

7. 半导体存储器

7.1 只读存储器

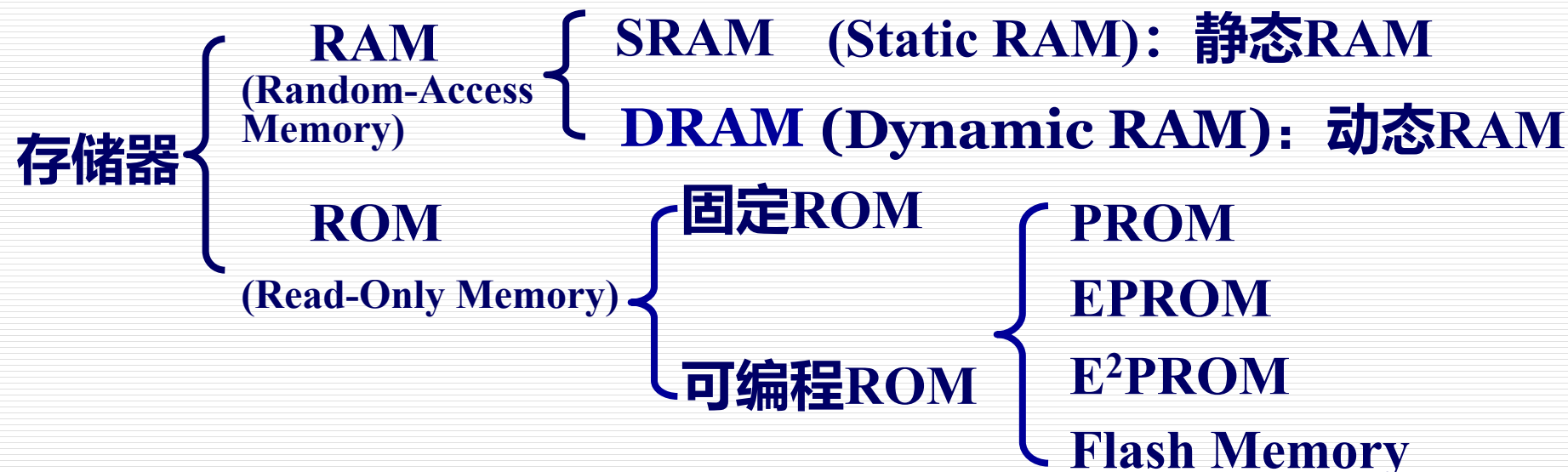
7.2 随机存取存储器

教学基本要求：

- 掌握半导体存储器字、位、存储容量、地址等基本概念。
- 掌握RAM、ROM的工作原理及典型应用。
- 了解存储器的存储单元的组成及工作原理。
- 了解随机存取存储器SRAM、DRAM 基本结构、工作原理和分类

概 述

半导体存储器是能存放大量二值信息的半导体器件。



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：

存储器的**存储容量**和**存取时间**是反应存储器性能好坏的重要技术指标。

存储容量 (M): **存储单元**的数目

存储容量 (M) = **字数** × **字长**

字的总量，字数
 $= 2^n$ ，这里n为地
址码的位数，即
地址线的总数

一个字所含的
位数，即数据
线的总数

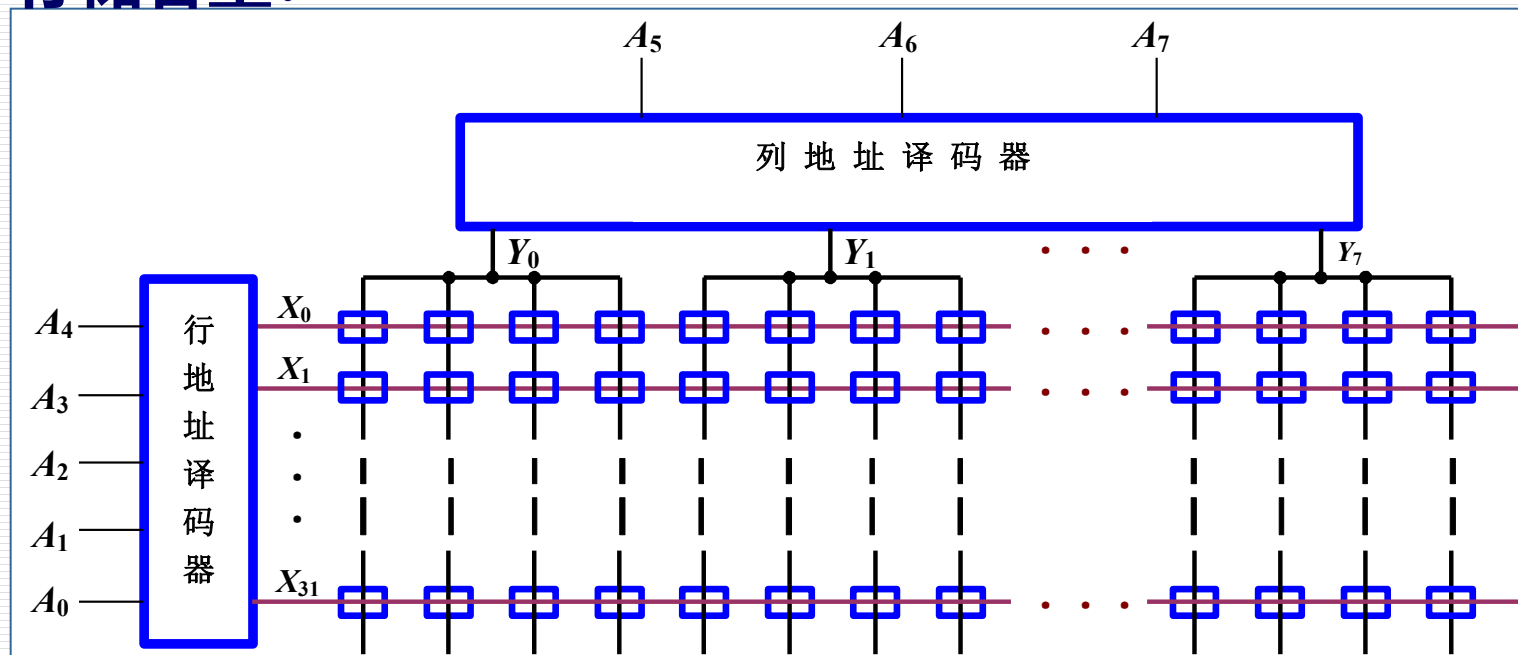
字：计算机中作为一个整体被存取传送处理的一组数据

地址单元：构成字的存储单元。**地址**：每个字的编号。

字数: $2^5 \times 2^3 = 256$ (行地址+列地址)

字长: 4位。

存储容量: 256×4



例7.1 容量为 $64K \times 4$ 的存储器有多少个存储单元？地址码至少需要几位？数据位是几位？

解：

存储单元数=字数 \times 位数

地址码的位数 n 与字数 N 的关系： $N=2^n$

数据位数=位数

**因此，存储单元=256k； $64k=2^{16}$ ，地址码位数为16。
数据位为4。**

思考1：容量为 $1M \times 4$ 的存储器有多少个存储单元，至少需要多少根地址线？多少根数据线？

思考2：当单个芯片无法满足存储容量要求时，如何进行容量扩展？

7.1 只读存储器

7.1.1 ROM的基本结构

7.1.2 二维译码与存储阵列

7.1.3 可编程ROM

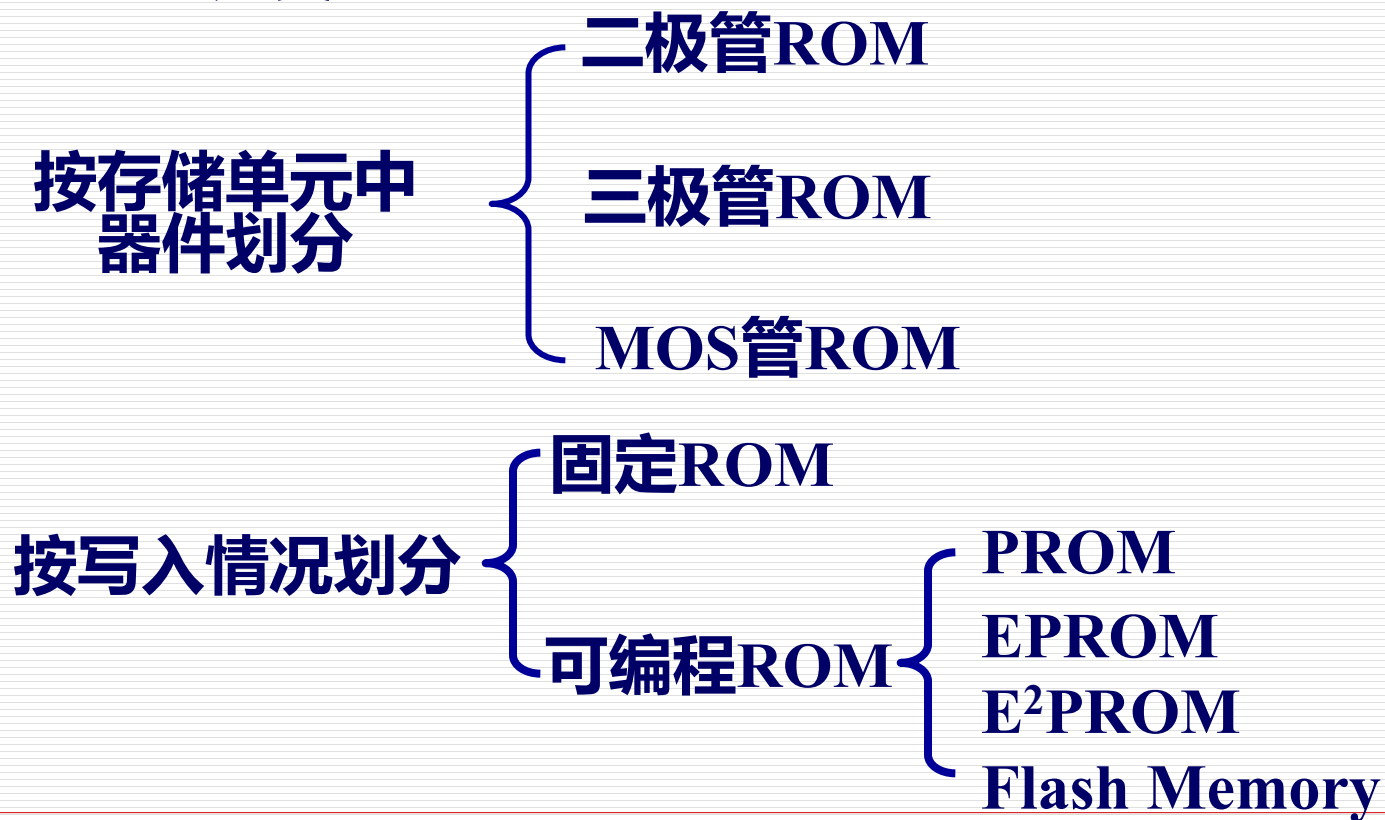
7.1.4 ROM的读操作实例

7.1.5 ROM的应用举例

7.1.1 ROM的基本结构

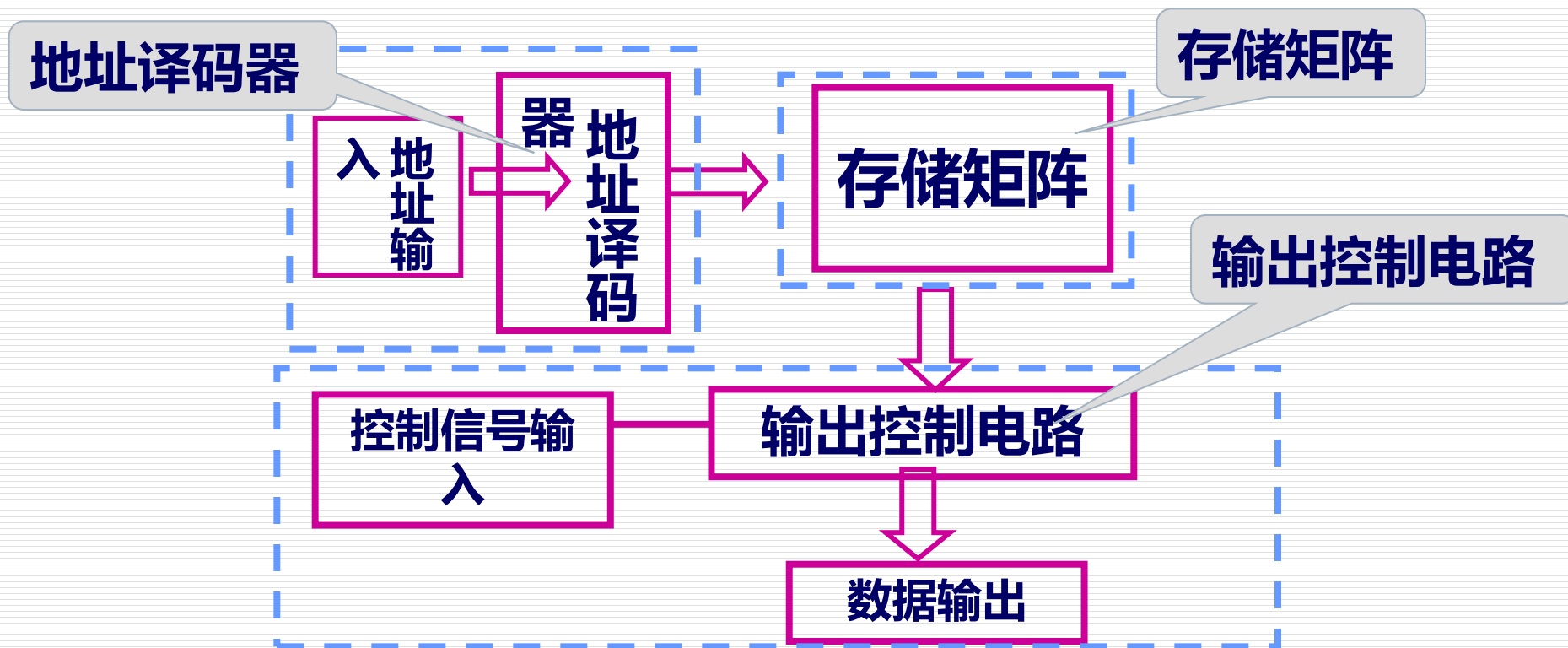
只读存储器，工作时内容只能读出，不能随时写入，所以称为只读存储器。(Read-Only Memory)

ROM的分类



7.1.1 ROM的基本结构

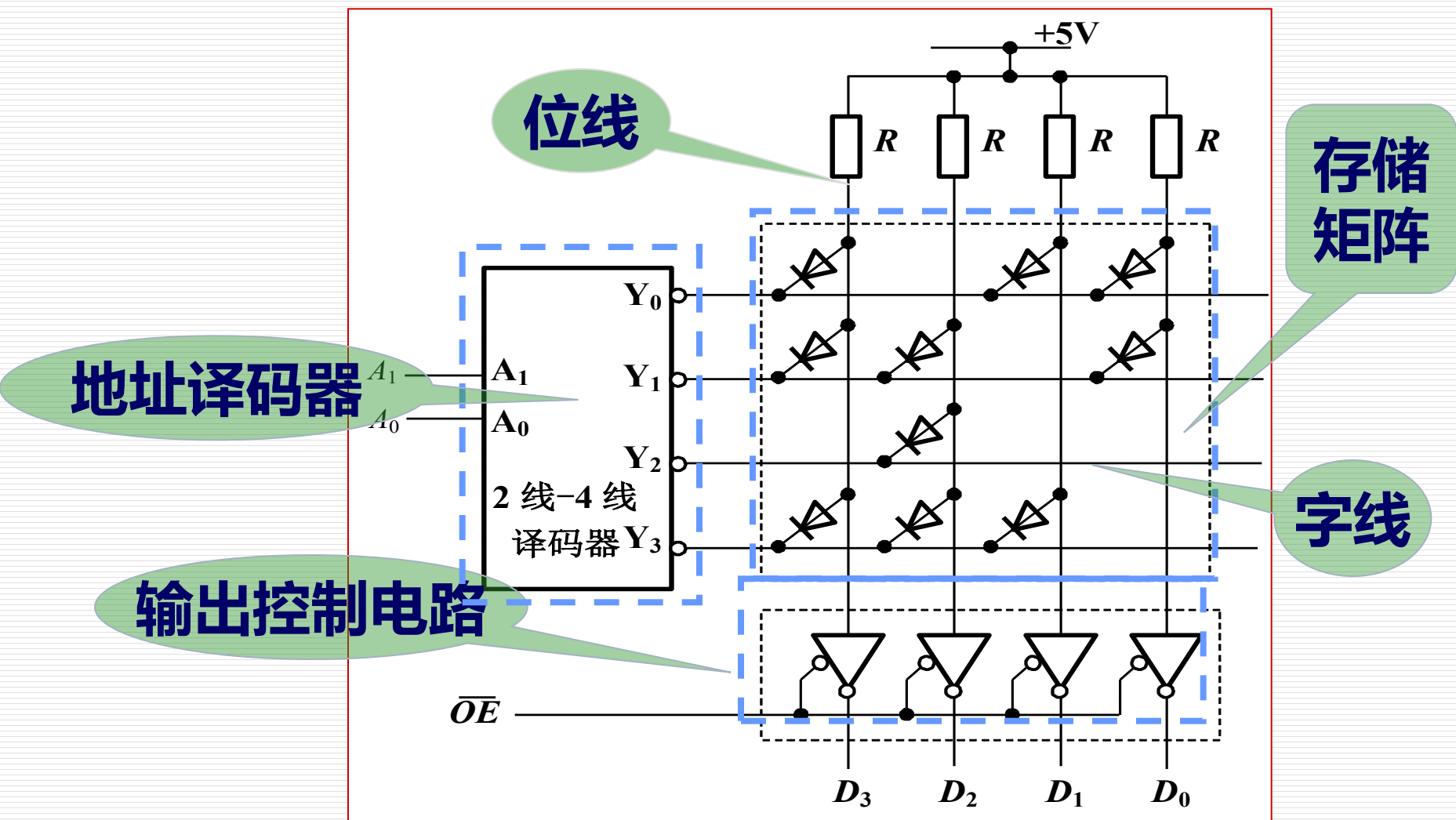
ROM是一种永久性数据存储器，其中的数据一般由专用的装置写入，数据一旦写入，不能随意改写，在切断电源之后，数据也不会消失。



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。

1) ROM (二极管PROM) 结构示意图

$M=4 \times 4$

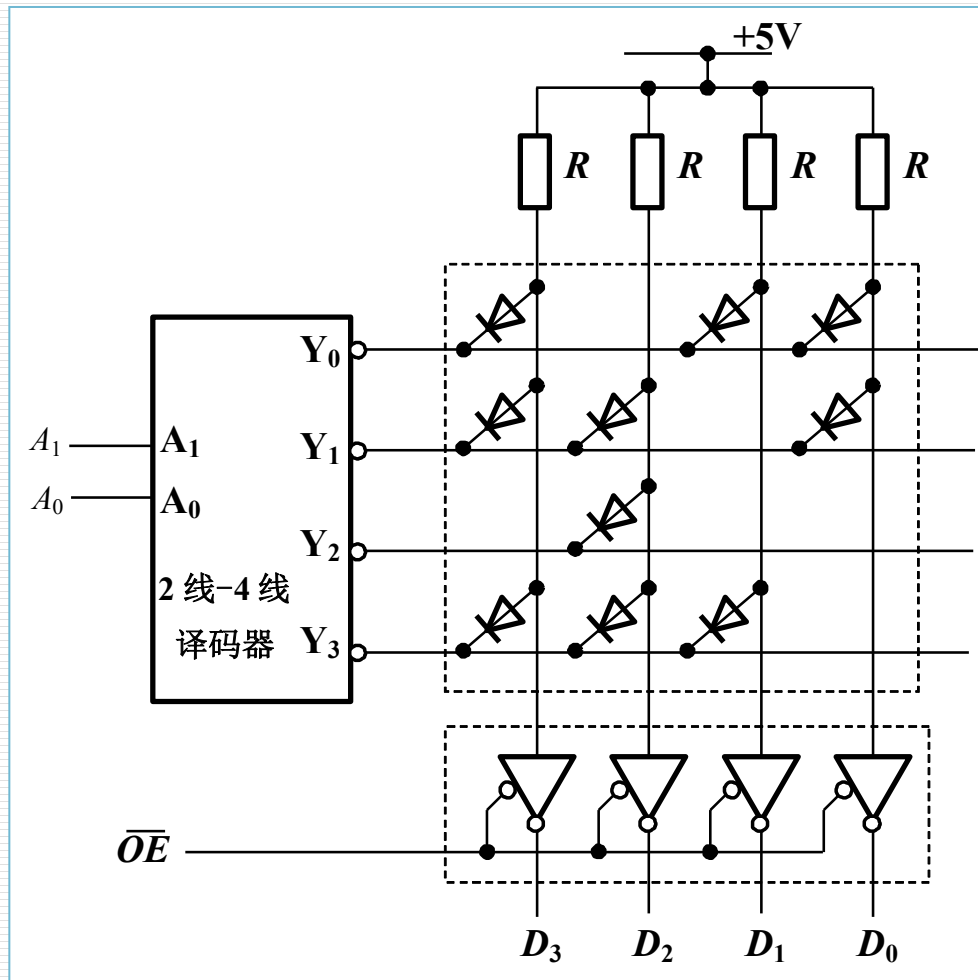


右图所示存储器存储的内容

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



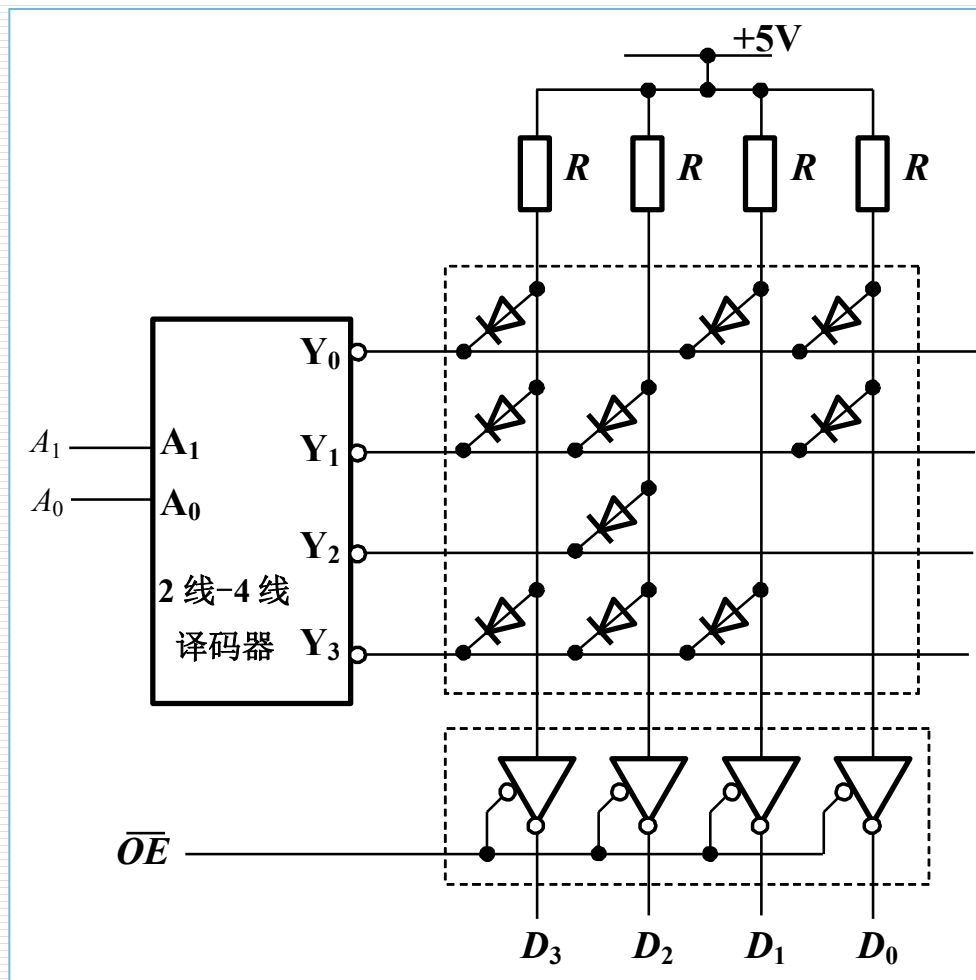
右图所示存储器存储的内容

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

思考1: ROM是组合电路么?

思考2: 如果ROM是组合电路, 那么输入和输出分别是什么?

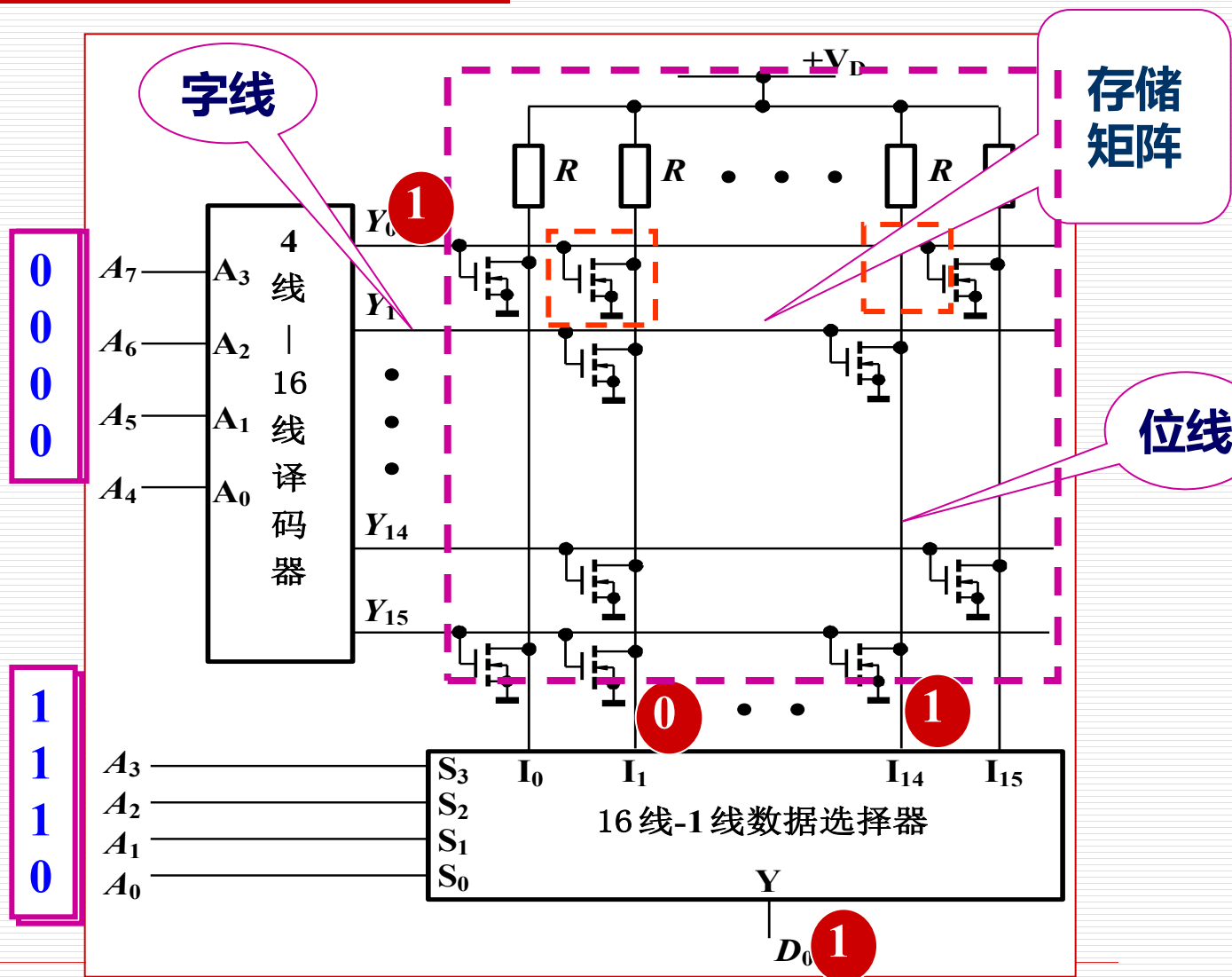


7.1.2 MOS管构成二维译码ROM

•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。

思考：该存储器的容量=？



7.1.3 可编程ROM

1、掩模ROM：可用二极管、MOS管或BJT管，出厂时数据已经“固化”了，不能更改

2. 一次性可编程存储器PROM：存储阵列由带金属熔丝的二极管构成，智能烧写一次

3. 可重复编程的ROM：

光可擦除可编程存储器EPROM：存储阵列由SIMOS管构成（OTP EPROM），有石英盖板，用紫外灯或者X射线照射15-20分钟擦除

电可擦除可编程存储器E²PROM：存储阵列由Flotox MOS管构成，在线电擦除，但集成度低

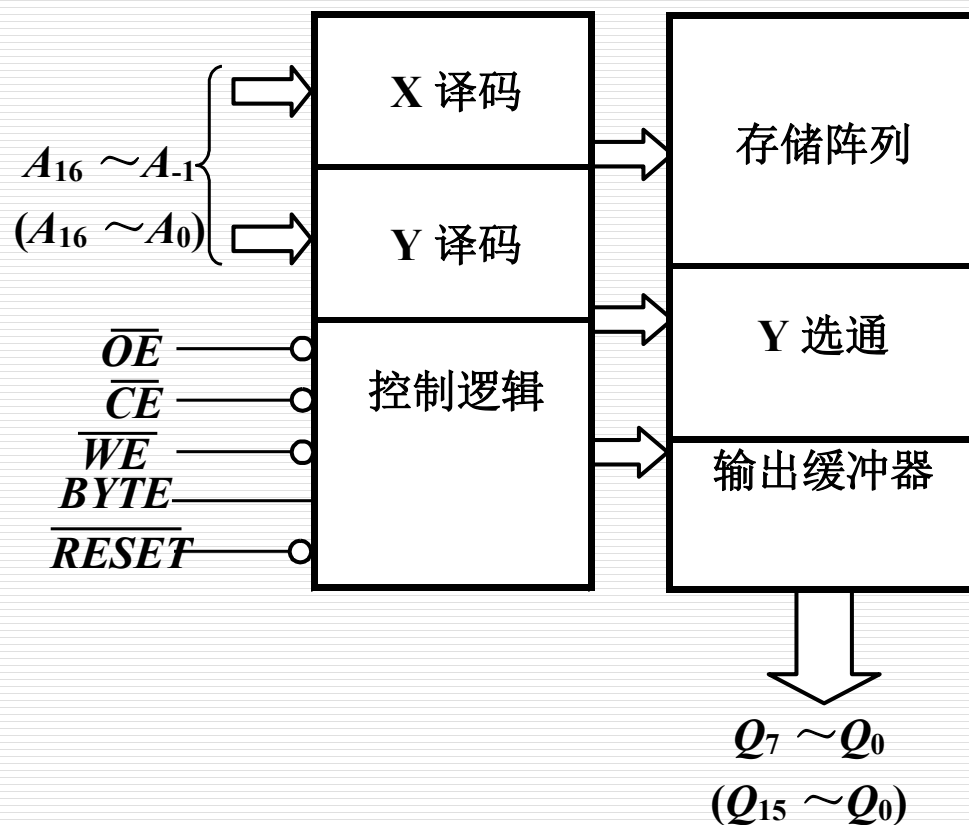
Flash Memory：存储阵列由快闪叠栅MOS管构成，有NOR和NAND型两类

几种ROM性能比较

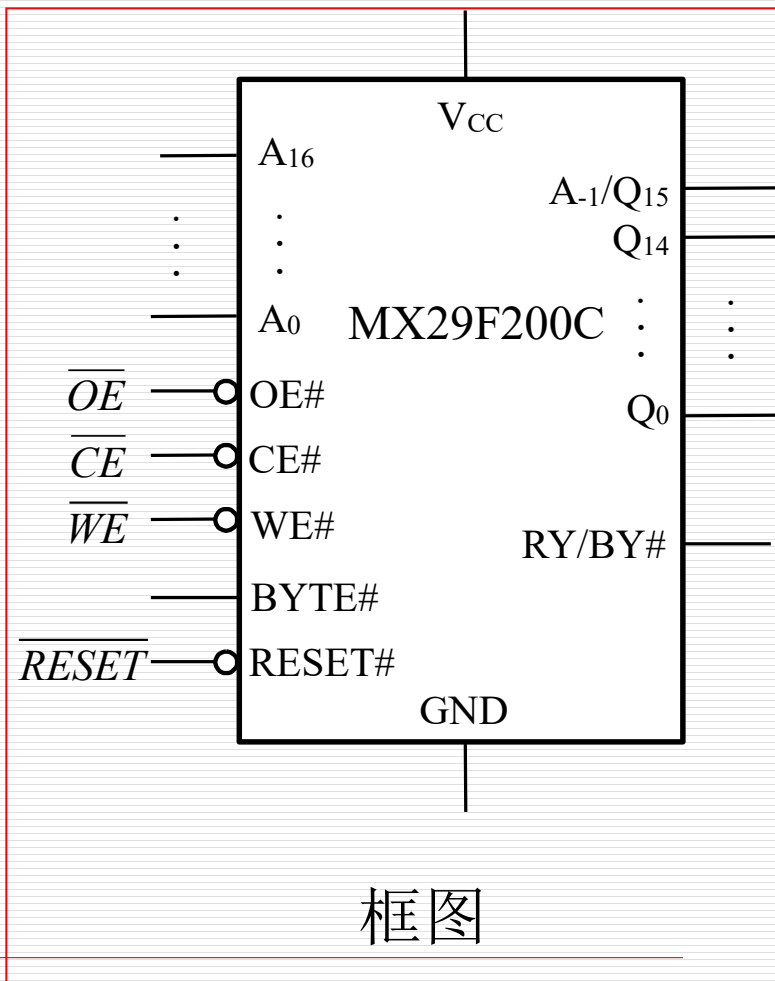
	Flash	ROM	EPROM	E ² PROM
非易失性	是	是	是	是
高密度	是	是	是	否
单管存储单元	是	是	是	否
在系统可写	是	否	否	是

7.1.4 ROM实例

1. NOR Flash芯片MX29F200C简介



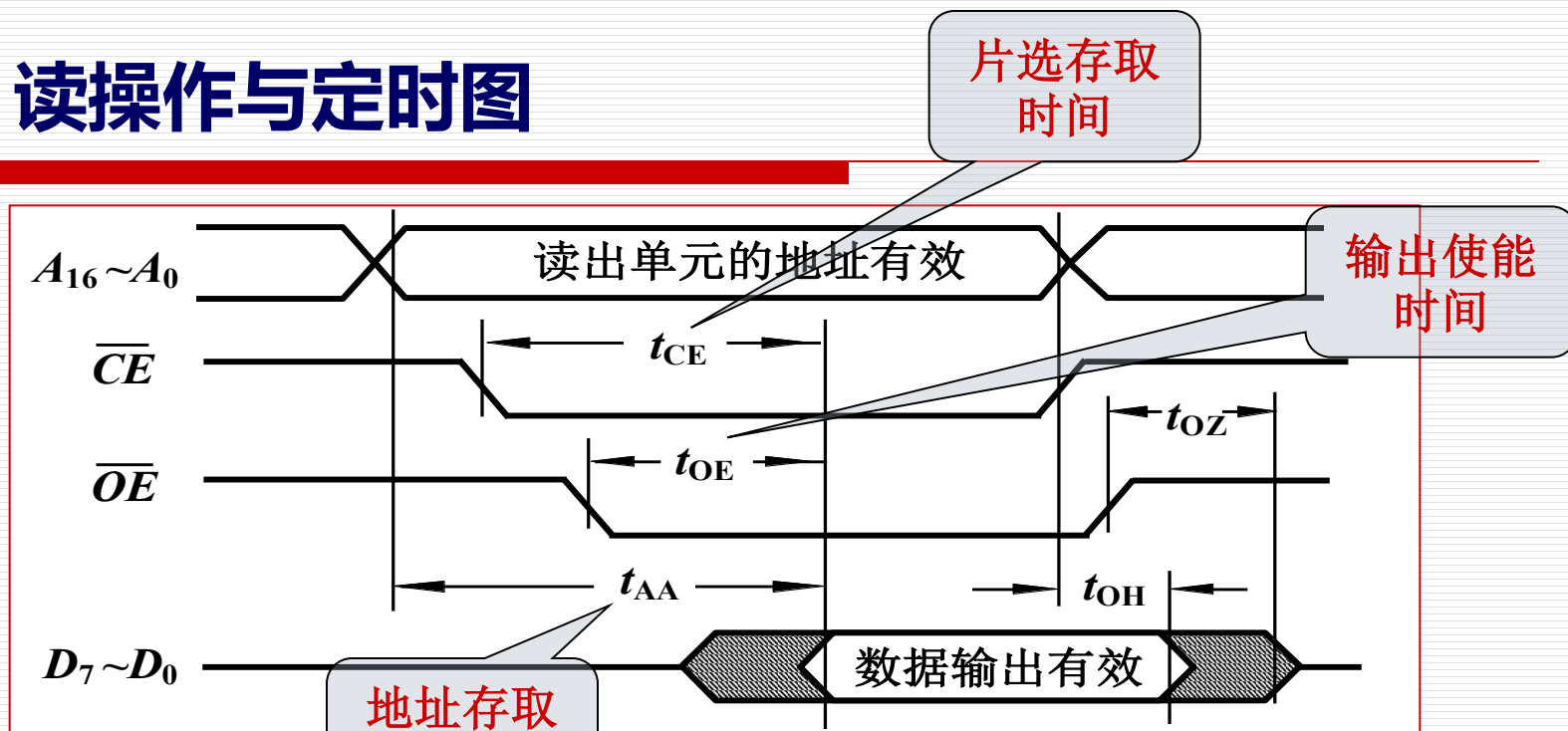
256K×8位/128K×18位ROM



部分工作模式

工作模式	\overline{RESET}	\overline{CE}	\overline{OE}	\overline{WE}	$A_{16} \sim A_1$	$Q_7 \sim Q_0$
读	H	L	L	H	A_i	数据输出
输出无效	H	L	H	H	X	高阻
等待	H	H	X	X	X	高阻
写	H	L	H	L	A_i	数据输入
复位	L	X	X	X	X	高阻

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, 有效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, 经过一定延时后数据线呈高阻态, 本次读出结束。

7.1.5 ROM应用举例

(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。

(2) 利用ROM实现逻辑函数

基本思想：逻辑函数的输入变量由ROM地址线输入、逻辑函数值由ROM数据线输出

□ **思考：**试确定用ROM实现3位二进制数相乘的乘法器所需要的容量。

例7.3 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	$O_3 O_2 O_1 O_0$ 格雷码	C	$I_3 I_2 I_1 I_0$ 格雷码	$O_3 O_2 O_1 O_0$ 二进制码
0	0000	0000	1	0000	0000
0	0001	0001	1	0001	0001
0	0010	0011	1	0010	0011
0	0011	0010	1	0011	0010
0	0100	0110	1	0100	0111
0	0101	0111	1	0101	0110
0	0110	0101	1	0110	0100
0	0111	0100	1	0111	0101
0	1000	1100	1	1000	1111
0	1001	1101	1	1001	1110
0	1010	1111	1	1010	1100
0	1011	1110	1	1011	1101
0	1100	1010	1	1100	1000
0	1101	1011	1	1101	1001
0	1110	1001	1	1110	1011
0	1111	1000	1	1111	1010

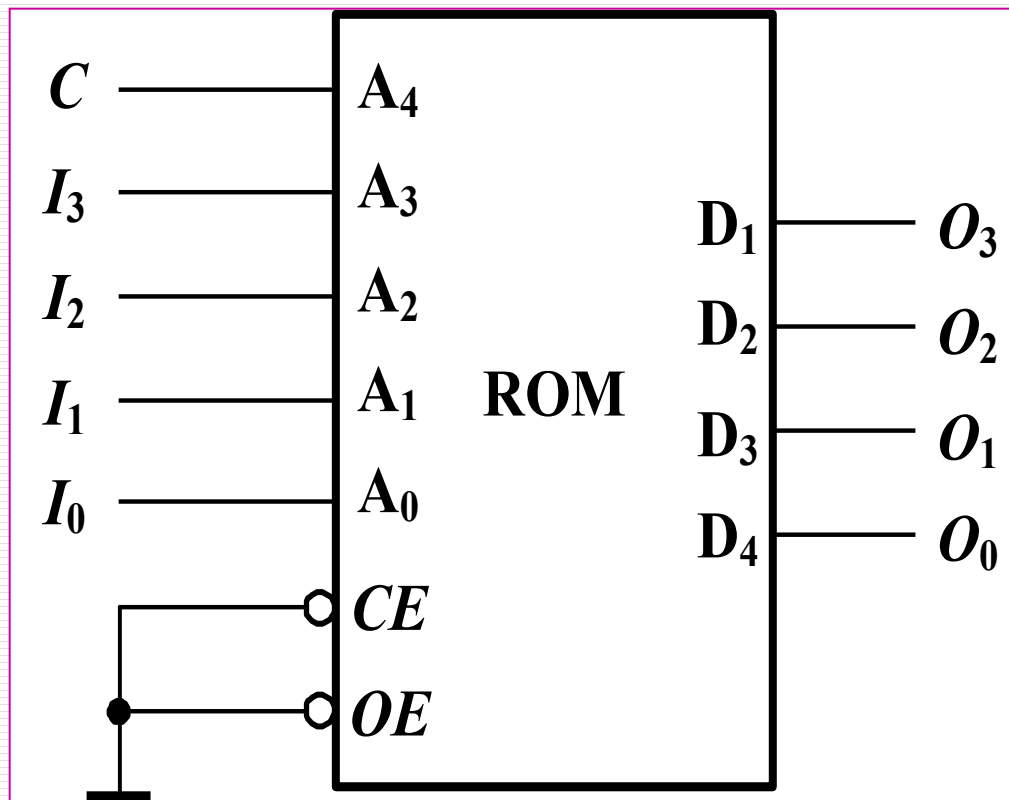
$$C=A_4$$

$$I_3 I_2 I_1 I_0 = A_3 A_2 A_1 A_0$$

$$O_3 O_2 O_1 O_0 = D_3 D_2 D_1 D_0$$

C (A_4)	$I_3 I_2 I_1 I_0$ ($A_3 A_2 A_1 A_0$) 二进制码	$O_3 O_2 O_1 O_0$ ($D_3 D_2 D_1 D_0$) 格雷码	C (A_4)	$I_3 I_2 I_1 I_0$ ($A_3 A_2 A_1 A_0$) 格雷码	$O_3 O_2 O_1 O_0$ ($D_3 D_2 D_1 D_0$) 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



例7.4 利用ROM实现逻辑函数

A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0
0	0	1	0	0	1	0
0	1	0	0	1	0	0
0	1	1	1	0	0	0
1	0	0	1	1	0	0
1	0	1	1	0	0	1
1	1	0	0	1	1	0
1	1	1	0	0	1	1

$$A_2 A_1 A_0 = ABC$$

$$D_3 = Y_3$$

$$D_2 = Y_2$$

$$D_1 = Y_1$$

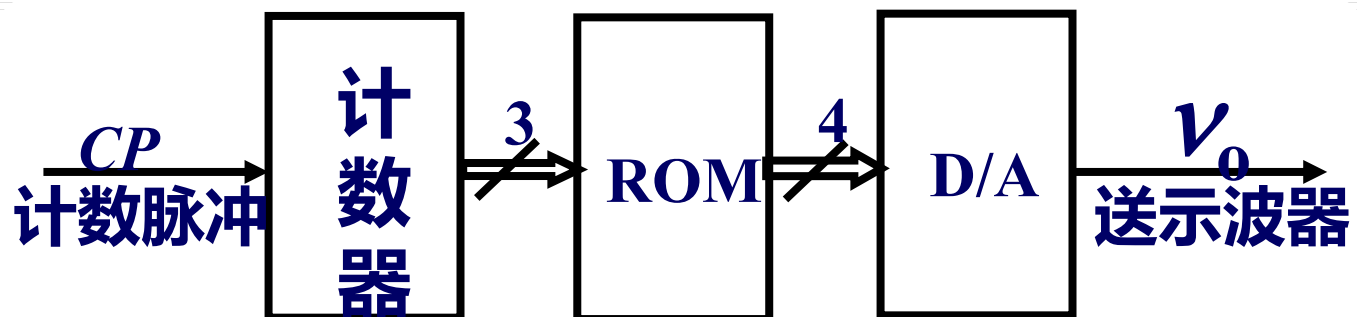
$$D_0 = Y_0$$

$$Y_3 = f(ABC) = \sum m(3,4,5) \quad Y_2 = f(ABC) = \sum m(2,4,6)$$

$$Y_0 = f(ABC) = \sum m(5,7) \quad Y_1 = f(ABC) = \sum m(1,6,7)$$

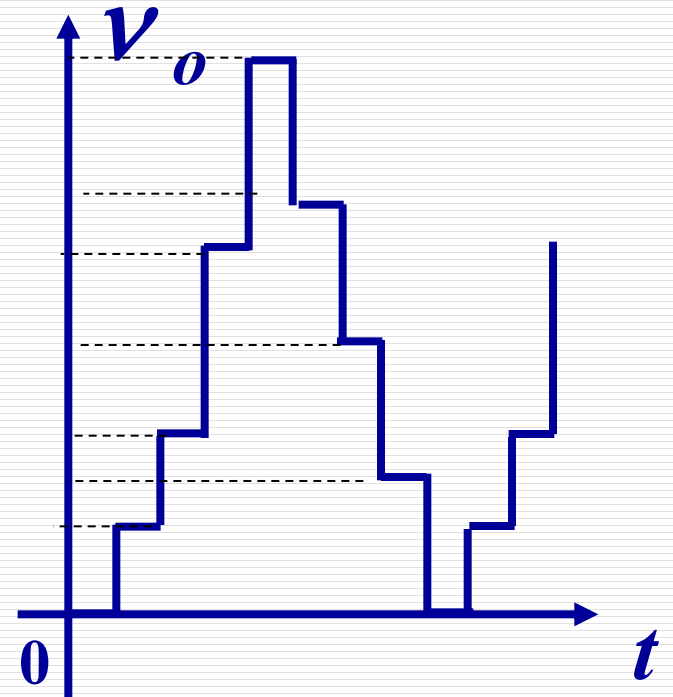
改变ROM的内容，即可得不同的逻辑函数

例7.5 ROM 在波形发生器中的应用

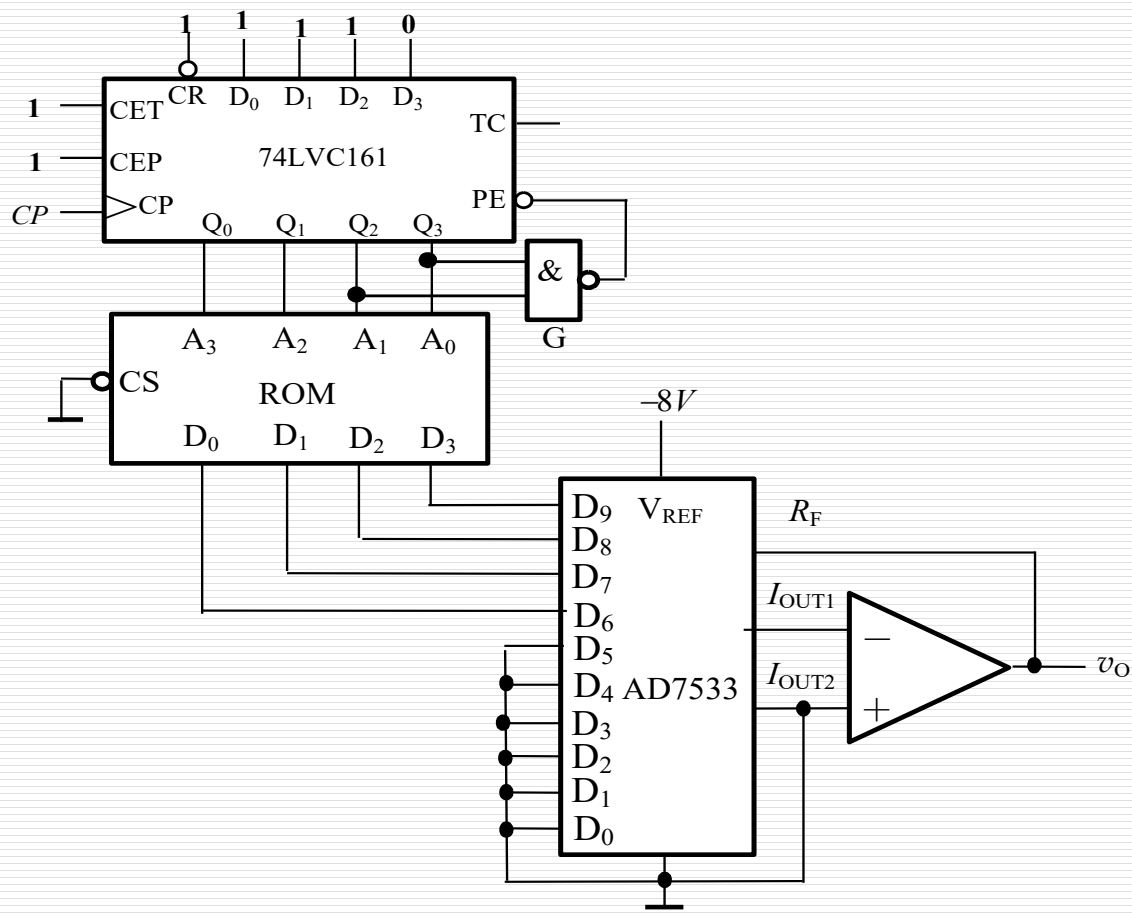


A_2	A_1	A_0	D_3	D_2	D_1	D_0	D/A
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	2
0	1	0	0	1	0	0	4
0	1	1	1	0	0	0	8
1	0	0	1	1	0	0	12
1	0	1	1	0	0	1	9
1	1	0	0	1	1	0	6
1	1	1	0	0	1	1	3

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



ROM的数据表



A_3	A_2	A_1	A_0	D_9	D_8	D_7	D_6
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	1	1
0	1	1	0	1	0	0	0
0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	1	0	0	0	1	1
1	0	1	1	0	1	0	0
1	1	0	0	0	0	1	1
1	1	0	1	0	1	1	0
1	1	1	0	0	1	1	1
1	1	1	1	1	0	0	0

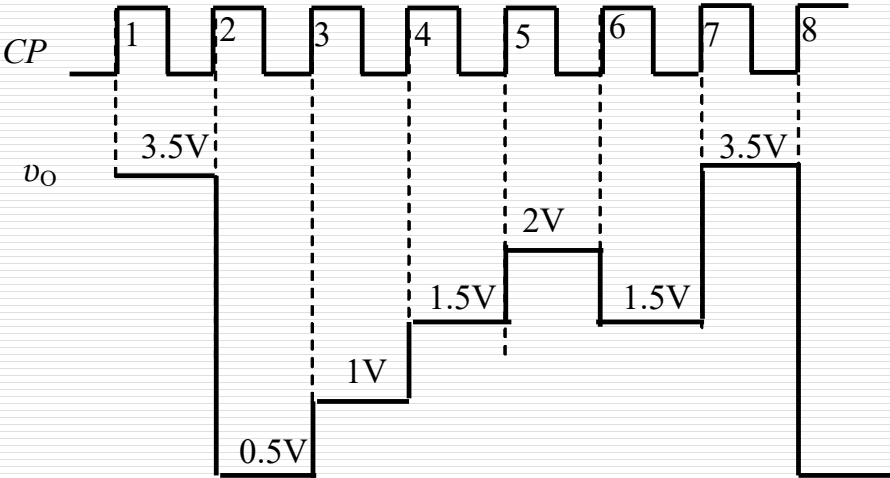
$D_9 D_8 D_7 D_6 = 0 0 0 1$ 时 $v_o = 0.5V$

计数器状态表

Q_3	Q_2	Q_1	Q_0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0

ROM的数据表

A_3	A_2	A_1	A_0	D_9	D_8	D_7	D_6
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	1	1
0	1	1	0	1	0	0	0
0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	1	0	0	0	1	1
1	0	1	1	0	1	0	0
1	1	0	0	0	0	1	1
1	1	0	1	0	1	1	0
1	1	1	0	0	1	1	1
1	1	1	1	1	0	0	0



$D_9 D_8 D_7 D_6 = 0 0 0 1$ 时 $v_0 = 0.5V$