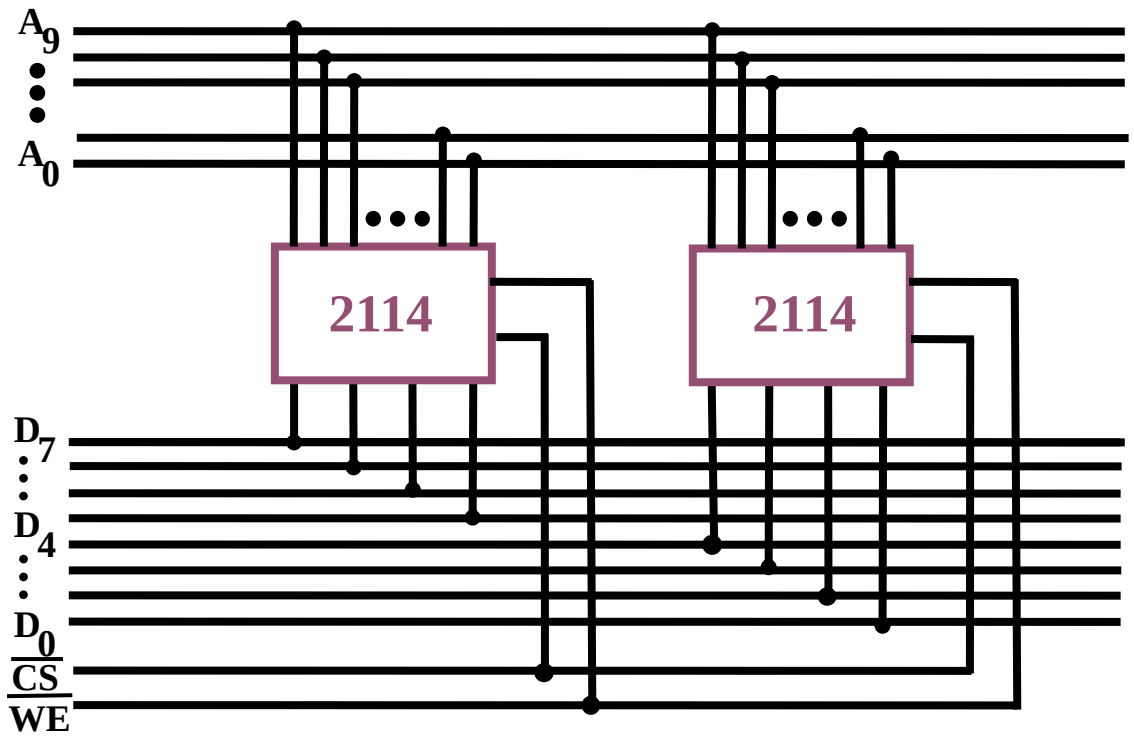
1. 存储器容量的扩展

(1) 位扩展（增加存储字长）

用 ?2片 1K×4位 存储芯片组成 1K×8位 的存储器

10根地址线

8根数据线

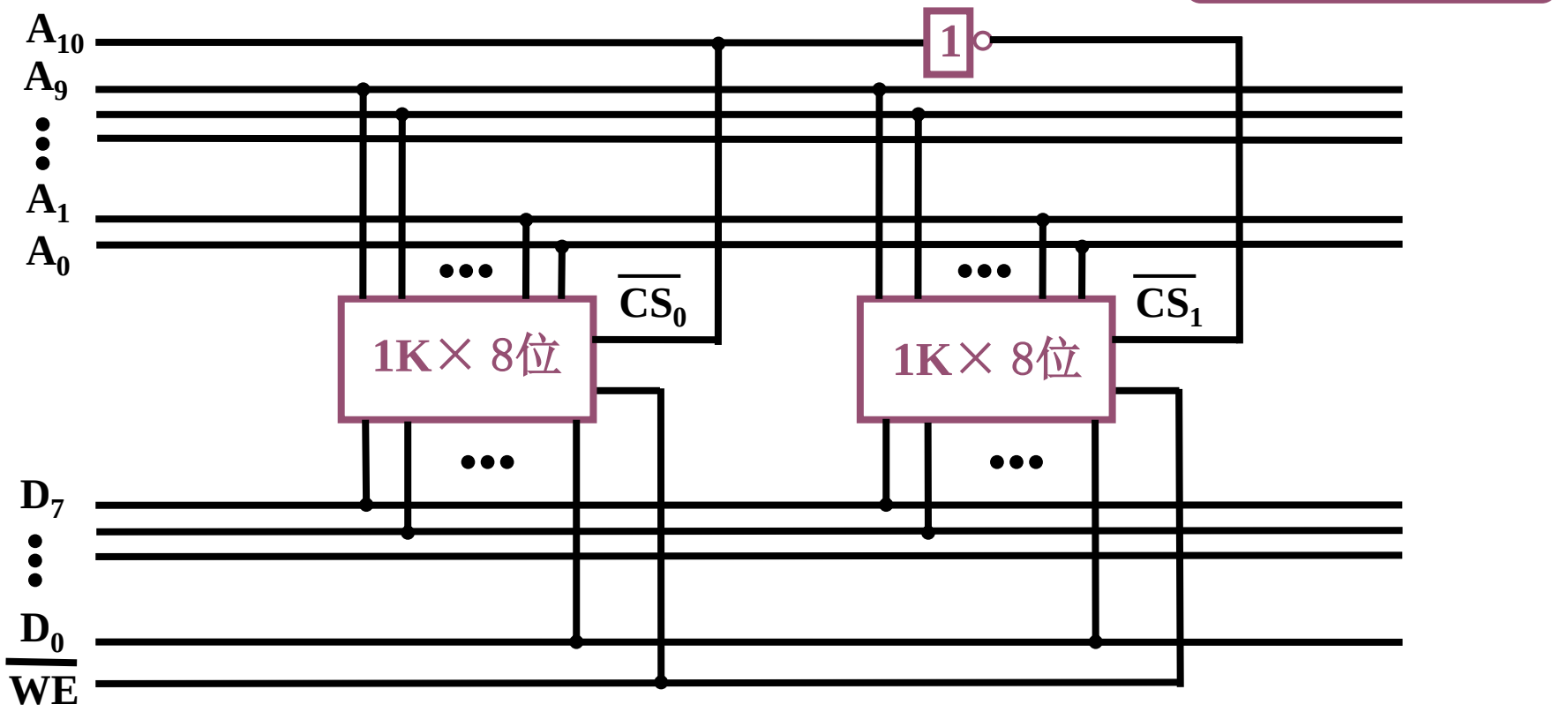


(2) 字扩展（增加存储字的数量）

11根地址线

用 2片 1K×8位 存储芯片组成 2K×8位 的存储器

8根数据线

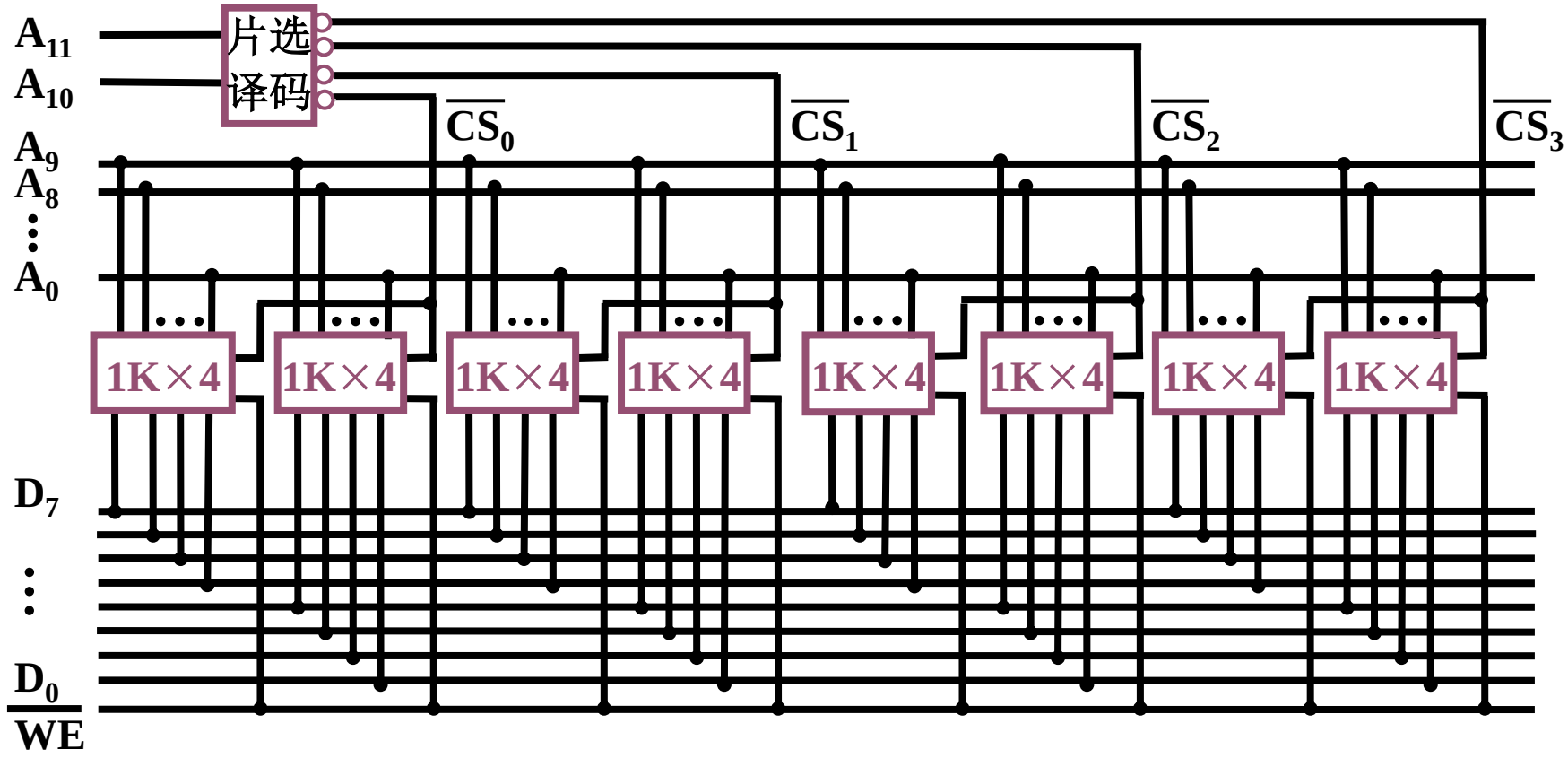


(3) 字、位扩展

用 8片 1K × 4位 存储芯片组成 4K × 8位 的存储器

12根地址线

8根数据线



4. 2主存储器

2. 存储器与 CPU 的连接

(1) 地址线的连接

(2) 数据线的连接

(3) 读/写命令线的连接

(4) 片选线的连接

(5) 合理选择存储芯片

(6) 其他　　时序、负载

例4.1

设 CPU 有 16 根地址线，8 根数据线，

$\overline{\text{MREQ}}$ 访存控制信号（低电平有效），

$\overline{\text{WR}}$ 读/写控制信号（高电平为读，低电平为写）

RAM：1K×4位；4K×8位；8K×8 位

ROM：2K×8位；4K×8位；8K×8 位

74LS138 译码器和各种门电路

画出 CPU 与存储器的连接图，要求

① 主存地址空间分配：

6000H～67FFH 为系统程序区；

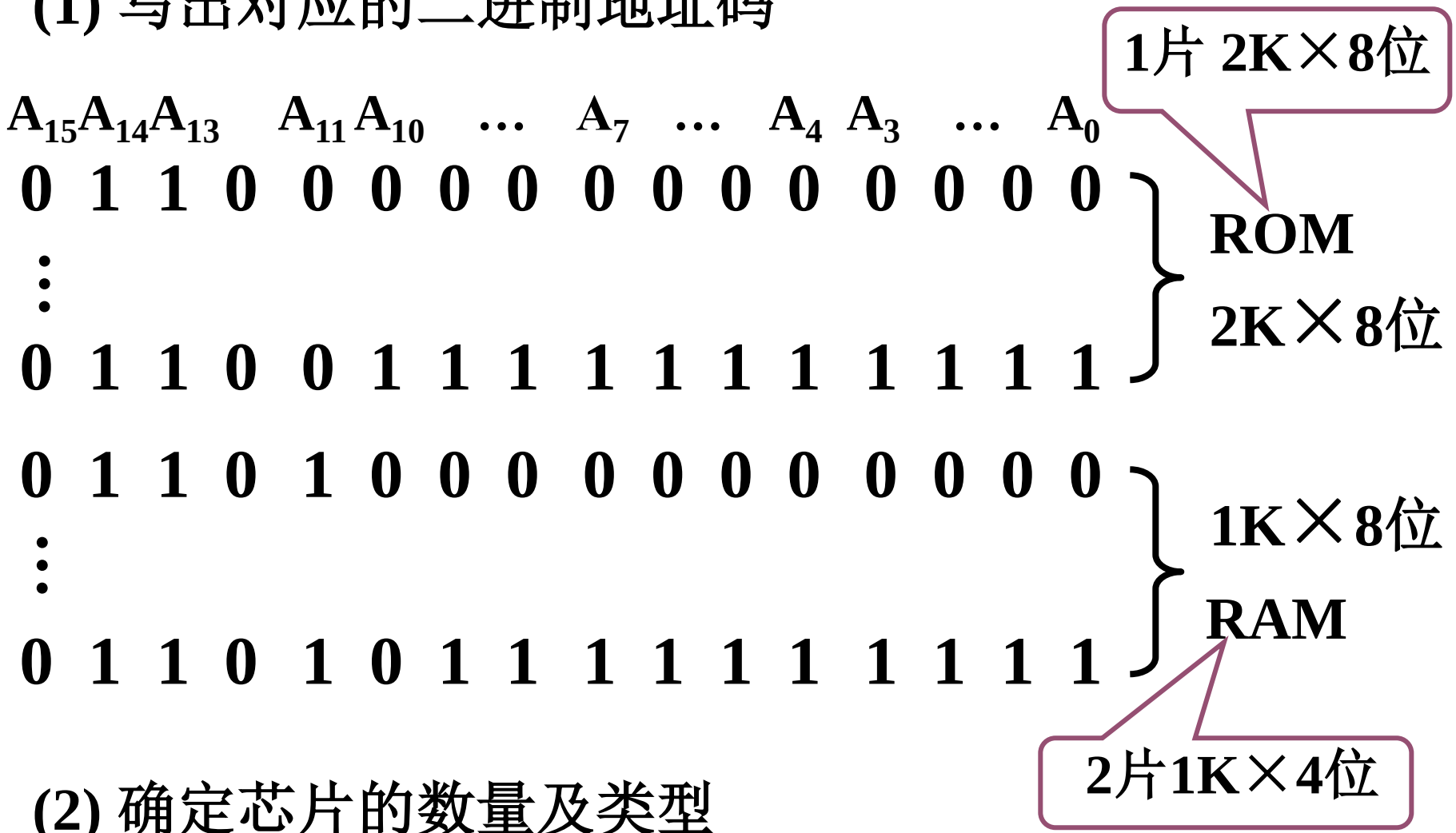
6800H～6BFFH 为用户程序区。

② 合理选用上述存储芯片，说明各选几片？

③ 详细画出存储芯片的片选逻辑图。

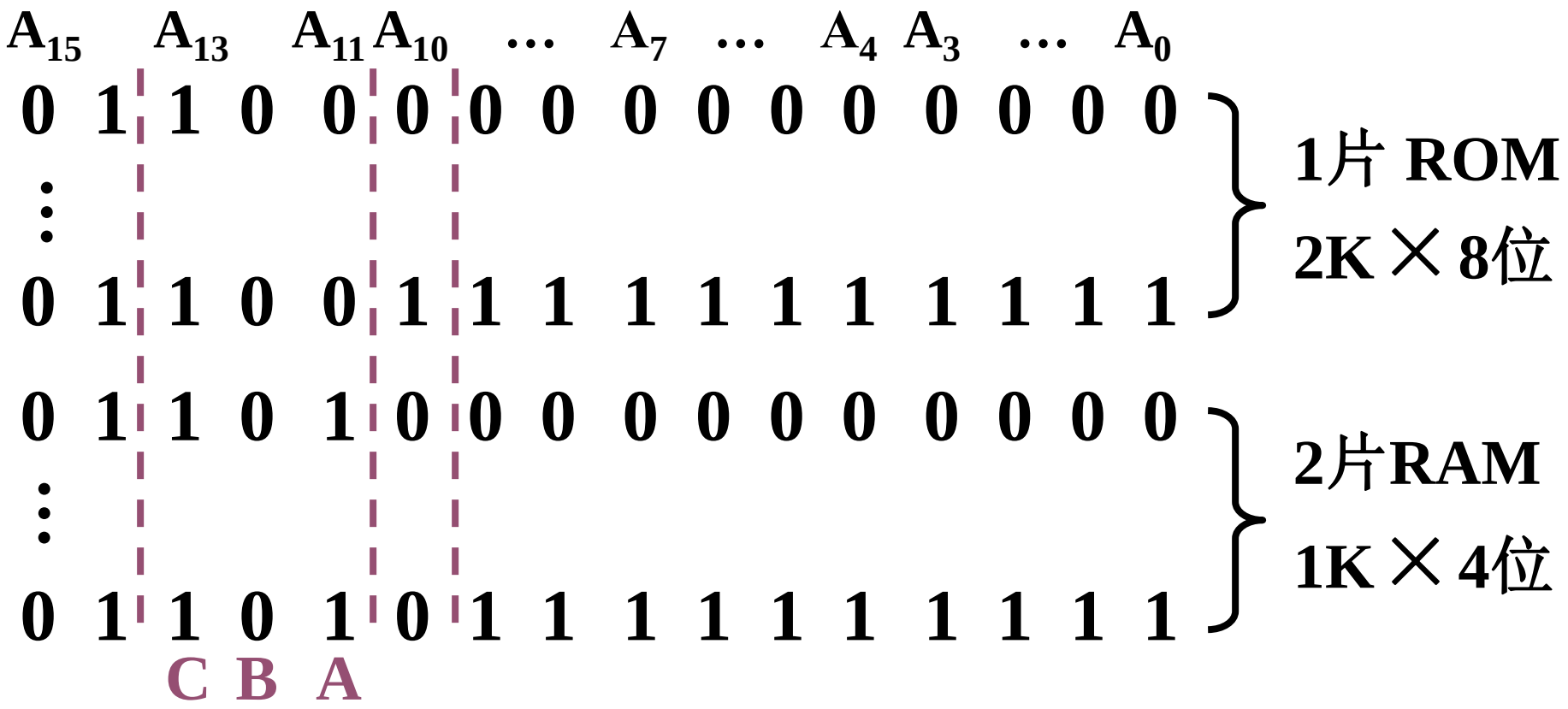
例4.1 解:

(1) 写出对应的二进制地址码



(2) 确定芯片的数量及类型

(3) 分配地址线

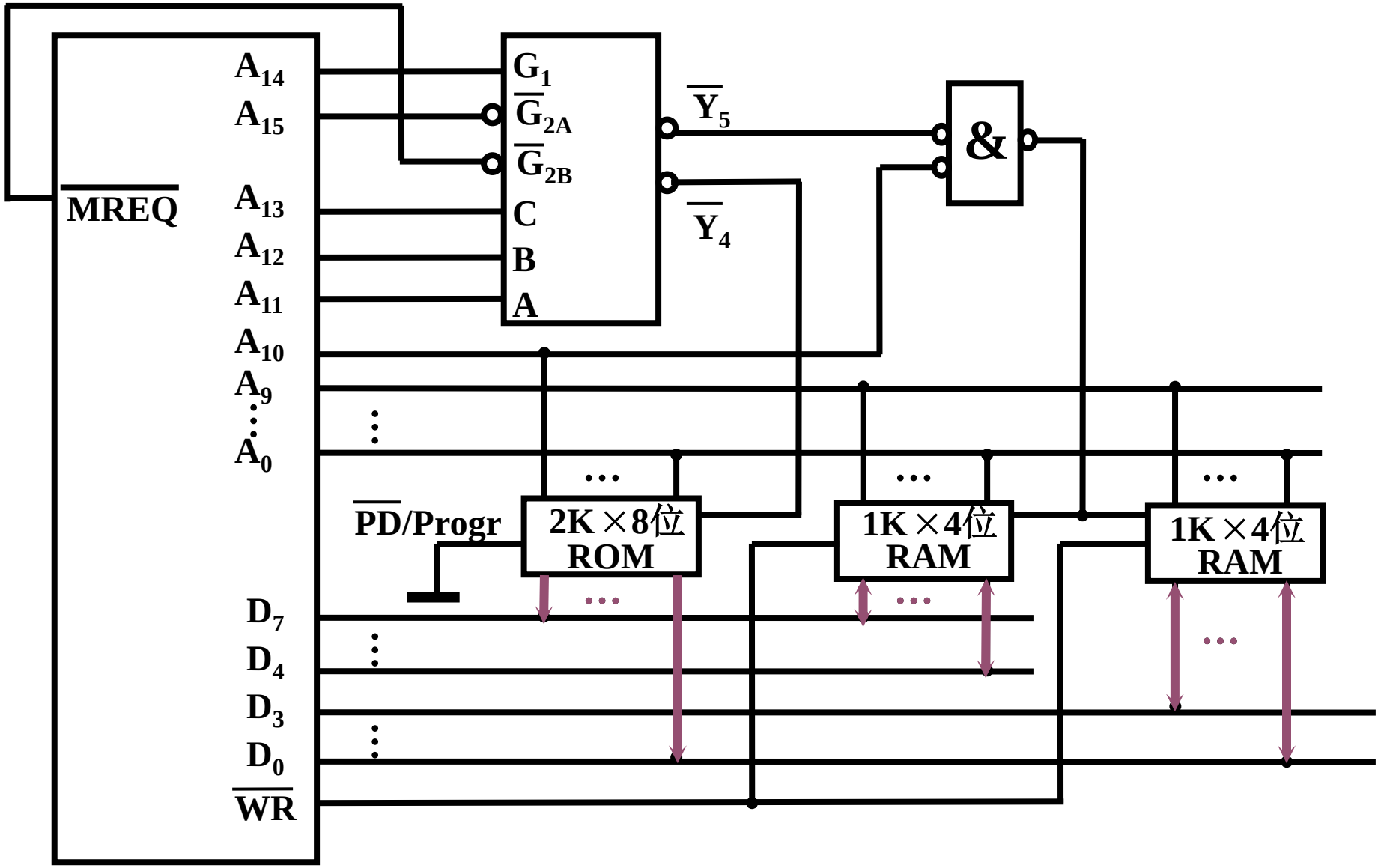


A_{10}~A_0 接 2K×8位 ROM 的地址线

A_9 ~A_0 接 1K×4位 RAM 的地址线

(4) 确定片选信号

例 4.1 CPU 与存储器的连接图



例4.2 假设同前，要求最小 4K为系统程序区，相邻 8K为用户程序区。

(1) 写出对应的二进制地址码

(2) 确定芯片的数量及类型

1片 4K×8位 ROM　2片 4K×8位 RAM

(3) 分配地址线

A_{11}~A_0 接 ROM 和 RAM 的地址线

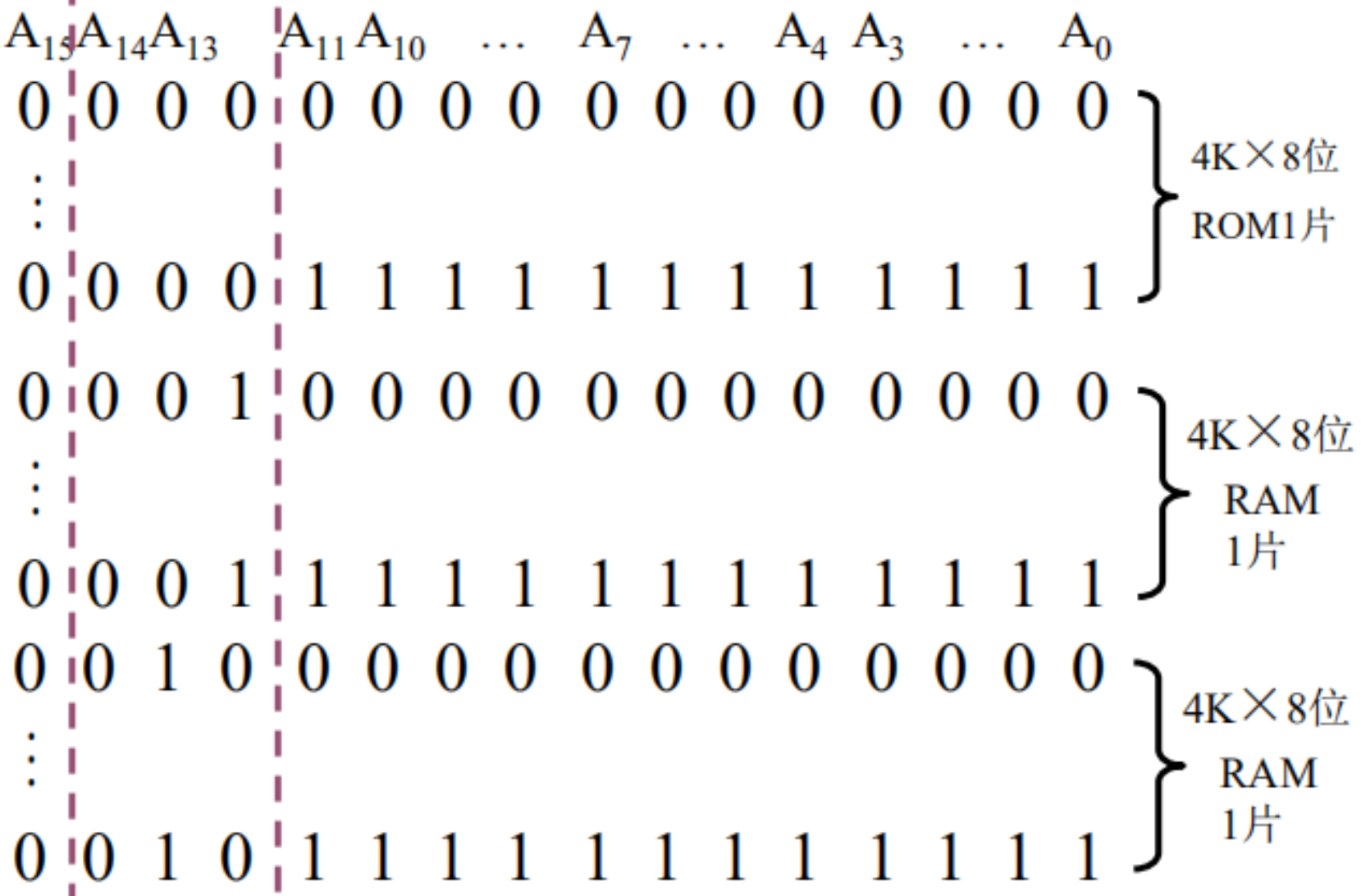
(4) 确定片选信号

例4.2 假设同前，要求最小 4K为系统程序区，相邻 8K为用户程序区。

(1) 写出对应的二进制地址码　　(2) 确定芯片的数量及类型

$A_{15}A_{14}A_{13}$	$A_{11}A_{10}$	…	A_7	…	A_4	A_3	…	A_0	
0 0 0	0 0	…	0	…	0	0	…	0	4K×8位 ROM1片
⋮									
0 0 0	1 1	…	1	…	1	1	…	1	
0 0 1	0 0	…	0	…	0	0	…	0	4K×8位 RAM 1片
⋮									
0 0 1	1 1	…	1	…	1	1	…	1	
0 1 0	0 0	…	0	…	0	0	…	0	4K×8位 RAM 1片
⋮									
0 1 0	1 1	…	1	…	1	1	…	1	

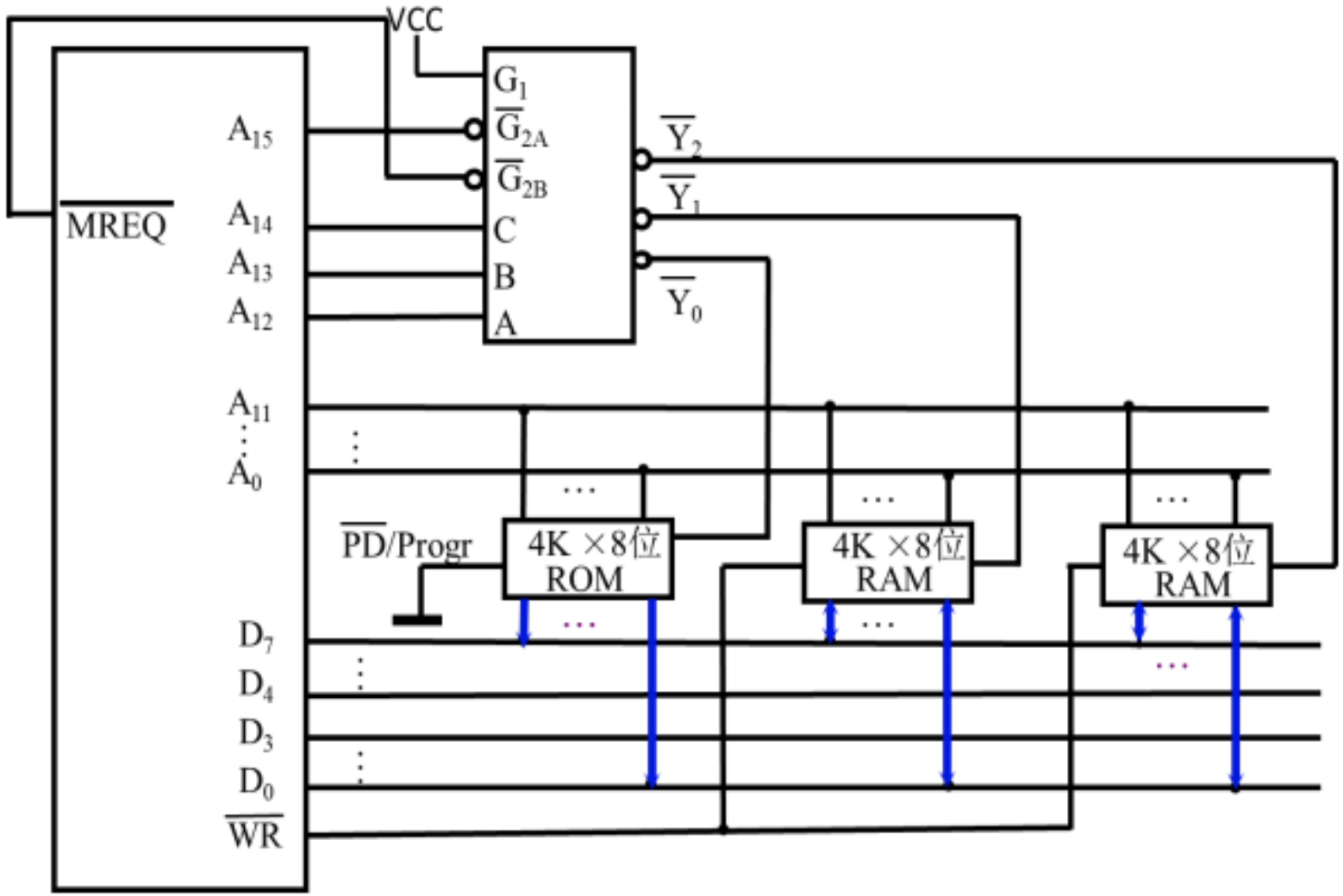
(3) 分配地址线



(4) 确定片选信号

例 4.2 CPU 与存储器的连接图

4.2



六、存储器的校验

- 为什么要对存储器的信息进行校验？
- 为了能够校验出信息是否正确，如何进行编码？
- 纠错和检错能力与什么因素有关？
- 校验出信息出错后是如何进行纠错？

六、存储器的校验

合法编码集合	检错、纠错能力
• {000, 001, 010, 011, 100, 101, 110, 111}	检0位错、纠0位错
• {000, 011, 101, 110}	检1位错，纠0位错
100（000，101，110）	
• {000，111}	检1位错，纠1位错
100（000）110	
• {0000，1111}	检2位错，纠1位错
1000 1100	
• {00000，11111}	检2位错，纠2位错
11000 11100	

编码的检测能力和纠错能力与什么有关？

任意两组合法编码之间二进制位的最小差异数

六、存储器的校验

1 . 编码的最小距离

任意两组合法代码之间 二进制位数 的 最少差异

编码的纠错 、检错能力与编码的最小距离有关

$$L - 1 = D + C \ (\ D \geq C\ )$$

L —— 编码的最小距离

D —— 检测错误的位数

C —— 纠正错误的位数

$L = 3$

具有 一位 纠错能力

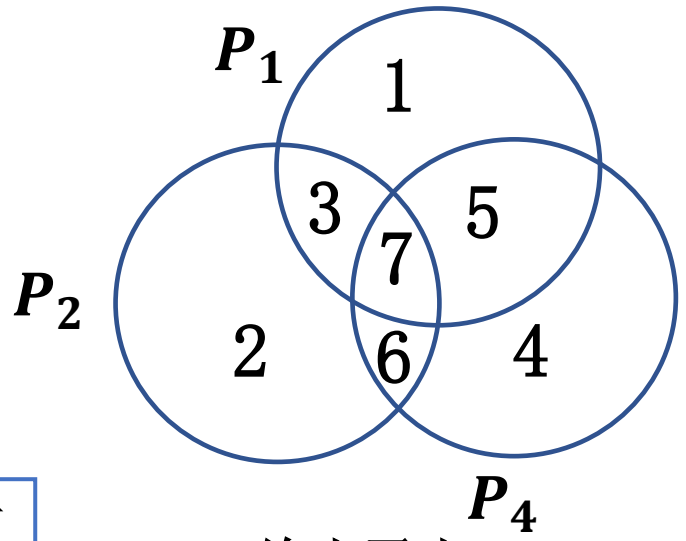
汉明码——一种具有一位纠错能力的编码

4.2主存储器

- 组成汉明码的三要素

汉明码采用奇偶校验

划分式分组校验例子：



校验位应放哪里呢？

1, 2, 4, 8, …
位置放校验码

校验位　　校验位

00100011　100100011　10010000011

给出了出错的位置

$P_4P_2P_1$	无差错
0 0 0	
0 0 1	1
1 0 1	5
1 1 0	6
1 1 1	7

汉明码的分组是一种非划分方式

1	2	3	4	5	6	7

如何分组的呢？

第1组 XXXX1
第2组 XXX1X
第3组 XX1XX
第4组 X1XXX
第5组 1XXXX

分成3组，每组有1位校验位，共包括4位数据位

4.2主存储器

- 组成汉明码的三要素

汉明码的组成需增添 k 位检测位

$$2^k \geq n + k + 1$$

检测位的位置 ？

$$2^i \ (i = 0，\ 1，\ 2，\ 3，\ \ldots)$$

检测位的取值 ？

与该位所在的检测“小组”中承担的奇偶校验任务有关

各检测位 C_i 所承担的检测小组

- C_1 检测 g_1 小组包含位 1,3,5,7,9,11,... (XXXX1)
- C_2 检测 g_2 小组包含位 2,3,6,7,10,11,... (XXX1X)
- C_4 检测 g_3 小组包含位 4,5,6,7,12,13,... (XX1XX)
- C_8 检测 g_4 小组包含位 8,9,10,11,12,13,14,15,24,... (X1XXX)

- g_i 小组独占第 2^{i-1} 位
- g_i 和 g_j 小组共同占第 $2^{i-1}+2^{j-1}$ 位
- g_i、g_j 和 g_l 小组共同占第 $2^{i-1}+2^{j-1}+2^{l-1}$ 位

例4.4 求 0101 按“偶校验”配置的汉明码

解：∵ $n = 4$

根据 $2^k \geq n + k + 1$

得 $k = 3$

汉明码排序如下：

二进制序号	1	2	3	4	5	6	7
名称	C_1	C_2	0	C_4	1	0	1
	0	1		0			

∴ 0101 的汉明码为 0100101

练习1　按配偶原则配置 0011 的汉明码

解：　∵ $n = 4$　　根据 $2^k \geq n + k + 1$

取 $k = 3$

二进制序号	1	2	3	4	5	6	7
名称	C_1	C_2	0	C_4	0	1	1
	1	0		0			

$C_1 = 3 \oplus 5 \oplus 7 = 1$

$C_2 = 3 \oplus 6 \oplus 7 = 0$

$C_4 = 5 \oplus 6 \oplus 7 = 0$

∴ 0011 的汉明码为 **1000011**

3. 汉明码的纠错过程

形成新的检测位 P_i，其位数与增添的检测位有关，如增添 3 位（$k = 3$），新的检测位为 $P_4 P_2 P_1$。

以 $k = 3$ 为例，P_i 的取值为

$$P_1 = \overset{C_1}{1} \oplus 3 \oplus 5 \oplus 7$$

$$P_2 = \overset{C_2}{2} \oplus 3 \oplus 6 \oplus 7$$

$$P_4 = \overset{C_4}{4} \oplus 5 \oplus 6 \oplus 7$$

对于按“偶校验”配置的汉明码

不出错时 $P_1 = 0$，$P_2 = 0$，$P_4 = 0$

例4.5　已知接收到的汉明码为 0100111

（按配偶原则配置）试问要求传送的信息是什么？

解：纠错过程如下

$$P_1 = 1 \oplus 3 \oplus 5 \oplus 7 = 0$$ 无错

$$P_2 = 2 \oplus 3 \oplus \boxed{6} \oplus 7 = 1$$ 有错

$$P_4 = 4 \oplus 5 \oplus \boxed{6} \oplus 7 = 1$$ 有错

$$\therefore \quad P_4P_2P_1 = 110$$

第 6 位出错，可纠正为 0100101，

故要求传送的信息为 0101。

练习2 写出按偶校验配置的汉明码

0101101 的纠错过程

$P_4 = 4 \oplus 5 \oplus 6 \oplus 7 = 1$

$P_2 = 2 \oplus 3 \oplus 6 \oplus 7 = 0$

$P_1 = 1 \oplus 3 \oplus 5 \oplus 7 = 0$

$\therefore P_4P_2P_1 = 100$ 第 4 位错，可不纠

练习3 按配奇原则配置 0011 的汉明码

配奇的汉明码为 0101011

4.2主存储器

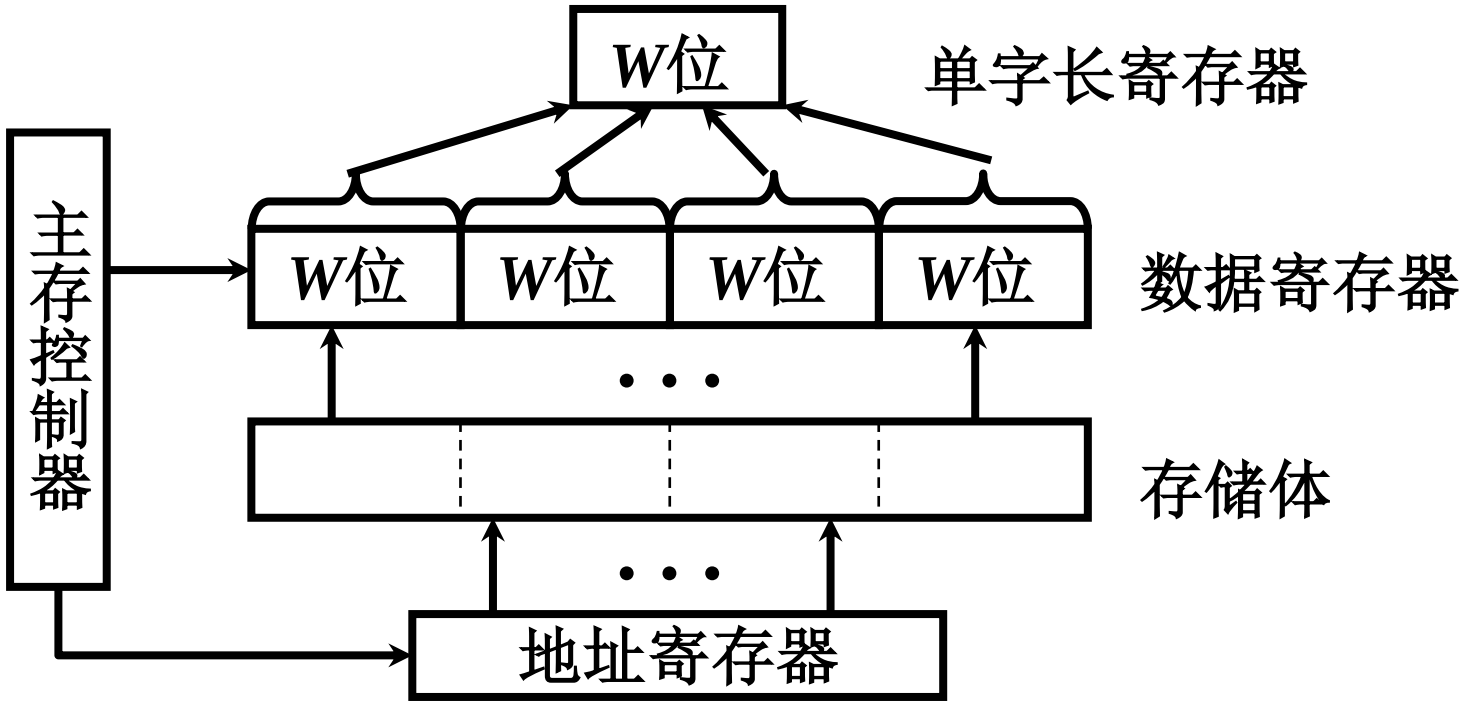
六、提高访存速度的措施

- 采用高速器件
- 采用层次结构 Cache –主存
- 调整主存结构

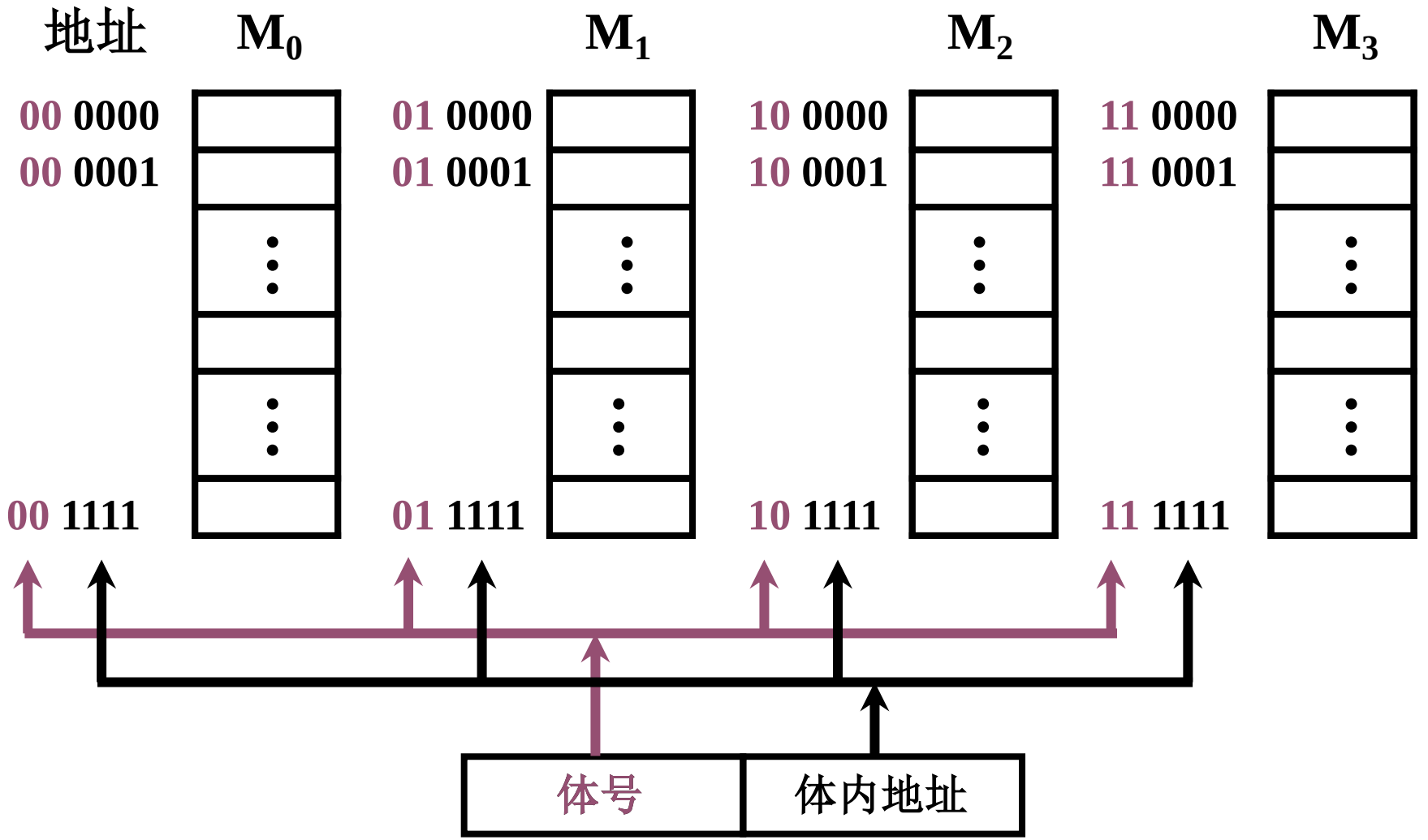
调整主存结构，提高访存速度

1. 单体多字系统

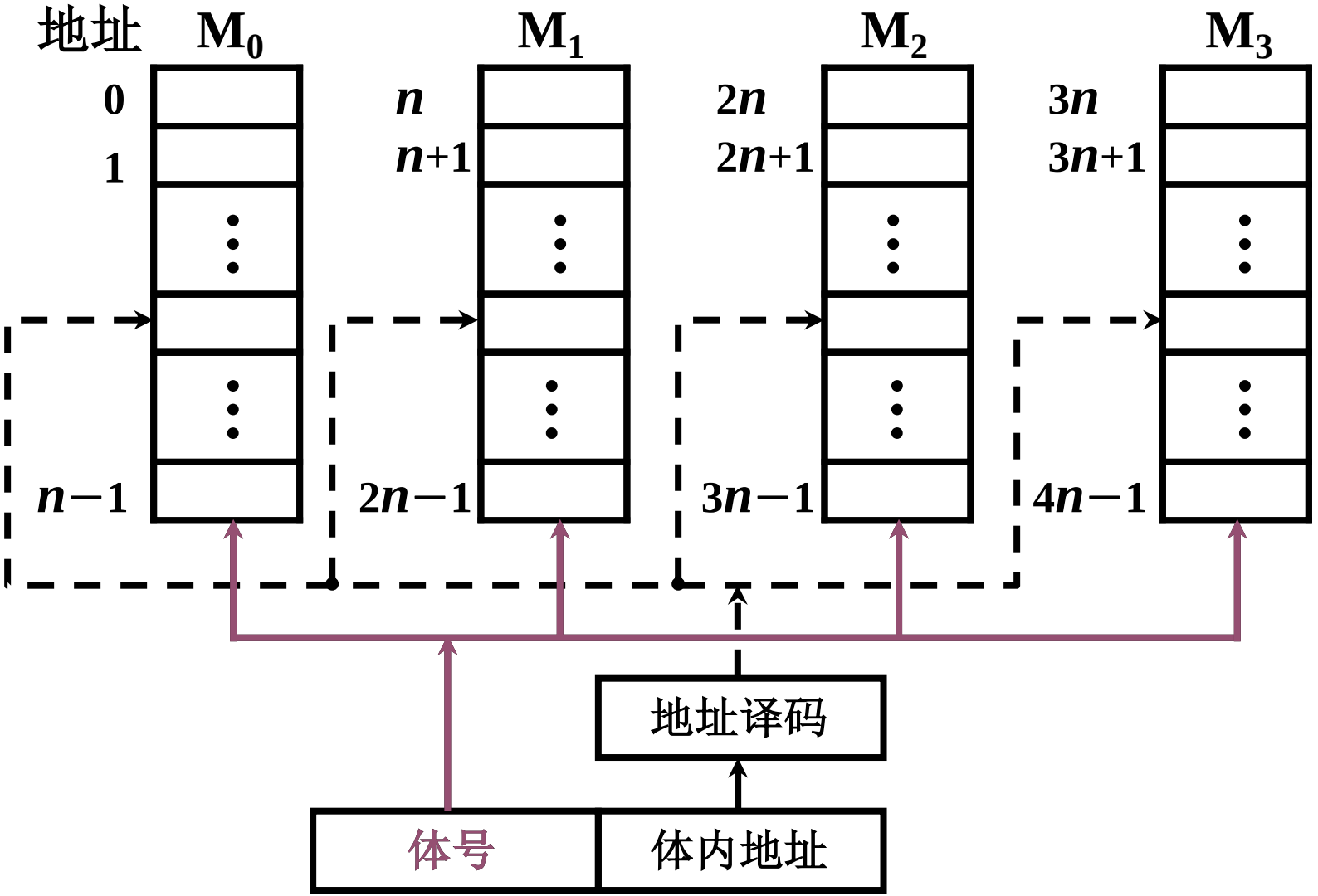
增加存储器的带宽



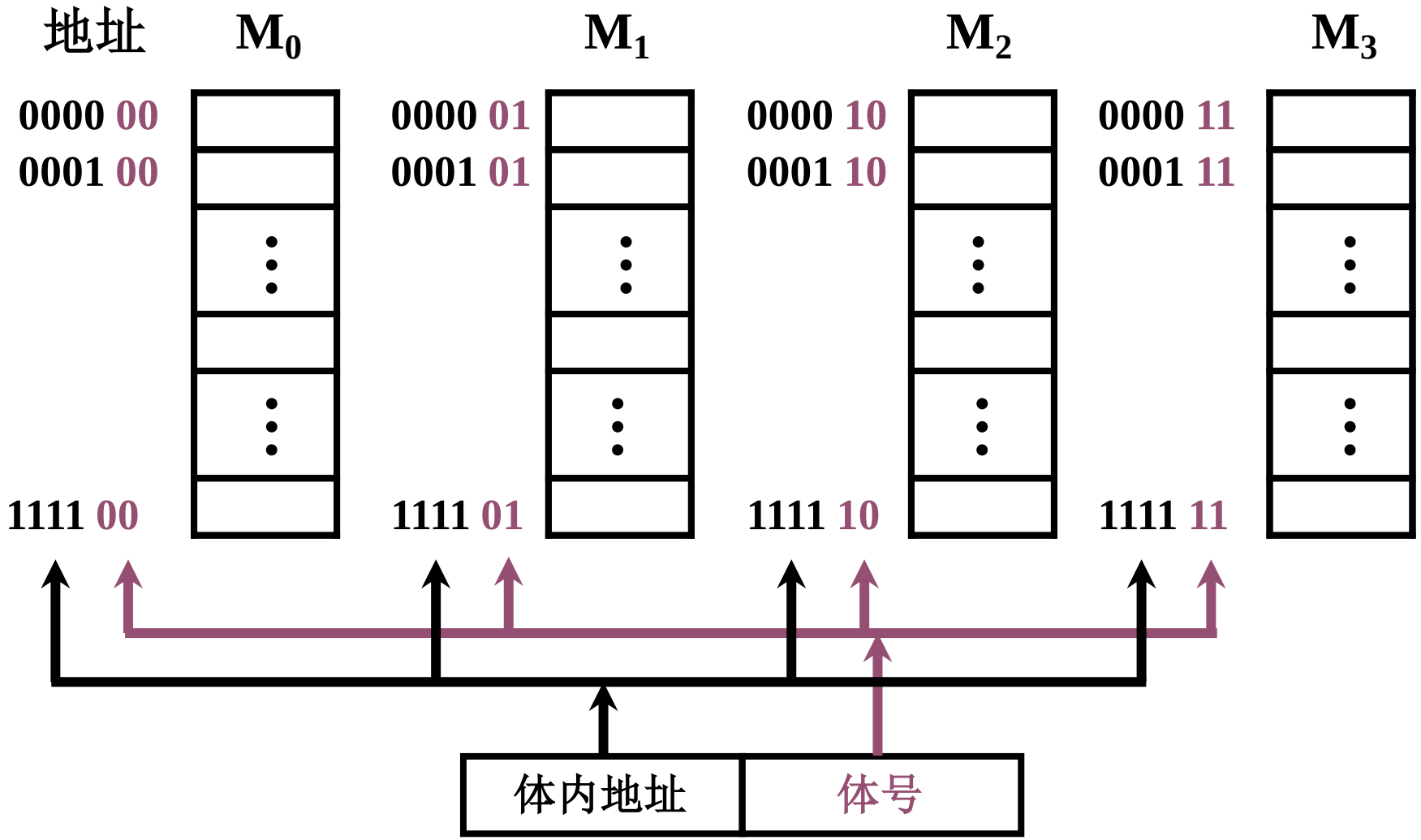
2. 多体并行系统—(1)高位交叉　顺序编址



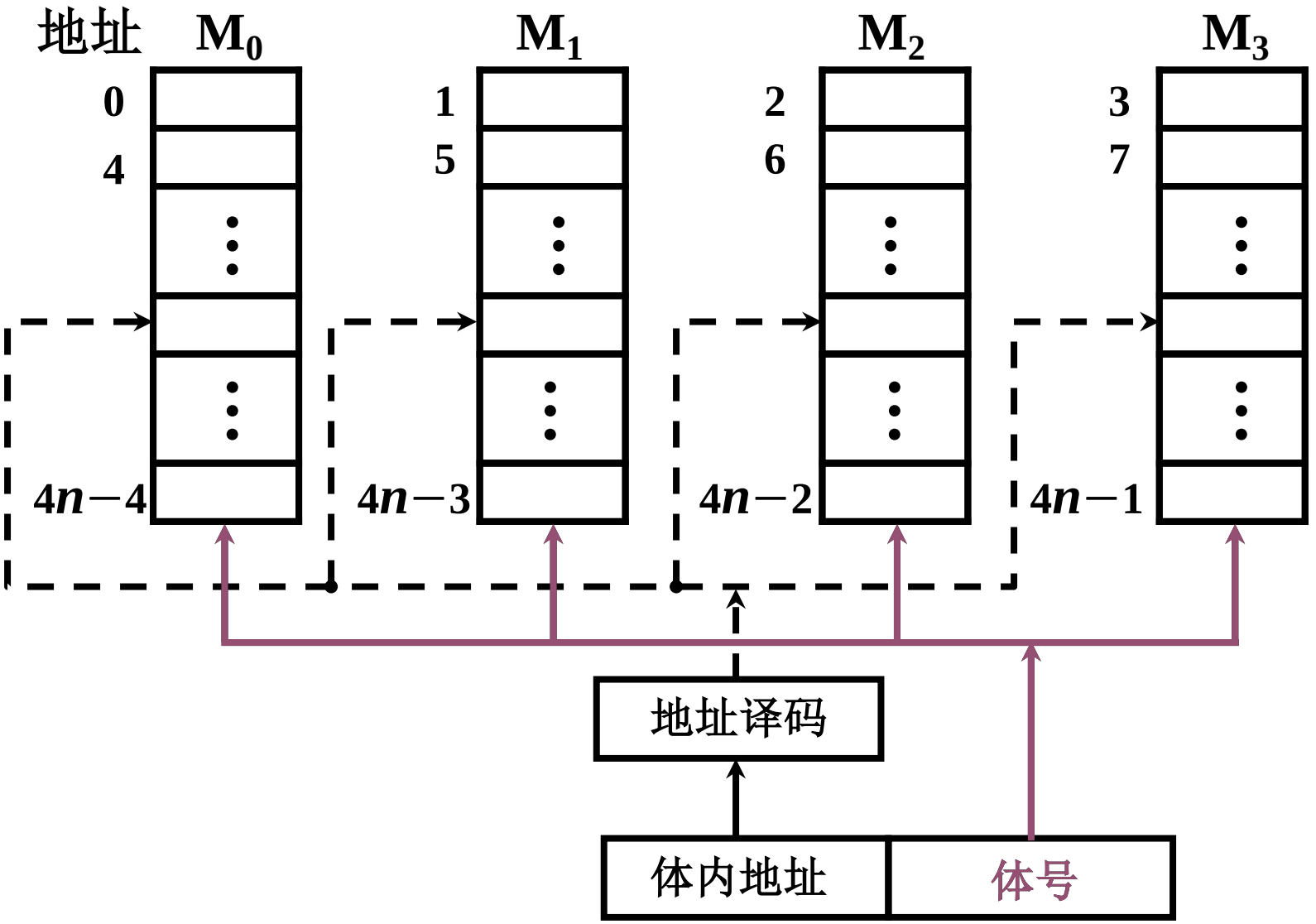
(1) 高位交叉——各个体并行工作



(2) 低位交叉——各个体轮流编址

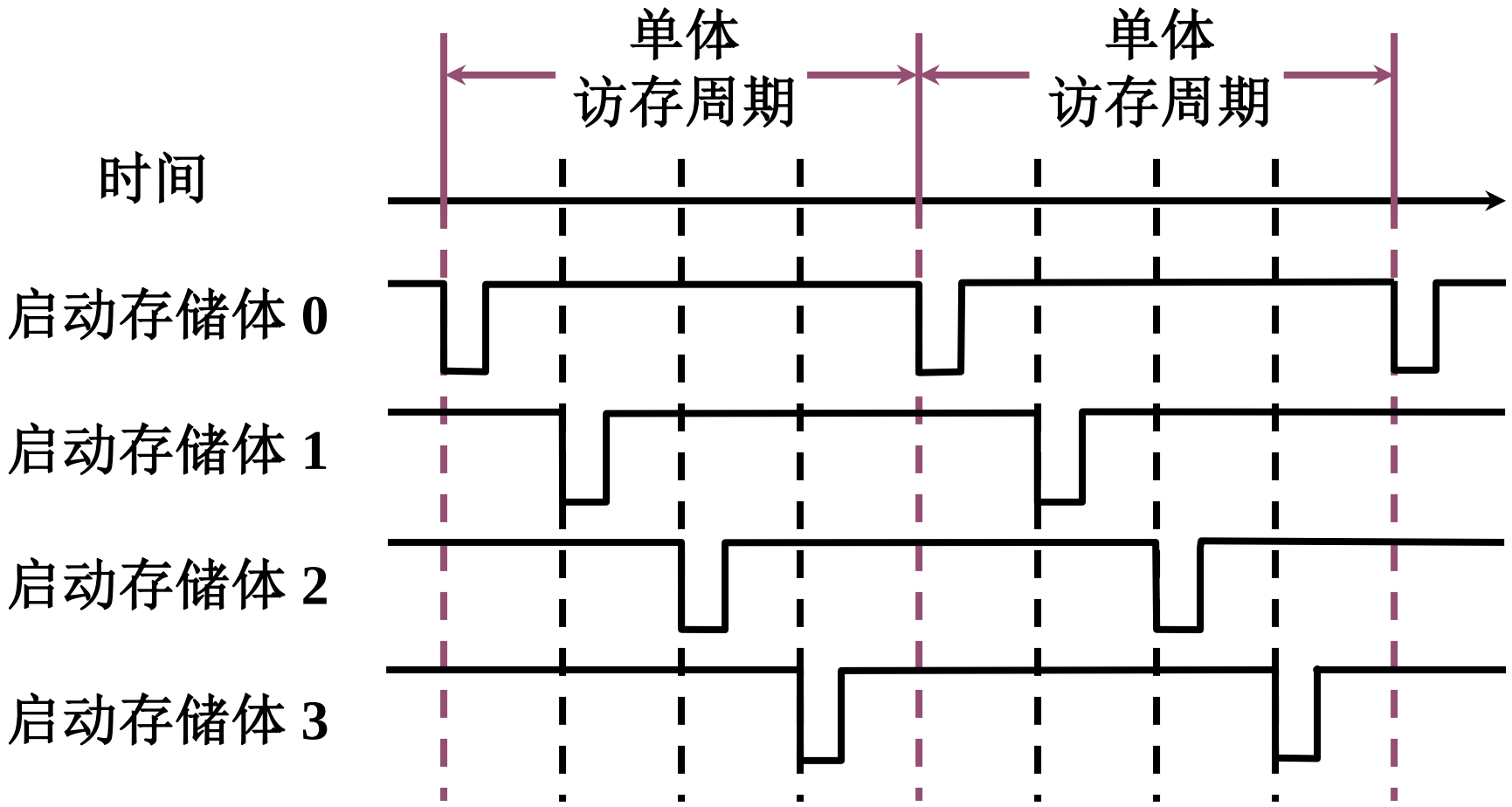


(2) 低位交叉——各个体轮流编址



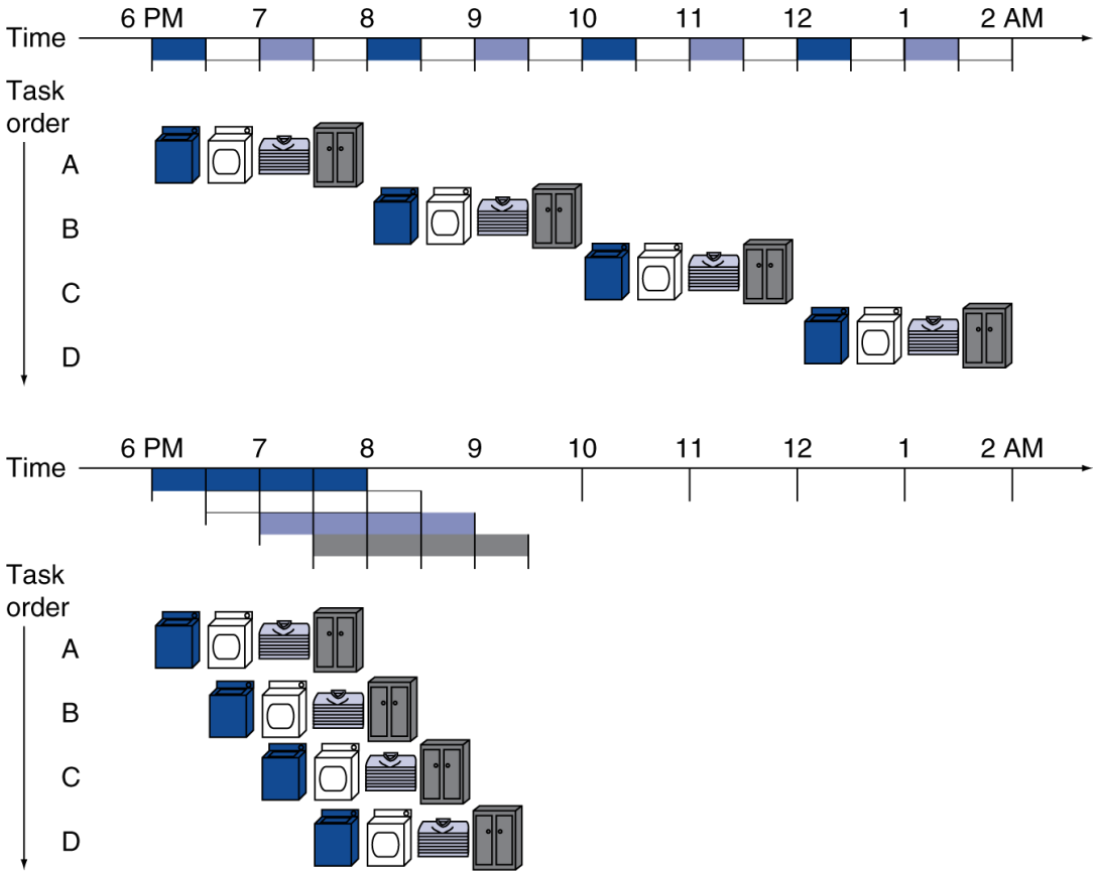
低位交叉的特点

在不改变存取周期的前提下，增加存储器的带宽



流水线的概念介绍：一个比喻

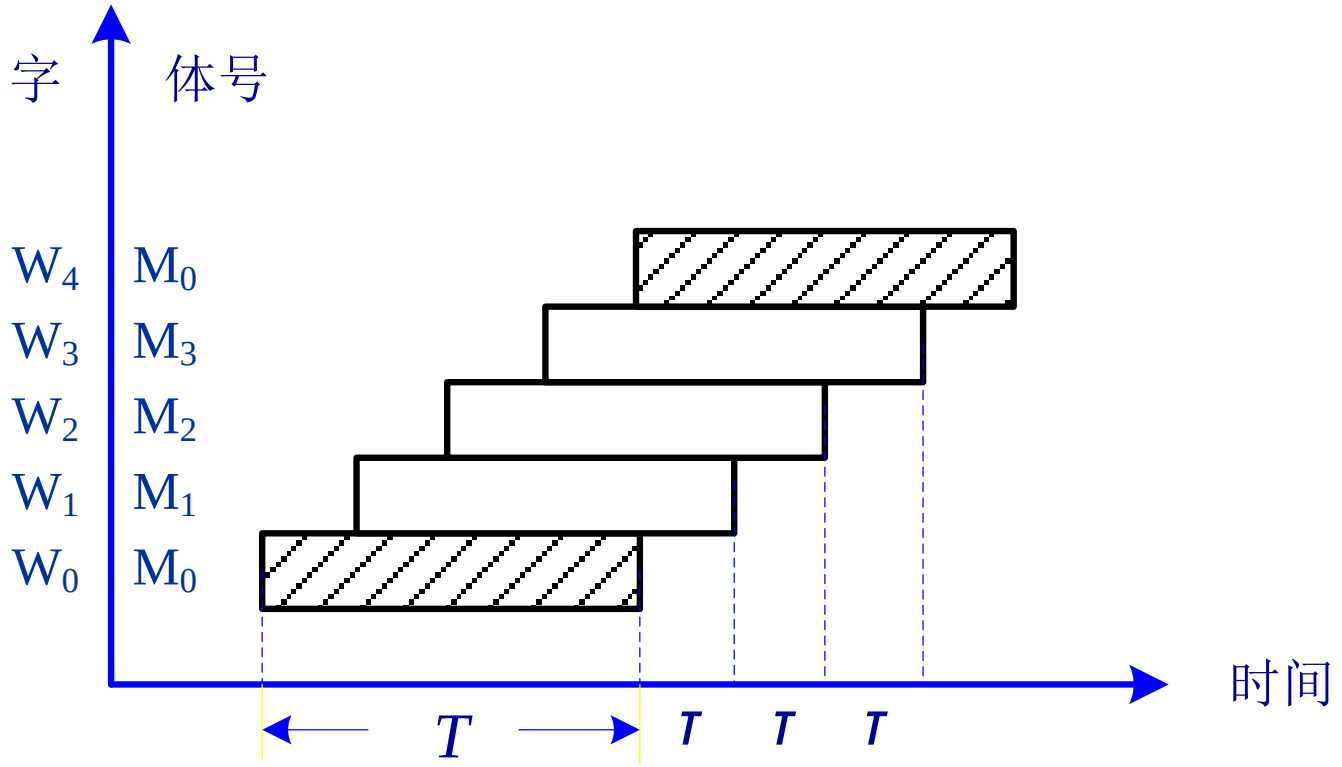
- 以洗衣服为例类比流水线：重叠执行
  - 假设洗衣包括四个步骤：洗衣机中洗衣、烘干机中烘干、叠衣服、收纳到柜子中



- 负载为4:
  - 加速比
    = 8/3.5 = 2.3
- 负载足够多:
  - 加速比
    = 2n/[2+(n-1)*0.5]≈ 4
    = 流水线中步骤的数目

流水线方式存取

- 设四体低位交叉存储器，存取周期为T，总线传输周期为τ，为实现流水线方式存取，应满足 $T = 4\tau$。



连续读取 4 个字所需的时间为 $T+(4-1)\tau$

例 4.6 设有四个模块组成的四体存储器结构，每个体的存储字长为 32 位，存取周期为 200ns。假设数据总线宽度为 32 位，总线传输周期为 50ns，试求顺序存储和交叉存储的存储器带宽。

解：

顺序存储（高位交叉编址）和交叉存储（低位交叉编址）连续读出 4 个字的信息量是 32×4=128 位。

顺序存储存储器连续读出 4 个字的时间是 $200ns \times 4 = 800ns = 8 \times 10^{-7}s$

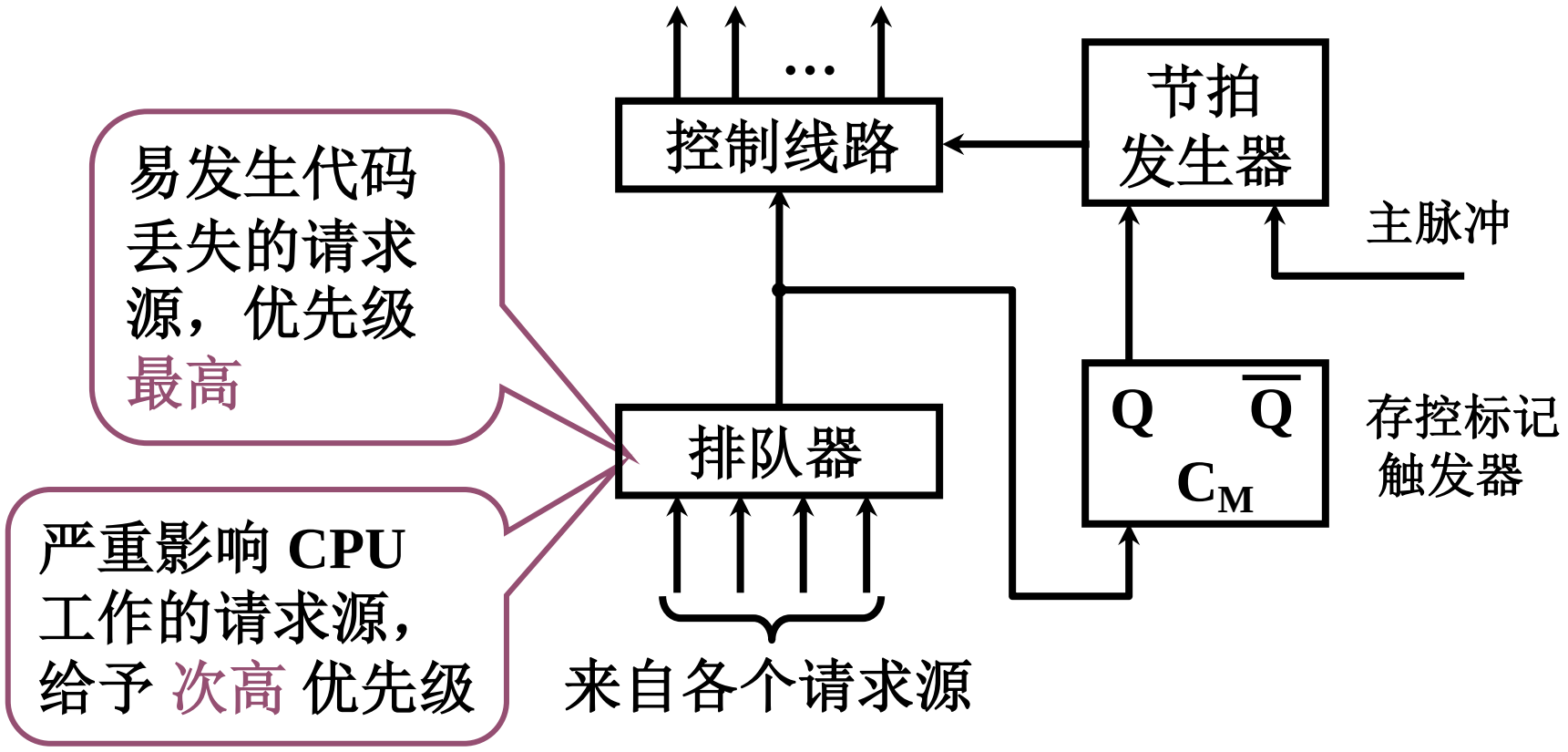
交叉存储存储器连续读出 4 个字的时间是 $200ns + 50ns \times (4-1) = 350ns = 3.5 \times 10^{-7}s$

顺序存储器的带宽是 $128/(8 \times 10^{-7}) = 16 \times 10^{7}bps$

交叉存储器的带宽是 $128/(3.5 \times 10^{-7}) = 37 \times 10^{7}bps$

(3) 存储器控制部件（存控）

- 见教材106页



4. 2主存储器

- 3. 高性能存储芯片
  - SDRAM (同步 DRAM)
    - 在系统时钟的控制下进行读出和写入，CPU 无须等待
  - RDRAM
    - 主要解决 存储器带宽 问题
  - 带 Cache 的 DRAM
    - 在 DRAM 的芯片内集成了一个由 SRAM 组成的 Cache, 有利于猝发式读取

第四章 存储器

4.1 存储器概述

4.2 主存储器

4.3 高速缓冲存储器

4.4 虚拟存储器

4.5 辅助存储器

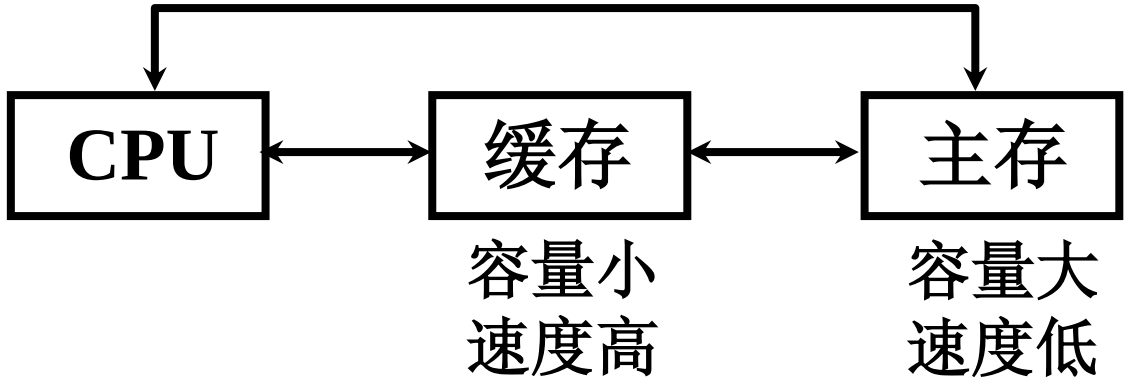
4. 3高速缓冲存储器

一、概述

1. 问题的提出

避免 CPU“空等” 现象（I/O设备访问优先级高于CPU访存）

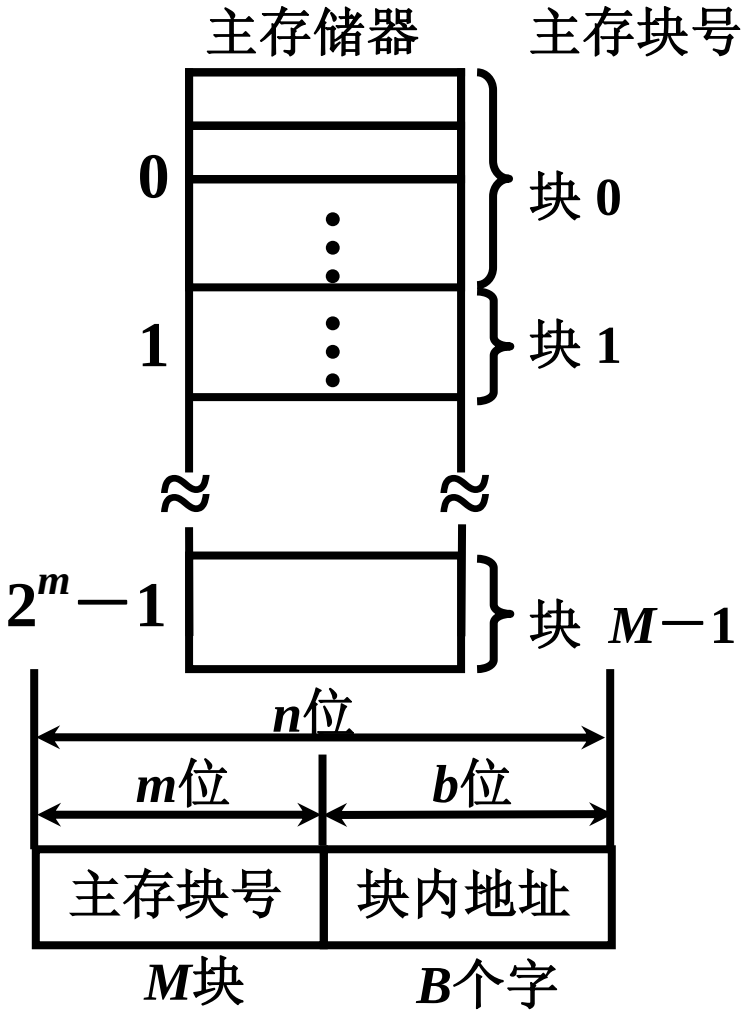
CPU 和主存（DRAM）的速度差异

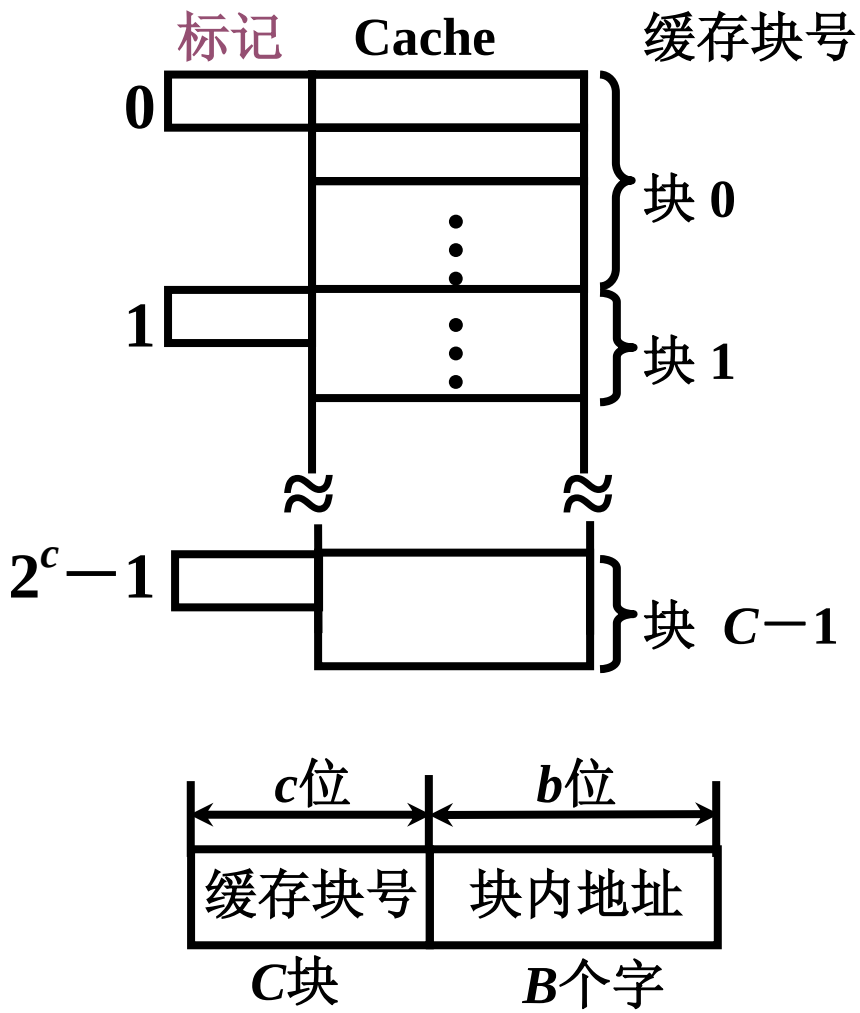


程序访问的局部性原理：1）时间局部性，当前正在使用的指令和数据在不久的将来还会被使用到。2）空间局部性，当前正在使用的指令和数据在不久的将来，其相邻的指令数据可能会被用到。

2. Cache 的工作原理

(1) 主存和缓存的编址





主存和缓存按块存储　　块的大小相同　　B 为块长

(2) 命中与未命中

缓存共有 C 块

主存共有 M 块　$M >> C$

命中　　主存块 调入 缓存

主存块与缓存块 建立 了对应关系

用 标记记录与某缓存块建立了对应关系的 主存块号

未命中　　主存块 未调入 缓存

主存块与缓存块 未建立 对应关系

(3) Cache 的命中率

CPU 欲访问的信息在 Cache 中的 比率

命中率 与 Cache 的 容量 与 块长 有关

一般每块可取 4 ~ 8 个字

块长取一个存取周期内从主存调出的信息长度

CRAY_1　16体交叉　块长取 16 个存储字

IBM 370/168　4体交叉　块长取 4 个存储字

（64位×4 = 256位）

(4) Cache –主存系统的效率

效率 e 与 命中率 有关

$$e = \frac{\text{访问 Cache 的时间}}{\text{平均访问时间}} \times 100\%$$

设 Cache 命中率 为 h，访问 Cache 的时间为 t_c，
访问 主存 的时间为 t_m

则 $$e = \frac{t_c}{h \times t_c + (1-h) \times t_m} \times 100\%$$

例 4.7 假设 CPU 执行某段程序时，共访问 Cache 2000 次，访问主存 50 次。已知 Cache 的存取周期为 50ns，主存的存取周期为 200ns。求 Cache—主存系统的命中率、效率和平均访问时间。

解：

（1）Cache 的命中率为 2000/(2000+50) = 0.97

（2）由题可知，访问主存的时间是访问 Cache 时间的 4 倍（200/50 = 4）

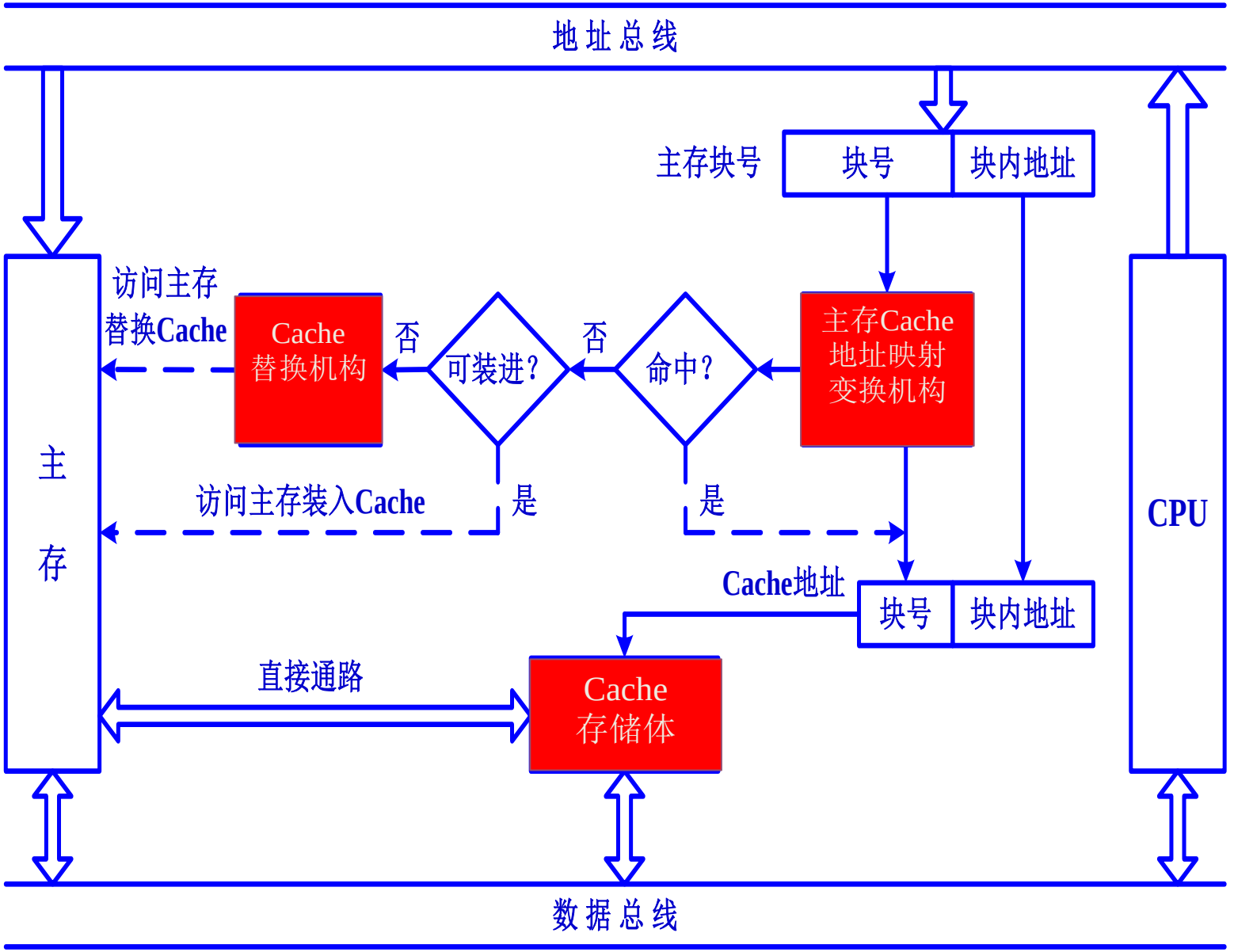
设访问 Cache 的时间为 t，访问主存的时间为 $4t$，Cache—主存系统的访问效率为 e，则

$$e = \frac{\text{访问Cache的时间}}{\text{平均访问时间}} \times 100\%$$

$$= \frac{t}{0.97 \times t + (1-0.97) \times 4t} \times 100\% = 91.7\%$$

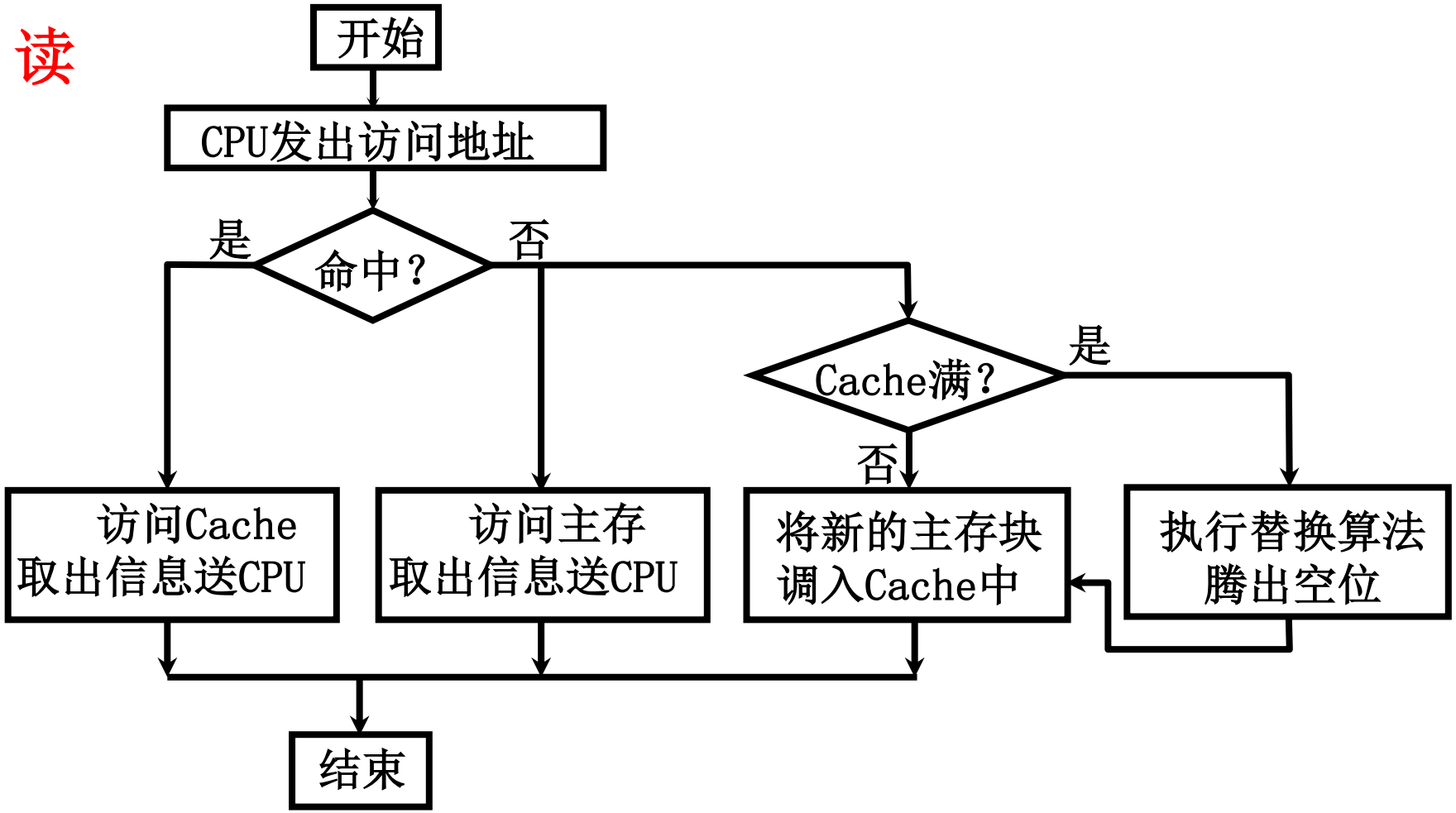
（3）平均访问时间 = 50ns×0.97＋200ns×(1–0.97) = 54.5ns

3. Cache 的基本结构



4.3高速缓冲存储器

- Cache 的 读写 操作
  - 读



4. Cache 的 读写 操作

写　Cache 和主存的一致性

- **写直达法（Write – through）**

  写操作时数据既写入Cache又写入主存

  写操作时间就是访问主存的时间，读操作时不涉及对主存的写操作，更新策略比较容易实现

- **写回法（Write – back）**

  写操作时只把数据写入 Cache 而不写入主存

  当 Cache 数据被替换出去时才写回主存

  写操作时间就是访问 Cache 的时间，

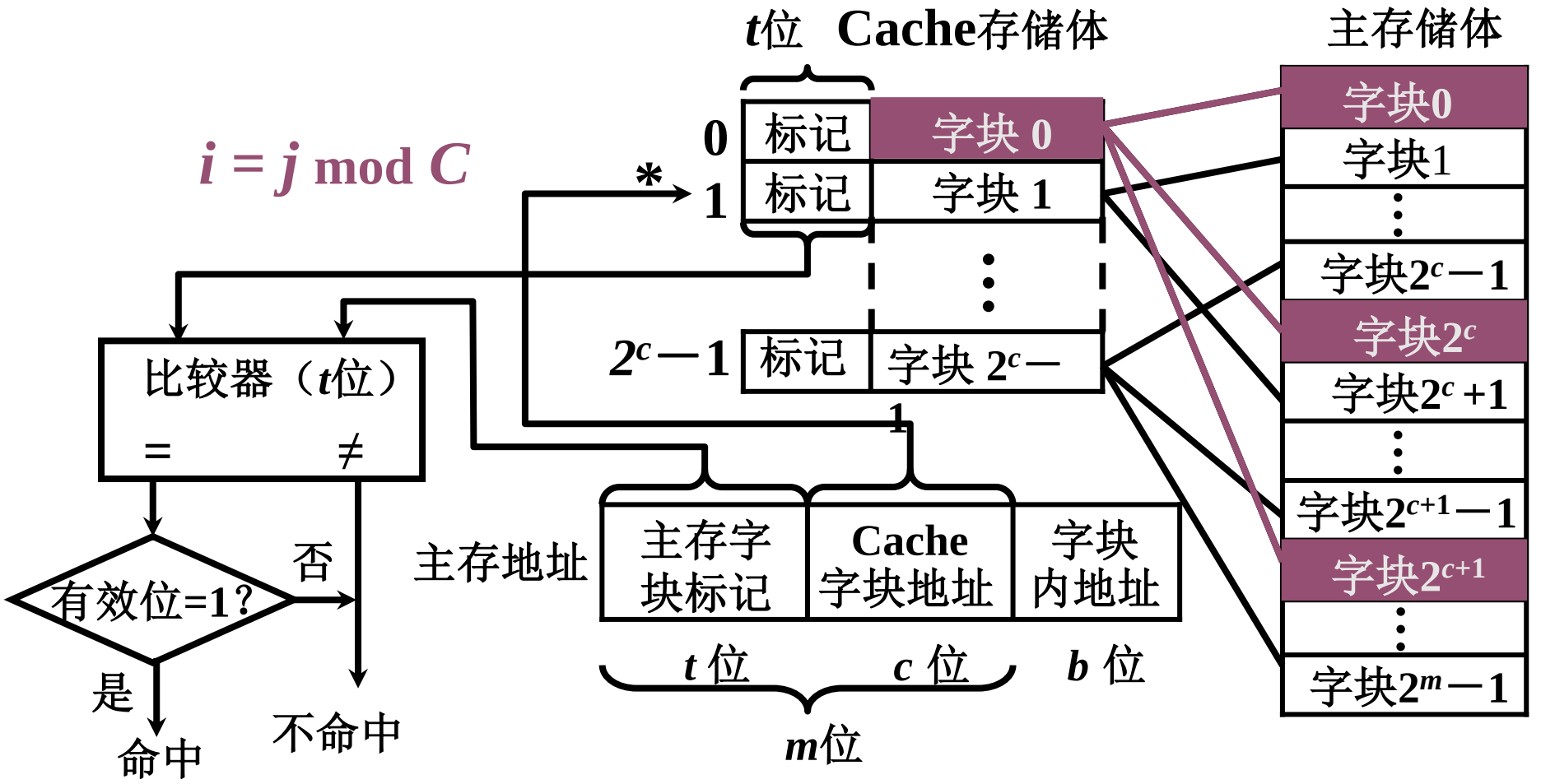
  读操作 Cache 失效发生数据替换时，

  被替换的块需写回主存，增加了 Cache 的复杂性

4. 3高速缓冲存储器

- 二、Cache – 主存的地址映射
  - 直接映射
  - 全相联映射
  - 组相联映射

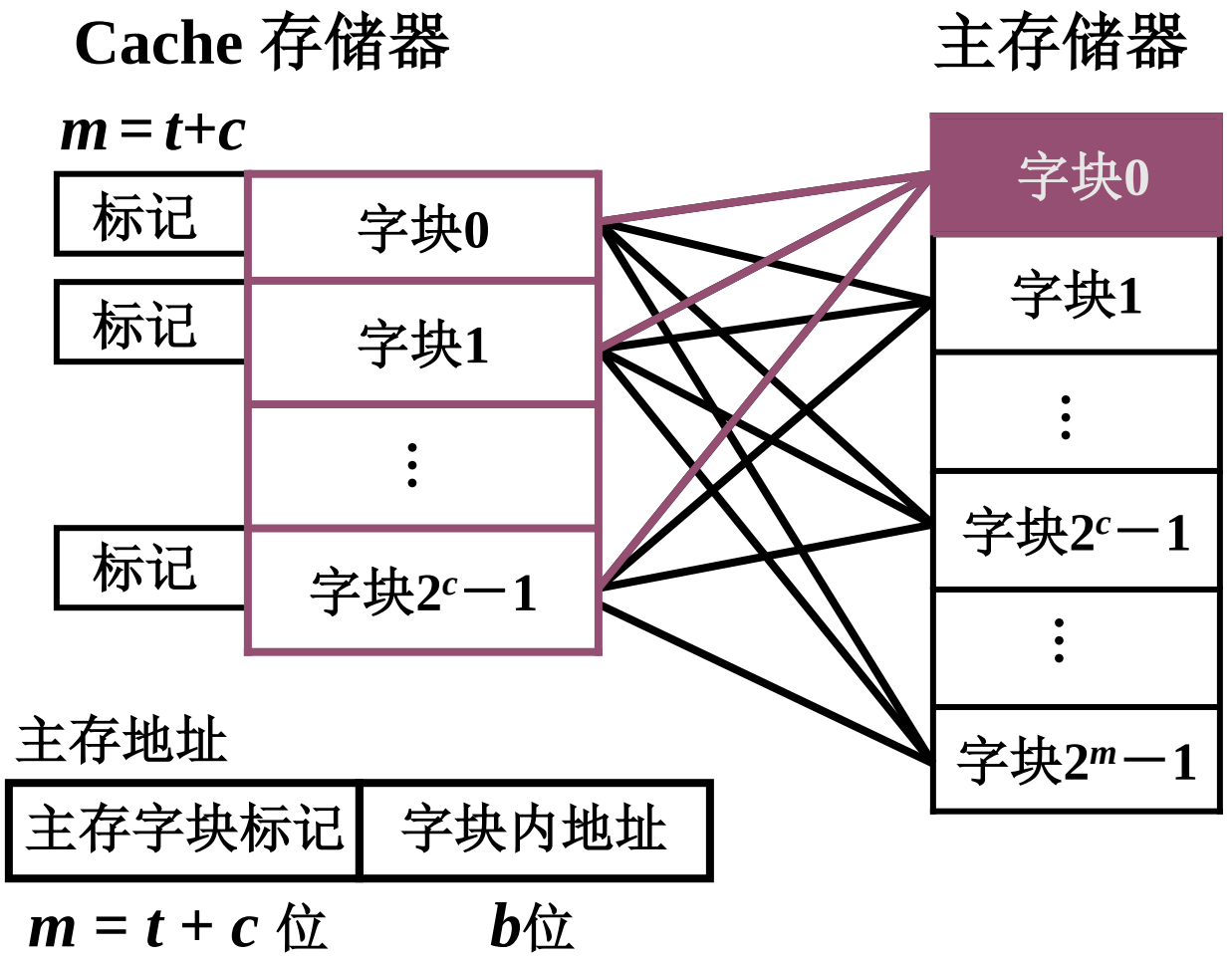
1. 直接映射



每个缓存块 i 可以和 若干 个 主存块 对应

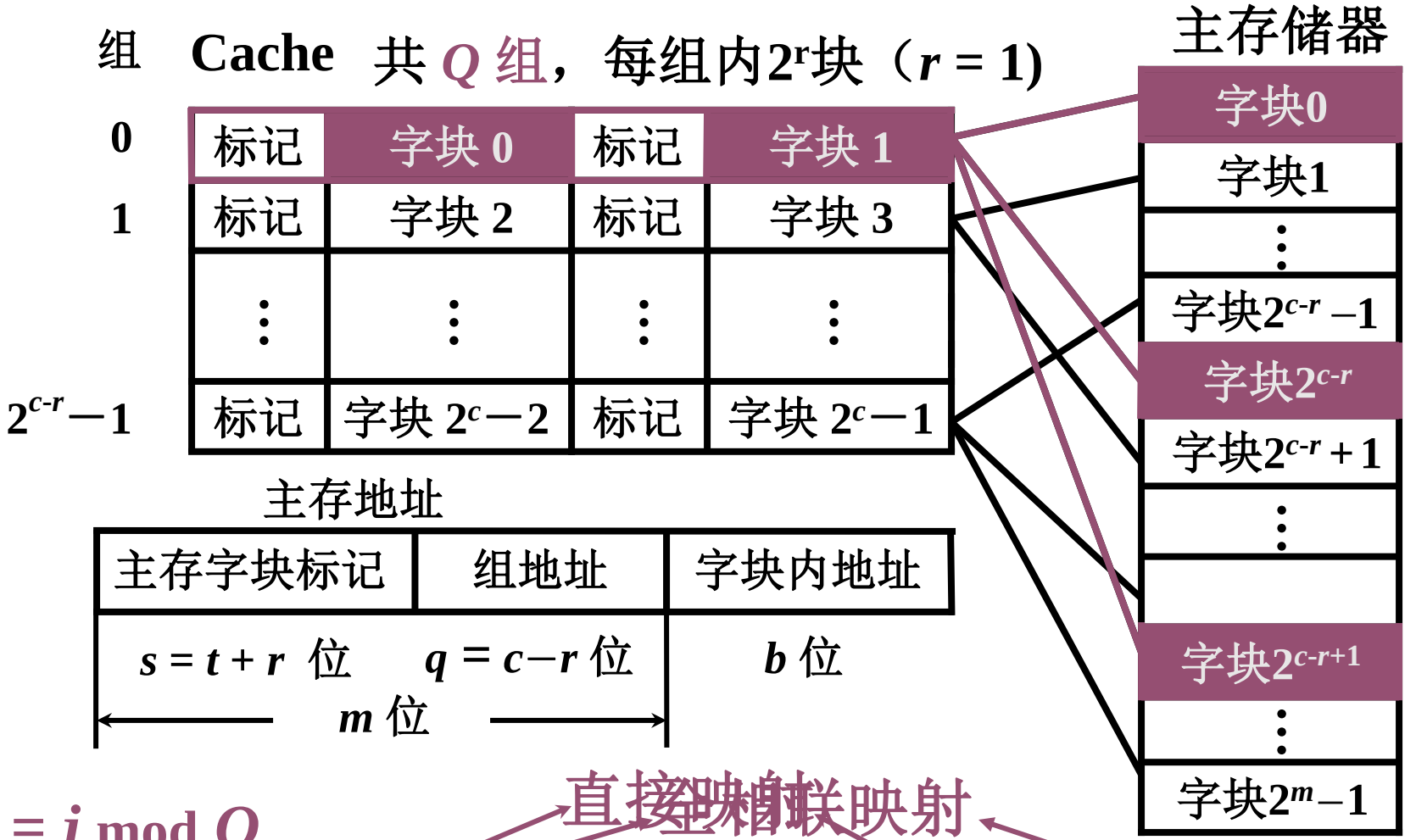
每个主存块 j 只能和 一 个 缓存块 对应

2. 全相联映射



主存 中的 任一块 可以映射到 缓存 中的 任一块

3. 组相联映射



$i = j \bmod Q$

直接映射 全相联映射

某一主存块 j 按模 Q 映射到 缓存 的第 i 组中的 任一块

例4. 8

假设主存容量为512KB，Cache容量为4KB，每个字块为 16个字，每个字32位。

1) Cache 地址有多少位？可容纳多少块？

Cache 容量 4KB—>地址 **12** 位。 4KB/(4B*16) =**64** 块

2) 主存地址有多少位？可容纳多少块？

512KB—>主存地址**19**位。 512KB/(4B*16) = **8192** 块。

3) 在直接映射方式下，主存的第几块映射到 Cache 中的第5 块（设起始字块为第 1 块）？

主存的第 5，64 + **5**，2×64 + **5**， … ， 8192 – 64 + **5** 块

例4. 8——续

(主存512KB，Cache 4KB，每个字块16个字，每个字32位)

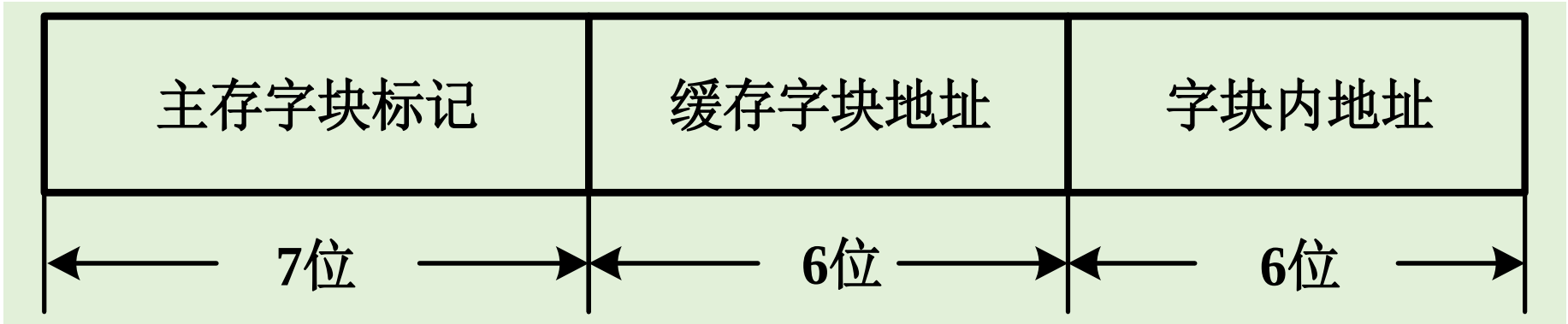
4）画出直接映射方式下主存地址字段中各段的位数

(主存字块标记+缓存字块地址+字块内地址)

字块内地址为 **6** 位（字块大小为：4B*16=64B）

缓存共 64 块，故缓存字块地址为 **6** 位

主存字块标记为主存地址与Cache 地址长度之差**19-12=7**位。



例子4. 9

- 假设主存容量 512K×16 位，Cache 容量 4096×16 位，每个块为4个16位的字，访存地址为字地址。

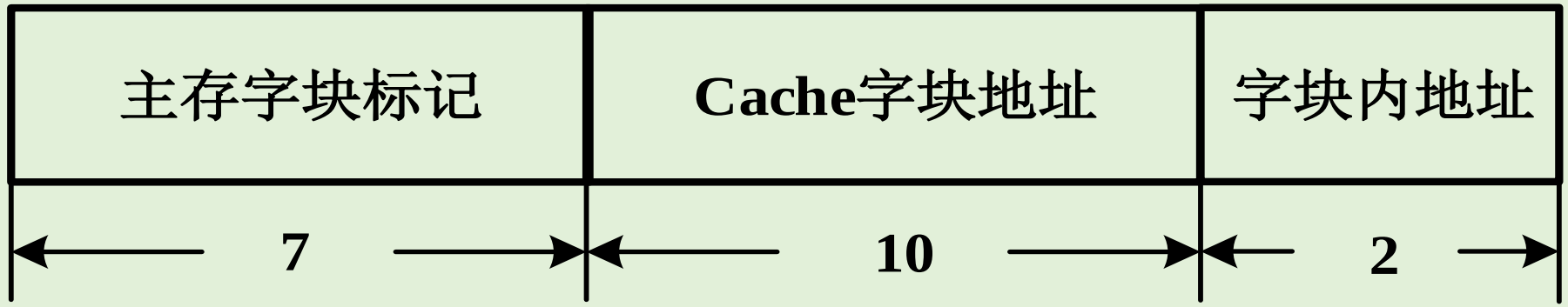
1）在直接映射方式下，设计主存的地址格式。

根据 Cache 容量为 4K字，得 **Cache 字地址为 12 位。**

根据块长为 4，**按字访问，得字块内地址 2 位，**

Cache 共有 4K/4 = 1024 块（10位）。

根据主存容量为 512K，得**主存字地址 19 位。**



例4.9 续

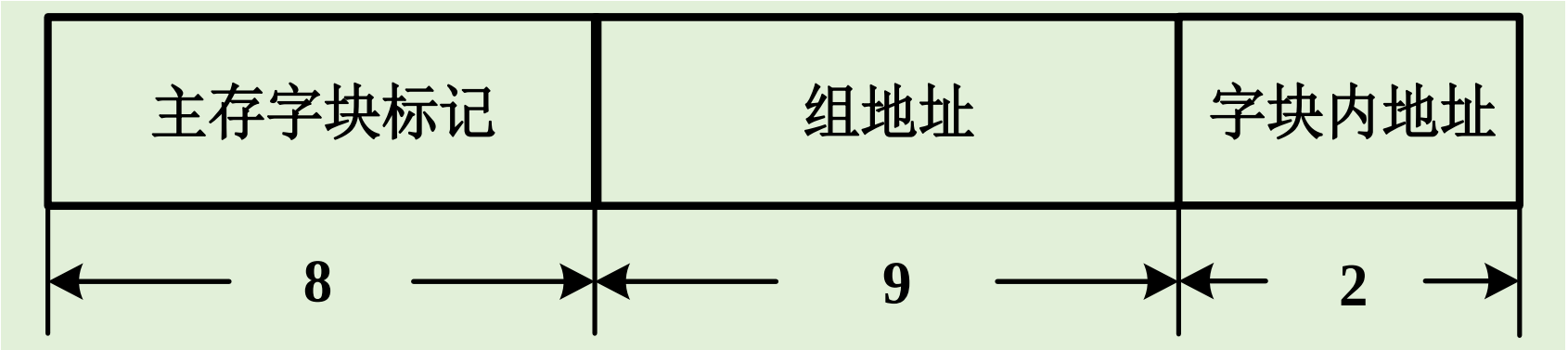
- 假设主存容量 512K×16 位，Cache 容量 4096×16 位，每个块为4个16位的字，访存地址为字地址。

2）在全相联映射方式下，设计主存的地址格式。

字块内地址 **2** 位，其余 **17** 位为主存字块标记

3）在二路组相联映射方式下，设计主存的地址格式。

一组内有 **2** 块，Cache有**1024块** ——>组地址 **9** 位

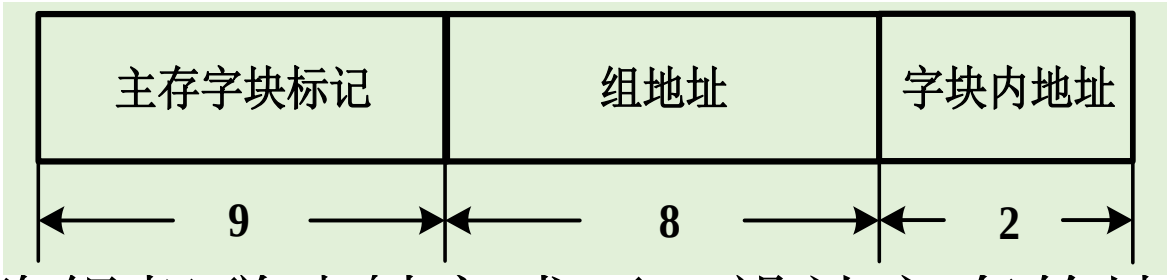


例4.9 续

- 假设主存容量 512K×16 位，Cache 容量 4096×16 位，每个块为4个16位的字，访存地址为字地址。

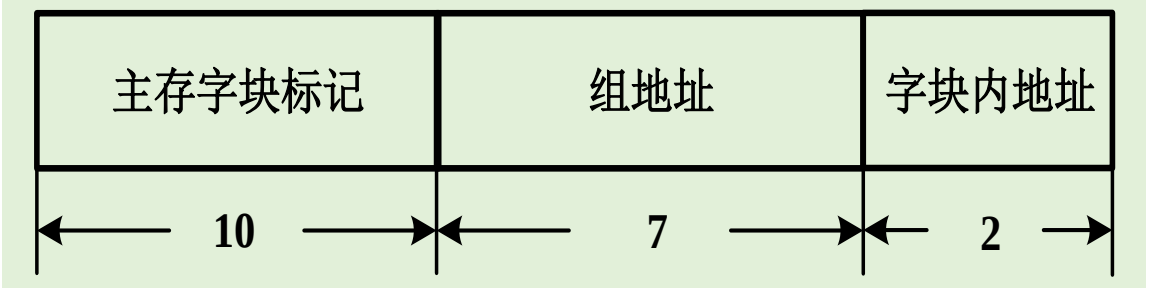
4）在四路组相联映射方式下，设计主存的地址格式。

一组内有 **4** 块，Cache有**1024块** ——>组地址 **8** 位



5）在八路组相联映射方式下，设计主存的地址格式。

一组内有 **8** 块，Cache有**1024块** ——>组地址 **7** 位



填空题 4分

设置

假设主存容量 512K×**32** 位，Cache 容量 4096×16 位，每个块为4个**16位的字**，访存地址为**字地址**。

在四路组相联映射方式下，

1. 主存地址为[填空1] 位，
2. 字块内地址为 [填空2] 位，
3. 组地址为 [填空3] 位，
4. 主存字块标记为[填空4] 位。

正常使用填空题需3.0以上版本雨课堂

作答

练习解析

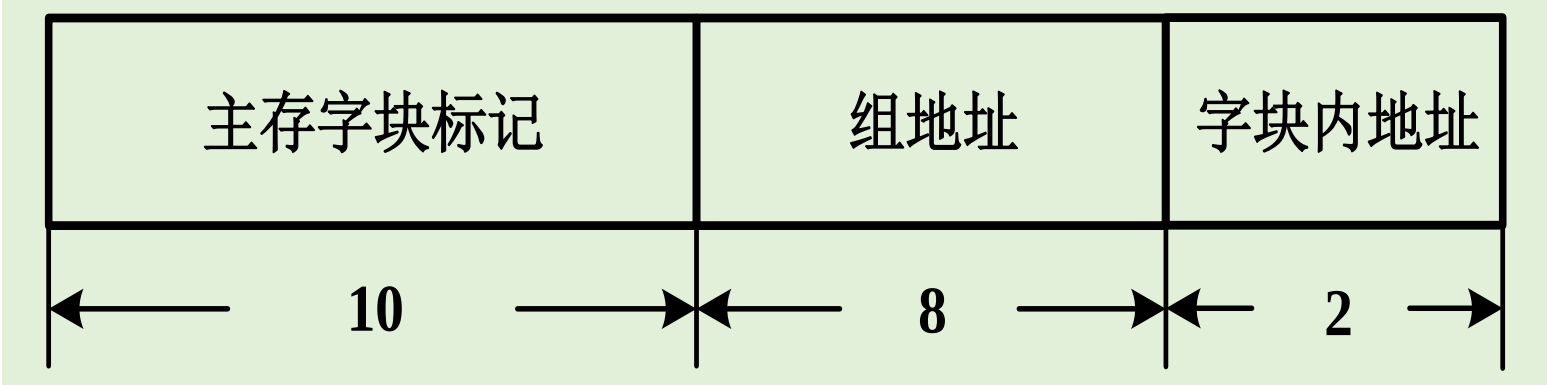
- 假设主存容量 512K×**32** 位，Cache 容量 4096×16 位，每个块为4个16位的字，访存地址为**字地址**。

在四路组相联映射方式下，设计主存的地址格式。

解：每块4个字—>**字块内地址为2位**

主存容量 512K×32 位—>1M×16 位 —>主存地址 20 位

四路组相联，一组内4块，Cache共有1024/4 = 256 组—>8位



4.3高速缓冲存储器——缓存替换算法

- 先进先出法（FIFO：First In First Out）
- 近期最少使用法（LRU：Least Recently Used）
- 最不经常使用法（LFU：Least Frequently Used）
- 随机替换法

4.3高速缓冲存储器——小结

- **实际应用：** 块设备缓存、硬盘缓存、web cache...

不灵活

直接映射　　某一 主存块 只能固定 映射到 某一 缓存块

全相联映射　某一 主存块 能 映射到 任一 缓存块

组相联映射　某一 主存块只能映射某一缓存组中的任一块

成本高

第四章 存储器

4.1 存储器概述

4.2 主存储器

4.3 高速缓冲存储器

4.4 虚拟存储器

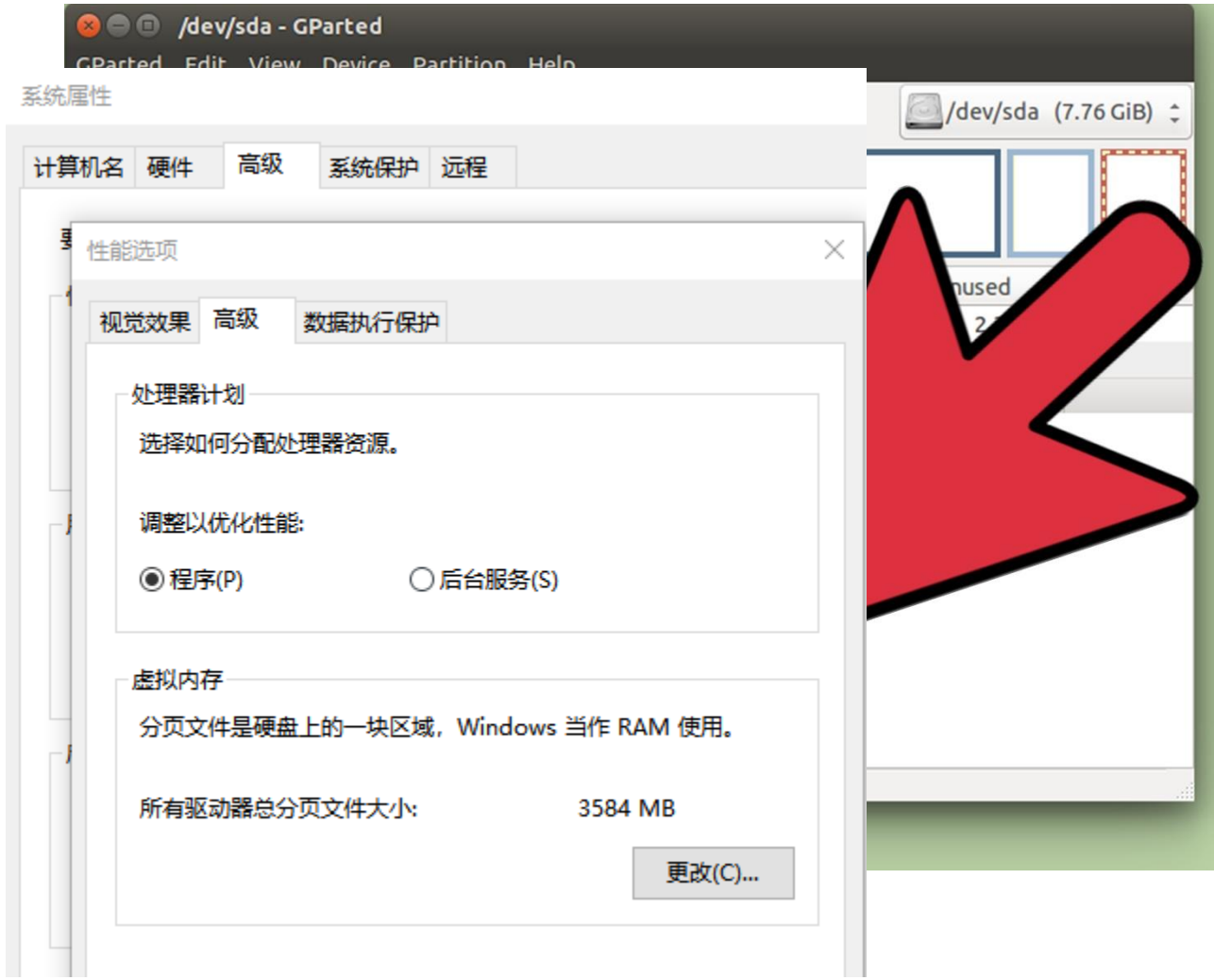
4.5 辅助存储器

4. 4虚拟存储器

- **“640K ought to be enough for anybody”比尔·盖茨？(1981)**
- 如何在有限的主存空间运行**较多**的**较大**的用户程序？
  - 初衷: 采用虚存来扩大寻址空间
  - 现在: 主要用于存储保护，程序共享
- 主存、辅存在OS和硬件的管理之下，提供比实际大得多的存储空间
- 虚存空间是靠辅存（磁盘）来支持的
- 虚拟存储采用与Cache类似原理，提高速度，降低成本
- 用户感觉到的是一个速度接近主存而容量极大的存储器

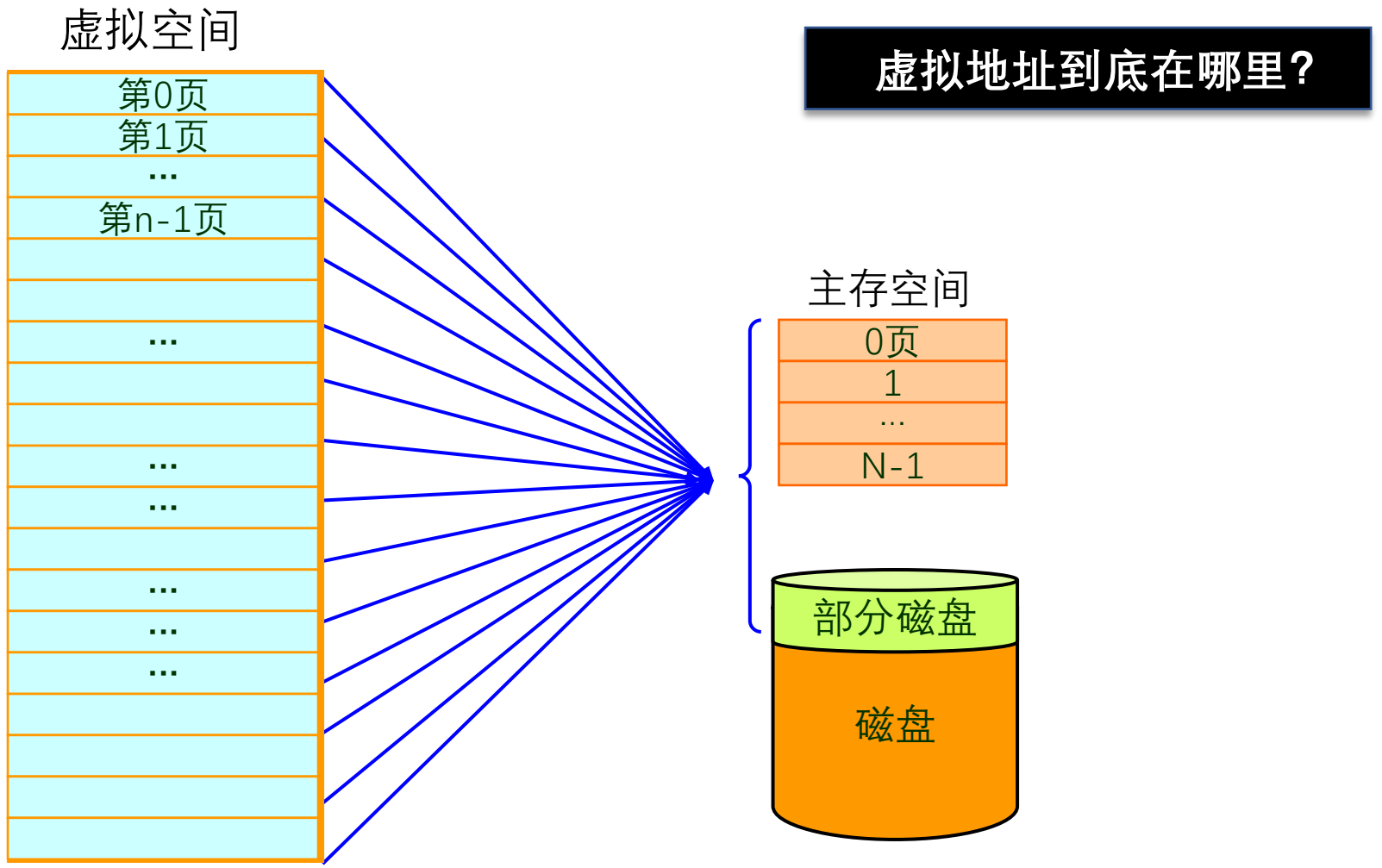
4.4虚拟存储器

- Linux
  - 交换分区swap
- Windows
  - 页面文件



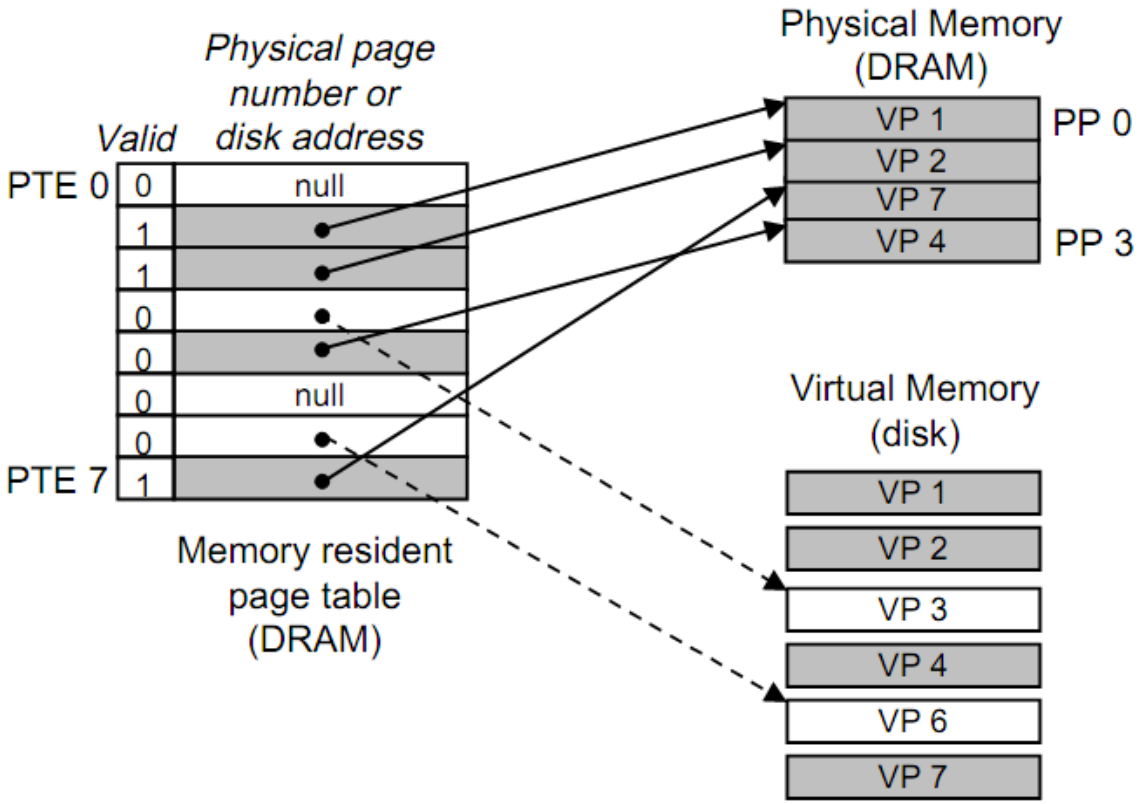
4.4虚拟存储器

- 页式虚拟地址空间



4.4虚拟存储器

- Page table(页表，查找表)

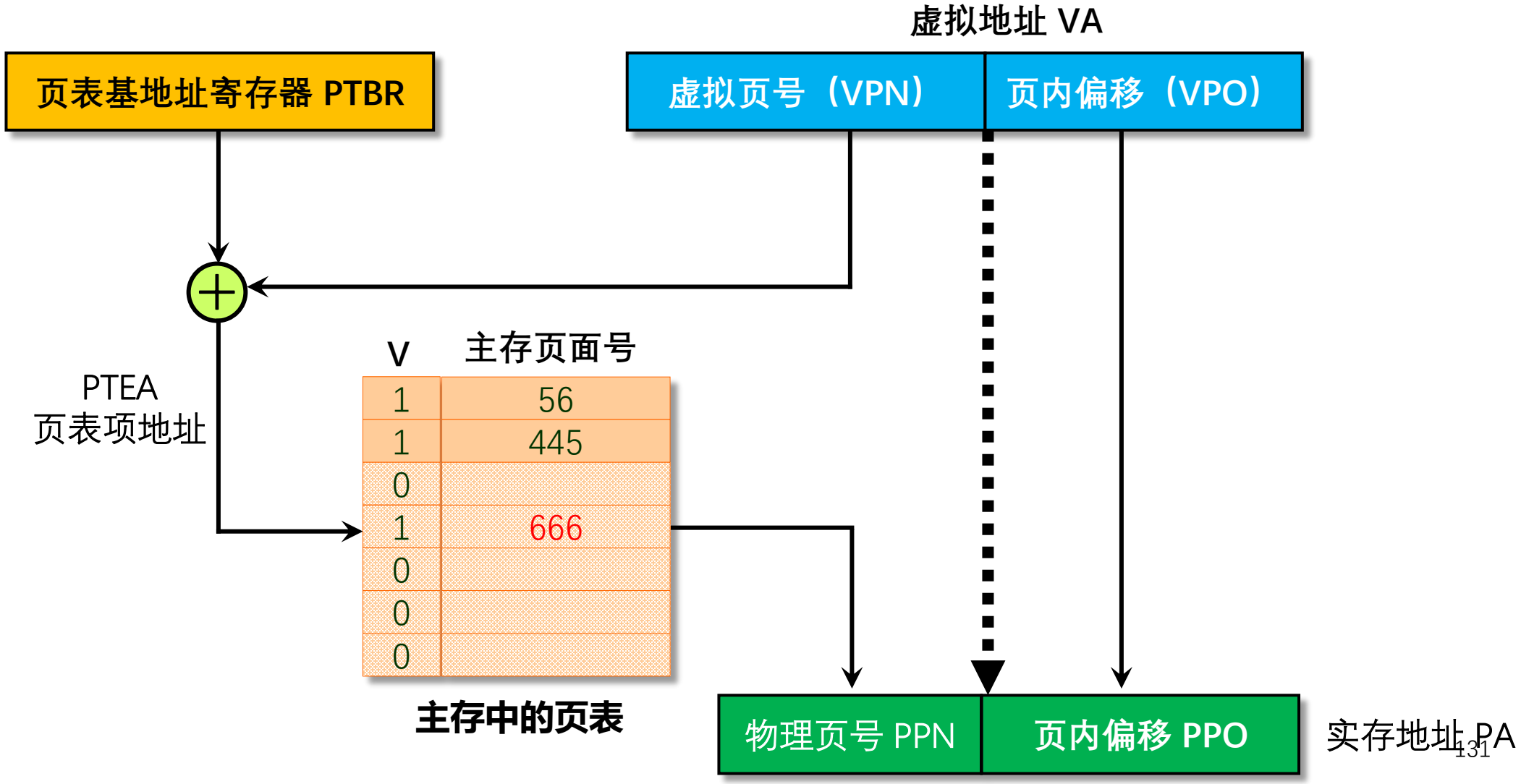


页表大小？页表存放哪里？ 页表有多少个?

进程切换时?

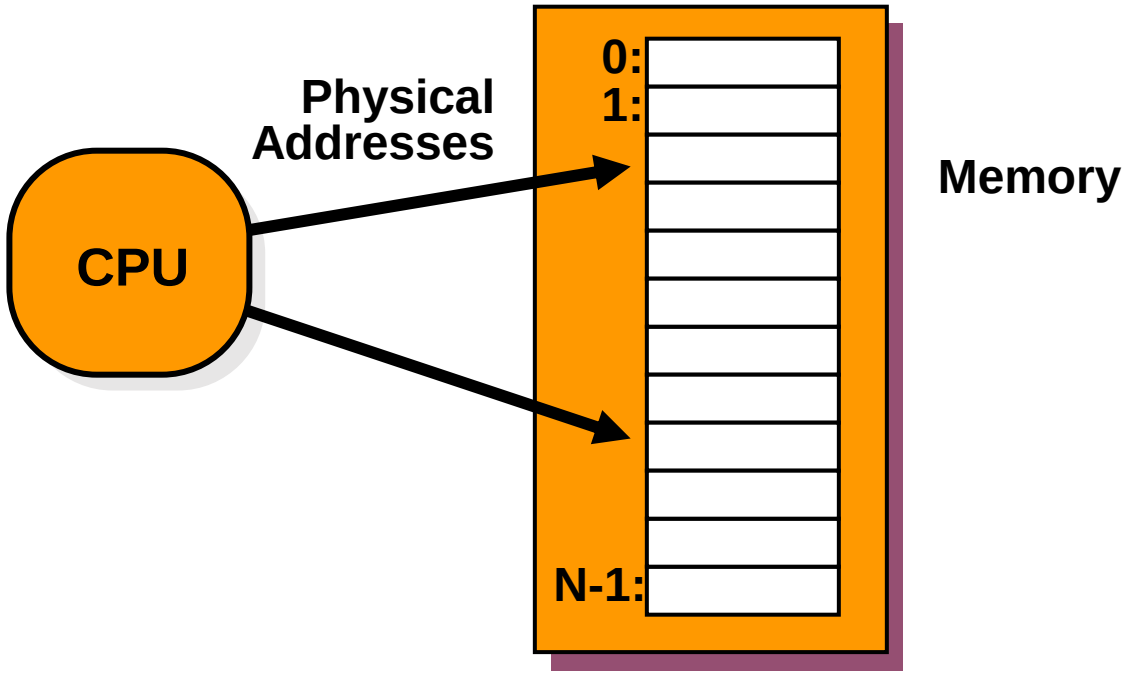
4.4虚拟存储器

- 页式虚拟存储器结构



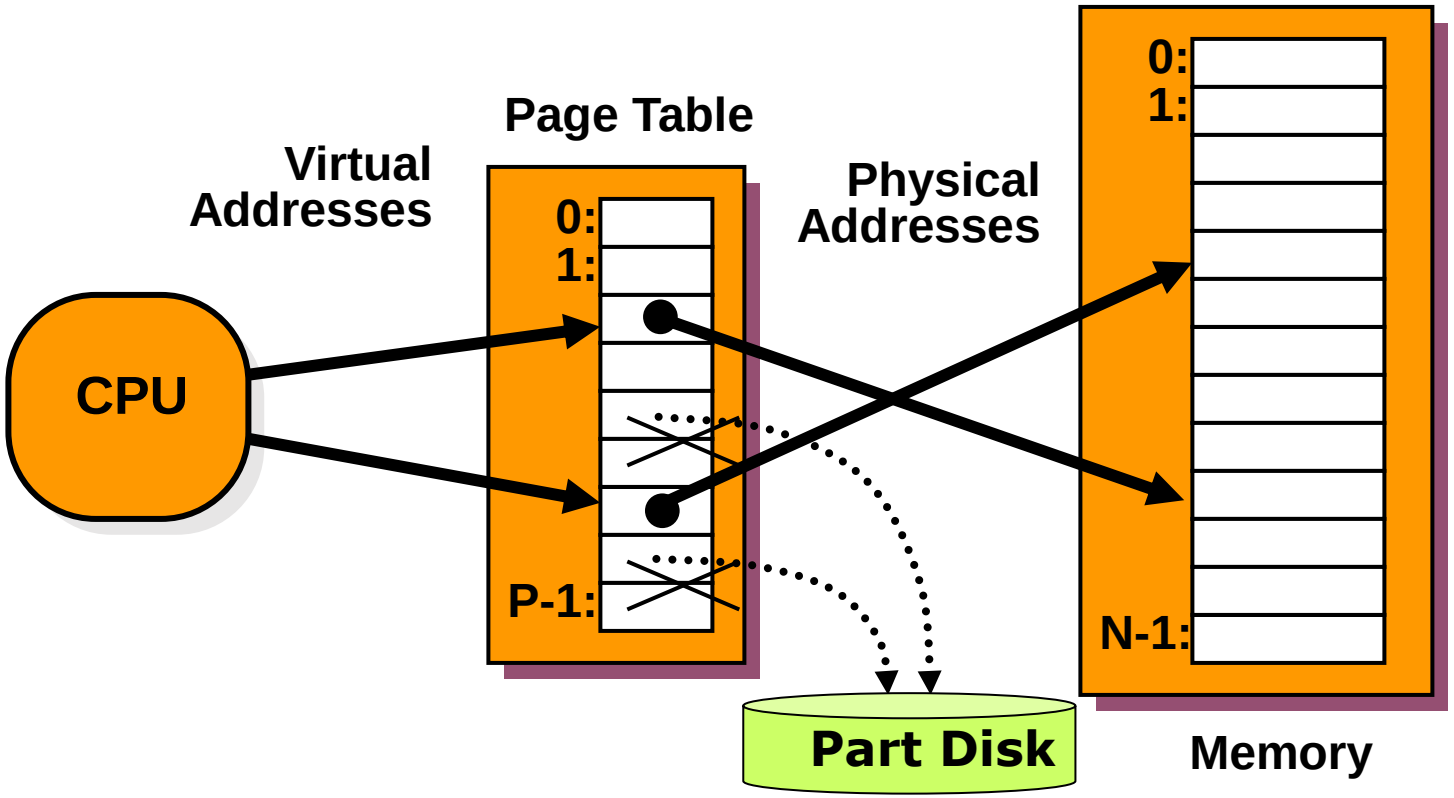
4. 4虚拟存储器

- 实地址计算机系统
  - CPU给出的地址直接访问物理内存
  - 大多数Cray计算机，早期PC，大多数嵌入式系统



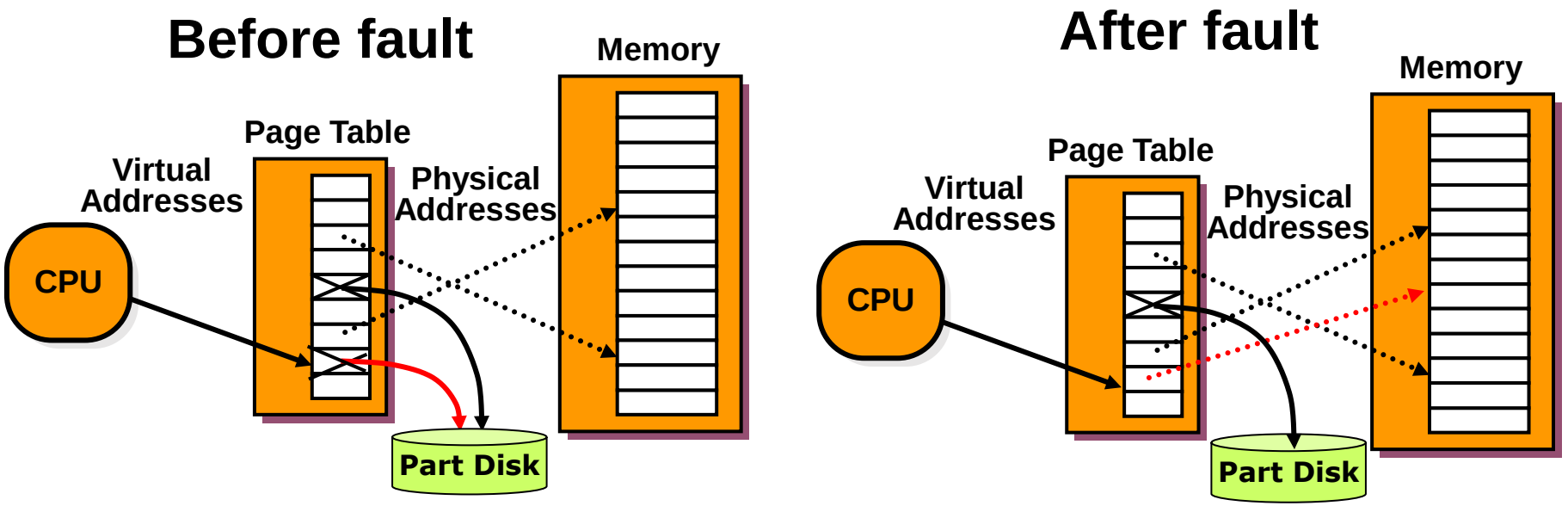
4.4虚拟存储器

- 虚地址计算机系统
  - CPU地址需要虚实变换
    - 硬件通过OS维护的页表将虚拟地址转换为物理地址
  - 工作站，服务器，现代PC



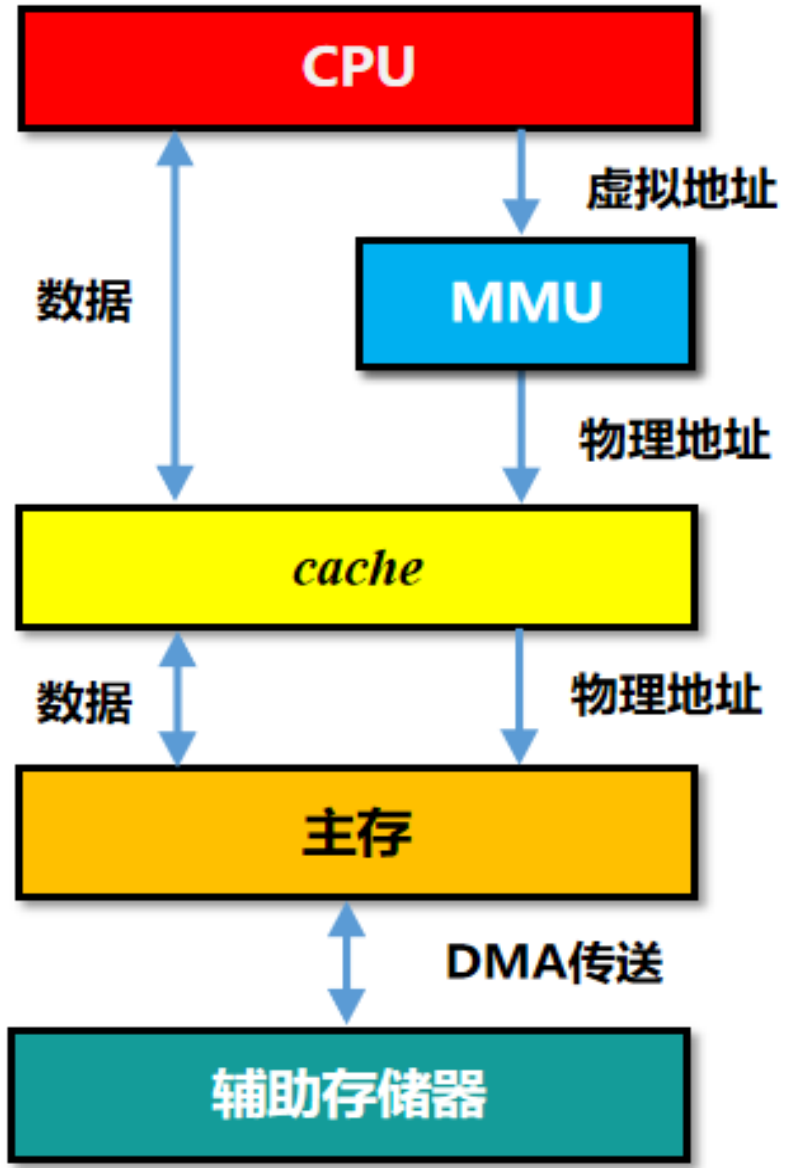
4.4虚拟存储器

- 缺页 Page Faults
  - 页表指示虚拟地址不在内存中
    - 操作系统负责将数据从磁盘迁移到内存中
    - 当前进程挂起
    - 操作系统负责所有的替换策略
    - 唤醒挂起进程



4.4虚拟存储器

- 层次结构



4.4虚拟存储器

- 虚拟存储器本质
  - 程序员在比实际主存大得多的逻辑地址空间中编程
  - 程序执行时，按需载入代码和数据，无需全部载入
  - 硬件将逻辑地址（虚拟地址）转化为物理地址（实地址）
  - 缺页时，由OS进行主存和磁盘之间的信息交换
  - 虚存机制由硬件与操作系统协作实现
    - 进程及进程上下文切换、存储器分配
    - 虚拟地址空间管理、缺页处理
    - 存储器保护

4.4虚拟存储器

- 与CPU cache的相似之处
  - 将程序中常用的部分驻留在高速存储器
    - 程序载入按需载入（不是全部载入）
    - 高速存储器分块或者分页（粒度问题）
    - 主存空间满，将不常用程序或数据淘汰或交换到辅存中
  - 数据的换入换出由硬件或操作系统完成，对用户透明
  - 利用程序局部性，使存储系统的性能接近于高速存储器；价格接近于低速存储器，并扩充主存容量

4.4虚拟存储器

- 与CPU Cache的差异
  - 虚存用于扩大主存容量，Cache用于加速主存性能
  - 虚存中未命中性能损失远大于Cache系统
    - 全相联提升命中率
    - 更大的交换单位页
    - 近似LRU算法 （CLOCK算法）
  - 虚存由硬件和OS联合管理，Cache由硬件管理

4.4虚拟存储器

- 段式管理
  - 段的分界与程序的自然分界相对应，段长不固定，因程序而异
  - 段的独立性—易于编译、管理、修改和维护，也便于多道程序共享
  - 各段的长度不同，给主存的分配带来麻烦
  - 段式管理容易产生碎块，浪费主存空间

4.4虚拟存储器

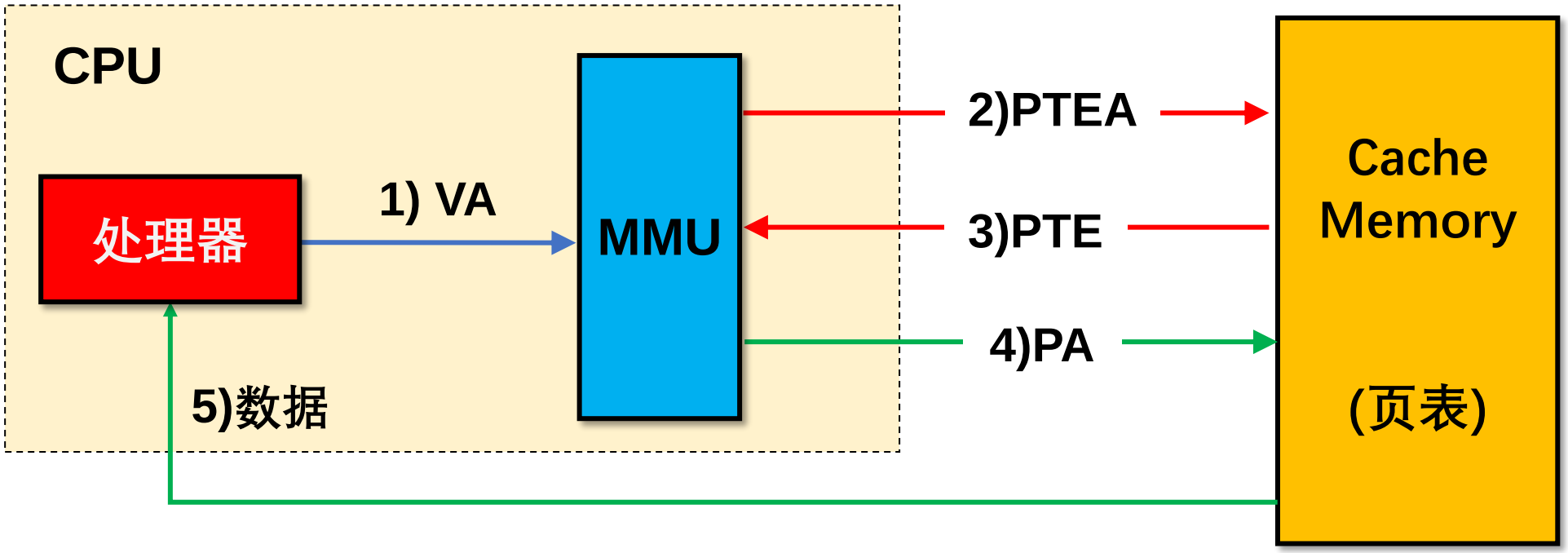
- 页式管理
  - 信息交换的单位是定长的页
  - 主存与外存均划分成等长的页面
  - 便于维护，便于生成页表，类似于cache的块表
  - 不容易产生碎块，存储空间浪费小
  - 页不是独立的实体，处理、保护和共享不方便

4.4虚拟存储器

- 段页式管理
  - 程序按模块分段
  - 段再分成长度固定的页
  - 程序调入调出按页面来进行
  - 程序共享保护按段进行
  - 兼备段式，页式管理的优点
  - 在地址映像中需要多次查表

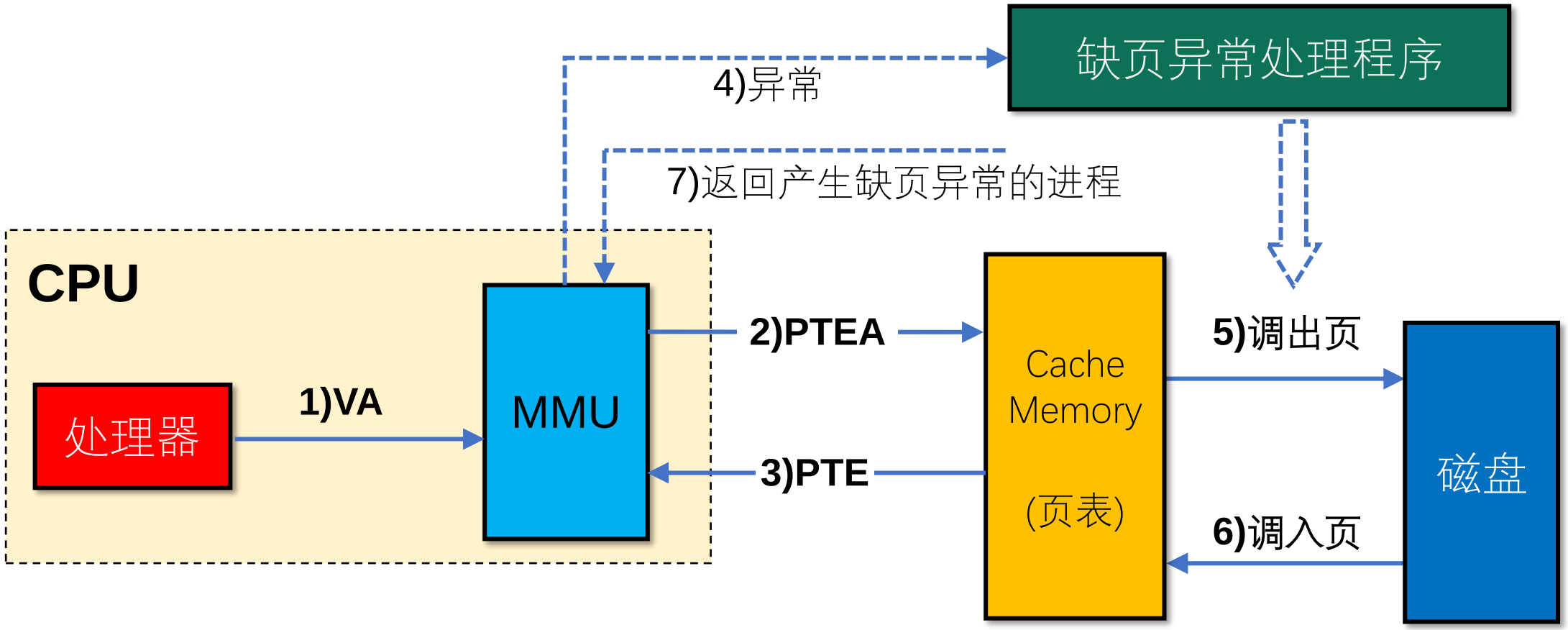
4.4虚拟存储器

- 虚拟地址→物理地址(页命中)



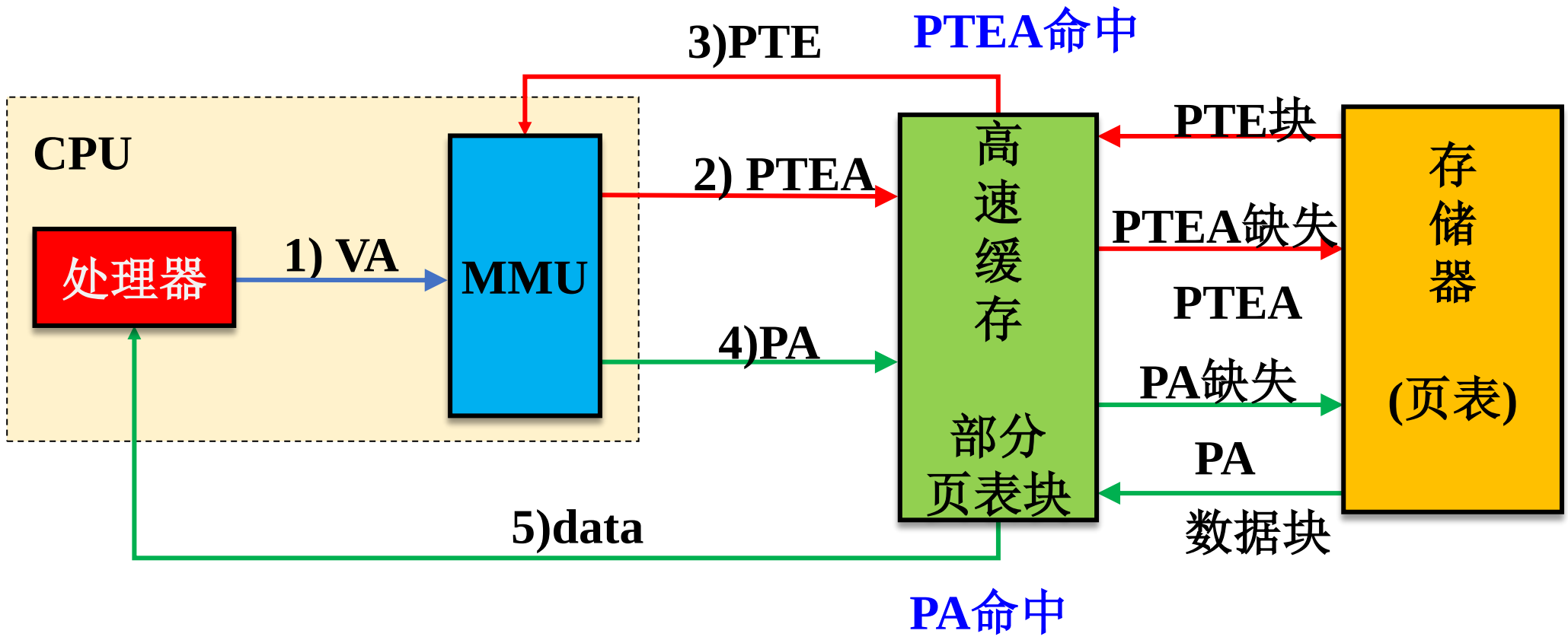
4.4虚拟存储器

- 虚拟地址→物理地址（缺页）



4. 4虚拟存储器

- 虚存，cache，主存层次



4. 4虚拟存储器

TLB:Translation Lookaside Buffer.

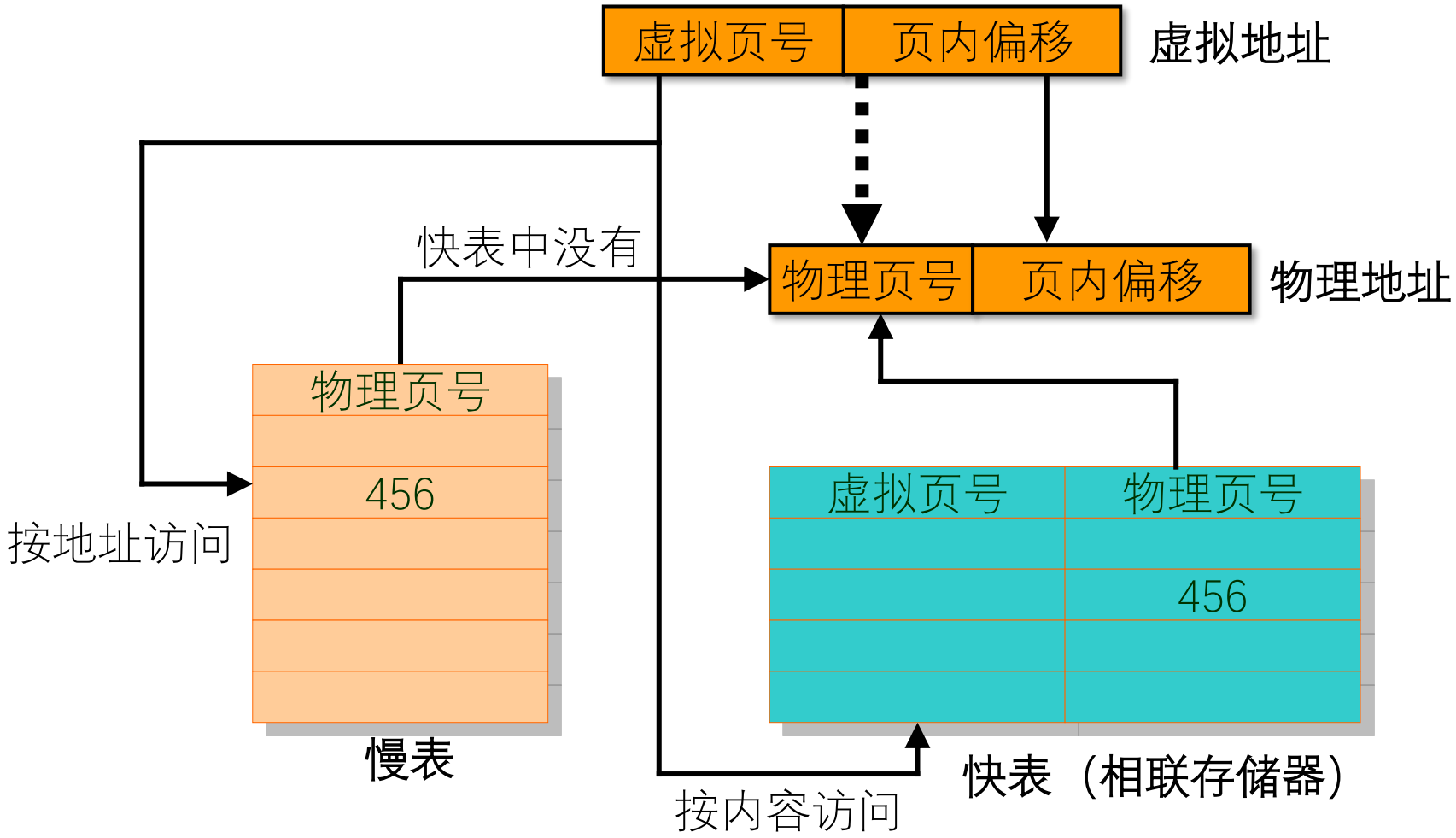
- 译为快表，直译为旁路转换缓冲，可把它理解成页表缓冲。
- 里面存放的是一些页表文件（虚拟地址到物理地址的转换表）。
- 当处理器要在主内存寻址时，不是直接在内存的物理地址里查找的，而是通过一组虚拟地址转换到主内存的物理地址，TLB就是负责将虚拟内存地址翻译成实际的物理内存地址，而CPU寻址时会优先在TLB中进行寻址。处理器的性能就和寻址的命中率有很大的关系。

4.4虚拟存储器

- 快表提高页式管理速度
  - 页表存放在主存，虚存空间即使命中也须先查页表，再访问主存，两次访存才能取出数据，速度慢
  - 为提高页表查找速度，采用cache的方法，引入一个体积小的快表，存放页表中经常被访问的表项
  - 快表与页表交换的基本单位（页表项）
  - 快表引入相联存储器机制，提高查找速度

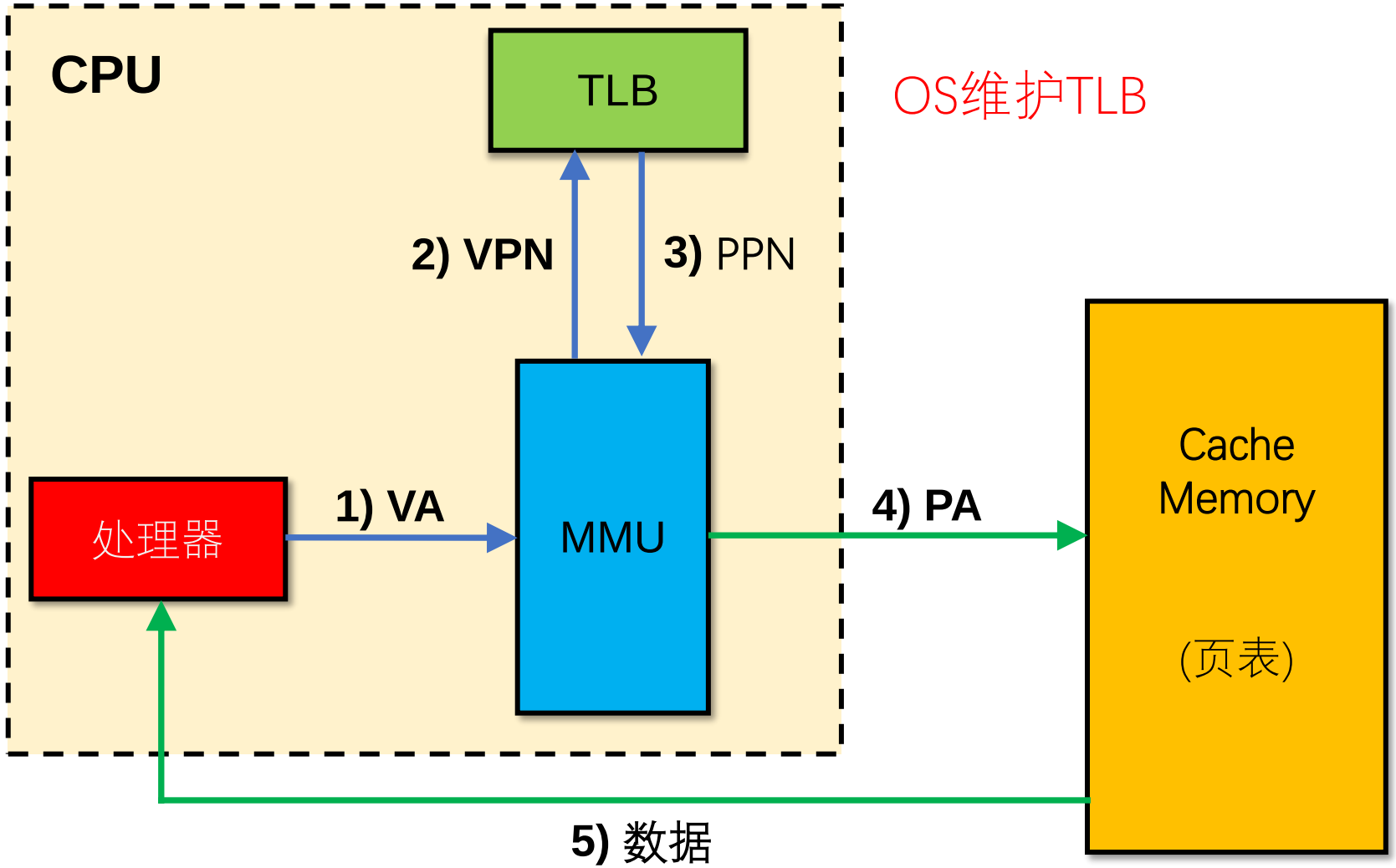
4.4虚拟存储器

- 经快慢表实现内部地址转换



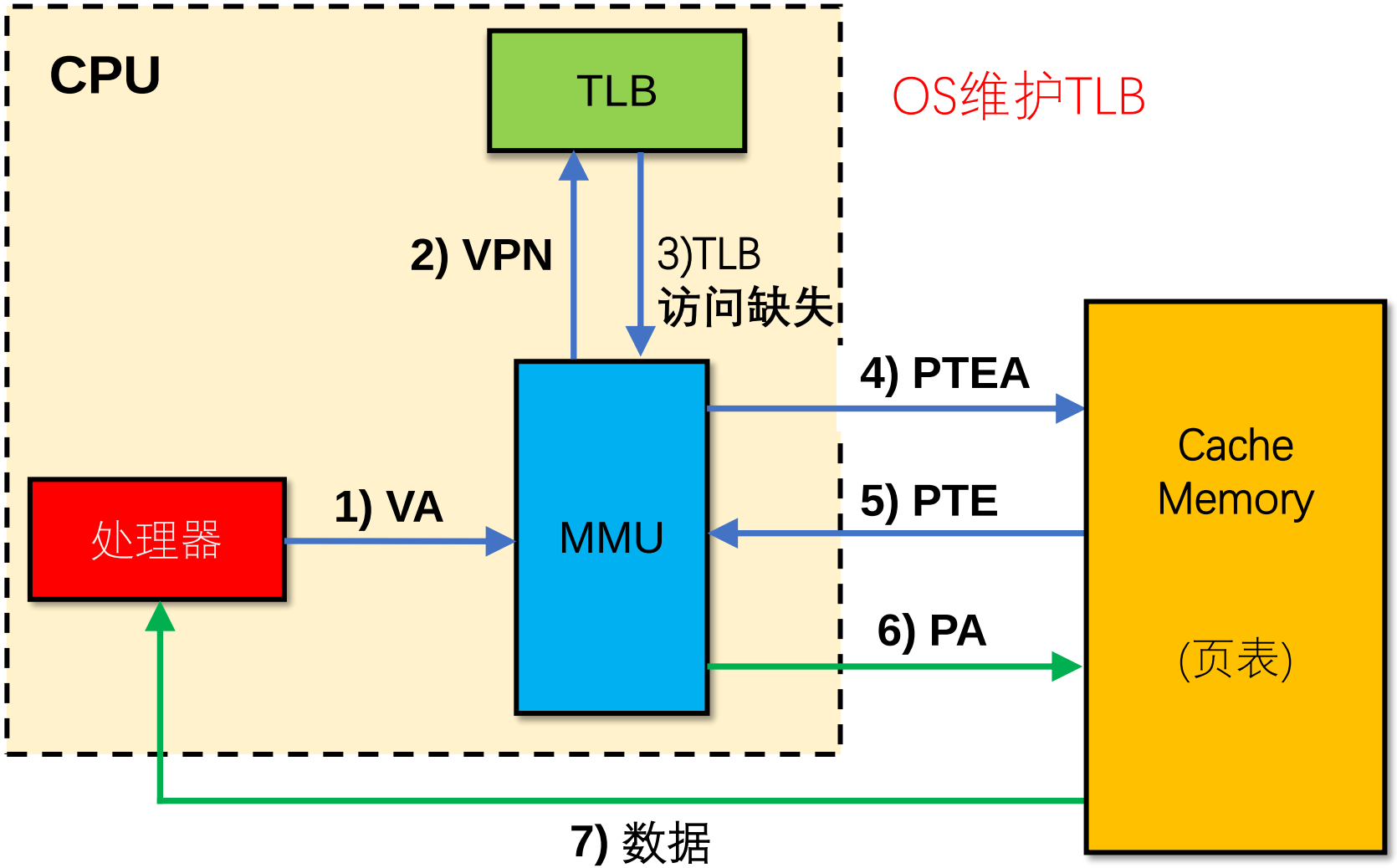
4.4虚拟存储器

- TLB 命中

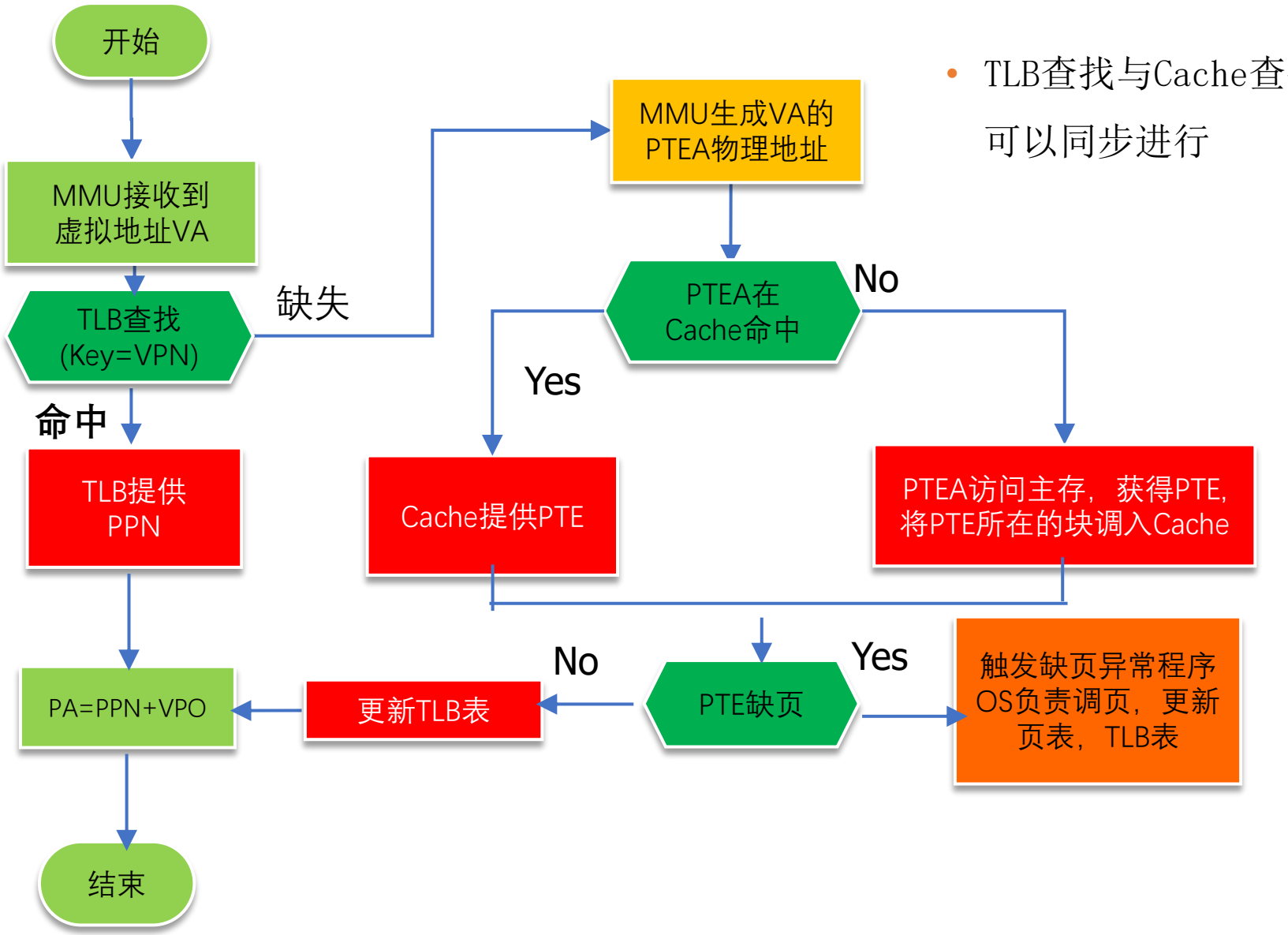


4.4虚拟存储器

- TLB 缺失



虚拟地址→物理地址流程



- TLB查找与Cache查找可以同步进行

3.4虚拟存储器

- 虚拟存储器载入过程
  - 程序按需载入，按页面载入
  - 载入页面同时更新内存中页表
  - 内存满时发生缺页，完成数据交换
    - 将不经常使用页面移到辅存
    - 从辅存程序或数据调入新的页面
    - 页面替换采用操作系统实现，cache数据替换由硬件实现
  - 内存满后系统性能？
    - 虚拟存储器替换算法涉及磁盘操作，性能损失大于cache

4.4虚拟存储器

- 存储保护的基本概念
  - 为避免多道程序相互干扰，防止某程序出错而破坏其他程序的正确性或不合法地访问其他程序或数据区，应对每个程序进行存储保护
  - 操作系统程序和用户程序都需要保护
  - 以下情况发生存储保护错
    - 地址越界（转换得到的物理地址不属于可访问范围）
    - 访问越权（访问操作与所拥有的访问权限不符）
    - **页表中设定访问（存取）权限**
  - 访问属性的设定
    - 数据段可指定R/W或RO；程序段可指定R/E或RO

第四章 存储器

4.1 存储器概述

4.2 主存储器

4.3 高速缓冲存储器

4.4 虚拟存储器

4.5 辅助存储器

4.5辅助存储器的主要类型

磁盘

磁带

软盘

光盘

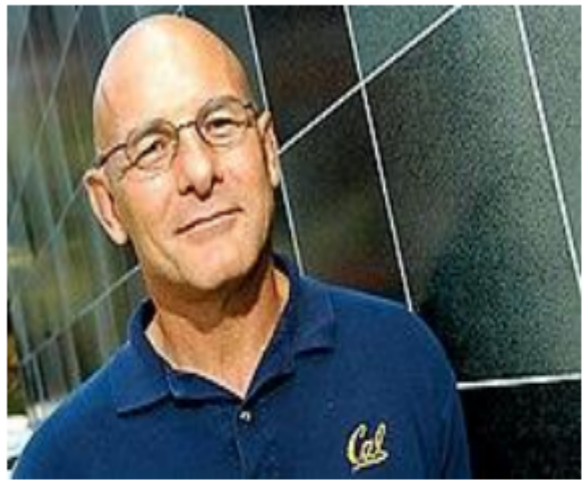
4.5辅助存储器——RAID

RAID (Redundant Arrays of Independent Disks)

1. RAID提出的背景

- 很多人有因磁盘故障而导致数据丢失的经历
- 磁盘访问速度过慢
- 多磁盘管理不方便

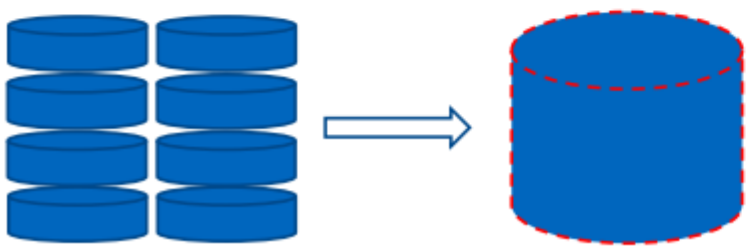
■ 由加州大学伯克利分校的D. A. Patterson于1988年提出。



- 被誉为计算机架构宗师
- RISC、RAID、NOW(工作站网络)

RAID: Redundant Arrays of Inexpensive Disks

↳ 廉价磁盘冗余阵列



Redundant Arrays of Independent Disks

↳ 独立磁盘冗余阵列

4.5辅助存储器——RAID

2. RAID的核心技术

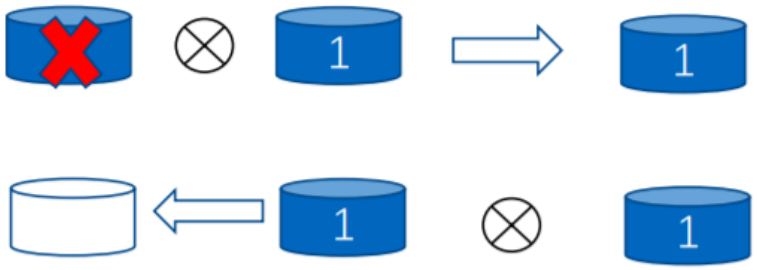
- 将数据条带化后存放在不同磁盘上，通过多磁盘的并行操作提高磁盘系统的读写速率

条带（strip）是把连续的数据分割成相同大小的数据块，把每段数据分别写入到阵列中的不同磁盘上的方法。这就能使多个进程同时访问数据的多个不同部分而不会造成磁盘冲突，而且在需要对这种数据进行顺序访问的时候可以获得最大程度上的 I/O 并行能力，从而获得非常好的性能。

- 使用基于异或运算为基础的校验恢复损坏的数据

$1 \otimes 0 = 1$
$1 \otimes 1 = 0$
$0 \otimes 0 = 0$

$1 = 0 \otimes 1$
$1 = 1 \otimes 0$
$0 = 0 \otimes 0$

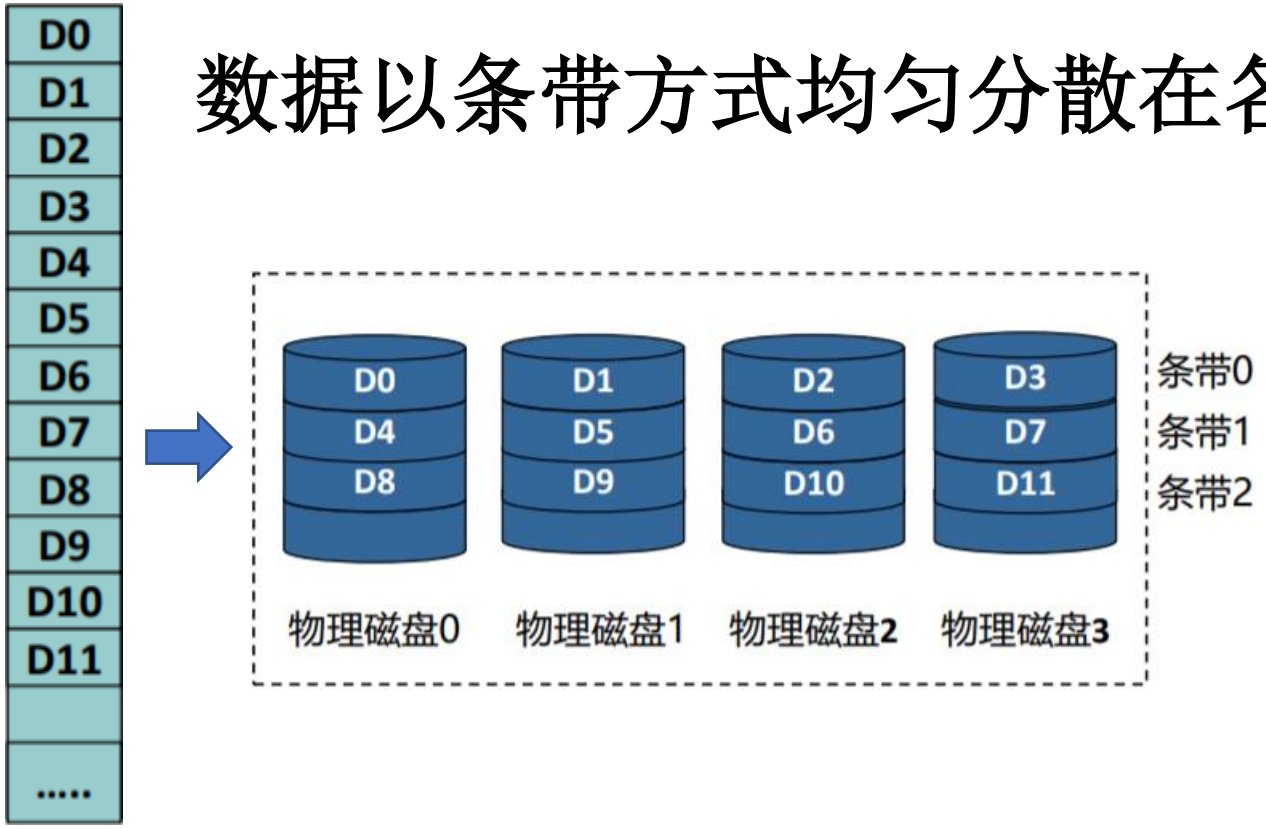


4.5辅助存储器——RAID

3.常见的几种RAID技术

- **RAID 0**

数据以条带方式均匀分散在各磁盘

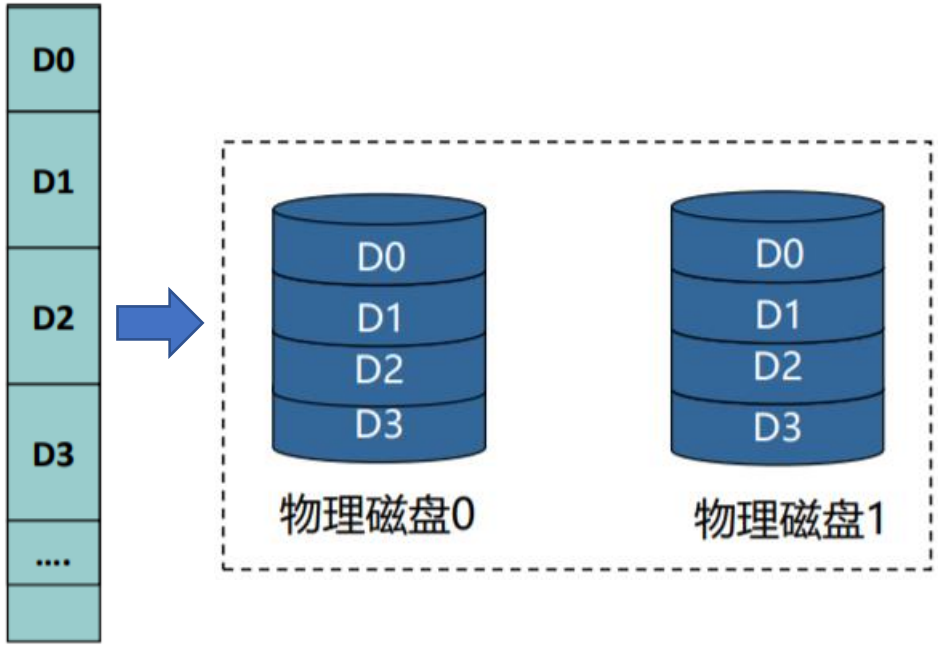


所需磁盘数	2个或更多
优点	磁盘读写效率高 无校验带来使用和配置方便
缺点	无冗余，数据安全性低
适用领域	视频、图像及需高传输带宽的应用

4.5辅助存储器——RAID

RAID 1

数据采用镜像的冗余方式，同一数据有多份拷贝

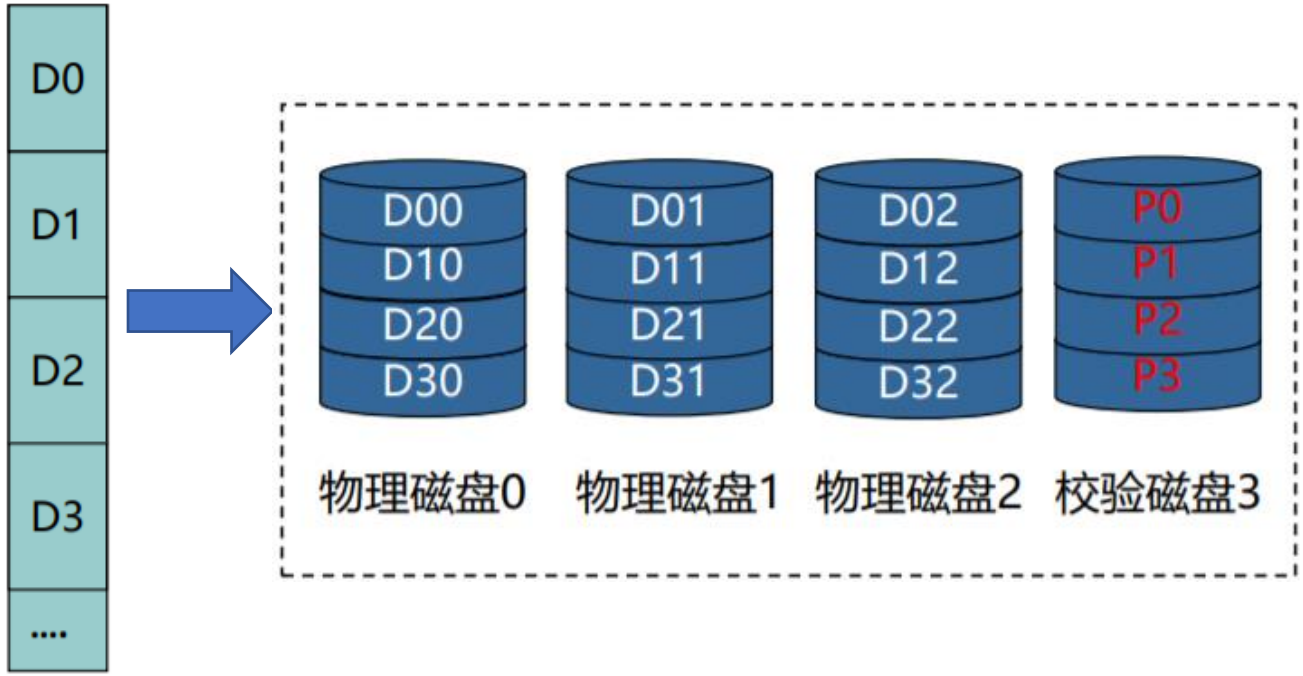


所需磁盘数	至少2个
优点	100%数据冗余，数据安全性高 理论上可以实现2倍的读取效率
缺点	空间利用率只有50%
适用领域	财务、金融等高可用应用

4.5辅助存储器——RAID

RAID 3/4

数据按 位/条带 并行传输到多个磁盘上，同时校验数据存放到专用校验盘上

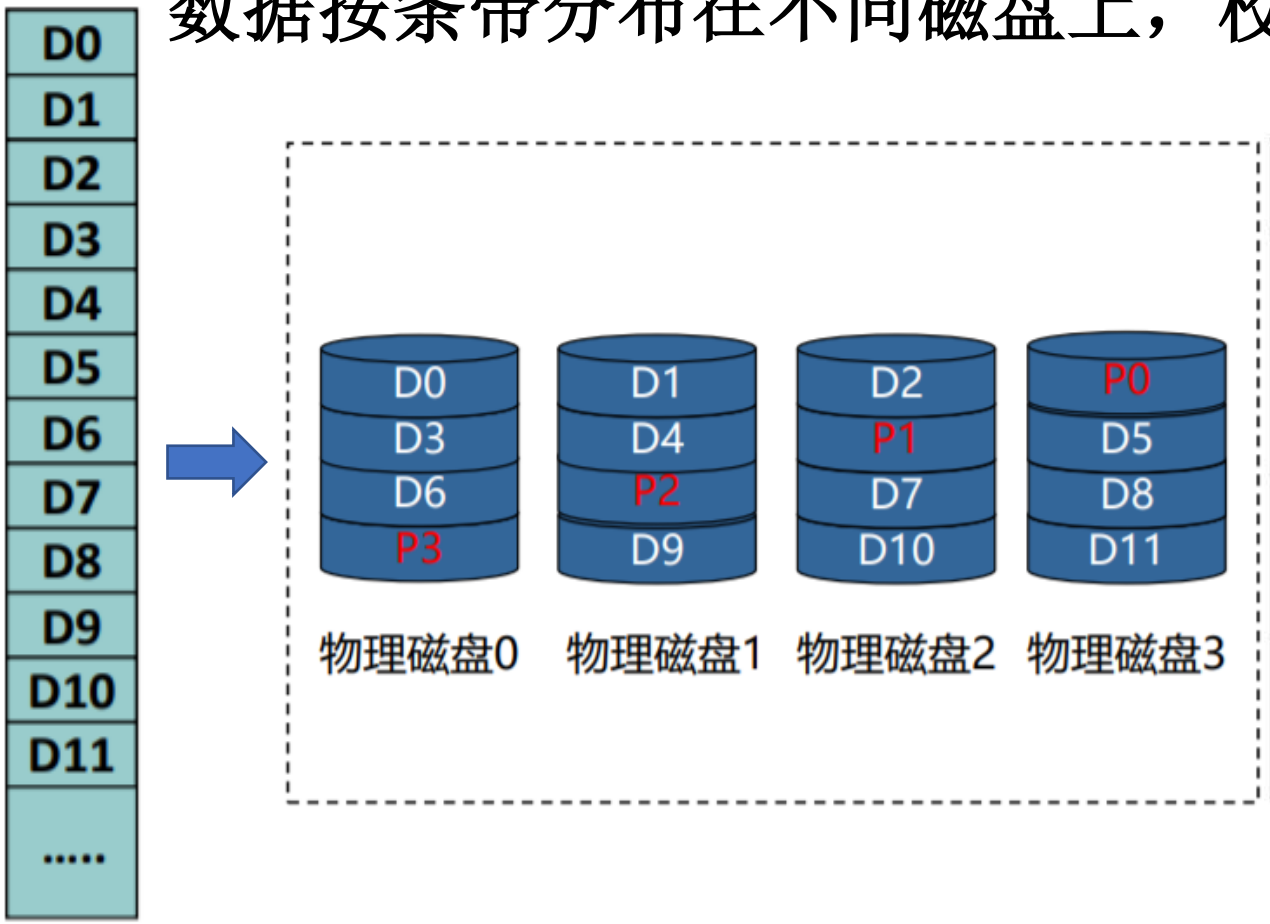


所需磁盘数	至少3块
优点	读写性能都比较好 磁盘利用率高， N-1/N
缺点	控制器设计复杂 校验磁盘的写性能有瓶颈
适用领域	视频生成和图像、视频编辑等 需要高吞吐量的应用环境

4.5辅助存储器——RAID

RAID 5

数据按条带分布在不同磁盘上，校验信息被均匀分散到各磁盘上

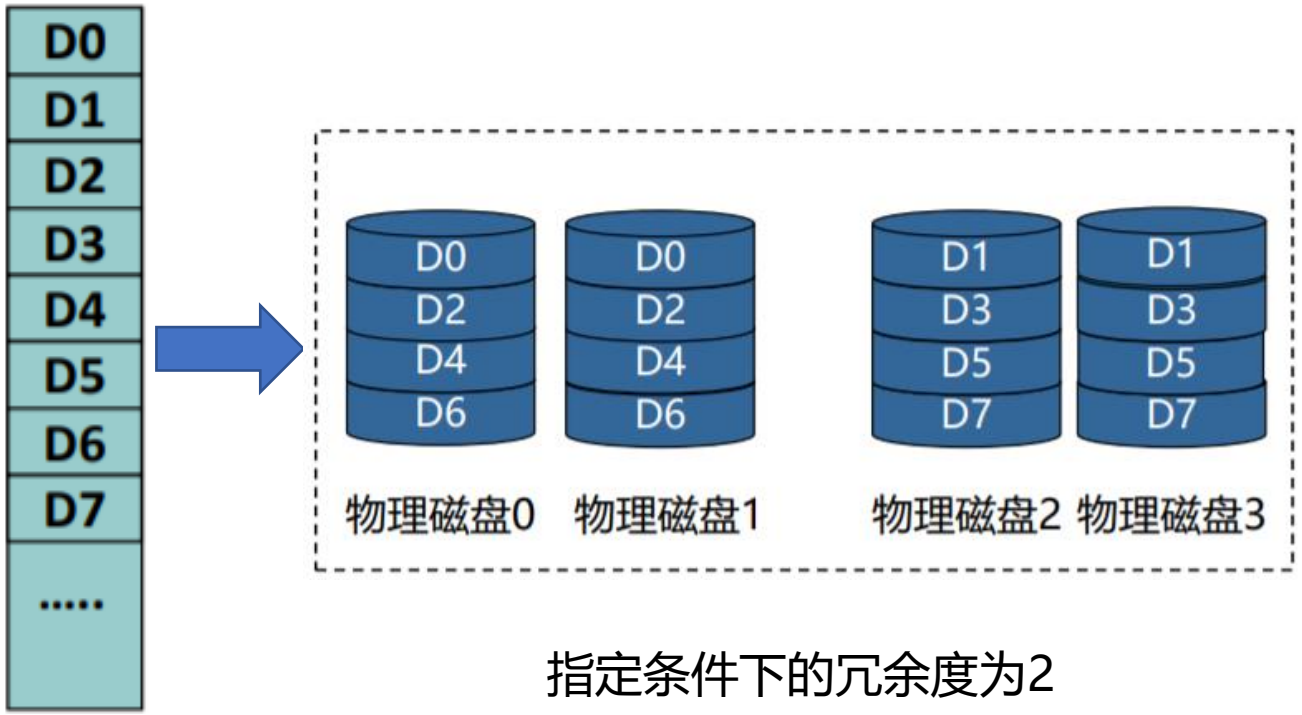


所需磁盘数	最低为3个
优点	读性能比较高 校验信息的分布式存取，避免出现写操作的瓶颈
缺点	控制器设计复杂 磁盘重建的过程比较复杂
适用领域	FTP、Email、Web、数据库

4.5辅助存储器——RAID

RAID 10

结合RAID 1和 RAID 0 ,先镜像，再条带化



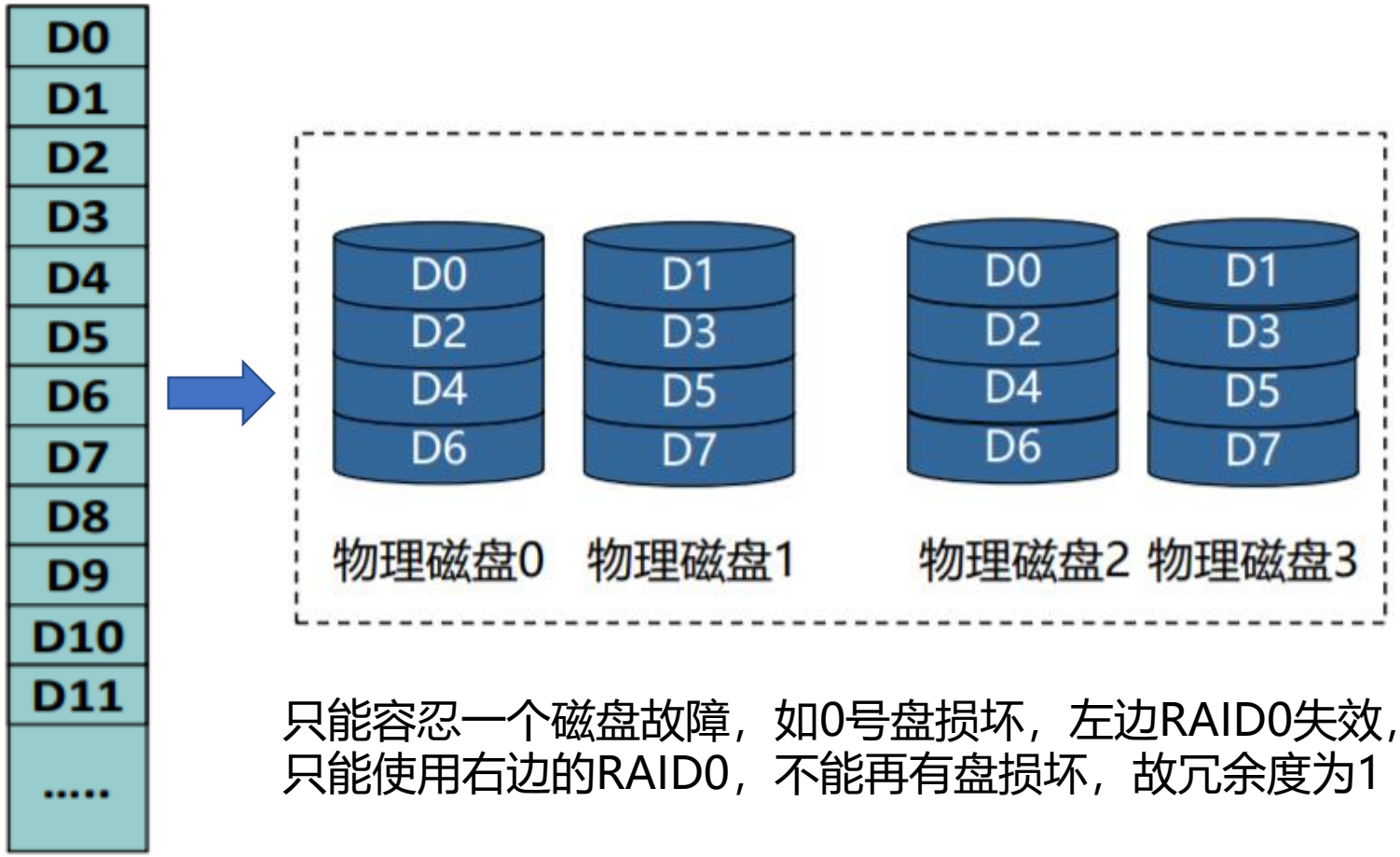
指定条件下的冗余度为2

所需磁盘数	最低为4个
优点	读性能高 数据安全性好，允许同时有半数磁盘失效
缺点	空间利用率也只有50%
适用领域	多用于高可用性和高安全性的应用场合

4.5辅助存储器——RAID

- **RAID 01**

  结合RAID 0和 RAID 1 ,先条带化，再镜像



只能容忍一个磁盘故障，如0号盘损坏，左边RAID0失效，
只能使用右边的RAID0，不能再有盘损坏，故冗余度为1

4.5辅助存储器——RAID

3. RAID的实现方式

- 软件RAID

  依赖于主机CPU完成，没有第三方的控制处理器和I/O芯片

- 硬件RAID

  专用RAID控制处理器和I/O芯片处理RAID任务，不占CPU

4.5辅助存储器——RAID

- 3. RAID技术总结

RAID级别	RAID0	RAID1	RAID3	RAID5	RAID10
容错性	无	有	有	有	有
冗余类型	无	镜像	奇偶校验	奇偶校验	镜像
备盘	无	有	有	有	有
读性能	高	低	高	高	中间
随机写性能	高	低	最低	低	中间
连续写性能	高	低	低	低	中间
需要的磁盘数	2个或更多	2个或2N个	3个或更多	3个或更多	4个或2N（N≥2）
可用容量	总的磁盘容量	磁盘容量的50%	磁盘容量的（N-1）/N	磁盘容量的（N-1）/N	磁盘容量的50%

第四章 存储器

4.1 存储器概述

4.2 主存储器

4.3 高速缓冲存储器

4.4 虚拟存储器

4.5 辅助存储器