数 字 逻 辑

丁 贤 庆

ahhfdxq@163.com

通知　实验安排

数字逻辑电路课程有16个学时的实验，初步安排：具体安排参见公共邮箱中的word文档。

本周开始进行实验环节，实验结束后16周周日（6月16号）晚23点前，各班学委要提交实验报告的电子版给我的邮箱ahhfdxq@163.com。不用收纸质报告了。

实验地点：综合实验楼306房间

关于实验报告

- 8次实验中，自己选择4次写到实验报告中就可以了。
- 补充的实验不需要写到实验报告中。

第七章　作业布置

1、本周有实验。

2、下次交作业第11周。

3、本周作业：从第7章课后习题中选2题写到作业本上。

第 七 章

半导体存储器

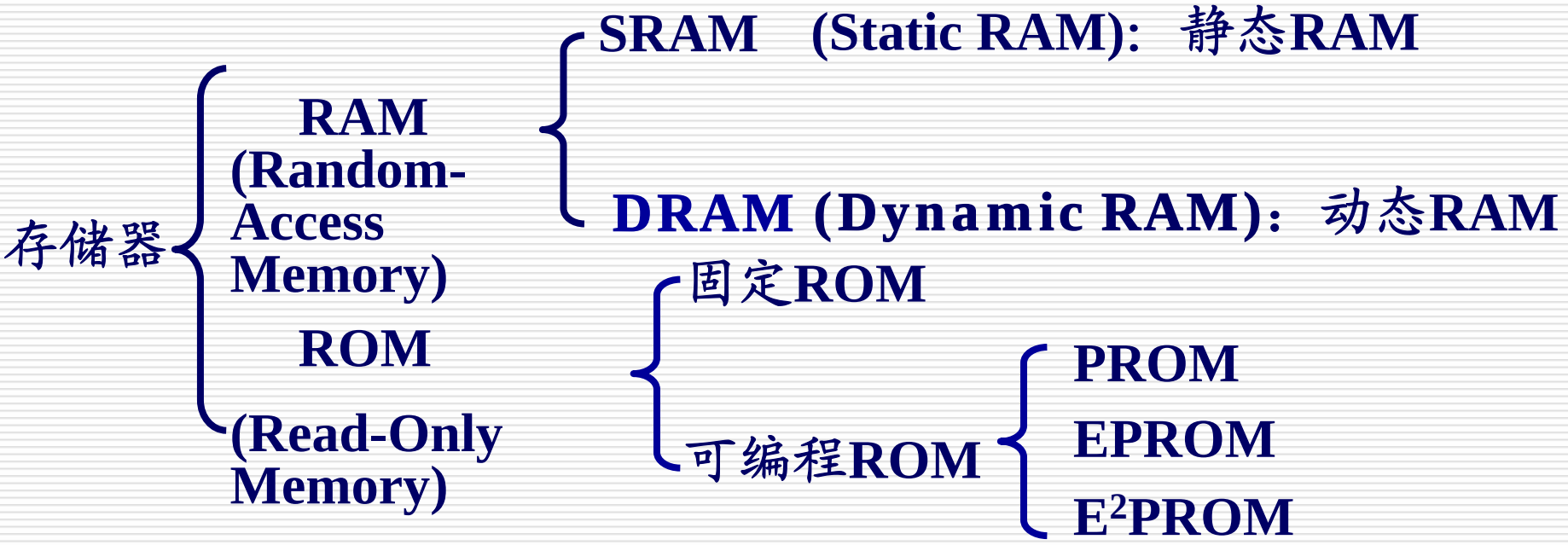
7. 半导体存储器

7.1 只读存储器

7.2 随机存取存储器

7.1 只读存储器

7.1 只读存储器



RAM(随机存取存储器): 在运行状态可以随时进行读或写操作。

存储的数据必须有电源供应才能保存, 一旦掉电, 数据全部丢失。

ROM(只读存储器): 在正常工作状态只能读出信息。

断电后信息不会丢失，常用于存放固定信息（如程序、常数等）。

几个基本概念：

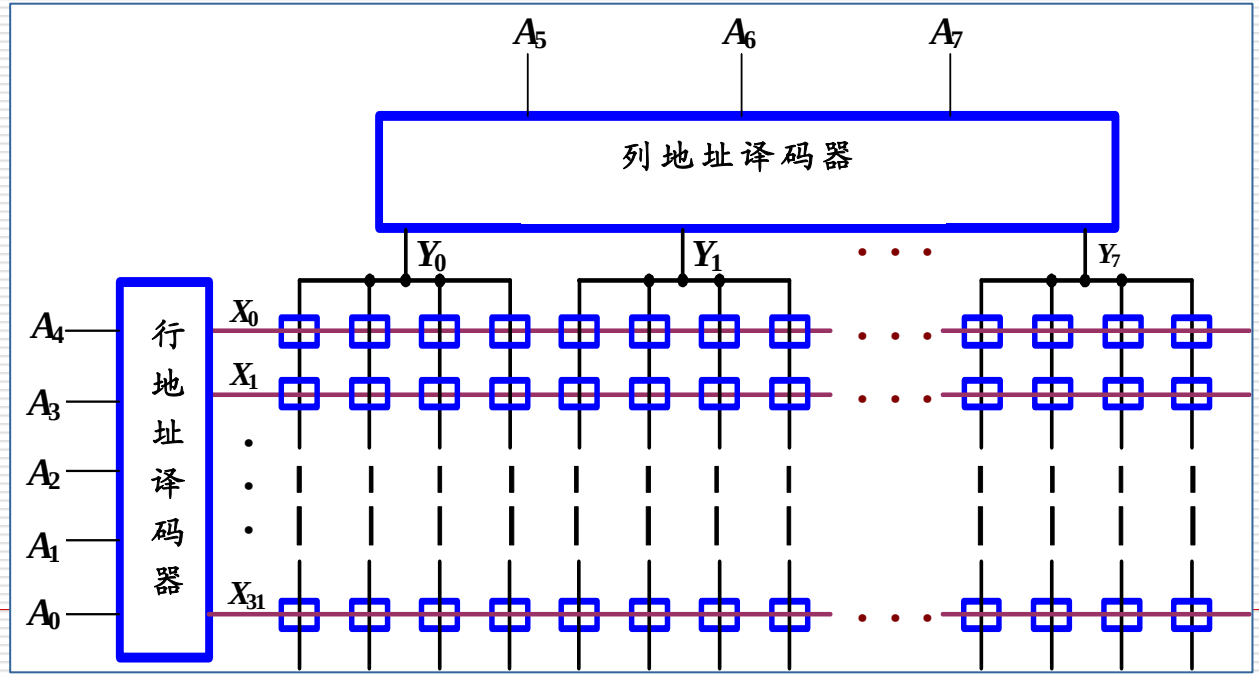
字：计算机中作为一个整体被存取传送处理的一组数据 。

字长：一个字所含的位数称为字长。4位

字数：字的总量。 $2^5 \times 2^3 = 256$　字数=2^n（n为地址线的总数）

地址：每个字的编号。

存储容量（M)：存储单元的数目。 存储容量（M) = 字数 × 位数



7.1.1 ROM的基本结构

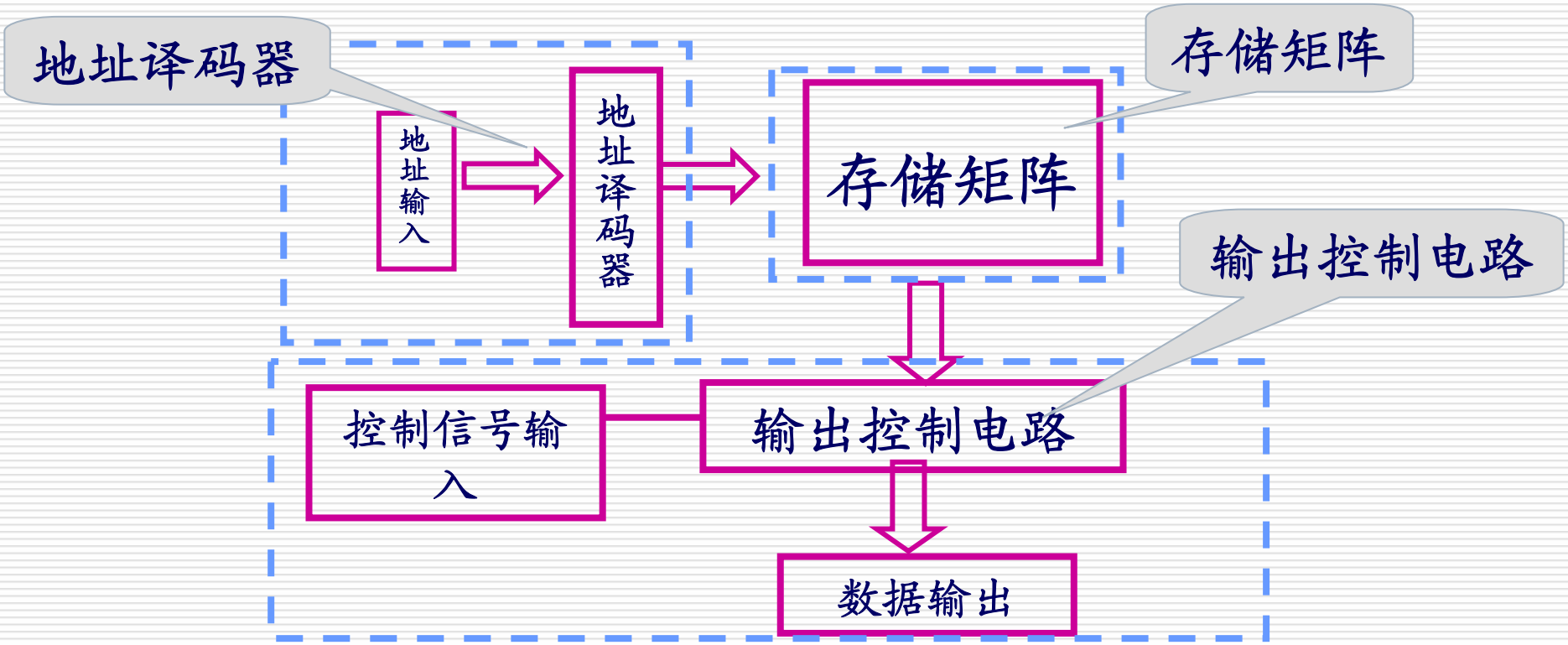
只读存储器，工作时内容只能读出，不能随时写入，所以称为只读存储器。(Read-Only Memory)

ROM的分类

- 按存贮单元中器件划分
  - 二极管ROM
  - 三极管ROM
  - MOS管ROM
- 按写入情况划分
  - 固定ROM
  - 可编程ROM
    - PROM
    - EPROM
    - E^2PROM

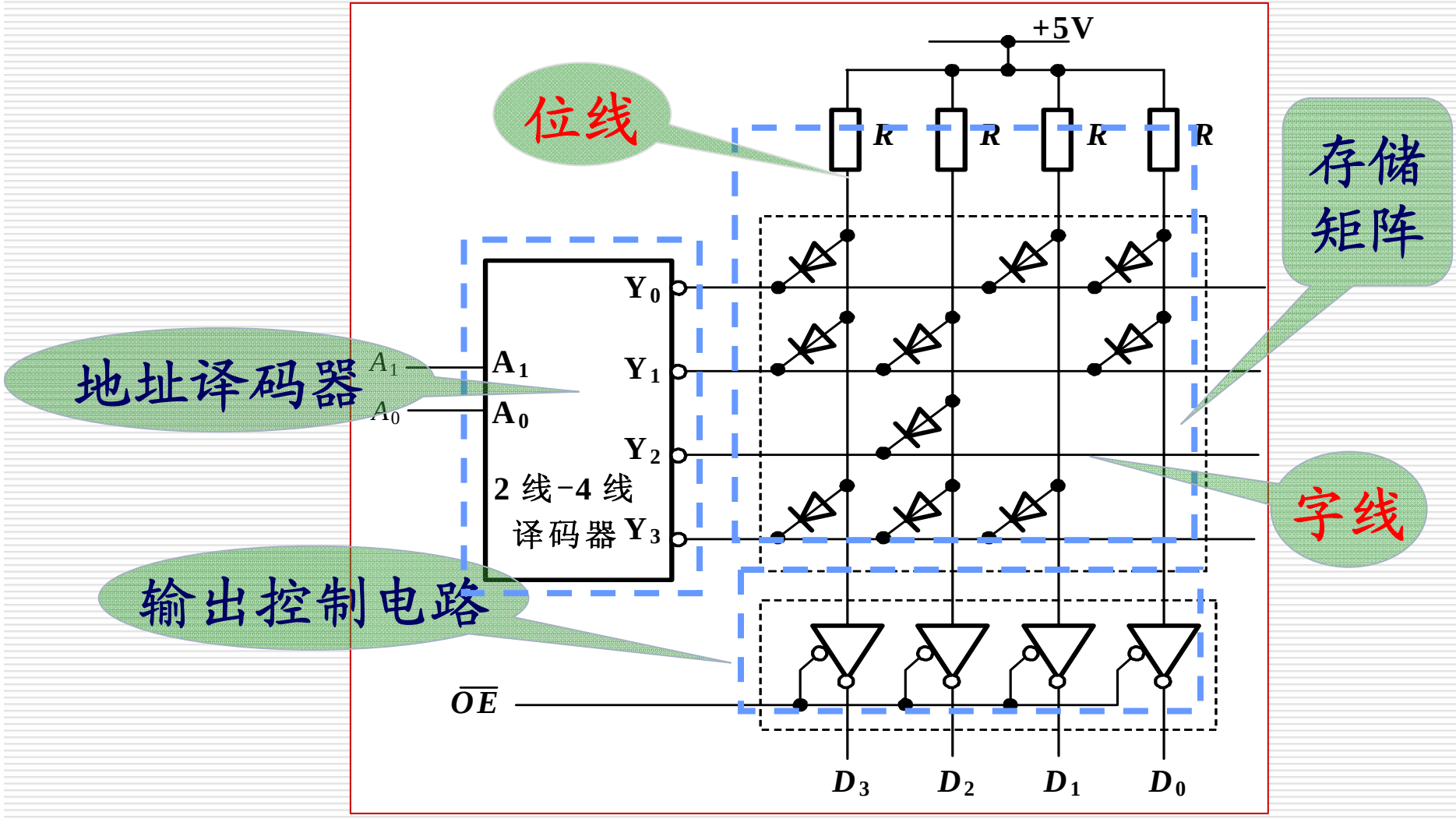
7.1.1 ROM的基本结构

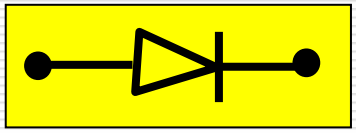
ROM是一种永久性数据存储器，其中的数据一般由专用的装置写入，数据一旦写入，不能随意改写，在切断电源之后，数据也不会消失。



ROM主要由地址译码器、存储矩阵和输出控制电路三部分组成。

ROM（二极管PROM）结构示意图　　M=4×4

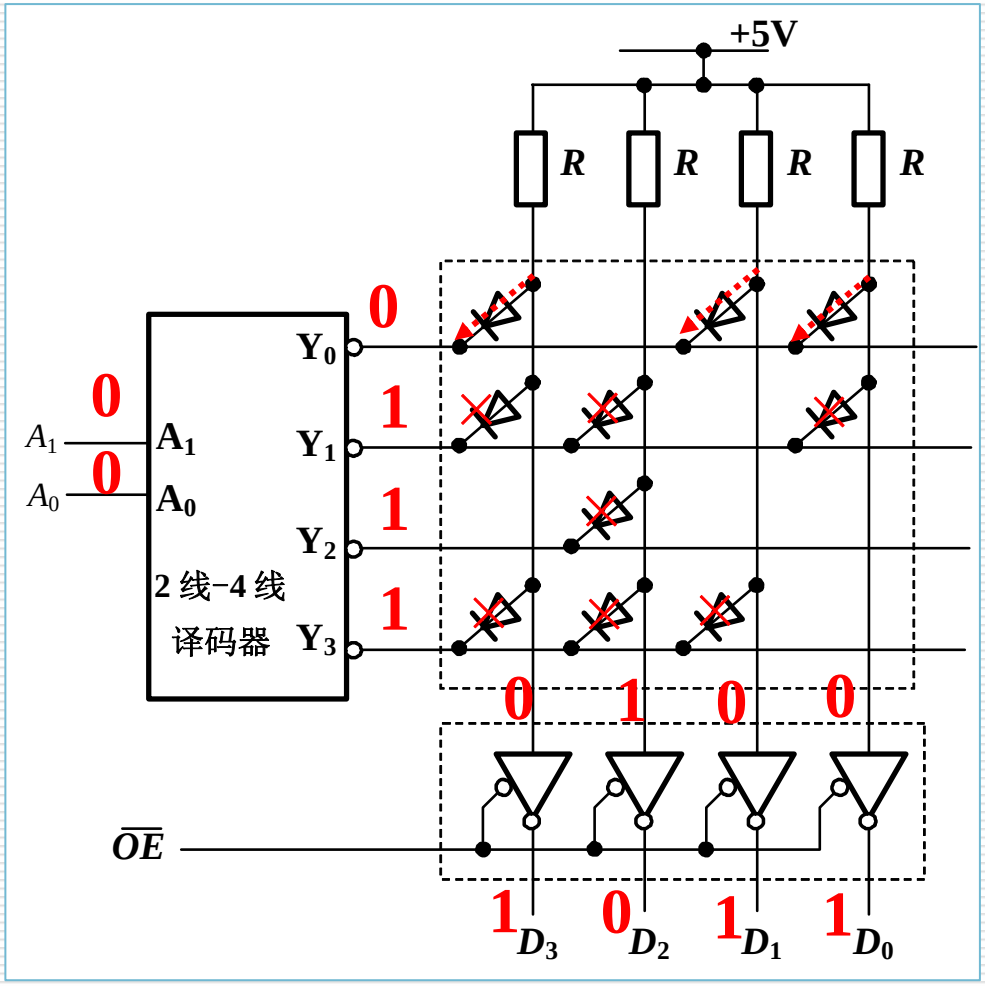




对于二极管，当正极比负极电压高0.7V时，二极管导通。当二极管导通后，正负极之间有一个恒定的电压差0.7V。

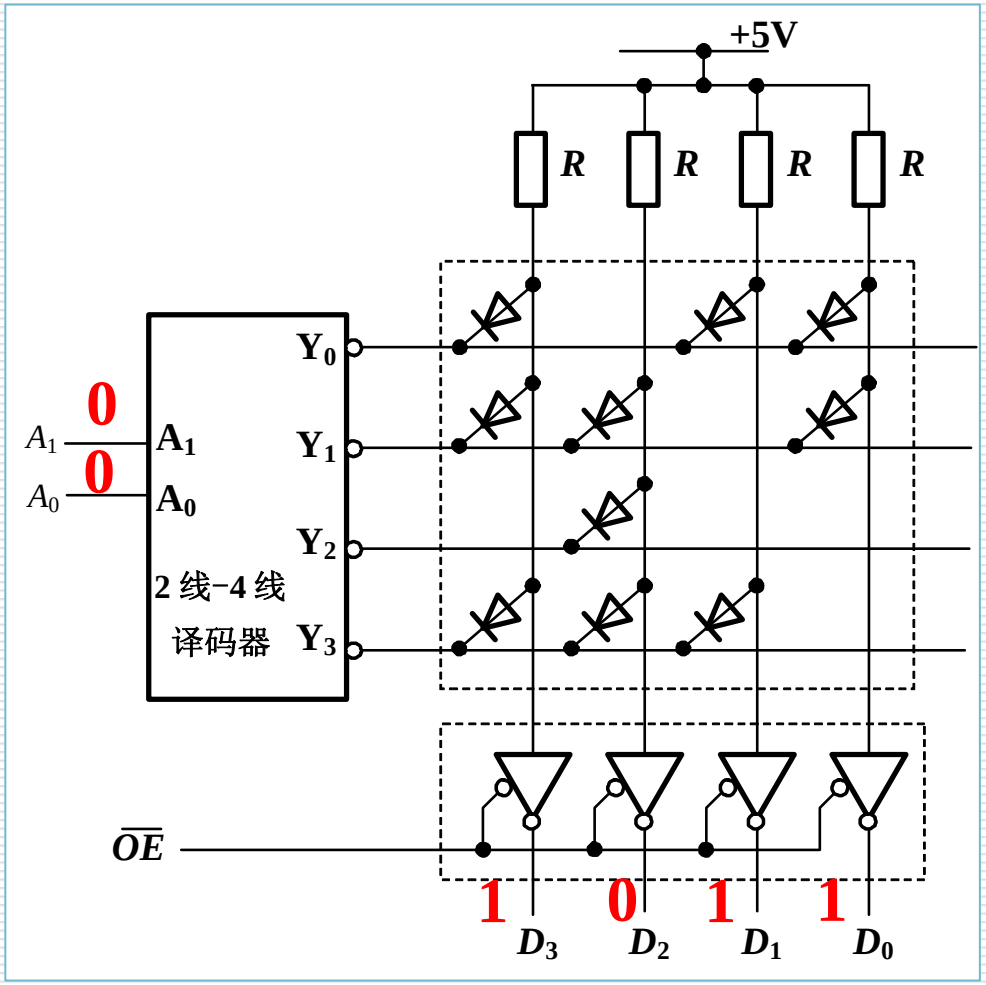
当$\overline{OE}$=0时

地址		内容			
A_1	A_0	D_3	D_2	D_1	D_0
0	0				



当$\overline{OE}$=0时

地址		内容			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	1	0	1	1
0	1	1	1	0	1
1	0	0	1	0	0
1	1	1	1	1	0



- 字线与位线的交点都是一个存储单元。交点处有二极管相当存1，无二极管相当存0

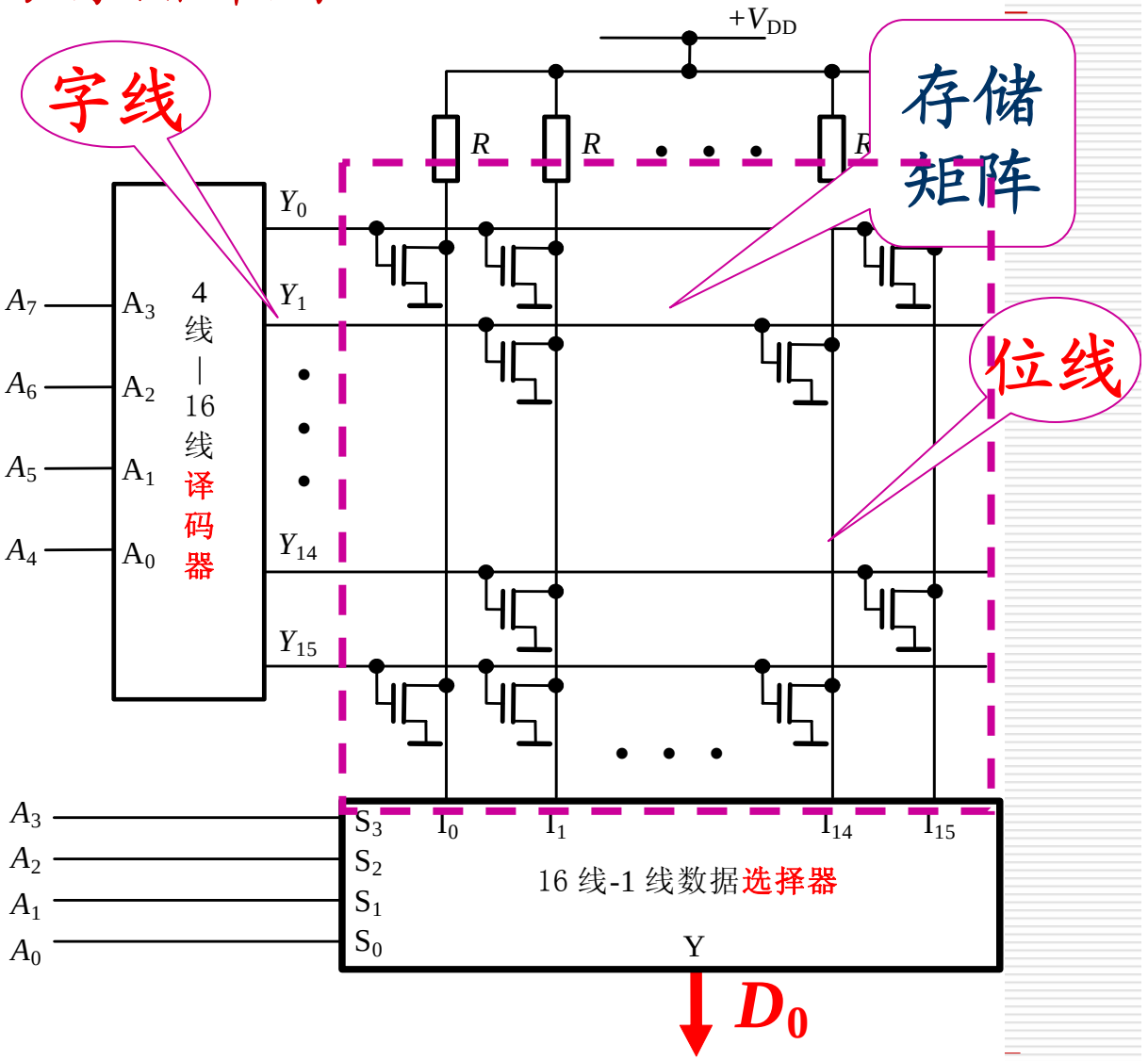
当$\overline{OE}$=1时输出为高阻状态

7.1.2 二维译码与存储阵列

•字线与位线的交点都是一个存储单元。

•交点处有MOS管相当存0，无MOS管相当存1。

该存储器的容量=?



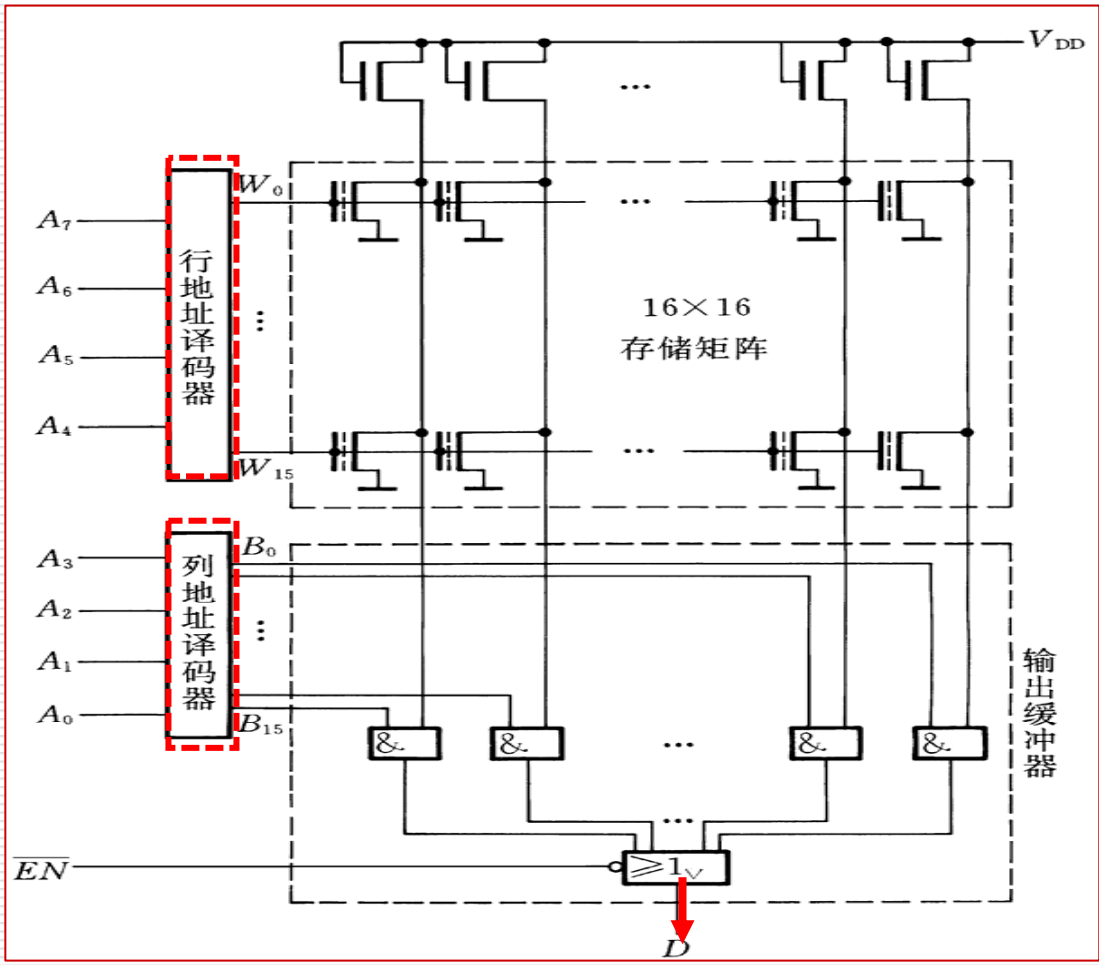
7.1.3 可编程ROM（256X1位EPROM）

256个存储单元排成16×16的矩阵

行译码器从16行中选出要读的一行

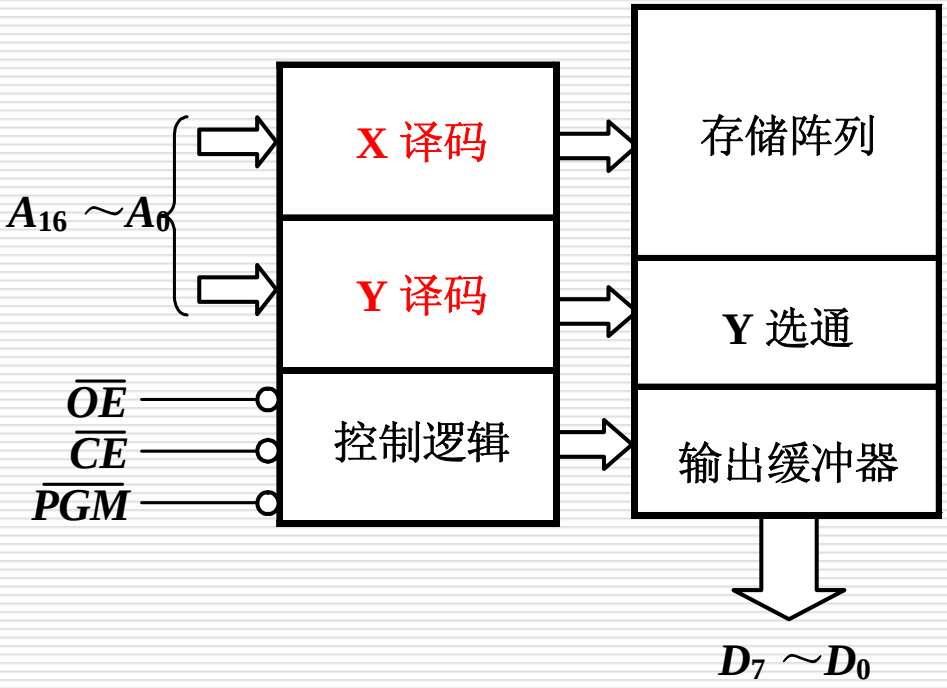
列译码器再从选中的一行存储单元中选出要读的一列的一个存储单元。

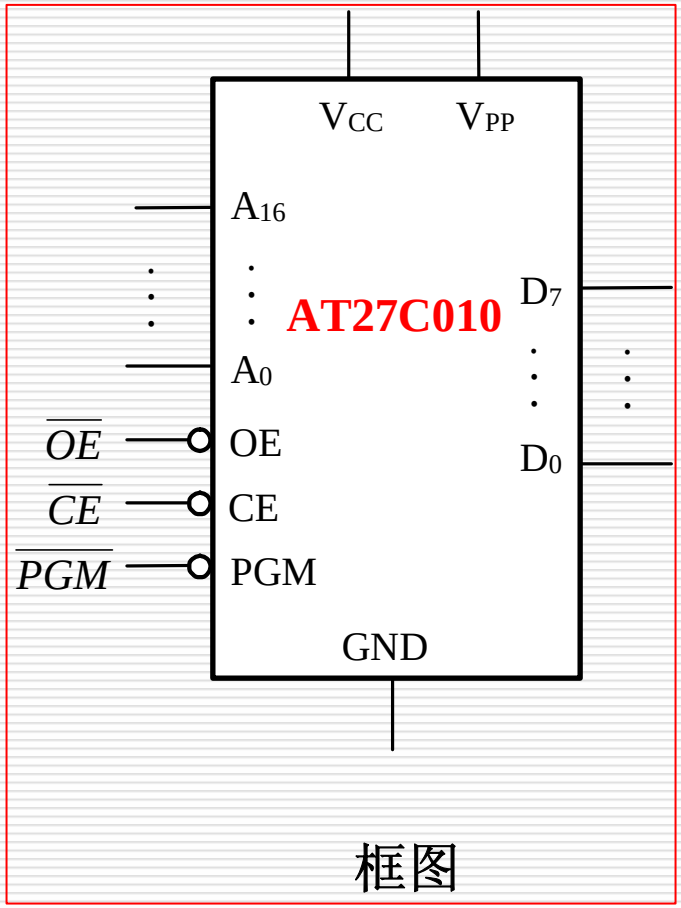
如选中的存储单元的MOS管的浮栅注入了电荷，该管截止，读得1；相反读得0



7.1.4 ROM读操作实例

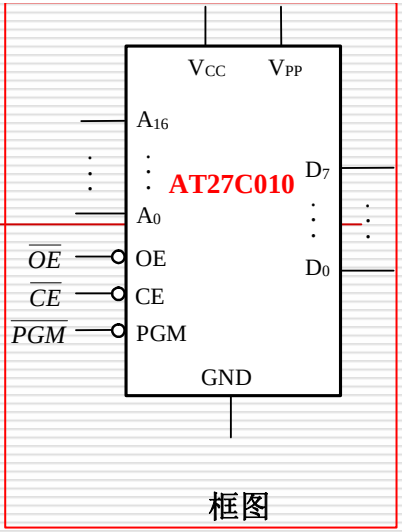
1. PROM芯片AT27C010简介





128K×8位ROM

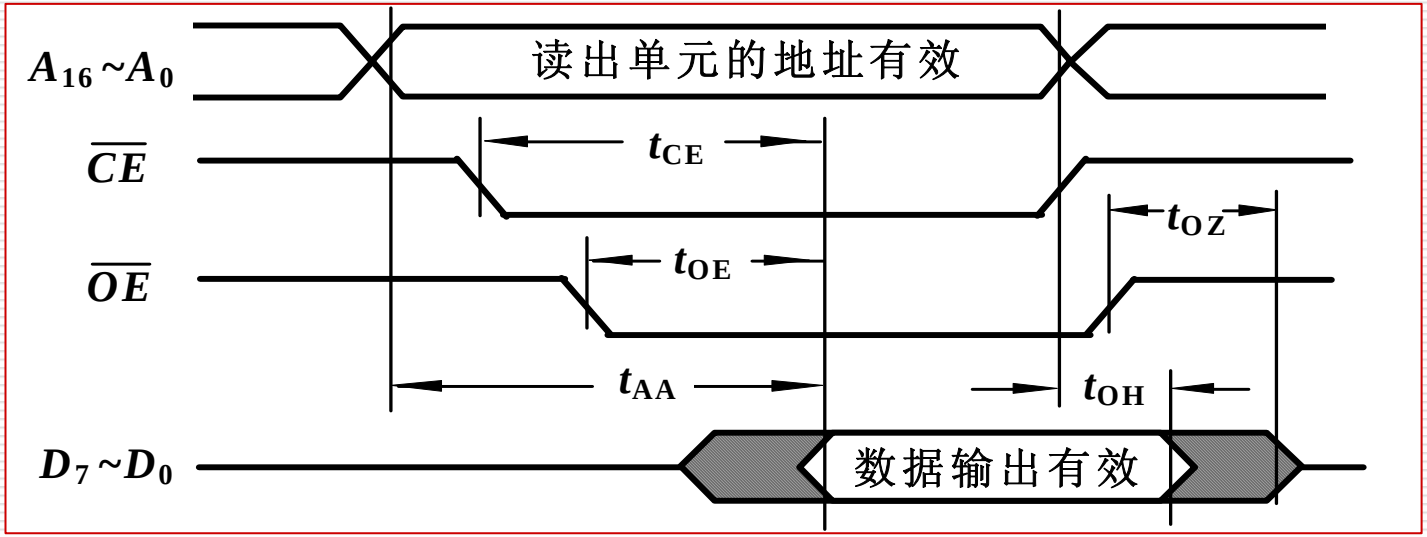
工作模式



框图

工作模式	$\overline{CE}$	$\overline{OE}$	$\overline{PGM}$	A_{16}~A_0	V_{PP}	D_7~D_0
读	0	0	X	A_i	X	数据输出
输出无效	X	1	X	X	X	高阻
等待	1	X	X	A_i	X	高阻
快速编程	0	1	0	A_i	V_{PP}	数据输入
编程校验	0	0	1	A_i	V_{PP}	数据输出

2. 读操作与定时图

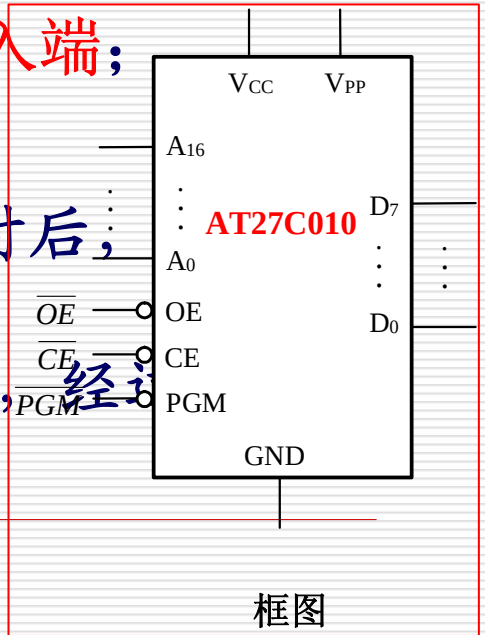


（1）欲读取单元的地址加到存储器的地址输入端；

（2）加入有效的片选信号 $\overline{CE}$

（3）使输出使能信号 $\overline{OE}$ 有效，经过一定延时后，数据出现在数据线上；

（4）让片选信号 $\overline{CE}$ 或输出使能信号 $\overline{OE}$ 无效，经过一定时后数据线呈高阻态，本次读出结束。



框图

7.1.5 ROM应用举例

ROM应用实例

（1）用于存储固定的专用程序

（2）利用ROM可实现查表或码制变换等功能

查表功能 – – 查某个角度的三角函数

把变量值（角度）作为地址码，其对应的函数值作为存放在该地址内的数据，这称为“造表”。使用时，根据输入的地址(角度)，就可在输出端得到所需的函数值，这就称为“查表”。

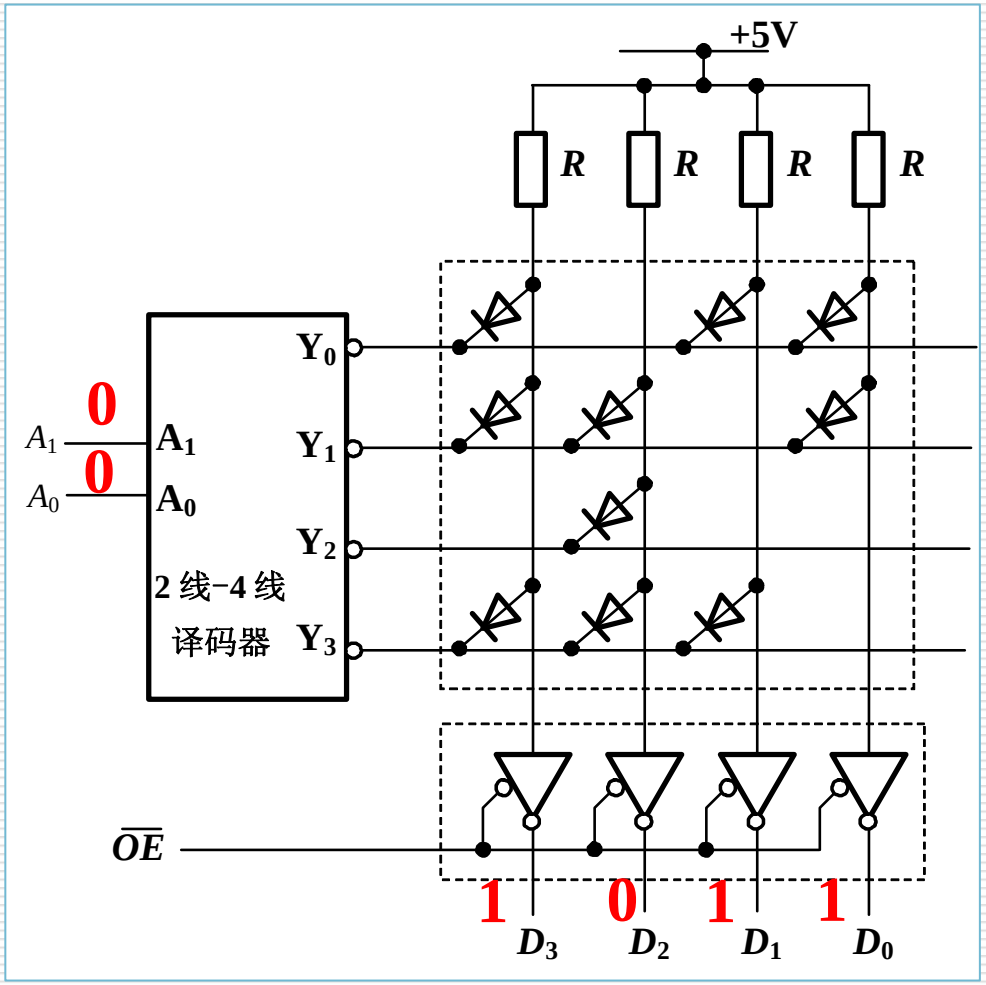
码制变换 – – 把欲变换的编码作为地址，把最终的目的编码作为相应存储单元中的内容即可。

用ROM实现二进制码与格雷码相互转换的电路

C	$I_3\,I_2\,I_1\,I_0$ 二进制码	$O_3O_2O_1O_0$ 格雷码	C	$I_3\,I_2\,I_1\,I_0$ 格雷码	$O_3O_2O_1O_0$ 二进制码
0	0 0 0 0	0 0 0 0	1	0 0 0 0	0 0 0 0
0	0 0 0 1	0 0 0 1	1	0 0 0 1	0 0 0 1
0	0 0 1 0	0 0 1 1	1	0 0 1 0	0 0 1 1
0	0 0 1 1	0 0 1 0	1	0 0 1 1	0 0 1 0
0	0 1 0 0	0 1 1 0	1	0 1 0 0	0 1 1 1
0	0 1 0 1	0 1 1 1	1	0 1 0 1	0 1 1 0
0	0 1 1 0	0 1 0 1	1	0 1 1 0	0 1 0 0
0	0 1 1 1	0 1 0 0	1	0 1 1 1	0 1 0 1
0	1 0 0 0	1 1 0 0	1	1 0 0 0	1 1 1 1
0	1 0 0 1	1 1 0 1	1	1 0 0 1	1 1 1 0
0	1 0 1 0	1 1 1 1	1	1 0 1 0	1 1 0 0
0	1 0 1 1	1 1 1 0	1	1 0 1 1	1 1 0 1
0	1 1 0 0	1 0 1 0	1	1 1 0 0	1 0 0 0
0	1 1 0 1	1 0 1 1	1	1 1 0 1	1 0 0 1
0	1 1 1 0	1 0 0 1	1	1 1 1 0	1 0 1 1
0	1 1 1 1	1 0 0 0	1	1 1 1 1	1 0 1 0

当$\overline{OE}$=0时

地址		内容			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	1	0	1	1
0	1	1	1	0	1
1	0	0	1	0	0
1	1	1	1	1	0



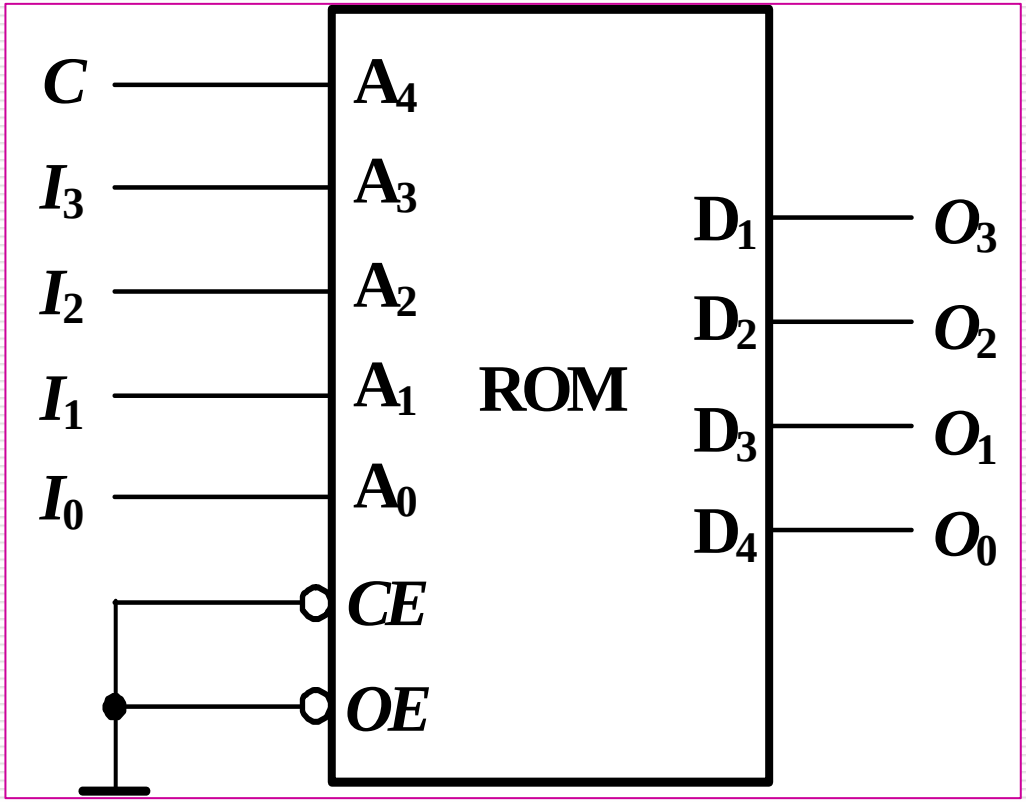
- 字线与位线的交点都是一个存储单元。交点处有二极管相当存1，无二极管相当存0

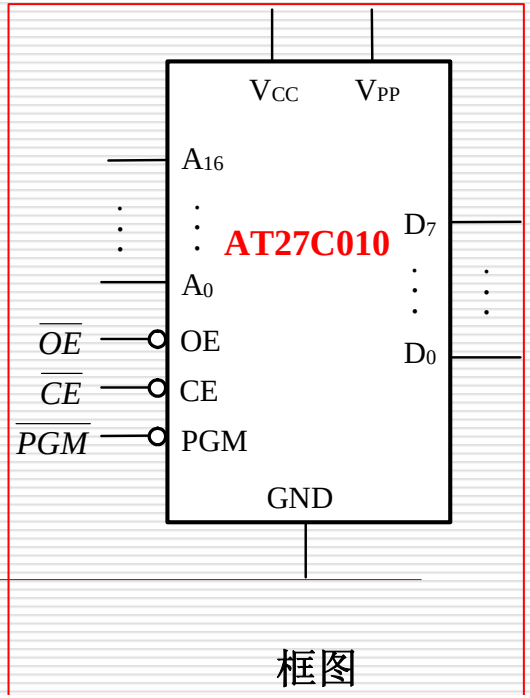
当$\overline{OE}$=1时输出为高阻状态

$C=A_4$　　$I_3\ I_2\ I_1\ I_0=A_3A_2A_1A_0$　　$O_3O_2O_1O_0=D_3D_2D_1D_0$

C (A_4)	$I_3\ I_2\ I_1\ I_0$ ($A_3A_2A_1A_0$) 二进制码	$O_3O_2O_1O_0$ ($D_3D_2D_1D_0$) 格雷码	C (A_4)	$I_3\ I_2\ I_1\ I_0$ ($A_3A_2A_1A_0$) 格雷码	$O_3O_2O_1O_0$ ($D_3D_2D_1D_0$) 二进制码
0	0 0 0 0	0 0 0 0	1	0 0 0 0	0 0 0 0
0	0 0 0 1	0 0 0 1	1	0 0 0 1	0 0 0 1
0	0 0 1 0	0 0 1 1	1	0 0 1 0	0 0 1 1
0	0 0 1 1	0 0 1 0	1	0 0 1 1	0 0 1 0
0	0 1 0 0	0 1 1 0	1	0 1 0 0	0 1 1 1
0	0 1 0 1	0 1 1 1	1	0 1 0 1	0 1 1 0
0	0 1 1 0	0 1 0 1	1	0 1 1 0	0 1 0 0
0	0 1 1 1	0 1 0 0	1	0 1 1 1	0 1 0 1
0	1 0 0 0	1 1 0 0	1	1 0 0 0	1 1 1 1
0	1 0 0 1	1 1 0 1	1	1 0 0 1	1 1 1 0
0	1 0 1 0	1 1 1 1	1	1 0 1 0	1 1 0 0
0	1 0 1 1	1 1 1 0	1	1 0 1 1	1 1 0 1
0	1 1 0 0	1 0 1 0	1	1 1 0 0	1 0 0 0
0	1 1 0 1	1 0 1 1	1	1 1 0 1	1 0 0 1
0	1 1 1 0	1 0 0 1	1	1 1 1 0	1 0 1 1
0	1 1 1 1	1 0 0 0	1	1 1 1 1	1 0 1 0

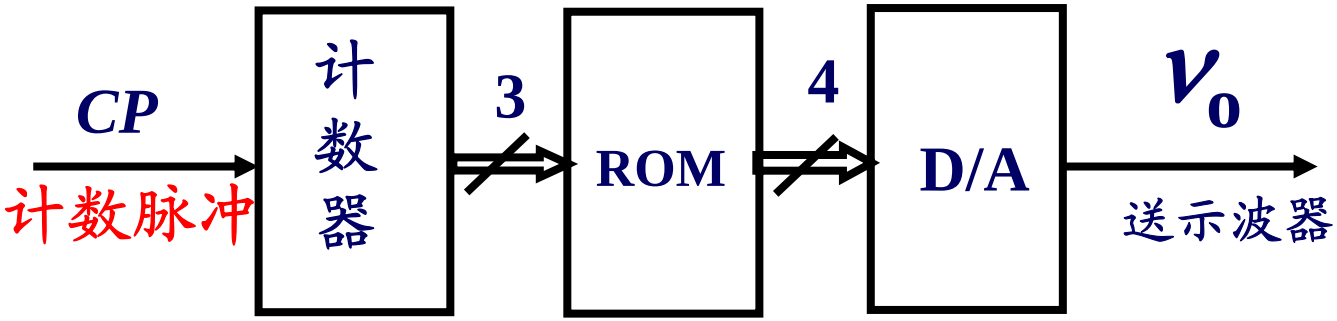
用ROM实现二进制码与格雷码相互转换的电路



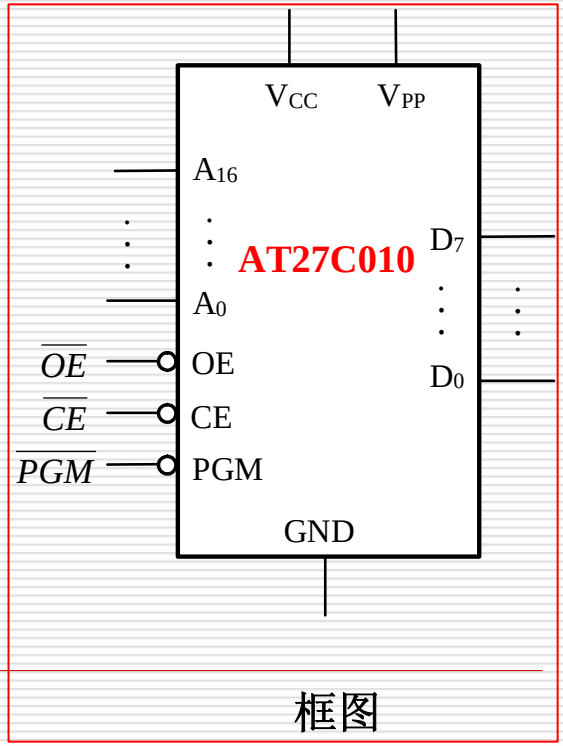


框图

（3）ROM 在波形发生器中的应用

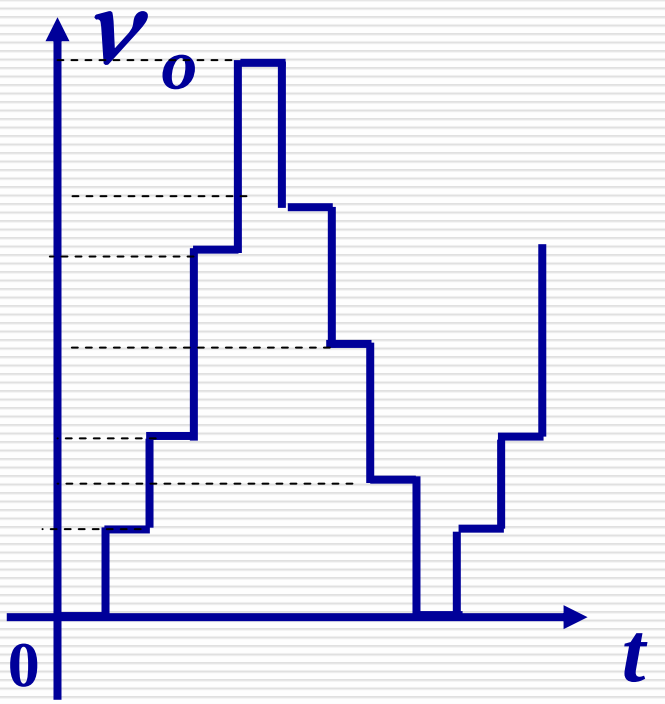


A_2 A_1 A_0	D_3 D_2 D_1 D_0	D/A
0 0 0	0 0 0 0	0
0 0 1	0 0 1 0	2
0 1 0	0 1 0 0	4
0 1 1	1 0 0 0	8
1 0 0	1 1 0 0	12
1 0 1	1 0 0 1	9
1 1 0	0 1 1 0	6
1 1 1	0 0 1 1	3



框图

A_2	A_1	A_0	D_3	D_2	D_1	D_0	D/A
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	2
0	1	0	0	1	0	0	4
0	1	1	1	0	0	0	8
1	0	0	1	1	0	0	12
1	0	1	1	0	0	1	9
1	1	0	0	1	1	0	6
1	1	1	0	0	1	1	3



7.2 随机存取存储器（RAM）

7.2 随机存取存储器（RAM）

7.2.1 静态随机存取存储器(SRAM)

1. SRAM 的基本结构及输出

$\overline{CE}$ $\overline{WE}$ $\overline{OE}$ =1XX

高阻

$\overline{CE}$ $\overline{WE}$ $\overline{OE}$ =010

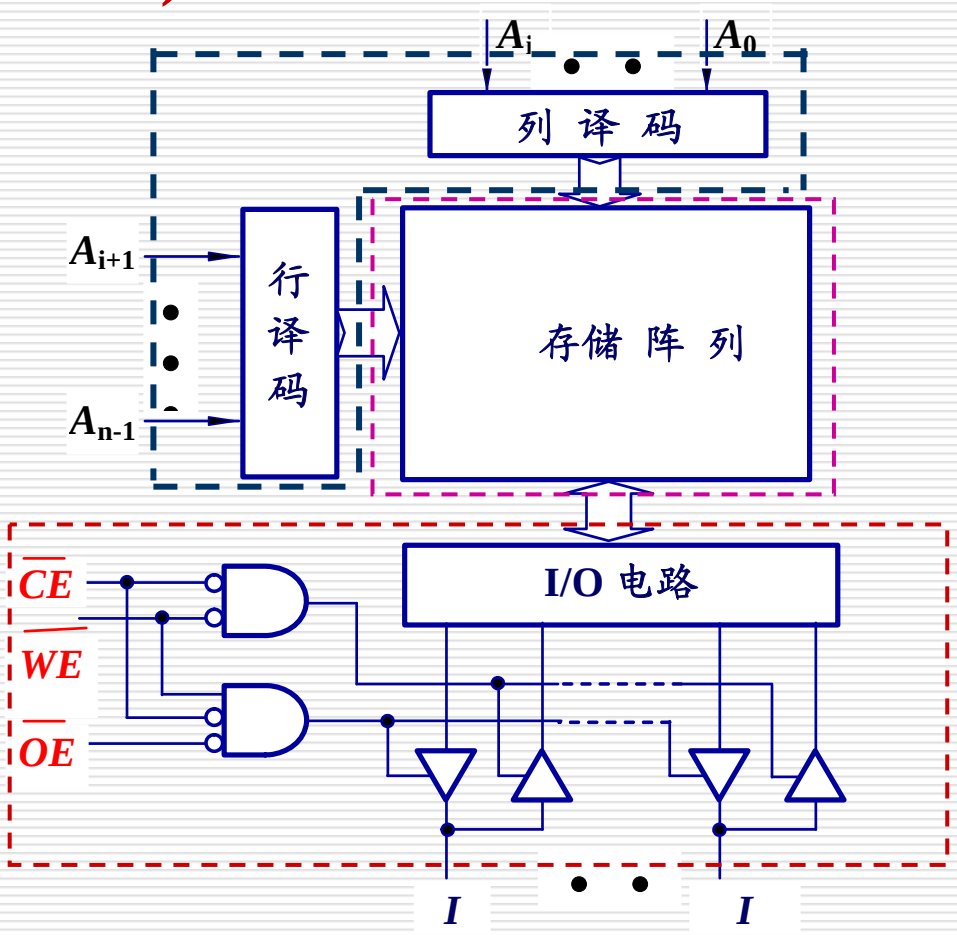
输出

$\overline{CE}$ $\overline{WE}$ $\overline{OE}$ =00X

输入

$\overline{CE}$ $\overline{WE}$ $\overline{OE}$ =011

高阻

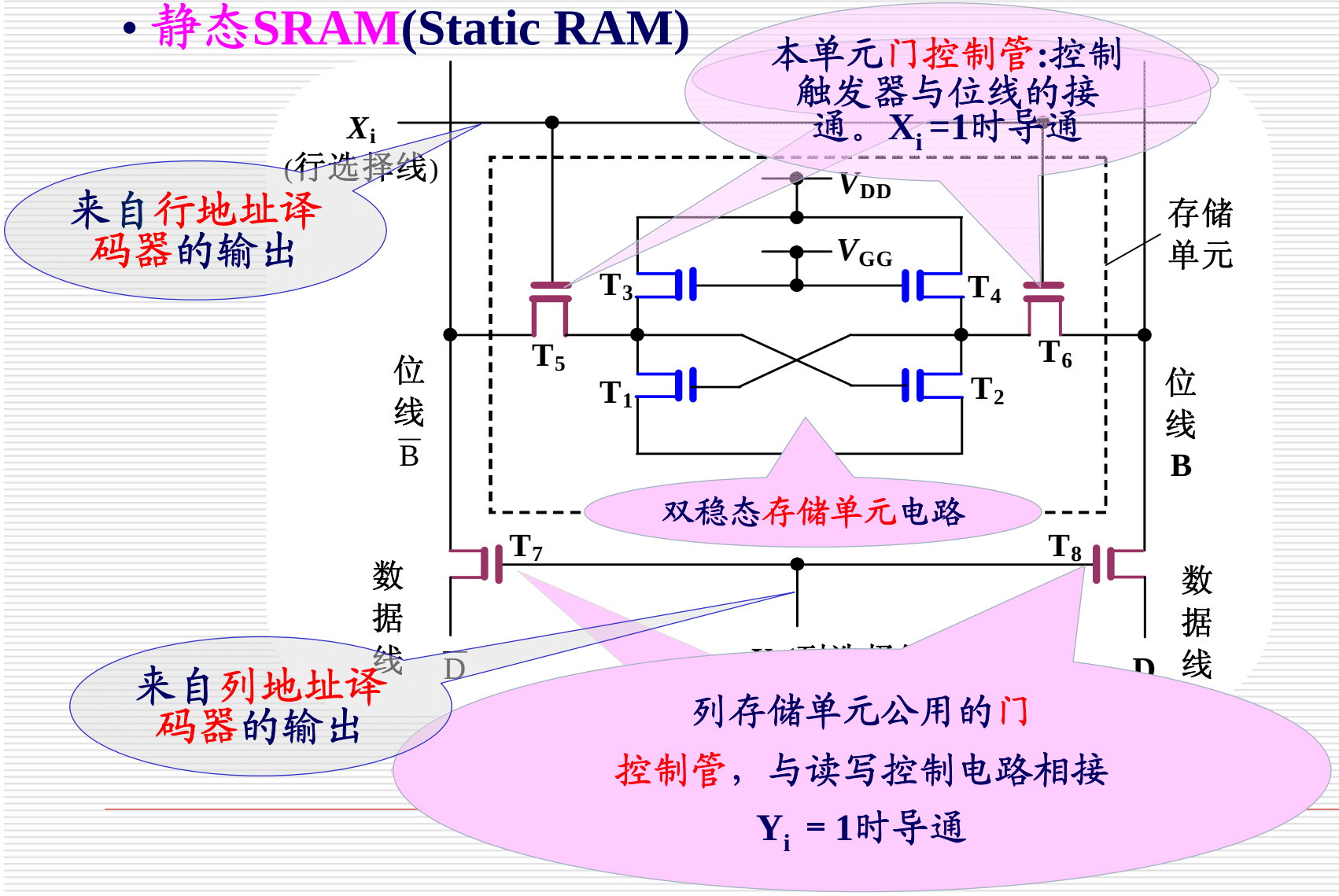


SRAM 的工作模式

工作模式	$\overline{CE}$	$\overline{WE}$	$\overline{OE}$	$I/O_0 \sim I/O_{m-1}$
保持 (微功耗)	1	X	X	高阻
读	0	1	0	数据输出
写	0	0	X	数据输入
输出无效	0	1	1	高阻

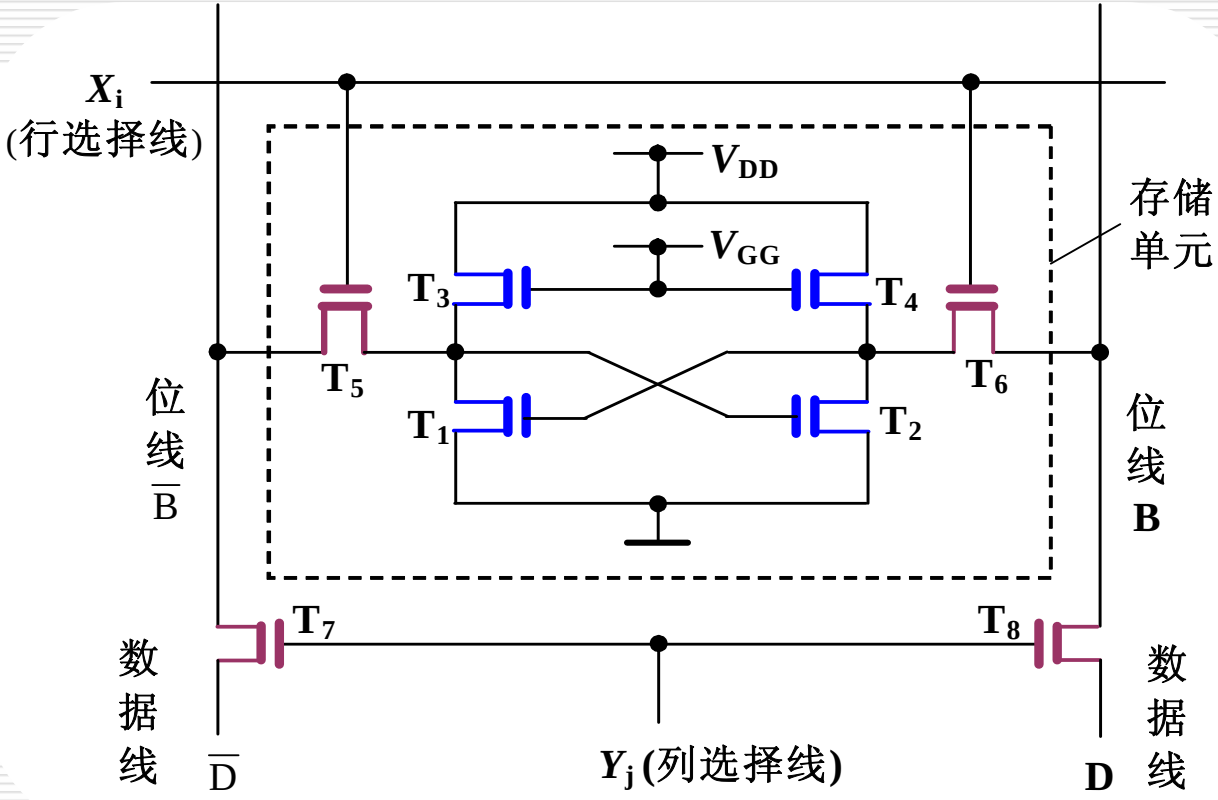
2. SRAM存储单元

- 静态SRAM(Static RAM)



2. SRAM存储单元

- 静态SRAM(Static RAM)



X_i =1

•T_5、T_6导通

触发器与位线接通

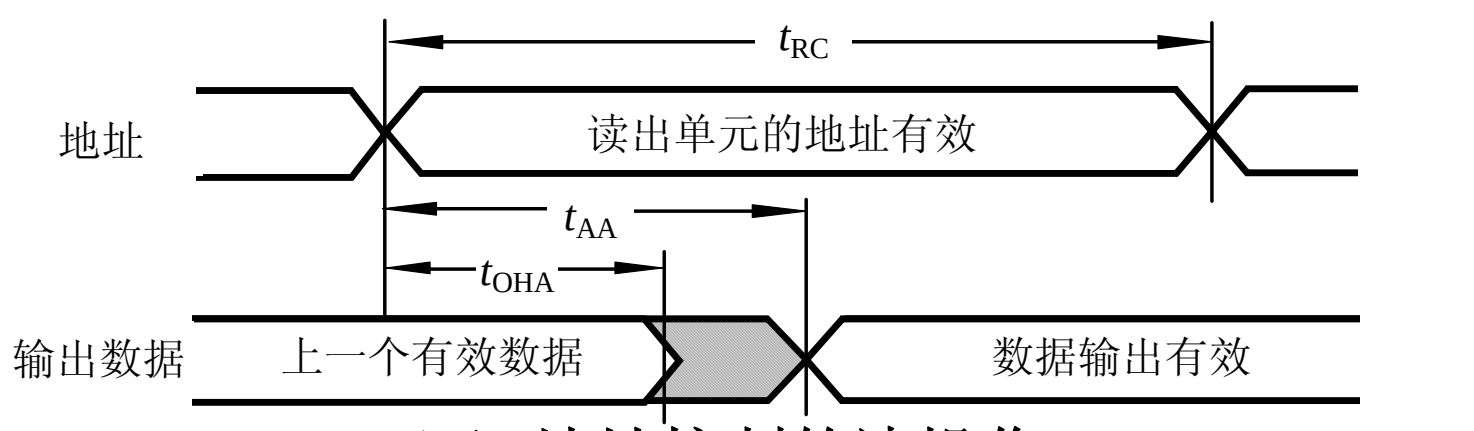
Y_j =1

•T_7、T_8均导通

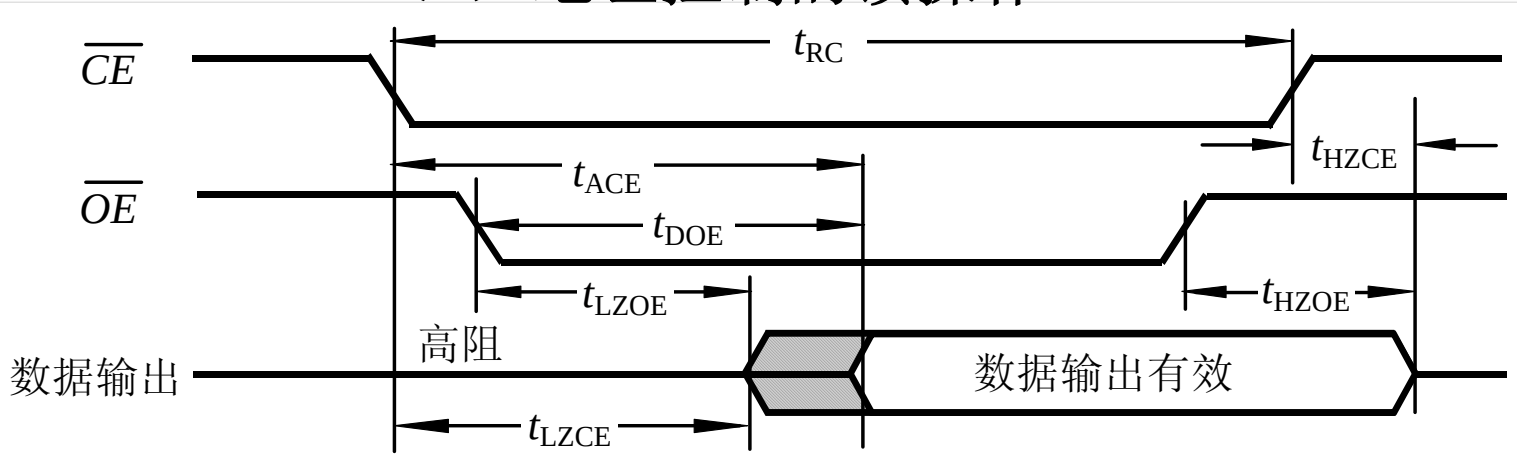
•触发器的输出与数据线接通，该单元通过数据线读取数据。

3. SRAM的读写操作及定时图

读操作定时图

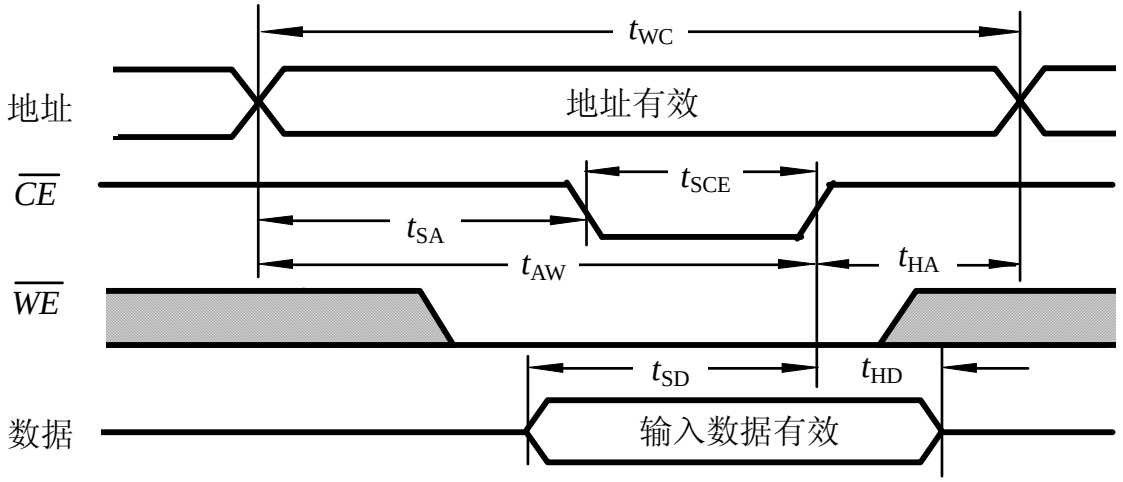


（a）地址控制的读操作

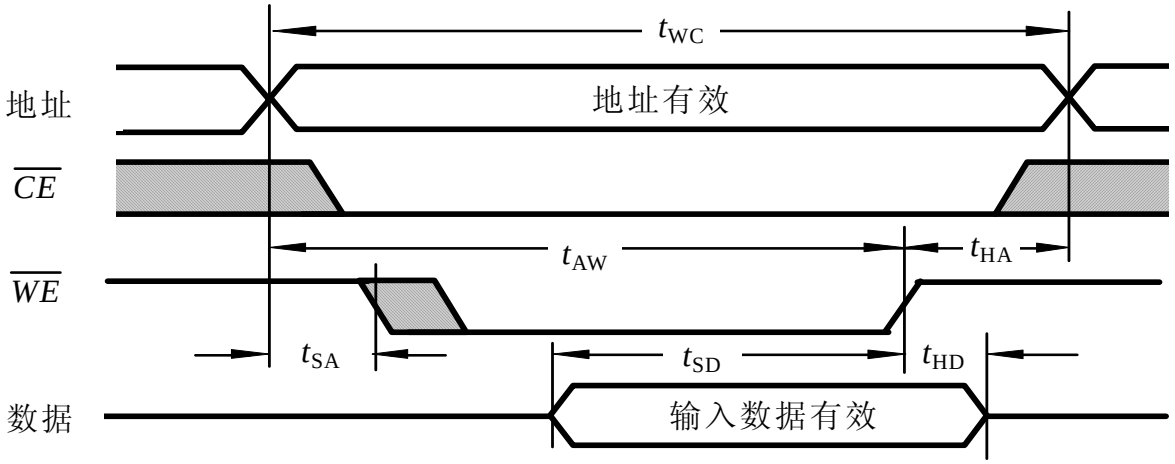


（b）片选控制的读操作

写操作定时图



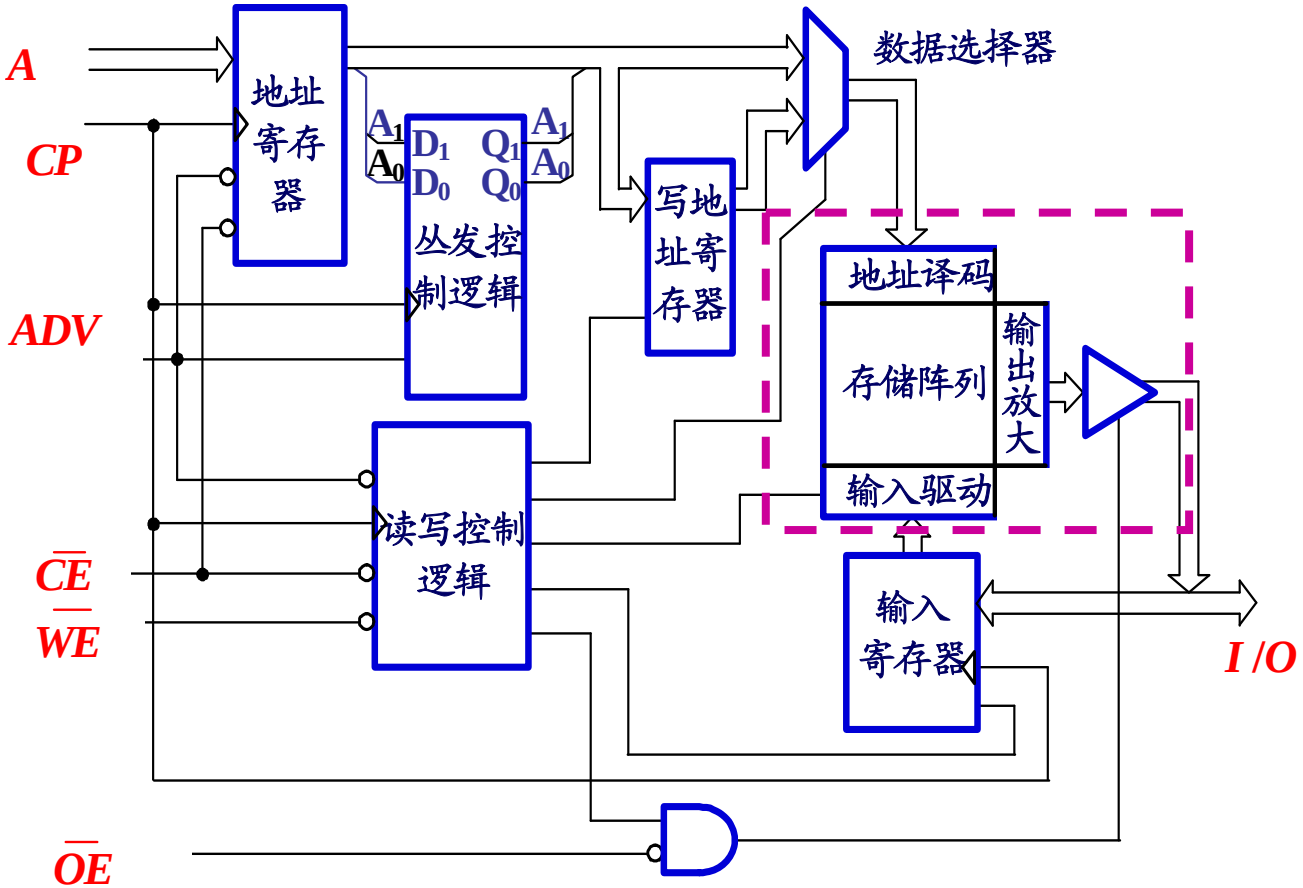
(a)片选控制的写操作



(b)写信号控制的写操

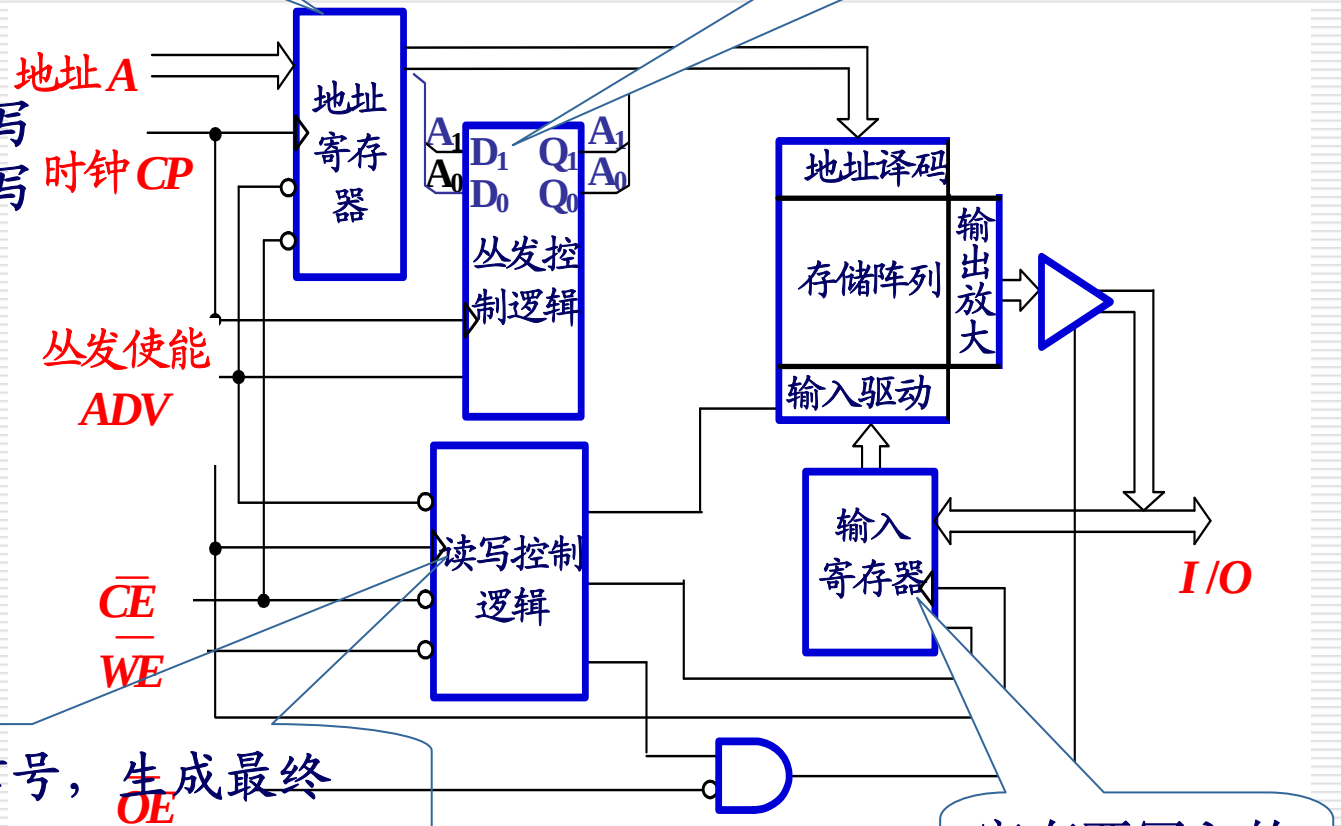
7.2.2 同步静态随机存取存储器（SSRAM）

SSRAM是一种高速RAM。与SRAM不同，SSRAM的读写操作是在时钟脉冲节拍控制下完成的。



寄存地址线上的地址

2位二进制计数器，处理A_1A_0



ADV=0:普通模式读写

ADV=1:丛发模式读写

$\overline{WE}$=0:写操作

$\overline{WE}$=1:读操作

寄存各种使能控制信号，生成最终的内部读写控制信号；

寄存要写入的数据

ADV=0:普通模式读写

$\overline{WE}$ =0:写操作　$\overline{WE}$ =1:读操作

普通模式读写:地址直接送到地址译码器，不受从发控制逻辑电路影响，按外部给定地址进行读(写)操作。

ADV=1:从发模式读写

从发模式读写：地址寄存器不接受外部新地址，在其原有的地址上，由从发计数器加1产生新地址。因此，可产生4个不同的地址。若超过4个CP后，*ADV*仍为1，则从发计数器循环计数。地址总线让出。

SSRAM的特点：

在由SSRAM构成的计算机系统中，由于在时钟有效沿到来时，地址、数据、控制等信号被锁存到SSRAM内部的寄存器中，因此读写过程的延时等待均在时钟作用下，由SSRAM内部控制完成。此时，系统中的微处理器在读写SSRAM的同时，可以处理其他任务，从而提高了整个系统的工作速度。

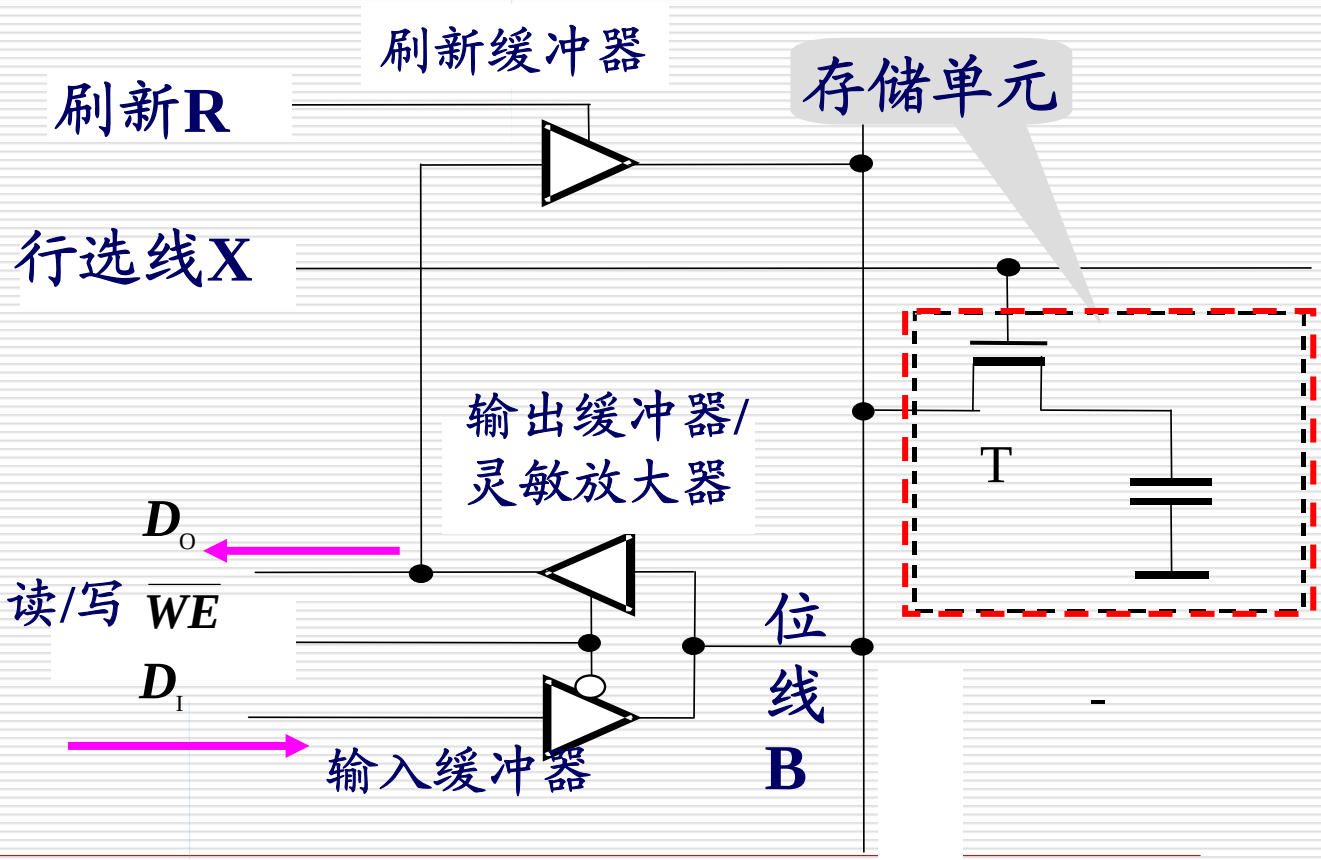
7.2.3 动态随机存取存储器(DRAM)

1、DRAM存储单元

写操作:X=1　$\overline{WE}$ =0　　T导通，电容器C与位线B连通

输入缓冲器被选通，数据D_I经缓冲器和位线写入存储单元

如果D_I为1，则向电容器充电，C存1;反之电容器放电,C存0 。

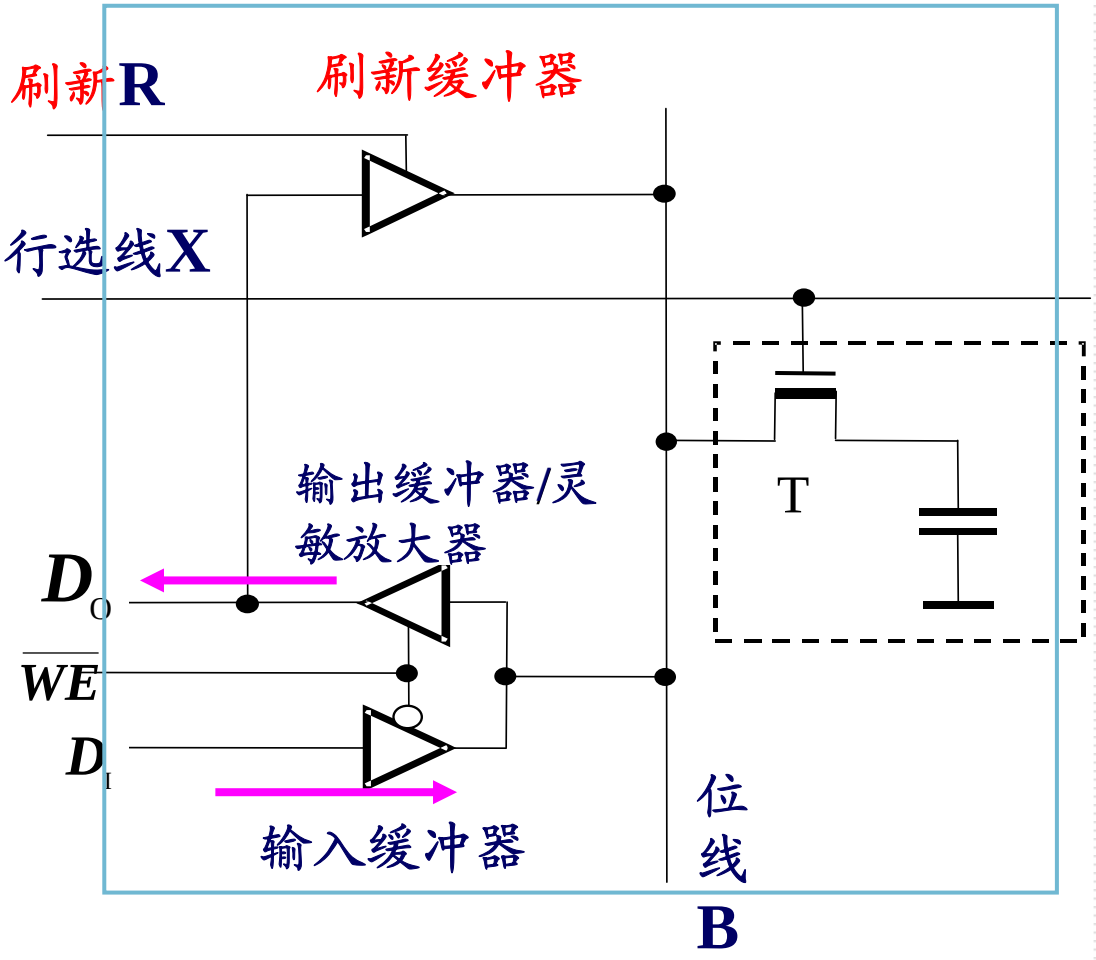


读操作:X=1　$\overline{WE}$=1

T导通，电容器C与位线B连通

输出缓冲器/灵敏放大器被选通，C中存储的数据通过位线和缓冲器输出

每次读出后，必须及时对读出单元刷新，即此时刷新控制R也为高电平，则读出的数据又经刷新缓冲器和位线对电容器C进行刷新。



7.2.4 存储器容量的扩展

1、RAM芯片的描述

2K×4

一片内的存储单元为2048

一个有效地址对应的存储元数

（1）、容量与地址线之间的计算

4K×1　　8K×4　　1K×8

表示片内有多少条地址线

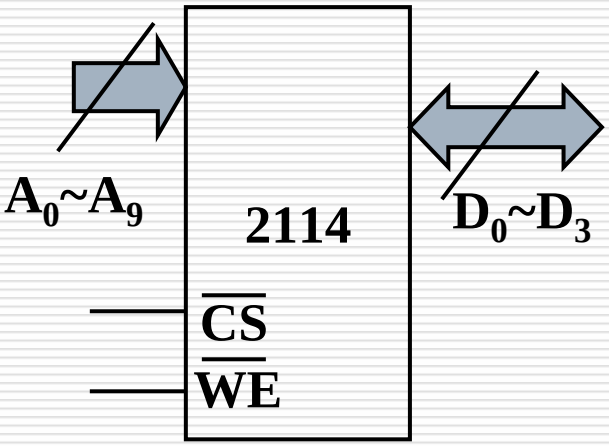
4K×1 → $4*2^{10}$

8K×4 → $8*2^{10}$

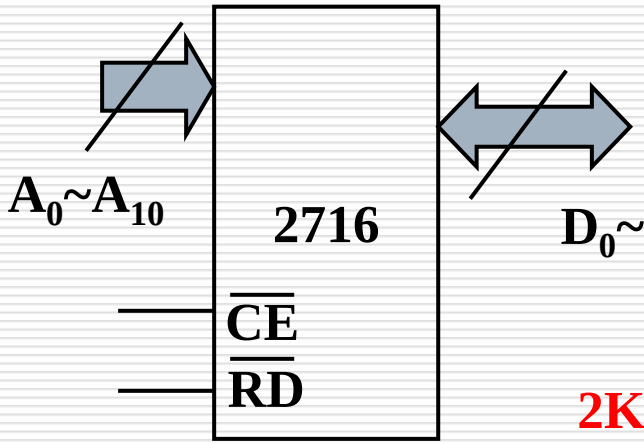
1K×8 → $1*2^{10}$

（2）、存储芯片外部接线

- 地址线
- 数据线
- 读/写线
- 片选线

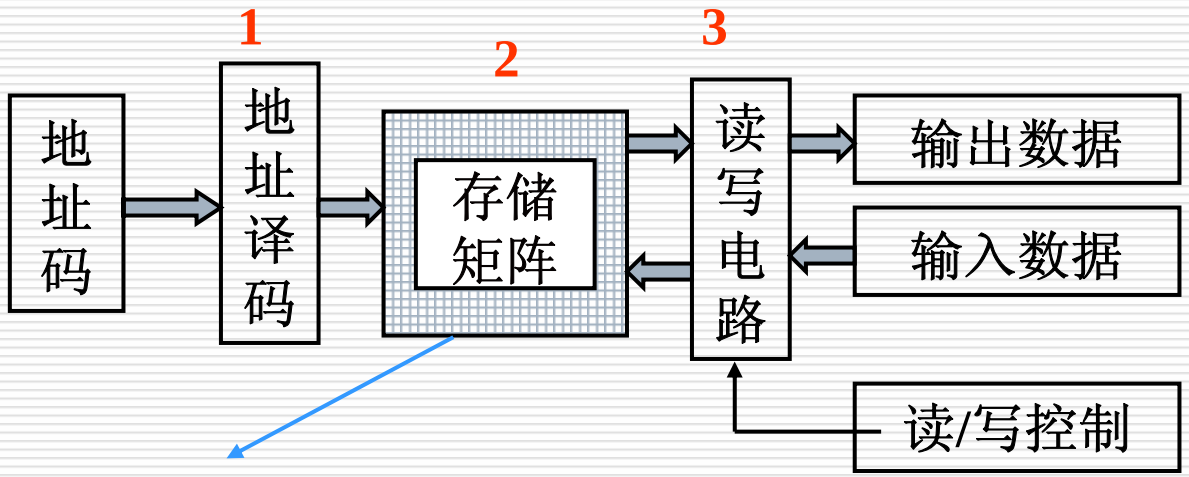


1K×4
RAM

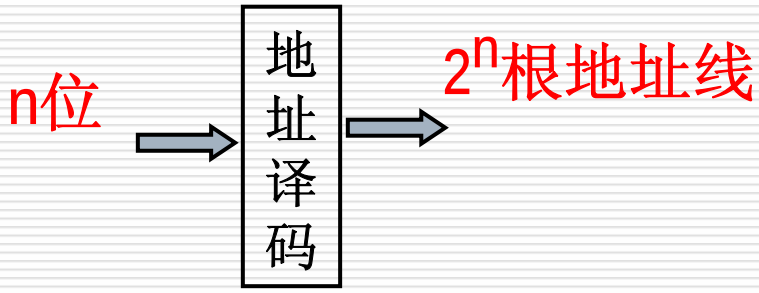


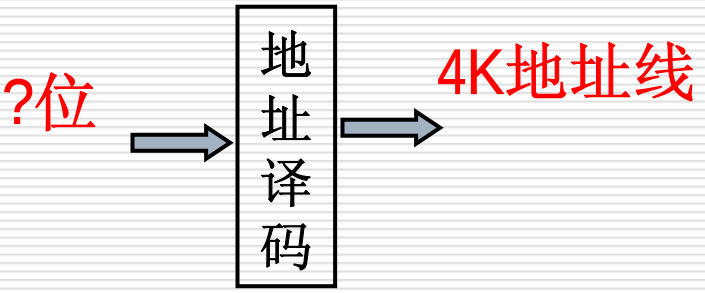
2K×8
ROM

2、RAM的结构



将若干个存储元排成矩阵形式

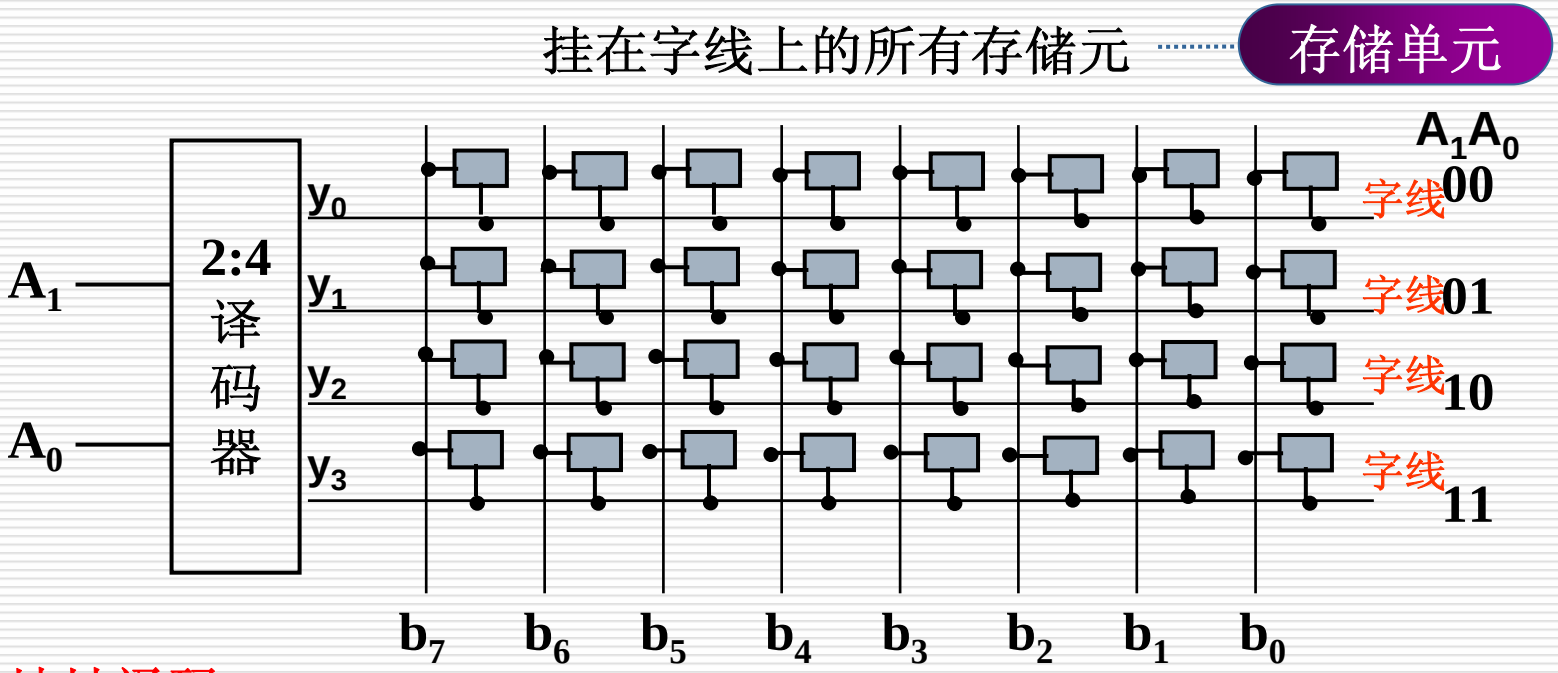




$1K = 2^{10}$

$4K = 2^{12}$

3、地址译码



* 单地址译码

4×8存储器

字线：译码器的输出线，是用来控制同一行数据的选取的。

字线的计算：单地址译码　　4条字线

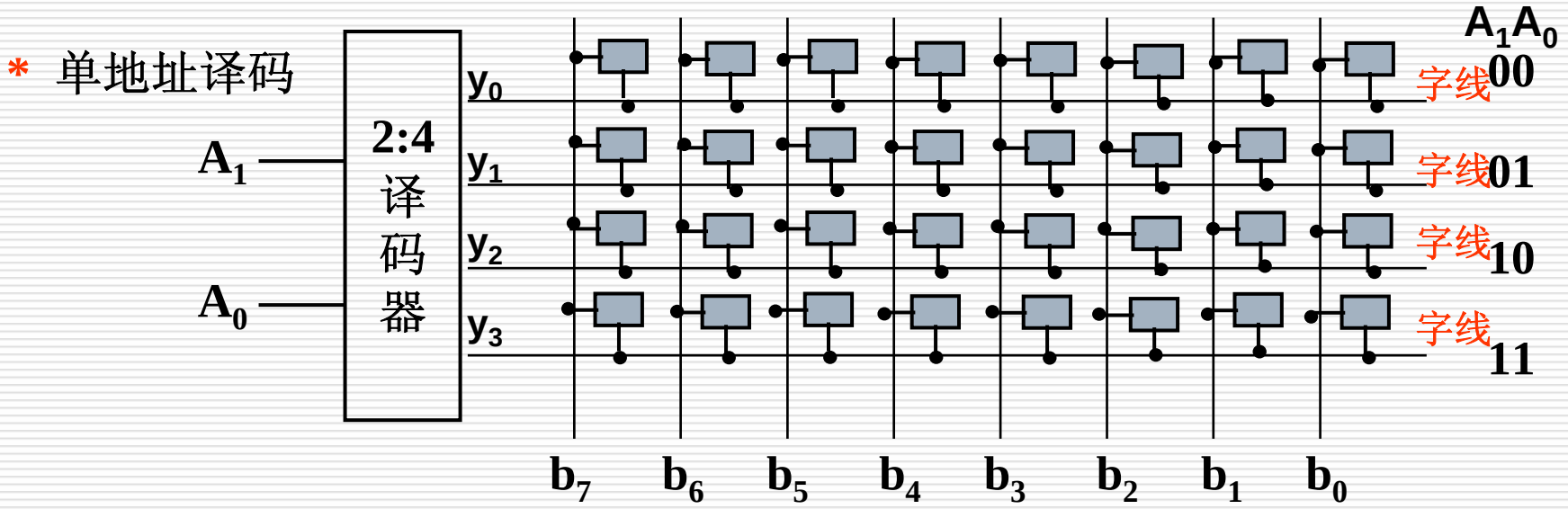
总存储单元个数的计算：

字线×位线

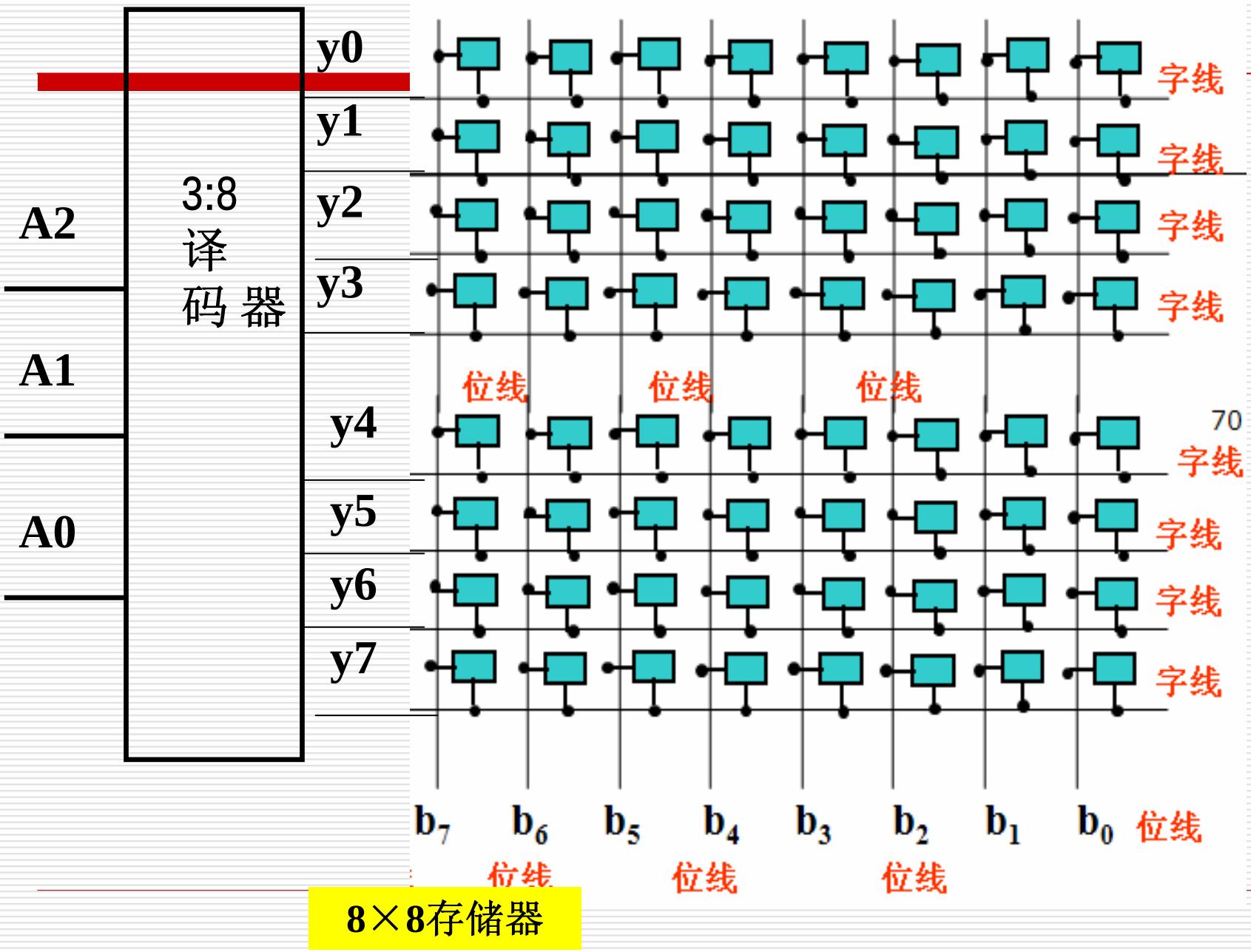
存储单元地址的表示法：

A_1A_0 ： 00，01，10，11

挂在字线上的所有存储元 …… 存储单元



4×8存储器

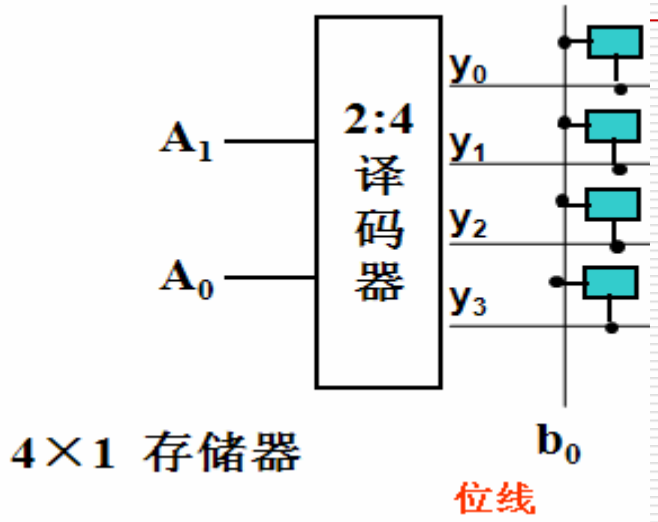

A2
A1
A0
3:8
译
码 器
y0
y1
y2
y3
y4
y5
y6
y7
字线
字线
字线
字线
位线
位线
位线
70
字线
字线
字线
字线
b7
b6
b5
b4
b3
b2
b1
b0
位线
位线
位线
位线

8×8存储器

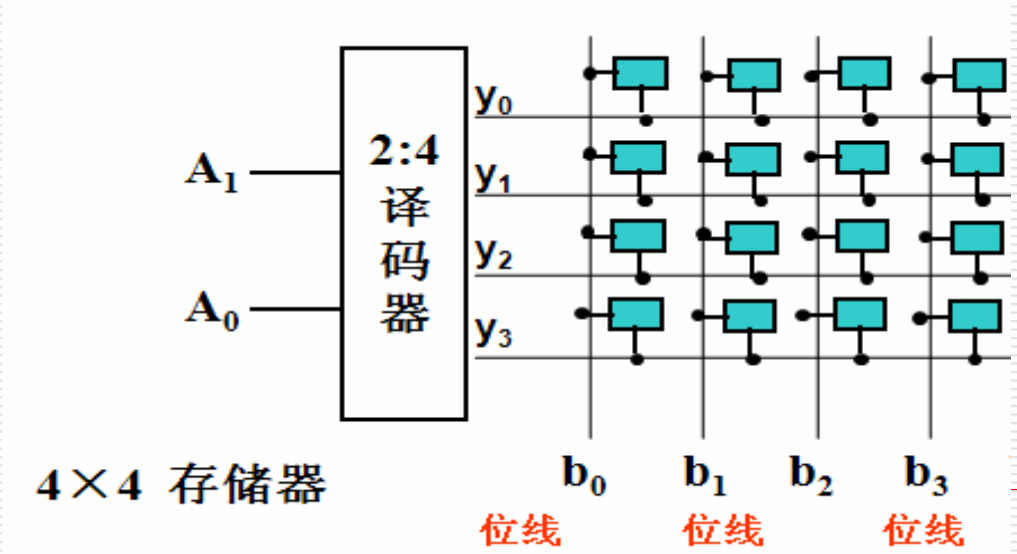
位扩展

* 单地址译码

适用于每片RAM,ROM字数够用而位数不够时



位扩展



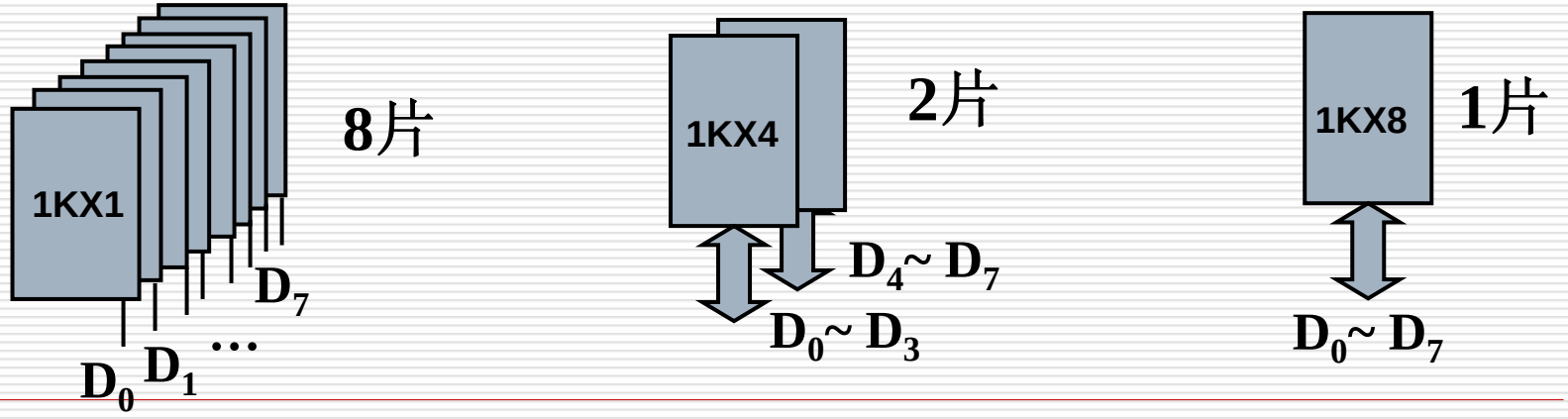
给你4片4*1芯片，怎样通过位扩展变成4*4存储器？

（1）、位扩展

- 位扩展不增加地址线。
- 原地址线、读／写控制线、输入／输出数据线并联共用。

要组成1K ×8的存储器

1K×1	8片
1K×4	2片
1K×8	1片



用4片1K×1位的RAM构成的1K×4位RAM系统

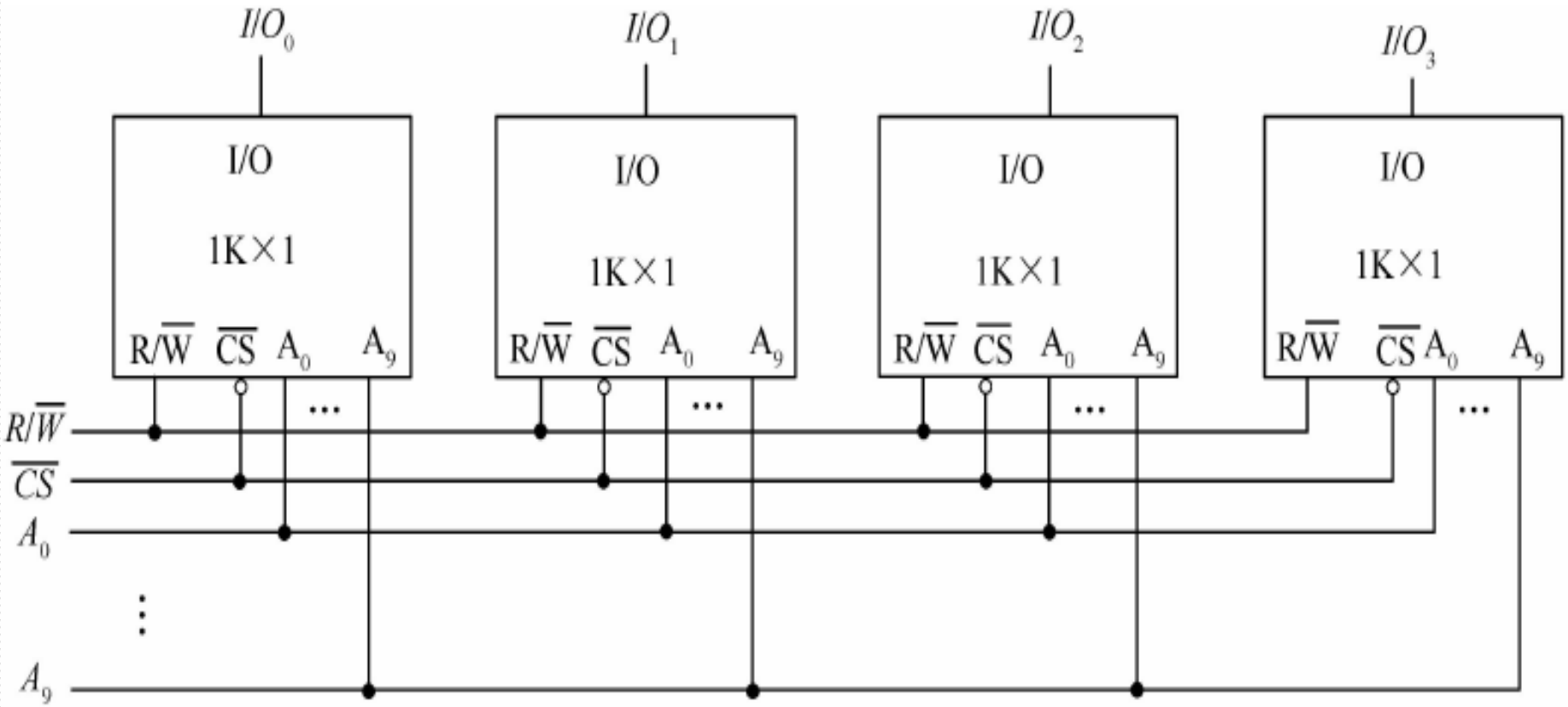


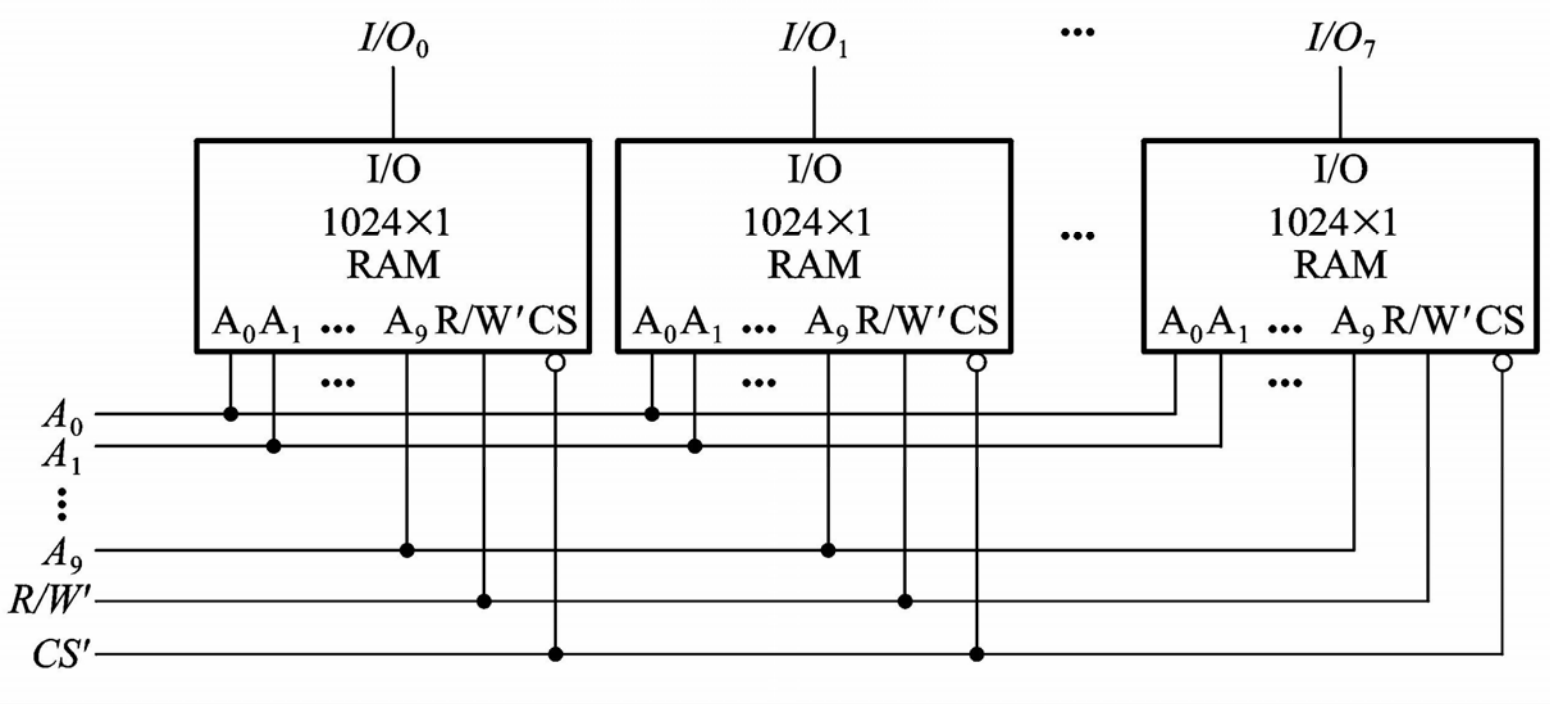
图 7-22　1K×1 位的 RAM 扩展成 1K×4 位 RAM 系统

位扩展方式

适用于每片RAM,ROM字数够用而位数不够时

接法：将各片的地址线、读写线、片选线并联即可

例：用八片1024 x 1位→ 1024 x 8位的RAM

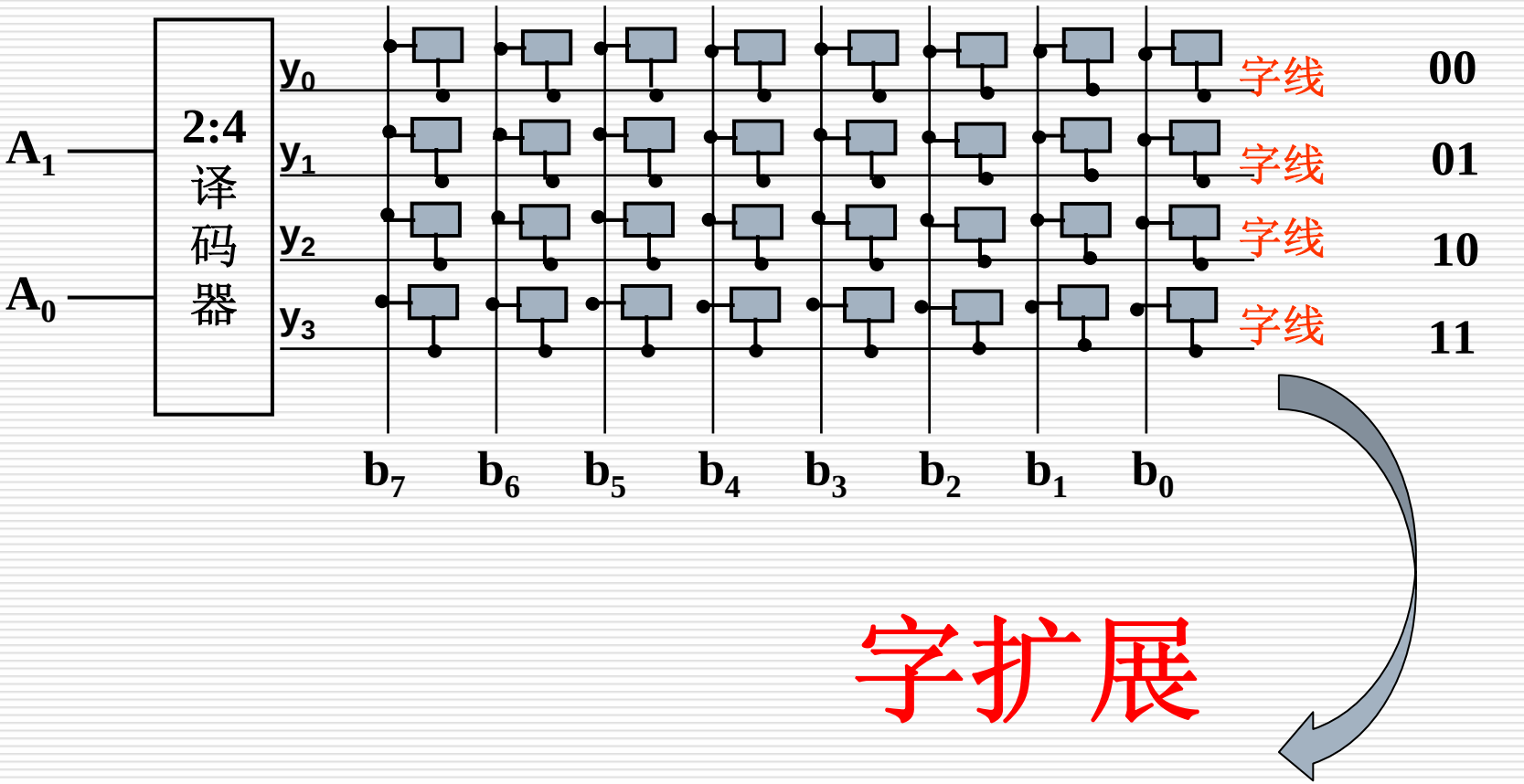


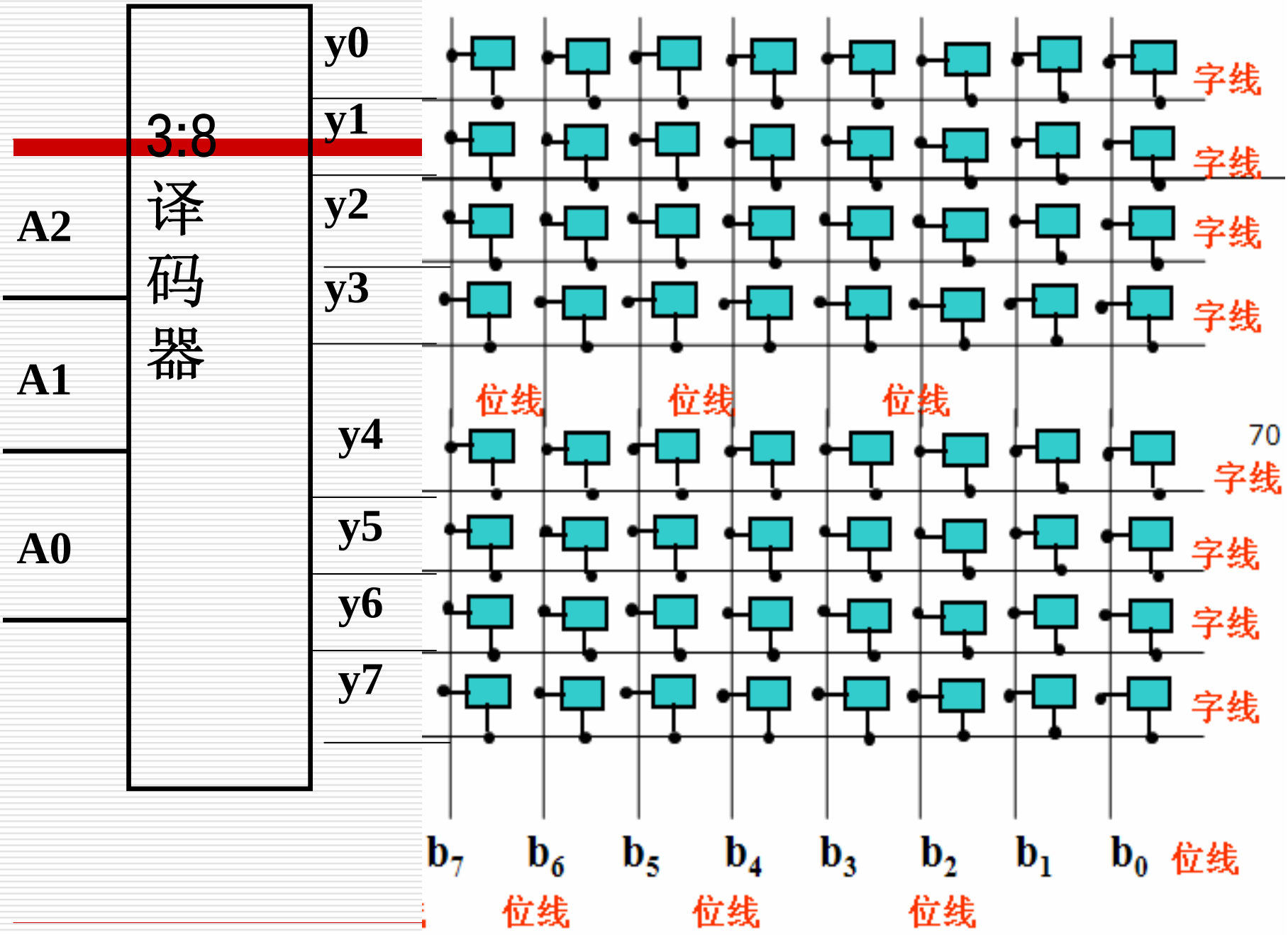
字扩展

- 扩展字数需要增加地址线。
- 可以利用外加译码器，控制存储器芯片的片选输入端来实现。
- 原地址线、读／写控制线、输入／输出数据线并联共用。

挂在字线上的所有存储元 …… 存储单元

适用于每片RAM,ROM位数够用而字数不够时.




3:8
译
码
器
A2
A1
A0
y0
y1
y2
y3
y4
y5
y6
y7
字线
字线
字线
字线
位线
位线
位线
70
字线
字线
字线
字线
b_7
b_6
b_5
b_4
b_3
b_2
b_1
b_0
位线
位线
位线
位线

（2）、字扩展

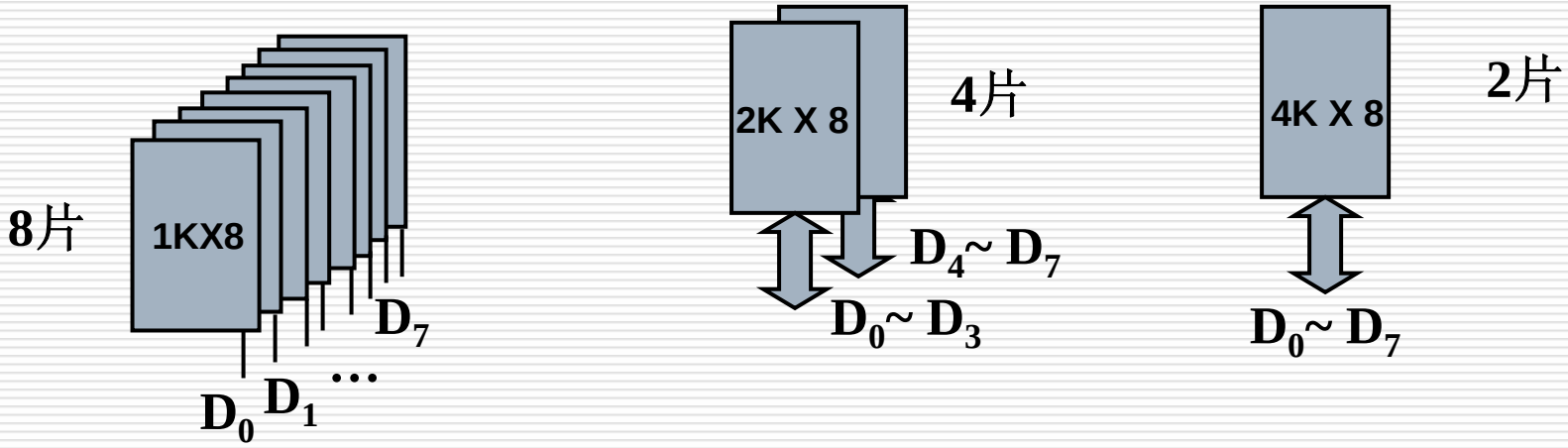
要组成 8K ×8的存储器

1K×8　8片

2K×8　4片

4K×8　2片

组成 8K ×8的存储器



将4片8K×8位的RAM扩展为32K×8位的存储器系统的示意图

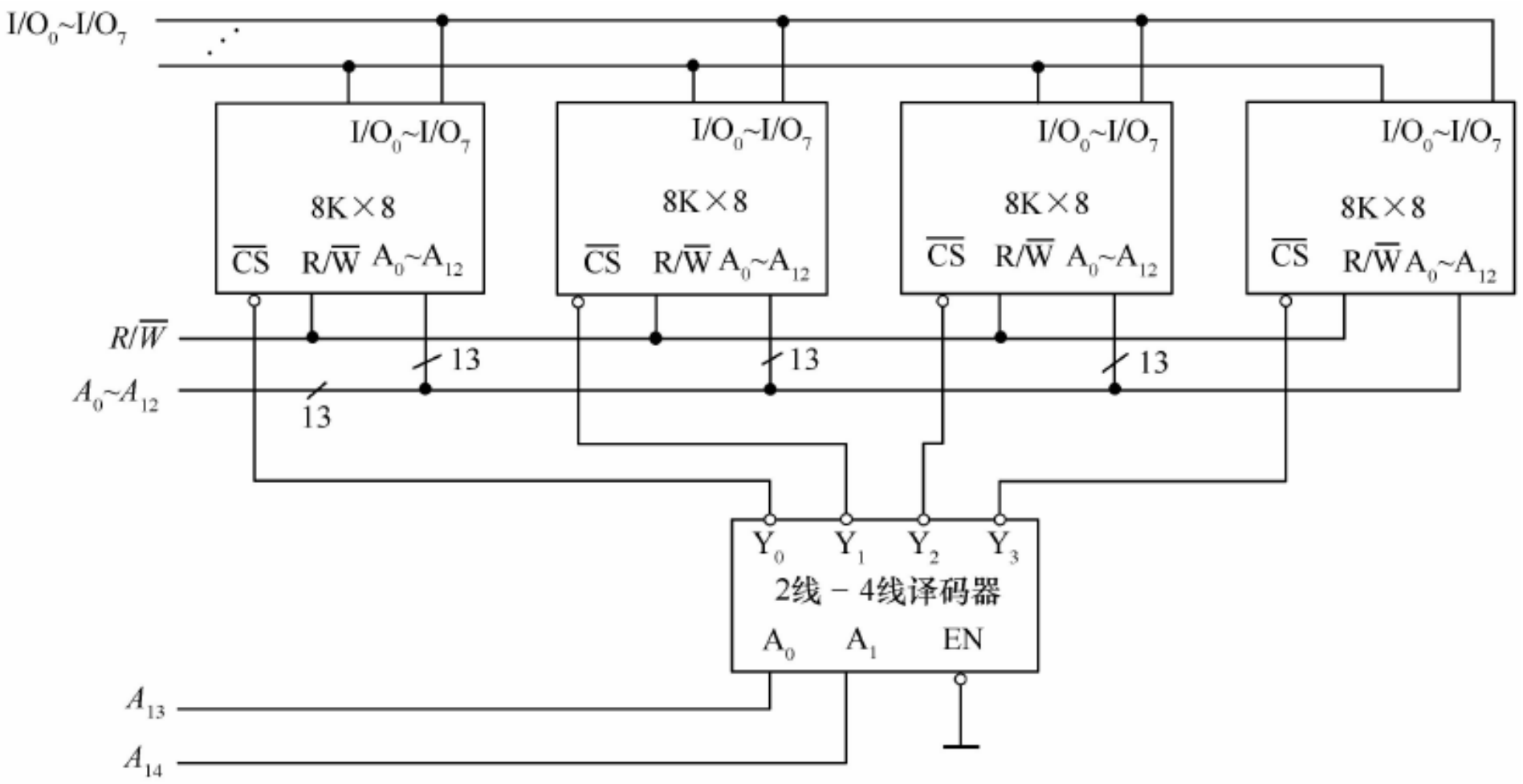
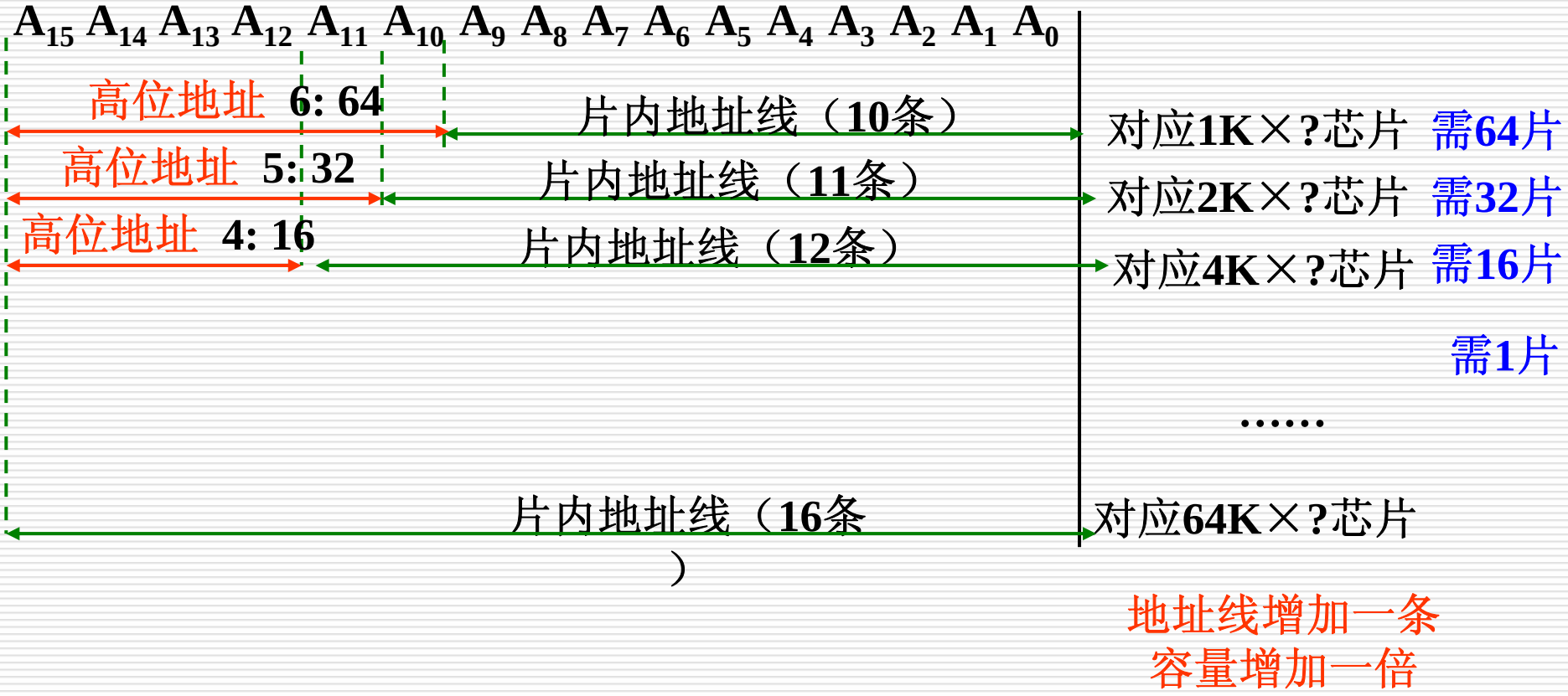


图 7－23　4 片 8K×8 位的 RAM 扩展成 32K×8 位的存储系统

组成64K的存储器



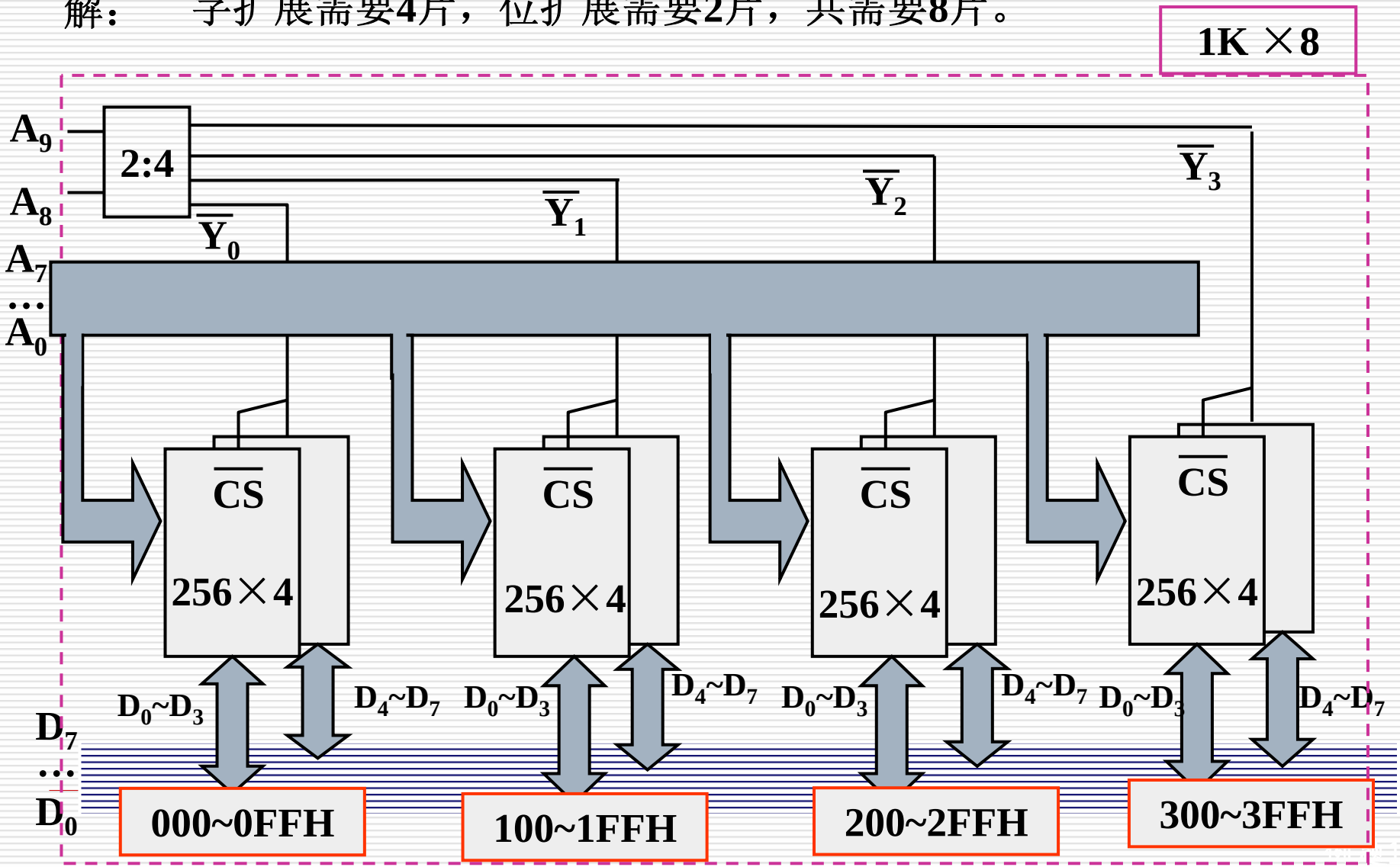
地址线增加一条
容量增加一倍

例1

现有256×4的存储芯片若干，试问要组成1K×8的存储器需要芯片多少片？画出连线图。

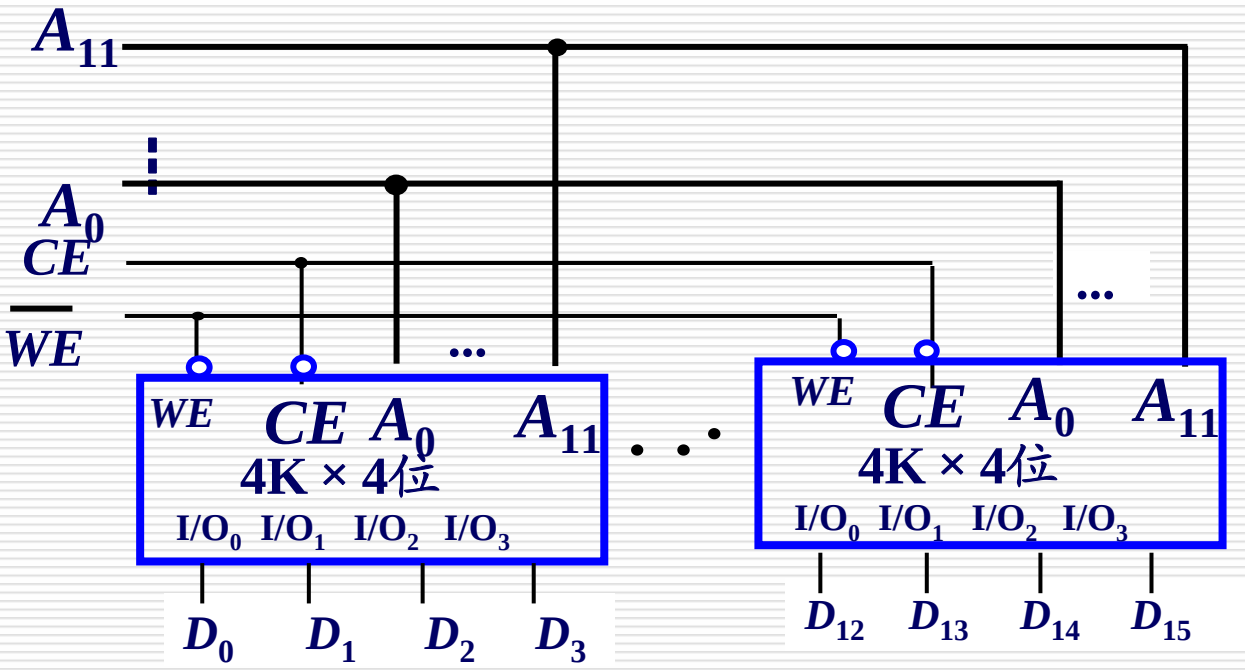
字位都扩展

解：　字扩展需要4片，位扩展需要2片，共需要8片。



存储器容量扩展实例

1. 字长（位数）的扩展**---**用**4KX4**位的芯片组成**4KX16**位的存储系统。

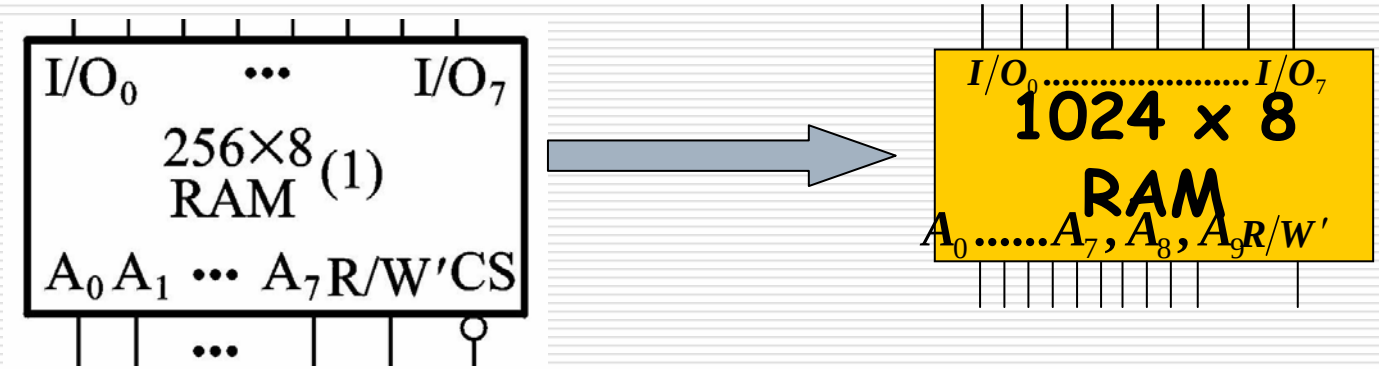


位扩展可以利用芯片的并联方式实现。

字扩展方式

适用于每片RAM,ROM位数够用而字数不够时

例：用四片256 × 8位→1024 × 8位 RAM



数据线：$I/O_0 \sim I/O_7$

地址线：$A_0 \sim A_7$

读/写信号：R/W'

片选信号：CS'

数据线：$I/O_0 \sim I/O_7$

地址线：$A_0 \sim A_7, A_8, A_9$

读/写信号：R/W'

每一片提供256个字，需要256个地址($A_{0\sim7}$：0～0－－－－1～1)

用A_9，A_8两位代码区分四片

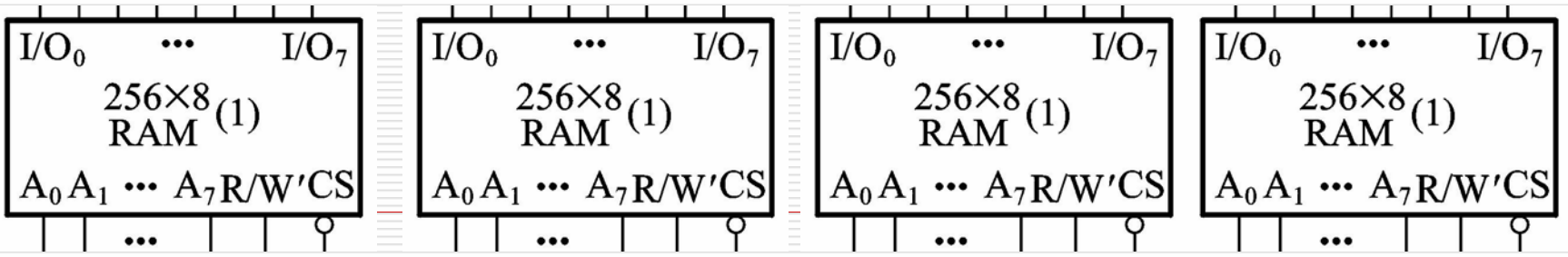
即将A_9A_8译成$Y_0'\sim Y_3'$，分别接四片的CS'

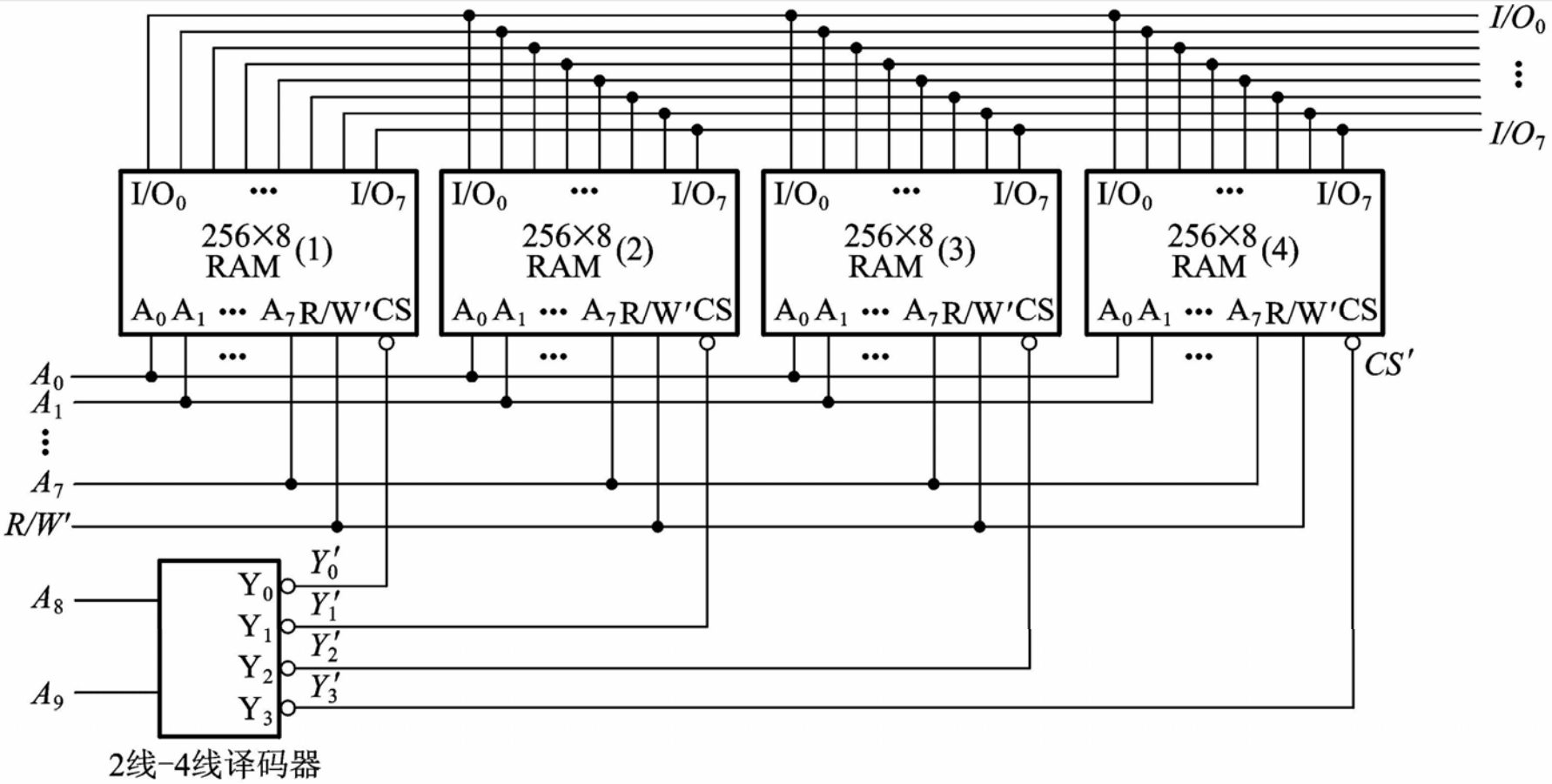
A_9	A_8	CS_1'	CS_2'	CS_3'	CS_4'
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

四片的地址分配就是：

00 $A_7\sim A_0$，　01 $A_7\sim A_0$，　10 $A_7\sim A_0$，　11 $A_7\sim A_0$

0 ～ 255　　256 ～ 511　　512 ～ 767　　768 ～ 1023

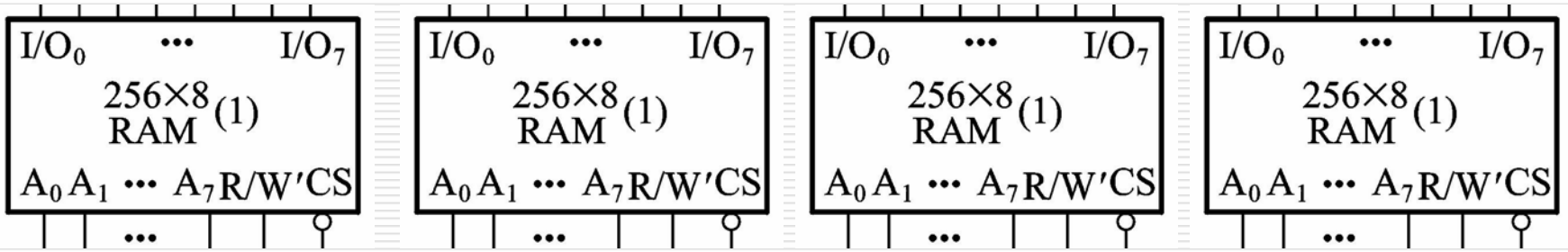


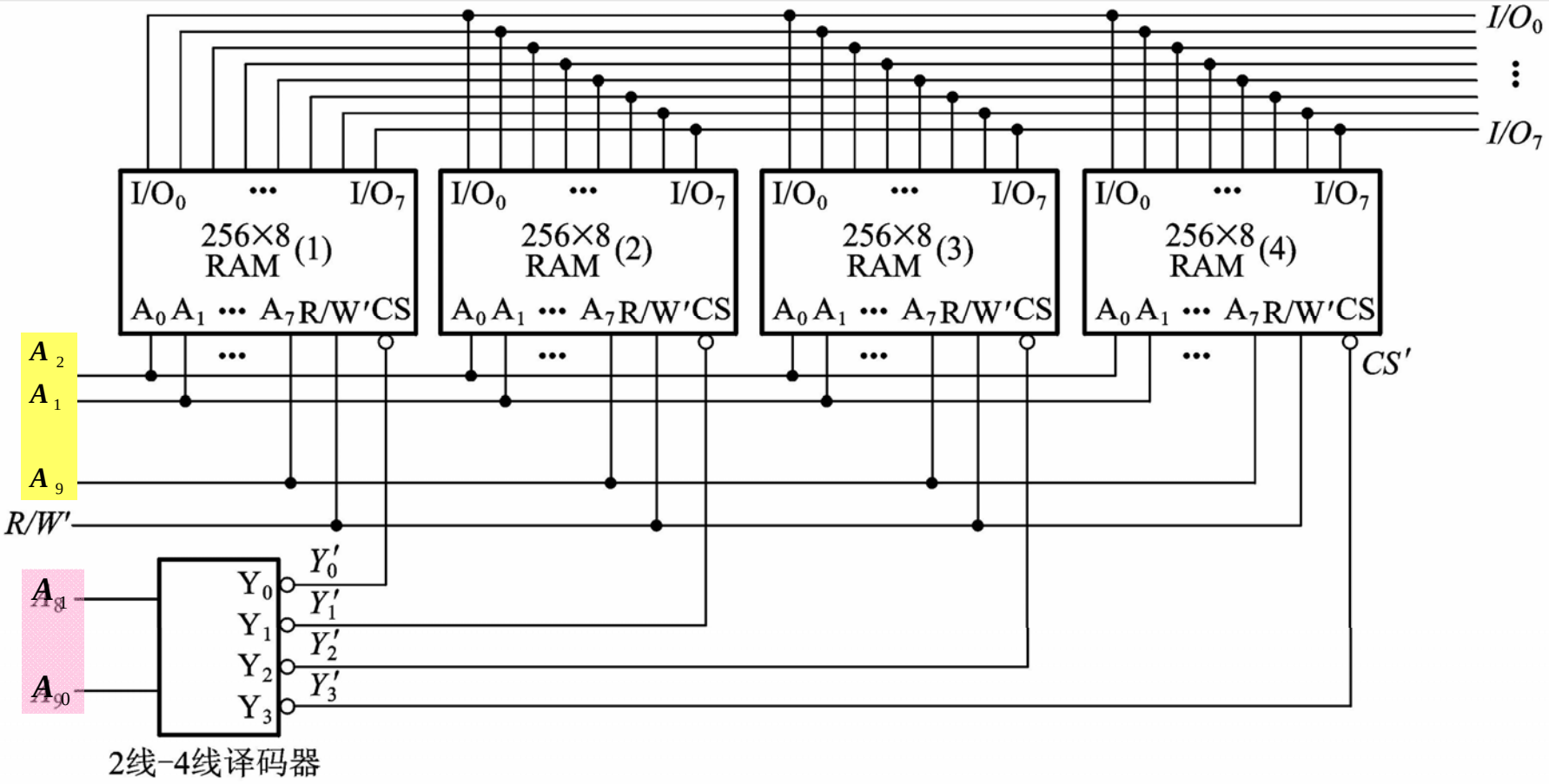

I/O0
I/O7
256×8 RAM (1)
256×8 RAM (2)
256×8 RAM (3)
256×8 RAM (4)
A0 A1 ··· A7 R/W′ CS
A0
A1
A7
R/W′
CS′
A8
A9
Y0
Y1
Y2
Y3
Y0′
Y1′
Y2′
Y3′
2线-4线译码器

A_1	A_0	CS_1'	CS_2'	CS_3'	CS_4'
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

四片的地址分配就是:

$A_9 \sim A_2 00$，　$A_9 \sim A_2 01$，　$A_9 \sim A_2 10$，　$A_9 \sim A_2 11$




I/O0
I/O7
256×8 RAM (1)
256×8 RAM (2)
256×8 RAM (3)
256×8 RAM (4)
I/O0 ··· I/O7
A0 A1 ··· A7 R/W′ CS
A2
A1
A9
R/W′
CS′
A81
A80
Y0
Y1
Y2
Y3
Y0′
Y1′
Y2′
Y3′
2线-4线译码器

存储器容量扩展实例

2. 字数的扩展—用8K×8位的芯片组成32K × 8位的存储系统。

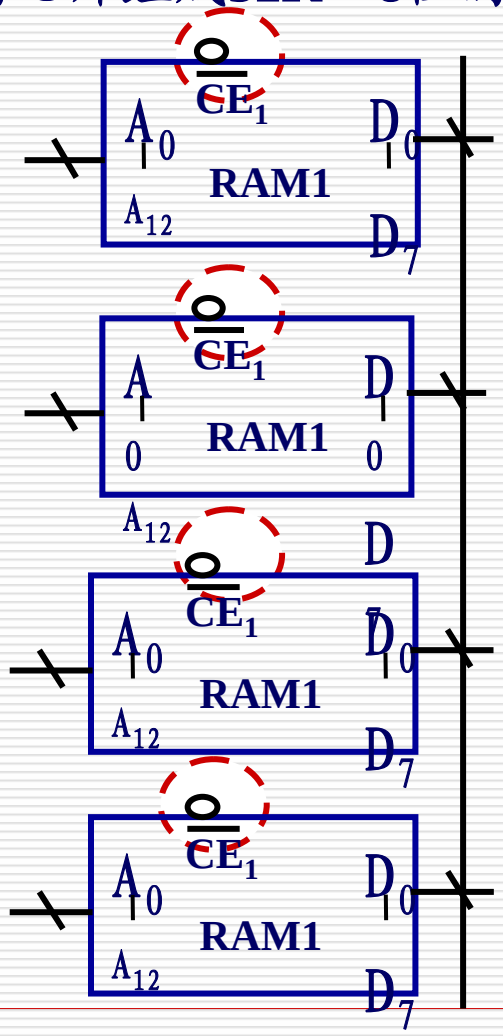
芯片:$A_0 \sim A_{12}$

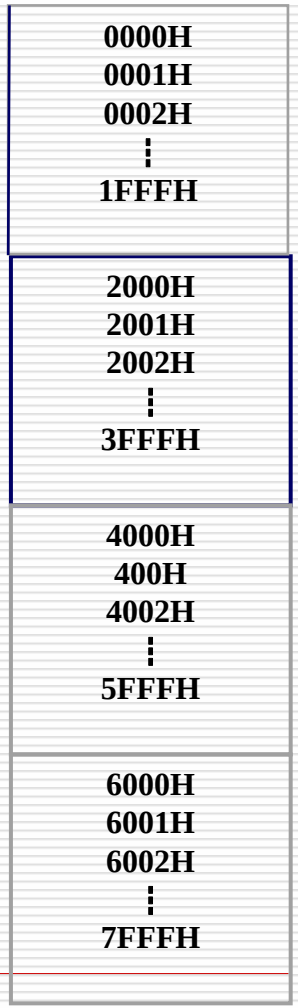
芯片数=4

系统地址线数=15

系统:$A_0 \sim A_{14}$

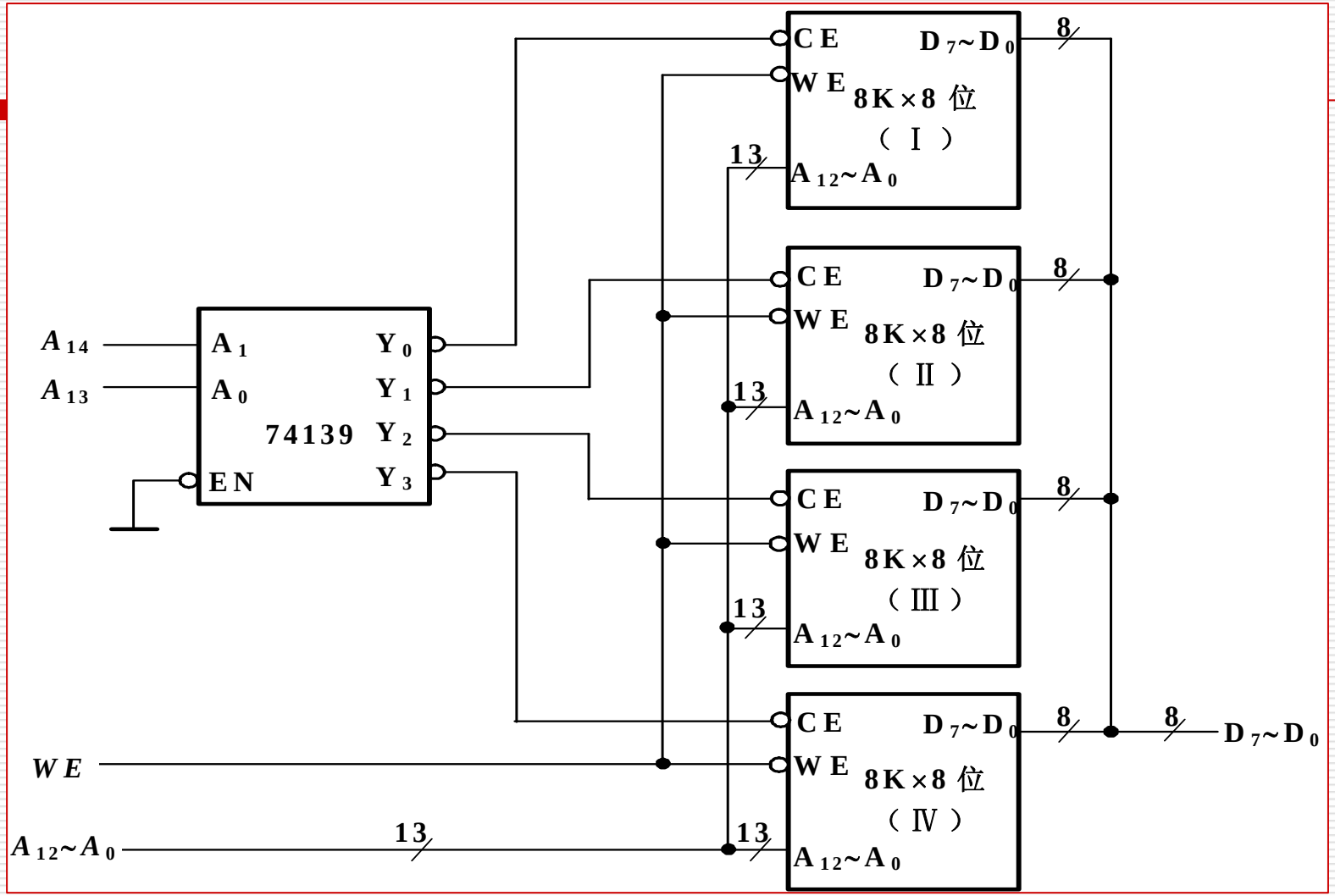
$A_{13} \sim A_{14}$?





32K × 8位存储器系统的地址分配表

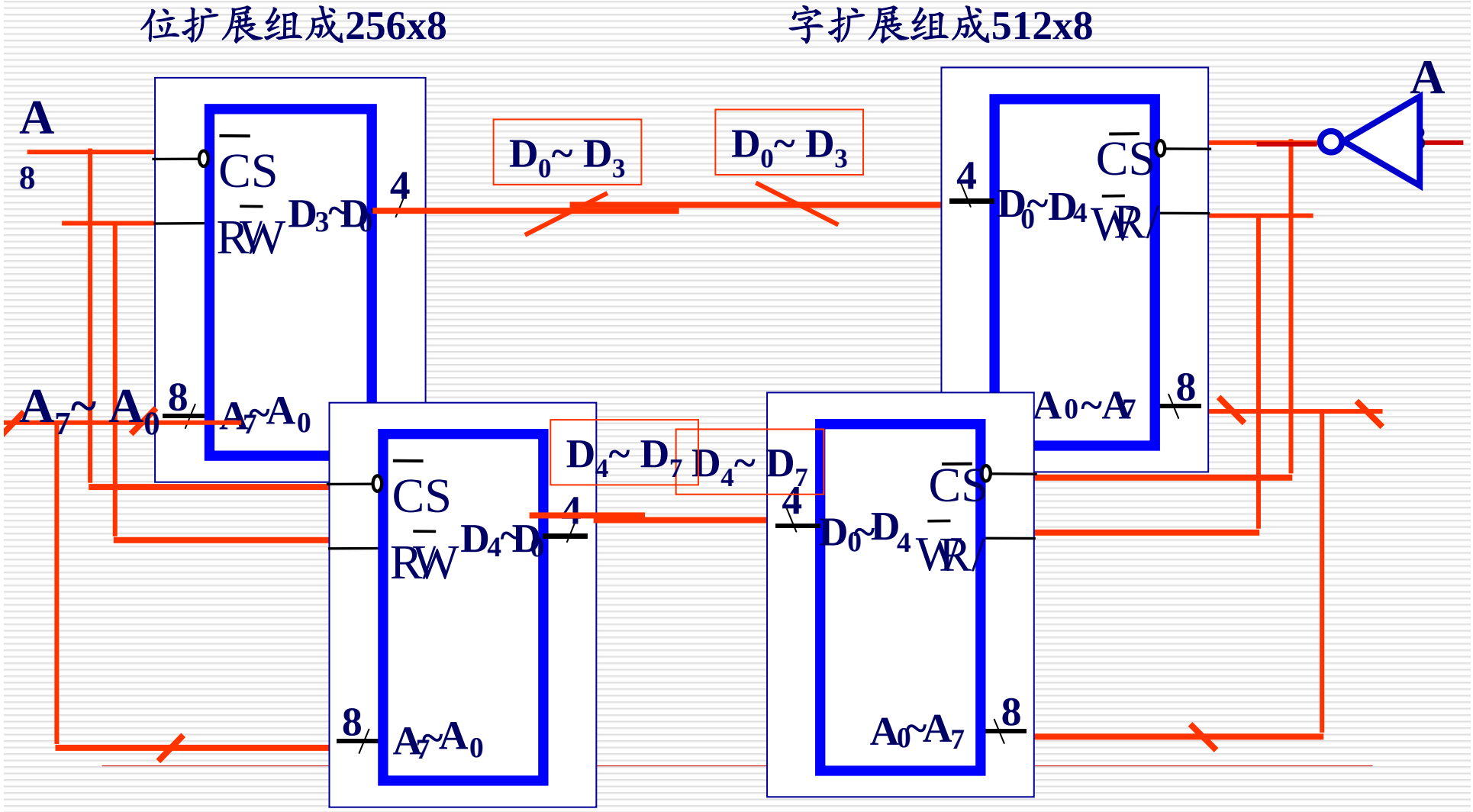
各RAM芯片	译码器有效输出端	扩展的地址输入端 A_{14} A_{13}	8K × 8位RAM芯片地址输入端 A_{12} A_{11} A_{10} A_9 A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0	对应的十六进制地址码
I	Y_0	0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 ⋮ 1 1 1 1 1 1 1 1 1 1 1 1	0000H 0001H 0002H ⋮ 1FFFH
II	Y_1	0 1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 ⋮ 1 1 1 1 1 1 1 1 1 1 1 1	2000H 2001H 2002H ⋮ 3FFFH
III	Y_2	1 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 ⋮ 1 1 1 1 1 1 1 1 1 1 1 1	4000H 400H 4002H ⋮ 5FFFH
IV	Y_3	1 1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 ⋮ 1 1 1 1 1 1 1 1 1 1 1 1	6000H 6001H 6002H ⋮ 7FFFH



字数的扩展可以利用外加译码器控制存储器芯片的片选输入端来实现。

3.同时实现字数、位数的扩展

使用256×4位芯片组成512×8位存储器，问需要多少芯片？电路应如何连接？



7.2.5 RAM应用举例—LED点阵显示屏