第八章　可编程逻辑器件

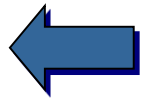
主讲教师：何云峰

本章知识要点

- PLD的基本概念
- 常用PLD及其在逻辑设计中的应用
- ISP技术简介

提 纲

1. PLD概述
2. 低密度可编程逻辑器件
3. 高密度可编程逻辑器件
4. 在系统编程技术简介



PLD概述

- 数字系统中常用的大规模集成电路可分为三大类
  - 非用户定制电路(Noncustom design IC)
  - 全用户定制电路(Fullcustom design IC)
  - 半用户定制电路(Semicustom design IC)
- 可编程逻辑器件(Programmable Logic Device，简称PLD)属于半用户定制电路
  - PLD结构灵活、性能优越、设计简单
  - PLD特别适宜于构造小批量生产的系统

PLD的发展

- 70年代初期
  - 可编程只读存储器(PROM)：由一个“与”阵列和一个“或”阵列组成，**“与”阵列是固定的，“或”阵列是可编程的**
- 70年代中期
  - 可编程逻辑阵列(PLA)：由一个“与”阵列和一个“或”阵列组成，其**“与”阵列和“或”阵列都是可编程的**

PLD的发展

- 70年代末期
  - 可编程阵列逻辑(PAL)：**PAL器件的“与”阵列是可编程的，而“或”阵列是固定的**
- 80年代中期
  - 通用阵列逻辑（GAL）
  - 复杂可编程逻辑器件（CPLD）
  - 现场可编程门阵列（FPGA）

PLD的发展

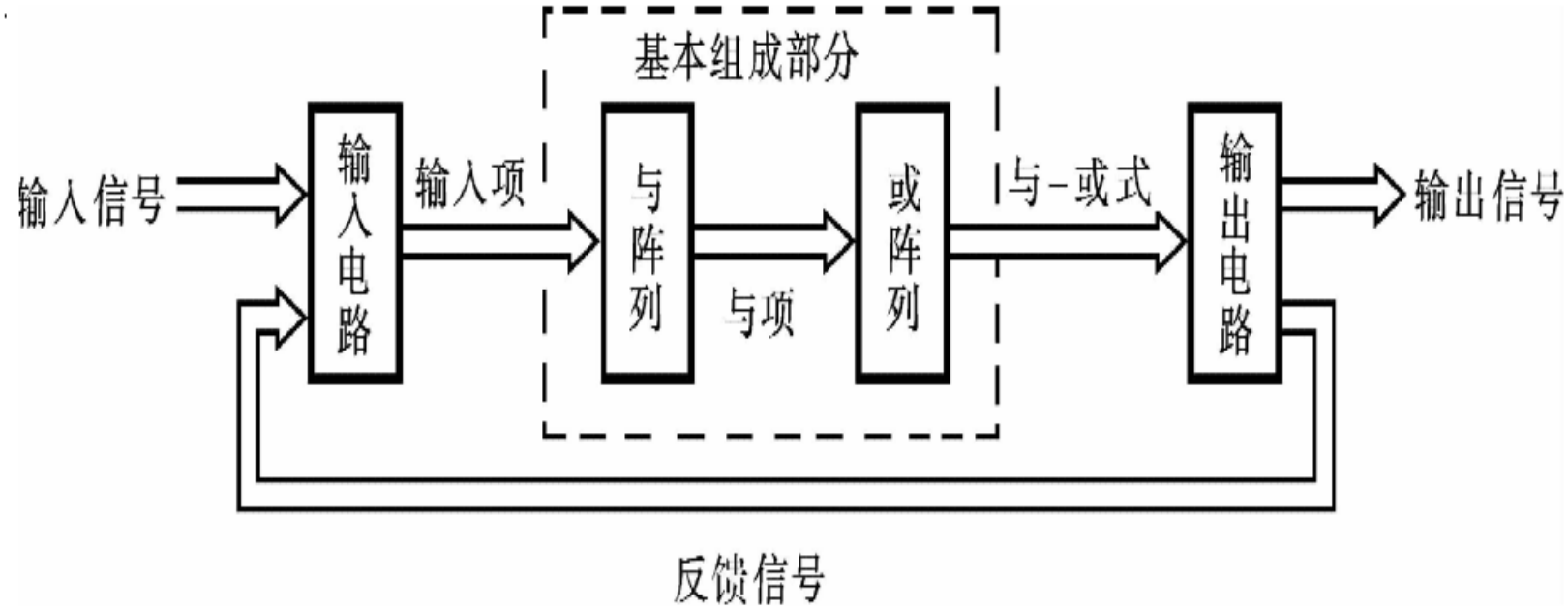
- 90年代
  - 在系统编程(ISP)器件：指用户具有在自己设计的目标系统中或线路板上为重构逻辑而对逻辑器件进行编程或反复改写的能力
- PLD从根本上改变了系统设计方法，使各种逻辑功能的实现变得灵活、方便

PLD的分类

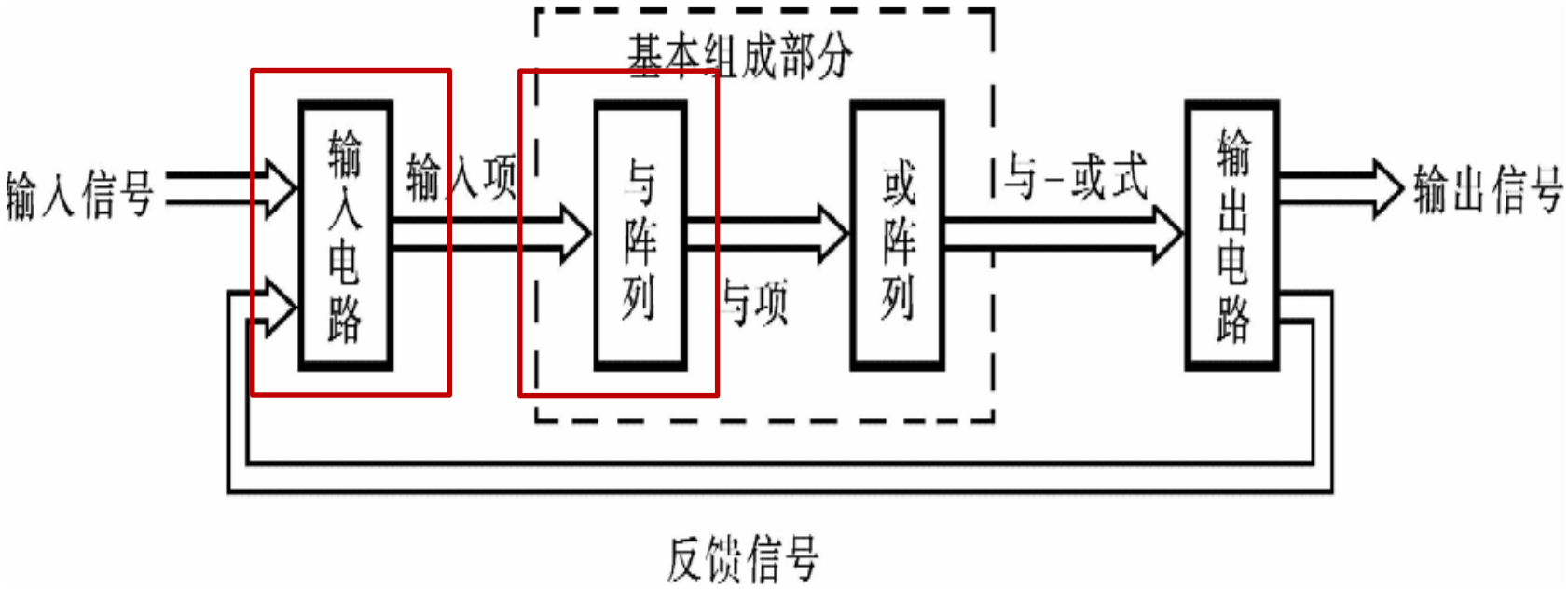
- 据集成度，通常将PLD分为
  - 低密度可编程逻辑器件（LDPLD）
    - 集成度小于1000门的可编程逻辑器件
    - 例如PROM，PLA，PAL和GAL
  - 高密度可编程逻辑器件（HDPLD）
    - 集成度达到1000门以上的可编程逻辑器件
    - 例如CPLD和FPGA等

PLD的结构

- PLD的一般结构由输入电路、与阵列、或阵列和输出电路所组成，输出是输入的与－或函数

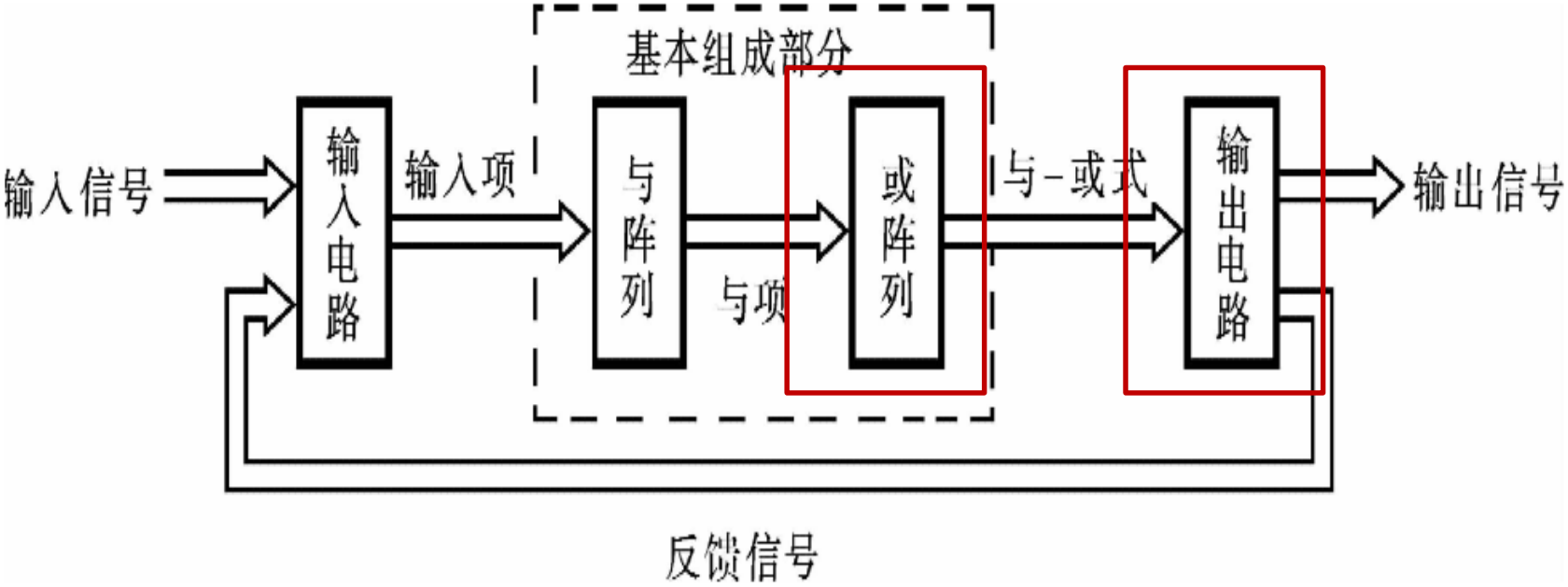


PLD的结构



- 输入电路起缓冲作用，并形成互补的输入信号送到与阵列
- 与阵列接收互补的输入信号，产生所需的与项作为或阵列的输入

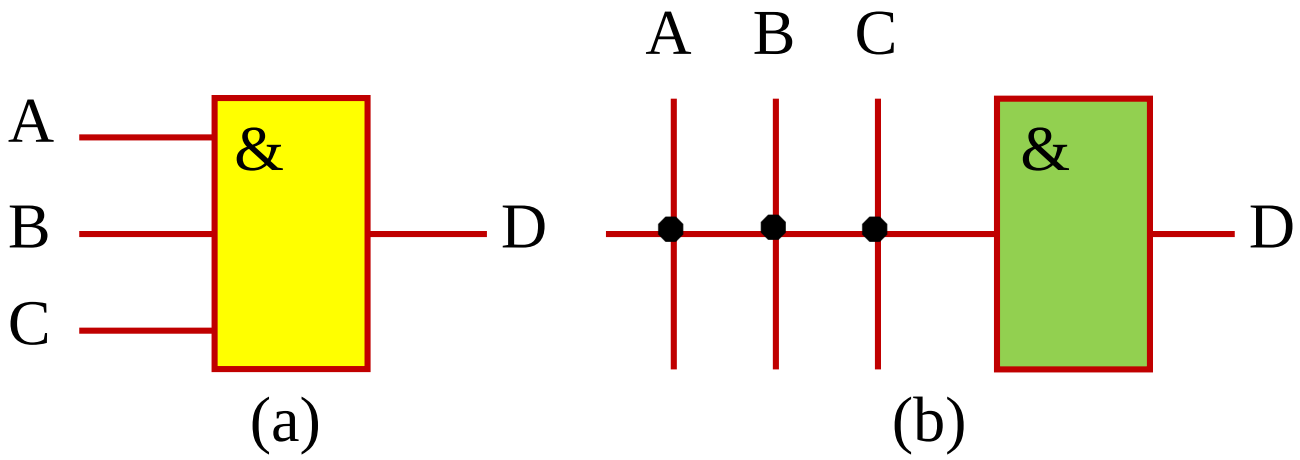
PLD的结构



- 或阵列产生输入变量的与-或函数表达式
- 输出电路既有缓冲作用，又提供不同的输出结构，如输出寄存器、内部反馈、输出宏单元等
- **与阵列和或阵列**是PLD的基本组成部分

PLD的电路表示法

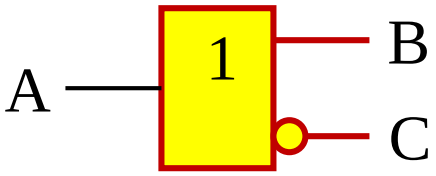
- 与门和或门
  - 3输入与门的两种表示法



PLD的电路表示法

输入缓冲器

- 典型输入缓冲器的PLD表示法如图所示
- 两个输出B、C是其输入A的原和反（见图中真值表）



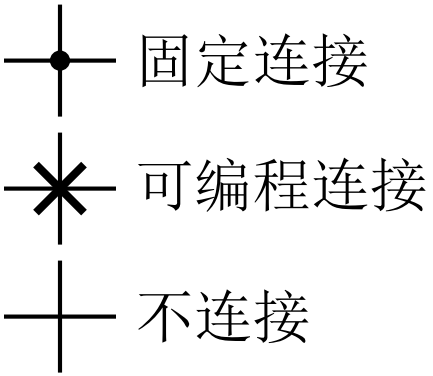
A	B	C
0	0	1
1	1	0

$$B = A$$
$$C = \overline{A}$$

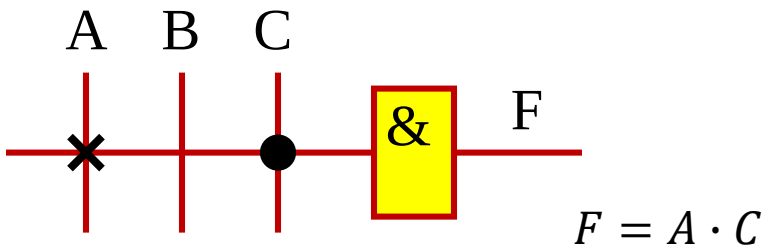
PLD的电路表示法

连接方式

- 实点“·”表示硬线连接，即固定连接
- “×”表示可编程连接
- 没有“×” 和“·”的表示两线不连接



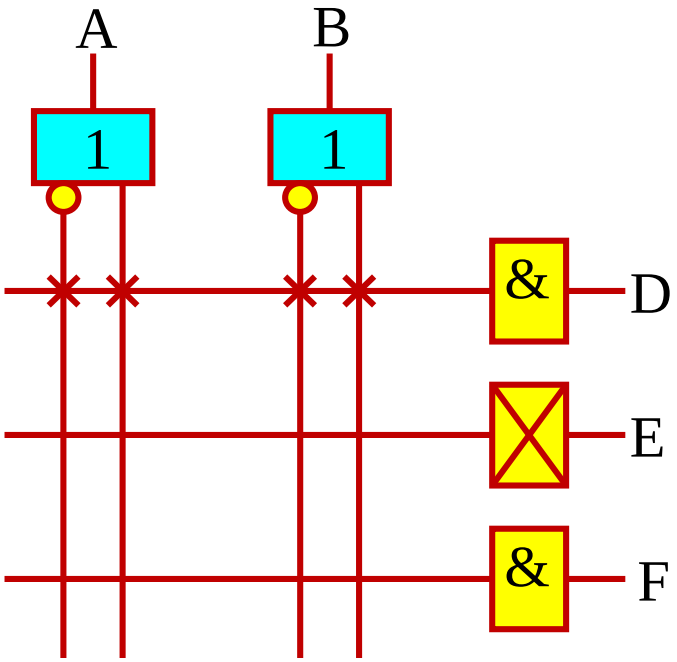
(a)



(b)

PLD的电路表示法

- 与门不执行任何功能时的连接表示

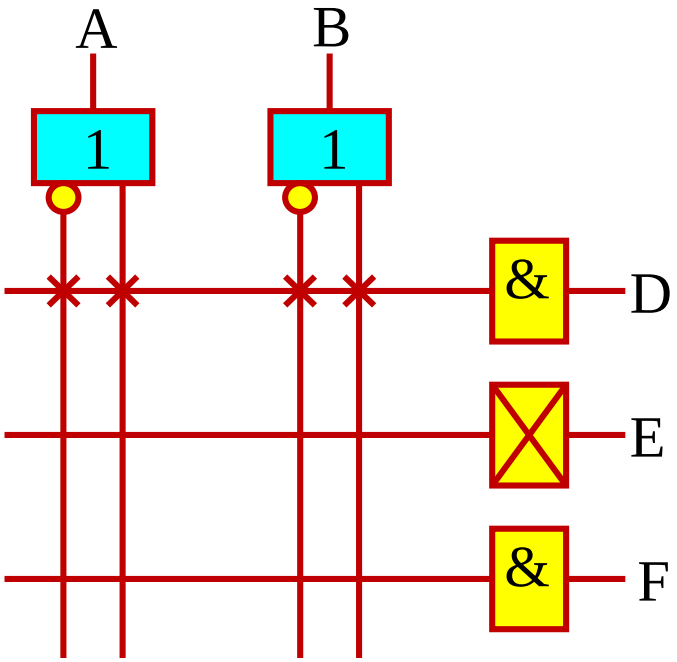


A	B	D	E	F
0	0	0	0	1
0	1	0	0	1
1	0	0	0	1
1	1	0	0	1

– 图中，输出为D的与门连接了所有的输入项，其输出方程为$D = A \cdot \bar{A} \cdot B \cdot \bar{B} = 0$

PLD的电路表示法

- 与门不执行任何功能时的连接表示

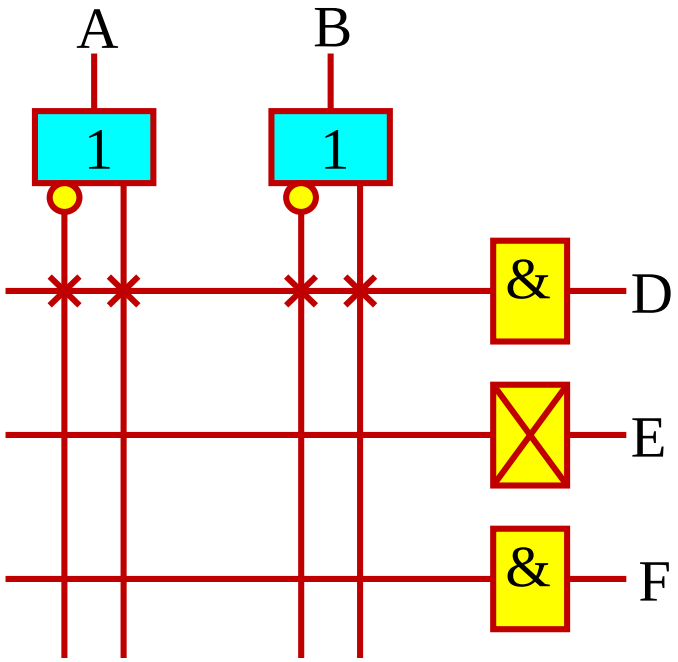


A	B	D	E	F
0	0	0	0	1
0	1	0	0	1
1	0	0	0	1
1	1	0	0	1

– 为了方便起见，用标有“×”标记的与门输出来表示所有输入缓冲器输出全部连到某一“与”项的情况，如图中输出E

PLD的电路表示法

- 与门不执行任何功能时的连接表示

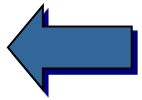


A	B	D	E	F
0	0	0	0	1
0	1	0	0	1
1	0	0	0	1
1	1	0	0	1

– 输出F表示无任何输入项与其相连，因此，该“与”项总是处于“浮动”的逻辑“1”

提 纲

1. PLD概述
2. 低密度可编程逻辑器件
3. 高密度可编程逻辑器件
4. 在系统编程技术简介



低密度可编程逻辑器件

- 根据PLD中阵列和输出结构的不同，目前常用的低密度PLD有4种主要类型：
  - 可编程只读存储器PROM
  - 可编程逻辑阵列PLA
  - 可编程阵列逻辑PAL
  - 通用阵列逻辑GAL

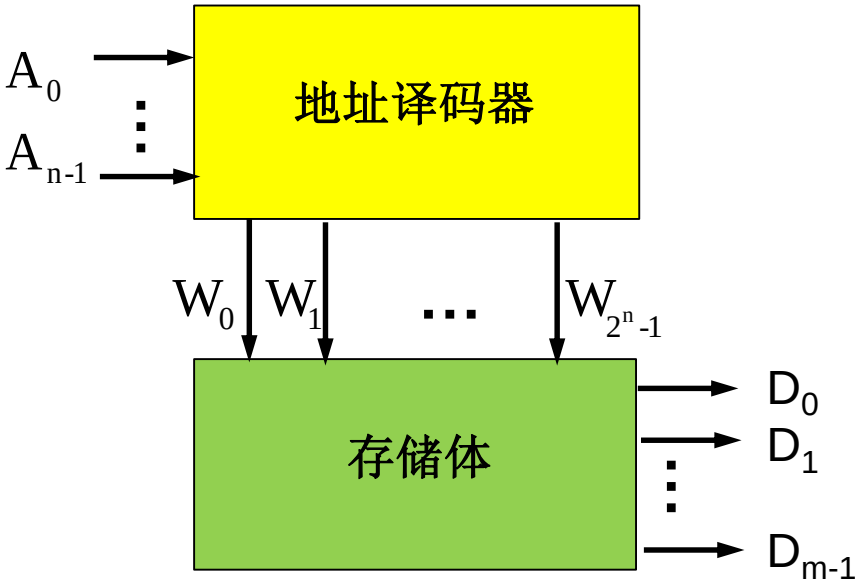
可编程只读存储器PROM

□可编程只读存储器PROM

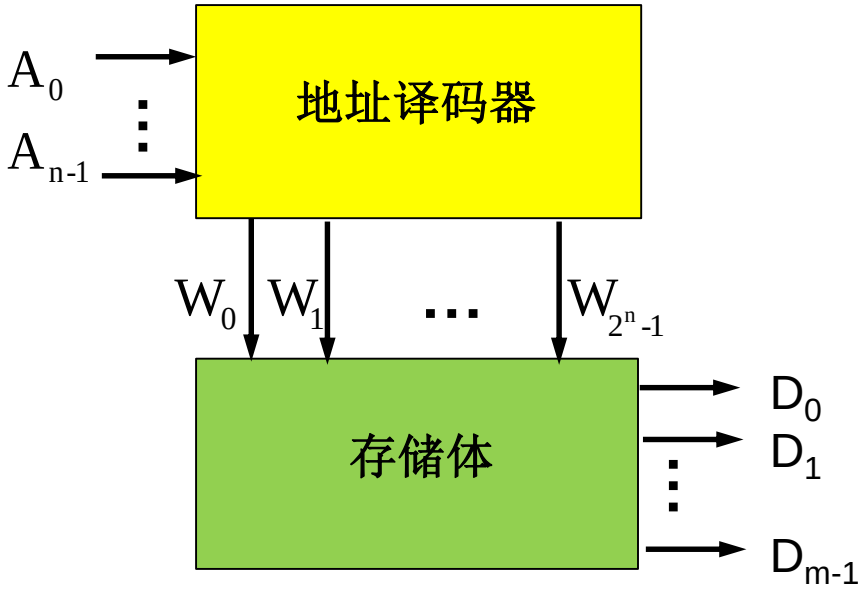
- 半导体存储器的分类
  - 随机存取存储器RAM(Random Access Memory)
    - 优点是读写方便，使用灵活
    - 缺点是一旦断电，所存储的信息便会丢失，属于易失性存储器
  - 只读存储器ROM(Read Only Memory)
    - ROM属于非易失性存储器，即使切断电源，ROM中的信息也不会丢失
    - 用户可编程ROM（简称PROM）：存放的内容是由用户根据需要在编程设备上写入的。优点是使用灵活方便，适宜于用来实现各种逻辑功能。

可编程只读存储器PROM

- PROM的逻辑结构
  - 主要由地址译码器和存储体两大部分组成

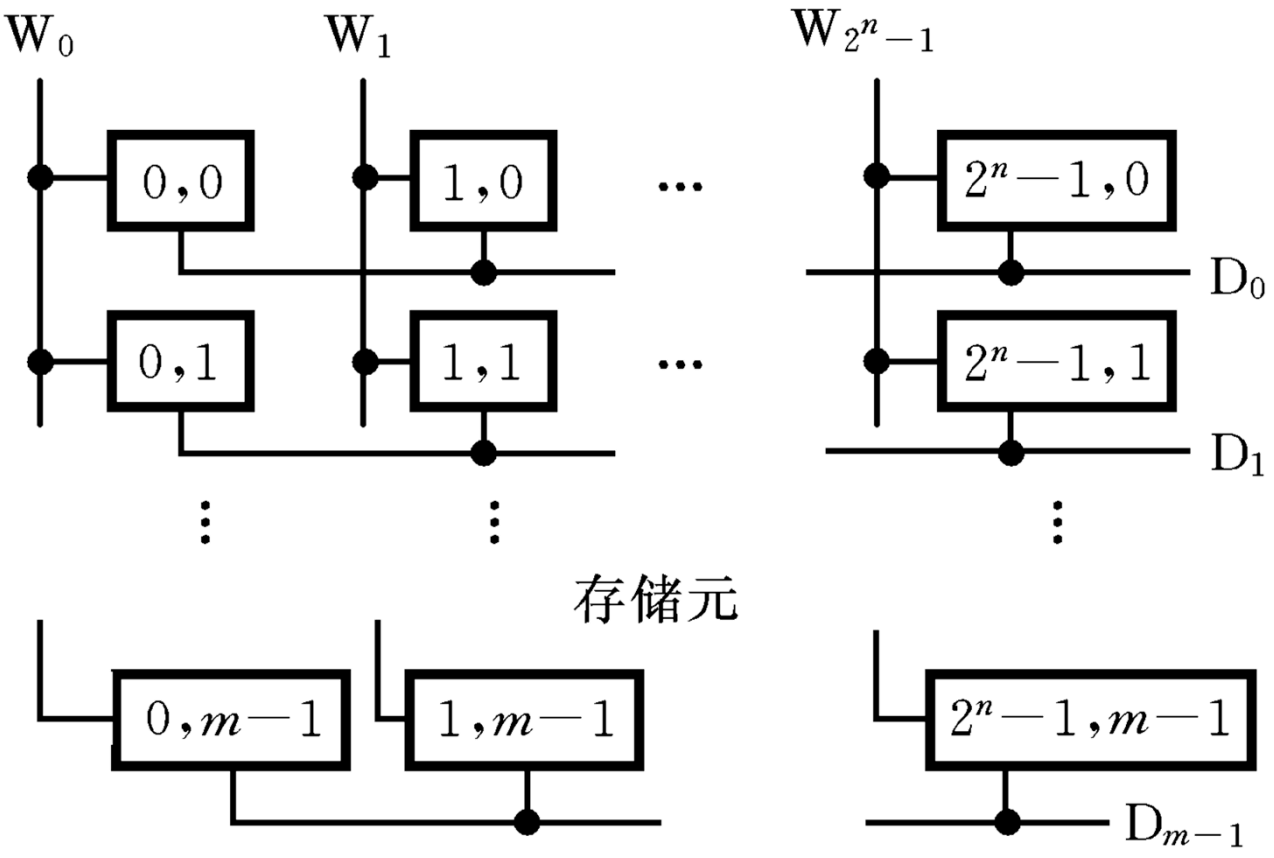


可编程只读存储器PROM



- $A_0 \sim A_{n-1}$为地址输入线
- $W_0 \sim W_{2^n-1}$为地址译码输出线，又称为字线
- $D_0 \sim D_{m-1}$为数据输出线，又称为位线
- 容量：将一个n位地址输入和m位数据输出的PROM的存储容量表示为**$2^n \times m$(位)**

可编程只读存储器PROM



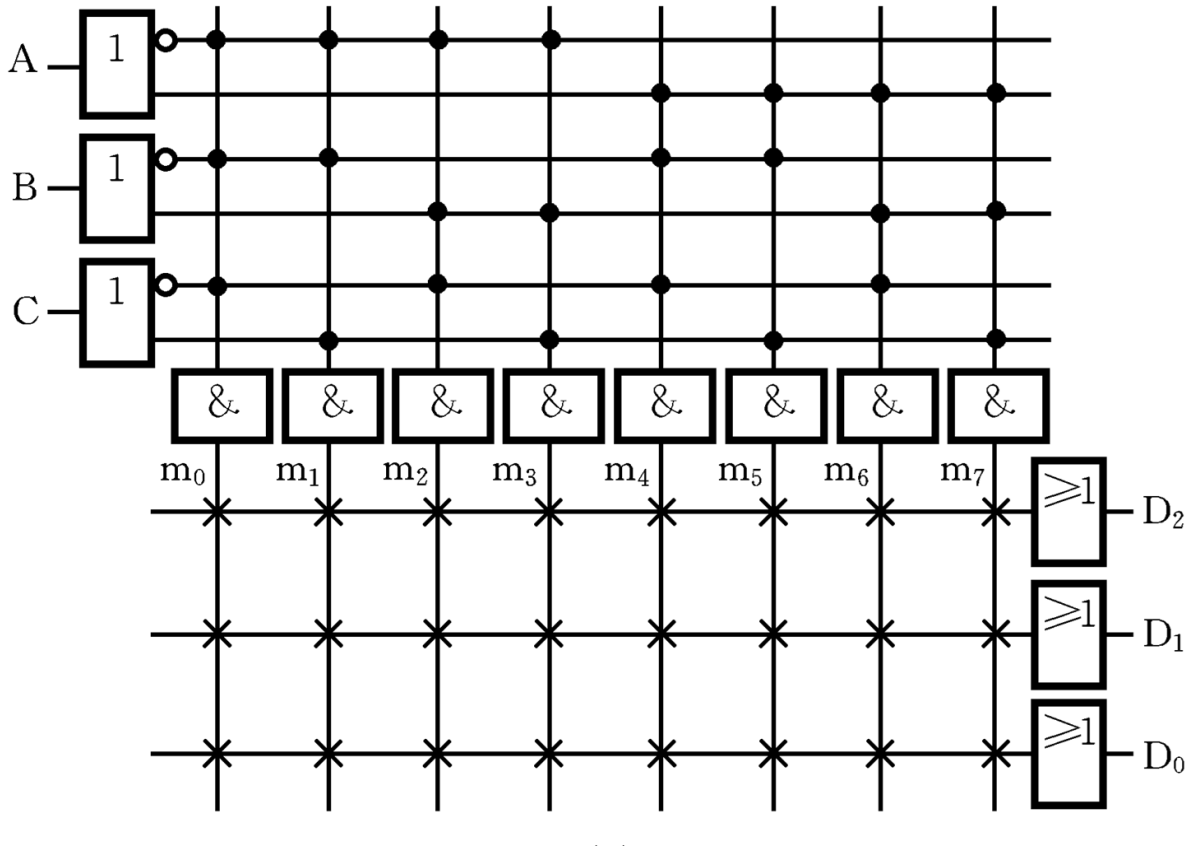
PROM存储体的结构示意图

可编程只读存储器PROM

- 存储器的角度
  - PROM由地址译码器和存储体两大部分组成
- 逻辑电路角度
  - PROM的由一个固定连接的与门阵列和一个可编程连接的或门阵列组成

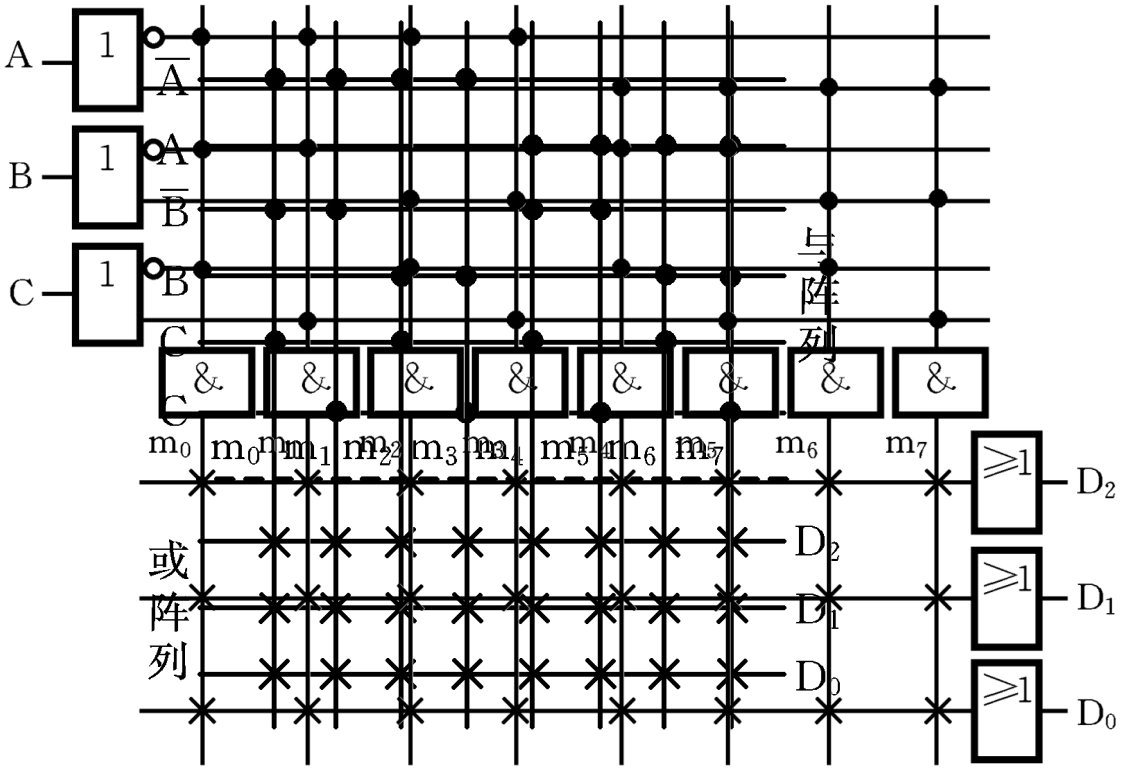
可编程只读存储器PROM

– 例如，一个8×3可编程ROM的逻辑结构图



可编程只读存储器PROM

- 为了PROM设计的方便，通常将逻辑结构图简化为阵列逻辑图，简称阵列图。
- **画阵列图时，将PROM中的每个与门和或门都简化成一根线。**



可编程只读存储器PROM

- PROM的类型
  - 一次编程的ROM(PROM)
    - 所有存储元均被加工成同一状态“0”(或“1”)，可通过编程改变某些存储元的状态
    - 编程只能进行一次，一旦编程完毕，其内容便不能再改变
  - 可抹可编程ROM(EPROM)
    - EPROM不仅可由用户编程存放指定的信息，而且可通过专用的紫外线灯照射芯片上的受光窗口，将原存储内容抹去，再写入新的内容。

可编程只读存储器PROM

- 电可抹可编程ROM(EEPROM)
  - 与EPROM相似，但擦除和编程均用电完成
  - 工作电流小、擦除速度快，而且允许改写的次数大大高于EPROM，一般允许改写1万次以上
- 快闪存储器(Flash Memory)
  - 快闪存储器是新一代用电信号擦除的可编程ROM
  - 既吸收了EPROM结构简单、编程可靠的优点，又具有EEPROM用隧道效应擦除的快速性
  - 集成度可以很高（有时将其归属于高密度可编程逻辑器件）

可编程只读存储器PROM

- PROM应用举例
  - 改变“或”阵列上连接点的数量和位置，就可以在输出端形成由输入变量“最小项之和”表示的任何一种逻辑函数
  - 设计过程
    - 根据逻辑要求列出真值表
    - 根据逻辑函数值确定对PROM“或”阵列进行编程的代码，画出相应的阵列图

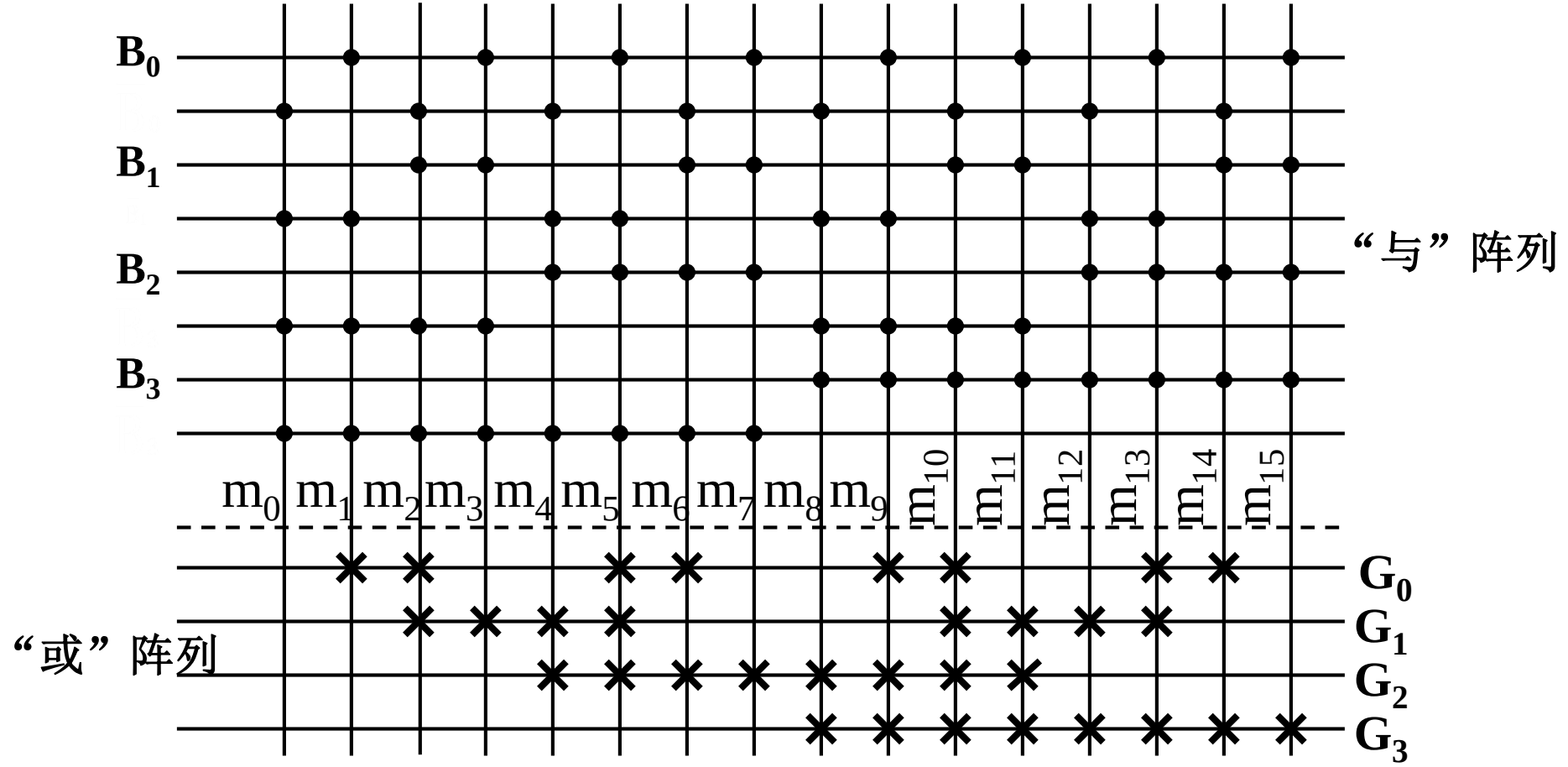
可编程只读存储器PROM

- **例1** 用PROM设计一个代码转换电路，将4位二进制码转换为Gray码。
  - **解** 设4位二进制码为$B_3B_2B_1B_0$，4位Gray码为$G_3G_2G_1G_0$，可列出真值表

二进制数	Gray码	二进制数	Gray码
$B_3\,B_2\,B_1\,B_0$	$G_3\,G_2\,G_1\,G_0$	$B_3\,B_2\,B_1\,B_0$	$G_3\,G_2\,G_1\,G_0$
0 0 0 0	0 0 0 0	1 0 0 0	1 1 0 0
0 0 0 1	0 0 0 1	1 0 0 1	1 1 0 1
0 0 1 0	0 0 1 1	1 0 1 0	1 1 1 1
0 0 1 1	0 0 1 0	1 0 1 1	1 1 1 0
0 1 0 0	0 1 1 0	1 1 0 0	1 0 1 0
0 1 0 1	0 1 1 1	1 1 0 1	1 0 1 1
0 1 1 0	0 1 0 1	1 1 1 0	1 0 0 1
0 1 1 1	0 1 0 0	1 1 1 1	1 0 0 0

可编程只读存储器PROM

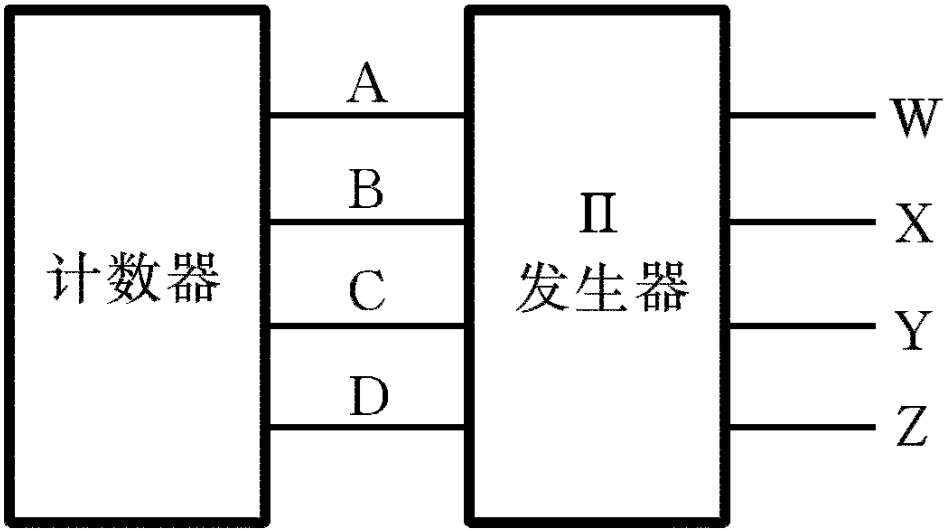
– 选容量为$2^4\times4$的PROM实现给定功能



$$G2 = \sum m(2,3,5,6,8,9,10,11,12,13,14,15)$$

可编程只读存储器PROM

- **例2** 用PROM设计一个Π发生器，输入为4 位二进制码，输出为8421码。该电路串行地产生常数π，取小数点后15位数字，即π=3.141592653589793，其逻辑框图如下图所示。



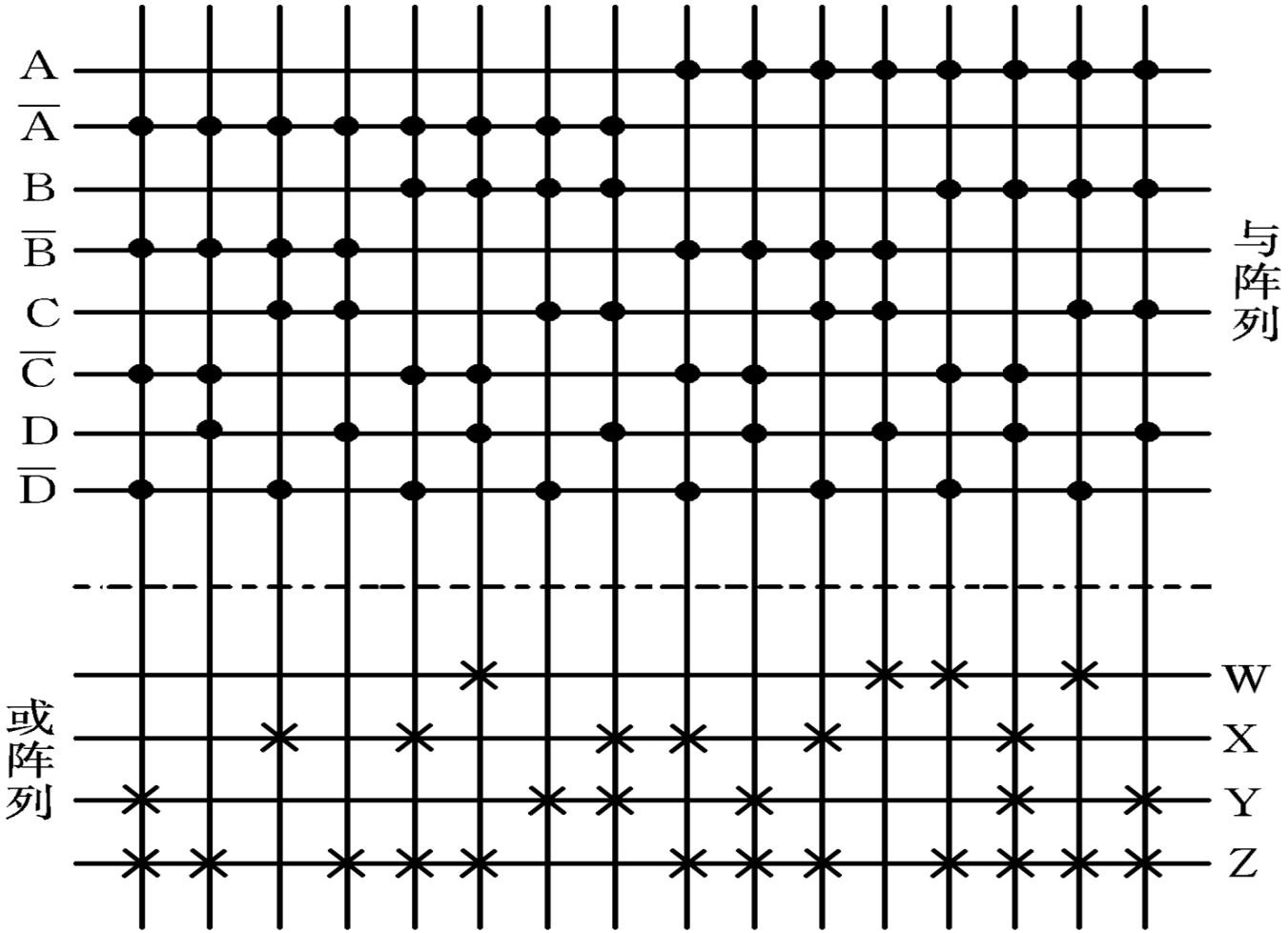
可编程只读存储器PROM

– 解: 根据题意，一个4位同步计数器控制PROM的地址输入端，顺序访问所有存储单元，存储单元中依次存放π的值，输出则为π的8421码

输 入				输 出				Π
A	B	C	D	W	X	Y	Z	
0	0	0	0	0	0	1	1	**3**
0	0	0	1	0	0	0	1	**1**
0	0	1	0	0	1	0	0	**4**
0	0	1	1	0	0	0	1	**1**
0	1	0	0	0	1	0	1	**5**
0	1	0	1	1	0	0	1	**9**
0	1	1	0	0	0	1	0	**2**
0	1	1	1	0	1	1	0	**6**

二进制数				Gray码				Π
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0	
1	0	0	0	0	1	0	1	**5**
1	0	0	1	0	0	1	1	**3**
1	0	1	0	0	1	0	1	**5**
1	0	1	1	1	0	0	0	**8**
1	1	0	0	1	0	0	1	**9**
1	1	0	1	0	1	1	1	**7**
1	1	1	0	1	0	0	1	**9**
1	1	1	1	0	0	1	1	**3**

可编程只读存储器PROM



低密度可编程逻辑器件

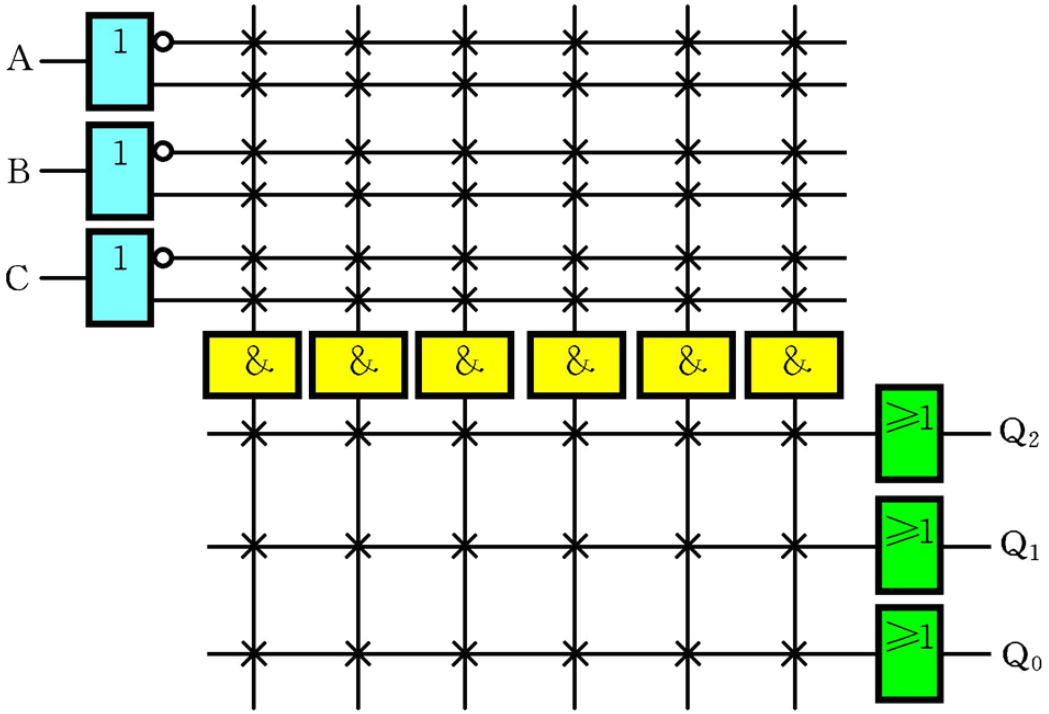
- 可编程逻辑阵列PLA
  - 对于大多数逻辑函数而言，并不需要使用全部最小项
  - PROM的“与” 阵列固定地产生n个输入变量的全部最小项，“与”阵列没有获得充分利用，芯片面积造成浪费
  - 为了克服PROM的不足，产生了一种“与”阵列和“或”阵列均可编程的逻辑器件，即可编程逻辑阵列PLA(Programmable Logic Array)

可编程逻辑阵列PLA

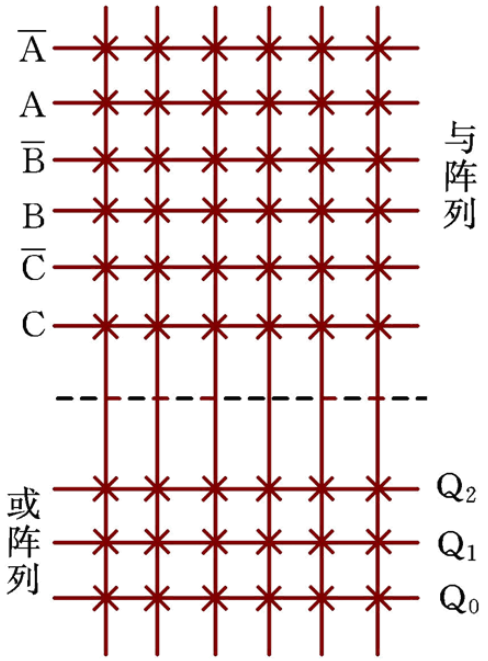
- 逻辑结构
  - 由一个“与”阵列和一个“或”阵列构成，“与”阵列和“或”阵列都是可编程的
  - 在PLA中，n个输入变量的“与”阵列通过编程提供需要的P个“与”项，“或”阵列通过编程形成“与-或”函数式
  - 由PLA实现的函数式是最简“与-或”表达式

可编程逻辑阵列PLA

– 例如，一个容量为3 - 6 - 3的PLA的逻辑结构图和阵列图如下：



(a)



(b)

可编程逻辑阵列PLA

- PLA的存储容量不仅与输入变量个数和输出端个数有关，而且还和它的“与”项数(即与门数)有关，存储容量用n（输入变量数）-p（与项数）-m（输出端数）来表示
- PLA的容量为3—6—3，表示3个输入3个输出，能够产生6个与项
- 目前常见的有容量为16—48—8和14—96—8等PLA器件

可编程逻辑阵列PLA

应用举例

- 相对PROM而言，PLA更灵活、更经济、结构更简单
- PLA设计组合逻辑电路
  - 将给定问题的逻辑函数按多输出逻辑函数的化简方法简化成最简“与-或”表达式
  - 根据最简表达式中的不同“与项” 以及各函数最简“与-或”表达式 确定“与”阵列和“或”阵列，并画出阵列逻辑图

可编程逻辑阵列PLA

- **例3** 用PLA设计一个代码转换电路，将一位十进制数的8421码转换成余3码
  - **解** 设ABCD------表示8421码，WXYZ-----表示余3码

A B C D	W X Y Z	A B C D	W X Y Z
0 0 0 0	0 0 1 1	1 0 0 0	1 0 1 1
0 0 0 1	0 1 0 0	1 0 0 1	1 1 0 0
0 0 1 0	0 1 0 1	1 0 1 0	d d d d
0 0 1 1	0 1 1 0	1 0 1 1	d d d d
0 1 0 0	0 1 1 1	1 1 0 0	d d d d
0 1 0 1	1 0 0 0	1 1 0 1	d d d d
0 1 1 0	1 0 0 1	1 1 1 0	d d d d
0 1 1 1	1 0 1 0	1 1 1 1	d d d d

可编程逻辑阵列PLA

– 据真值表写出函数表达式，并按照多输出函数化简法则利用卡诺图进行化简，可得到最简与-或表达式

$$W = A + BC + BD$$

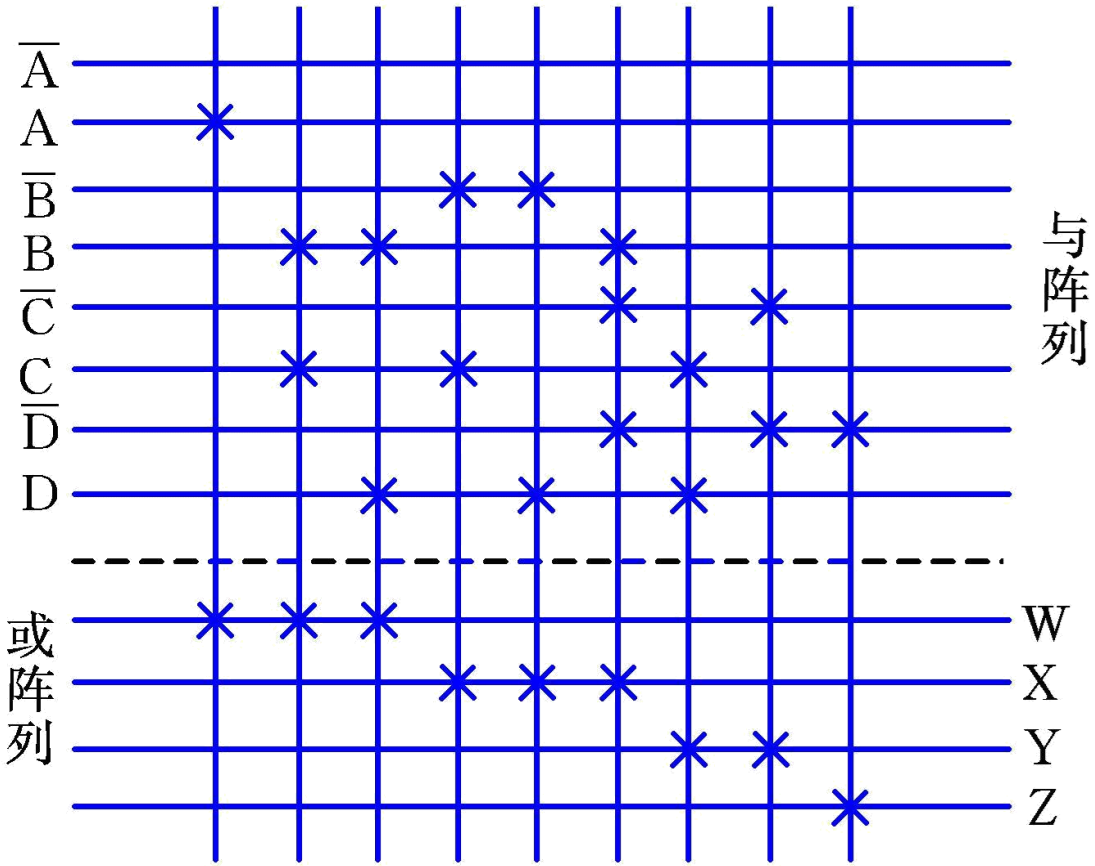
$$X = \overline{B}C + \overline{B}D + B\overline{C}\,\overline{D}$$

$$Y = CD + \overline{C}\,\overline{D}$$

$$Z = \overline{D}$$

可编程逻辑阵列PLA

– 阵列逻辑图



$$W = A + BC + BD$$

$$X = \overline{B}C + \overline{B}D + B\overline{C}\,\overline{D}$$

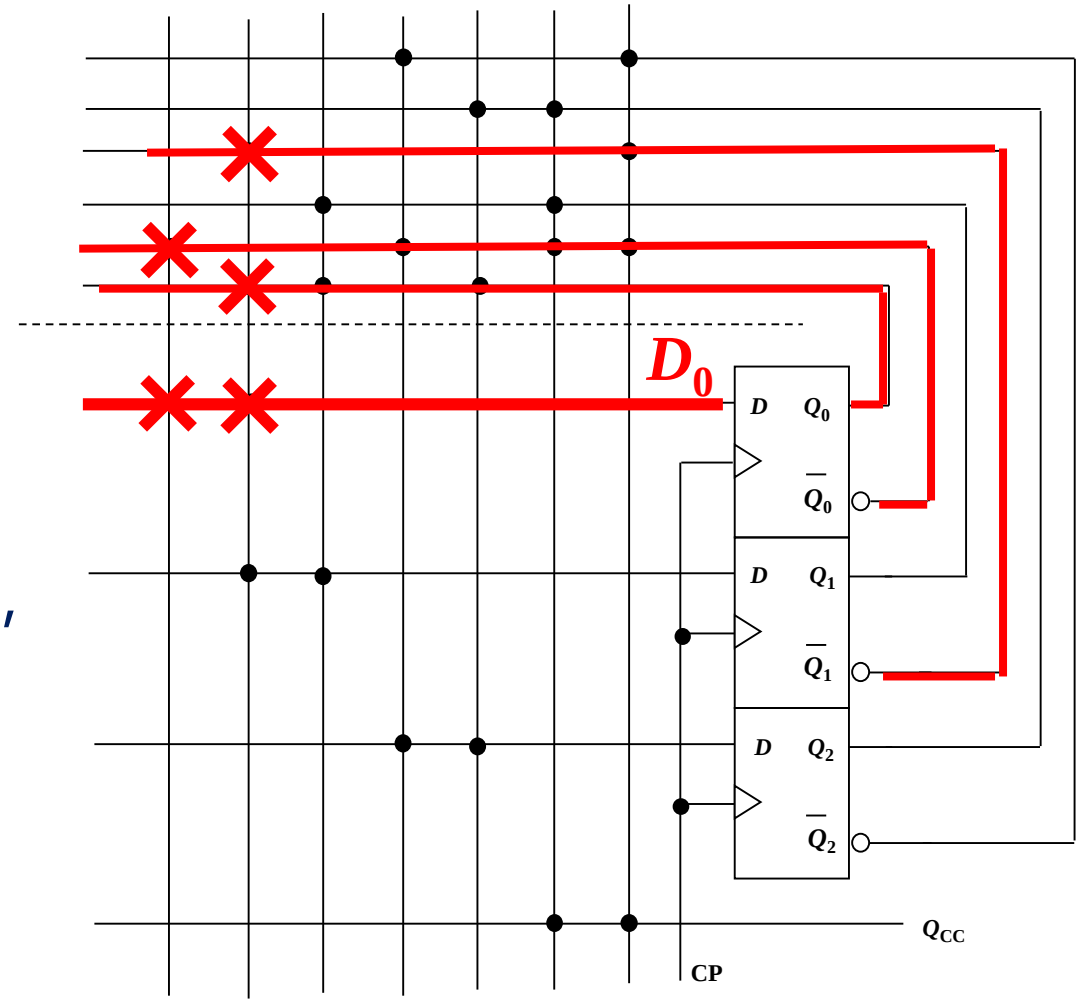
$$Y = CD + \overline{C}\,\overline{D}$$

$$Z = \overline{D}$$

可编程逻辑阵列PLA

例4 PLA和D触发器组成的同步时序电路如图所示，要求：（1）写出电路的驱动方程、输出方程。

（2）分析电路功能，画出电路的状态转换图。



可编程逻辑阵列PLA

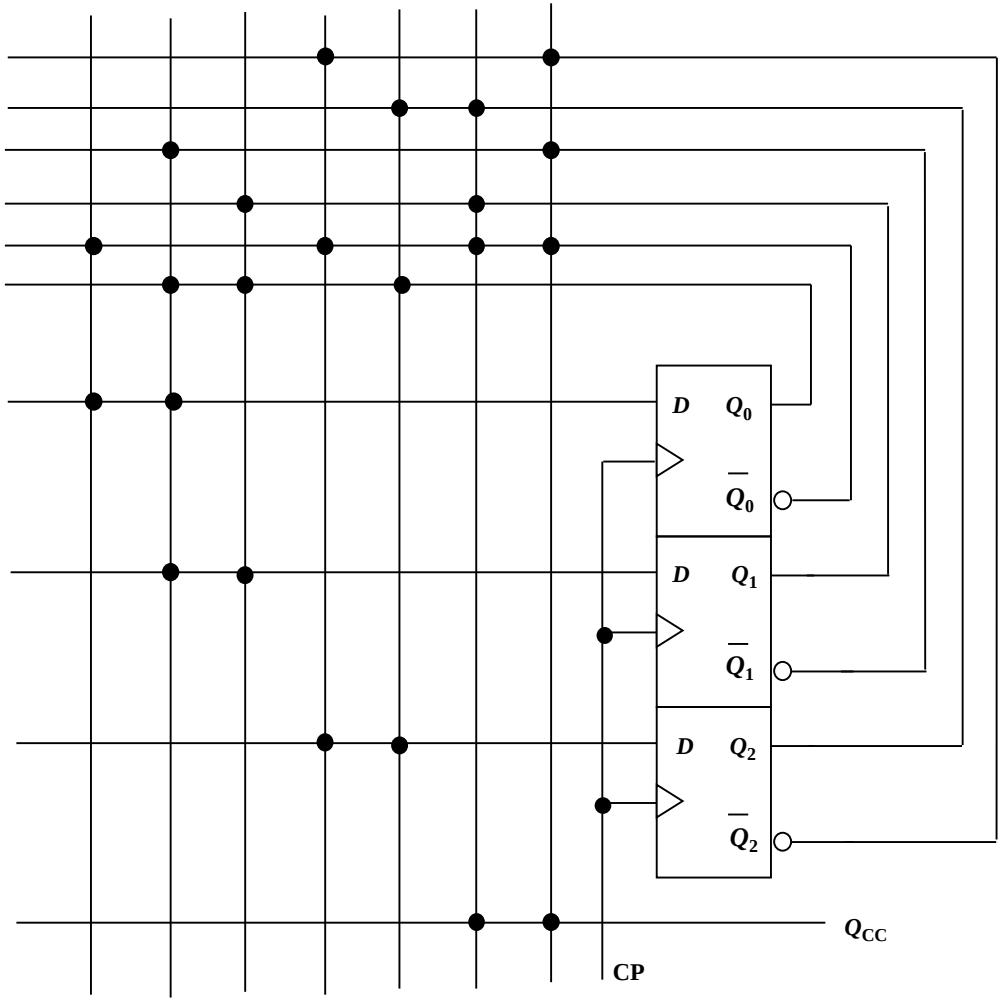
口解（1）驱动方程、输出方程

$$D_0 = \overline{Q}_0 + \overline{Q}_1 Q_0$$

$$D_1 = \overline{Q}_1 Q_0 + Q_1 Q_0$$

$$D_2 = \overline{Q}_0\,\overline{Q}_2 + Q_2 Q_0$$

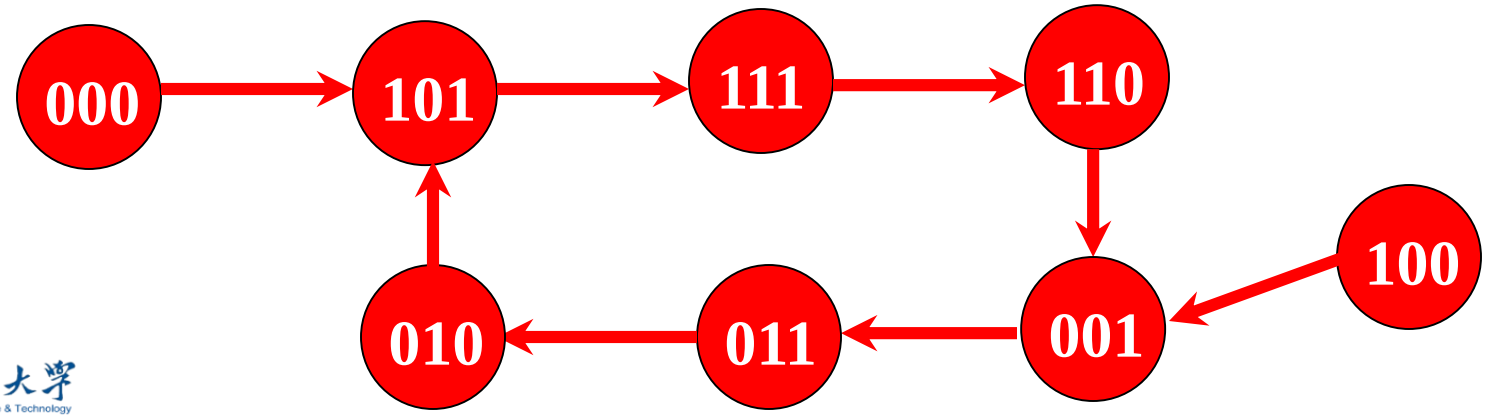
$$Q_{CC} = \overline{Q}_0\,Q_1 Q_2 + \overline{Q}_0\,\overline{Q}_1\,\overline{Q}_2$$



可编程逻辑阵列PLA

Q_2	Q_1	Q_0	D_2	D_1	D_0	Q_{2n+1}	Q_{1n+1}	Q_{0n+1}	Q_{CC}
0	0	0	1	0	1	1	0	1	1
0	0	1	0	1	1	0	1	1	0
0	1	0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	1	0	0
1	0	0	0	0	1	0	0	1	0
1	0	1	1	1	1	1	1	1	0
1	1	0	0	0	1	0	0	1	1
1	1	1	1	1	0	1	1	0	0

根据状态转换表，画出下图所示的电路状态转换图。



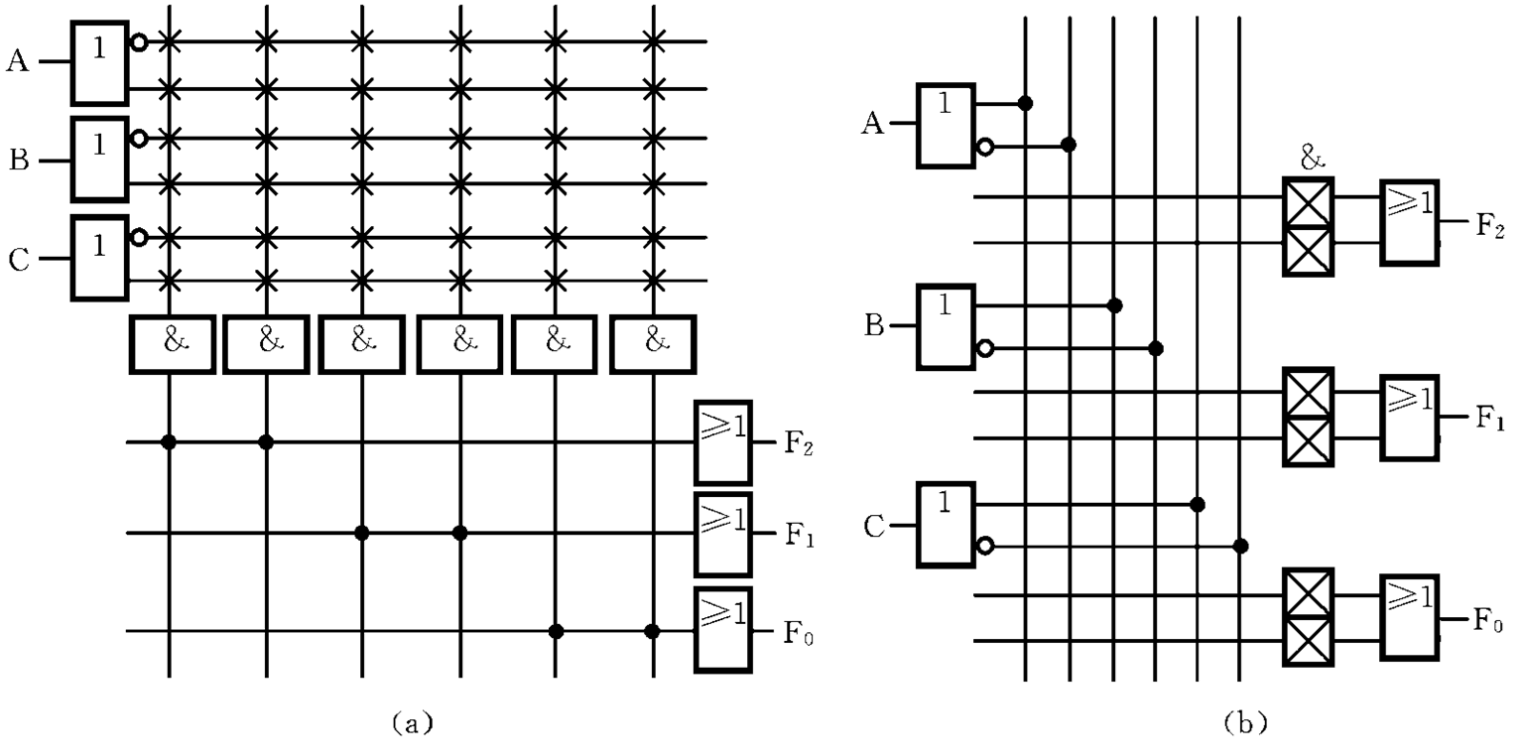
低密度可编程逻辑器件

- 可编程阵列逻辑（PAL）
  - PAL(Programmable Array Logic)是在PROM和PLA的基础上发展起来的一种可编程逻辑器件
  - 相对于PROM而言，使用更灵活，且易于完成多种逻辑功能，同时又比PLA工艺简单，易于实现
  - 可编程阵列逻辑 (PAL)是一种**“与”阵列可编程**，**“或”阵列固定**的逻辑器件

可编程阵列逻辑PAL

PAL的逻辑结构

– 图 (a)给出了一个三输入三输出PAL的逻辑结构图，通常将其表示成图 (b)所示形式



可编程阵列逻辑PAL

- PAL每个输出包含的“与”项数目是由固定连接的“或”阵列提供的
- 在典型逻辑设计中，一般函数约包含3个～4个“与”项，而现有PAL器件最多可为每个输出提供8个“与”项，能很好地完成各种常用逻辑电路的设计
- PAL器件的结构(包括输入、输出、“与”项数目)是由生产厂家固定的

可编程阵列逻辑PAL

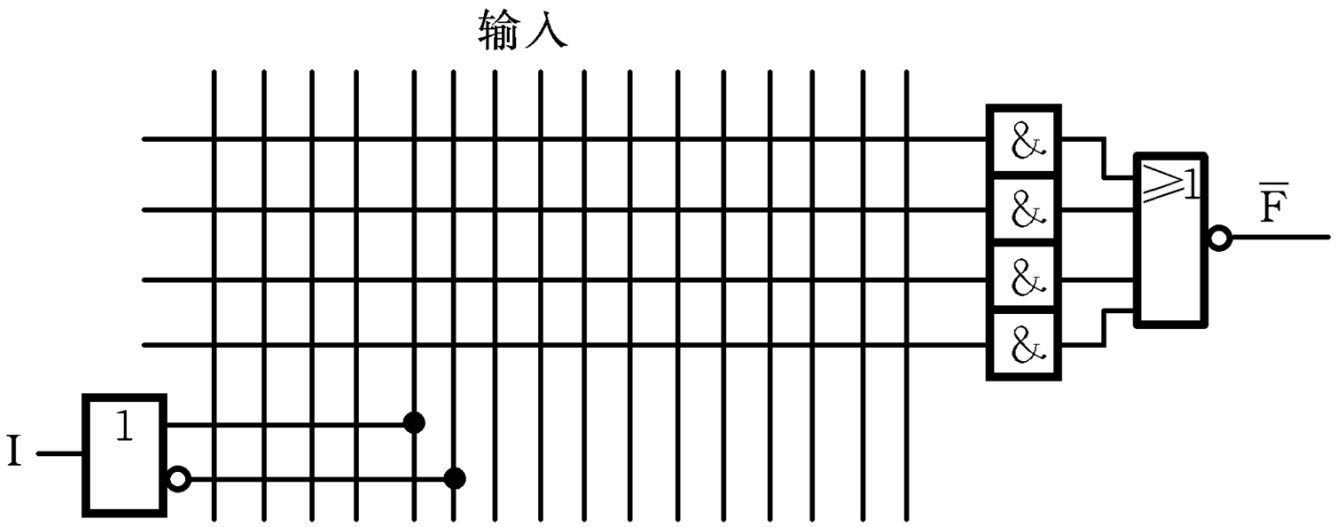
- 按照PAL的输出和反馈结构，通常可分为5种基本类型
  - 专用输出的基本门阵列结构
  - 带反馈的可编程I/O结构
  - 带反馈的寄存器输出结构
  - 加异或、带反馈的寄存器输出结构
  - 算术选通反馈结构

可编程阵列逻辑PAL

- 专用输出的基本门阵列结构
  - 这种结构类型适用于实现组合逻辑函数
  - 常见产品有PAL10H8(10个输入，8个输出，输出高电平有效)，PAL12L6(12个输入，6个输出，输出低电平有效)等

可编程阵列逻辑PAL

- 下图表示专用输出的基本门阵列结构类型的1个输入、1个输出、4个“与”项的局部电路



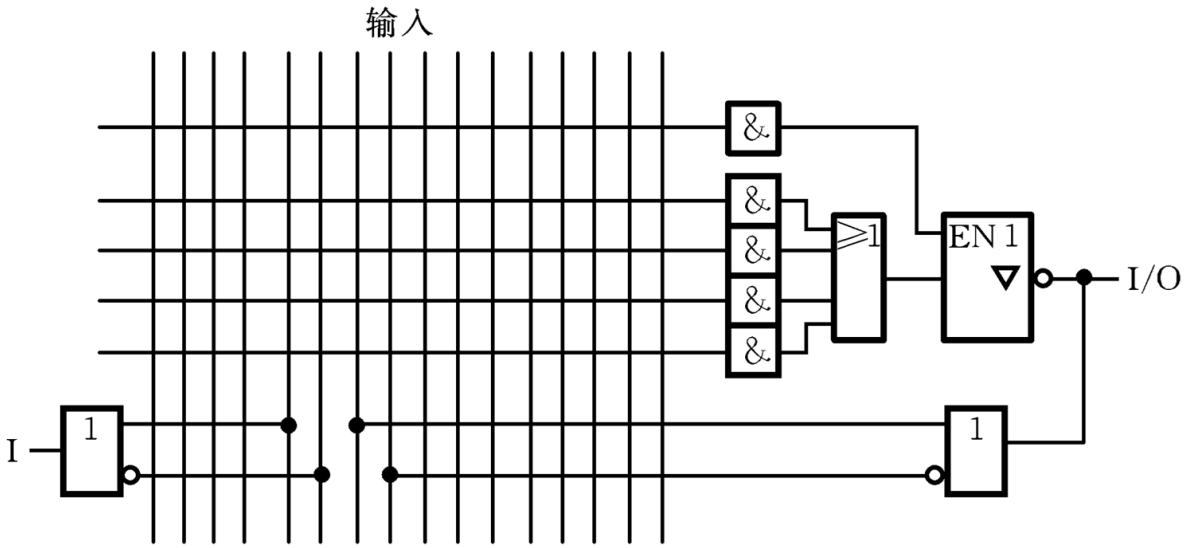
- 采用或非门，为低电平有效器件
- 采用或门结构，则为高电平有效器件
- 采用互补输出的或门，则称为互补输出器件

可编程阵列逻辑PAL

- 带反馈的可编程I/O结构
  - 又称为异步可编程I/O结构
  - 常见产品有PAL16L8(10 个输入，8个输出，6个反馈输入)以及PAL20L10(12个输入，10个输出，8个反馈输入)

可编程阵列逻辑PAL

– 下图给出了这种结构类型的一个局部电路



– 最上面一个与门作为输出三态缓冲器的选通控制

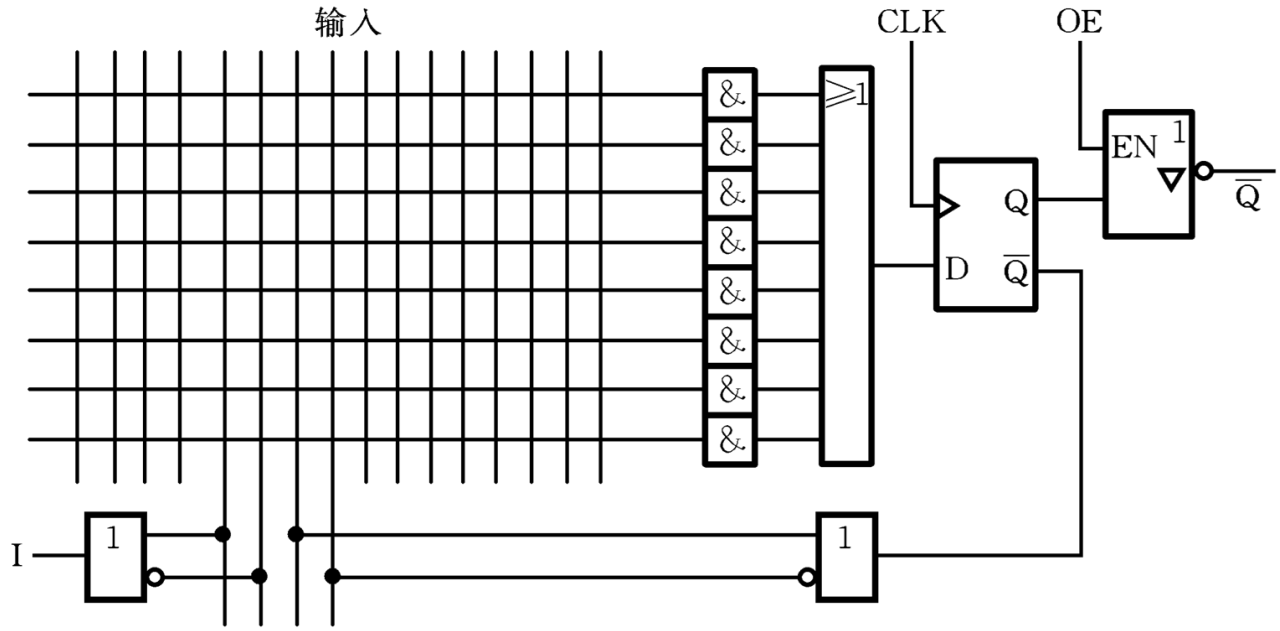
– 通过编程指定某些I/O端方向，可改变器件输入/输出线数目的比例

可编程阵列逻辑PAL

- 带反馈的寄存器输出结构
  - 带反馈的寄存器输出结构使PAL构成了典型的时序网络结构
  - 典型产品有PAL16R8(8个输入、8个寄存器输出、8个反馈输入、1个公共时钟和1个公共选通)

可编程阵列逻辑PAL

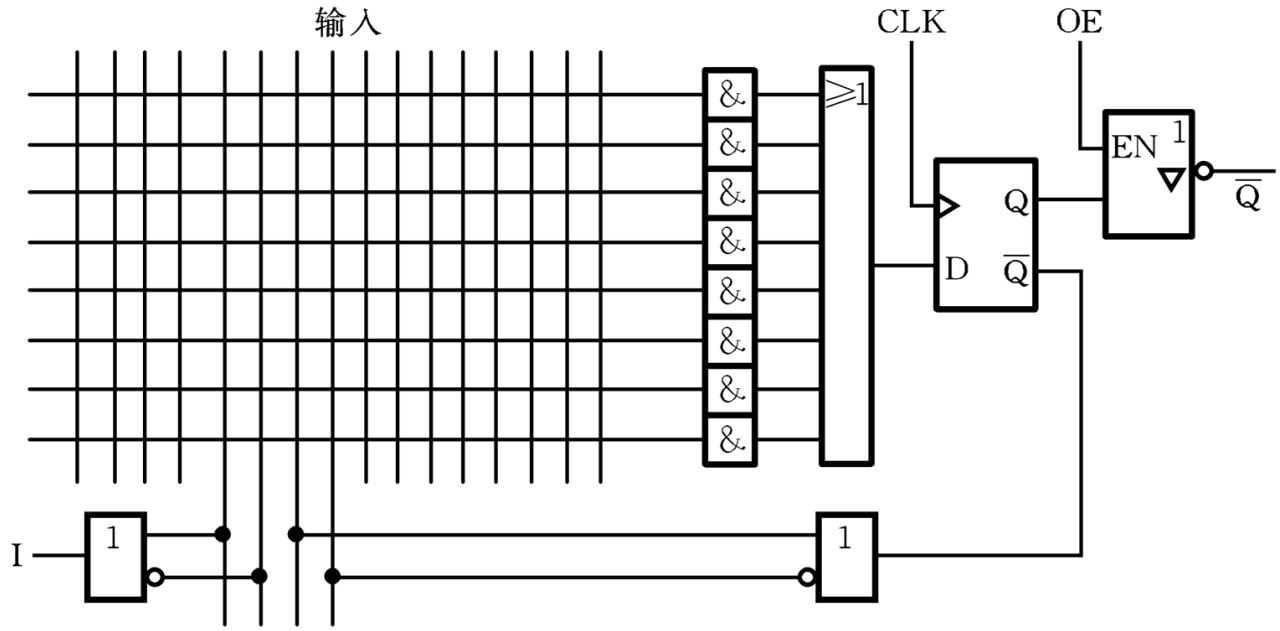
– 带反馈的寄存器输出结构的局部电路如图所示



– 由或门产生的输出在系统时钟CLK(公共的)作用下存入到D触发器中，触发器的输出通过带有公共选通(OE)的三态缓冲器送到输出端，输出低电平有效

可编程阵列逻辑PAL

– 带反馈的寄存器输出结构的局部电路如图所示



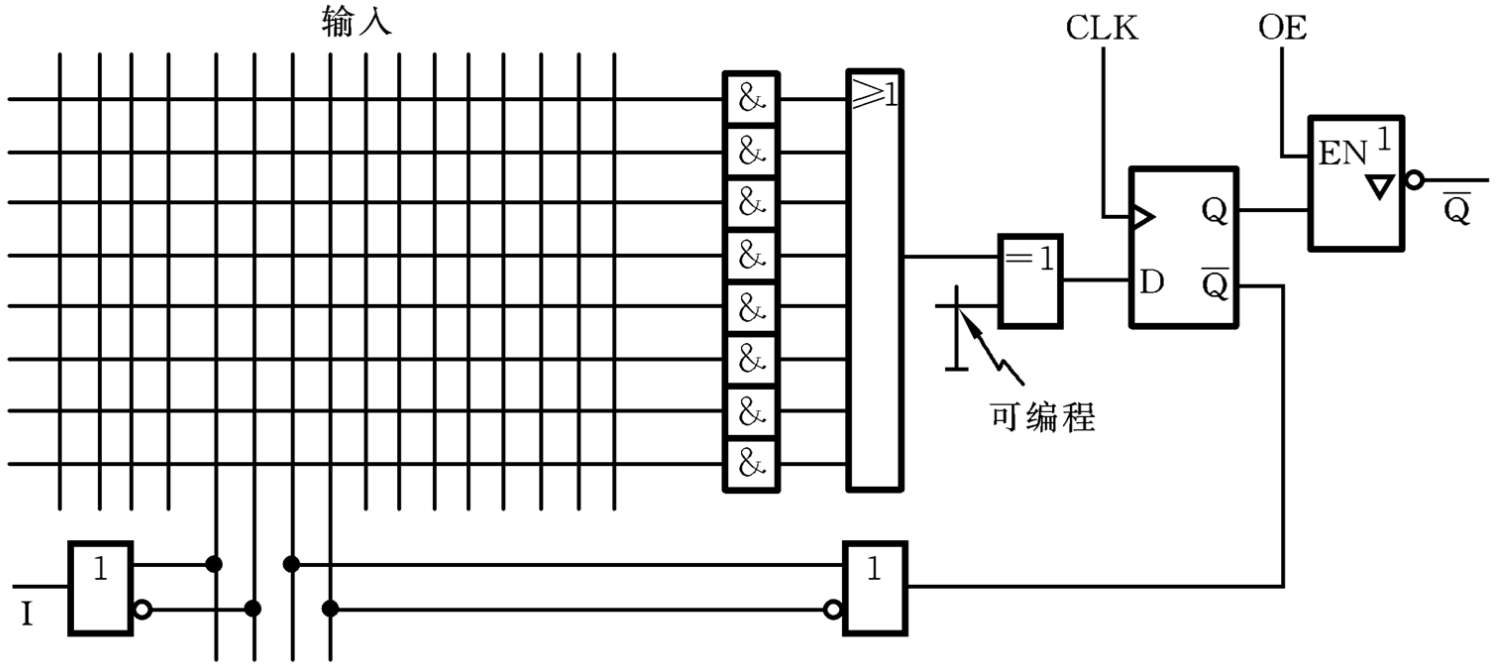
– D触发器的输出通过一个缓冲器反馈回“与”阵列，这种反馈功能使PAL构成了典型的时序网络结构，从而能实现时序逻辑电路功能

低密度可编程逻辑器件

- 加“异或”、带反馈的寄存器输出结构
  - 在带反馈寄存器输出结构的基础上增加了一个异或门
  - 典型产品有PAL16RP8 (8个输入，8个寄存器输出，8个反馈输入)

低密度可编程逻辑器件

– 该类PAL的一个局部电路如下图所示



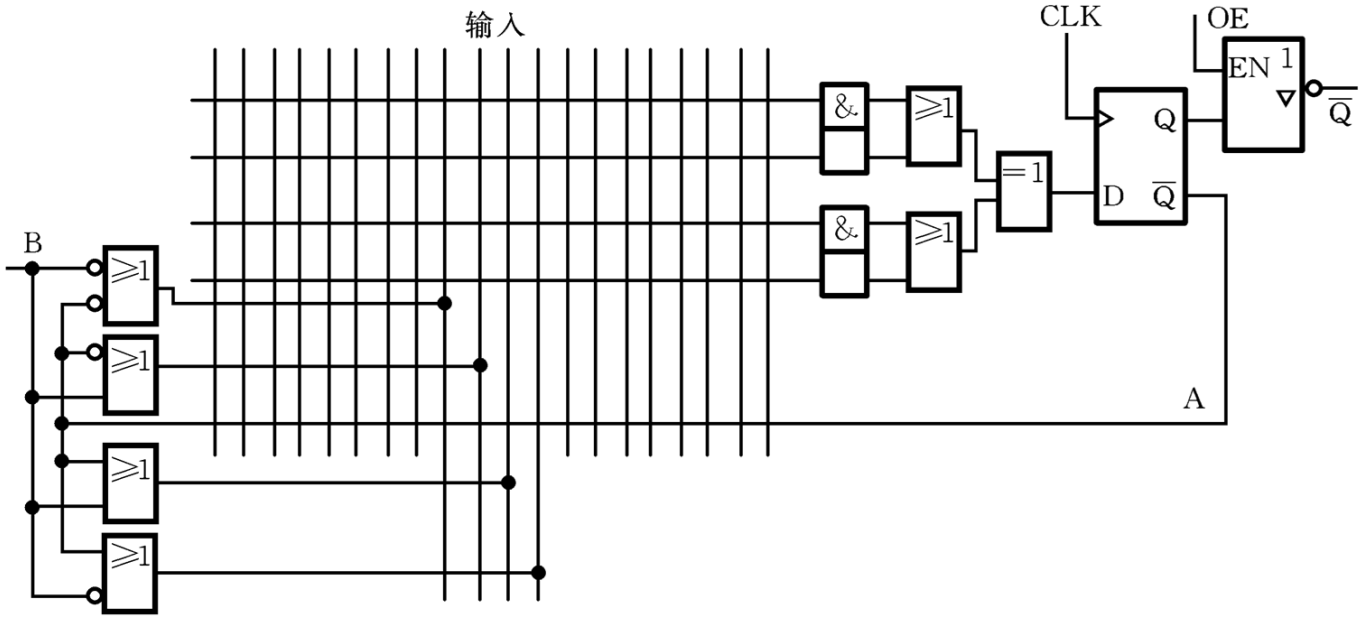
– 在D触发器的D端引入一个异或门，使D端的极性可通过编程设置，这实际上是允许把输出端置为高电位有效或者低电位有效

低密度可编程逻辑器件

- 算术选通反馈结构
  - 算术PAL是在综合前几种PAL结构特点的基础上，增加了反馈选通电路，使之能实现多种算术运算功能
  - 典型产品有PAL16A4(8个输入、4个寄存器输出、4个可编程I/O输出、4个反馈输入、4个算术选通反馈输入)

低密度可编程逻辑器件

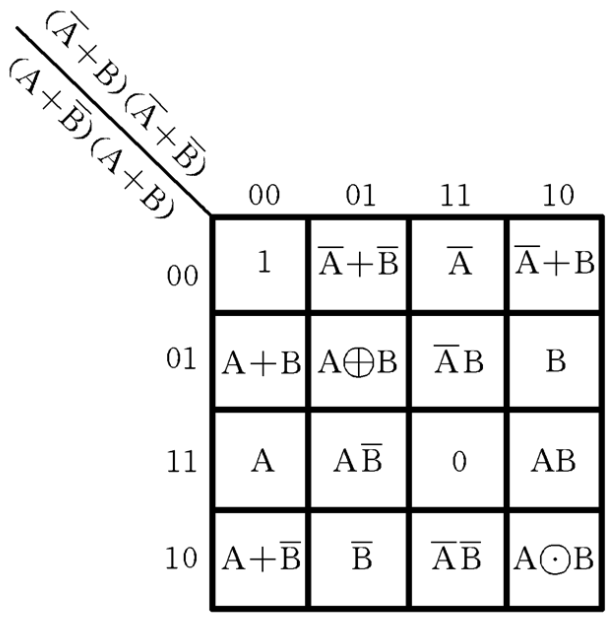
– 下图给出了这种结构的一个局部电路

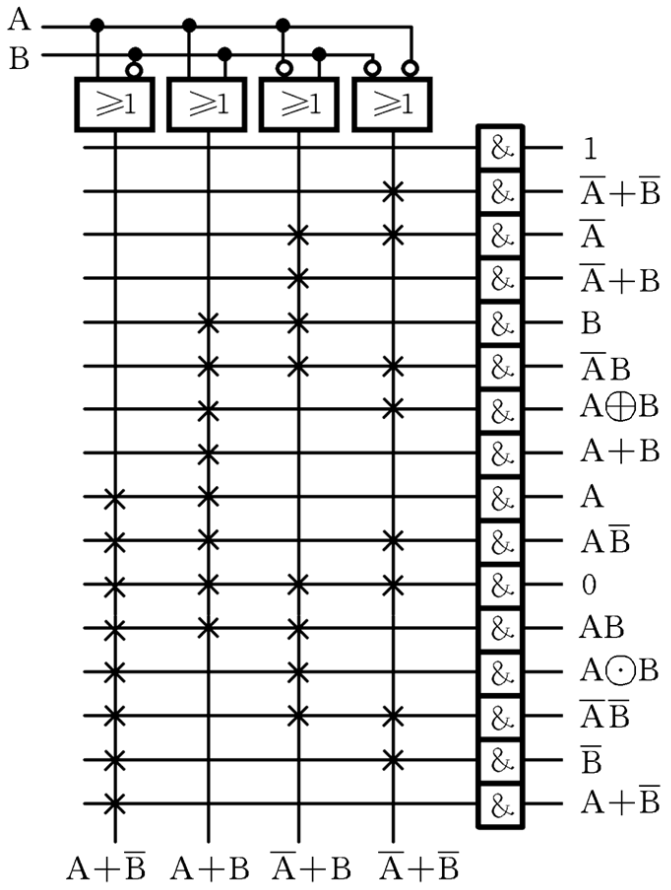


– 反馈选通电路中的4个或门接收反馈项A和输入项B，输出$(A+B),(A+\bar{B}),(\bar{A}+B),(\bar{A}+\bar{B})$，以此4个项作为逻辑变量送“与”阵列进行编程

低密度可编程逻辑器件

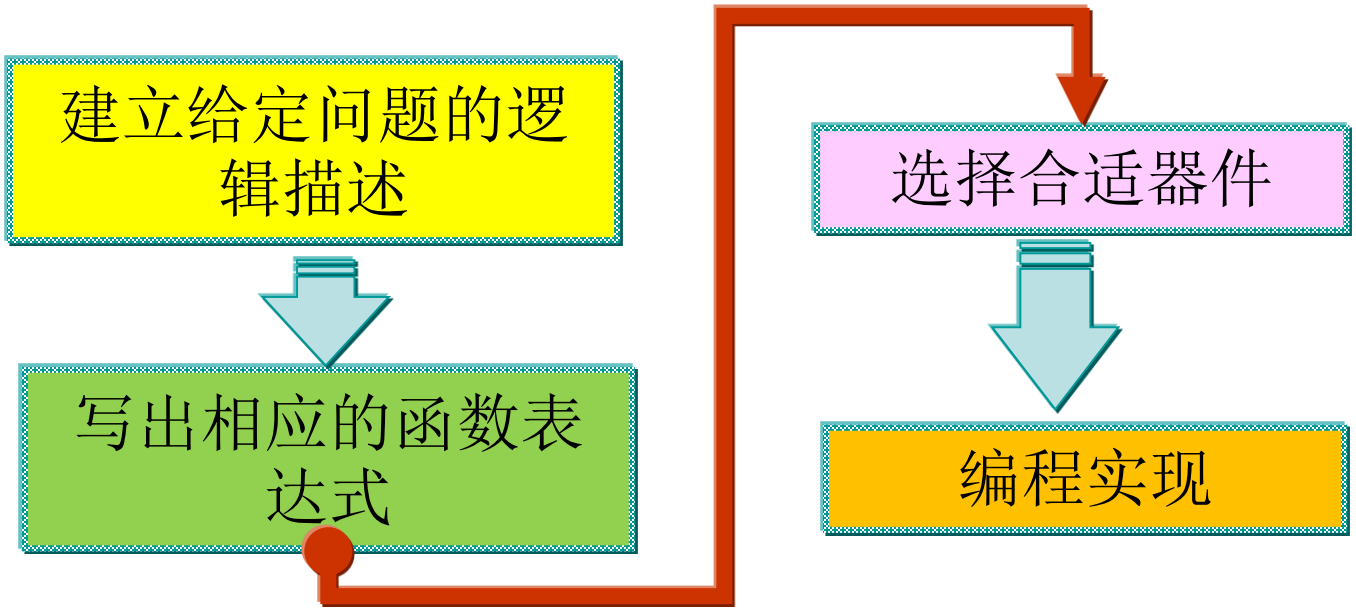
– 下图给出了反馈选通电路的算术功能编程





低密度可编程逻辑器件

- PAL设计的一般步骤



低密度可编程逻辑器件

- 通用阵列逻辑（GAL）
  - GAL的基本结构与PAL相类似，都是由一个可编程的与阵列去驱动一个固定连接的或阵列，所不同的是输出部件结构不同
  - GAL在每一个输出端都集成有一个输出逻辑宏单元(OLMC)，允许用户定义每个输出的结构和功能

通用阵列逻辑GAL

- GAL器件具有PAL器件所没有的可擦除、可重写及结构可组态等特点。这些特点形成了器件的可测试性和高可靠性，且具有更大的灵活性。
- 典型器件GAL16V8：GAL16V8芯片是具有8个固定输入引脚、最多可达16个输入引脚，8个输出引脚，输出可编程的一种GAL器件

通用阵列逻辑GAL

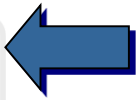
- 组成: 由8个输入缓冲器、8个反馈输入缓冲器、8个输出逻辑宏单元OLMC，8个输出三态缓冲器、“与”阵列以及系统时钟、输出选通信号等组成
- 其中，“与”阵列包含32列和64行，32列表示8个输入的原变量和反变量及8个输出反馈信号的原变量和反变量；64行表示“与”阵列可产生64个“与”项，对应8个输出，每个输出包括8个“与”项

通用阵列逻辑GAL

- GAL的开发过程
  - 第一步：分析设计要求，完成逻辑设计
  - 第二步：根据逻辑设计结果，选择GAL器件并对器件进行引脚分配
  - 第三步：根据开发软件要求，编写设计源文件，并输入到计算机中
  - 第四步：调用开发软件对设计源文件进行编译、优化以及功能仿真
  - 第五步：硬件编程
  - 第六步：数据校验

提 纲

1. PLD概述
2. 低密度可编程逻辑器件
3. 高密度可编程逻辑器件
4. 在系统编程技术简介



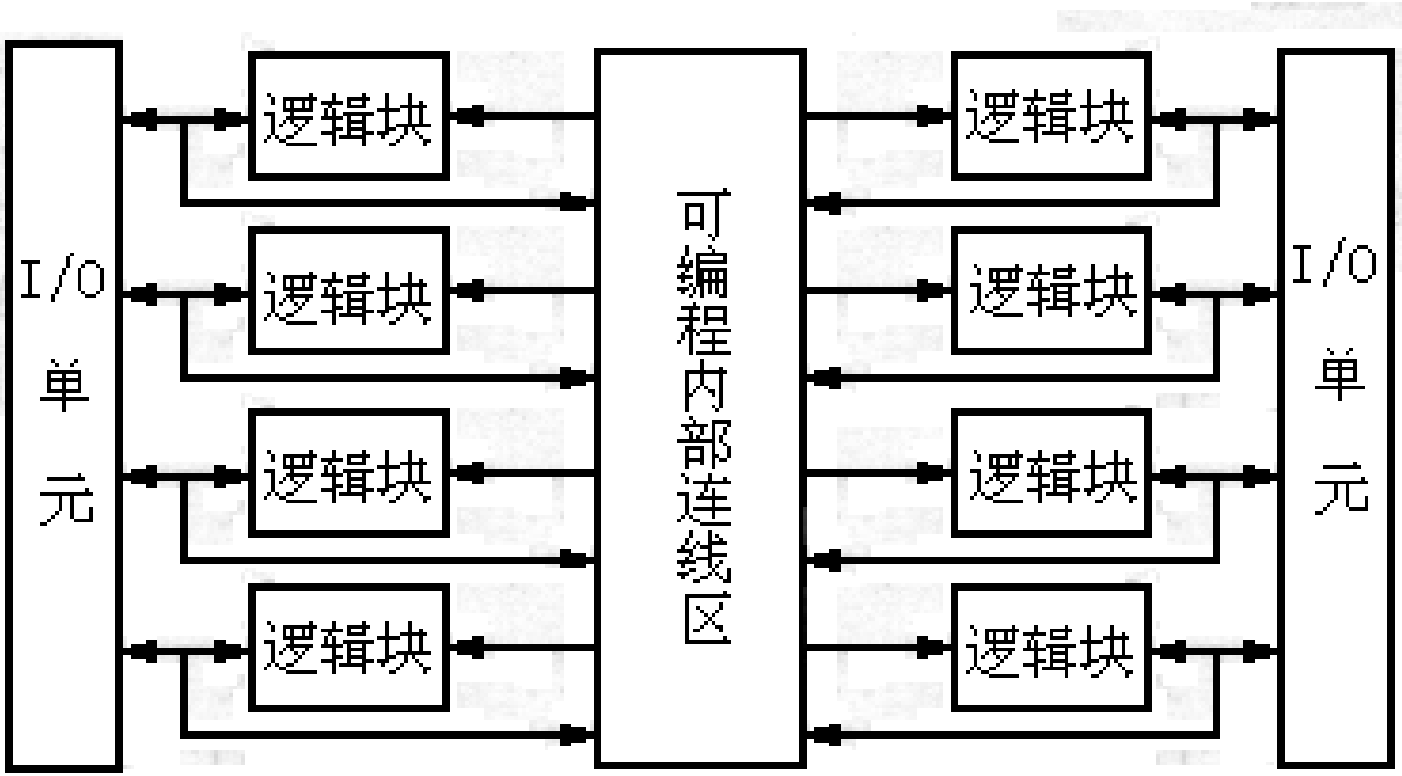
高密度可编程逻辑器件

□复杂可编程逻辑器件

- 复杂可编程逻辑器件（complex programmable logic device，简称CPLD）是从简单PLD发展而来的高密度PLD器件
- 一般由逻辑块、可编程内部连线区和I/O单元组成

复杂可编程逻辑器件CPLD

– 常见CPLD的结构示意图



复杂可编程逻辑器件CPLD

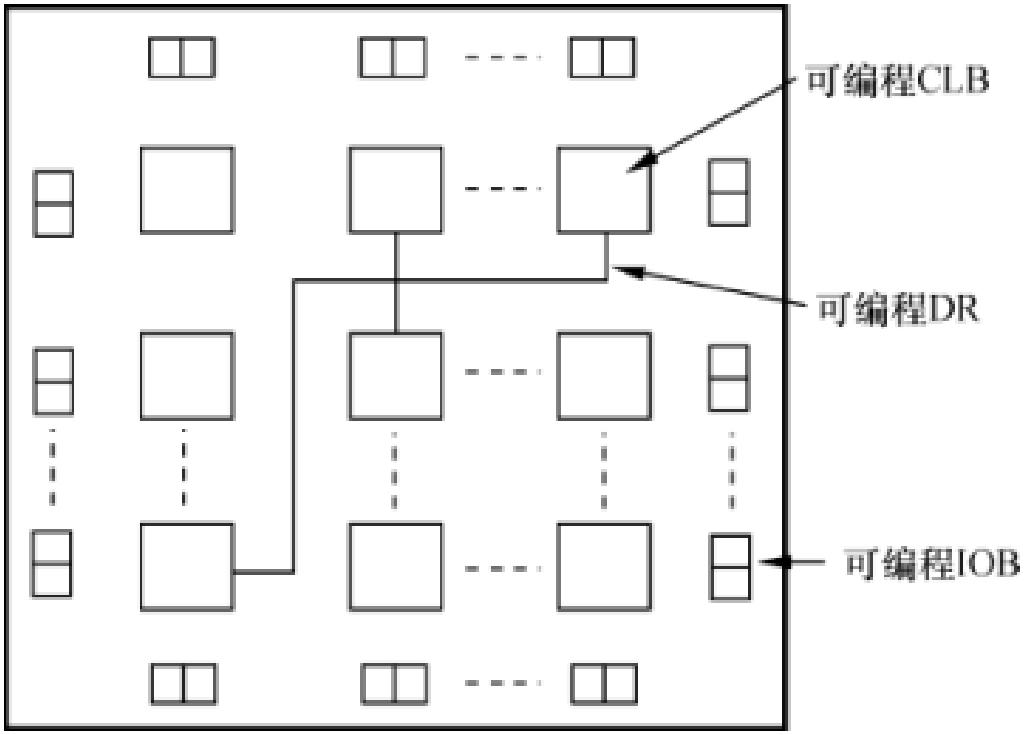
- CPLD的结构
  - 通用互连阵列结构
  - 大块结构
  - 灵活逻辑单元阵列结构
- 典型器件
  - 最常用的CPLD有Altera公司生产的FLEX 10K系列器件
  - FLEX 10K是一种嵌入式的PLD，它采用灵活逻辑单元阵列结构和重复可构造的CMOS SRAM工艺，具有高密度、低成本、低功率等特点

高密度可编程逻辑器件

- 现场可编程门阵列
  - 现场可编程门阵列FPGA（Field Programmable Gate Array）是20世纪80年代中后期发展起来的一种高密度可编程逻辑器件，它由世界著名的可编程逻辑器件供应商Xilinx公司最初提出

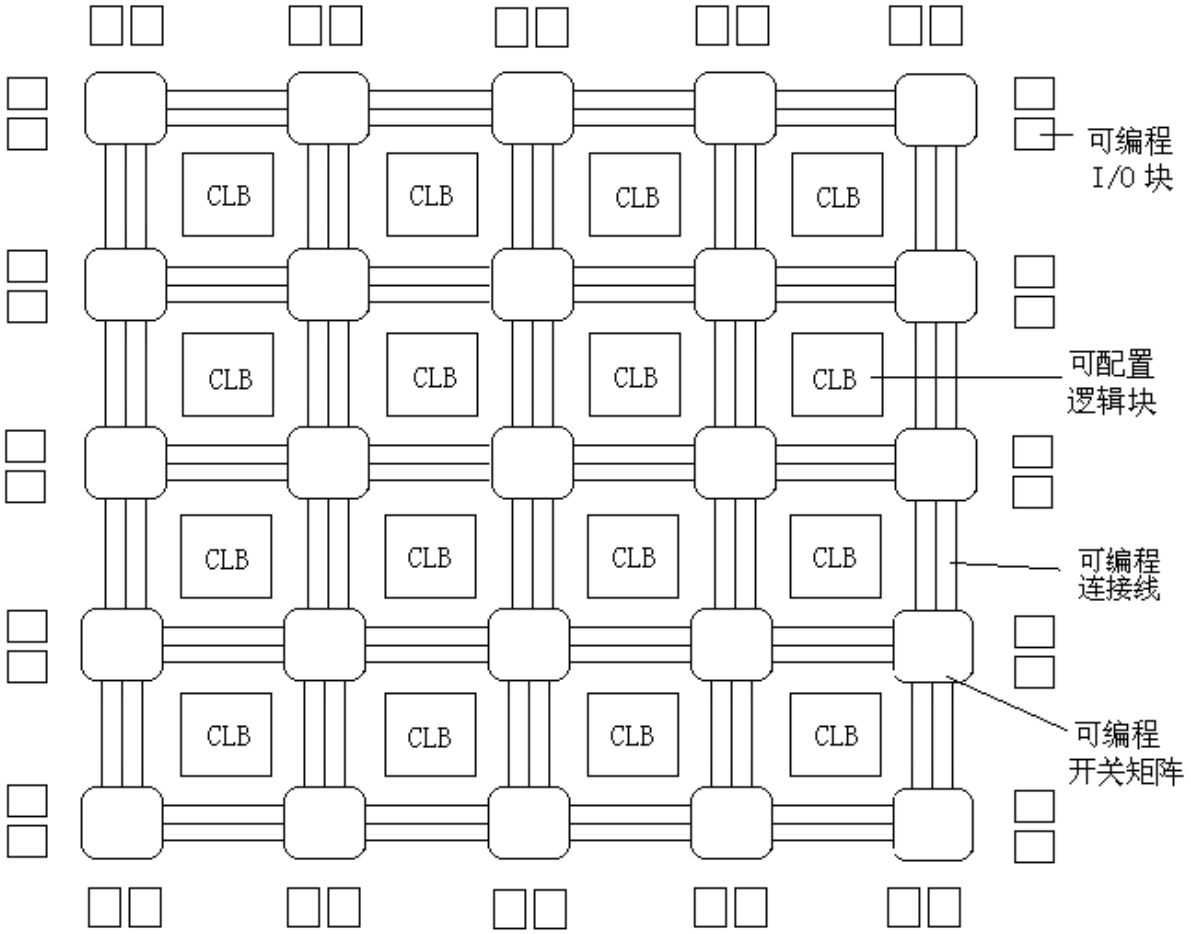
现场可编程门阵列FPGA

- FPGA的基本结构
  - 可配置逻辑块CLB
  - 可编程输入/输出块IOB
  - 可编程互连资源PIR



高密度可编程逻辑器件

– Xilinx公司的XC4000系列FPGA器件的结构示意图

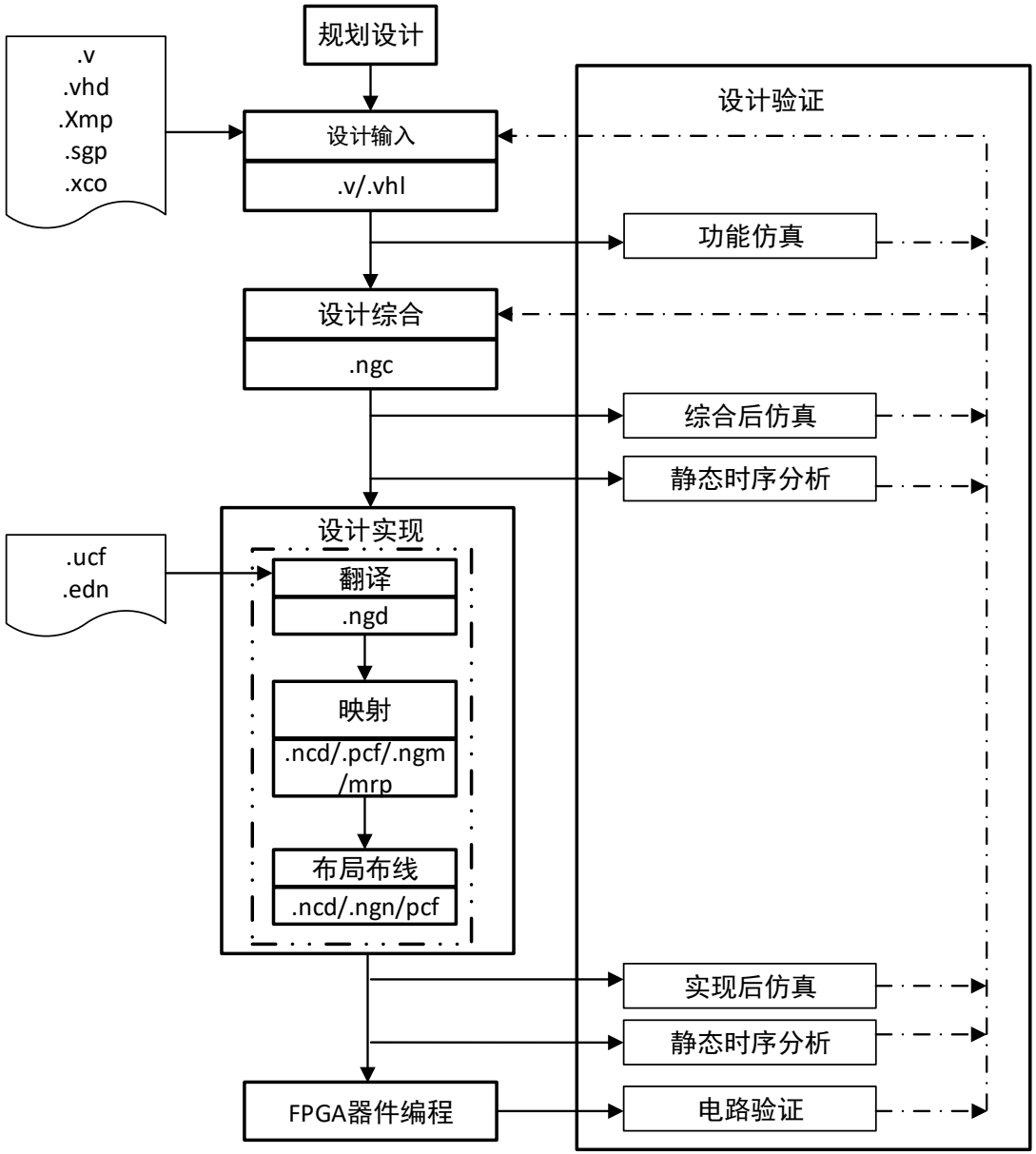


现场可编程门阵列FPGA

- 典型器件
  - 典型器件有Xilinx公司的XC4062XL，该器件包含2304个CLB（构成48×48 CLB矩阵）、62000个逻辑门、5376个触发器、最大用户I/O达384个、最大RAM位数达73728位

现场可编程门阵列FPGA

□设计流程



FPGA设计流程

- 设计规划
  - 也称为架构阶段
  - 针对用户需求、任务书、技术协议书等的规定进行需求的定义和分析
  - 确定要实现的系统功能和大概的模块划分
  - 根据任务要求，对工作速度和器件本身的资源、成本等方面进行权衡，选择合适的设计方案和合适的器件类型

FPGA设计流程

- 设计输入
  - 利用EDA工具将概念设计转化为硬件描述的过程
  - 创建FPGA工程，并创建或添加设计源文件、约束文件等到工程中
  - 概念设计转化常用的方法有硬件描述语言(HDL)和原理图输入方法等

FPGA设计流程

- 设计综合（Synthesis）
  - 将硬件语言或原理图等设计输入转换成由基本门电路、RAM和触发器等基本逻辑单元组成的逻辑连接网表的过程
  - 该过程中，针对输入设计以及约束条件，按照一定的优化算法进行优化处理，将RTL级推演的网表文件映射到FPGA器件原语（也称为技术映射），生成综合的网表文件。

FPGA设计流程

- 设计实现
  - 通过翻译（Translate）、映射（Map）、布局布线（Place & Route）等过程来将逻辑设计进一步转译为可以下载烧录到目标FPGA器件中的特定物理文件格式的过程

FPGA设计流程

- FPGA配置
  - 设计的最后一步就是FPGA配置，进行芯片编程和调试
  - 芯片编程是指生成位数据流文件，并将编程数据下载到FPGA芯片中

FPGA设计流程

- 下载验证
  - 下载是在功能仿真与时序仿真正确的前提下，将综合后形成的位流下载到具体的 FPGA 芯 片中，也叫芯片配置
  - FPGA 设计有两种配置形式
    - 直接由计算机经过专用下载电缆进行配 置
    - 由外围配置芯片进行上电时自动配置

FPGA设计流程

- 设计验证
  - 是FPGA开发流程中最重要的一个内容
  - 设计验证的方
    - 仿真
    - 静态时序分析
    - 电路验证

FPGA设计流程

- 仿真
  - **行为仿真**，又称功能仿真、RTL级仿真，是在编译之前对用户所设计的电路进行逻辑功能验证
  - **综合后仿真**，又称为门级仿真，目的在于检查综合结果是否和原设计一致
  - **实现后仿真**，又称为时序仿真，其目的和综合后仿真一致，但实现后的时序仿真加入了走线延时信息，使得仿真与FPGA本身运行状态一致

FPGA设计流程

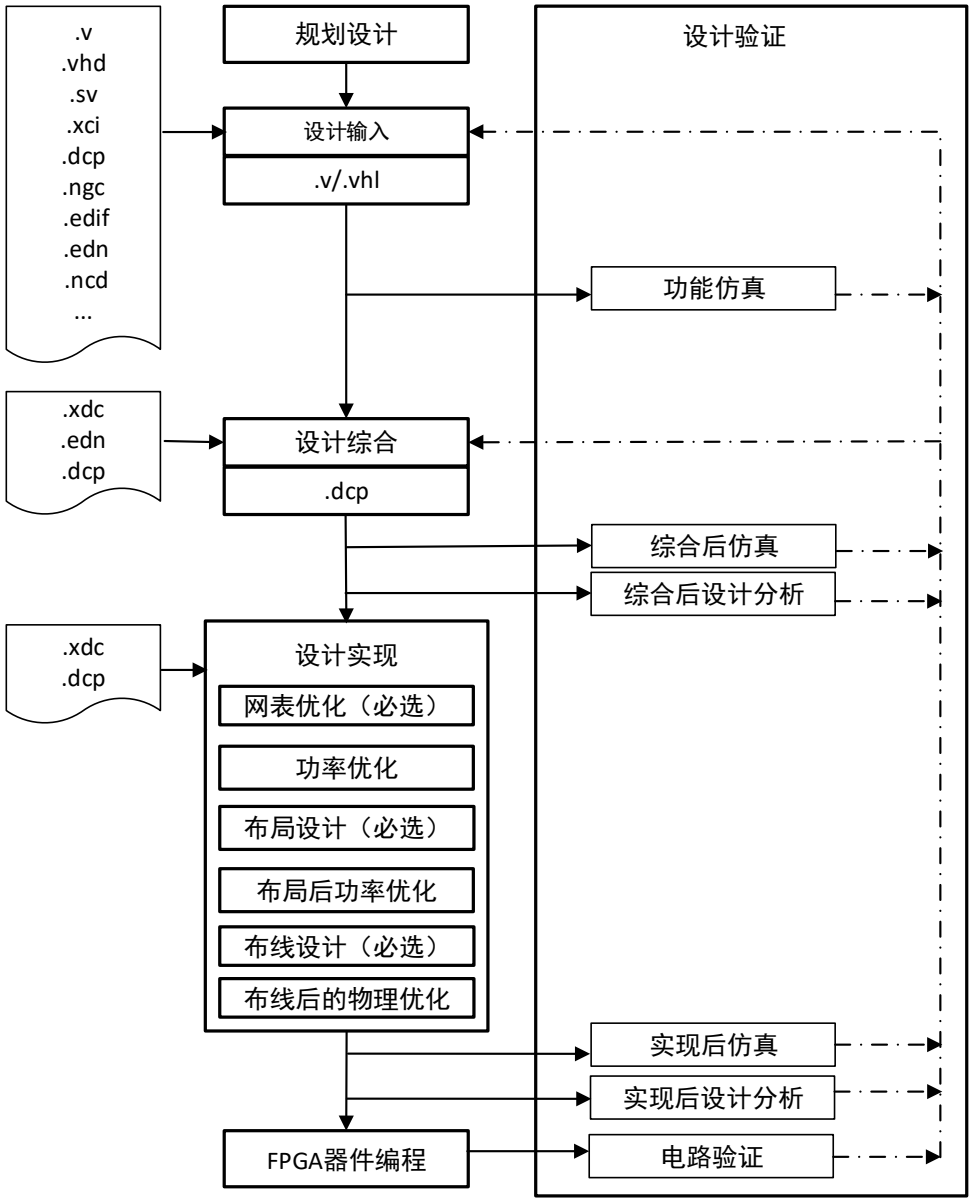
- 静态时序分析
  - 在综合设计或布局布线之后，对设计进行快速时序检查
  - 验证设计是否满足时序约束，并列举输入约束冲突，以分析部分或全部的布局布线设计
- 电路验证
  - 作为最后的测试，验证设计在目标应用中的表现
  - 在典型的运行条件下，验证测试电路

Vivado设计套件

- Xilinx公司于2012年推出了一种以知识产权核（Intellectual Property core，IP核）和系统为中心的、领先一代的全新SoC（System on Chip）增强型综合开发环境Vivado设计套件

Vivado设计套件

□设计流程



Vivado设计流程

- 设计输入
  - 可综合的HDL代码、测试文件、IP以及网表文件
  - 可综合的HDL代码和测试文件可以是Verilog、VHDL或者System Verilog代码。
  - 网表文件可以是Vivado生成的网表文件，也可以是第三方网表文件EDIF

Vivado设计流程

- 设计综合
  - Vivado集成环境综合是基于时间驱动的，对存储器的利用率和性能进行了优化
  - 综合时可以加入第三方网表文件EDIF以及约束文件
  - Vivado设计套件内置Synthesis综合功能，也可以支持第三方综合工具

Vivado设计流程

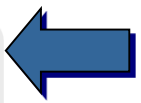
- 设计实现
  - 网表优化（opt_design）：必选
  - 功率优化（power_opt_design）
  - 布局设计（place_design）：必选
  - 布局后功率优化
  - 物理优化（phys_opt_design）
  - 布线设计（route_design）：必选
  - 布线后的物理优化

Vivado设计流程

- 配置和下载验证
  - 可以查看静态报告，也可以动态地查看设计综合实现的结果
  - 在Vivado内置工具中可以查看时序结果和功耗结果
  - 系统调试时可以使用在线逻辑分析仪ILA

提 纲

1. PLD概述
2. 低密度可编程逻辑器件
3. 高密度可编程逻辑器件
4. 在系统编程技术简介



在系统编程技术简介

- 在系统编程(In System Programmable，简称ISP)技术是90年代发展起来的一种PLD技术
- 可以在用户自己设计的目标系统上、为实现预定逻辑功能而对逻辑器件进行编程或改写
- ISP器件的出现，使数字系统设计更加灵活、方便，为用户带来了显著的经济效益和时间效益

在系统编程技术简介

- ISP技术的主要特点
  - 全面实现了硬件设计与修改的软件化
  - 简化了设计与调试过程
  - 容易实现系统硬件的现场升级
  - 可降低系统成本，提高系统可靠性
  - 器件制造工艺先进，性能参数好

在系统编程技术简介

- ISP逻辑器件类型
  - ispLSI逻辑器件
  - ispGAL器件
  - ispGDS器件

在系统编程技术简介

- ispLSI逻辑器件
  - ISP技术是美国Lattice公司于1991年率先推出的，该公司将ISP技术应用到高密度可编程逻辑器件(HDPLD)中，形成了ispLSI系列高密度在系统可编程逻辑器件
  - 具有集成度高、速度快、可靠性好、灵活方便等优点
  - 能满足在高性能系统中实现各种复杂逻辑功能的需要
  - 广泛应用于数据处理、图形处理、空间技术、军事装备及通信、自动控制等领域

在系统编程技术简介

- ispGAL器件（ispGAL22V10）
  - 把流行的GAL22V10与ISP技术相结合形成的产品，功能和结构上与GAL22V10完全相同
  - 性能：传输时延低于7.5ns；系统速度高达111MHz；编程次数可达1万次以上；编程电源为+5V，无需外接编程高压电源；与GAL22V10的引脚相互兼容
  - 适应范围：高速图形处理和高速总线管理，状态控制、数据处理、通信工程、测量仪器以及实现诸如地址译码器之类的基本逻辑功能

在系统编程技术简介

- ispGDS器件
  - ispGDS(在系统可编程数字开关)是ISP技术与开关矩阵相结合的产物
  - 标志着ISP技术已从系统逻辑领域扩展到系统互连领域
  - 独特功能：在不拨动机械开关或不改变系统硬件的情况下，快速地改变或重构印制电路板的连接关系
  - 适合于重构目标系统的连接关系

在系统编程技术简介

- ISP器件的开发软件
  - PDS软件
  - Synario软件
  - ISP Synario System软件

在系统编程技术简介

□PDS 软件

- PDS是设计工具软件，它向用户提供基于PC机的设计输入与器件之间的映射关系
- 利用PDS进行设计时可以采用逻辑描述方式或宏方式
- 逻辑描述方式是最基本的也是最低一级(门、触发器级)的方式
- 宏(MACRO)是一组预先编好，存放在库中的逻辑方程，每个宏器件代表一个逻辑模块，在设计中可作为逻辑器件调用。

在系统编程技术简介

□SYNARIO软件

- SYNARIO是美国Lattice公司和Data I/O公司合作开发的一种运行于PC机Windows环境下的通用电子设计工具软件
- 有一个较完善的宏库，库中包括各种常用逻辑器件和模块。设计中能进行逻辑图输入和ABEL硬件描述语言输入，并包括功能模拟显示和波形显示
- 该软件还具有将多个ABEL设计文件编译成高密度PLD设计的能力，从而开拓出一条将多个低密度PLD设计升级成为高密度PLD设计的捷径

在系统编程技术简介

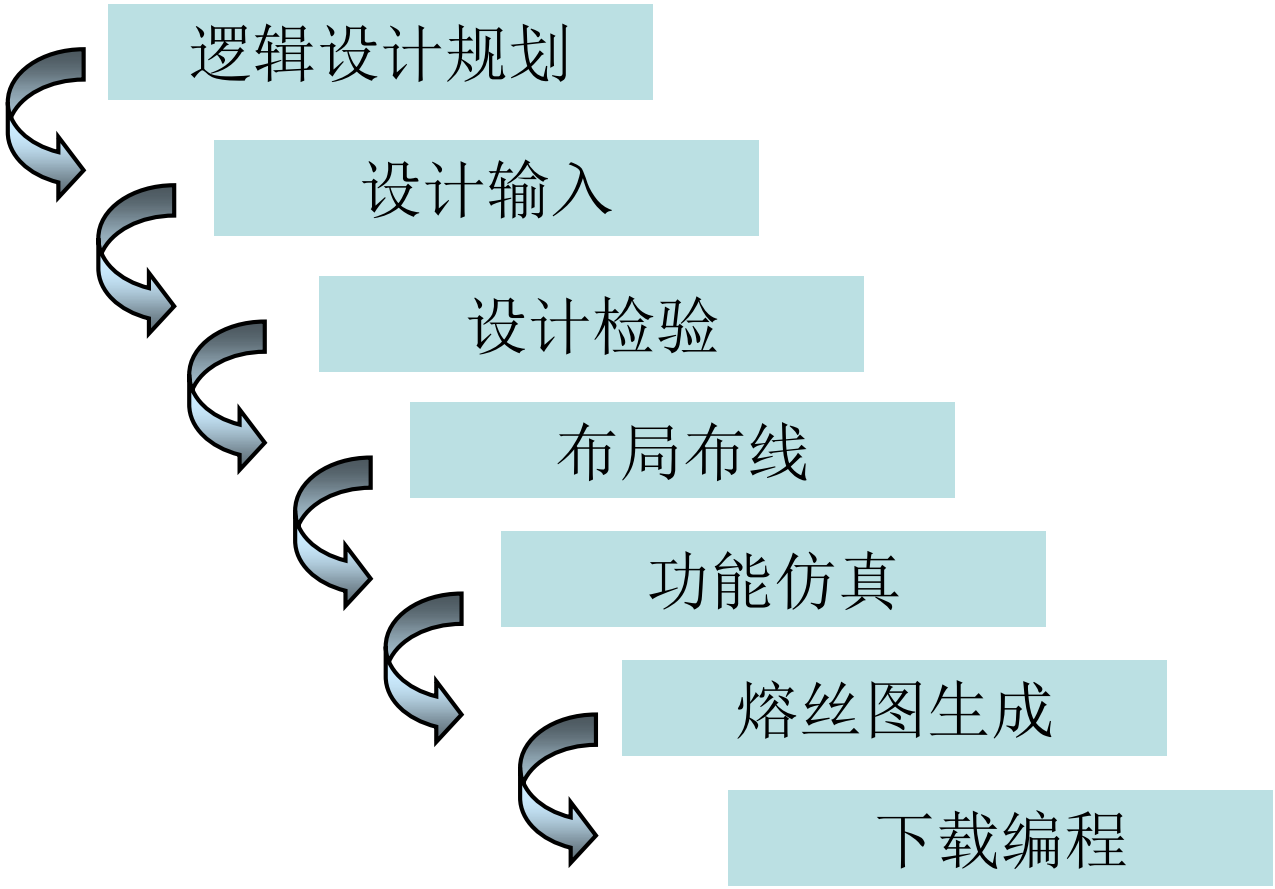
ISP Synario System软件

- ISP Synario System软件是一个基于SYNARIO的完整的在系统编程设计系统
- 具有设计输入、编译、逻辑模拟等功能，支持ispLSI器件、ispGAL器件、ispGDS器件以及全系列GAL器件
- 包括了SYNARIO的全部功能，同样有较为完善的宏库，库中的宏是用ABEL语言编写
- ISP Synario System采用混合输入方式，允许在同一器件的设计中同时采用逻辑图、逻辑方程、真值表和状态图输入，从而使设计输入十分方便

在系统编程技术简介

口设计流程

– 利用ISP器件开发软件进行设计一般步骤为：



在系统编程技术简介

□逻辑设计规划

- 目的：选择合适的ispLSI器件实现预定功能
- 具体事宜
  - 定义I/O端口，以便考虑器件的I/O单元是否够用
  - 进行任务划分，即将要求完成的设计任务分配到各个通用逻辑块GLB中
  - 统计出所需要的GLB数目和I/O单元数目，作为器件型号选择的依据

在系统编程技术简介

口设计输入

- 将设计者所设计的电路，按照开发软件要求的某种形式表达出来，并输入计算机中
- ISP Synario System软件有逻辑图和硬件描述语言(HDL)两种输入方式；PDS软件只有语言输入方式
- 使用ABEL-HDL对逻辑电路进行描述时，应正确使用逻辑变量、逻辑常量、运算符、赋值符和器件的描述方法

在系统编程技术简介

口设计检验

- **语法检验**：列出语法错误
- **设计规划检验**：要求设计资源不超过GLB或IOC的限制，如输入、输出数等
- **逻辑最小化**：简化所有逻辑方程或用户自建的宏，使耗费的资源达到最少

在系统编程技术简介

口设计检验

- **逻辑适配**：确定最小化的逻辑能否与所安排的GLB或IOC适配，然后将逻辑映射到相应的单元中并提供输入到熔丝图的程序
- **全局设计规划检验**：保证总的设计资源不超出器件资源或规划的限制，并查出网络中是否有漏连或信号是否有双重来源等错误

在系统编程技术简介

口布局布线

- 布局布线工作是由软件自动完成的，它能以最优方式对逻辑元件布局和准确地实现元件间的互连
- 布线后生成一个布线报告，提供关于单元输入、GLB平均输入和输出数、扇出数和复用信号，以及设计中GLB和IOC的使用情况等信息

在系统编程技术简介

口逻辑模拟

- 逻辑模拟包括功能模拟和定时模拟
- PDS中没有模拟程序，需要用其他软件处理
- ISP Synario System有自己的模拟程序，模拟过程为：按软件要求建立模拟测试向量源文件、输入测试向量、编译测试向量文件、测试向量模拟和波形图显示

在系统编程技术简介

口熔丝图生成

- 熔丝图生成是指生成JEDEC文件(熔丝图)，它是设计过程的重要目标
- 在生成熔丝图过程中，凡设计中未使用的I/O端，皆被自动接上有源上拉电阻，这样有利于降低功耗和减少噪声

在系统编程技术简介

下载编程

- 下载编程就是将设计生成的熔丝图文件JEDEC装入ispLSI器件中
- 编程时，开发软件在收到设计者发出的DOWNLOAD(下载)命令后，首先检查编程电缆是否接上，计算机与器件之间通信是否正常，然后执行编程操作，并对编程结果进行检验。检验通过后，器件便可在系统中运行