第四章　存储逻辑

- 存储逻辑是时序逻辑和组合逻辑结合的产物。
- 存储有限个字(比特)的逻辑电路，称为存储逻辑部件。

4.1 特殊存储部件

4.2 随机读写存储器

4.3 只读存储器

4.4 FLASH存储器

4.5 存储器容量扩展*

4.1 特殊存储部件

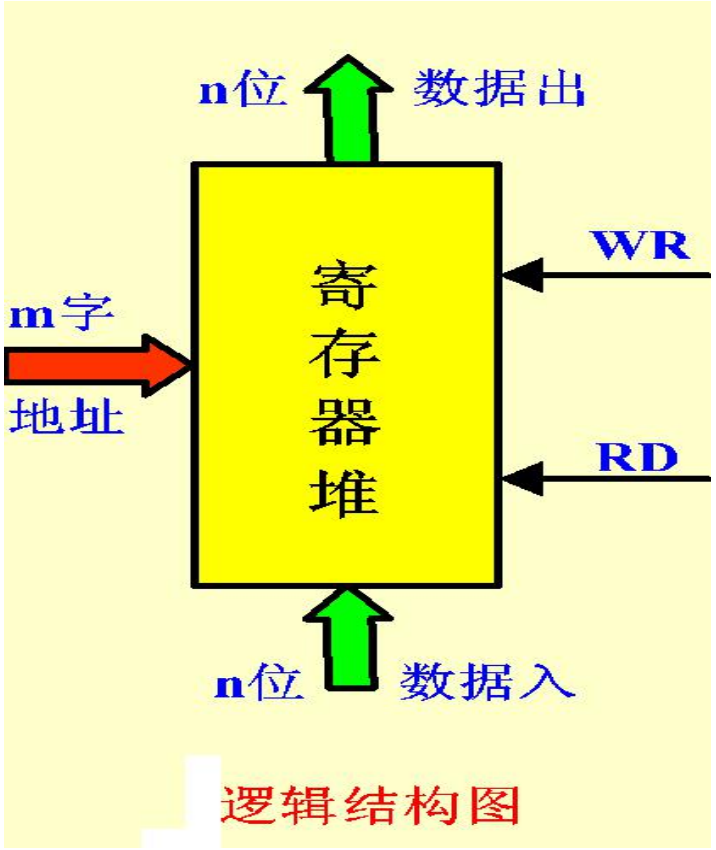
特殊存储部件包括：寄存器堆、寄存器队列、寄存器堆栈，均为小容量存储器,由寄存器组成。

特点：

- 寄存器组成
- 存储容量小
- 结构简单
- 工作速度快

1　寄存器堆

- 由寄存器组组成
- 数目是4、8、16、32…
- 一维数组结构
- 字（n位）为存储单位
- 输入（写）\输出（读）
- 地址存取方式
- 分单/双端口输出



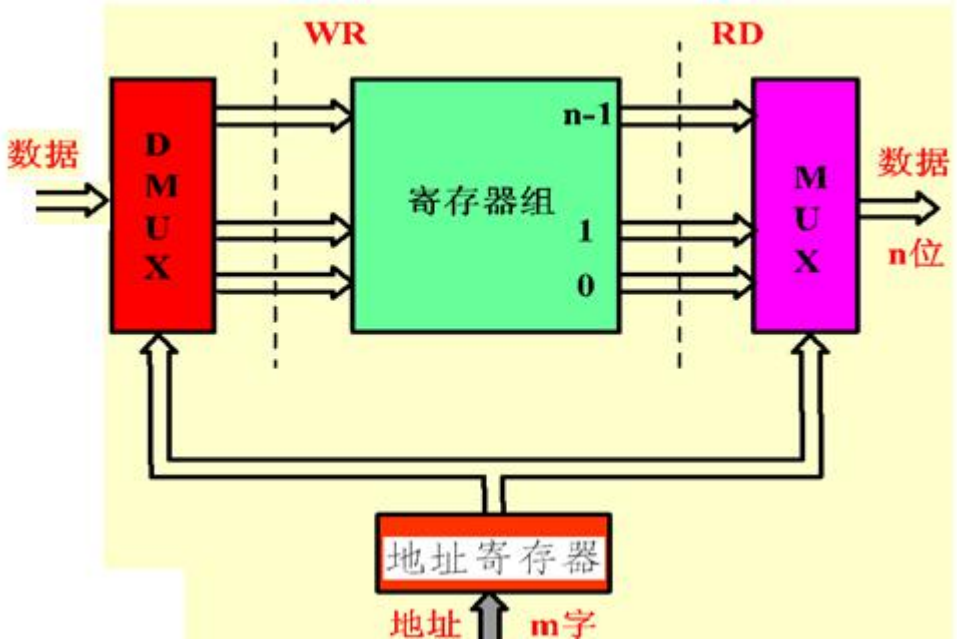
单端口寄存器堆

- 组成：寄存器组、地址寄存器、数据选择器、分配器
- 功能：按地址向寄存器写数或读数。
- 读／写控制:读／写命令分时进行
- 与随机存储器区别:

速度\寻址\容量\存储元

存储形式\读写电路

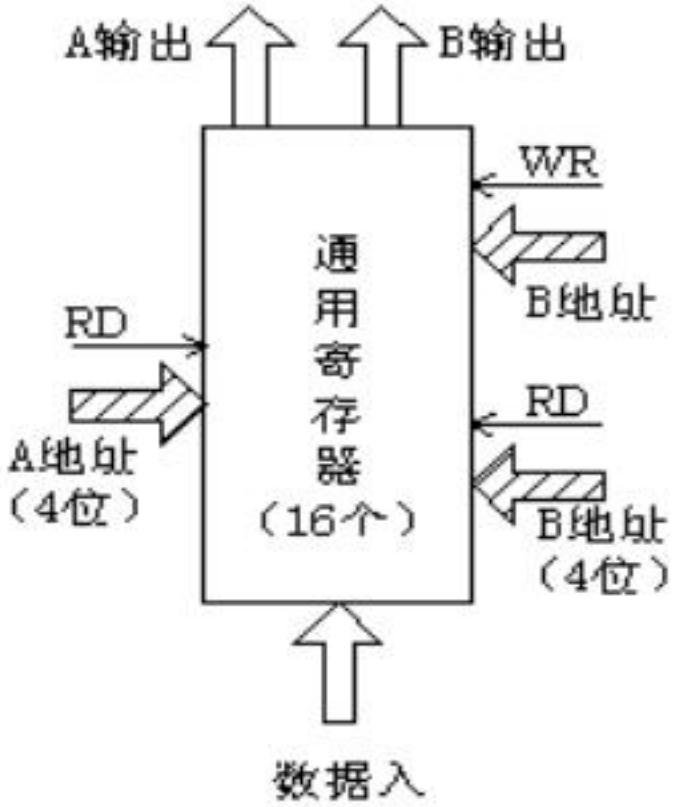
- 单端口输出



双端口输出寄存器堆

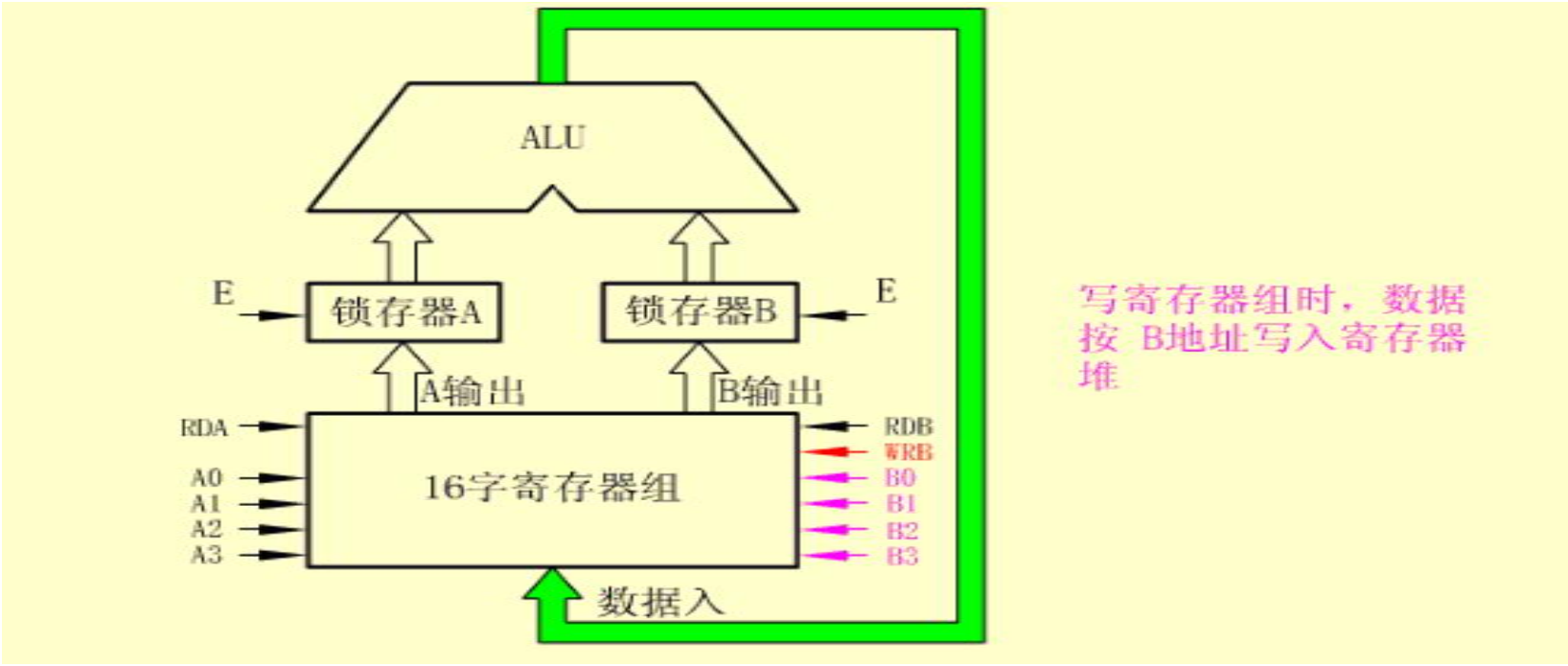
A输出
B输出
A地址：读数
B地址:写数/读数
数据输入

应用:构成运算器



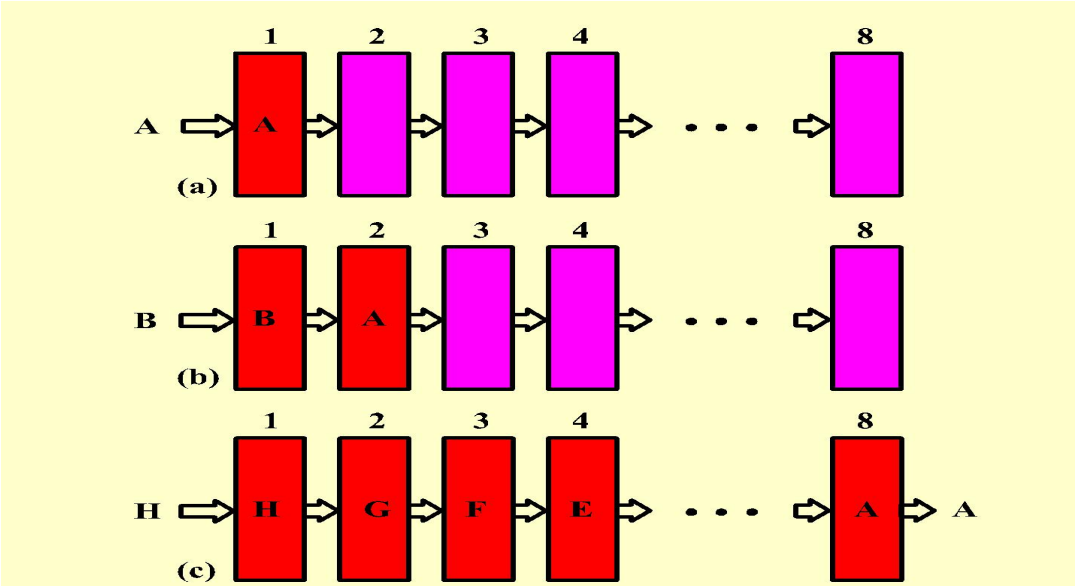
应用

- 与ALU（算术逻辑单元）构成运算器。
- 锁存器为ALU和寄存器堆间缓冲。
- RDA(B)有效，寄存器读出， WRB有效， 寄存器写入.
- A地址：寄存器从端口A读出
- B地址：寄存器端口B读出，或寄存器写入按B写入。



2　寄存器队列

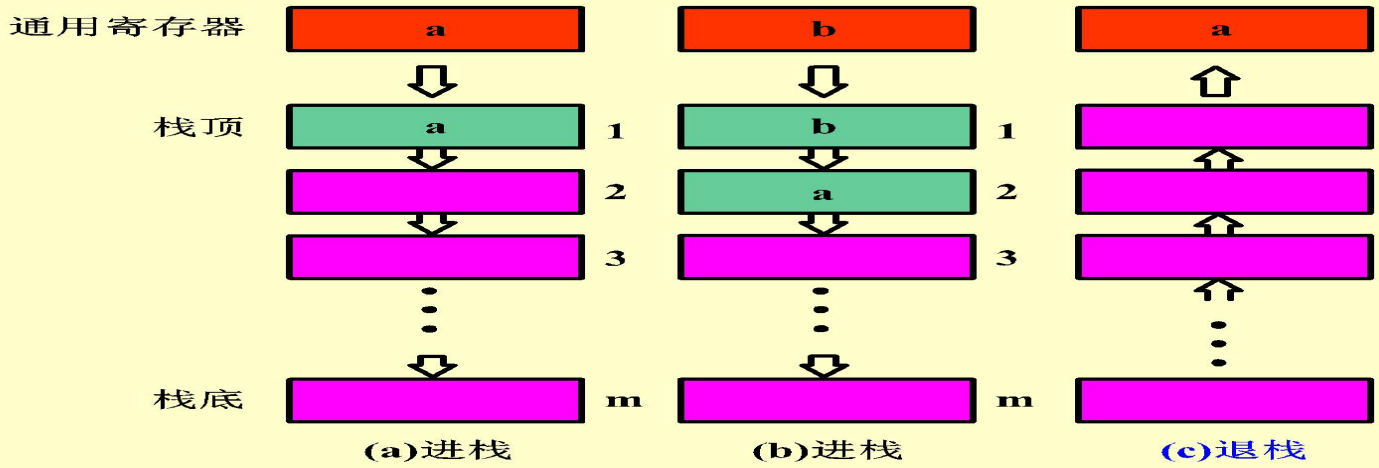
- 移位寄存器串连构成(小型存储部件)
- 先进先出（FIFO）
- 空队列：队列中无数据。
- 队列未满：数据(单向)移位
- 数据满:
- 应用：流水线



4.3 寄存器队列逻辑结构示意图

3 寄存器堆栈

- 若干寄存器以LIFO（后进先出）方式构建，与**寄存器队列**功能（先进先出）相反。
- 逻辑结构：通用寄存器、栈顶寄存器、中间寄存器、栈底寄存器
- 主要操作：**进栈**（ Push）、**出栈**（**弹出** Pop）
- 进栈：数据由通用寄存器压入栈顶寄存器；原栈顶寄存器送到下一寄存器，依次类推。
- 出栈：栈顶寄存器数据弹出到通用寄存器（最后进栈的数据）。
- 堆栈中无数据称为空栈，进栈数据超过容量，数据溢出（破坏）。



4.4 寄存器堆栈逻辑结构示意图

4.2 随机读写存储器(RAM)

- RAM结构、特点和工作原理
- 地址译码方法
- 存储元结构
- 存储容量的扩展*

随机读写存储器（RAM）

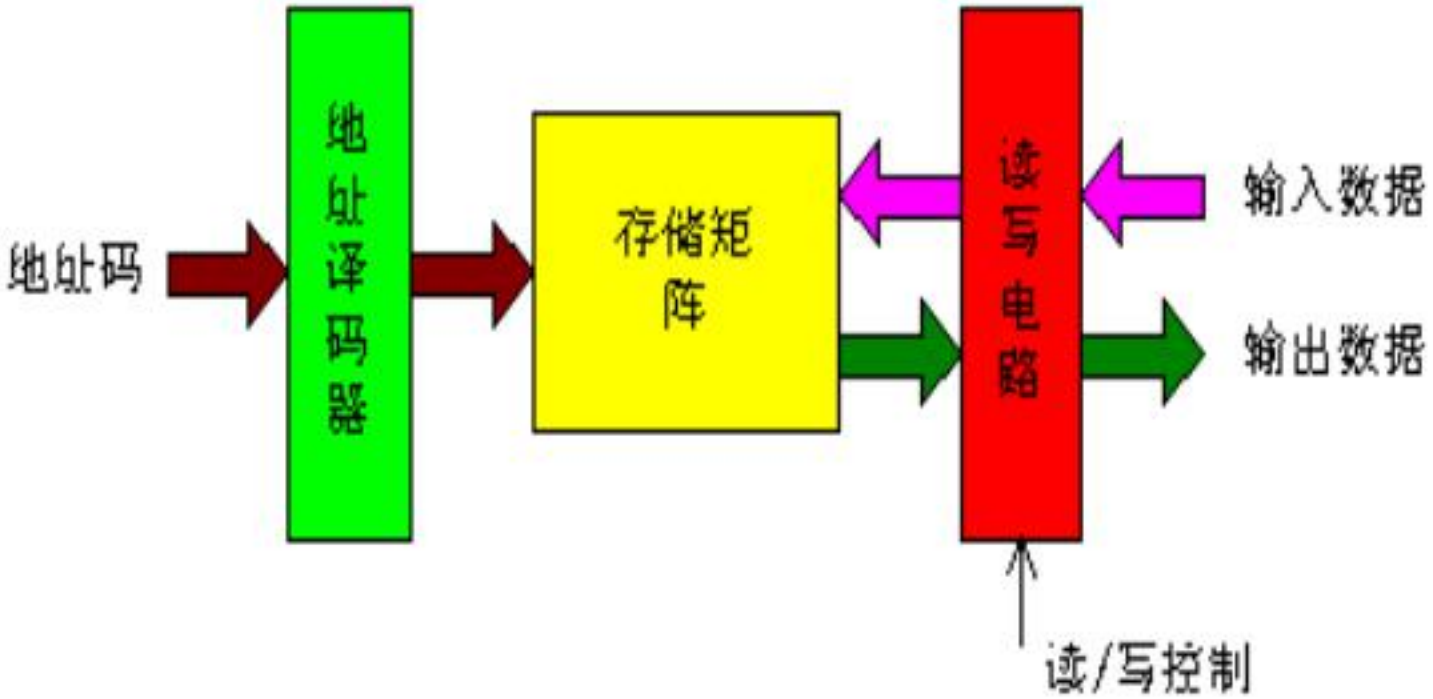
1　功能和特点

- 按地址方式寻址并进行读/写操作。随机存取，类似寄存器堆。
- 易失性 存储器：掉电后，所存数据全部丢失。
- 根据生产工艺，RAM分为双极型和MOS型两类。
- 双极型RAM:速度高，工艺复杂，功耗大，集成度低，用于高速工作场合。
- MOS型RAM:分为静态MOS和动态MOS，工艺简单，成本低，功耗小，集成度高，速度稍低。

随机读写存储器RAM

2　基本结构：

地址译码器，存储矩阵，读／写电路，输入／输出电路



RAM外部信号：地址线、数据线、控制线。

存储矩阵

- 存储矩阵是RAM核心，由排成阵列的存储单元（存储元）组成。
- 一个存储单元包含多个（W）存储元。一个存储元存放一个(位)二进制数据。
- 每个存储单元存放的信息为一个“字”或 （W位） 。
- 每个存储单元(字)有一个二进制编码(地址) 。通过地址译码器找到相应的存储单元,称为寻址。
- **存储矩阵中**所有存储单元**（或存储元）**数目为容量.
- n个地址线最多对应2^n地址，容量为2^n W（BIT）。

地址译码器

- 地址译码器完成寻址。通过寻址 读、写一个存储单元的过程称为（读、写）访问。
- 地址译码方式：单译码和双译码。

单译码：1个地址译码器。输出一条有效信号线（字选线）

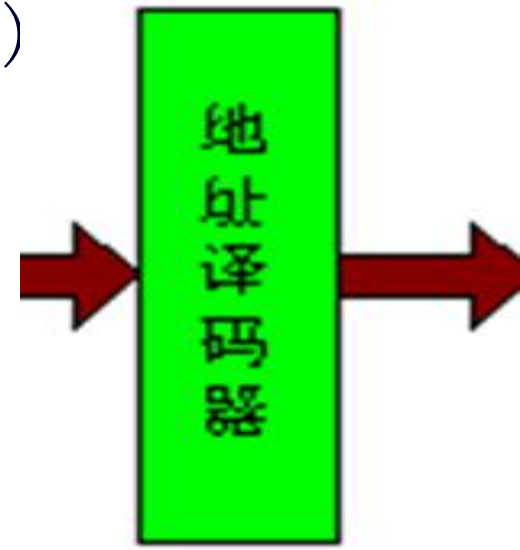
双译码：2个地址译码器。输出两条有效信号线（X，Y线）

- 访问（包括读出或写入）分三步进行：

1 把地址输入到地址译码器.

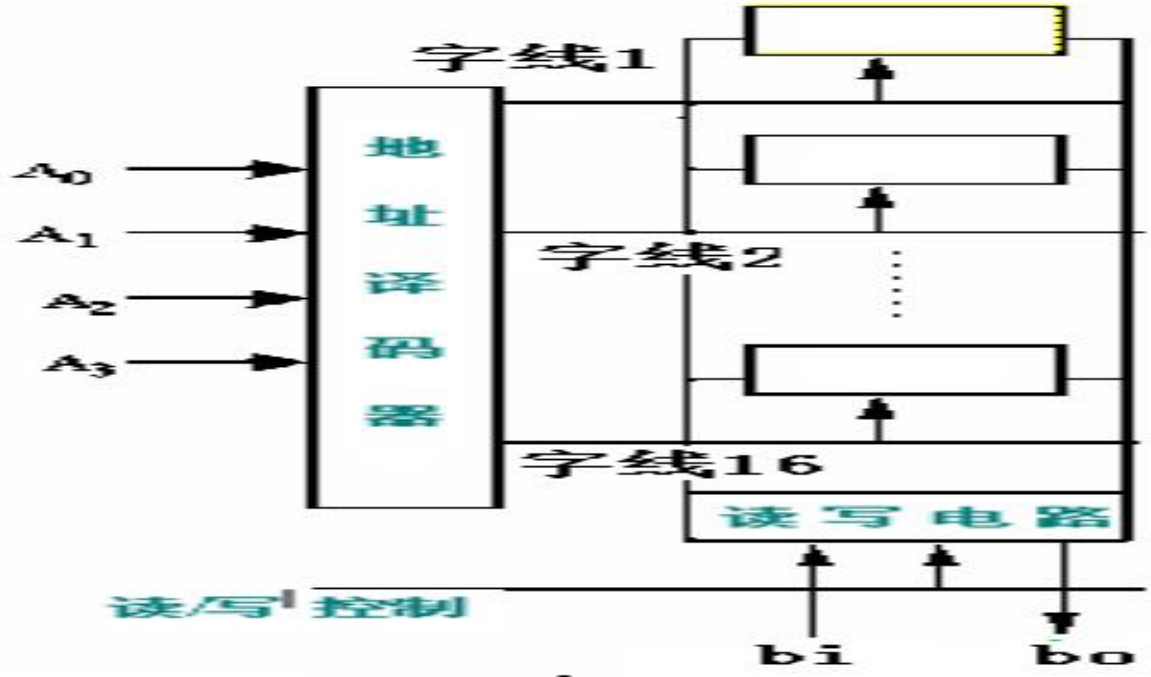
2 地址译码器译码输出一条(两条，双译码)有效信号线，选中一个存储单元.

3 按读\写命令，对选中的存储单元进行读\写。



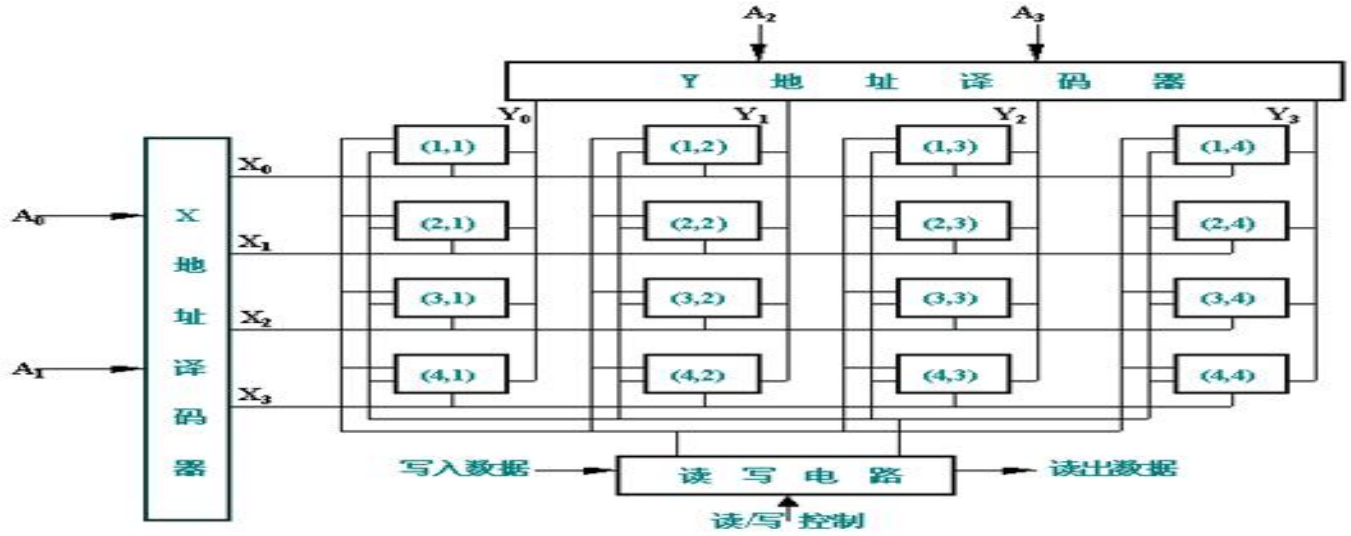
单译码结构

- 一个地址译码器
- 每个地址产生一条有效的字线，选通一个存储单元
- 16个存储单元，存放16个字。四位地址.
- n个地址码，输出2^n条字线。
- 单译码地址译码器地址难以很大。



双译码结构

- X和Y两个译码器。分别输出行选线X和列选线Y
- 行选线X和列选线Y进行交叉译码。X线和Y线均有效的存储单元(交叉点)被选中。
- 双译码结构的行线和列线总数少。适合大容量存储。如256字容量的存储，单译码需256条字选线,双译码仅需要（16＋16）32条字选线。
- n个地址码，只需2* $2^{n/2}$条字选线。



读写电路

- 读\写操作：　通过读写电路实现读操作和写操作。
- 读\写操作由读\写控制线决定。分时进行。
- 读\写控制线有两种：

  1　高\低电平控制(高电平为读，低电平为写)

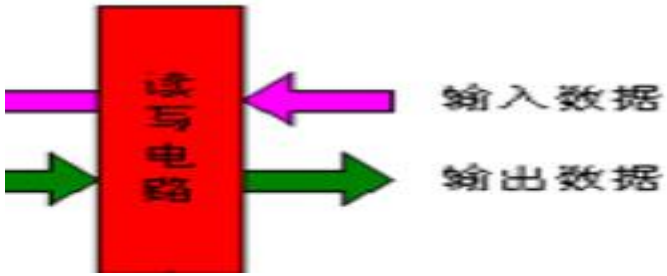
  2　读\写控制线分开，一根为读，另一根写。

输入输出电路

- 存储矩阵通过输入输出电路与外界交换数据。
- 读操作时输出电路工作，写操作时输入电路工作。
- 一条数据线（位线）传送一位信息。输入输出线条数与存储单元位数相同，每个存储单元数据并行传送。
- RAM输入线和输出线有分开或共享（分时用）两种。

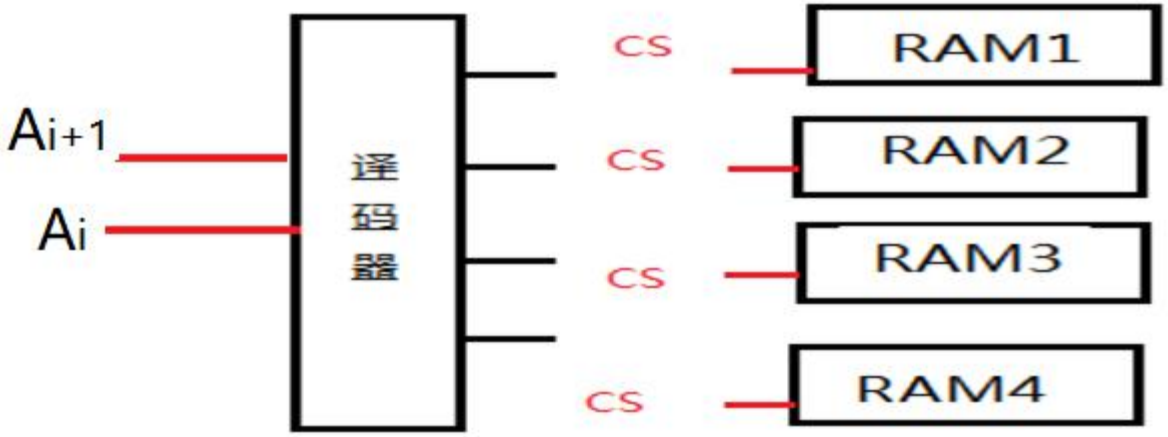
例如，在1024×1位的RAM中，每个地址只有1个存储元，只要1条输入输出线；

而在256×4位的RAM中，每个地址有4个存储元，需4条输入/输出线。



片选控制

- 每个 RAM有一个片选控制信号。片选控制线有效，该RAM工作（RAM与外界交换信息）。
- 片选控制线无效，RAM 与外界处于断开状态
- 存储器一般由多片RAM组合而成。通过片选控制线可以扩充RAM容量（或地址）。



RAM存储元

- 存储元相当于触发器。一个或多个存储元组成一个存储单元。
- 每个存储单元共享一条字选择线，当字选择线有效，该存储单元可以读或写.
- 按工作方式有静态RAM和动态RAM
- 静态RAM(SRAM)由静态MOS管电路组成,速度快。
- 动态RAM（DRAM）由动态MOS管电路组成。内部有刷新控制电路，操作比SRAM复杂。

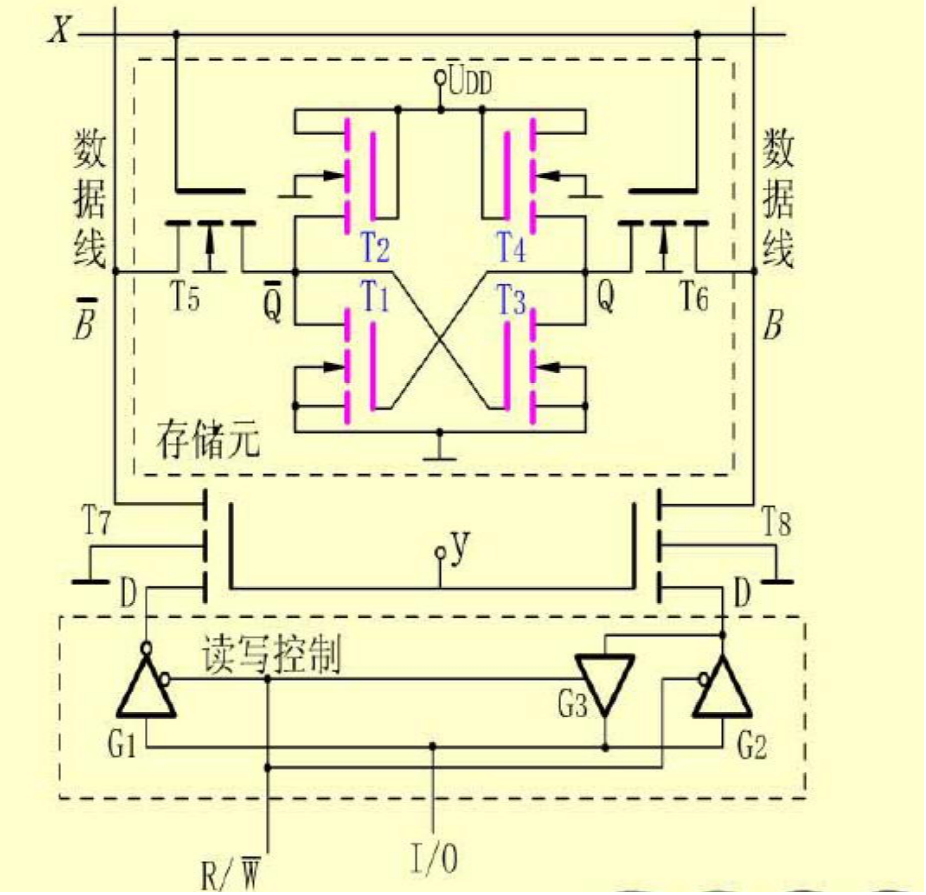
静态存储元SRAM

6管静态存储元

x控制 ：T_5T_6行存储元

y控制：T_7T_8列存储元（公共）

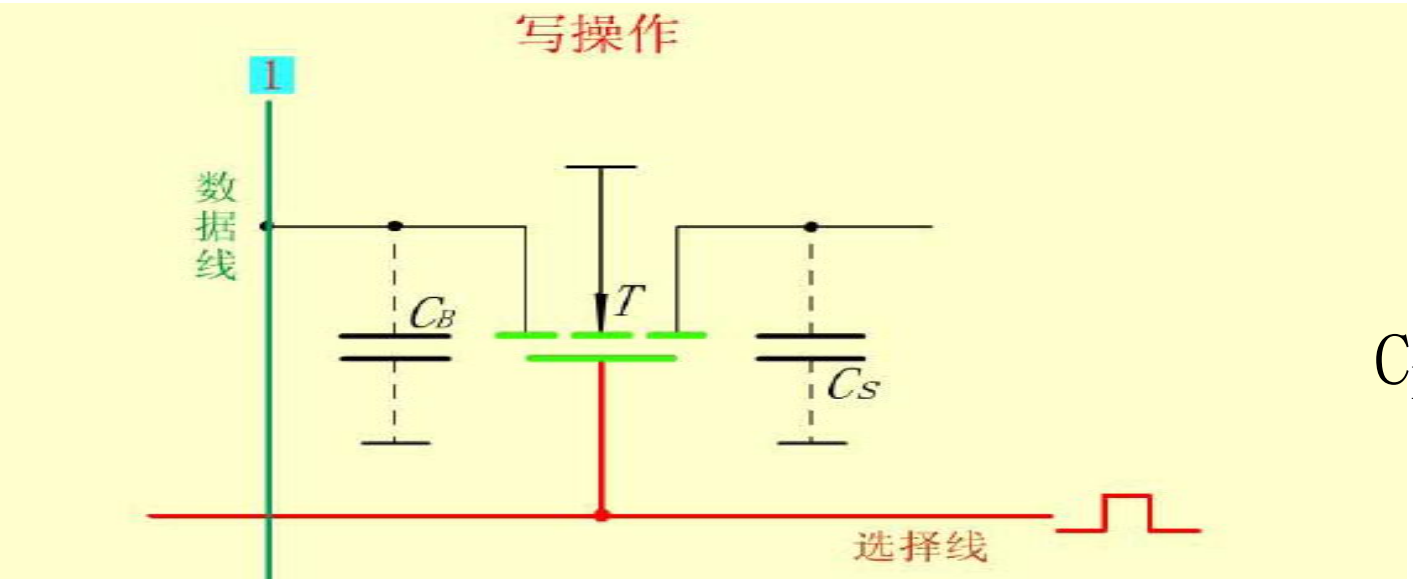
读写结束后，x,y信号消失



SRAM：MOS管数目多，不利于集成度提高。

单管动态存储元DRAM

- 动态RAM由动态MOS组成。数据存在电容C_S。
- 电容上存储信息不能长久保持(有漏电流），须定期给电容补充电荷，称为再生或刷新。
- 读/写时,选择线T为高电平. MOS管T导通.

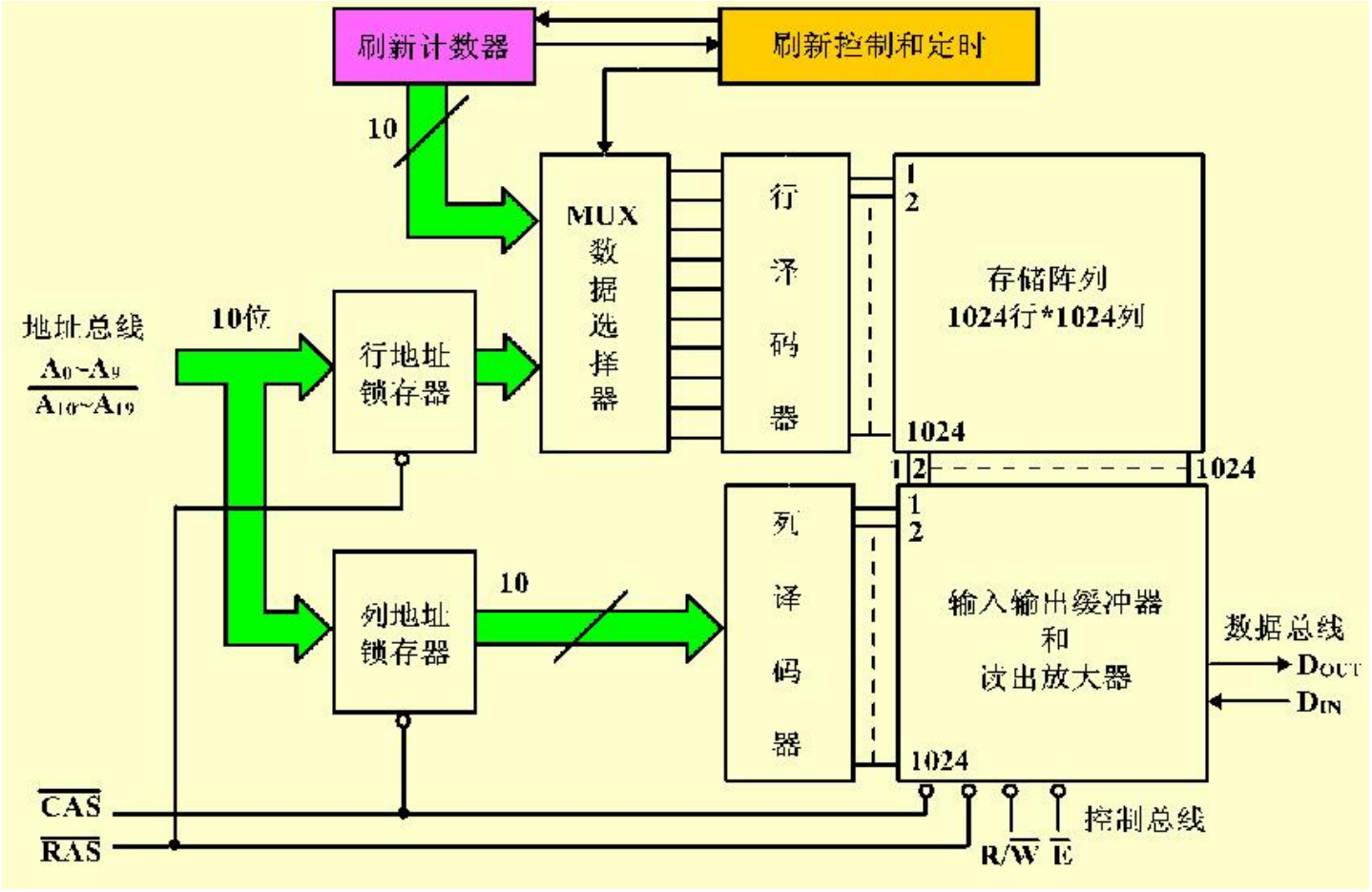


C_B：数据线电容

DRAM结构简单，元件少、功耗低，集成度提高。是RAM的主流产品。

DRAM基本结构

双地址译码
刷新计数器
定时刷新
2选1MUX
读写行地址
刷新行地址
行列地址线
(RAS)’行选通
(CAS)’列选通



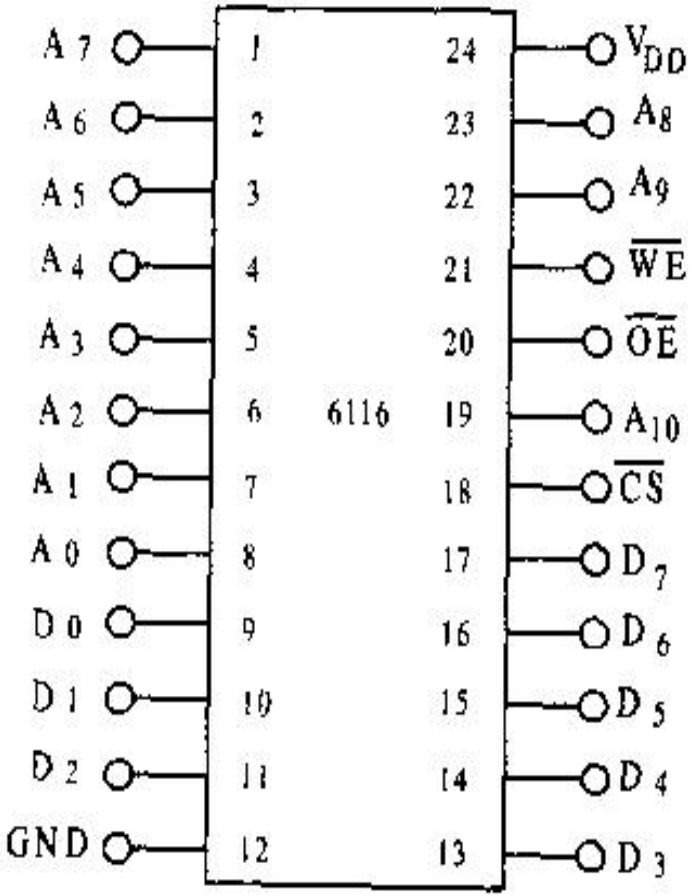
1 M× 1 位DRAM存储器框图

RAM的芯片简介(RAM6116)

- 图示为2K×8位静态RAM6116 的引脚排列图。
- A0--A10：地址码输入端
- D0--D7：数据输出端
- CS‘：选片端（0有效)
- OE‘：输出使能端（0有效)
- WE’：读写控制端
- GND 和VDD:

静态 RAM6116 工作方式与控制信号之间的关系

$\overline{CS}$	$\overline{OE}$	$\overline{WE}$	$A_0 \sim A_{10}$	$D_0 \sim D_7$	工作状态
1	×	×	×	高 阻 态	低功耗维持
0	0	1	稳定	输 出	读
0	×	0	稳定	输 入	写



静态 RAM 6116 引脚排列图

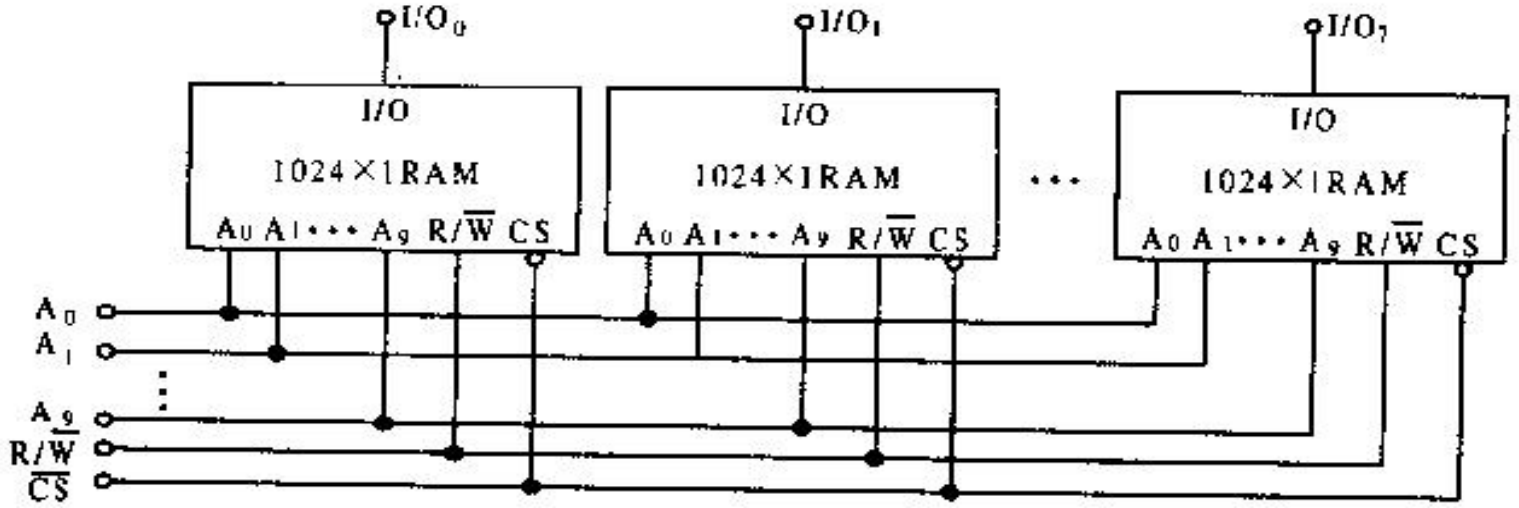
存储器容量的扩充*

一片RAM器件不能满足存储容量要求时，可将若干片RAM连在一起，以扩展存储容量。扩展的方法有三种：

- 位扩展： RAM位数增加.地址数目不变
- 字扩展： RAM地址数目增加.位数数不变
- 字位均扩展 ： RAM地址数目和位数均增加

位扩展

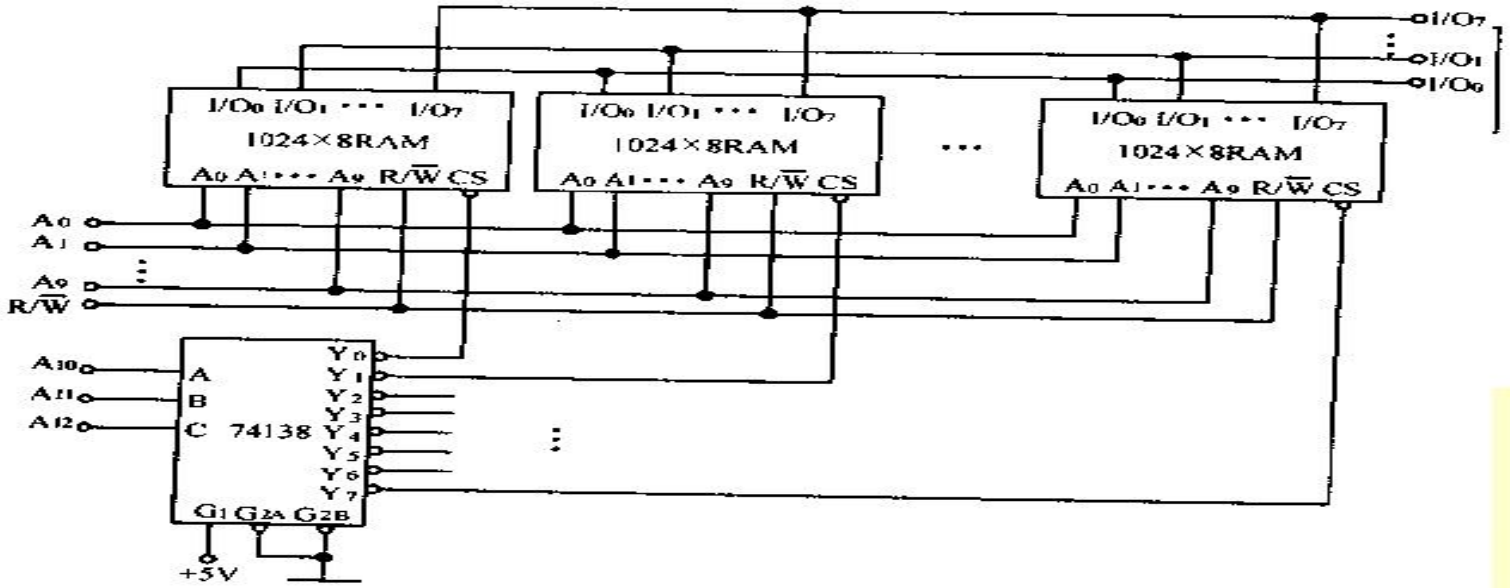
- 单片RAM位数不够时，可选择位扩展连接方式。使每个字的位数增加.但地址数不变
- 图示是8片1024×1位的RAM构成的1024×8位RAM系统。
- 位扩展：地址线、读/写和片选线均并联
- 每个输入/输出通路（I/O）作为整个存储单元的一位使用



1K×1位RAM扩展成1K×8位RAM

字扩展

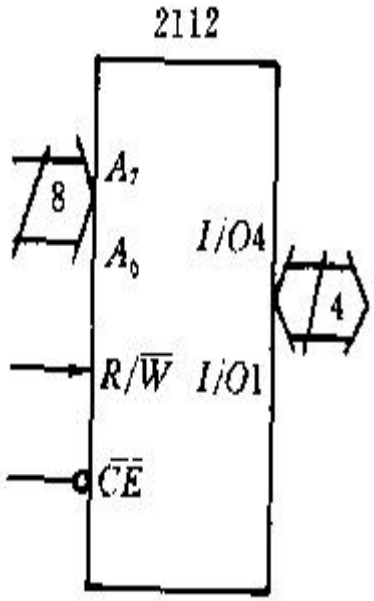
- RAM字数不够用，可选择字扩展连接。地址数增加。
- 多片RAM接成一个字数更多的存储,通常使高位地址线增加.
- 图示是用8片1K×8位RAM构成8K ×8位的RAM. 图中输入／输出线，读／写线和低位地址线并联，增加高位地址
- 高位地址经译码器输出端分别控制RAM的片选端.
- 如需要，采用位与字同时扩展的方法扩大RAM的容量。



1K×8 位 RAM 扩展成 8K × 8 位 RAM

示　例

[例] 静态RAM 2112(256×4)芯片的逻辑符号如图所示。



1．用多少片RAM 112芯片能组成1K×8的RAM?

2．寻址1K×8内存单元需用多少根地址线?

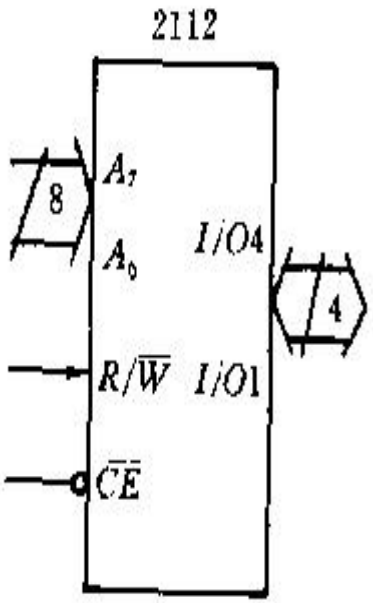
3．画出1K×8RAM的接线图。

示　例

解　1．芯片容量为256×4。组成1K×8的容量，需同时进行位和字扩展。

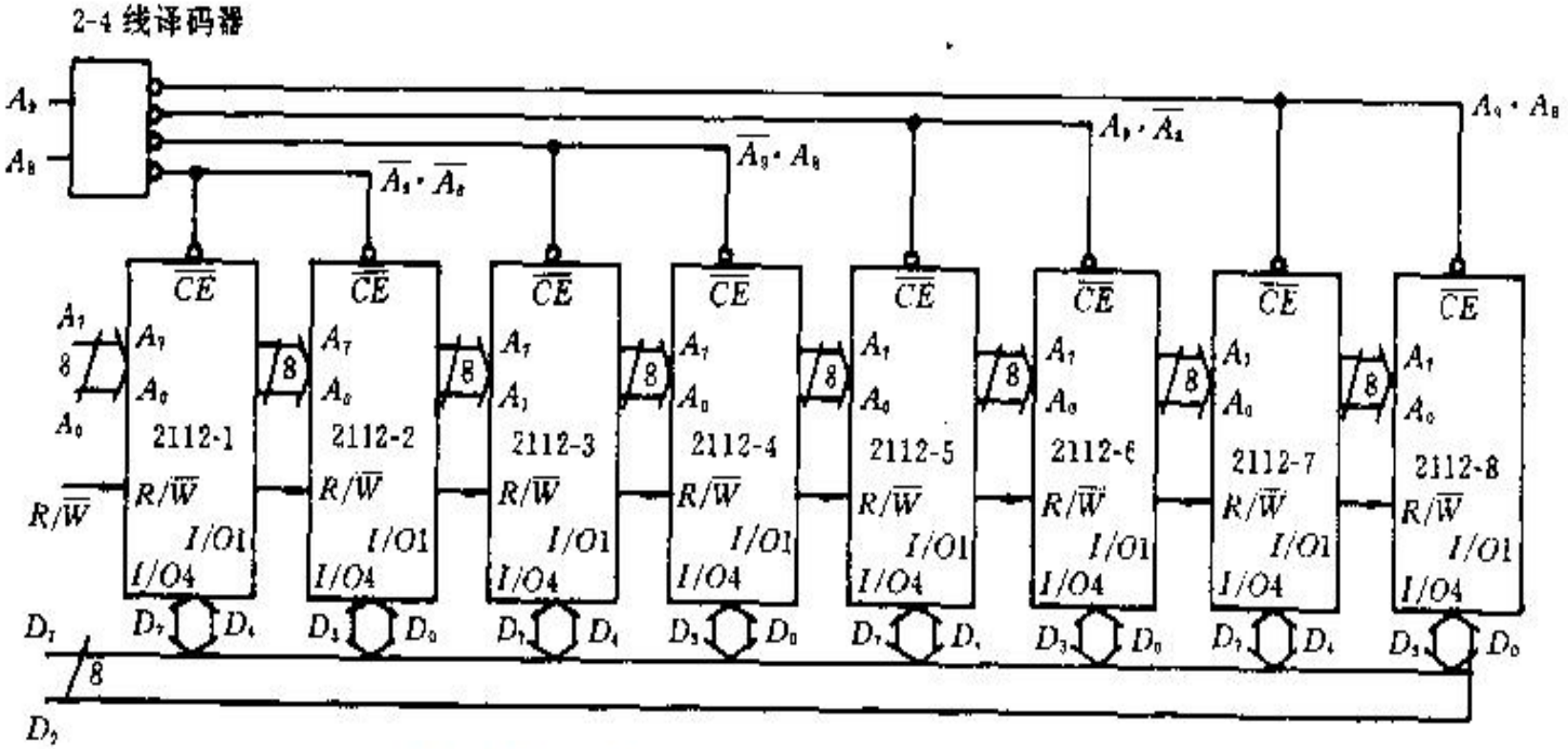
- 位扩展：两片2112可扩展为256×8。
- 字扩展：位扩展2倍基础上，将256条字线扩展成1024条字线，字扩展4倍，共需八片2112

2、 256条字线需八条地址线，寻址1K地址线为十条。增加两条。



示　例

3．八片2112芯片扩展为1K×8的RAM,接线图为



寻址范围

$A_9=A_8=0$ ⇨ 2112-1,2 为 000H～0FFH

$A_9=0$　$A_8=1$ ⇨ 2112-3,4 为 100H～1FFH

$A_9=1$　$A_8=0$ ⇨ 2112-5,6 为 200H～2FFH

$A_9=A_8=1$ ⇨ 2112-7,8 为 300H～3FFH

4.3 只读存储器

1. ROM的特点和性能指标
2. ROM的分类
3. ROM的结构与工作原理
4. ROM的应用

ROM的特点和性能指标

- ROM 只能读出

  工作时将地址加到ROM地址输入端，数据输出端得到一个事先存入的数据。
- 优点：不易失性. 断电后存储数据不会丢失. 可长久保存. 常存放固定数据、程序和函数表。
- 缺点：不能重写或改写。
- 最初存入数据的过程，称为对ROM进行编程.

ROM的性能指标

存储容量和存取时间

- **存储容量**

  存储器存放信息的能力，存储容量越大，所能存储的信息越多，功能越强。

- **存取时间**

  存取时间指一个读(或写)周期。读(或写)周期越短，存取时间越短，存储器工作速度越高。

ROM的分类

根据编程方式，分为固定ROM、一次编程ROM、多次改写编程ROM和闪速存储器四类。

- 固定ROM

掩模式只读存储器，数据在芯片制造过程“固化”在ROM中，使用时读出，不能改写。

通常存放固定数据、程序和函数表等。可向厂家定做。固定ROM可靠性好，集成度高,适宜大批量生产。

- 一次编程只读存储器（PROM）

出厂时所有存储元全0或全1，用户可自行改1或0。用熔丝烧断或PN结击穿法编程，烧断或PN击穿不能恢复，一次性编程.编程完毕内容永久保存.已少使用。

ROM的分类

- 可多次编程的只读存储器

  光擦可编程只读存储器EPROM

  电擦可编程只读存储器E2PROM

  电改写只读存储器EAROM

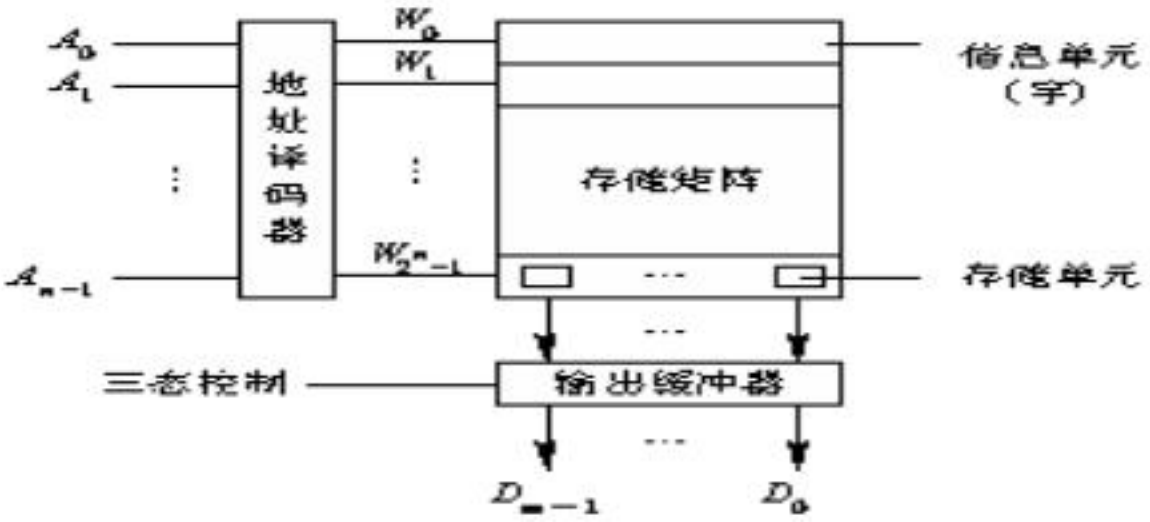
  可用紫外光照射或加电法擦除已写入的数据，用电方法可重新写入新的数据。可多次改写内容。

- 闪速存储器（FLASH)

  英特尔上世纪90年代发明的一种高密度、非易失性的可读/写存储器,既有EEPROM的特点，又有RAM的特点，无须编程器.

ROM的结构与工作原理

组成：由存储矩阵、地址译码器、输出缓冲电路组成。



存储矩阵：存储单元组成。存储单元中存储元个数为字长

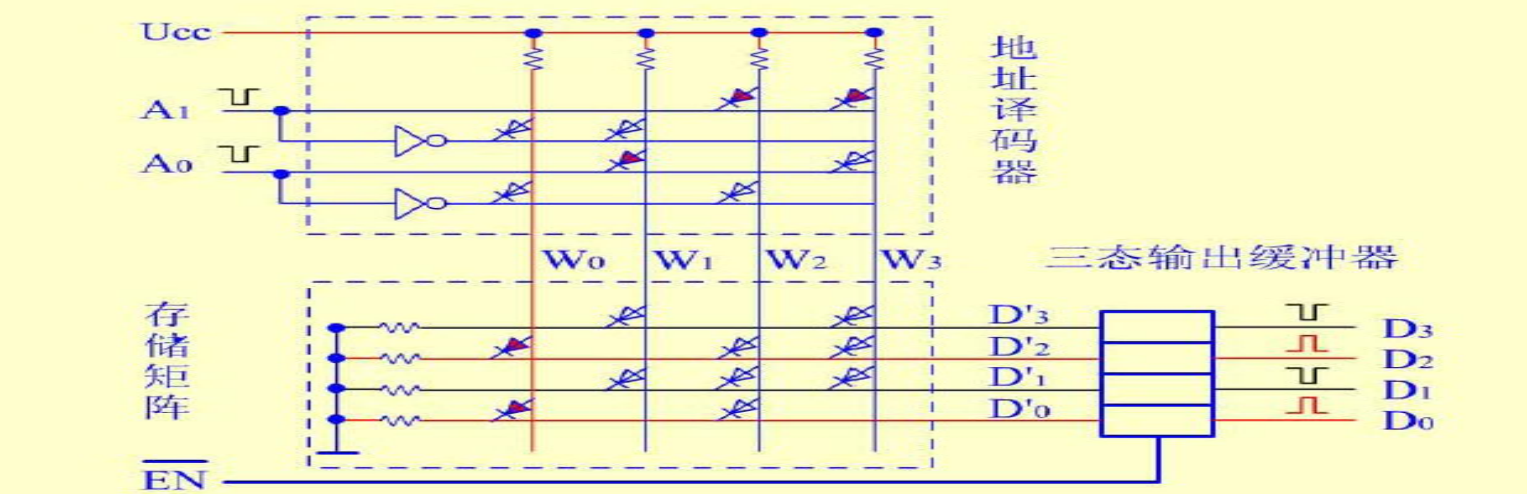
地址译码器：地址码到存储单元的映射. 每个地址使一条字线有效，该单元信息将送至输出缓冲器读出

输出缓冲器： 读出电路，数据输出线D_{01}—D_{m-1} 又称位线

一次编程只读存储器的结构

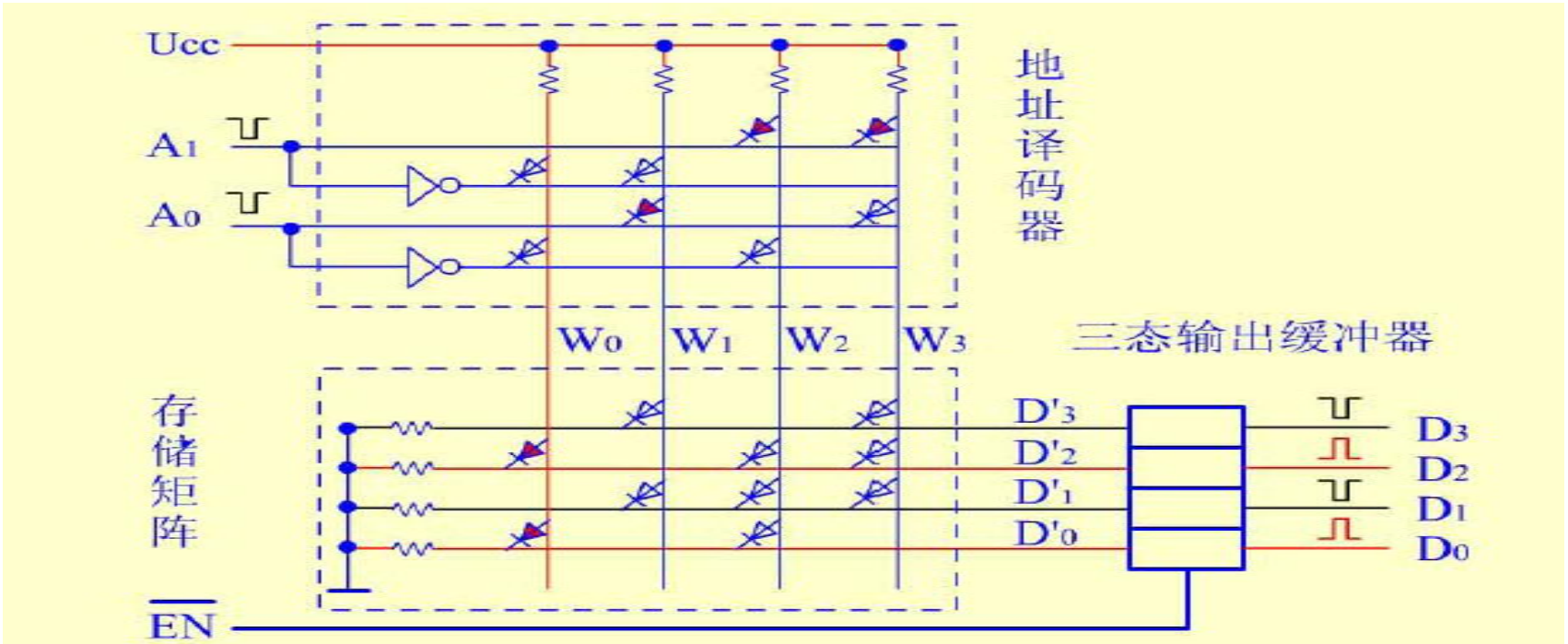
- 地址译码器和存储矩阵分别由与门和或门阵列组成.
- 与门阵列和或门阵列可用二极管构成(如图)。
- 地址线：A1、A0;
- 字线(W0--W3);与门阵列的输出.
- 位线(数据线):D3--D0 或门阵列的输出.
- 存储容量:存储单元数乘位数（字数乘位数）。

图例：存储容量为4x4(位)



二极管ROM模型

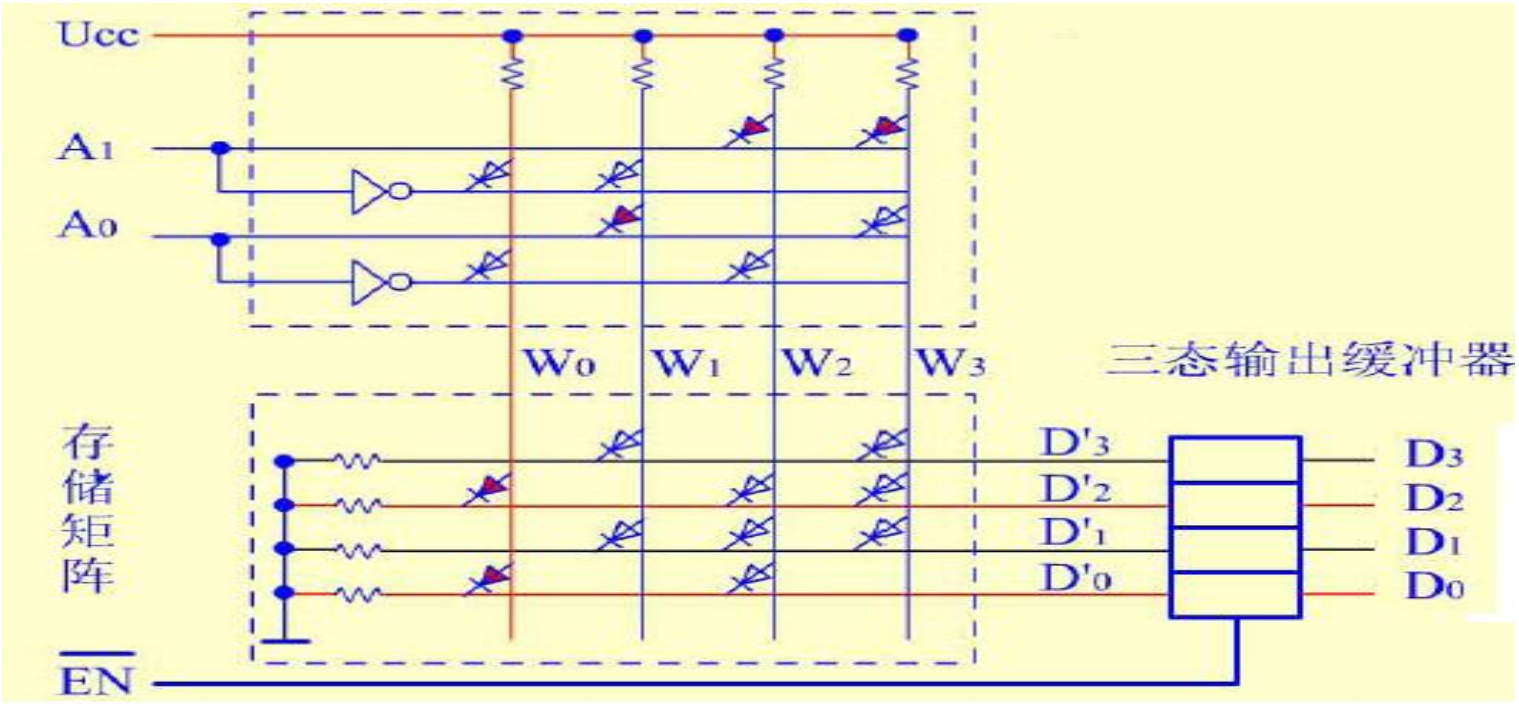
- 地址译码--与阵列：每个地址产生一条（高电平）有效字线 (W0---W3),使或阵列一个单元输出数据D3---D0
- 存储矩阵--或阵列:字线W与位线的交叉点是存储元。交叉点接二极管相当于存储1，不接相当于存储0
- 如 A1A0=11　　W_3线有效,若 EN'=0，则D3-D0=1110



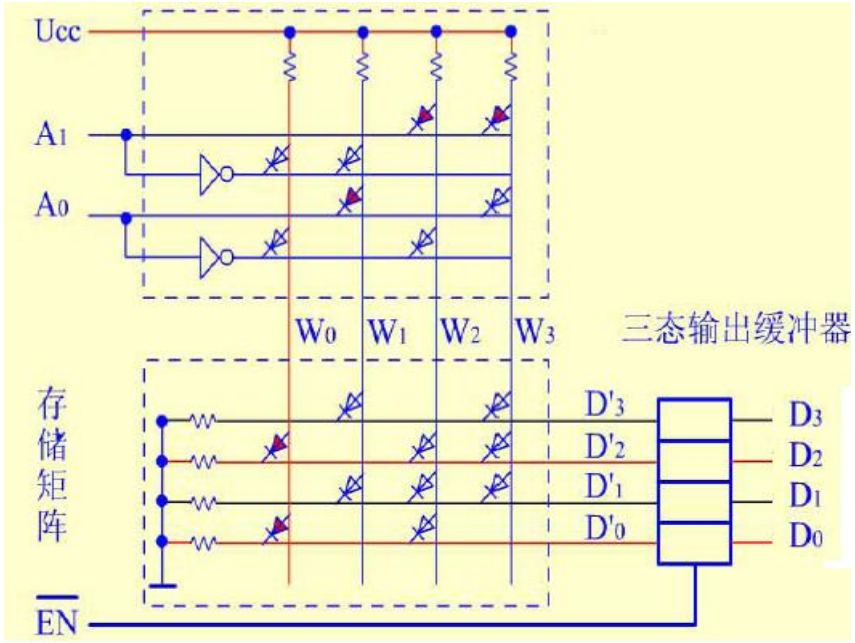
二极管ROM模型

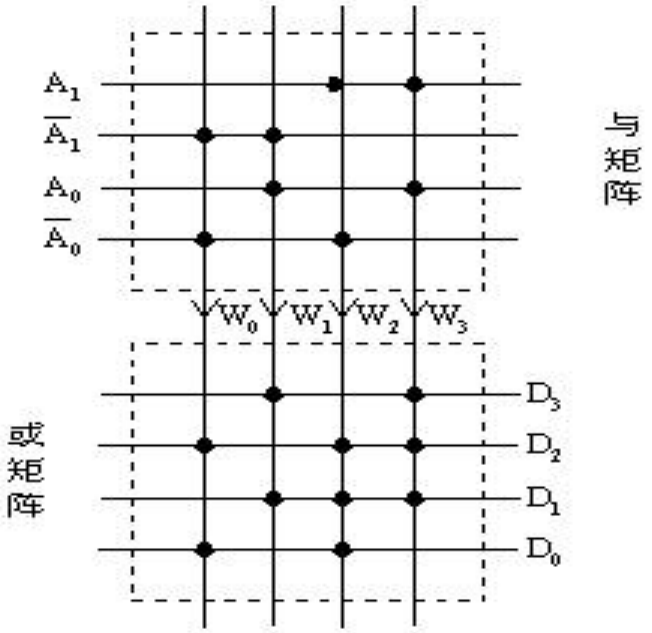
【例】A1A0=00时，W0线高电平（左上两二极管截止）或门阵列中D2'、D0'为1（左下两个二极管导通），若EN'=0，ROM输出为$D_3D_2D_1D_0$=0101。

分析：A1A0=01时，ROM输出$D_3D_2D_1D_0$=？



ROM点阵结构表示法





ROM的结构可用阵列图来表示：

- 地址线、字线，（数据）位线。
- 字线和位线相互垂直，与阵列在上（左），或阵列在下（右）.
- 与阵列交叉点（黑点）表示有一个二极管.无黑点表示没有。
- 存储矩阵交叉点(黑点)表示存储元存1， 无黑点表示存0。

ROM工作原理(编程)

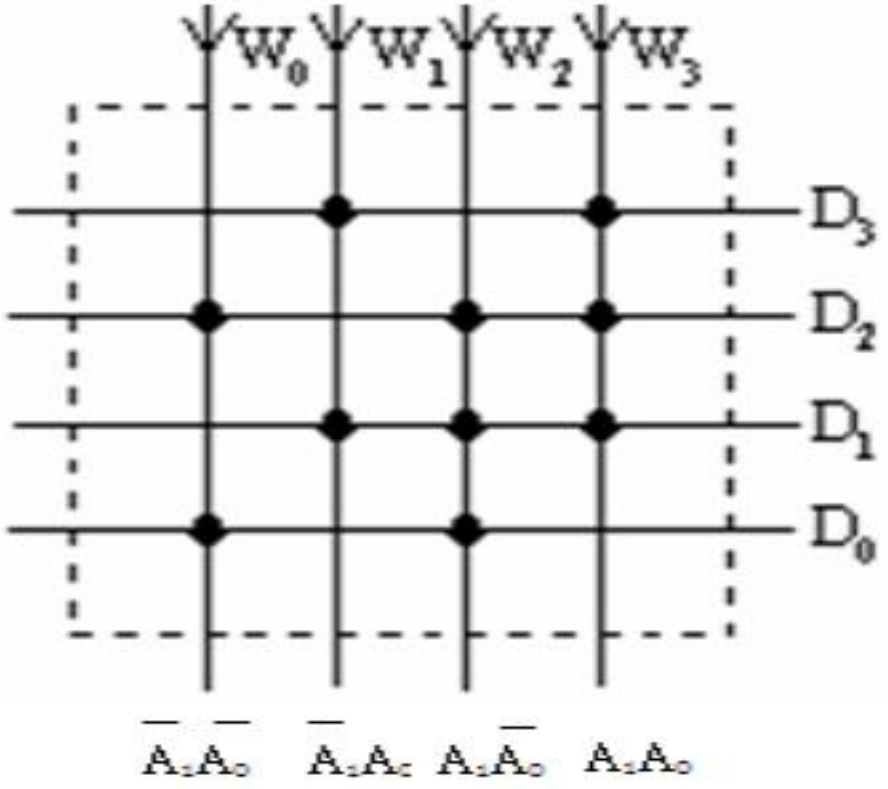
把ROM看作**组合电路**，地址码A1A0是输入变量，数据码D3---D0是输出变量，则输出：

$$D_3=A_1A_0+A_1A_0$$

$$D_2=A_1A_0+A_1A_0+A_1A_0$$

$$D_1=A_1A_0+A_1A_0+A_1A_0$$

$$D_0=A_1A_0+A_1A_0$$

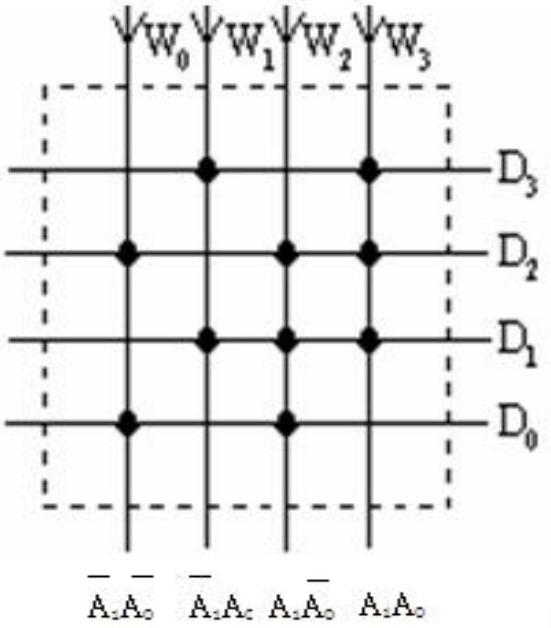


ROM的工作原理

把ROM看作存储单元

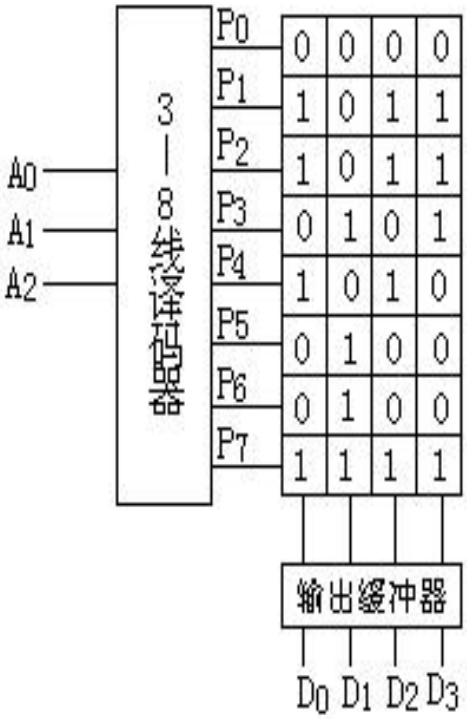
逻辑函数真值表

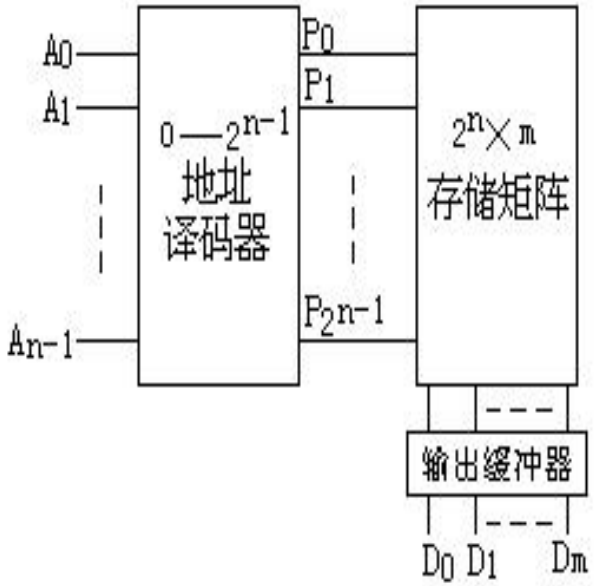
地址		数据			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	1	0	1
0	1	1	0	1	0
1	0	0	1	1	1
1	1	1	1	1	0



- 与阵列不编程。或阵列（存储矩阵）编程。
- 或阵列不同，每个单元中存储的数据不同.

示例1： 一个8（字线）×4（数据）的存储器阵列图





若n条地址线，可产生字线为2^n条，寻址2^n个单元。若输出是m位，存储器的总容量是$2^n \times m$。

如当地址码$A_2A_1A_0$＝000时，使字线P_0＝1，数据0000输出。

如当地址码A2A1A0＝011时，数据输出是什么?。

多次改写编程的只读存储器

- **除** 一次编程只读存储器（PROM）外，多次改写编程的只读存储器包括:
- 光（紫外线）擦可编程只读存储器
- 电擦可编程只读存储器
- 电改写只读存储器

图例：下图是紫外线擦除、电可编程的EPROM2716器件逻辑框图和引脚图。

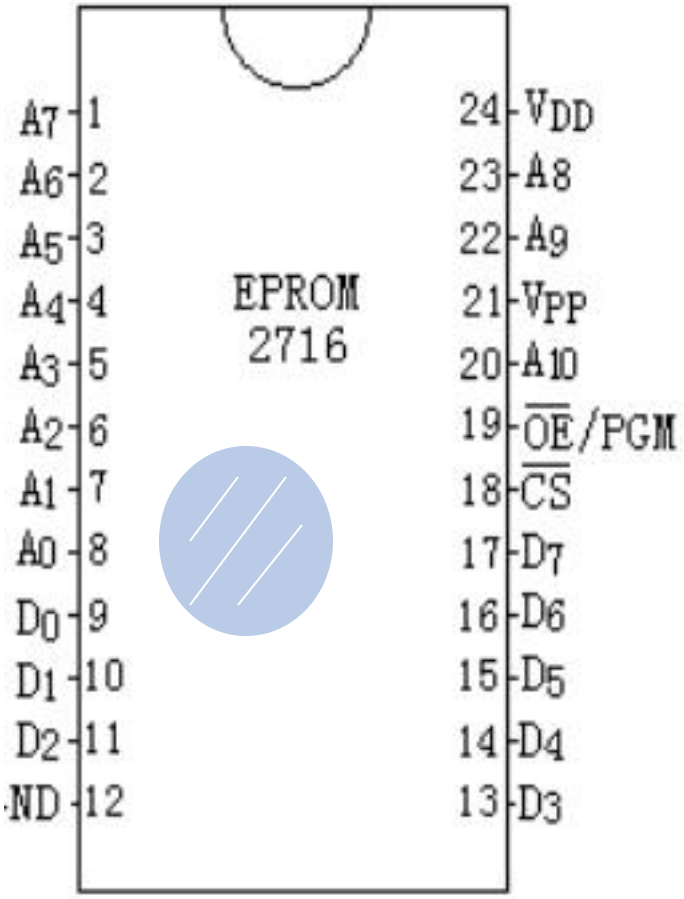
地址线A_0～A_{10}，字线2048条.

数据输出/输入线D_7～D_0

容量：$2^{11}\times 8$位。

CS为片选控制信号。

OE/PGM为读出/写入控制端：低电平时输出有效，高电平时进行编程（写入数据）



及引脚图

ROM的应用

组合逻辑设计

- 代码转换器
- 组合逻辑函数

计算机系统

- 初始引导和加载程序的固化；
- 微程序控制器的设计；
- 函数运算表
- 字符图形发生器
- 控制系统中用户程序的固化。

ROM的应用（代码转换器）

[例] 用ROM实现4位二进制码到格雷码的转换

【解】 1）输入：二进制码B0---B3，输出：格雷码G3--G0，16个字,每个字四位，故选容量为 $2^4\times4$的ROM

2）真值表

字	二进制码				格雷码			
	B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
W_0	0	0	0	0	0	0	0	0
W_1	0	0	0	1	0	0	0	1
W_2	0	0	1	0	0	0	1	1
W_3	0	0	1	1	0	0	1	0
W_4	0	1	0	0	0	1	1	0
W_5	0	1	0	1	0	1	1	1
W_6	0	1	1	0	0	1	0	1
W_7	0	1	1	1	0	1	0	0
W_8	1	0	0	0	1	1	0	0
W_9	1	0	0	1	1	1	0	1
W_{10}	1	0	1	0	1	1	1	1
W_{11}	1	0	1	1	1	1	1	0
W_{12}	1	1	0	0	1	0	1	0
W_{13}	1	1	0	1	1	0	1	1
W_{14}	1	1	1	0	1	0	0	1
W_{15}	1	1	1	1	1	0	0	0

3）位线编程和字线编程

ROM的应用

方法一：位线编程---输出函数为若干最小项之和

$$G_3 = \sum m(8, 9, 10, 11, 12, 13, 14, 15)$$

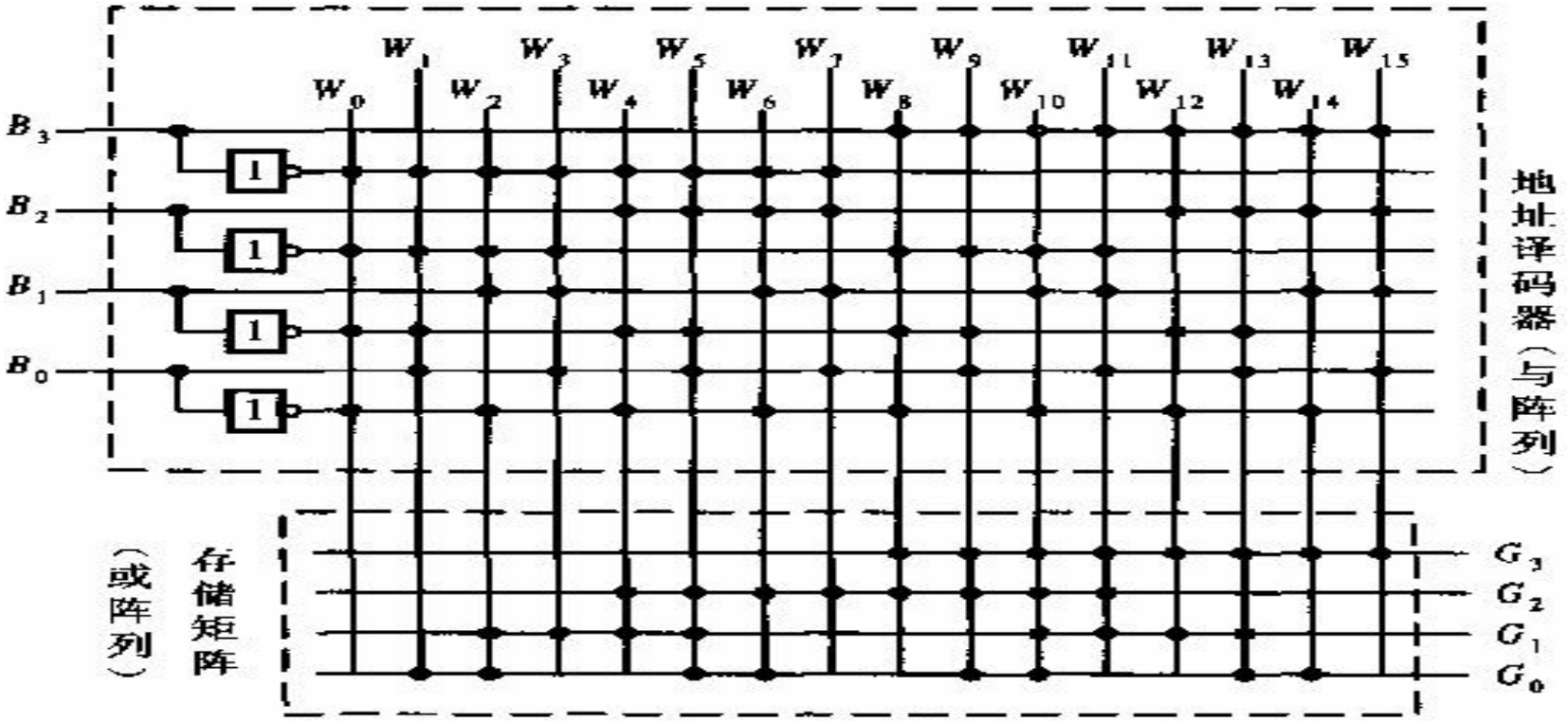
$$G_2 = \sum m(4, 5, 6, 7, 8, 9, 10, 11)$$

$$G_1 = \sum m(2, 3, 4, 5, 10, 11, 12, 13)$$

$$G_0 = \sum m(1, 2, 5, 6, 9, 10, 13, 14)$$

ROM的应用

按位线编程——各行为某些最小项之和



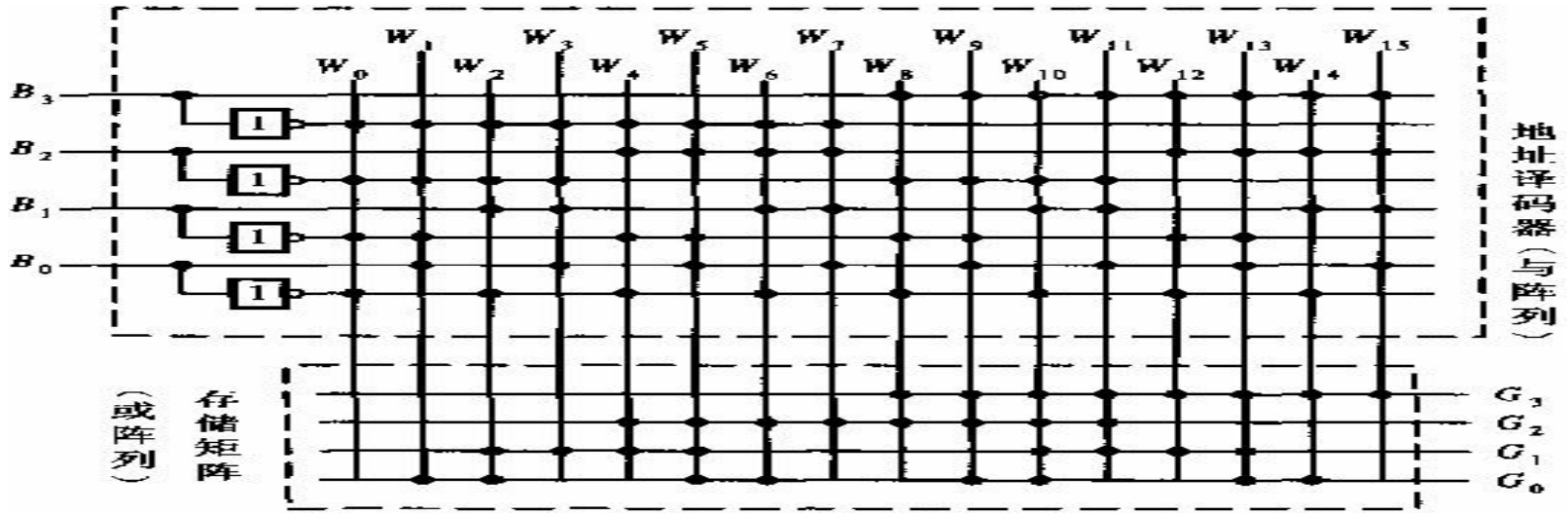
$$G_3 = \sum m(8, 9, 10, 11, 12, 13, 14, 15) \qquad G_1 = \sum m(2, 3, 4, 5, 10, 11, 12, 13)$$

$$G_2 = \sum m(4, 5, 6, 7, 8, 9, 10, 11) \qquad G_0 = \sum m(1, 2, 5, 6, 9, 10, 13, 14)$$

ROM的应用（代码转换器）

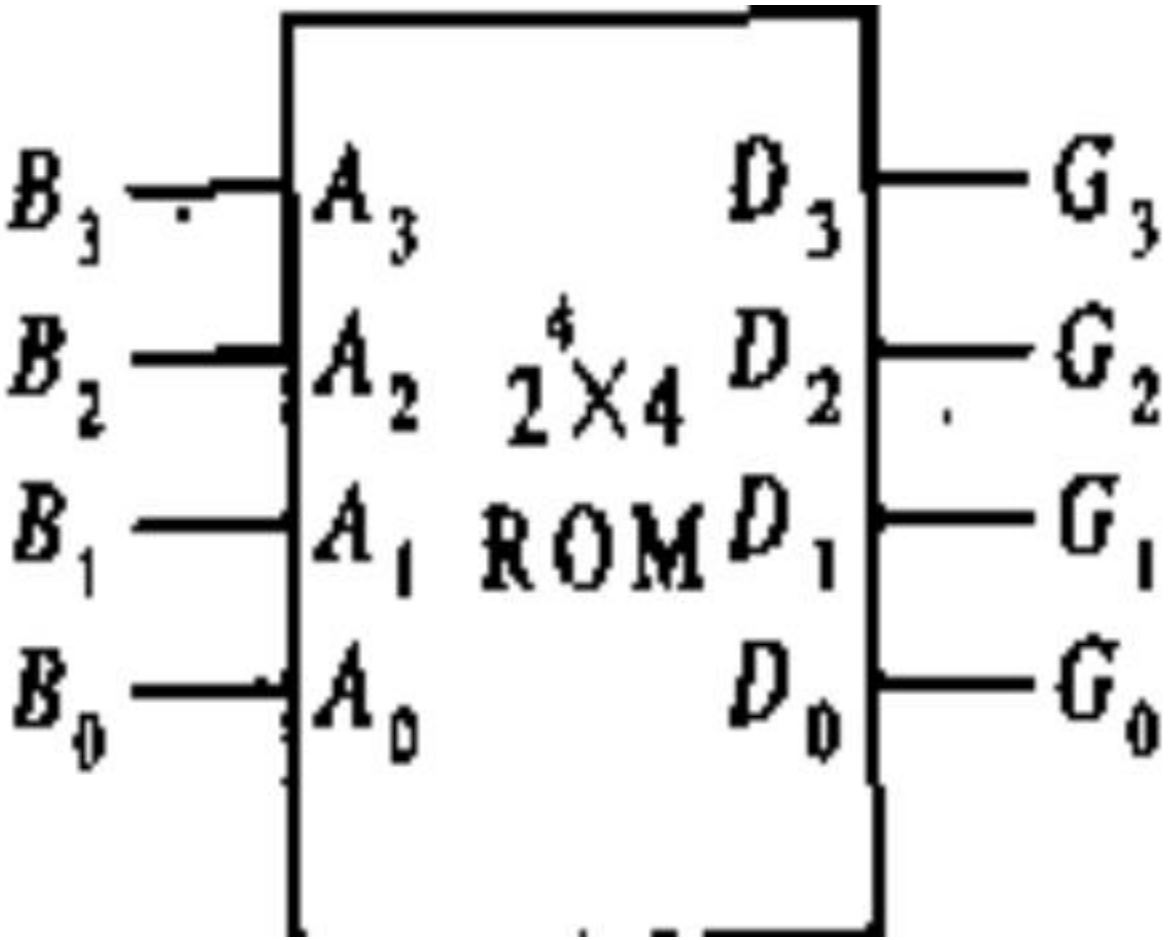
方法二：按字线编程—各列是存储单元中数据

字	二进制码				格雷码			
	B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
W_0	0	0	0	0	0	0	0	0
W_1	0	0	0	1	0	0	0	1
W_2	0	0	1	0	0	0	1	1
W_3	0	0	1	1	0	0	1	0
W_4	0	1	0	0	0	1	1	0
W_5	0	1	0	1	0	1	1	1
W_6	0	1	1	0	0	1	0	1
W_7	0	1	1	1	0	1	0	0
W_8	1	0	0	0	1	1	0	0
W_9	1	0	0	1	1	1	0	1
W_{10}	1	0	1	0	1	1	1	1
W_{11}	1	0	1	1	1	1	1	0
W_{12}	1	1	0	0	1	0	1	0
W_{13}	1	1	0	1	1	0	1	1
W_{14}	1	1	1	0	1	0	0	1
W_{15}	1	1	1	1	1	0	0	0



代码转换器

逻辑符号图



ROM的应用（组合逻辑函数）

ROM实现逻辑函数的步骤：

- 列出函数最小项表达式(或真值表)。
- 选择合适的ROM。
- 画出函数的阵列图。

[示例]　用ROM实现函数

$$\begin{cases} Y_1 = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + AB\bar{C}D \\ Y_2 = AB + AC + BC \\ Y_3 = AB\bar{D} + BCD + \bar{B}\bar{C}D \\ Y_4 = \bar{A}\bar{C} + B\bar{C} + \bar{B}D + A\bar{B}C \end{cases}$$

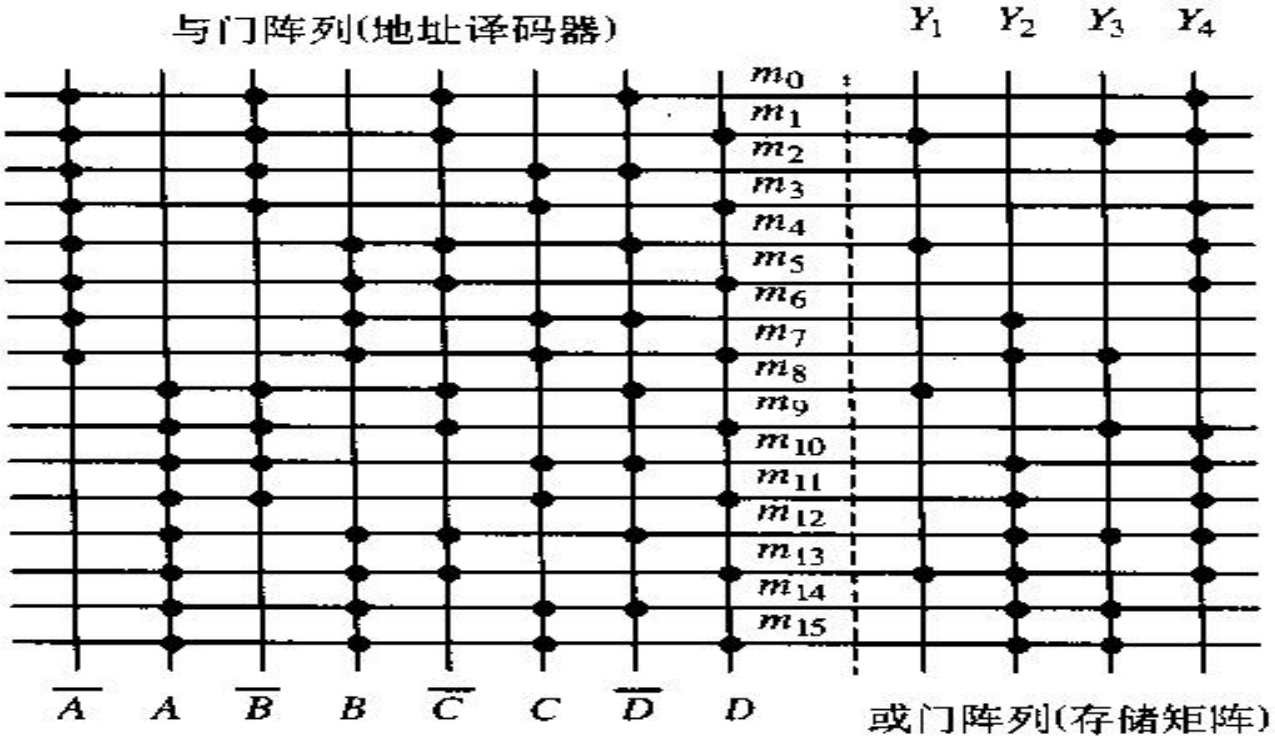
解　方法一：按位线编程:列出函数的最小项表达式

1)　按A、B、C、D 排列变量，将Y1，Y2,Y3,Y4　视作A、B、C、D 的逻辑函数，写出各函数的最小项表达式

$$\begin{cases} Y_1 = \sum m(1,4,8,13) \\ Y_2 = \sum m(6,7,10,11,12,13,14,15) \\ Y_3 = \sum m(1,7,9,12,14,15) \\ Y_4 = \sum m(0,1,3,4,5,9,10,11,12,13) \end{cases}$$

2)　ABCD作为ROM地址输入，4个函数对应四输出。选容量为16×4位的ROM .

3) 由函数最小项表达式，按位线编程，画阵列图



$$\begin{cases} Y_1 = \sum m(1,4,8,13) \\ Y_2 = \sum m(6,7,10,11,12,13,14,15) \\ Y_3 = \sum m(1,7,9,12,14,15) \\ Y_4 = \sum m(0,1,3,4,5,9,10,11,12,13) \end{cases}$$

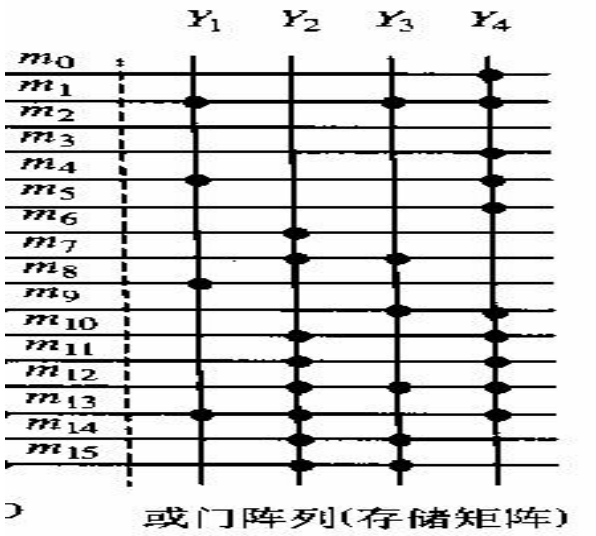
方法二：按字线编程:列出函数的真值表

- 视ROM的地址为输入,函数值为地址中内容。
- 根据地址0000－1111，分别计算Y1,Y2,Y3,Y4,依次写入到ROM中。
- 如地址0000，则 Y1Y2Y3Y4＝0001。
  对地址0001，　　Y1Y2Y3Y4＝1011。

  ….

  对地址1111，　　Y1Y2Y3Y4＝0110。

$$\begin{cases} Y_1 = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + AB\bar{C}D \\ Y_2 = AB + AC + BC \\ Y_3 = AB\bar{D} + BCD + \bar{B}\bar{C}D \\ Y_4 = \bar{A}\bar{C} + B\bar{C} + \bar{B}D + A\bar{B}\bar{C} \end{cases}$$



ROM的应用（函数运算表）

[例] 用ROA实现函数$y=x^2$的运算表电路。设x的取值范围为0---15的正整数.

分析：

- x的取值范围为0---15 的正整数，则对应的4位二进制数，用$B=B_3B_2B_1B_0$表示。
- 算出y的最大值是$15^2=225$，用8位二进制$Y=Y_7Y_6Y_5Y_4Y_3Y_2Y_1Y_0$表示。
- 由此可列出$y=x^2$的真值表。

用 ROM 实现函数 $y=x^2$ 的真值表

输入				输出								备注
B_3	B_2	B_1	B_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0	十进制数
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	1	1
0	0	1	0	0	0	0	0	0	1	0	0	4
0	0	1	1	0	0	0	0	1	0	0	1	9
0	1	0	0	0	0	0	1	0	0	0	0	16
0	1	0	1	0	0	0	1	1	0	0	1	25
0	1	1	0	0	0	1	0	0	1	0	1	36
0	1	1	1	0	0	1	1	0	0	0	1	49
1	0	0	0	0	1	0	0	0	0	0	0	64
1	0	0	1	0	1	0	1	0	0	0	1	81
1	0	1	0	0	1	1	0	0	1	0	0	100
1	0	1	1	0	1	1	1	1	0	0	1	121
1	1	0	0	1	0	0	1	0	0	0	0	144
1	1	0	1	1	0	1	0	1	0	0	1	169
1	1	1	0	1	1	0	0	0	1	0	0	196
1	1	1	1	1	1	1	0	0	0	0	1	225

方法1　按字线编程，分别将0、1、4、9、…、225,写入到地址0000、0001、0010、0011、…、1111。

- 方法2：按位线编程.

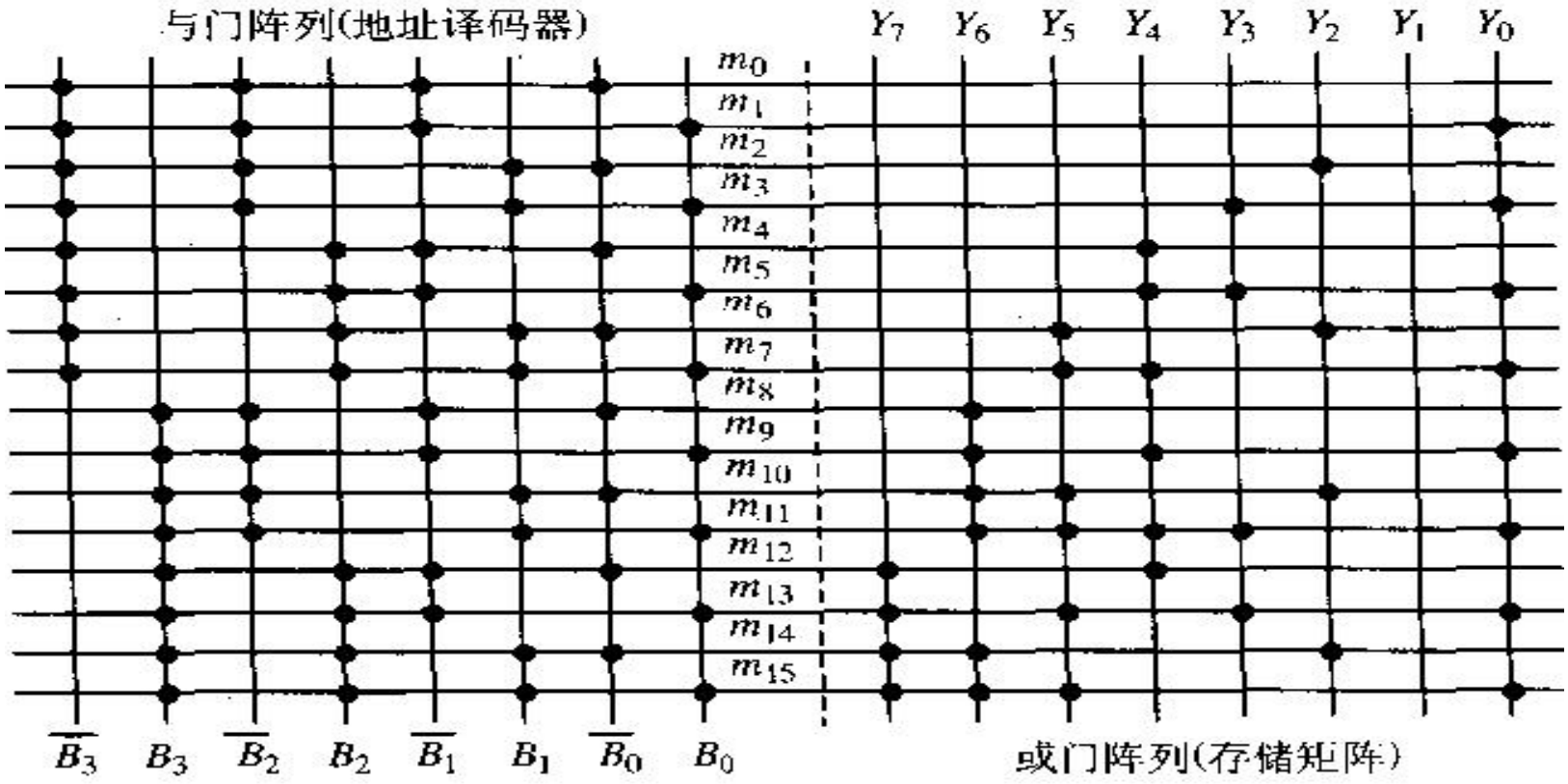
  写出输出函数的最小项表达式

$$
\begin{cases}
Y_7 = \sum m(12,13,14,15) \\
Y_6 = \sum m(8,9,10,11,14,15) \\
Y_5 = \sum m(6,7,10,11,13,15) \\
Y_4 = \sum m(4,5,7,9,11,12) \\
Y_3 = \sum m(3,5,11,13) \\
Y_2 = \sum m(2,6,10,14) \\
Y_1 = 0 \\
Y_0 = \sum m(1,3,5,7,9,11,13,15)
\end{cases}
$$

ROM的应用

ROM的连线图

- 按字线编程:
- 按位线编程:



用 ROM 实现函数 $y=x^2$ 的连线图

FLASH存储器

也称闪存，高密度非易失性的读／写存储器。有RAM和ROM的优点。

- FLASH存储元
- FLASH的基本操作
- FLASH的阵列结构

FLASH存储元

存储元 单MOS晶体管　含控制栅和浮空栅。漏极D,源极S

浮空栅:　　电子多:　状态0

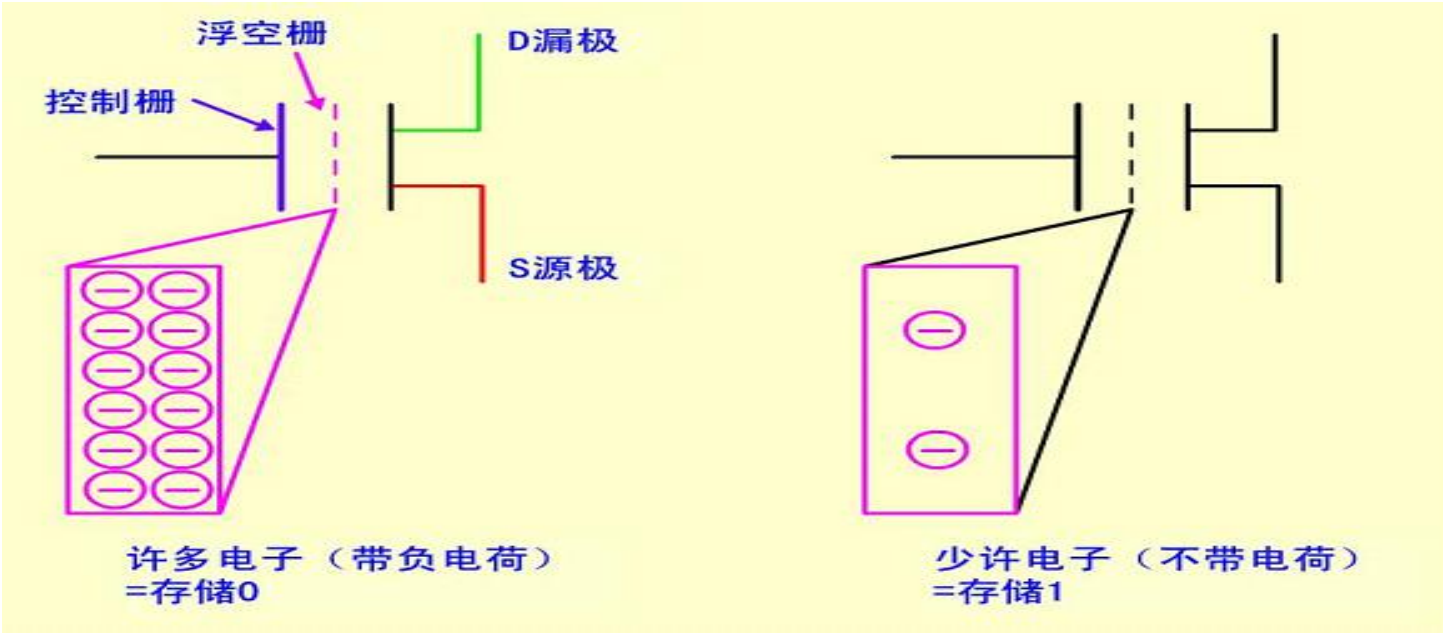
　　电子少:　状态1

.

控制栅正电压时读出: 状态1:SD导通 ;状态0:SD不导通 ;

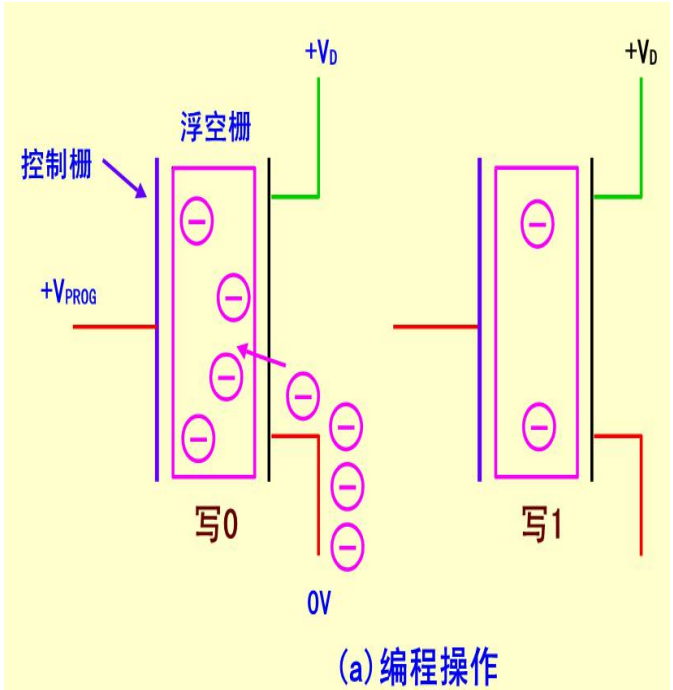
源极S正电压时擦除: 吸收浮空栅电子　　等同　　写1

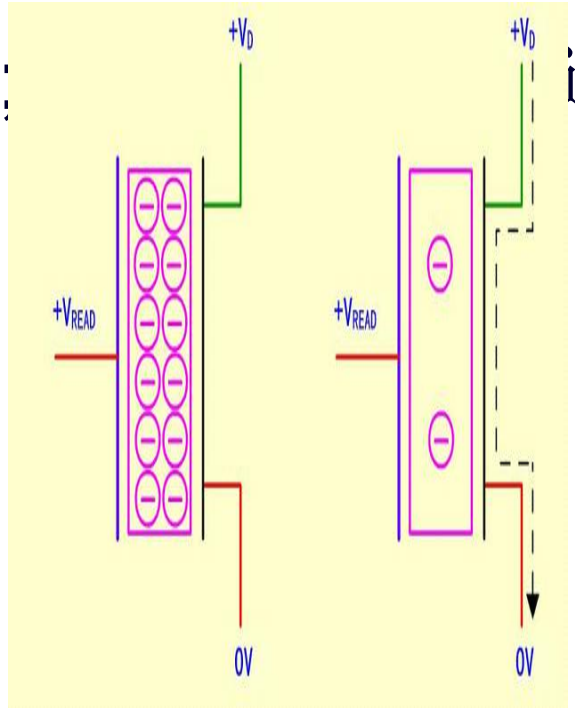
控制栅正电较大时编程: 浮空栅聚集电子　　写0

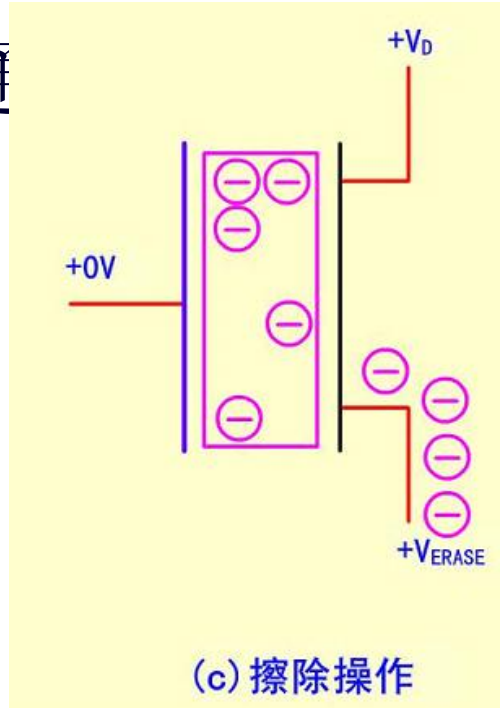


三种基本操作

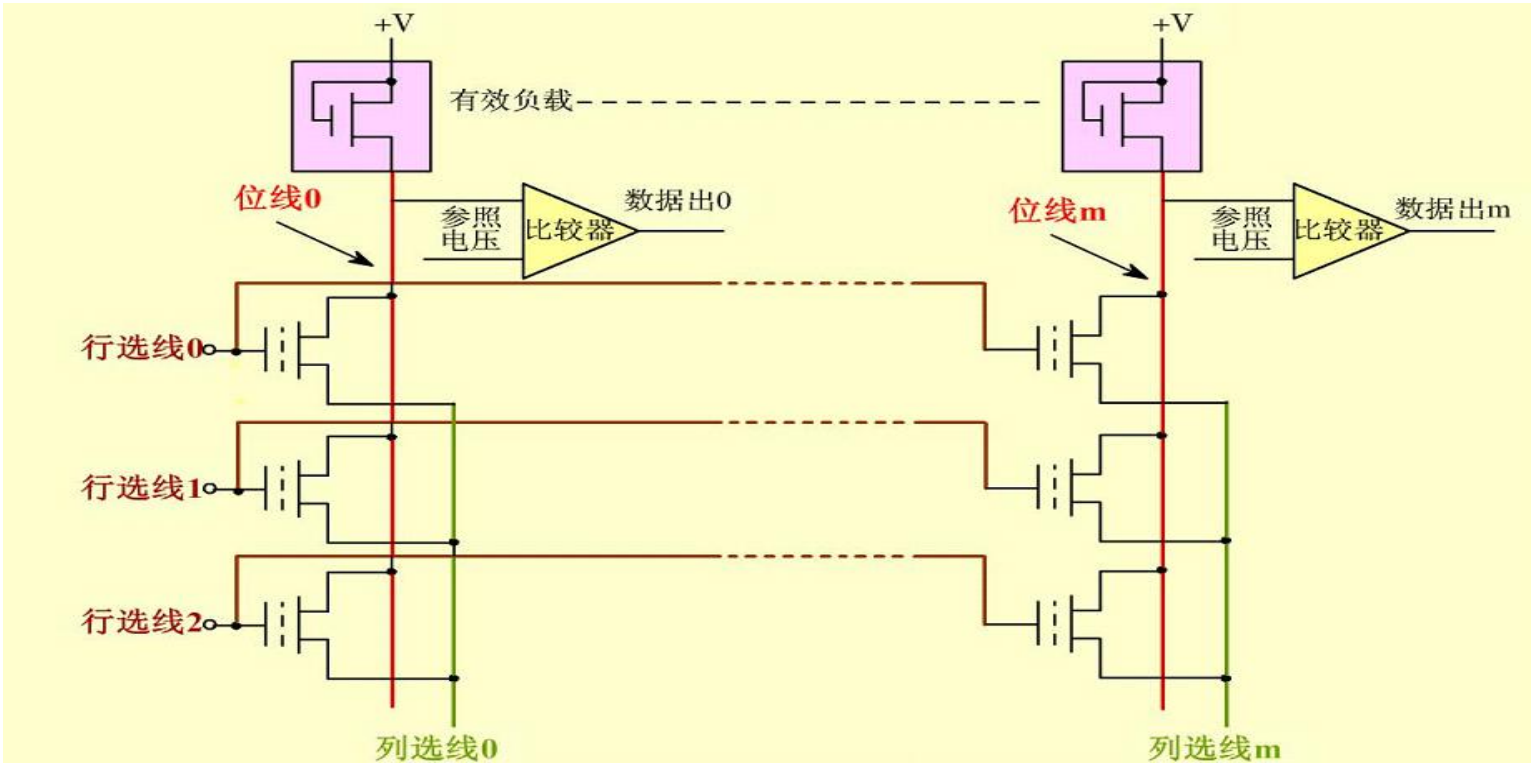
- 编程:控制栅为编程电压V_P,定义为写0 操作。
  控制栅无正电　　：定义为 写1操作
- 擦除:源极S为正电压，相当于写1.
- 读出：　控制栅加读出电压V_R







FLASH阵列结构


+V
+V
有效负载
位线0
参照电压
比较器
数据出0
位线m
参照电压
比较器
数据出m
行选线0
行选线1
行选线2
列选线0
列选线m

FLASH 主要优点

1　非易失
2　高密度
3　单晶体管存储元
4　可写性

各种存储器性能比较(表4.2，P，118）

小　结

- ROM主要由与门—或门”二级电路组成。与**阵列不可编程，**或门可**编程。**

- 组合逻辑函数可写为标准与或表达式．可用PROM实现．但PROM只能实现函数的标准与或表达式。

- PROM能制作数码转换器、函数转换表、字符显示电路等。
- PROM的编程:按字线编程，按位线编程。

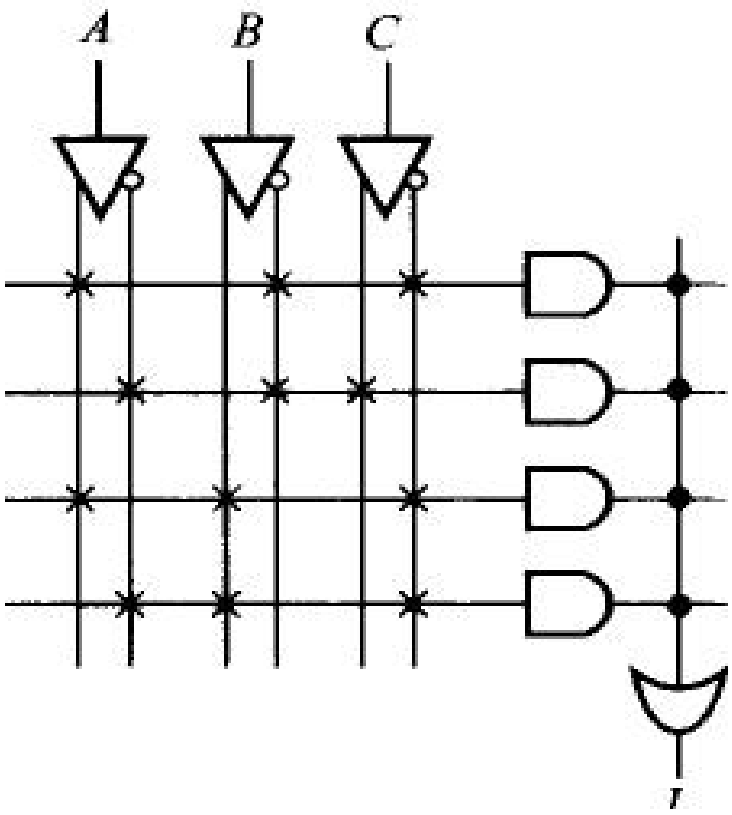
第四章 作 业

- 4, 6,10

- 以下只回答需要几片？地址线、数据线各几条？
- 11，12，13，14，15

课堂 练习

1　写出电路逻辑表达式



2　如图2-2：1）描述存储器2716的容量。2）用2716组成8K×16的存储器需要几片？

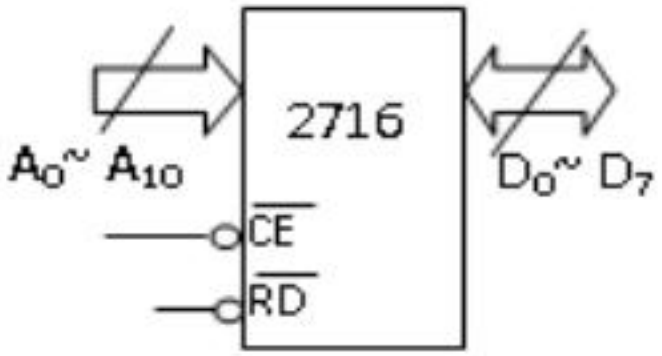


图 2-2

3. RAM 2112(256×4)芯片的逻辑符号如图所示。

1）用多少片RAM 2112芯片能组成512×8的RAM?

2）512×8 RAM共有多少根地址线?

3）画出512×8RAM的连线图（用非门实现）