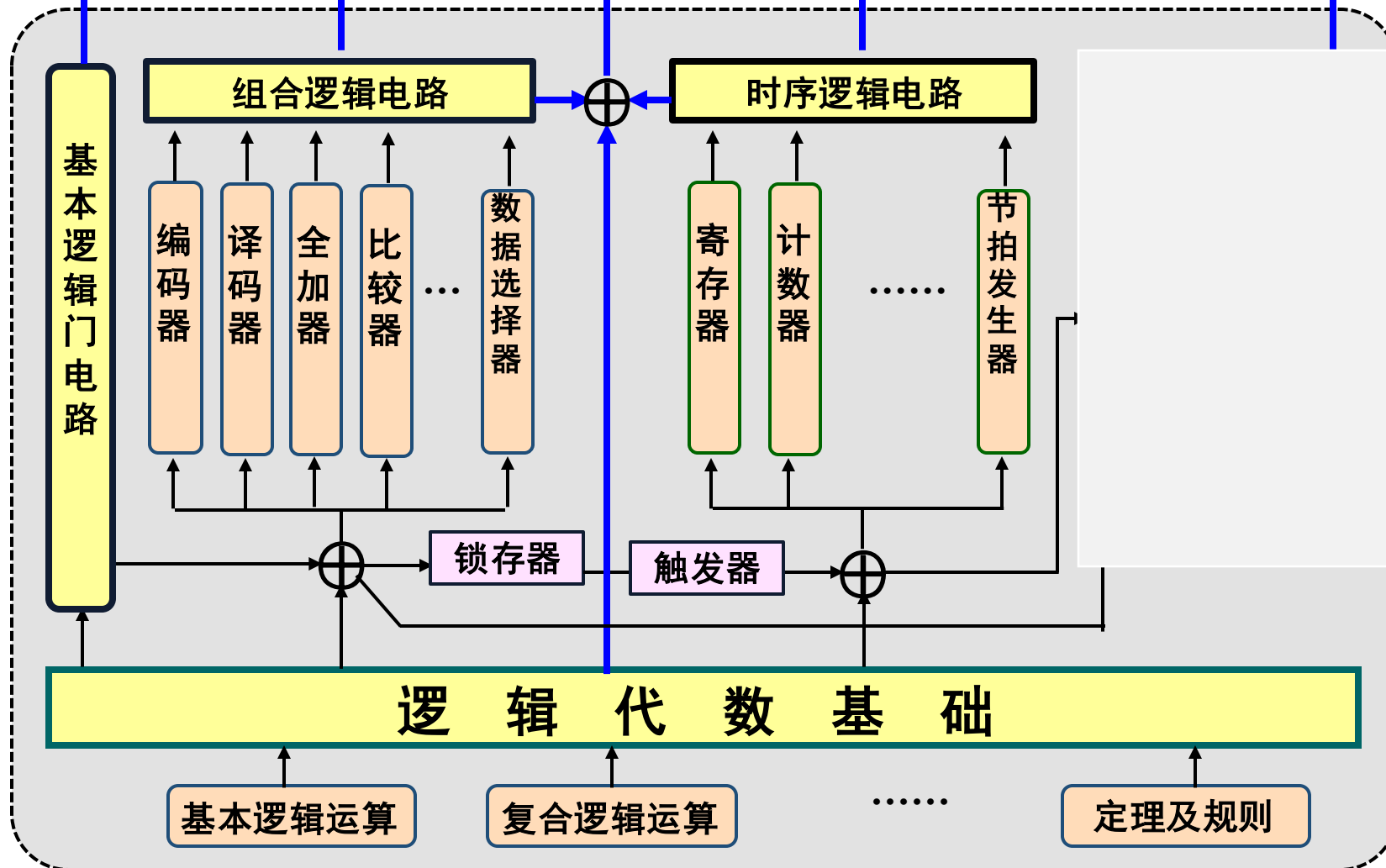


数字逻辑的知识脉络

后续课程：
如计算机组成原理等

数字系统或计算机应用系统

本课程的理论内容



考核方法

讲课 —— 36 学时

实验 —— 20 学时

总计 56 学时

成绩 { 考 试: 50% (会考核10分左右的Verilog内容)
作 业: 20%
实 验: 30%

数制编码

- 数制
 - 编码
- 
- BCD码 (BCD code)
 - 余3码 (Excess-3 code)
 - 格雷码 (Gray code)

BCD码

BCD码 (Binary-Coded Decimal) 也叫二-十进制编码, 用4位二进制数表示1位十进制数

4位二进制码共有 $2^4=16$ 种码组, 在这16种代码中, 可以任选10种来表示10个十进制数码

每位二进制数都带有权值

- 根据权值不同, 称其为:

8421BCD

2421BCD

4221BCD ...

Decimal	8421BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

BCD码

BCD码 (Binary-Coded Decimal) 也叫二-十进制编码，用4位二进制数表示1位十进制数

4位二进制码共有 $2^4=16$ 种码组，在这16种代码中，可以任选10种来表示10个十进制数码

每位二进制数都带有权值

- 根据权值不同，称其为：

8421BCD

2421BCD

4221BCD ...

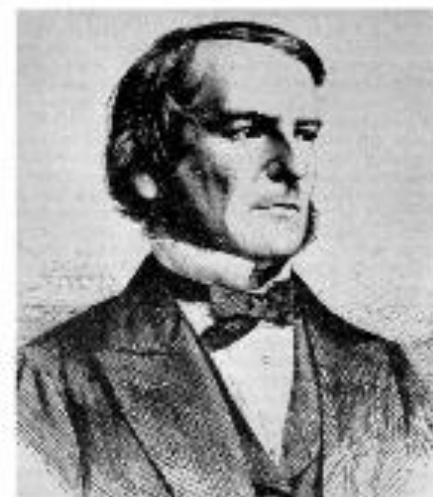
Decimal	8421BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

格雷码 (Gray Code)

- 由贝尔实验室的Frank Gray在1940年代提出的，1953年获得批准的专利“Pulse Code Communication”，当初是为了通信，后来则常用于模拟—数字转换中。
- 在一组数的编码中，若任意两个相邻的代码只有一位二进制数不同，则称这种编码为格雷码 (Gray Code)
- 另外由于最大数与最小数之间也仅一位数不同，即“首尾相连”，因此又称循环码或反射码。
- 格雷码有多种编码形式——典型格雷码。

Unit 2 布尔代数

- 逻辑运算
- 布尔表达式和真值表
- 逻辑代数定理及规则
- 代数化简法



George Boole

Laws and Theorems

- 与普通代数相似的定理 交换律, 分配律, 结合律

- *Two or three variables is involved*

交换律

$$(T6) \mathbf{A+B=B+A}$$

$$(T6D) \mathbf{A \cdot B = B \cdot A}$$

结合律

$$(T7) \mathbf{(A+B)+C=A+(B+C)}$$

$$(T7D) \mathbf{(A \cdot B) \cdot C = A \cdot (B \cdot C)}$$

分配律

$$(T8) \mathbf{A \cdot (B+C) = AB+AC}$$



真值表

第二分配律

$$(T8D) \mathbf{A+BC=(A+B) \cdot (A+C)}$$

$- A+AC+AB+BC = A+BC$

普通代数
不支持



■ *Two or three variables is involved*

$$(T9) \quad \mathbf{A + AB = A}$$

$$(T9D) \quad \mathbf{A(A + B) = A} \quad (\text{吸收律})$$

$$(T10) \quad \mathbf{AB + A\bar{B} = A}$$

$$(T10D) \quad \mathbf{(A + B)(A + \bar{B}) = A} \quad (\text{合并律})$$

$$(T11) \quad \mathbf{A + \bar{A}B = A + B}$$

(消除律)

$$(T12) \quad \mathbf{AB + \bar{A}C + BC = AB + \bar{A}C}$$

(蕴含律)

$$(T12D) \quad \mathbf{AB + \bar{A}C + BCD = AB + \bar{A}C}$$

$$(T12D)' \quad \mathbf{(A + B)(B + C)(A' + C) = (A + B)(A' + C)}$$

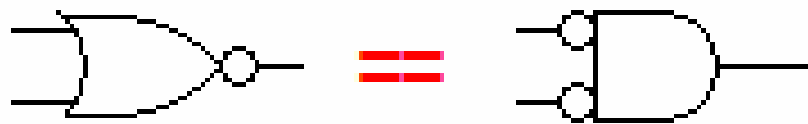
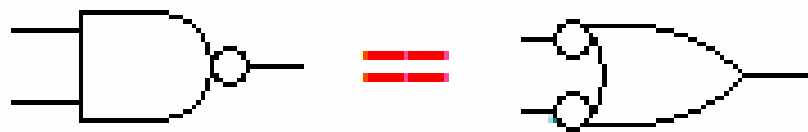
$$(T13) \quad \mathbf{\overline{A\bar{B} + \bar{A}B} = \bar{A}\bar{B} + AB}$$

定理和规则

- *N variables is involved*

—德摩根定理 (DeMorgan's Laws) 😊

$$(13) \quad \overline{A+B} = \bar{A} \cdot \bar{B} \quad (13)' \quad \overline{A \cdot B} = \bar{A} + \bar{B}$$



定理和规则

特殊定理

——对偶规则



① $F \xleftrightarrow{\text{Dual Rule}} (F)^D$

② 两个逻辑表达式相等，它们的对偶也相等

$$A + BCD = (A + B)(A + C)(A + D)$$



Dual Rule



Dual Rule

$$A \cdot (B + C + D) = AB + AC + AD$$

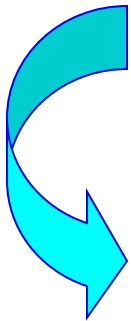
代数法化简



最简 (Minimum Expressions) ?

① 与项 (和项) 的个数最少

② 每个与项 (和项) 中变量的个数最少



minimum cost

① 逻辑门的数量最少

② 逻辑门的输入个数最少

目的:

- 降低成本
- 提高可靠性

Methods {
■ 代数法 (Algebraic techniques)
■ 卡诺图法 (K. map method)

重要的三个规则

$$(T8D) \quad A+BC=(A+B) \cdot (A+C)$$

$$(T11) \quad A+\bar{A}B = A+B$$

$$(T12D) \quad AB+\bar{A}C+BC = AB+\bar{A}C$$

Unit 3 布尔代数的应用

- 布尔代数的应用
- 最大项、最小项表达式
- 不完全给定函数

组合逻辑电路的设计方法

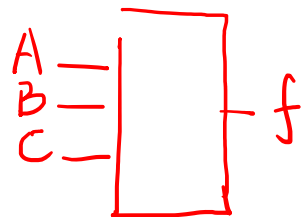
已知 —— 设计要求
待求 —— 逻辑图

• 步骤:

1. 根据设计要求确定 —— 真值表
2. 根据真值表 —— 卡诺图(表达式)
3. 化简
4. 按设计要求, 变换逻辑表达式
5. 画出逻辑图



设计一个比赛需要用到的三人投票器，如果有两人或者三人投赞成票，则显示通过；否则不通过。请设计逻辑电路。



A=1 投票 A=0 不投票
f=1 通过 f=0 不通过

① 真值表

输入 ABC	输出 f
000	0
001	0
010	0
011	1
100	0
101	1
110	1
111	1

② 标准与或式

$$f = A'BC + AB'C + \underline{ABC'} + \underline{ABC}$$

③ 化简

$$= A'BC + \underline{AB'C} + AB$$

$$= A'BC + A(\underline{B'C} + B)$$

$$= A'BC + A(\underline{C} + B)$$

$$= \underline{A'BC} + \underline{AC} + AB$$

$$= \underline{(A'B + A)C} + AB$$

$$= \underline{(A+B)C} + AB = \underline{AC + BC} + AB$$

④ 逻辑图

正常使用主观题第2.0以上版本雨课堂

作答

最大项Maxterm、最小项Minterm的定义

Row No.	A B C	Minterms	Maxterms
0	0 0 0	$A'B'C' = m_0$	$A + B + C = M_0$
1	0 0 1	$A'B'C = m_1$	$A + B + C' = M_1$
2	0 1 0	$A'BC' = m_2$	$A + B' + C = M_2$
3	0 1 1	$A'BC = m_3$	$A + B' + C' = M_3$
4	1 0 0	$AB'C' = m_4$	$A' + B + C = M_4$
5	1 0 1	$AB'C = m_5$	$A' + B + C' = M_5$
6	1 1 0	$ABC' = m_6$	$A' + B' + C = M_6$
7	1 1 1	$ABC = m_7$	$A' + B' + C' = M_7$

- n个变量组成的最小项：是一个与项（包含n个变量）
- n个变量组成的最大项：是一个或项（包含n个变量）
- 每个变量或者以原变量的形式、或者以反变量的形式出现，并且只出现一次。
- n个变量能组成的最小（大）项的个数是 2^n

最小项和最大项的性质

1. 最小项的反是最大项，最大项的反是最小项；

$$\overline{\overline{A}\overline{B}\overline{C}} = \overline{m_0} = A + B + C = M_0$$

$$\overline{A + \overline{B} + \overline{C}} = \overline{M_3} = \overline{A}BC = m_3$$

2. 全部最小项之和恒等于“1”；

$$m_0 + m_1 + m_2 + m_3 = 1$$

3. 全部最大项之积恒等于“0”；

$$M_0 M_1 M_2 M_3 = 0$$

4. 一部分最小项之和的反等于其余所有最小项之和

$$\overline{m_1 + m_2} = m_0 + m_3 \quad \overline{m_0} = m_1 + m_2 + m_3$$

最小项和最大项的性质——续

5. 两个不同的最小项之积恒等于“0”；

例如： $ABC \cdot A\overline{B}\overline{C} = 0$

6. 两个不同的最大项之和恒等于“1”；

例如： $(A + B + C) + (A + B + \overline{C}) = 1$

7. 与或标准型

$$Y = \sum m_i = \sum m(0,1,4,6,7) = m_0 + m_1 + m_4 + m_6 + m_7$$

8. 或与标准型

$$Y = \prod M_i = \prod M(0,1,4,6,7) = M_0 M_1 M_4 M_6 M_7$$

Unit 4 卡诺图 Karnaugh Maps

- 开关函数的最简形式
- 多变量卡诺图
- 填写卡诺图
- 卡诺图化简法

开关函数的最简形式

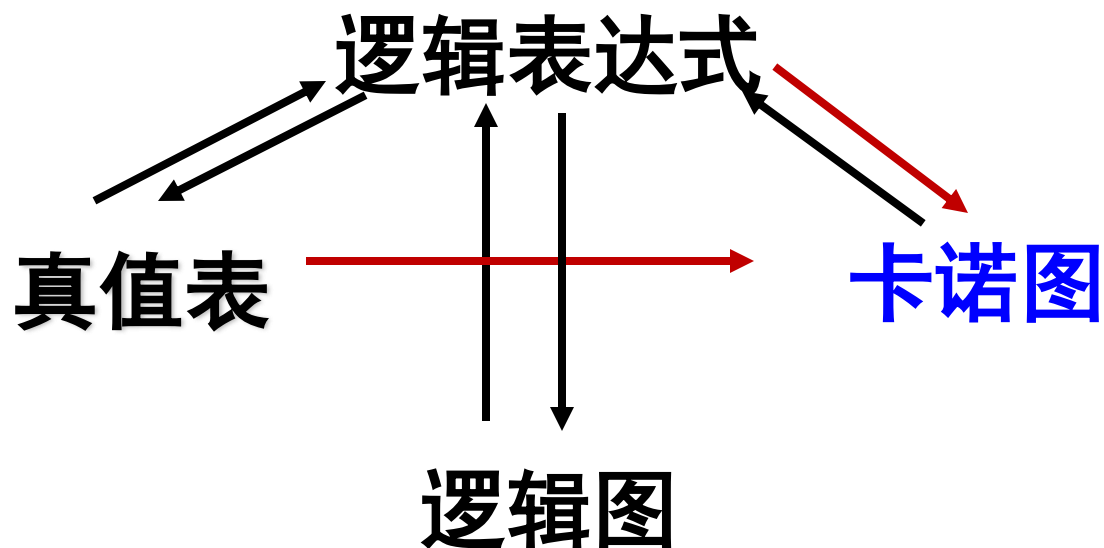
一个最简表达式中

- ① 逻辑门的数量最少
- ② 逻辑门的输入个数最少

与最小项（最大项）表达式不同

- 最简表达式**不一定是唯一的**.
- 但最简表达式的实现代价是相同的（逻辑门的数量相同、输入变量的个数相同）

逻辑函数的表达方式之一



- 化简三变量或者四变量的逻辑函数时，卡诺图特别有用！

卡诺图的性质

- 卡诺图通常为正方形或矩形均匀分成 2^n 个小格，每个小格代表一个**最小项**。
- 单元格对应的最小项按**格雷码**摆放
- 任何两个相邻单元格对应的最小项只有一个变量取值不同

1. 两变量 K. Map

$$F=f(AB)$$

	\bar{B}	B
\bar{A}	$\bar{A}\bar{B}$	$\bar{A}B$
A	$A\bar{B}$	AB

(a)

B	0	1
A 0	0 0 0	0 1 1
1	1 0 2	1 1 3

(b)

B	0	1
A 0	0	1
1	2	3

如何从卡诺图读最简与或式

Step ①: 画圈

- a). 将**相邻**为**1**的小方格圈在一起。(小方格的个数必须为 2^m , $m=0,1,2,\dots$)
- b). 圈**越大越好**
- c). 小方格可以**重复**使用



Adjacent: 紧靠在一起的、行列首尾的、对称的
(本质上: 满足格雷码特点)

A \ BC	00 01 11 10			
	00	01	11	10
0	0	0	1	0
1	0	1	1	1

AB \ CD	00 01 11 10			
	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	0	0	0
10	1	0	0	1

AB \ CD	00 01 11 10			
	00	01	11	10
00	0	1	1	0
01	0	0	0	0
11	0	0	0	0
10	0	1	1	0

如何从卡诺图读最简与或式

Step ② : 每个圈代表一个与项

观察 $\left. \begin{array}{l} \text{Left} \\ \text{Top} \end{array} \right\}$ 变量取值不同——消去

变量取值相同 $\left\{ \begin{array}{l} 1: \text{原变量} \\ 0: \text{反变量} \end{array} \right.$

		CD			
		00	01	11	10
AB	00	1	0	0	1
	01	0	0	0	0
	11	0	0	0	0
	10	1	0	0	1

		CD			
		00	01	11	10
AB	00	0	1	1	0
	01	0	0	0	0
	11	0	0	0	0
	10	0	1	1	0

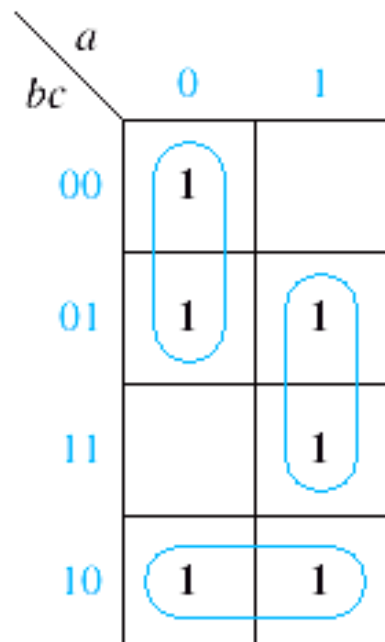
如何从卡诺图读最简与或式

Step ③: 将所有的与项相加

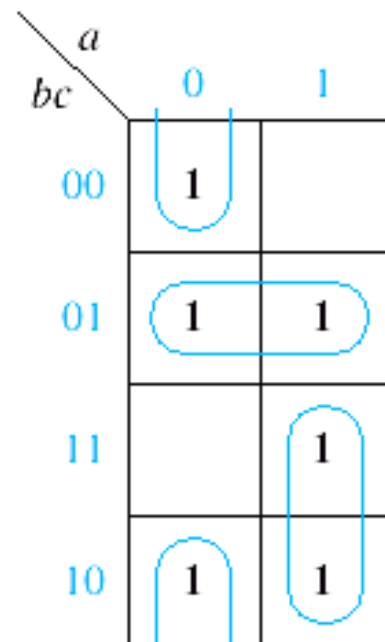
		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	1	0	1
	01	1	1	0	0
	11	0	0	1	1
	10	1	0	1	1

$$F = \bar{A}\bar{C} + AC + \bar{B}\bar{D}$$

如何从卡诺图读最简与或式



$$F = a'b' + bc' + ac$$



$$F = a'c' + b'c + ab$$

The two minimum solutions For F

与最小项（最大项）表达式不同

- 最简表达式**不一定是唯一的**。
- 但最简表达式的**实现代价是相同的**（逻辑门的数量相同、输入变量的个数相同）

Unit 5 多级门电路

- 多级门电路（Multi-Level Circuits）
- 两级门电路的设计
- 多输出电路的设计
- 多级门电路实例

多级门电路

前提：忽略输入端原、反变量的差别。

门的级数——

电路输入与输出之间串联的门的最大数值

□ 二级电路

AND-OR 电路（积之和）

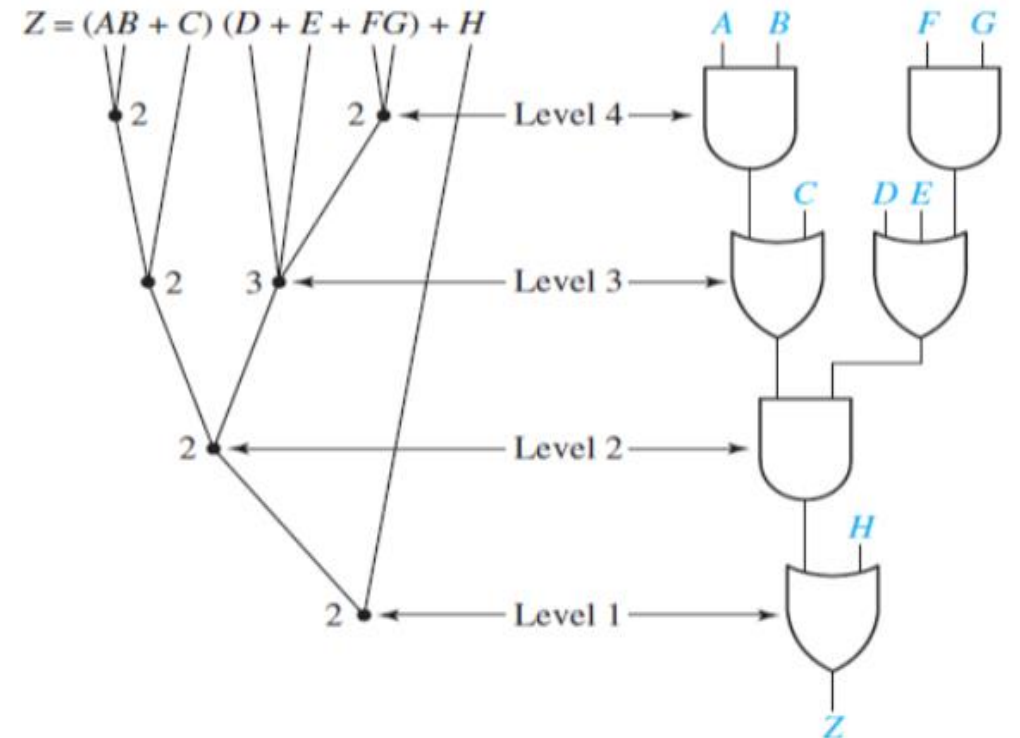
OR-AND 电路（和之积）

□ 三级电路

OR-AND-OR 电路

□ 各门没有特定的排列顺序

□ 输出门可以使与门也可以是或门



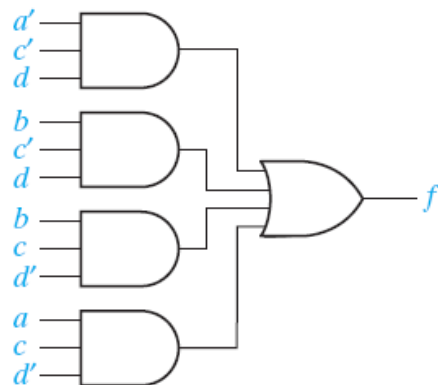
多级门电路

1. 二级电路

AND-OR 电路（积之和）

$$f = a'c'd + bc'd + bcd' + acd'$$

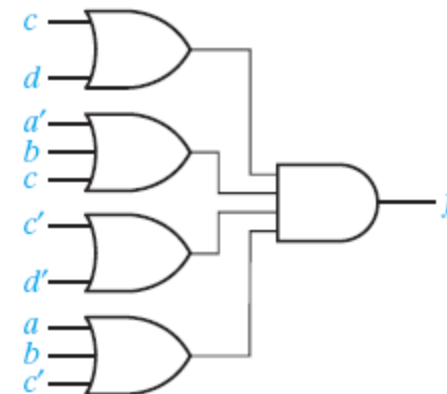
5个门, 16 个输入端



OR-AND 电路（和之积）

$$f = (c + d)(a' + b + c)(c' + d')(a + b + c')$$

5个门, 14 个输入端



		<i>ab</i>			
		00	01	11	10
<i>cd</i>	00	0	0	0	0
	01	1	1	1	0
	11	0	0	0	0
	10	0	1	1	1

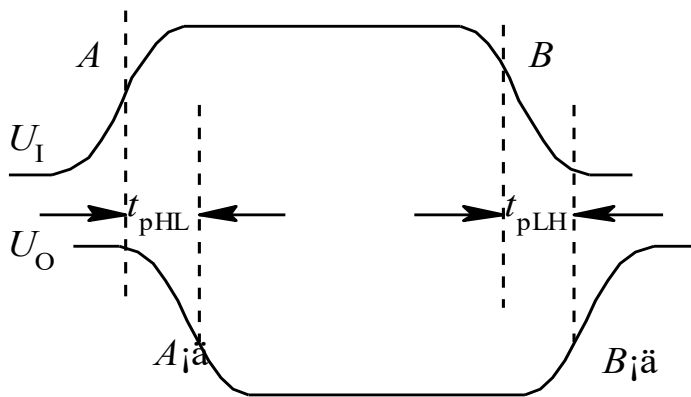
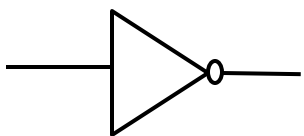
Unit 6 险象及消除

- 使用有限扇入门设计组合电路
- 组合电路中的险象
 - 门延迟
 - 静态冒险
- 险象判断及消除
 - 代数法
 - 卡诺图法

组合电路中的险象

1. 门延迟

当输入发生变化，逻辑门的输出不会同步发生改变



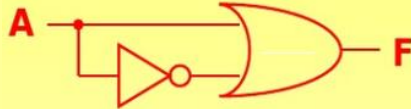
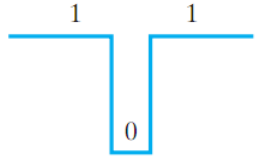
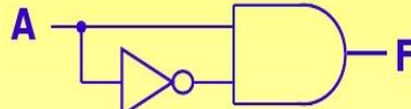
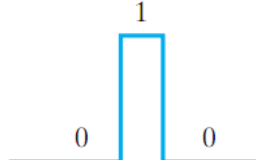
可能引发非预期的尖峰干扰

- 对于组合逻辑电路，多数情况下可以忽略门的延迟。
- 但是，门的延迟对时序电路的影响不容忽视

组合电路中的险象

当一个逻辑门的两个输入端的信号同时向**相反**方向变化，则该电路存在**竞争**。

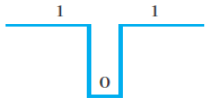
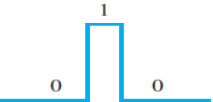
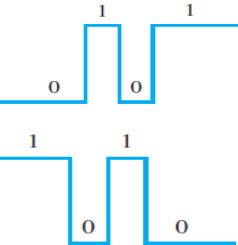
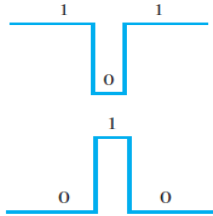
两路信号到达逻辑门的时间存在差异。

存在竞争的电路	险象
	
	

逻辑门因输入端的竞争而导致输出了不应有的尖峰干扰脉冲（又称过渡干扰脉冲）称为**冒险**。

组合电路中的险象

2. 险象

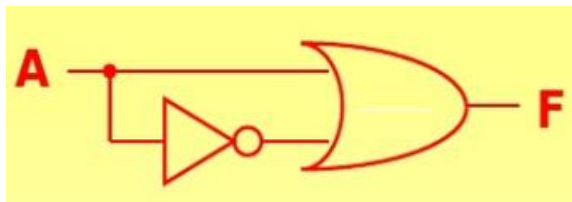
险象类型	概念		输出波形
■ 静态冒险	输入信号发生一次变化只引起一个错误信号脉冲	■ 静态1冒险	
		■ 静态0冒险	
■ 动态冒险	输入信号发生一次改变引起多个错误信号脉冲		
■ 功能冒险	多个输入信号的变化不同步而产生的错误信号脉冲		

险象判断及消除

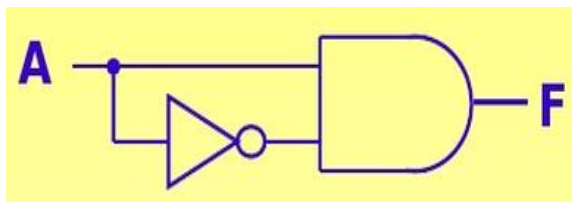
3. 险象的判断——代数法

检查表达式中是否存在某个变量 X ，它同时以原变量和反变量的形式出现；并能在特定条件下简化成下面形式之一：

■ $X + \overline{X}$



■ $X \cdot \overline{X}$



险象判断及消除

4. 险象的判断—— k. maps

化简后是否存在相切的卡诺圈

C \ AB				
	00	01	11	10
0		1	1	
1	1	1		

$$F1 = A' \cdot C + B \cdot C'$$

$$\text{When } A = 0, B = 1: F1 = C + C'$$

C \ AB				
	00	01	11	10
0	1	1		
1		1	1	

$$F2 = (A' + C) \cdot (B + C')$$

$$\text{When } A = 1, B = 0: F2 = C \cdot C'$$

险象判断及消除

5. 险象的消除

① 添加卡诺圈

C \ AB	AB			
	00	01	11	10
0		1	1	
1	1	1		

$$F1 = A' \cdot C + B \cdot C' + \boxed{A' \cdot B}$$

When $A = 0, B = 1: F1 = 1$

C \ AB	AB			
	00	01	11	10
0	0	0		
1		0	0	

$$F2 = (A' + C) \cdot (B + C') \cdot \boxed{(A' + B)}$$

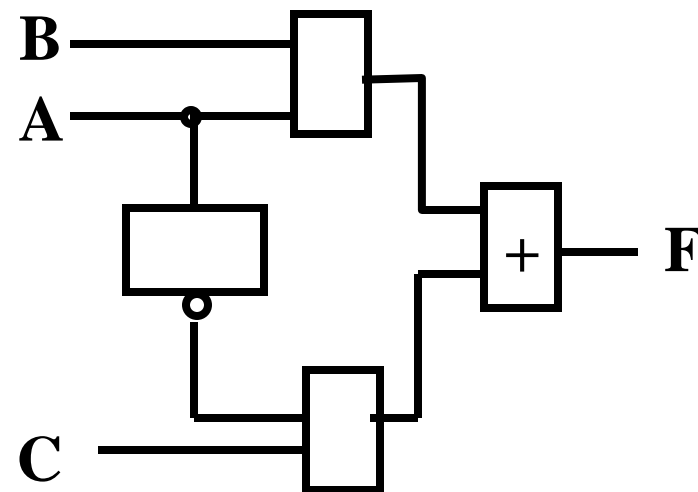
When $A = 1, B = 0: F2 = 0$

险象判断及消除

$$F = AB + \overline{A}C$$

② 添加冗余项: BC

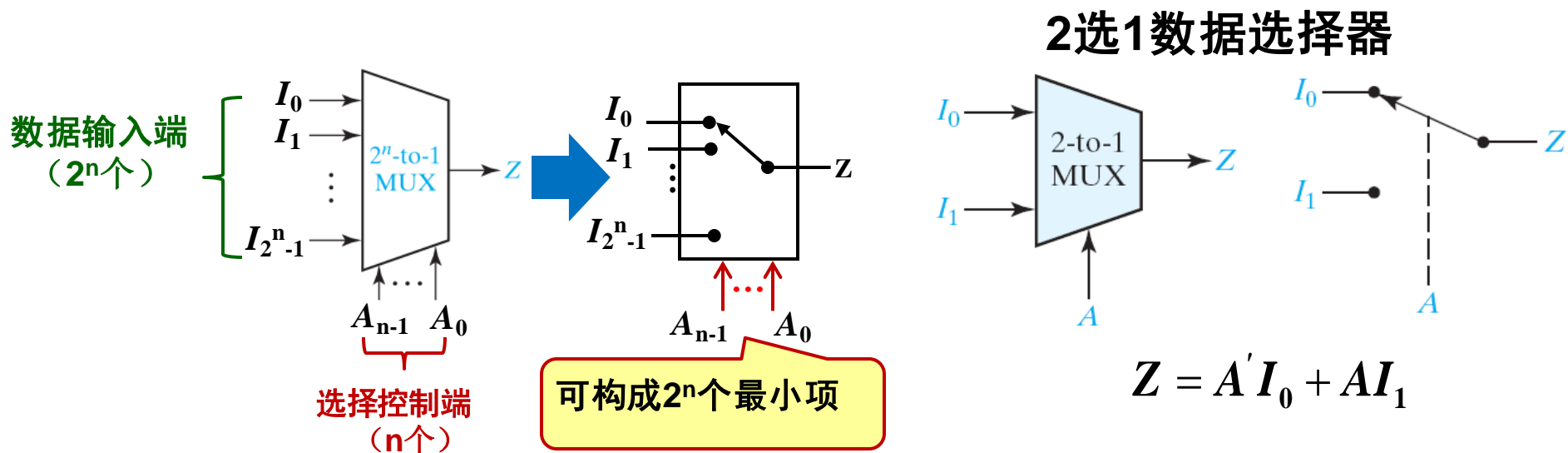
$$F = AB + \overline{A}C + \boxed{BC}$$



Unit 7 组合逻辑元件

- 多路复用器(multiplexers)
- 三态器件(Three-state Buffer)
- 译码器(Decoders)
- 编码器(Encoders)
- 奇偶校验器
- 比较器
- 只读存储器(ROM)
- 利用MSI设计组合逻辑电路

数据选择器/多路开关



$$Z = \sum_{k=0}^{2^n-1} m_k I_k$$

控制端最小项 m_k 的序号 K , 指向了第 K 路数据输入端 I_k 。

m_k —— n 个控制变量的最小项

I_k ——第 k 路数据输入

数据选择器的功能:

- ① 从多路输入中选择一个送往输出端 (2^n 选1);
- ② 选择哪一路输入送到输出端由控制信号决定;

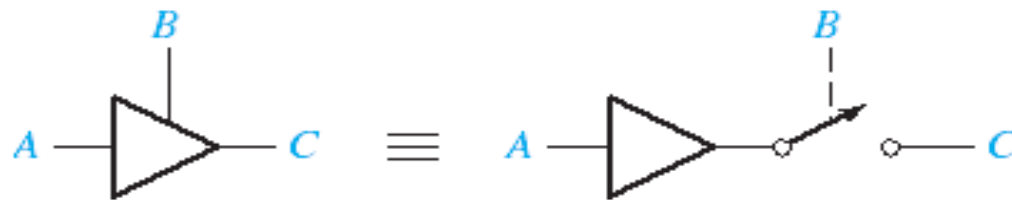
用途: 实现多通道的数据传送;

三态门(Three-State Buffers)

三态——

- 0
- 1
- Z: 高阻态

- 包括三态恒等门、三态非门、三态与非门等，**缓冲器**（驱动门）。
- 用途之一：可用来增强输出驱动能力



三态门（恒等）

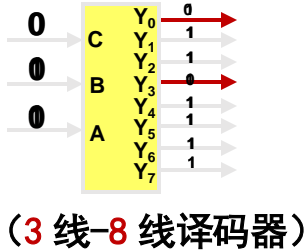
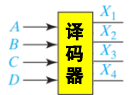
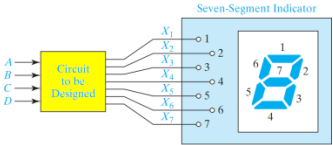
B: 使能端，高电平有效

真值表

B	A	C
0	0	Z
0	1	Z
1	0	0
1	1	1

译码器及分类

- ◆ 特点：多输入、多输出的组合逻辑电路
- ◆ 功能：将一种编码转换为另一种编码

分类	特点	译码演示
二进制译码器	<ul style="list-style-type: none">• n 位二进制码• N位 ($N=2^n$)，每根输出线都与一个输入最小项唯一对应（输出线编号值=最小项编号值）• 每个最小项输入，只能使 N 根输出线中的一个输出有效 → N ($N=2^n$) 中取一译码器，也称最小项译码器。	 <p>(3 线-8 线译码器)</p>
代码转换译码器	从一种编码转换为另一种编码 (例如：8421BCD码→余3码)	
显示译码器	将输入的编码信号转换为十进制码或其它特定编码，用来驱动显示器件显示相应的文字符号。	

地址译码例题

■ 地址译码

图示电路的整个地址译码范围？

各个外设的地址译码范围？

整个译码器的地址译码范围：

A ₁₅	A ₁₄	A ₁₃	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
5	4	3	2	1	0										
0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

最小取值 1000H

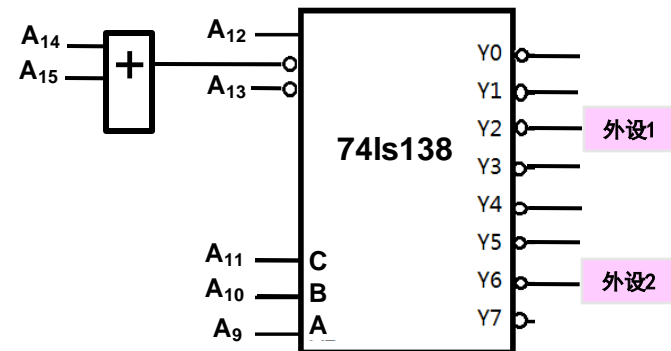
最大取值 1FFFH

外设1的地址译码范围：

A ₁₅	A ₁₄	A ₁₃	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
5	4	3	2	1	0										
0	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1

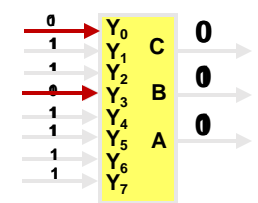
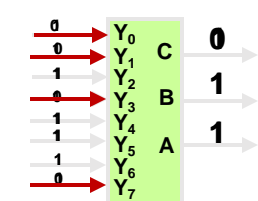
最小取值 1400H

最大取值 15FFH



编码器

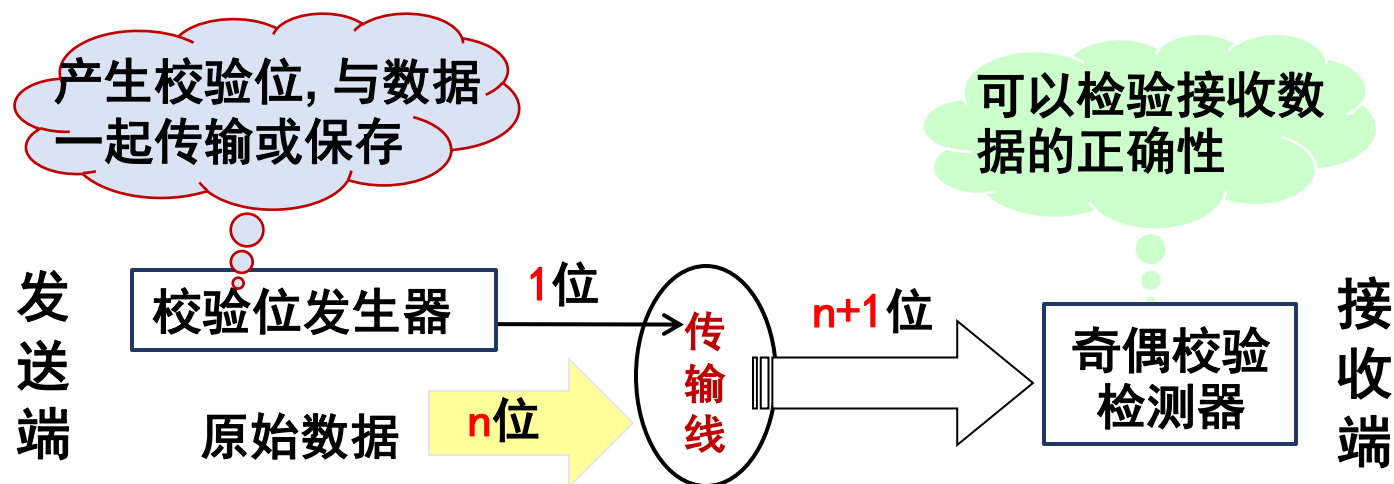
- ◆ 特点：多输入、多输出的组合逻辑电路
- ◆ 功能：将二进制码按照一定规律编排，使其具有特定含义，与译码器互逆。

常用编码器	特点	编码演示
普通编码器 (二进制编码器)	输入： N 位，任何时刻 N 根输入线中只能有一个输入有效， N ($N=2^n$) 中取一。 输出： n 位二进制码	 <p>(8 线-3 线编码器)</p>
优先编码器	允许同时输入两个以上的有效编码输入信号，优先编码器能按照预先设定的优先级别，只对其中优先级最高的输入进行编码。	 <p>(8 线-3 线优先编码器)</p>

奇偶校验器

- 用来检查数据传输和存取过程中是否产生错误的组合逻辑电路。
(就是检测数据中包含“1”的个数是奇数还是偶数)
- 广泛用于计算机的内存储器以及磁盘等外部设备中

{ 奇偶校验发生器：可产生奇偶校验位，与数据一起传输或保存
奇偶校验检测器：可以检验所接受数据的正确性



被校验的原始数据和1位校验位组成 $n+1$ 位校验码。



校验码: $n+1$ 位

偶校验位逻辑值的表达式:

$$P_E = A_3 \oplus A_2 \oplus A_1 \oplus A_0$$

奇校验位逻辑值的表达式:

$$P_O = A_3 \oplus A_2 \oplus A_1 \oplus A_0$$

偶校验位逻辑值电路是在奇校验位逻辑值电路输出端加非门实现

4位二进制数校验码真值表

$A_3A_2A_1A_0$	P_E	P_O
0 0 0 0	0	1
0 0 0 1	1	0
0 0 1 0	1	0
0 0 1 1	0	1
0 1 0 0	1	0
0 1 0 1	0	1
0 1 1 0	0	1
0 1 1 1	1	0
1 0 0 0	1	0
1 0 0 1	0	1
1 0 1 0	0	1
1 0 1 1	1	0
1 1 0 0	0	1
1 1 0 1	1	0
1 1 1 0	1	0
1 1 1 1	0	1

奇偶校验器一般由异或门构成

异或门真值表

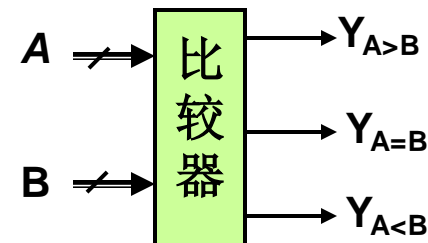
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

异或门特性

- 两个输入中有奇数个“1”，输出为1；有偶数个“1”，输出为0。
- 扩展: n 个1位二进制数中有奇数个“1”，输出为1；有偶数个“1”，输出为0。

数值比较器

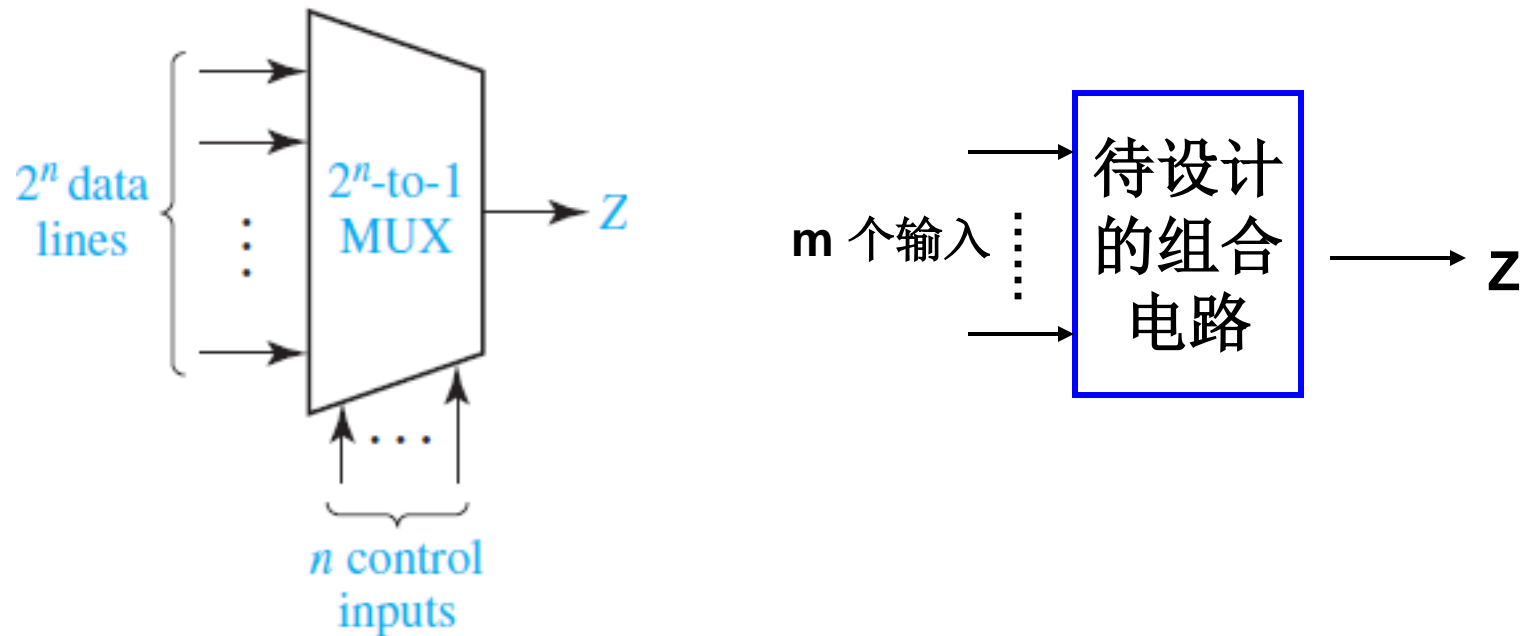
- 计算机中对数据的基本处理方法
 - 加、减、乘、除
 - 比较运算
- 数值比较器：一种关系运算电路
 - 能对2个 n 位二进制数 A 和 B 进行比较的多输入、多输出的组合逻辑电路
 - 比较结果： $Y_{A>B}$ 、 $Y_{A<B}$ 、 $Y_{A=B}$



1.用数据选择器来实现组合逻辑电路

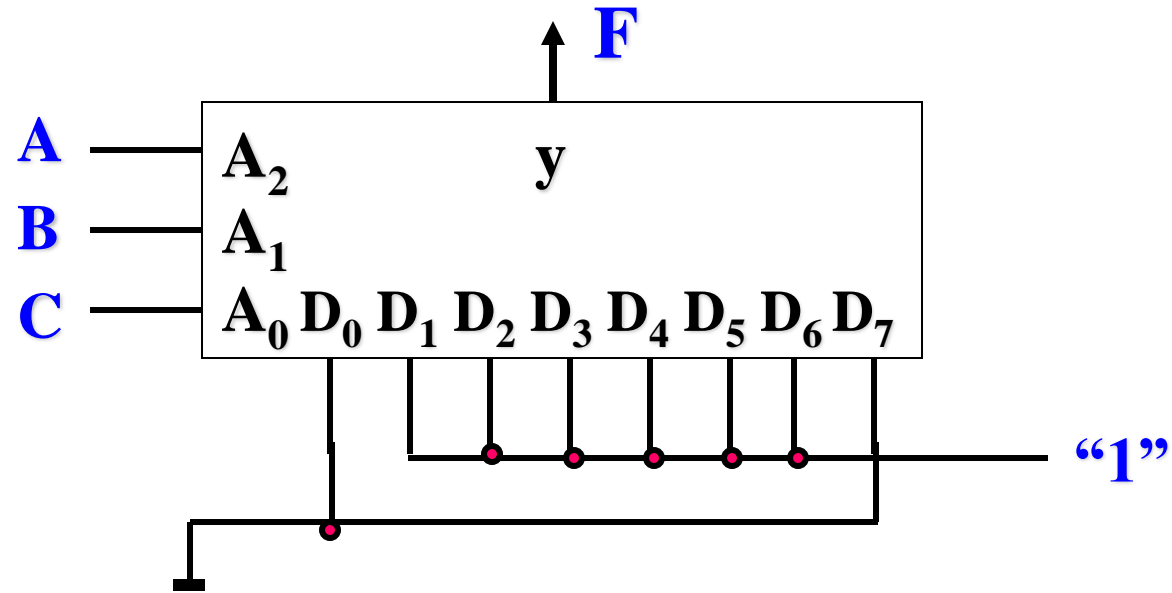
(1) $m = n$

数据选择器的控制端个数
组合电路的输入变量个数



Use 8 to 1 MUX realize $F = A\bar{B} + \bar{A}C + B\bar{C}$

$A_2A_1A_0$	y
0 0 0	D_0
0 0 1	D_1
0 1 0	D_2
0 1 1	D_3
1 0 0	D_4
1 0 1	D_5
1 1 0	D_6
1 1 1	D_7



A	BC			
	00	01	11	10
0	0	1	1	1
1	1	1	0	1

K.Map of F

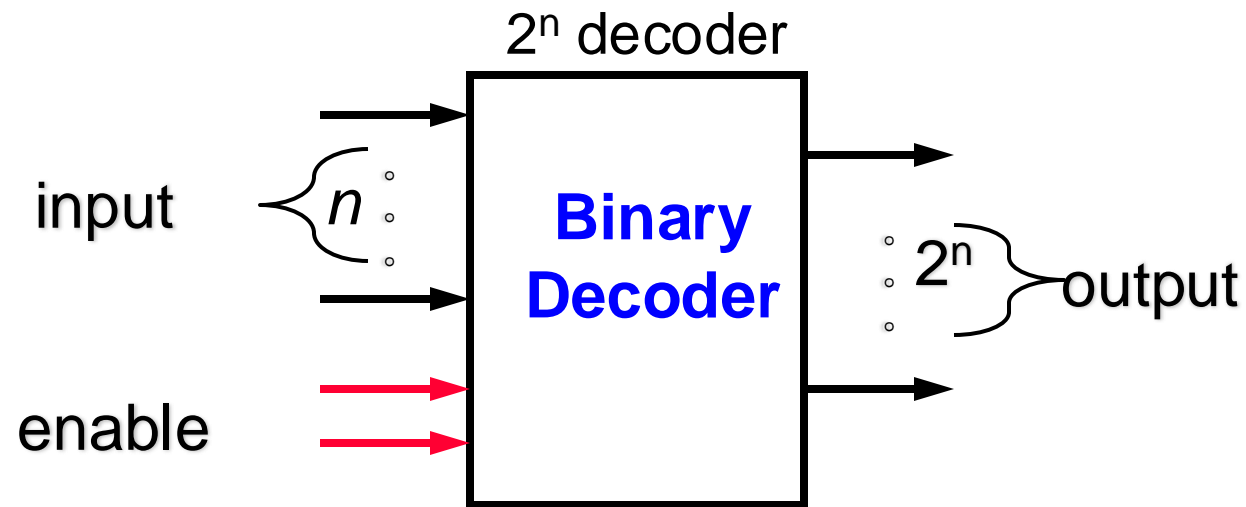
A_2	A_1A_0			
	00	01	11	10
0	D_0	D_1	D_3	D_2
1	D_4	D_5	D_7	D_6

K.Map of MUX



2.用译码器来实现组合逻辑电路

MSI blocks {
▪ Multiplexers
▪ Decoders



$$y_i = m_i, \quad i = 0 \text{ to } 2^n - 1 \quad (\text{noninverted outputs})$$

$$y_i = m_i' = M_i, \quad i = 0 \text{ to } 2^n - 1 \quad (\text{inverted outputs})$$

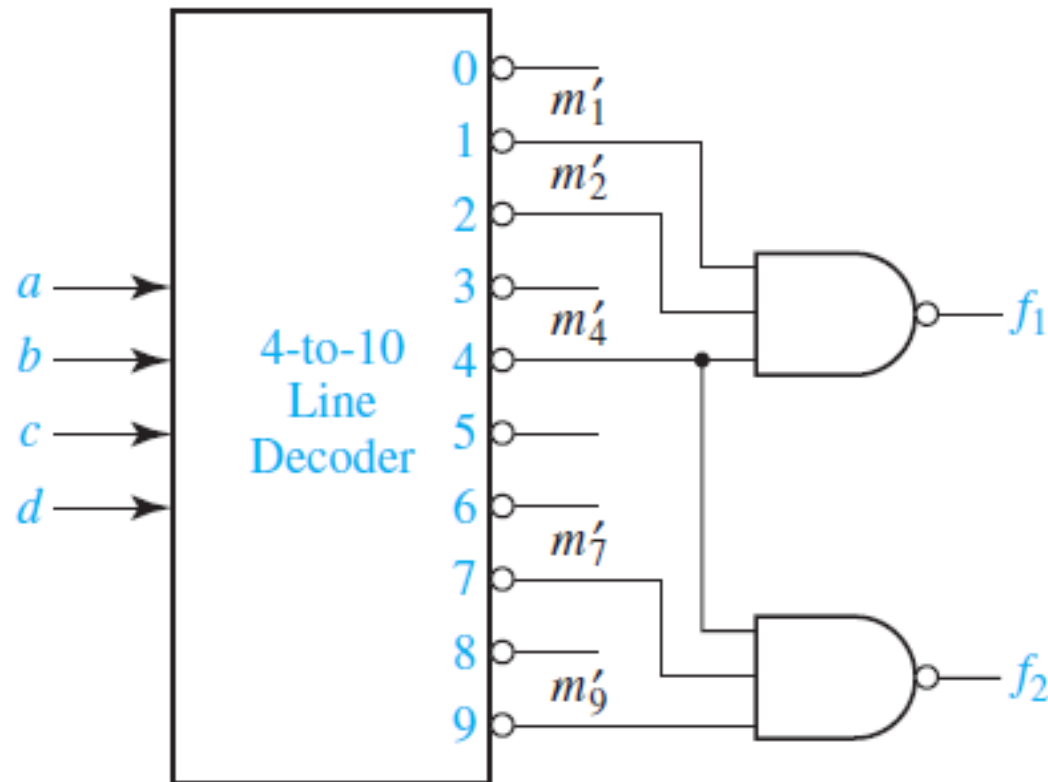
2.用译码器来实现组合逻辑电路

$$f_1(a, b, c, d) = m_1 + m_2 + m_4$$

$$f_2(a, b, c, d) = m_4 + m_7 + m_9$$

$$f_1 = (m'_1 m'_2 m'_4)'$$

$$f_2 = (m'_4 m'_7 m'_9)'$$

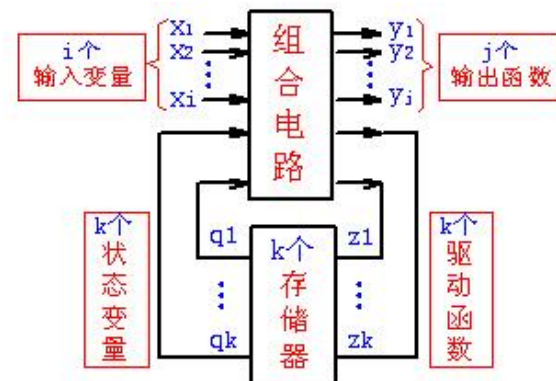


Unit 8 时序逻辑元件

- 锁存器 (Latch)
- 触发器 (Flip-Flop)
- 带附加输入端的边沿触发器
- 触发器类型转换

组合逻辑电路 vs 时序逻辑电路

- **锁存器和触发器**是构成存储电路的基本元件
- 两个重要概念：现态（原态）和 次态（新态）



构成		定义	结构	电路框图	逻辑函数表达式
数字逻辑电路	组合逻辑电路	任意时刻的输出—— ■ 仅与当前时刻的输入有关 $Z_m = f_m(x_1, \dots, x_n)$	不包含存储元件		只有一组： $Z_m = f_m(x_1, \dots, x_n)$
	时序逻辑电路	任意时刻的输出与以下均有关： ■ 当前时刻的输入 ■ 电路过去（上一个时刻）的工作状态 $Z_m = f_m(x_1, \dots, x_n, y_1, \dots, y_s)$	包含存储元件		有三组： 输出方程, 驱动方程, 状态方程： $Z_m = f_m(x_1, \dots, x_n, y_1, \dots, y_r)$ $Y_r = g_r(x_1, \dots, x_n, y_1, \dots, y_s)$ $Y_s^{n+1} = q_s(x_1, \dots, x_n, Y_1^n, \dots, Y_s^n)$

锁存器和触发器

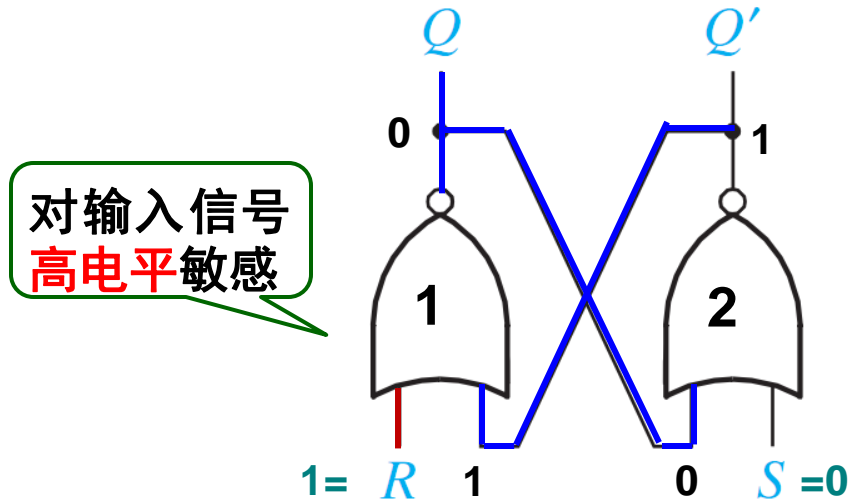
- 锁存器：没有时钟输入端
- 触发器：有时钟输入端，并且只在时钟信号到来时，才发生状态转换

◆ 锁存器与触发器的特性（双稳态）

1. 有两个互补的输出端 Q 和 Q'
2. 有两个稳定的状态: 0态, 1态
3. 在外界信号的刺激下, 可以从一个稳定状态转变到另一个稳定状态。
4. 没有（或无效的）外界信号刺激, 维持当前状态不变。

基本RS锁存器（触发器的鼻祖）

(1) 电路构成（或非门）



$Q (Q_n)$ ——现态

$Q^+ (Q_{n+1})$ ——次态

$Q = 0 (\bar{Q} = 1)$: state 0

$Q = 1 (\bar{Q} = 0)$: state 1

R : 置0端(Reset the output to $Q=0$)

S : 置1端(Set the output to $Q=1$)

(2) 功能表

置0端 R	置1端 S	现态 Q_n	次态 Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	—
1	1	1	—

保持

置 1

置 0

× 不允许

RS对同时取
1互斥

置0端 R	置1端 S	次态 Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	—

输入高电平
有效

RS锁存器小结

- 优点：结构简单

- 缺点：

 - ① 输入存在约束，使用不便；

 - ② 状态改变由输入直接控制。给使用带来局限性。

- 用途：记忆输入状态

- 基本RS锁存器是众多触发器的鼻祖

 - 其余的触发器都是在其基础上逐步改进和完善后形成的

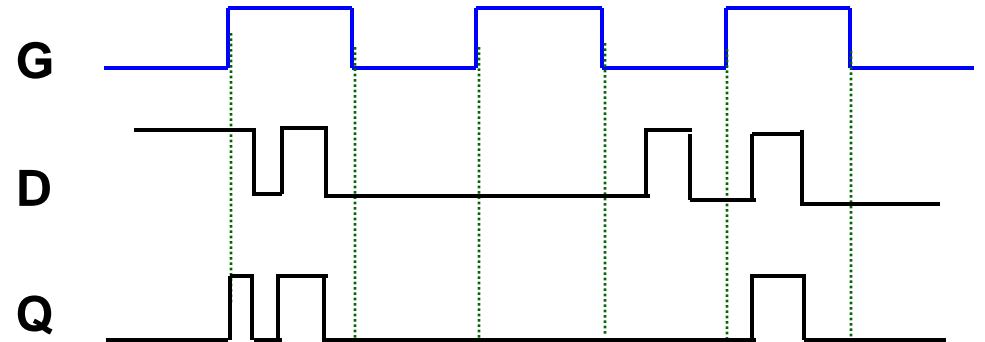
门控D锁存器的优缺点

- ◆ 特点：结构简单，仅一个输入端，不存在输入约束问题。
- ◆ 缺点：使能电位G作用期间，只要输入信号D改变（有时是干扰信号），Q也跟着改变，存在“空翻”现象

违背了构造时钟触发器的初衷：一个时钟内，最多允许触发器状态翻转一次

锁存器的使能端送时钟信号，电平触发方式的触发器

一个时钟内，触发器状态发生多次变化



“空翻”现象是锁存器（或电平方式触发器）共有的问题

“空翻”使以上器件不能正确实现计数功能！

- ☆ 关键问题：电平（电位）触发
- ☆ 解决方案：改电平触发为边沿触发

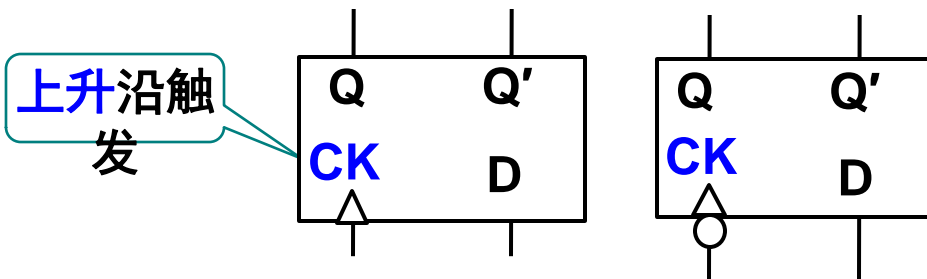
特征：时钟信号的上升沿或下降沿，触发器改变状态

边沿触发器

- D触发器
- RS触发器
- JK锁存器
- T触发器
- T'触发器
- 带附加输入端的触发器

边沿触发器——D触发器

(1) 逻辑符号



(2) 功能表（上升沿为例）

时钟端 CK	输入端 D	现态 Q_n	次态 Q_{n+1}
↑	0	0	0
↑	0	1	0
↑	1	0	1
↑	1	1	1

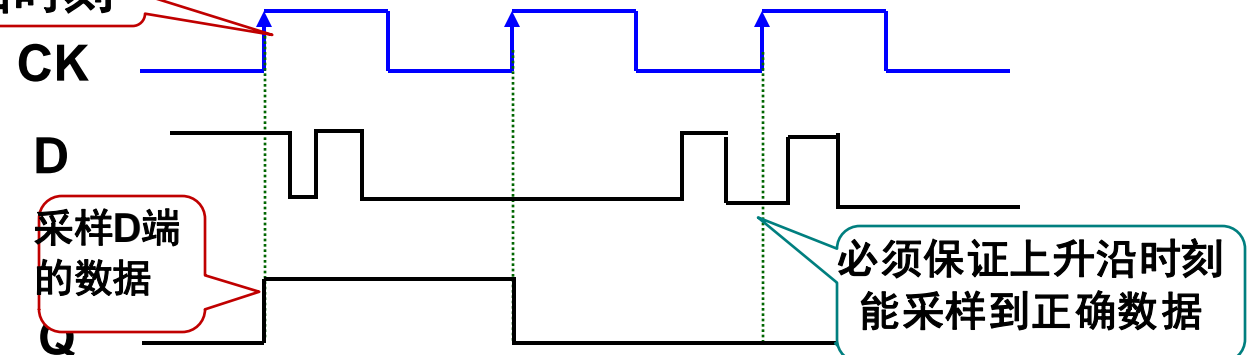
(3) 次态方程

$$Q^{n+1} = D$$

时钟触发器的特点

- ◆由时钟脉冲确定状态转换的时刻(即何时转换?)
- ◆由输入信号确定触发器状态转换的方向(即如何转换?)

上升沿时刻



必须保证上升沿时刻能采样到正确数据

避免了空翻

(4) 驱动表

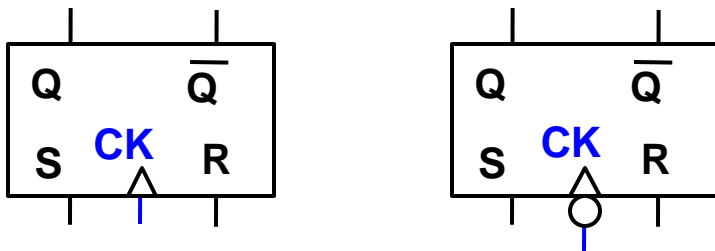
Q_n	→	Q_{n+1}	D
0	→	0	0
0	→	1	1
1	→	0	0
1	→	1	1

D触发器的特点:

最简单, 应用最广

边沿触发器——RS触发器

(1) 逻辑符号



(3) 次态方程

$$Q_{n+1} = S + \bar{R}Q_n$$

$$SR = 0 \quad (\text{约束条件})$$

(2) 功能表（上升沿）

时钟端 CK	输入端 R	输入端 S	现态 Q_n	次态 Q_{n+1}
↑	0	0	0	0
↑	0	0	1	1
↑	0	1	0	1
↑	0	1	1	1
↑	1	0	0	0
↑	1	0	1	0
↑	1	1	0	—
↑	1	1	1	—

(4) 驱动表

Q_n	→	Q_{n+1}	R	S
0	→	0	X	0
0	→	1	0	1
1	→	0	1	0
1	→	1	0	X

驱动表可以从触发器功能推导出来

输入存在约束

RS触发器：输入存在约束

D触发器：没有约束，但是只有一个输入端

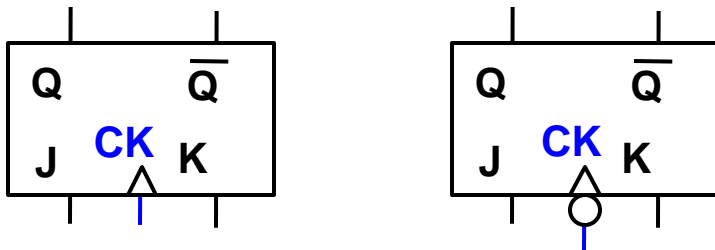
边沿触发器——JK触发器

(3) 次态方程

$$Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n$$

JK Q _n	JK			
	00	01	11	10
0	0	0	1	1
1	1	0	0	1

(1) 逻辑符号



(2) 功能表（下降沿）

时钟端 CK	输入端 J	输入端 K	现态 Q _n	次态 Q _{n+1}	
↓	0	0	0	0	保持
↓	0	0	1	1	置0
↓	0	1	0	0	
↓	0	1	1	0	置1
↓	1	0	0	1	
↓	1	0	1	1	翻转
↓	1	1	0	1	
↓	1	1	1	0	

功能最全，输入没有约束

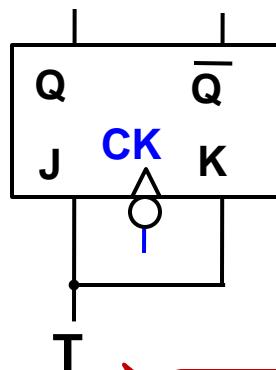
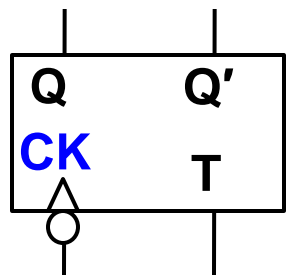
输入端		次态 Q _{n+1}
J	K	
0	0	Q _n
0	1	0
1	0	1
1	1	\bar{Q}_n

(4) 驱动表

Q _n → Q _{n+1}	J	K
0 → 0	0	X
0 → 1	1	X
1 → 0	X	1
1 → 1	X	0

边沿触发器——T触发器

(1) 逻辑符号



是JK触发器的特例

(2) 功能表（下降沿）

时钟端 CK	输入端 T	现态 Q_n	次态 Q_{n+1}
↓	0	0	0
↓	0	1	1
↓	1	0	1
↓	1	1	0

保持

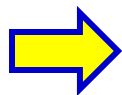
翻转

输入端 T	次态 Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

(3) 次态方程

$$Q_{n+1} = J \overline{Q_n} + \overline{K} Q_n$$

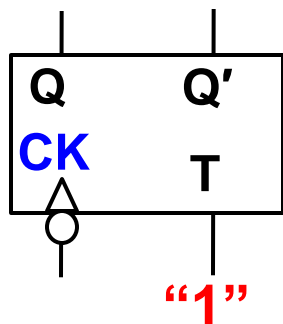
IF $J=K=T$



$$\begin{aligned} Q_{n+1} &= T \overline{Q_n} + T Q_n \\ &= T \oplus Q_n \end{aligned}$$

边沿触发器——T'触发器

(1) 逻辑符号



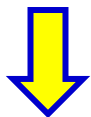
是T触发器的特例

(2) 功能表（下降沿）

时钟端 CK	输入端 T	现态 Q_n	次态 Q_{n+1}
↓	1	0	1
↓	1	1	0

(3) 次态方程

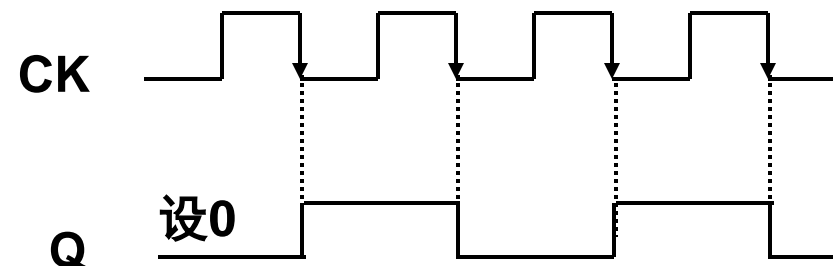
$$Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n$$



IF $J=K=T=1$

$$Q_{n+1} = \bar{Q}_n$$

(4) 波形分析



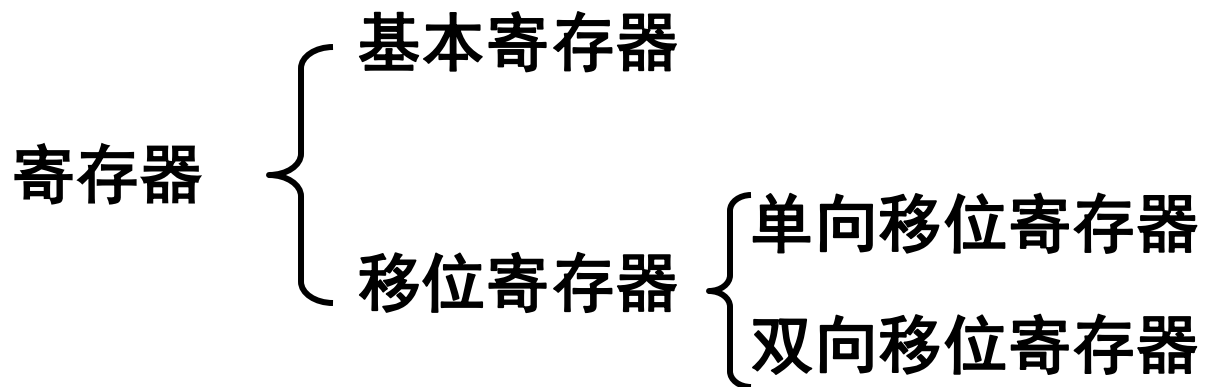
二分频

Unit 9 寄存器和计数器

- 寄存器 (Registers)
- 移位寄存器 (Shift Registers)
- 计数器 (Counters)
- 节拍发生器 (Beat Generator)

基本寄存器定义、分类等

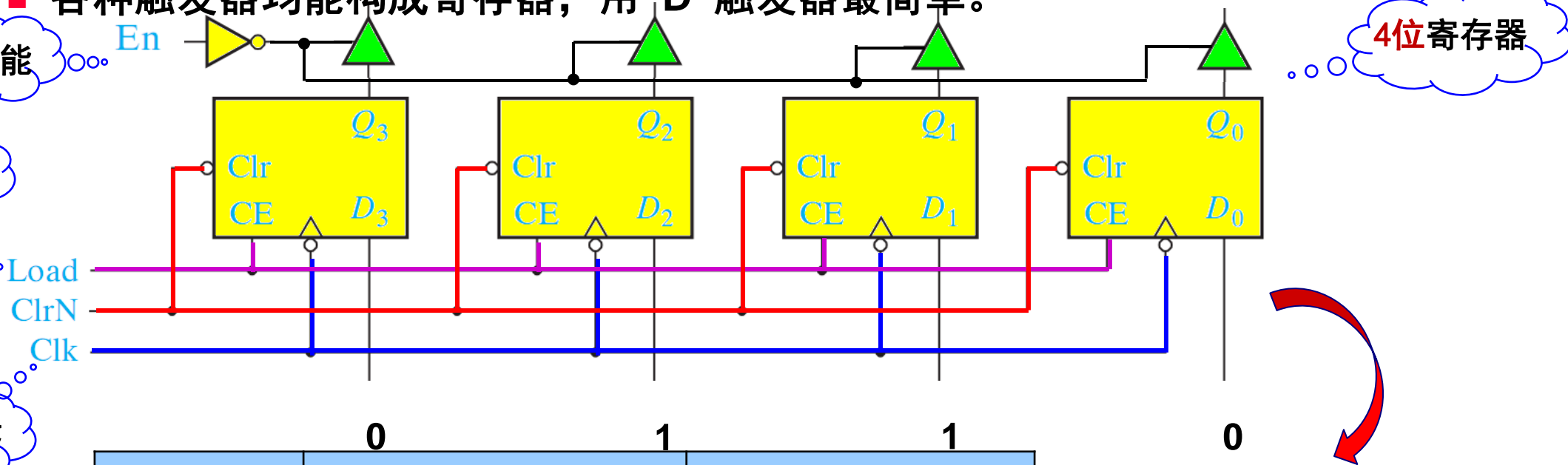
- 寄存器是计算机的一个重要部件，用于暂时存放一组二值代码（如参加运算的数据、运算结果、指令等）。
- 由触发器及控制门组成



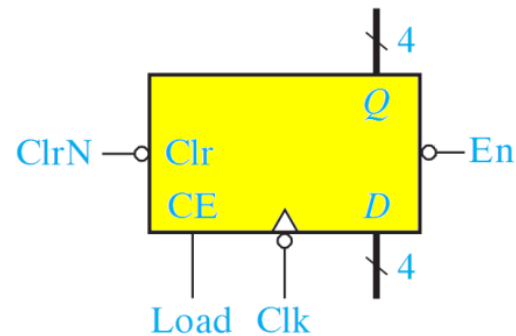
- 基本寄存器的操作：读出/写入/复位（清零）
- 移位寄存器的操作：读出/写入/复位（清零）/左移（右移）

基本寄存器

- 一个 n 位寄存器由 n 个触发器构成，能存放 n 位二进制数。
- 各种触发器均能构成寄存器，用 D 触发器最简单。



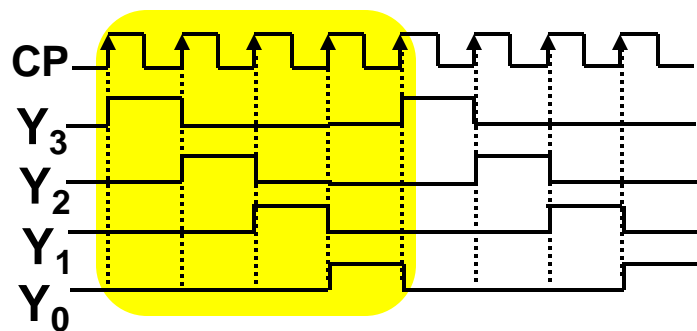
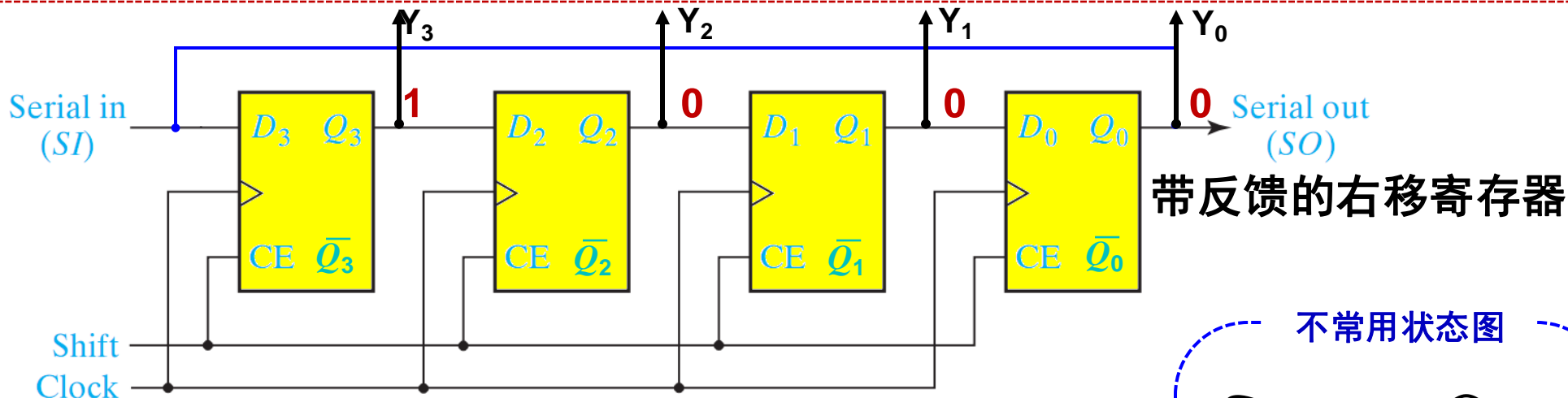
	0	1	1
功能	条件	寄存器输出	
异步清零	ClrN=0	$Q_3Q_2Q_1Q_0=0000$	
保持	ClrN=1, 且Load=0	$Q^{n+1}_3Q^{n+1}_2Q^{n+1}_1Q^{n+1}_0=$ $Q^n_3Q^n_2Q^n_1Q^n_0$	
写入	ClrN=1, Load=1, clk ↓	$Q_3Q_2Q_1Q_0=D_3D_2D_1D_0$	
读出	En=0	$Q_3Q_2Q_1Q_0=D_3D_2D_1D_0$	



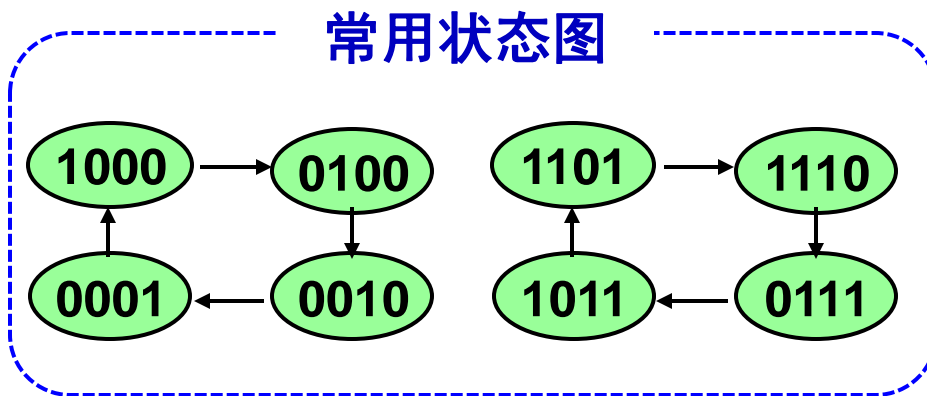
基本寄存器

右移寄存器的应用——环形计数器

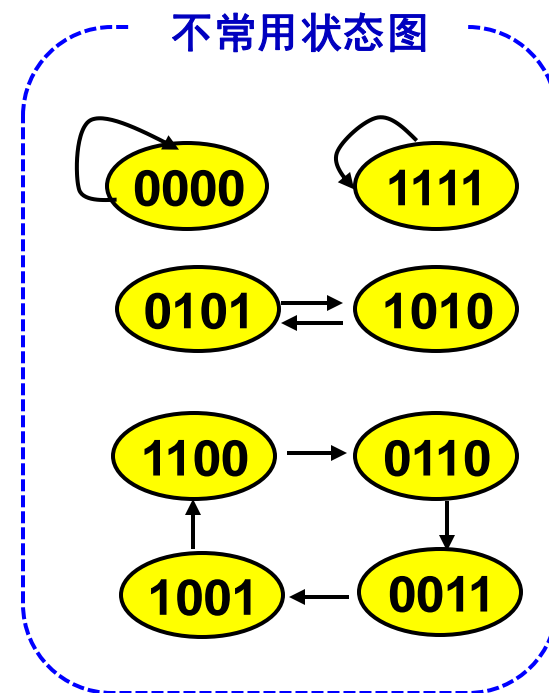
计数器：一种能在输入信号作用下依次循环通过预定状态的时序逻辑电路。



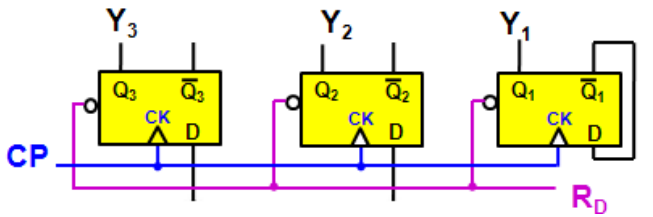
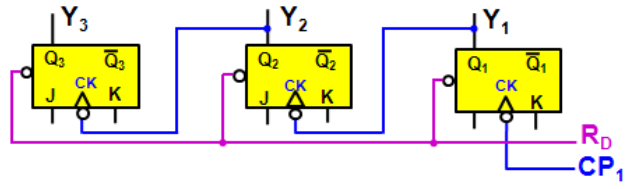
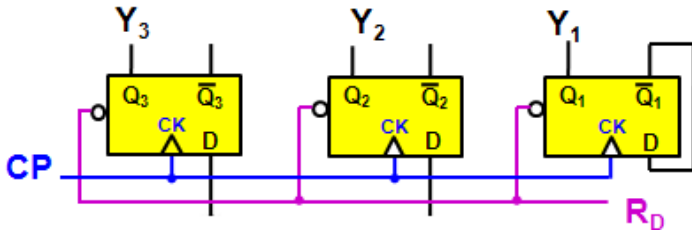
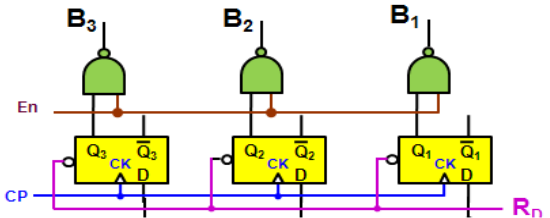
优点：电路简单
输出具有**二进制译码器**的特点



缺点： 2^n 个状态只用了 n 个；
不能自启动，需要**预置**

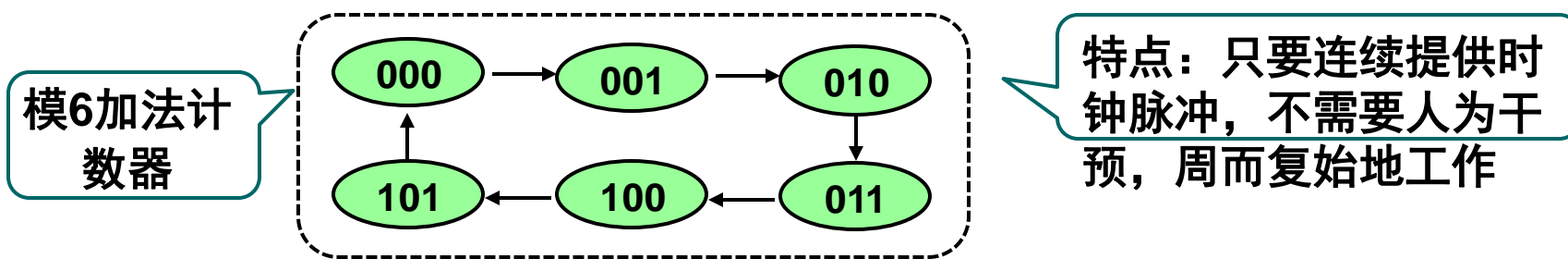


时序逻辑电路的分类

分类方式		种类	特点	电路框图示例
时序逻辑电路	按照 时钟 信号的连接方式	同步时序——	■ 所有的 时钟端 连接在一起，状态的改变 同时 发生（数字系统中用到的最多）	
		异步时序——	■ 没有 统一的时钟脉冲同步，状态的改变有先有后， 不同时 发生 ■ 容易产生毛刺（有不利影响）	
	按照电路 输出与输入及电路状态 的关系	摩尔型电路（ <i>Moore</i> ）	■ 电路的 输出仅与现态有关 ，与电路的 输入无关 ；或者直接以电路状态作为输出。	
		米里型电路（ <i>Mealy</i> ）	■ 电路 输出与电路的现态及电路的输入均有关 ；	

典型时序逻辑部件——计数器

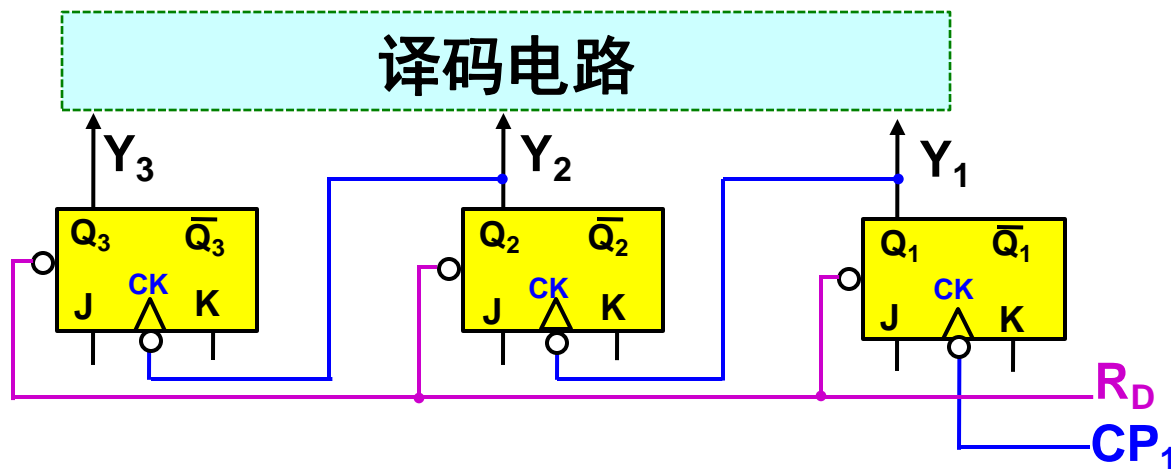
计数器是一种能在输入信号作用下依次通过预定状态的时序逻辑电路，是数字系统和计算机广泛使用的逻辑器件，可用于计数、分频、定时、控制、产生节拍脉冲（顺序脉冲）和序列脉冲等。



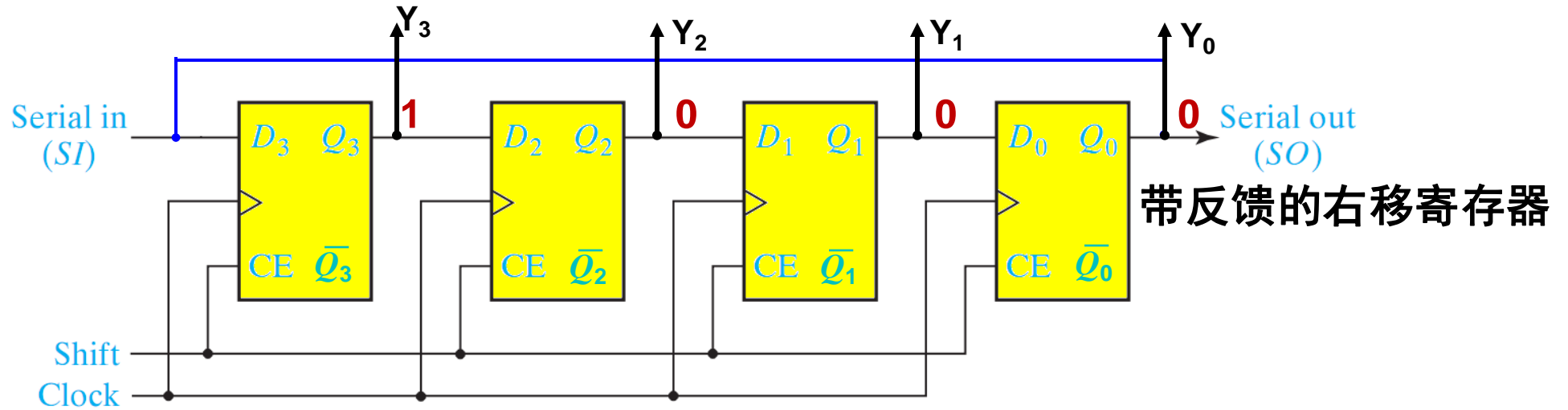
- 由一组触发器构成, 计数器中的“数”是用触发器的状态组合来表示的。
- 计数器在运行时，所经历的状态是周期性的，总是在有限个状态中循环。
- 将一次循环所包含的**状态总数**称为计数器的“**模**”，记为 N , 包含 n 个触发器的最大模值 $N = 2^n$ 。
- 把作用于计数器的时钟脉冲称为计数脉冲，用 CP (或 CLK) 表示。

异步计数器总结

- 外接时钟源只作用于**最低位触发器**，高位触发器的时钟信号通常由低位触发器的输出提供，高位触发器的翻转**有待**低位触发器翻转后才能进行。
- 每一级触发器都存在**传输延迟**，位数越多计数器工作速度越慢，在大型数字设备中较少采用。
- 对计数器状态进行译码时，由于触发器不同步，译码器输出会出现**尖峰脉冲**（位数越多，尖峰信号越宽），使仪器设备产生误动作。
- **优点**：结构比较简单，所用元件较少。



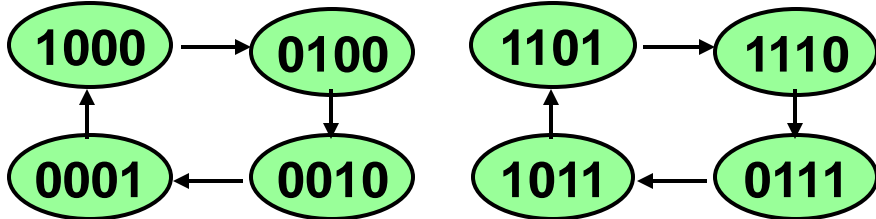
同步计数器——环形计数器



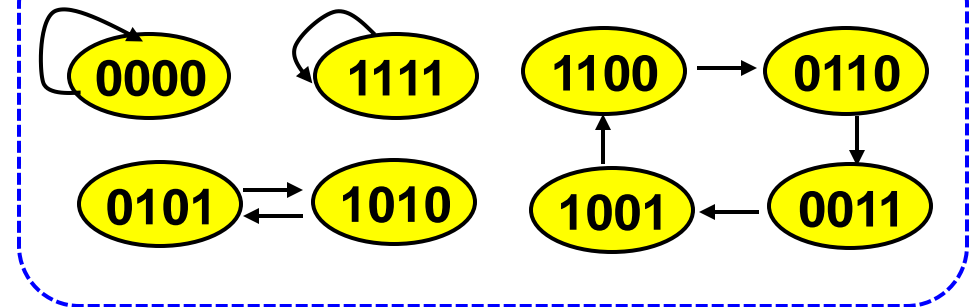
优点：电路简单，
输出具有二进制译
码器的特点

缺点： 2^n 个状态只使用了 n 个；
不能自启动，需要**预置**

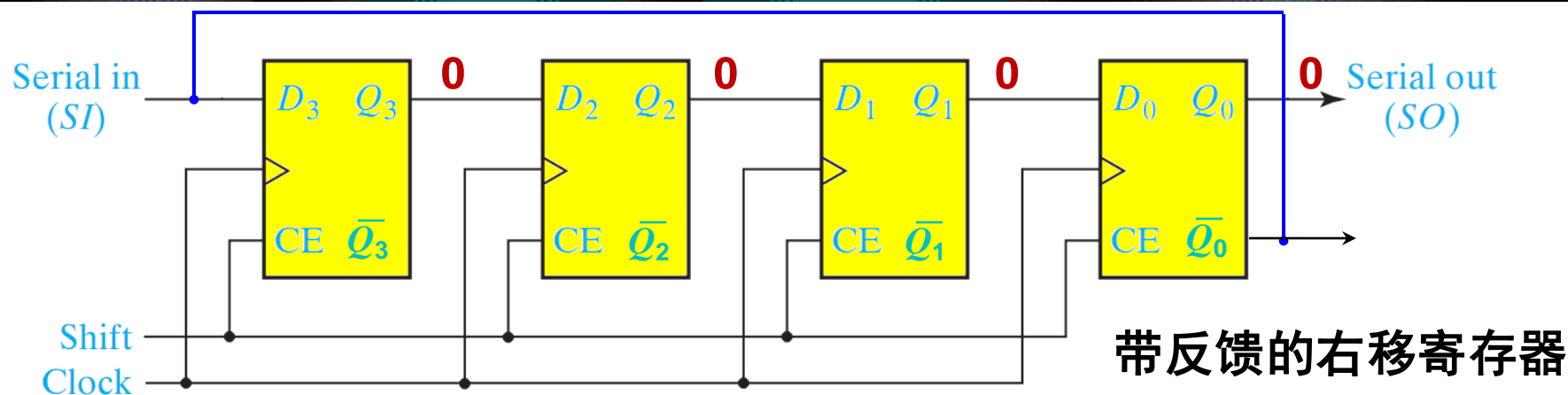
常用状态图



不常用状态图

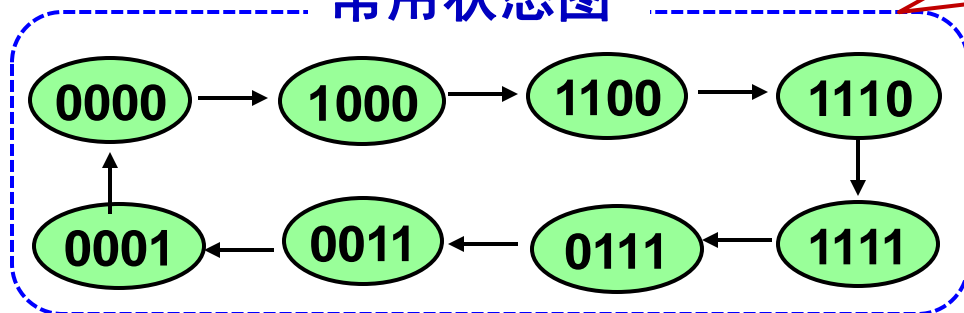


同步计数器——扭环形计数器Johnson Counter



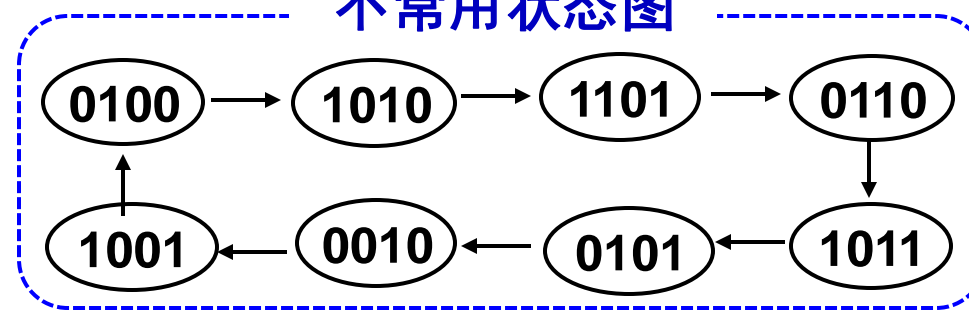
电路具有格雷码输出的特点

常用状态图



2^n 个状态使用了 $2n$ 个；
不能自启动，需要预置

不常用状态图



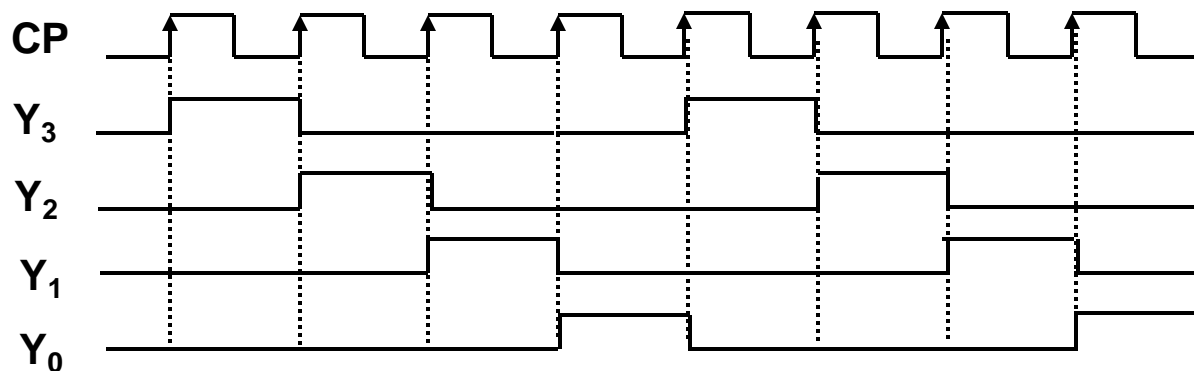
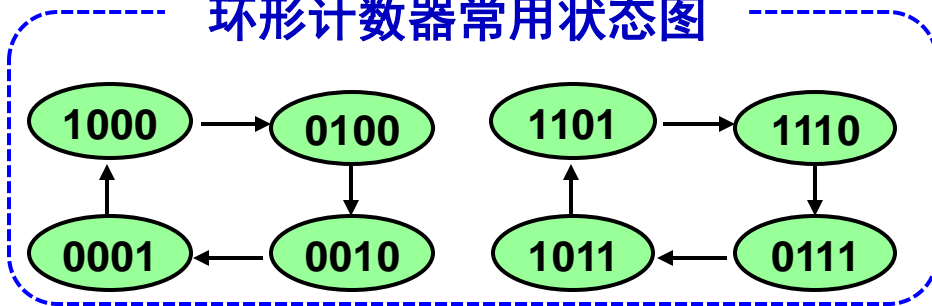
环形、扭环形计数器总结

特点：在移位寄存器的基础上，增加反馈逻辑电路组成。

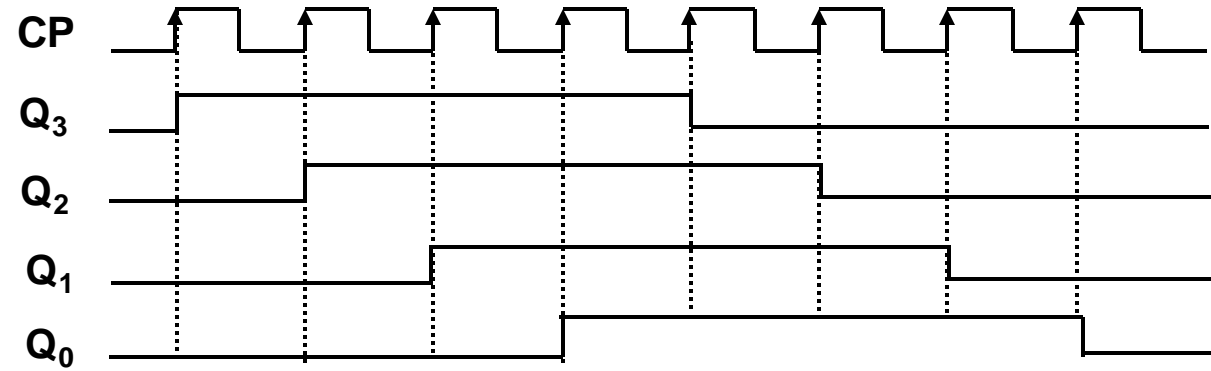
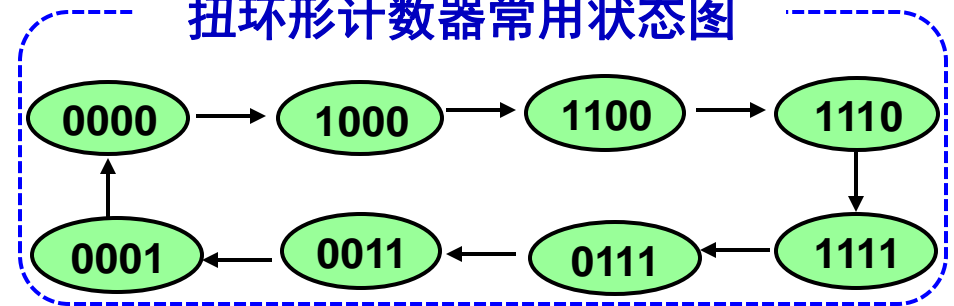
用途：

- 构成**特殊编码**的计数器（非二进制计数器）
- 环形计数器和扭环形计数器在计算机中可用于组成时序信号发生器（节拍发生器）

环形计数器常用状态图

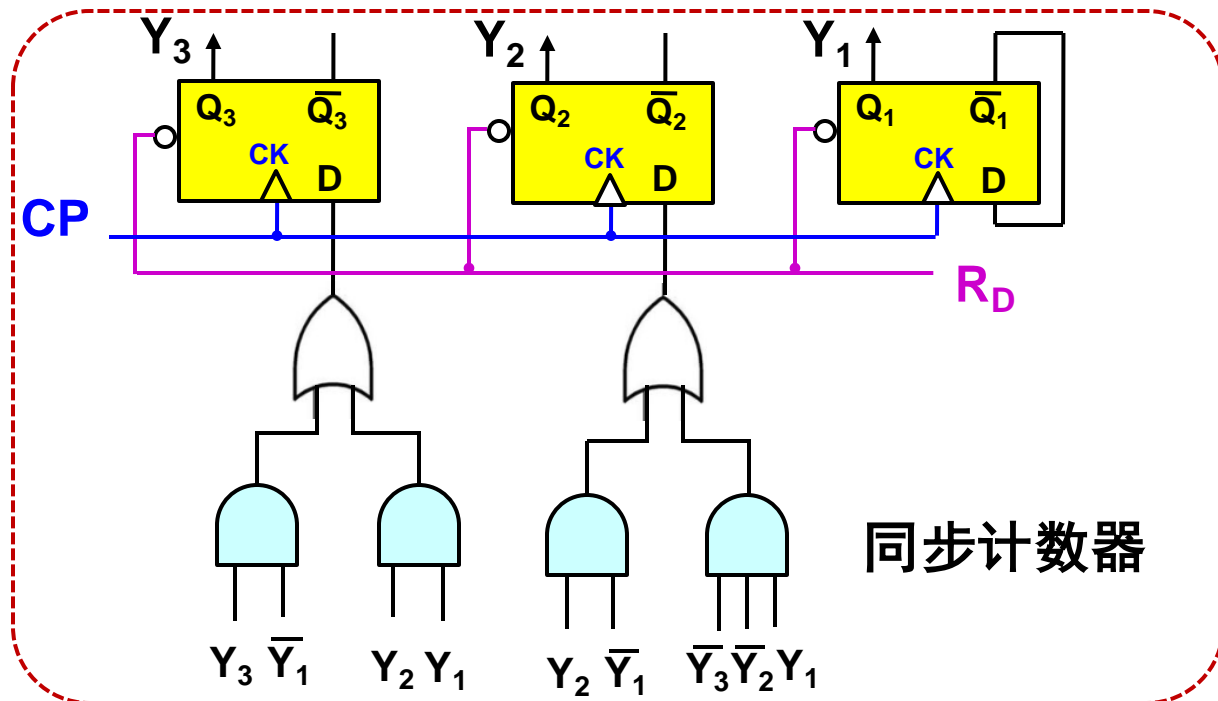
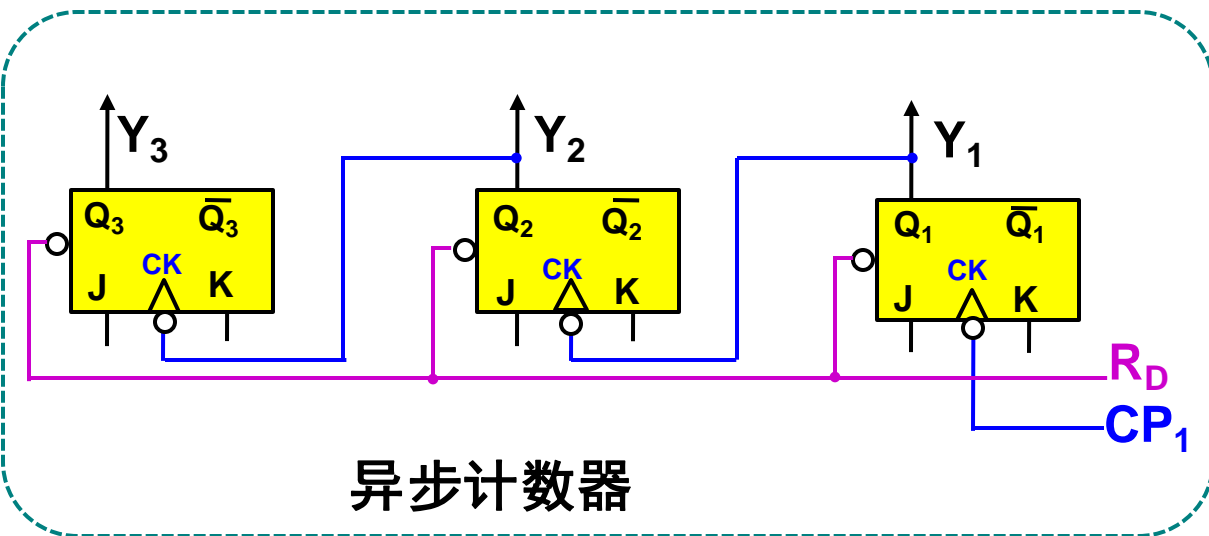


扭环形计数器常用状态图



同步计数器总结

- ❑ 所有触发器的时钟端并联在一起，受控于同一个外接时钟源；
- ❑ 所有触发器**同时翻转**，不存在时钟到各触发器输出的传输延迟的积累；
- ❑ 同步计数器的工作频率只与一个触发器的时钟到输出的传输延迟有关，所以它的工作频率比异步计数器高；
- ❑ 由于各触发器同时翻转，因此，同步计数器的输出**不会产生毛刺**；
- ❑ 缺点：结构比较复杂（各触发器的输入由多个Q输出的组合逻辑得到），所用元件较多。



Unit 10.1 时序逻辑电路分析

时序逻辑电路的分析方法

确定系统变量（输入变量、输出变量、状态变量）

- ① 列驱动方程（控制函数）
- ② 列输出方程（输出函数）
- ③ 列状态方程（次态方程）
- ④ 列写状态转换表
- ⑤ 画出状态图
- ⑥ 画出波形图（如必要）



- 同步时序电路
- 异步时序电路

时序逻辑电路分析——同步时序总结

同步时序逻辑电路分析方法总结

确定系统变量（输入变量、输出变量、状态变量）

① 列写三组方程：

- 驱动方程（控制函数）
- 状态方程（次态方程）
- 输出方程（输出函数）

② 列写状态转换表：

- 写出所有输入及现态的取值组合；
- 将每一种取值组合带入次态方程和输出方程，计算后的得出次态值和输出值；
- 从表中第一行开始，寻找状态转换规律；

③ 画出完整的状态图；

④ 得出电路功能，并说明能否自启动

时序逻辑电路分析——异步时序总结

异步时序逻辑电路分析方法总结

确定系统变量（输入变量、输出变量、状态变量）

① 确定每个触发器的时钟由谁供给？

② 列写三组方程：

- 驱动方程（控制函数）、状态方程（次态方程）、输出方程（输出函数）

③ 列写状态转换表：

- 首先，从假定（或给定）的某一个初始状态开始，每来一个外输入及外接时钟脉冲，确定与之对应的触发器次态及输出；
- 其次，确定该触发器的状态改变能否给其它触发器提供需要的时钟边沿。若能，则与之相应的其它触发器动作。否则，与之相应的其它触发器保持；重复该步骤，直到所有触发器的次态都确定为止。
- 接着，该次态成为新的现态，来一个外输入及外接时钟脉冲，重复上述操作，直到所有的 2^n 个现态到次态的转换都已计算完毕；从表中第一行开始，寻找状态转换规律；

③ 画出完整的状态图； ④ 得出电路功能，并说明能否自启动

Unit 10.2 同步时序逻辑电路设计方法

利用触发器设计同步时序逻辑的方法

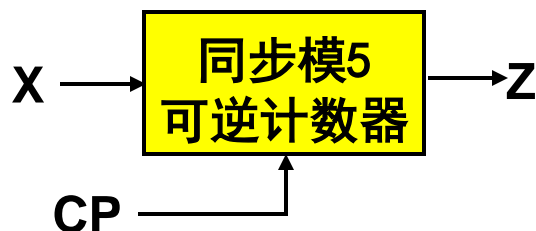
- (1) 根据需求 → 获得原始状态图、状态表
- (2) 最小化状态图、状态表
- (3) 状态编码（分配）→ 获得状态转移表
- (4) 状态转移表
触发器特征 } → 触发器激励（状态转移真值表）
- (5) 卡诺图化简 → { 激励（输入）函数表达式
输出函数表达式
- (6) 电路实现 (7) 检查无关项

直接构图法

直接构图法

- 1) 根据文字描述的设计要求，先**假定一个初态**；
- 2) 从这个初态开始，**每加入一个输入取值**，就可确定其次态和输出；
- 3) 该次态可能是现态本身，也可能是已有的另一个状态，或是新增加的一个状态。
- 4) 这个过程持续下去，直至**每一个现态向其次态的转换都被考虑**，并且不再构成新的状态。

例1：给出同步模5可逆计数器的**状态表**

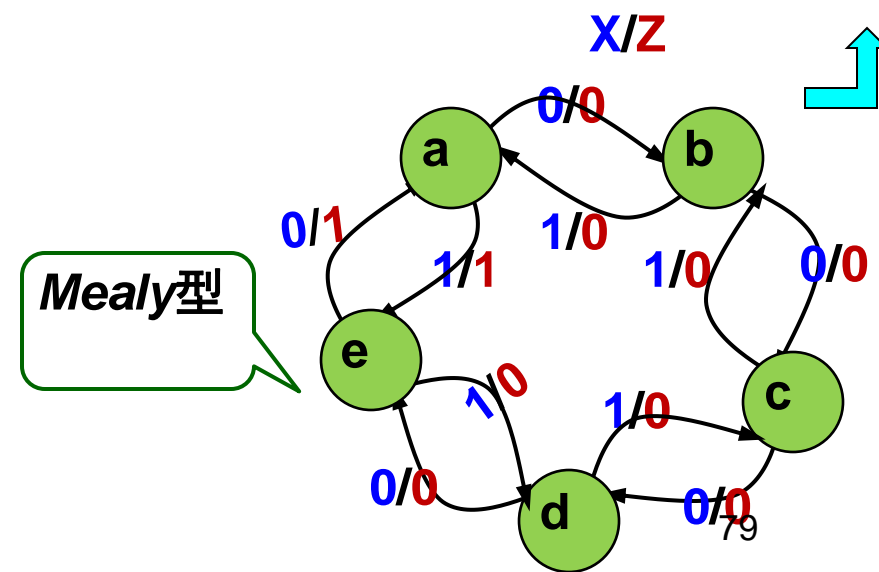


X=0：加计数

X=1：减计数

Z：进位、借位输出标志

现态 Q^n	Q^{n+1}/Z	
	X=0	X=1
a	b / 0	e / 1
b	c / 0	a / 0
c	d / 0	b / 0
d	e / 0	c / 0
e	a / 1	d / 0



序列检测电路设计

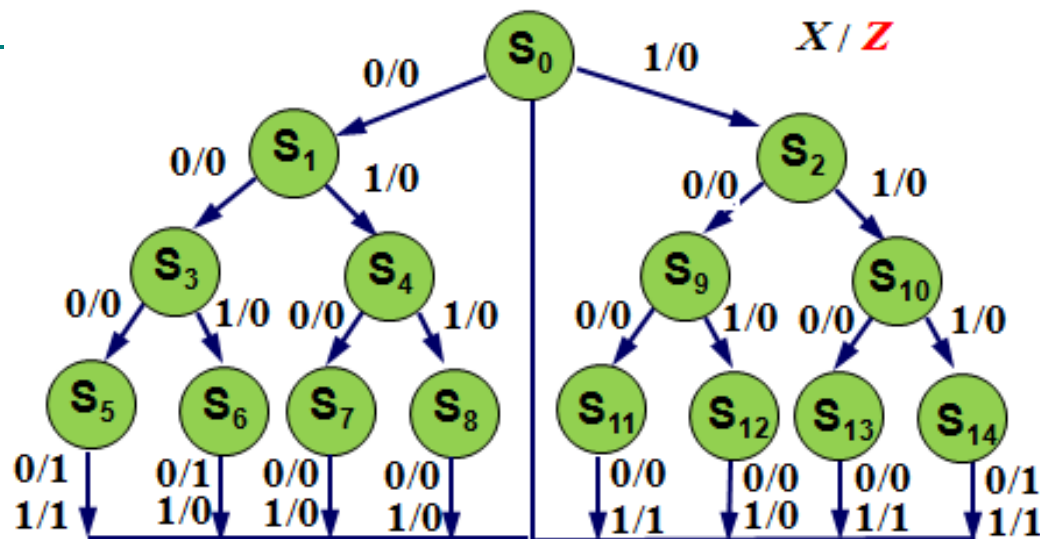
序列检测的原始状态图构造方法总结

- (1) 检测器输入端收到1位数据时，有两种可能：0或1，分别用 S_0 和 S_1 标记这两个状态，通常用 S_0 表示**初始状态**。
- (2) 收到2位数据时，只标记我们**感兴趣的子串**，用 S_2 表示（例如 10）
- (3) 同理，收到3位数据时，只标记我们感兴趣的子串，用 S_3 表示（例如 101）……，直到把我们感兴趣的**完整子串也已标记为止**。
- (4) 从初始状态开始，采用**直接构图法**，将每一个当前状态在所有取值下的次态转换及输出情况已都考虑到，并且**没有遗漏为止**。

码制检测电路设计

N位码制检测电路的原始状态图构造方法总结

- (1) 从初始状态 S_0 开始（这个初始状态**没有特殊含义**，仅代表一个起点），每来一个输入，次态总是分成左右两种情况。
- (2) 状态图由上至下分为**N**层：第一层代表起点；第二层代表检测器收到**1**位数据时，电路的状态情况；第三层代表检测器收到**2**位数据时，电路的状态情况……；直到第**N**层，代表检测器收到 **N-1**位数据时，电路的状态情况。再来一位输入数据，则构成了**N**位待检测码制。此时，检测器可以给出判读，该码制正确还是错误。
- (3) 一轮检测结束，回到初始状态，等待下一组输入。



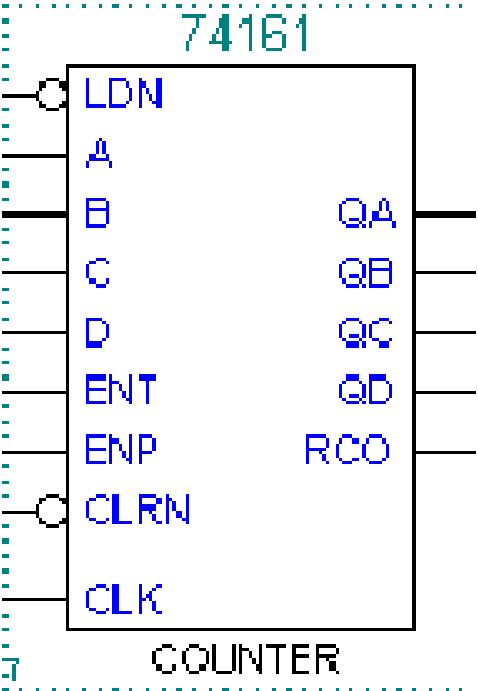
Unit 11 利用中规模芯片设计时序逻辑电路

- **计数器芯片**
 - 计数器芯片的级联
 - 计数器芯片的应用
- **寄存器芯片**
- **综合应用——序列信号发生器的设计**

用计数器芯片设计模10计数器

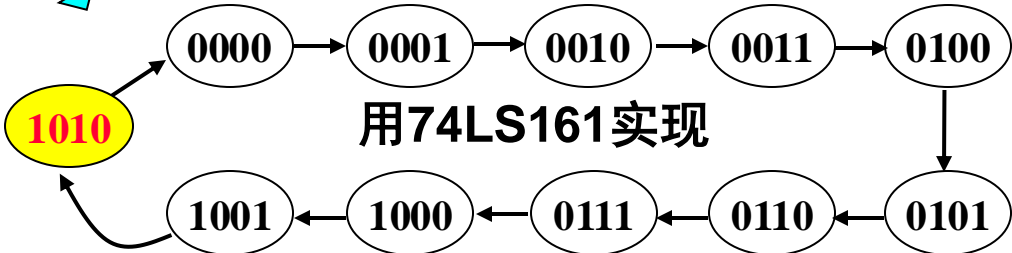
例1: 利用74LS161设计模10 计数器

① 清零法



异步清零只需要1个条件：清零端给有效信号立即回零

设计M进制计数器：需要M+1个状态



1010只在极短的瞬态出现，不包括在稳定的循环中

芯片型号	计数进制	输出特点	置数方式	清零方式
74LS160	十进制	8421BCD码	同步	异步
74LS161	十六进制	4位二进制码	同步	异步
74LS162	十进制	8421BCD码	同步	同步
74LS163	十六进制	4位二进制码	同步	同步

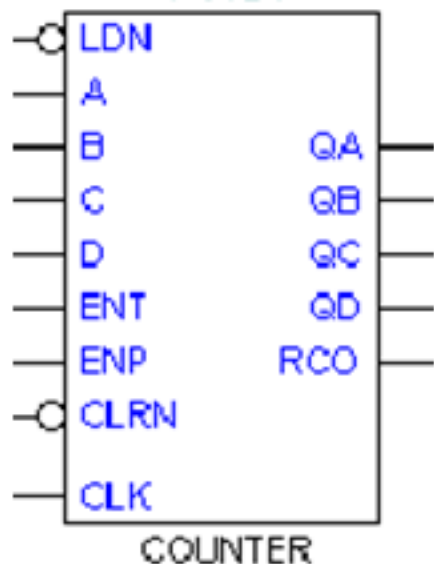
74LS161/160功能表

输入					输出			
CP	CLR _N	LD _N	ENT	ENP	Q _D	Q _C	Q _B	Q _A
X	0	X	X	X	0	0	0	0
↑	1	0	X	X	D	C	B	A
X	1	1	0	X	保持			
X	1	1	X	0	保持			
↑	1	1	1	1	计数,计满时RCO=1			

模10计数器

例2: 利用74LS163 设计模10 计数器

① 清零法



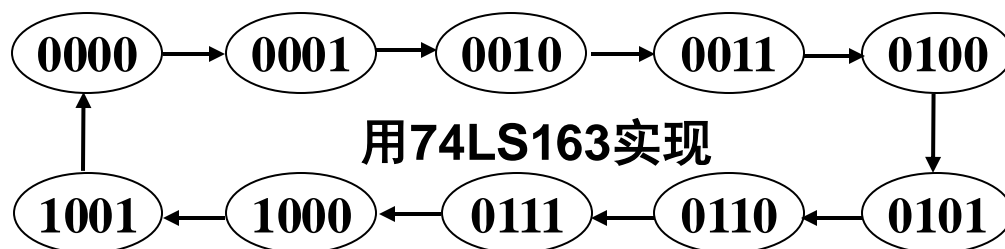
同步清零需要2个条件同时具备: 清零端给有效信号+时钟边沿到来

设计M进制计数器:
需要**M**个状态

芯片型号	计数进制	输出特点	置数方式	清零方式
74LS160	十进制	8421BCD码	同步	异步
74LS161	十六进制	4位二进制码	同步	异步
74LS162	十进制	8421BCD码	同步	同步
74LS163	十六进制	4位二进制码	同步	同步

74LS163/162功能表

输入					输出			
CP	CLR _N	LD _N	ENT	ENP	Q _D	Q _C	Q _B	Q _A
↑	0	X	X	X	0	0	0	0
↑	1	0	X	X	D	C	B	A
X	1	1	0	X	保持			
X	1	1	X	0	保持			
↑	1	1	1	1	计数, 计满时RCO=1			



序列信号发生器的设计-1

任意类型

- 使用**D触发器**设计
- 使用**计数器** + **数据选择器**设计；
- 用**移位寄存器** + **反馈电路**设计 (逻辑门 or 译码器 or 数据选择器)
- 用**计数器** + **PROM**设计

大体思路：

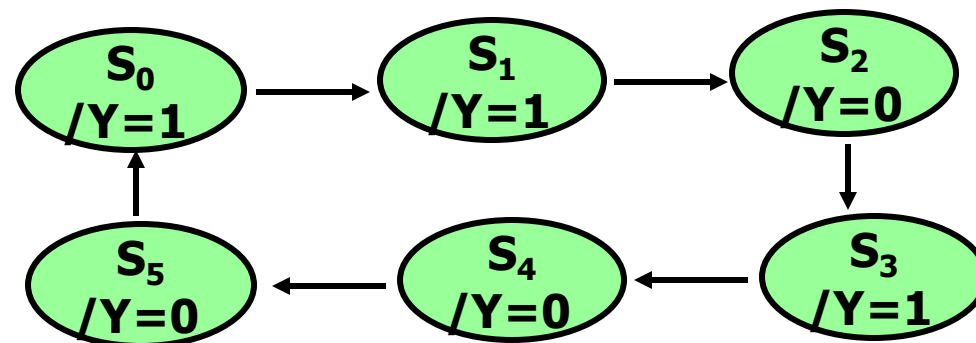
1. 实现序列信号一个周期之内的波形
2. 将此波形循环再现

例1：用D触发器设计一个 **110100** 序列信号发生器

方法1：利用D触发器

- 序列信号长度为 L ，则取 L 个不同的状态
- 每个状态下时序电路的输出就是序列信号中的一位。

1. 画状态转换图



时序电路的不同状态对应输出序列中的各位。

Verilog硬件描述语言

- 概述
- 结构化设计与Verilog模块
- 标识符和数据类型
- 运算符及表达式
- 赋值语句和过程块（组合逻辑电路描述）
- 条件语句和条件表达式
- 模块的测试

Verilog语言的层次

• 行为描述语言&结构描述语言

系统级：

用高级语言结构实现设计模块的外部性能模型；

算法级：

用高级语言结构实现设计算法模型；

RTL级（寄存器传输级）：

描述数据在寄存器之间流动和如何处理这些数据的模型；

门级：

描述逻辑门以及逻辑门之间的连接模型；

开关级：

描述器件中三极管和存储节点以及它们之间连接的模型。

行为级描述

侧重对模块**行为功能**
的抽象描述

结构级描述

侧重对模块**内部结构实现**
的具体描述

描述方式

```
module decoder_38_struct(  
    input [2:0] data_in,  
    input [2:0] en,  
    output reg[7:0] data_out  
);  
  
    // 内部连线定义  
    wire en_out, en0_out, en1_not;  
  
    // 调用基本库元件  
    not not0(en0_out, en[0]);  
    not not1(en1_out, en[1]);  
    and and0(en_out, en0_out, en1_not, en[2]);  
endmodule
```

结构化描述

```
module decoder_38_dataflow(  
    input [2:0] data_in,  
    input [2:0] en,  
    output [7:0] data_out  
);  
  
    assign data_out[5] = data_in[2] && ~data_in[1]  
        && data_in[0];  
    // 或 assign data_out[5] = (data_in == 3'd5);  
  
    // 其他赋值语句  
endmodule
```

数据流描述

```
module decoder_38(  
    input [2:0] data_in,  
    input [2:0] en,  
    output reg[7:0] data_out  
);  
  
    always @( * ) begin //非完整实现，需要自行补全其他信号处理  
        case( data_in )  
            3'b000: data_out = 8'b0000_0001;  
            3'b001: data_out = 8'b0000_0010;  
            3'b010: data_out = 8'b0000_0100;  
            3'b011: data_out = 8'b0000_1000;  
            3'b100: data_out = 8'b0001_0000;  
            3'b101: data_out = 8'b0010_0000;  
            3'b110: data_out = 8'b0100_0000;  
            3'b111: data_out = 8'b1000_0000;  
        endcase  
    end  
endmodule
```

行为描述

结构化描述：通过调用库中的元件或已设计好的模块来完成设计。

数据流描述：主要使用assign连续赋值语句，多用于组合逻辑电路。

行为描述：从电路的功能出发，关注逻辑电路输入、输出的因果关系。

即在何种输入条件下产生何种输出，描述的是一种行为特性。

在较复杂的电路设计中，三种描述方法往往混合使用。

如何选择正确的数据类型？

模块U:

```
module U(  
  output wire Y,  
  input wire A, B);
```

//功能描述

```
endmodule
```

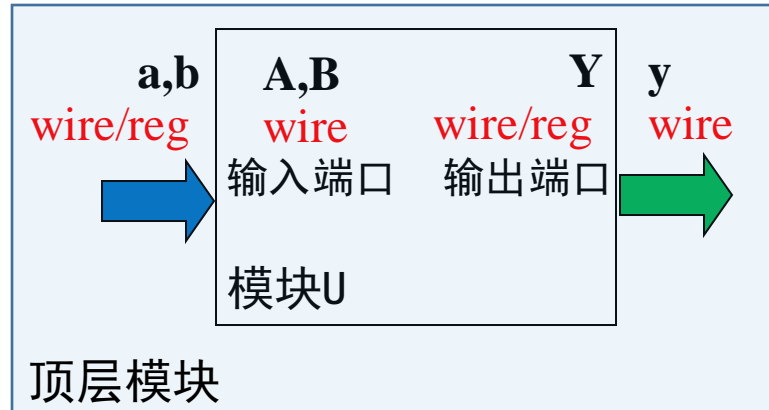
顶层模块:

```
module top(  
  端口定义  
);
```

```
  wire y;  
  reg a, b;
```

```
  U u1(  
    .Y(y),  
    .A(a),  
    .B(b)  
  );
```

```
  //其他逻辑  
endmodule
```



模块实例化不是参数传递

模块端口与外部信号按照其名字进行连接

赋值语句

- 在Verilog中，变量**不能随意赋值**，需要使用**连续赋值语句**或**过程赋值语句**。
- assign称为**连续赋值**；
- 过程块initial或always中的赋值称为**过程赋值**。

阻塞赋值与非阻塞赋值

- 阻塞 (Blocking) 赋值方式 (符号 “=”, 如 `b=a;`)
 - 在同一个 `always` 块中, 一条阻塞赋值语句如果没有执行结束, 那么该语句后面的语句就不能被执行, 即被 “阻塞”
 - `b` 的值在赋值语句执行完后立刻就改变;
 - 用在触发事件为电平敏感信号的 `always` 块中, 描述组合逻辑电路
- 非阻塞赋值 (Non-Blocking) 赋值方式 (符号 “`<=`”, 如 `b<=a;`)
 - 用在触发事件为时钟边沿的 `always` 块中, 描述时序逻辑电路
 - `b` 的值并不是立刻就改变;

时序逻辑的Verilog描述

- 时序逻辑电路在逻辑功能上的特点是任意时刻的输出不仅取决于当时的输入信号，而且还取决于电路原来的状态。
- 时序逻辑电路的变化通过时钟沿触发，需要使用时钟沿触发的always块描述。
- 用always块描述时序逻辑电路时，用非阻塞赋值。

隨堂小測



如果運氣不行，
那就試試勇氣

回顾

- ★ • 进制和编码
 - 8421BCD码
 - 余3码
 - 格雷码

- ★ 非常重要
- ★ 重点理解
- ★ 理解

电路最简

- ★ • 布尔代数
 - 逻辑代数定理和规则
 - 代数化简法 → 本质：提取公因子，消除冗余项
 - 最小项、最大项
- ★ • 卡诺图
 - 卡诺图化简法

提升效率

- 多级门电路
 - 两级门电路的设计
 - ~~多输出电路的设计~~

回顾

实际出错



• 险象以及消除

- 静态冒险的判断
- 静态冒险的消除

• 加减法器



- 全加器的设计
- 全减器的设计
- ~~0G门~~

提升器件复用率



• 组合逻辑元件

- 数据选择器（多路复用器）
- 三态门
- 译码器
- 编码器
- 奇偶校验器、比较器
- ~~只读存储器~~



非常重要



重点理解



理解

回顾

时序电路
基础元件
(次态方程)



- 锁存器和触发器
 - RS锁存器和D锁存器
 - RS触发器
 - JK触发器
 - D触发器
 - T (T') 触发器

时序电路典
型组合元件



- 寄存器与计数器
 - 基本寄存器
 - 移位寄存器
 - 计数器
 - ✓ 环形计数器
 - ✓ 扭环形计数器
 - 节拍发生器



非常重要



重点理解



理解

回顾

- ★ • **同步/异步时序电路分析方法**
 - ① 分析三组方程
 - ② 列出状态转换表
 - ③ 画出状态图
 - ④ 分析是否能够自启动

- ★ • **同步时序电路设计方法**
 - ① 原始状态图、状态表
 - ② 状态化简
 - ③ 状态分配
 - ④ 状态转换真值表
 - ⑤ 卡诺图化简得到输入、输出方程
 - ⑥ 画出电路图
 - ⑦ 检查无关项



非常重要



重点理解



理解

回顾

提升器件复用率 \Rightarrow

- 中规模芯片设计时序逻辑电路

- ★
 - 不同模值计数器
 - 节拍发生器
 - 序列信号发生器



非常重要



重点理解

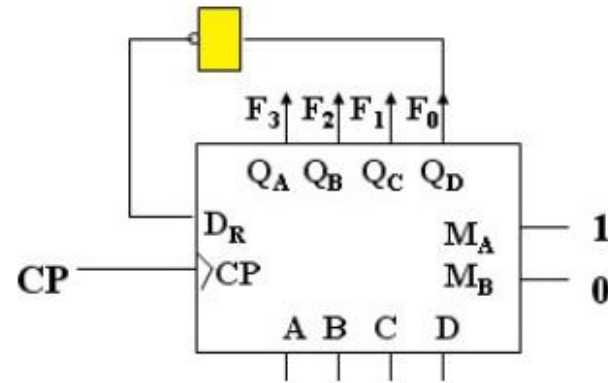


理解

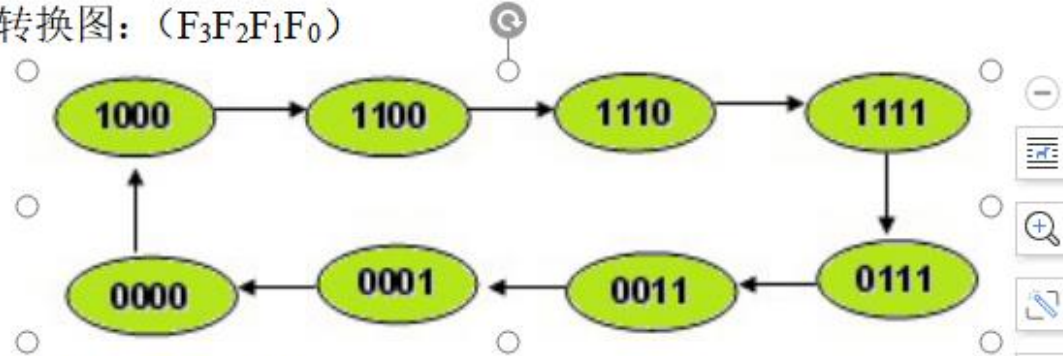
- Verilog

- 理解代码功能

5. 由寄存器芯片 74LS194 构成的电路如下图所示, $Q_DQ_CQ_BQ_A$ 是数据并行输出端, 初始值为 0000。ABCD 是数据并行输入端, D_R 是右移串行输入端, M_B 和 M_A 是方式控制端($M_BM_A=01$ 表示右移工作方式)。下面给出的对该时序电路的分析存在错误的是 ()



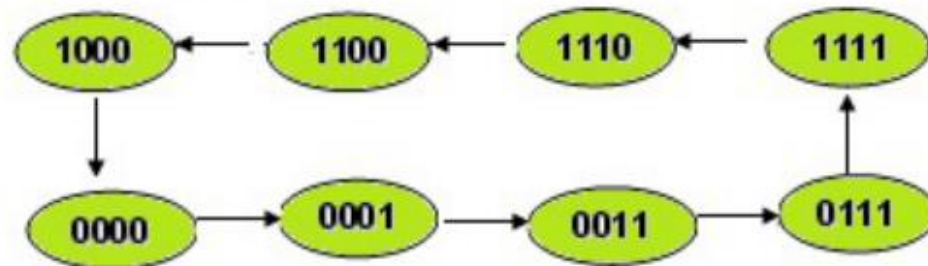
A. 电路的状态转换图: ($F_3F_2F_1F_0$)



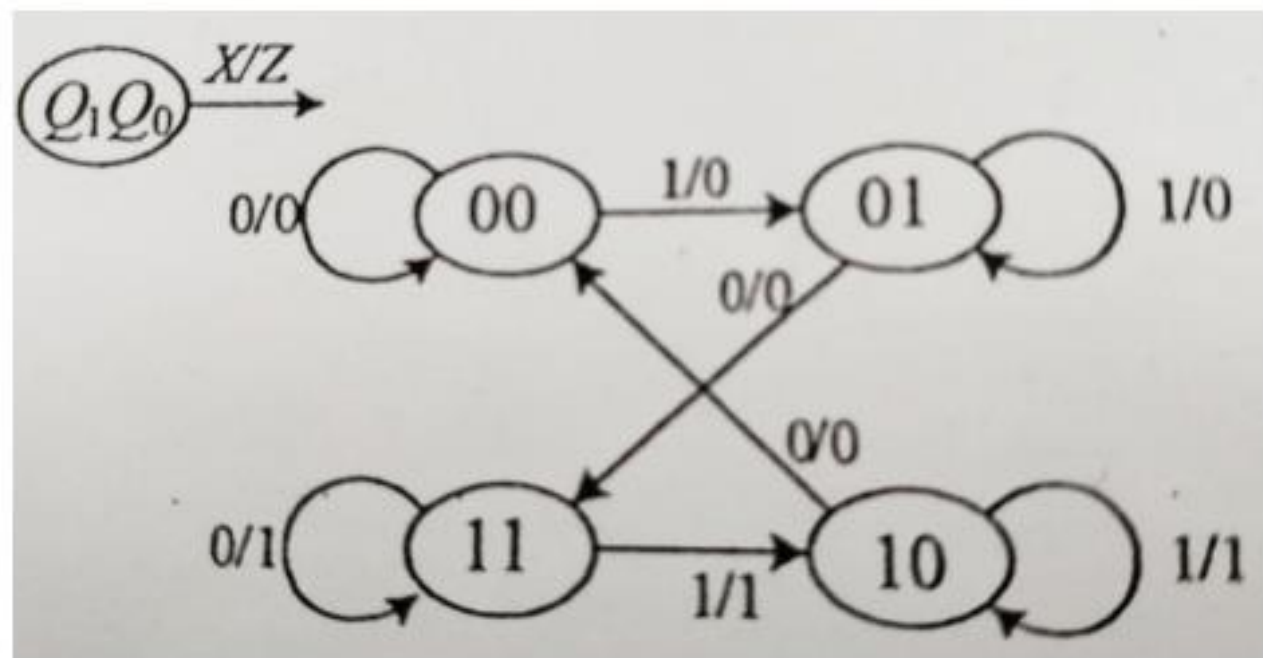
B. 电路功能为 4 位扭环形计数器

C. 电路功能为模 8 计数器

D. 电路的状态转换图: ($F_3F_2F_1F_0$)



七、（15 分）试用上升沿触发的 JK 触发器设计一同步时序电路，其状态转换图如下图所示， X 为电路的输入信号， Z 为电路的输出信号，请列出状态方程、驱动方程和输出方程，画出逻辑电路图。



祝大家考试顺利！

