前　言

数字逻辑器件通常分为三大类:一是目前广泛使用的由基本逻辑门和触发器构成的中小规模集成逻辑器件;二是由软件组态的大规模和超大规模集成器件,如微处理器、单片机等;三是专用集成电路 ASIC (Application-Spesific Integrated Circuit)。ASIC 又分为标准单元 (Standard Cell)、门阵列 (Gate Array) 和可编程逻辑器件 PLD (Programmable Logic Device),是近期迅速发展起来的新型逻辑器件。PLD 是用户可自行定义功能的逻辑器件,使用方便灵活。自 70 年代初期只读存储器 PROM (Programmable Read Only Memory) 问世以来,PLD 已有了长足的发展。70 年代中期出现了现场可编程逻辑阵列 FPLA (Field Programmable Logic Array),70 年代末期又出现了可编程阵列逻辑 PAL (Programmable Array Logic),80 年代中期又推出了通用阵列逻辑 GAL(Generic Array Logic),近几年又推出了可擦可编程逻辑器件 EPLD (Erasable Programmable Logic Device)。目前,GAL、EPLD 等高级逻辑器件已广泛应用于数字系统设计中。这些高级逻辑器件具有可重复擦写、可重复编程以及可加密的功能,并具有体积小、可靠性高、低功耗以及完全可测试等特点。这些器件的灵活性和通用性使得它们成为研制和设计数字系统的最理想选择。

目前,这些高级逻辑器件应用技术又有了新的发展。器件本身的性能越来越完备,规模也越来越大。比如,一片 EPLD (ATV 5000)可代替上百片中小规模集成电路,也可代替几十片 GAL 或 PAL,可用来设计很复杂的时序逻辑电路。在设计方法上,为硬件电路的设计带来了根本性的变革,设计人员只要一台微机及相应的开发工具,用这些逻辑器件就可以研制出所需的各种逻辑电路。设计软件也越来越完善,比如,本书所介绍的 ABEL 高级设计语言,不但可以直接形成编程器下载文件 *.JED,还可以进行设计级仿真测试 SIM,只要在微机上仿真无误,设计的电路就正确。更高级的用法,只需在微机上输入原理图即可设计。编程工具已经通用化,对各种 PLD 器件都可进行编程,为了适应超大规模高级逻辑器件的编程,插脚配件也更加齐备。总之,现已形成了一整套 PLD 器件计算机辅助设计系统,为 PLD 器件的推广和应用创造了良好环境。

近年来,为了适应数字电子技术发展的需要,有些国家的高等院校已为高年级学生和研究生开出了类似内容的课程,国内也有些院校开出了同样内容的讲座和选修课程。本书就是在为高年级学生开设选修课程的基础上,将讲稿经过加工、整理而完成。本书分上下两篇。上篇主要介绍可编程逻辑器件的结构与设计方法。第 1 章到第 5 章分别介绍了只读存储器(PROM、EPROM、EEPROM)、可编程逻辑阵列(PLA、PLS)、可编程阵列逻辑(PAL)、通用阵列逻辑(GAL)以及可擦除可编程逻辑器件(EPLD)等各种 PLD 器件的结构与应用。第 6 章介绍了 PLD 的设计方法,包括真值表法、逻辑方程法以及状态图法。下篇主要介绍 ABEL 设计语言与编程设计。第 7 章到第 9 章介绍了 ABEL 设计语言的结构和语言处理程序。第 10 章和第 11 章给出了用各种方法设计 PLD 的设计样例以及设计仿真时应注意的一些问题。第 12 章进一步阐述了如何运用高级器件特性进行设计。第 13 章给出了五个实验题目,用于指导上机如何执行 ABEL 语言。

本书全面系统介绍了可编程逻辑器件的结构原理和设计方法，取材新颖，包括新近推出的EPLD器件。把器件的结构原理和设计软件相结合，融汇为一个整体，并加以全面系统的介绍是本书的特点，这样，既便于读者学习和掌握，又有资料性。书中所选用的设计样例，有一部分来源于设计手册，有一部分是作者自行编制的，均经过作者上机运行，检查无误，具有实用价值，可作为复杂逻辑设计的单元模块应用。阅读本书读者要求具有数字电子技术基础的知识。本书适于从事电子技术、自动控制以及计算机硬件设计的工程技术人员和设计人员，也可作为高等学校数字电子技术基础课的后续课教材使用。

本书由北京理工大学程震先副教授审稿，经北京电子科技学院教材编审委员会审定，作为本院选用教材。

作　　者

1996年1月 于北京电子科技学院

目　录

上篇　PLD 器件的结构与设计法

下篇 ABEL 设计语言与编程设计

上篇　PLD 器件的结构与设计法

目前推出的 PLD 器件种类很多，型号各异，为了使读者能够正确地选用器件，在这一篇中介绍了各种常见 PLD 器件的结构原理和技术特性。在叙述方法上，主要以存储阵列结构的不同逐一展开，加以介绍。在应用基础知识时，先引入组合逻辑的概念而后引入时序逻辑的概念，进而引入状态机的概念。为了读者对 PLD 器件有一清楚的了解，本篇先从常用的 ROM 电路开始而后逐步介绍各种高级逻辑器件，直至近年才推出的 EPLD 器件。

为了加深认识，在每一章中都举出了一些应用实例。这些例题是下篇中采用计算机辅助设计的基础知识，应该掌握。第 3 章与第 4 章是这篇的重点，了解了 PAL 与 GAL 器件，不仅对当前常用的 PLD 器件有了认识，而且再进一步学习高级逻辑器件也就没有什么困难了。

由于 PLD 器件，尤其是高级逻辑器件若采用手工编程是不可能的事情，即使采用计算机辅助设计由编程器编程还需有一个方法。在本篇的第 6 章介绍了如何采用真值表、逻辑表达式以及状态图(状态机)方法来设计 PLD 器件。所介绍的这些设计方法对了解下篇的内容无疑起到了承上启下的作用。

第 1 章　只读存储器 ROM

1.1　概　　述

1.1.1　ROM 的分类

ROM (Read Only Memory)是存放固定信息的存储器，信息是在制造或编程时写入，使用时只能读出不能更改。ROM 的应用很广泛，除了用作存储器以外，还可以用来实现逻辑函数，配合其它电路产生时序控制信号等。

ROM 的种类很多，按使用的器件类型分有二极管 ROM、双极型三极管 ROM 和单极型 MOS 管 ROM；按数据的写入方式又可分为固定 ROM、可编程 ROM 即 PROM、可擦可编程 ROM 即 EPROM 以及电可擦可编程 ROM 即 EEPROM。固定 ROM 所存信息是出厂前由厂方通过掩膜技术已经完全固定下来，使用过程中无法再更改，这种 ROM 灵活性差，但是生产成本低，可靠性高，主要用于已经定型的批量生产的产品中。PROM 的信息内容是由用户自己根据需要编程写入，但是，只能写入一次，一经写入也不能再修改了。EPROM 的信息内容可以多次编程改写，当需要改写时，用紫外线灯照射就可以擦除原来所存储的内容，通过编程再写入新的信息内容。目前市售各种编程器都可用于 EPROM 的编程。由于开发设备齐全，改写方便灵活，目前 EPROM 应用很广泛。EEPROM 除了具有 EPROM 的性能外，其主要优点是擦除所存信息

时，不用紫外线灯照射，也不用外加编程电源 V_{PP}，自身内部带有编程电压发生器，可以使用单一的 +5V 电源擦除和写入，使用很方便，因而受到人们的重视。

1.1.2 ROM 的构造原理

1. 标准与或表达式

在逻辑电路的设计中经常用逻辑表达式的形式描述一个逻辑关系（即逻辑函数）。由数字逻辑电路的基本知识可知，同一个逻辑关系可以有多种不同的逻辑表达式，例如，由三个变量组成的逻辑关系可以写成以下各种不同的逻辑表达式：

$F = AB + \overline{A}C$ 与或表达式

$= (A + C)(\overline{A} + B)$ 或与表达式

$= \overline{\overline{AB}\ \overline{AC}}$ 与非－与非表达式

$= \overline{(\overline{A} + \overline{B})(A + \overline{C})}$ 或－与非表达式

$= \overline{\overline{A + C} + \overline{\overline{A} + B}}$ 或非－或非表达式

$= \overline{A\overline{B} + \overline{A}\,\overline{C}}$ 与－或非表达式

上式还可以继续推导成其它不同形式的逻辑表达式。这种性质我们叫做逻辑表达式的多样性。逻辑表达式的多样性使得在设计逻辑电路时，可以采用灵活多样的逻辑电路来实现同一个逻辑关系，给设计带来一定方便。同时，也可以将各种表达式用不同的方法化到最简，用最少的逻辑元件来实现该逻辑关系。

在实际应用中，用基本逻辑门电路来实现某个逻辑关系时，才能体会到这种优点。当逻辑关系很复杂，输入端和输出端较多时，表达式的多样性给设计带来一定麻烦。因为较复杂的逻辑关系，尤其是存储大量信息时一般不是由基本逻辑门来实现，而是由集成度较高的电路来实现，因此，希望电路标准化，易生产。

从上面的例子可以看出，如果把每个输入变量的正变量和反变量都看作是已知的，则与或表达式和或与表达式只包含两种逻辑关系，即“与”和“或”。而其它表达式还多了“非”的逻辑关系。即在制作高集成度的逻辑电路时与或表达式和或与表达式所表示的逻辑关系有着优越性。

上面例子中的与或表达式还可以写成用最小项表示的形式：

$$F = AB + \overline{A}C$$
$$= AB\overline{C} + ABC + \overline{A}BC + \overline{A}\,\overline{B}C$$

这里引入了“最小项”的概念。当一个逻辑函数有 n 个逻辑变量时，可以有 2^n 个乘积项，每个乘积项中不同的变量以其正变量或反变量的形式必须出现一次，而且只出现一次。这样的乘积项叫做最小项。比如用三个变量 A、B、C 来描述一个逻辑函数时，最小项最多有八个，即 $\overline{A}\,\overline{B}\,\overline{C}$、$\overline{A}\,\overline{B}C$、$\overline{A}B\overline{C}$、$\overline{A}BC$、$A\overline{B}\,\overline{C}$、$A\overline{B}C$、$AB\overline{C}$、$ABC$。当变量的个数确定时，通常用符号 m_i 来表示最小项，其中 i 代表最小项的编号，因为每一个最小项仅和一组变量取值相对应，只有该组取值为 1，其余取值为 0。即 $\overline{A}\,\overline{B}\,\overline{C}$ 可以写为 m_0，$\overline{A}\,\overline{B}C$ 可以写为 m_1，以下类推。

我们知道，最小项有一个重要性质，n 个变量的全体最小项之和恒为 1，即：

$$\sum_{i=0}^{2^n-1} m_i^n = 1$$

或

$$\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}C + AB\overline{C} + ABC = 1$$

全体最小项之和恒为 1，从逻辑概念上讲，是指用 n 个变量表示的任一个逻辑事物，用全体最小项之和表示时，该事物必定发生。也就是说，任一个逻辑事物都可以用最小项之和的形式来表示，逻辑关系不同，最小项的个数不同，最多是全体之和。如果有一个全体最小项之和的电路，就可以设计任意的逻辑函数。比如：

$$\begin{aligned} F &= AB + \overline{A}C \\ &= AB\overline{C} + ABC + \overline{A}BC + \overline{A}\,\overline{B}C \\ &= m_1 + m_3 + m_6 + m_7 \end{aligned}$$

上式是由四个最小项构成，如果用全体最小项之和的电路来实现，只要将其余四个最小项去掉就可以了。

用最小项之和表示的逻辑表达式，叫做“标准与或表达式”，这种表达式的形式是唯一的，一个逻辑函数只有一个最小项之和的表达式，不同的最小项之和所表示的逻辑函数是不同的。

用标准与或表达式来描述逻辑函数时，看起来比较复杂，但是实现标准与或表达式的元件在结构上就简单多了，各个最小项都是输入端的相与，又叫乘积项，输出则是最小项相或，又叫和数项。ROM 电路的结构就是一个全体最小项之和的逻辑阵列，因此，用 ROM 电路可以设计任意逻辑函数。比如有 n 个输入变量，m 个输出变量的 ROM 有 2^n个最小项(决定了与阵列的大小)，每个输出最多有 2^n个最小项相或，m 个输出共有 $2^n \times m$ 个最小项相或(决定了或阵列的大小)。

用真值表设计逻辑电路时，标准与或表达式和真值表有着一一对应的关系，直读性很强。比如有一组逻辑函数：

$$F1 = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + ABC$$
$$F2 = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$$
$$F3 = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + \overline{A}BC + ABC$$
$$F4 = \overline{A}B\overline{C} + A\overline{B}\,\overline{C}$$

列真值表如表 1-1 所示。

表 1-1　由式列真值表

A	B	C	F1	F2	F3	F4
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	1	1
1	0	1	0	0	0	0
1	1	0	0	1	0	0
1	1	1	1	0	1	0

用二极管 ROM 来实现时结构图如图 1-1 所示。A、B、C 为三个输入变量，亦称地址码的输入端；F1、F2、F3、F4 为函数输出端，亦称数据输出端，或称位线。三个输入变量，与阵列有 $2^3 = 8$ 列，用 W0 ~ W7 表示，称为字线，在 ROM 中，为了得到所有的最小项，与阵列的结构是固定不变的，实际上是一个译码器，而或阵列则随着逻辑函数的不同是可以编程的。因此 ROM 的存

储容量是按或阵列的大小而确定的。图 1-1 所示的二极管 ROM 电路，容量为 $2^n \times 4$ 或者 8×4 个存储单元，每个单元存放一位二值代码，接入二极管表示存 1，未接二极管表示存 0。

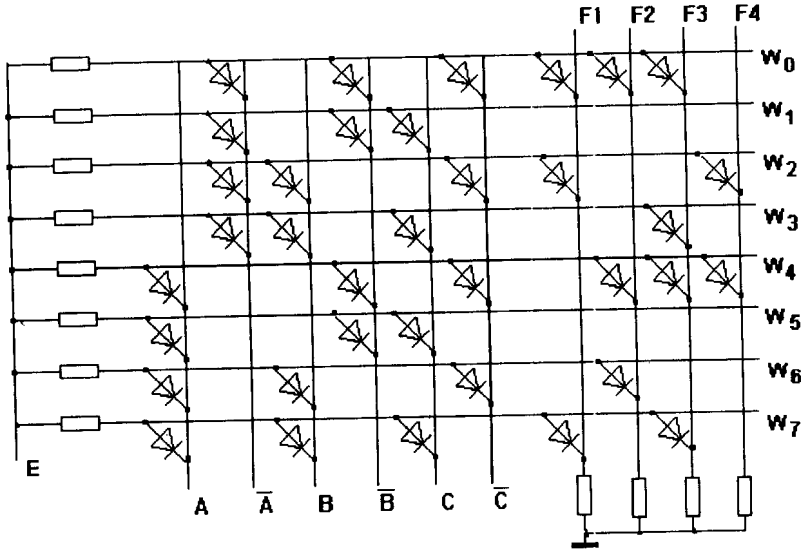


图 1-1　二极管 ROM 结构图

2. 标准或与表达式

在数字逻辑基本知识中，还经常用到最大项的概念，用最大项实现逻辑函数时，为标准或与表达式，它和相应的标准与或表达式是等效的。因此在 ROM 中经常采用与或阵列结构，而较少采用或与阵列结构。

1.1.3　逻辑图表示法

数字逻辑电路的逻辑图，科技文献中一般采用两种表示方法，即国内传统画法，如图 1-2 (a) 所示，国外传统画法，如图 1-2(b) 所示。

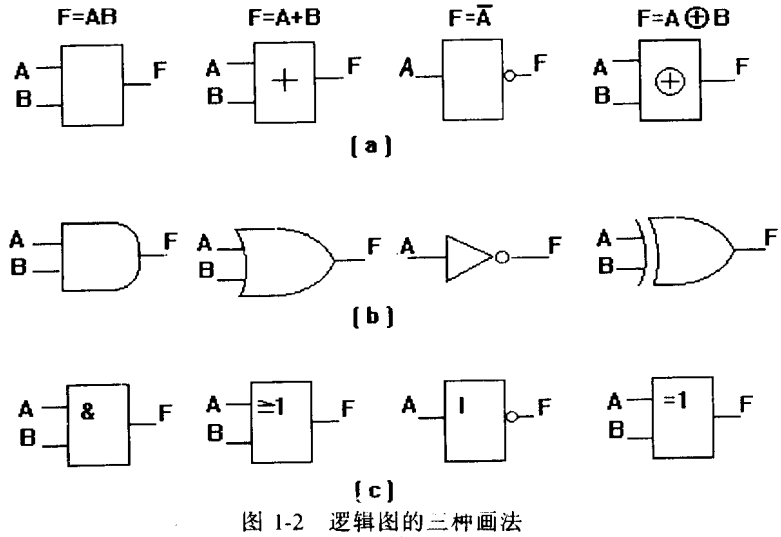


图 1-2　逻辑图的三种画法

1987 年国家标准局发出通知，要求所有电气技术文献和图纸一律使用新的国家标准，如图 1-2(c) 所示，它有利于计算机制图。本书所涉及的 ROM 以及各种 PLD 器件，内部电路庞

大，结构紧凑，用上述画法都很难描述清楚，所以采用下述易读易画的简化表示法，且已为厂家和用户所接受。

1. 与门和或门电路如图 1-3 所示，电路简单时也可以采用传统画法，非门电路仍采用传统画法。

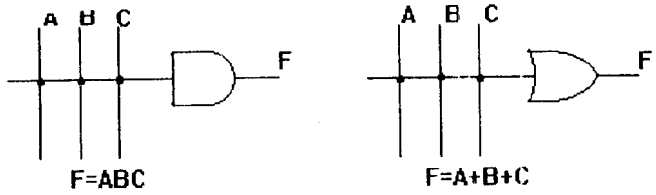
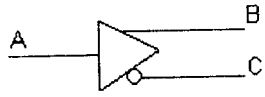


图 1-3　与或门电路画法

2. 输入缓冲器和反馈缓冲器都采用互补输出结构，其表示方法如图 1-4 所示，逻辑关系为 $B=A$、$C=\overline{A}$。变量 B 是变量 A 的“真”，变量 C 是变量 A 的“假”或“非”。



A	B	C
0	0	1
1	1	0

图 1-4　缓冲器画法

3. 阵列各交叉点的连接方式，分为三种情况，如图 1-5 所示。交叉点有实心黑点时为固定连接，不能编程；交叉点为 × 时，表示该单元经过编程后已经由门电路连接在一起；第三种情况表示该单元经过编程后已被擦除，即断开状态。在表示一个完整的没编程的 PLD 器件时，各存储单元都按此画法，表示各存储单元处于未编程的“断开”状态。

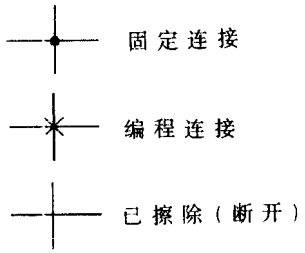


图 1-5　交叉点的连接方式

4. 编程与门的表示方法，如图 1-6 所示。输出端为 D 的与门，其输入端已被全部编程连接，所以 $D=A\overline{A}\,B\overline{B}=0$。当输入端较多时，可用 E 的方法表示，即 $E=A\overline{A}\,B\overline{B}=0$。而 F 的输入端全没被编程，仍处于断开状态，所以 $F=1$。同理，$Z=\overline{B}\,B$。表 1-2 列出了各逻辑关系。

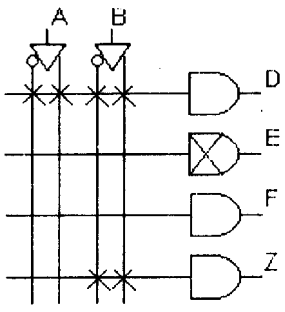


图 1-6　编程与门的表示方法

用上述逻辑图表示法，很容易将图 1-1 二极管 ROM 电路画出来，如图 1-7 所示。其中与阵列是固定的，不用编程，或阵列是经过编程连接的。

表 1-2　与门的输出状态

A	B	D	E	F	Z
0	0	0	0	1	0
0	1	0	0	1	0
1	0	0	0	1	0
1	1	0	0	1	0

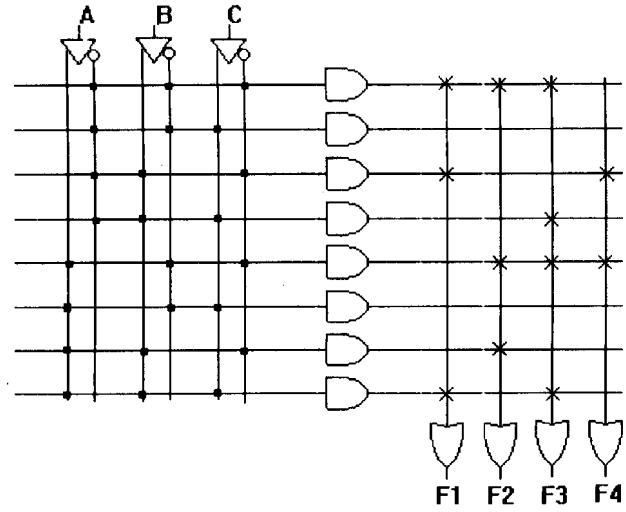


图 1-7　ROM 的简单画法

1.2　PROM

1.2.1　PROM 的结构

PROM 在出厂时，存储单元处于全连接或全断开状态。分别表示逻辑 1 或 0。使用时用户可以根据需要，通过编程将某些单元断开或连接上，得到预定的逻辑 0 或 1。但是只能改变一次，不能反复修改。

图 1-8 是采用双极型三极管加熔断丝连接的一种典型 PROM 结构图。出厂时所有字线和位线交叉点上都制作了三极管，在每个三极管的发射极上都有快速熔断丝与位线相接。实际上字线和位线之间只是发射结的连接，相当于一个二极管，所以每个存储单元都处于逻辑 1 状态。在写入数据时，只要设法把要存入 0 的那些单元的熔断丝切断，使三极管的发射极不起作用就可以了。

具体步骤是，首先要找到要写入 0 的单元地址，并且输入相应的地址代码，使相应的字线输出高电平，然后在相应的位线上加入所规定的高电压脉冲，使稳压管 D_Z 反向击穿导通，写入放大器 A_W 此时输入端为高电平，输出呈低电平低内阻状态。所选中的要存 0 的单元处的三极管，基极为高电平，而发射极为低电平状态。这样，有较大的脉冲电流流过发射极的熔断丝，将其熔断，使之变为存 0 状态。在该位线上的读出放大器 A_R，在正常输出高电平时，不足以使 D_Z 导通，所以 A_W 不工作。

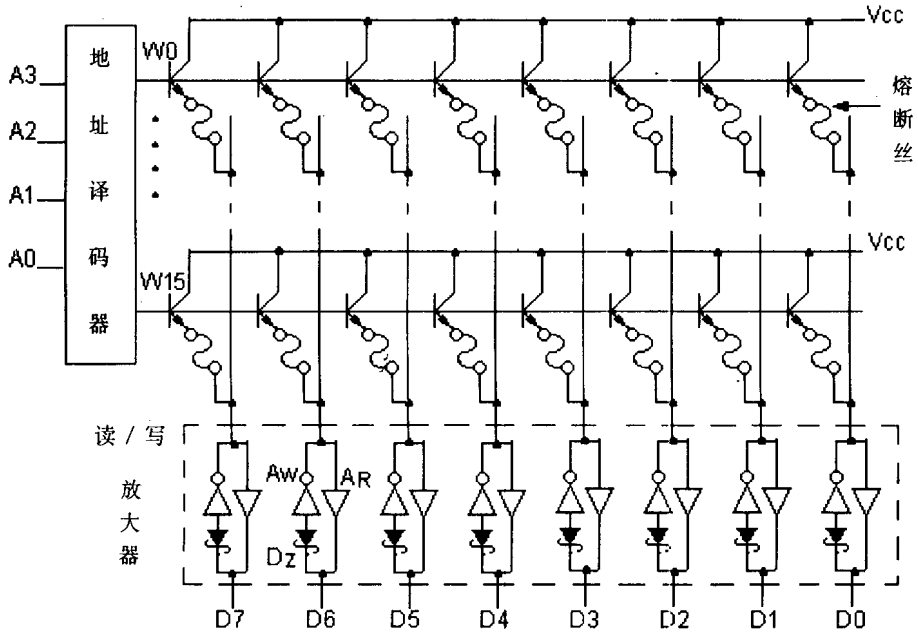


图 1-8　双极型 PROM 结构

也有的 PROM,不用熔断丝连接三极管的发射极,而是采用肖特基二极管代替熔断丝。出厂时肖特基二极管全处于反向截止状态,存储单元全为逻辑 0 状态。若将某单元改写成逻辑 1,可在肖特基二极管上加上适当的反向电压,使其击穿造成永久性短路。

这个过程就是设计人员对 PROM 编程过程。如果用手工操作,当 PROM 存储容量很大时,是很麻烦的事情,并且在操作过程中也容易出现差错。目前市售通用编程器,一般都支持各厂家生产的 PROM 电路。采用适当的编程软件,用编程器编程,给应用 PROM 的设计人员带来很大方便。

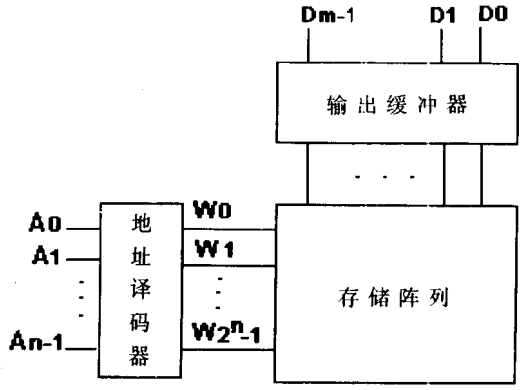


图 1-9　单译码器的 PROM 结构图

PROM 电路一般由地址译码器、存储阵列以及输出缓冲器三部分组成,如图 1-9 所示。一般 PROM 存储容量都很大,地址输入经过译码后产生的字线根数很多。比如,地址输入为 11 位时,字线根数是 2^{11},如果仍然采用图 1-9 的结构,由于内部连线过多而显得繁琐杂乱。为了克服这个问题,通常把地址译码器分为行地址译码器和列地址译码器。图 1-10 是输出 1 位、地址为 11 位的 PROM 结构图。

A0 ~ A3 为列地址输入端,产生列字线 $2^4 = 16$ 根;A4 ~ A10 为行地址输入端,产生行字线

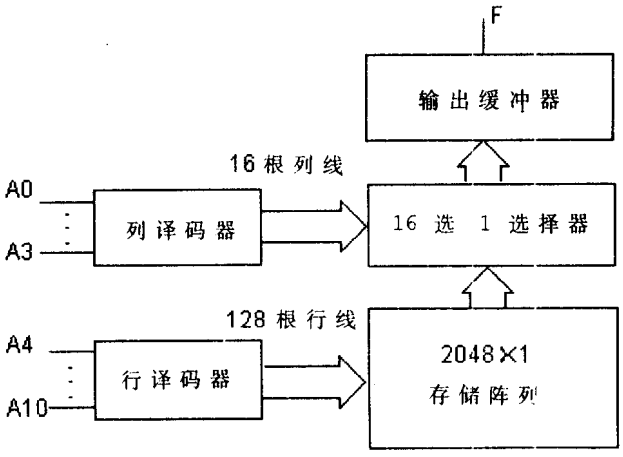


图 1-10 两个译码器的 PROM 结构图

$2^7 = 128$ 根，每一行都对应接通 16 个列字线，经过一个 16 选 1 多路选择器选通，得到输出 F。这种结构，比用一个译码器产生的字线要少得多，节省了芯片空间，相对加大了存储阵列的容量。缓冲器用来提高存储器的带负载能力，通常由三态门 TS 或集电极开路的门 OC 构成，也有的为了与时序逻辑电路连接方便，输出端带有触发器。

1.2.2 PROM 举例

PROM 电路类型很多，各厂家生产的产品型号也不完全相同，在应用中主要注意以下几个问题：一是存储容量，PROM 的存储容量如前所述，是按字线与位线之积来计算的，而字线又取决于地址码的位数，PROM 一般有 4～16 位地址码，所以每一位线的存储容量在 $2^4 \sim 2^{16}$ 之间；而位线一般有 1 位、4 位以及 8 位 等几种规格。所以大容量的 PROM 其存储容量为 $2^{16} \times 8$，或写作 65536×8。二是读取速度，PROM 一般都是双极型 TTL 电路，速度较快，在 25ns～100ns 之间，可以根据需要来选择，如果需要读取速度更高的，可以选择发射极耦合的 ECL 电路，其速度在 15ns～25ns 之间。三是输出方式，可以根据不同的需要，选择 TS 结构或者 OC 结构，带有触发器的 PROM 其输出端也往往是 TS 结构。下面列举两个具体型号供参考。

图 1-11 为 Am27S43 方框图，存储容量是 4096×8，双极型 PROM，TS 输出方式，标准读取速度是 55ns，Am27S43A 型速度是 40ns，每个存储单元都用硅化铂熔断丝连接并存入逻辑 0，通过在电路上加适当电压，就可有选择地编程为逻辑 1，A0～A11 为地址码输入端，片选输入端有两个，$\overline{CS1}$ 低电位有效，CS2 高电位有效。当 $\overline{CS1} = 0$、$CS2 = 1$ 时，数据输出端 Q0～Q7 有输出。当 $\overline{CS1} = 1$ 或 $CS2 = 0$ 时，Q0～Q7 为高阻状态。图 1-12 是其引脚图和逻辑图。芯片封装分两种形式，一是双列直插式 DIP，二是片式扁平封装 Chip-Pak。

图 1-13 为 53/63RA481 方框图，是带有 D 型触发器的 PROM，数据在时钟脉冲 CLK 为上升沿时被传输到触发器中，$\overline{E}$ 为异步片选输入端，$\overline{ES}$ 为同步片选输入端，当 $\overline{E} = 0$、$\overline{ES} = 0$ 时，则数据输出端 Q0～Q7 有输出。当 $\overline{E} = 1$ 或者在时钟脉冲 CLK 为上升沿时 $\overline{ES} = 1$，都可以使输出端处于高阻状态。由于这样设置，使得存储器的扩展和数据控制都是比较灵活的。$\overline{PR}$ 为触发器的置位端，$\overline{PR} = 0$ 时，触发器的输出均为逻辑 1。$\overline{CLR}$ 为清零端，$\overline{CLR} = 0$ 时，触发器的输出均为逻辑 0。图 1-14 是其引脚图和逻辑图。

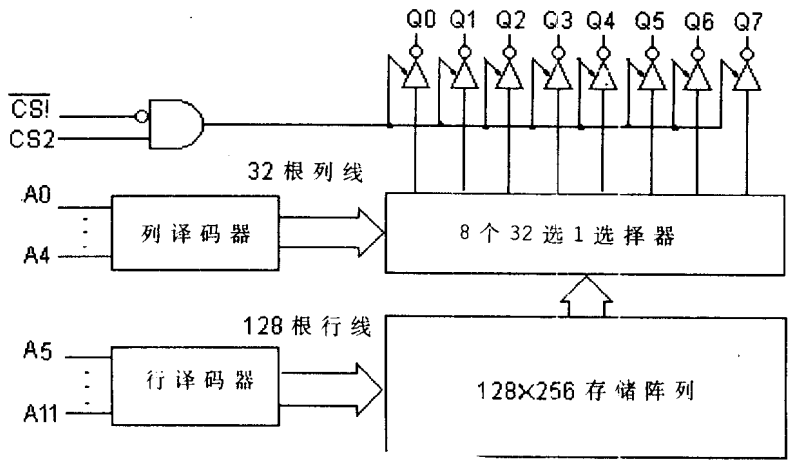
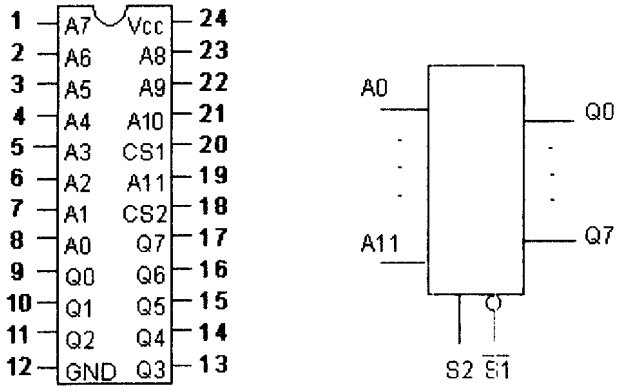


图 1-11　Am27S43 方框图



（引脚图）　　（逻辑图）

图 1-12　Am27S43 引脚图和逻辑图

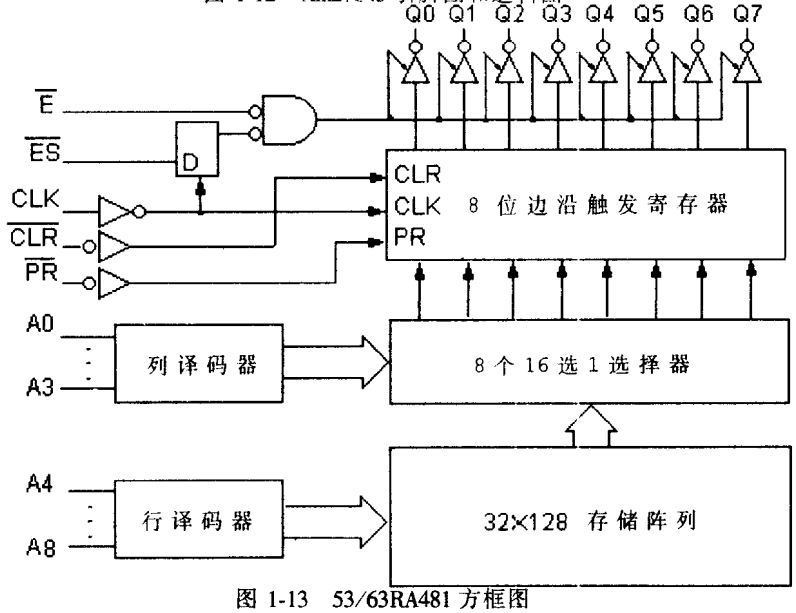


图 1-13　53/63RA481 方框图