

# 数字逻辑设计

Digital Logic Design

秦阳

School of Computer Science

csyqin@hit.edu.cn


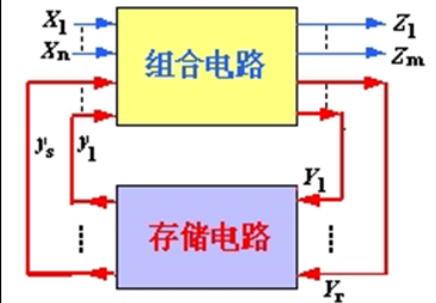
# 第8章 时序逻辑元件

---

- 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
- **现态（原态）和 次态（新态）**

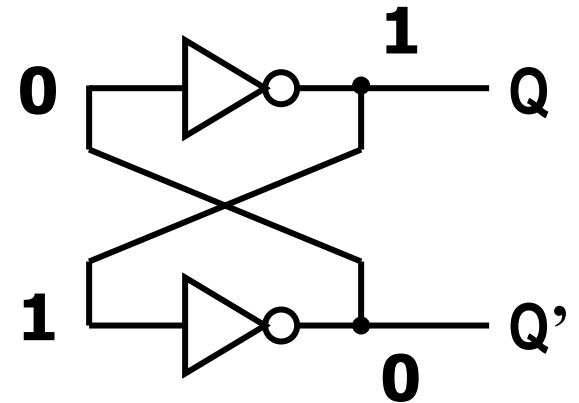
构成		定义	结构	电路框图	逻辑函数表达式
数字逻辑电路	组合逻辑电路	<p>任意时刻的输出——</p> <ul style="list-style-type: none"> <li>■ 仅与当前时刻的输入有关</li> </ul> $Z_m = f_m(x_1, \dots, x_n)$	不包含存储元件		<p>只有一组：</p> $Z_m = f_m(x_1, \dots, x_n)$
	时序逻辑电路	<p>任意时刻的输出与以下均有关：</p> <ul style="list-style-type: none"> <li>■ 当前时刻的输入</li> <li>■ 电路过去（上一个时刻）的工作状态</li> </ul> $Z_m = f_m(x_1, \dots, x_n, y_1, \dots, y_s)$	包含存储元件		<p>有三组：</p> <p><b>输出方程,驱动方程,状态方程：</b></p> $Z_m = f_m(x_1, \dots, x_n, y_1, \dots, y_r)$ $Y_r = g_r(x_1, \dots, x_n, y_1, \dots, y_s)$ $Y_s^{n+1} = q_s(x_1, \dots, x_n, Y_1^n, \dots, Y_s^n)$

# 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

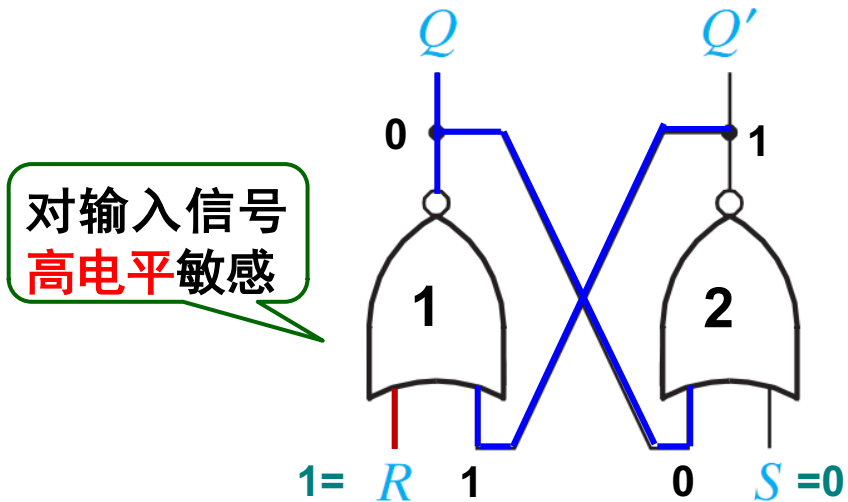
## 锁存器与触发器的特性（双稳态）

1. 有两个互补的输出端  $Q$  和  $Q'$
2. 有两个稳定的状态: state 0, state 1
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. 没有外界信号刺激，维持当前状态不变。



# 基本SR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

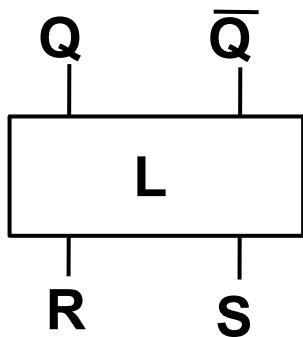
# 基本SR锁存器次态方程、逻辑符号等

## (3) 次态方程

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

(SR = 0) 约束条件

## (4) 逻辑符号



		S	
		0	1
RQ	00	0	1
	01	1	1
	11	0	X
	10	0	X

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	—

# 基本SR锁存器驱动表

(5) 驱动表：完成状态转换需要满足的输入条件

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

用于时序  
电路设计

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	—

保持

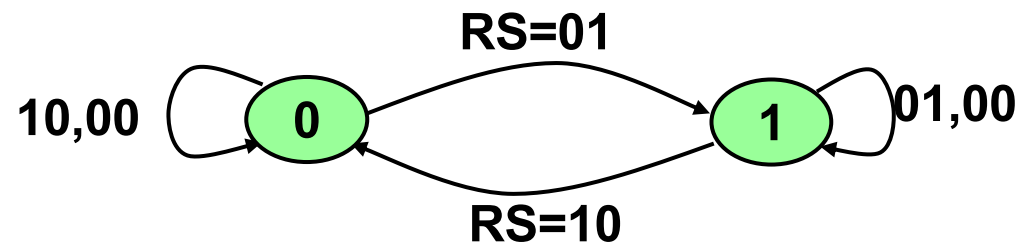
置 0

(6) 状态图

反映时序电路状态转移规律及相应输入、输出取值关系的有向图

图中元素的含义

- 圆圈：表示电路的状态
- 有向线段：表示状态的转换关系
- 有向线段旁的文字：表示转换条件，即输入信号取值



# SR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

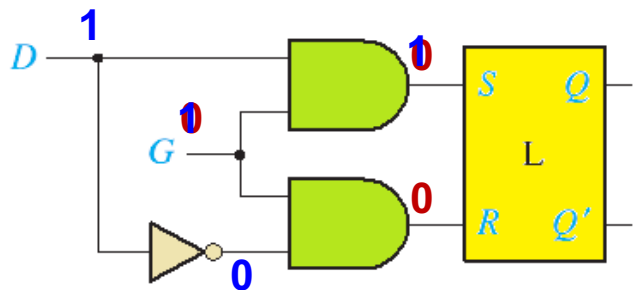
- **基本SR锁存器**是众多触发器的鼻祖

  - 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

## (1) 电路构成



## (2) 功能表

使能端 G	输入端 D	现态 $Q_n$	次态 $Q_{n+1}$
0	X	0	0
0	X	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

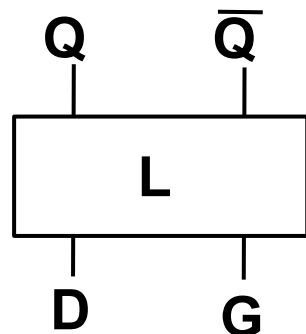
## (3) 次态方程

$Q \backslash GD$	00	01	11	10
0	0	0	1	0
1	1	1	1	0

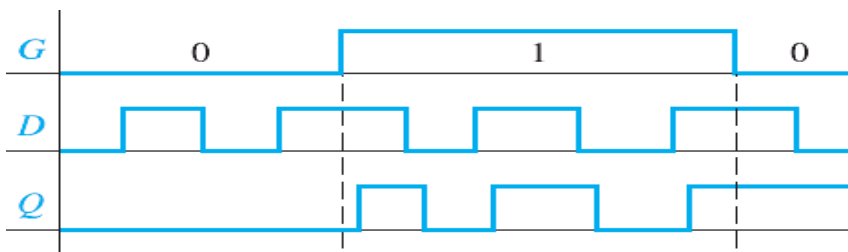
$$Q_{n+1} = GD + \bar{G}Q_n$$

在G为高电平期间, Q端的输出直接拷贝D端波形

## (4) 逻辑符号



## (5) 时序分析



## (6) 典型芯片

74LS373: 8D锁存器

# 门控D锁存器的优缺点

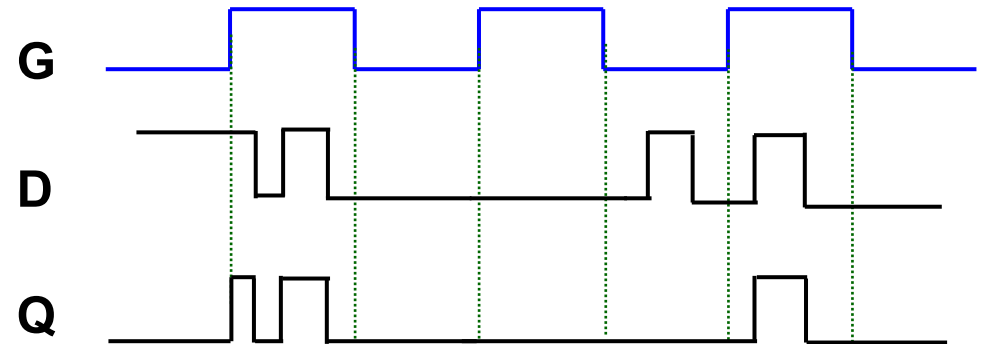
特点：结构简单，仅一个输入端，不存在输入约束问题。

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

时钟信号的上升沿或下降沿，触发器改变状态

# 目 录

---

- 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

# 时钟触发器

- 受时钟脉冲控制的触发器称作时钟触发器。
- 时钟也称同步信号。将多个触发器的时钟端相连，可以控制它们同一时刻动作。

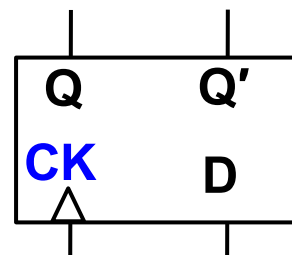
## 时钟触发器分类

### 按逻辑功能

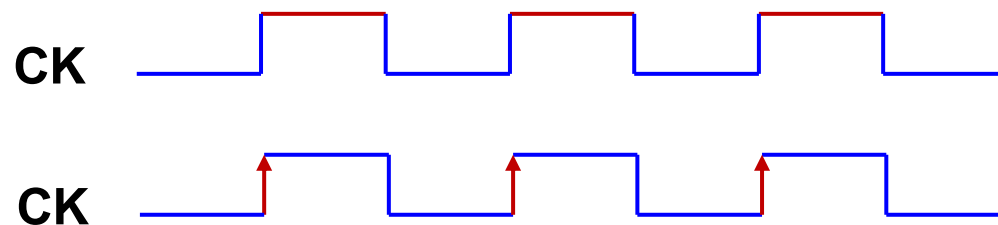
SR触发器  
D触发器  
JK触发器  
T触发器  
T'触发器

### 按触发方式

电平触发  
边沿触发



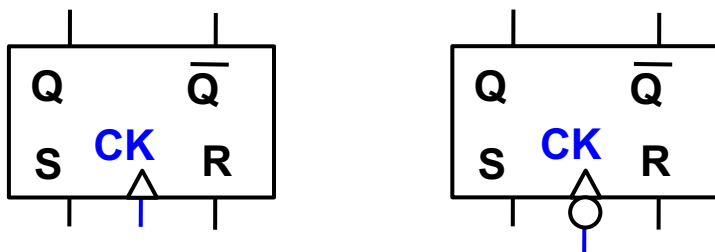
电平触发方式：时钟信号高电平期间，触发器可以做状态翻转



边沿触发方式：时钟上升沿到来时刻，触发器可以做状态翻转

# 边沿触发器——SR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

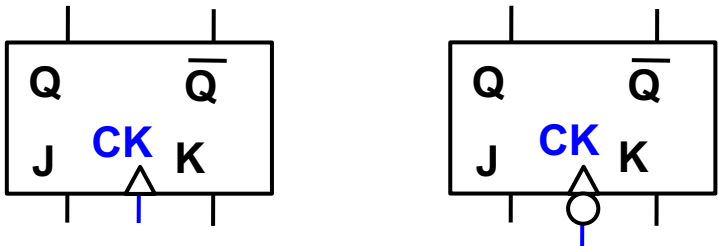
SR触发器：输入存在约束

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

### (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
↓	0	0	1	1
↓	0	1	0	0
↓	0	1	1	0
↓	1	0	0	1
↓	1	0	1	1
↓	1	1	0	1
↓	1	1	1	0

功能最全，输入没有约束

输入端 J	输入端 K	次态 $Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	$\bar{Q}_n$

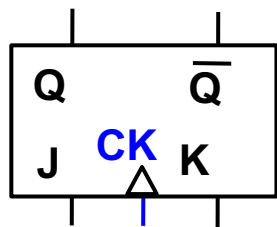
$$Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n$$

JK	00	01	11	10
0	0	0	1	1
1	1	0	0	1

### (4) 驱动表

$Q_n \rightarrow Q_{n+1}$	J	K
0 → 0	0	X
0 → 1	1	X
1 → 0	X	1
1 → 1	X	0

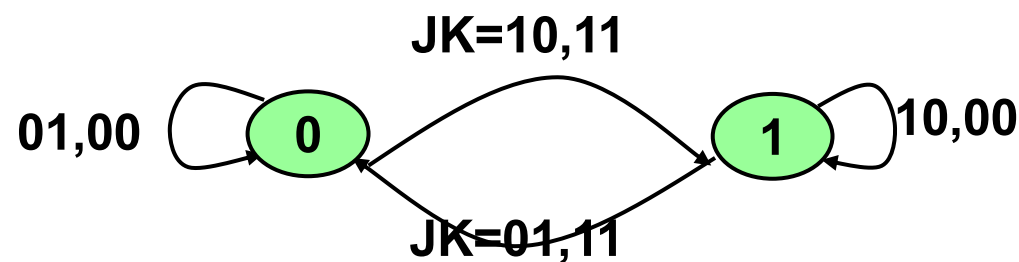
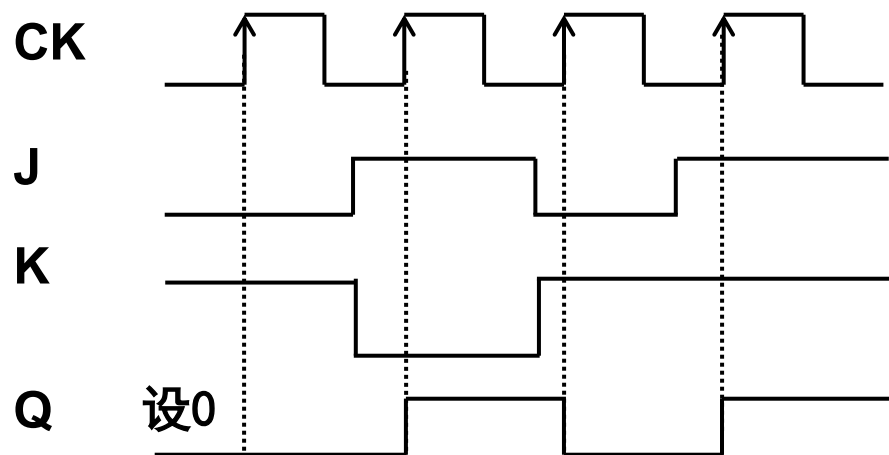
# 边沿触发器—— JK触发器



## 时钟边沿触发器

何时转换? ——时钟脉冲有效边沿到来时刻  
如何转换? ——输入信号取值确定

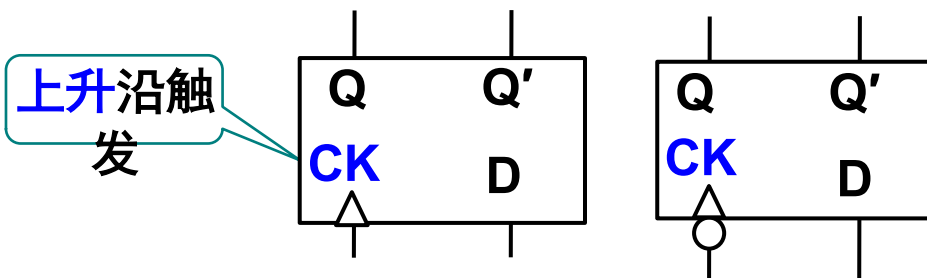
输入端		次态
J	K	$Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	$\overline{Q_n}$



JK触发器状态图

# 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

CK

D

采样D端的数据

Q

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

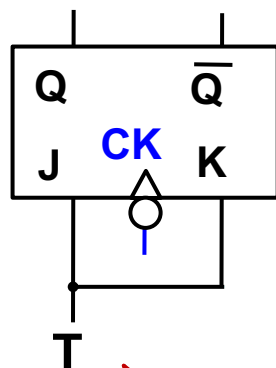
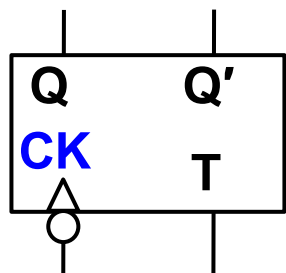
D触发器的特点：

最简单，应用最广



# 边沿触发器——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

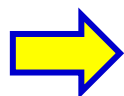
翻转

输入端 T	次态 $Q_{n+1}$
0	$Q_n$
1	$\bar{Q}_n$

## (3) 次态方程

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

IF  $J=K=T$



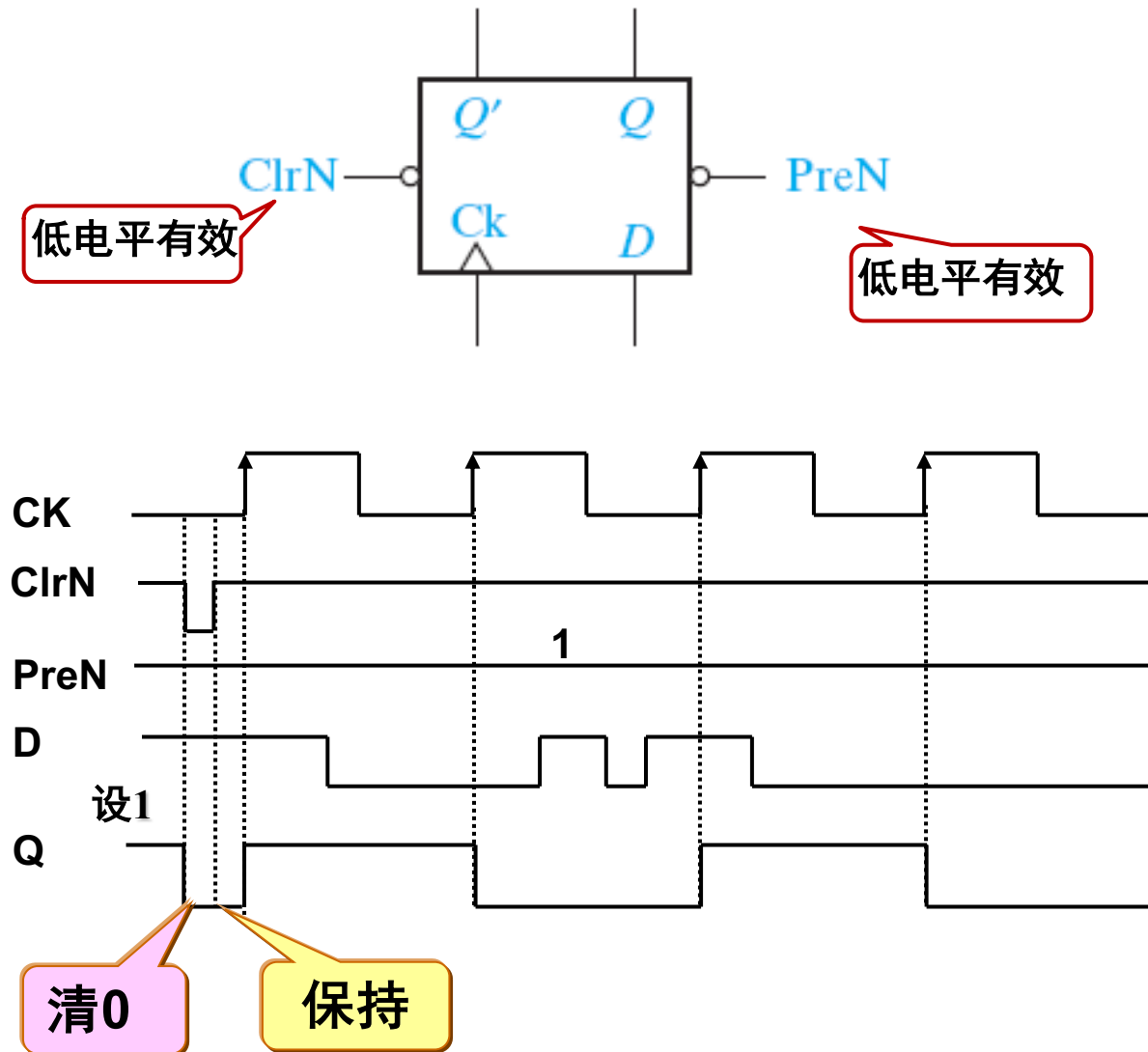
$$\begin{aligned} Q_{n+1} &= T \bar{Q}_n + T Q_n \\ &= T \oplus Q_n \end{aligned}$$

# 带附加输入端的边沿触发器

## 带异步清零端和异步置1端

异步：独立于  
时钟信号

用途：为触发器  
设置指定状态



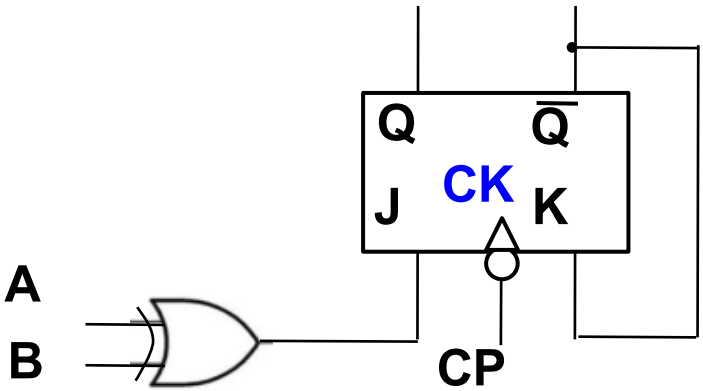
时钟端 CK	输入端 D	异步置1端 PreN	异步清零端 ClrN	次态 $Q_{n+1}$
X	X	0	0	不允许
X	X	0	1	1
X	X	1	0	0
↑	0	1	1	0
↑	1	1	1	1
0,1, ↓	X	1	1	$Q_n$

$$Q_{n+1} = D$$

# JK触发器的应用实例2

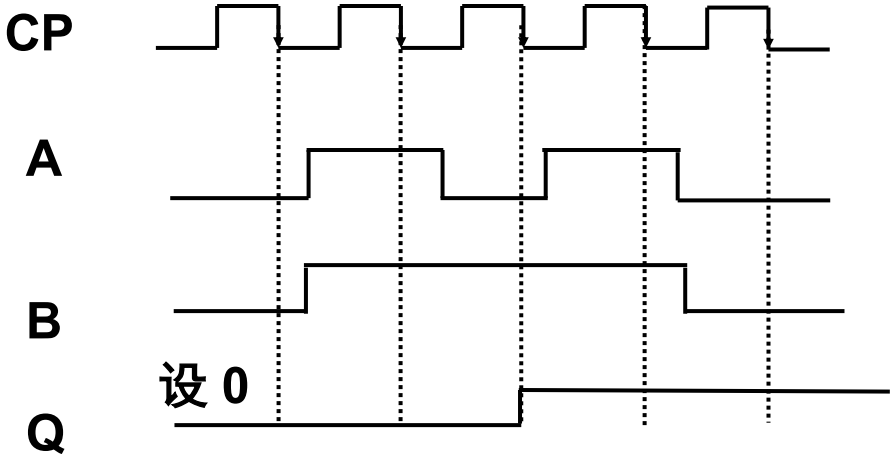
例2：画出Q端波形图

方法1：写出JK触发器的次态方程



$$\begin{aligned} Q_{n+1} &= J \bar{Q}_n + \bar{K} Q_n \\ &= (A \oplus B) \bar{Q}_n + Q_n Q_n \\ &= A \oplus B + Q_n \end{aligned}$$

按照次态方程，在每一个时钟下降沿画出Q端波形



方法2：在每一个时钟下降沿，计算J和K的取值，从而确定Q端波形

- 第1个↓：J=0，K=1 置0功能
- 第2个↓：J=0，K=1 置0功能
- 第3个↓：J=1，K=1 翻转功能
- 第4个↓：J=0，K=0 保持功能
- 第5个↓：J=0，K=0 保持功能

输入端		次态
J	K	$Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	$\bar{Q}_n$

# 边沿触发器——总结

## 时钟边沿触发器的特点

由时钟脉冲边沿确定状态转换的时刻(即何时转换?), 其余时刻都是保持功能

由输入信号确定触发器状态转换的方向(即如何转换?)



思考: 对于一个下降沿触发的JK触发器, 如果让它实现保持功能, 有几种方法可以做到?

### 方法1:

最简单的方法: 不给有效的时钟边沿(此时不用考虑J端和K端的信号)

### 方法

### 方法2:

给时钟下降沿, 此时触发器的保持功能就必须依靠J端和K端的信号配合才能完成

# 目 录

---

- 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

# 触发器类型转换——代数法

- 触发器类型主要有5种，用到最多的是D触发器
- 触发器类型可以相互转换（例如，设计中手头没有需要的触发器类型）

转换方法

- 代数法
- 卡诺图法

代数法

从次态方程入手

## 1. $JK \rightarrow D$ 、 $T$ ( $T'$ )、 $SR$

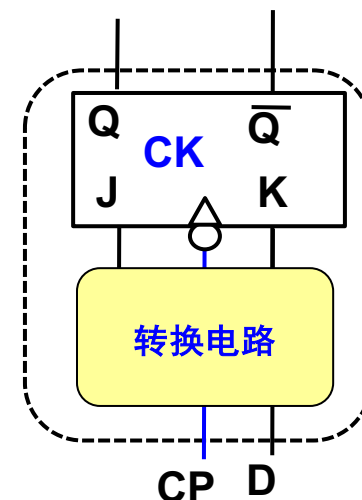
### (1) $JK \rightarrow D$

$$\left. \begin{array}{l} JK: Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n \\ D: Q_{n+1} = D \end{array} \right\}$$

$$D = J \bar{Q}_n + \bar{K} Q_n$$

$$D(Q_n + \bar{Q}_n) = J \bar{Q}_n + \bar{K} Q_n$$

$$\begin{aligned} J &= f(D, Q) \\ K &= f(D, Q) \end{aligned}$$



$$\begin{cases} J=D \\ \bar{K}=D \quad (K=\bar{D}) \end{cases}$$

# 触发器类型转换——卡诺图法

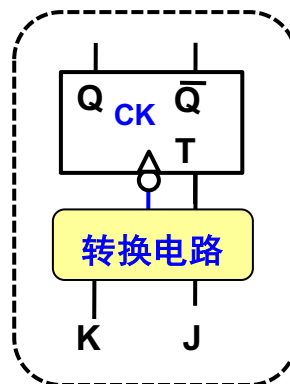
## □ 卡诺图法

### 1. $T \rightarrow JK$ 、 $D$ 、 $SR$

#### (1) $T \rightarrow JK$

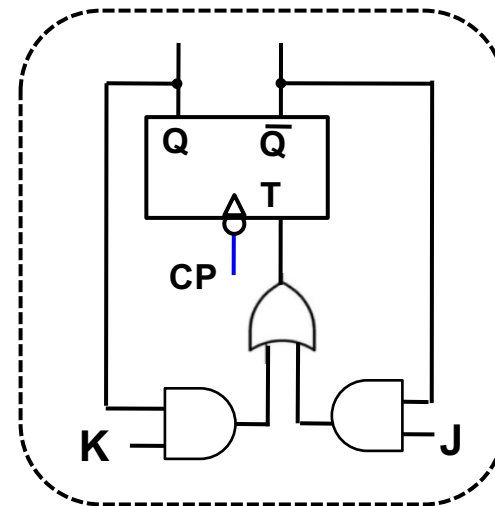
$Q_n$	$\rightarrow$	$Q_{n+1}$	$T$	$J$	$K$
0	$\rightarrow$	0	0	0	X
0	$\rightarrow$	1	1	1	X
1	$\rightarrow$	0	1	X	1
1	$\rightarrow$	1	0	X	0

$$T = f(J, K, Q)$$



$$T = JQ_n + KQ_n$$

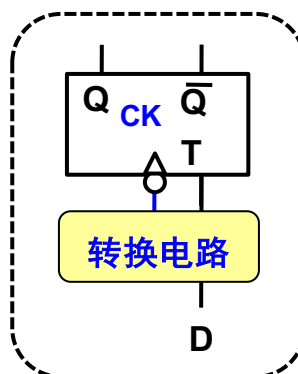
$Q_n$	$JK$			
	00	01	11	10
0	0	0	1	1
1	0	1	1	0



#### (2) $T \rightarrow D$

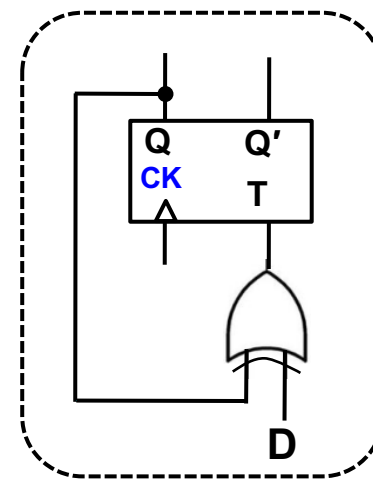
$Q_n$	$\rightarrow$	$Q_{n+1}$	$T$	$D$
0	$\rightarrow$	0	0	0
0	$\rightarrow$	1	1	1
1	$\rightarrow$	0	1	0
1	$\rightarrow$	1	0	0

$$T = f(D, Q)$$



$$T = D \oplus Q_n$$

$D$	$Q_n$	
	0	1
0	0	1
1	1	0



# 第9章 寄存器和计数器

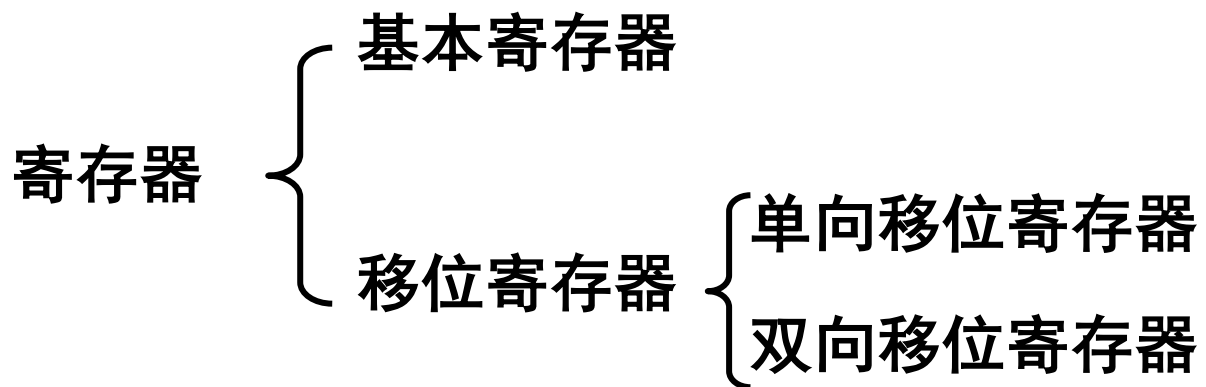
---

- 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

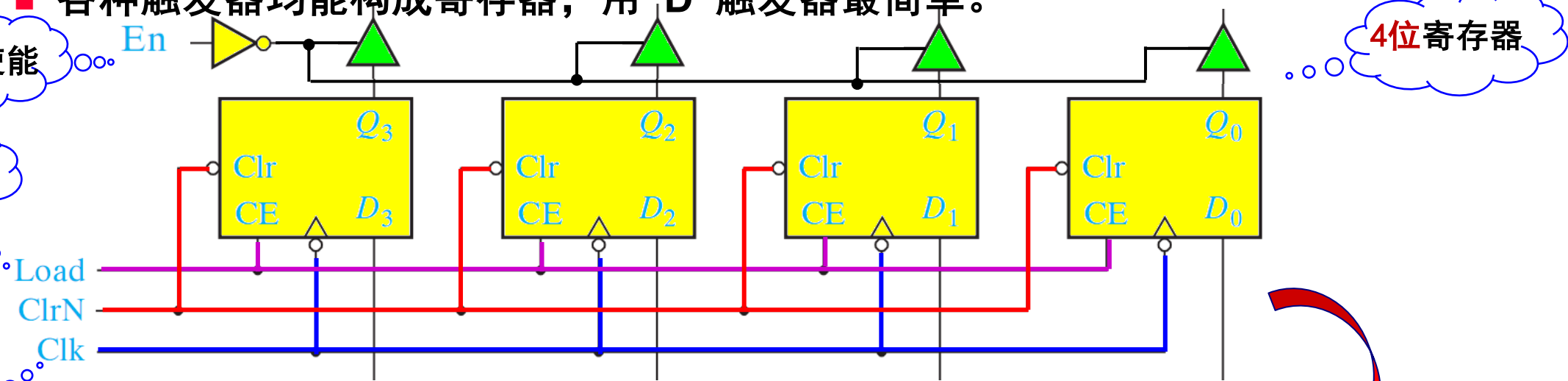
- 寄存器是计算机的一个重要部件，用于暂时存放一组二值代码（如参加运算的数据、运算结果、指令等）。
- 由触发器及控制门组成



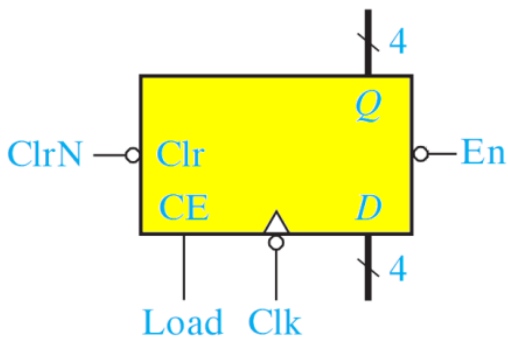
- 基本寄存器的操作：读出/写入/复位（清零）
- 移位寄存器的操作：读出/写入/复位（清零）/左移（右移）

# 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。



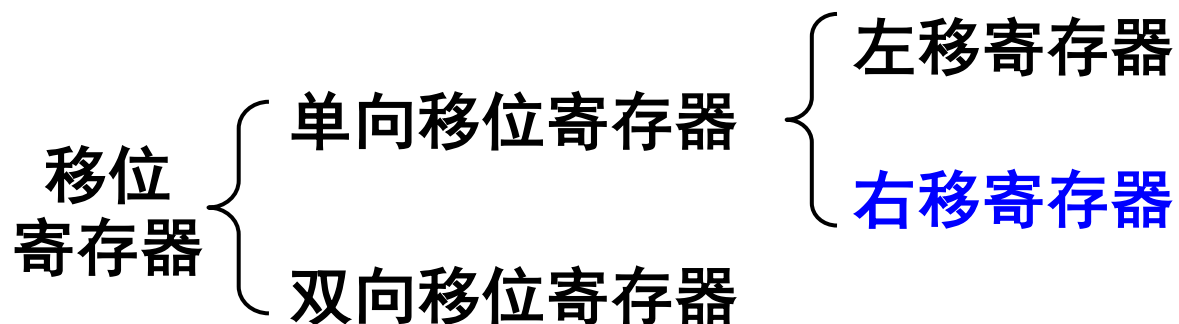
功能	条件	寄存器输出
异步清零	ClrN=0	$Q_3Q_2Q_1Q_0=0000$
保持	ClrN=1, 且 Load=0	$Q^{n+1}_3Q^{n+1}_2Q^{n+1}_1Q^{n+1}_0=Q^n_3Q^n_2Q^n_1Q^n_0$
写入	ClrN=1, Load=1, clk ↓	$Q_3Q_2Q_1Q_0=D_3D_2D_1D_0$
读出	En=0	$Q_3Q_2Q_1Q_0=D_3D_2D_1D_0$



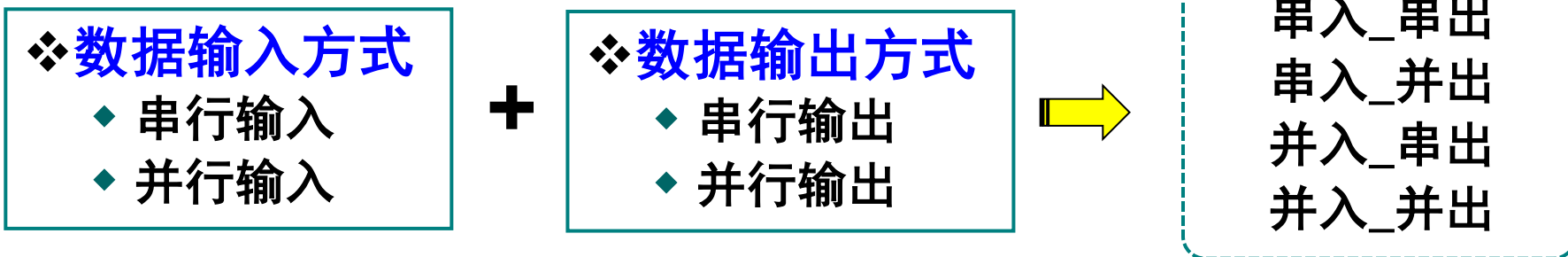
# 移位寄存器定义和分类

- 每来一个时钟脉冲，寄存器里存储的数据，能依次左移或右移1位。
- 可以实现代码的串、并行转换、数值运算和数据处理等。

- 分类



- 工作方式

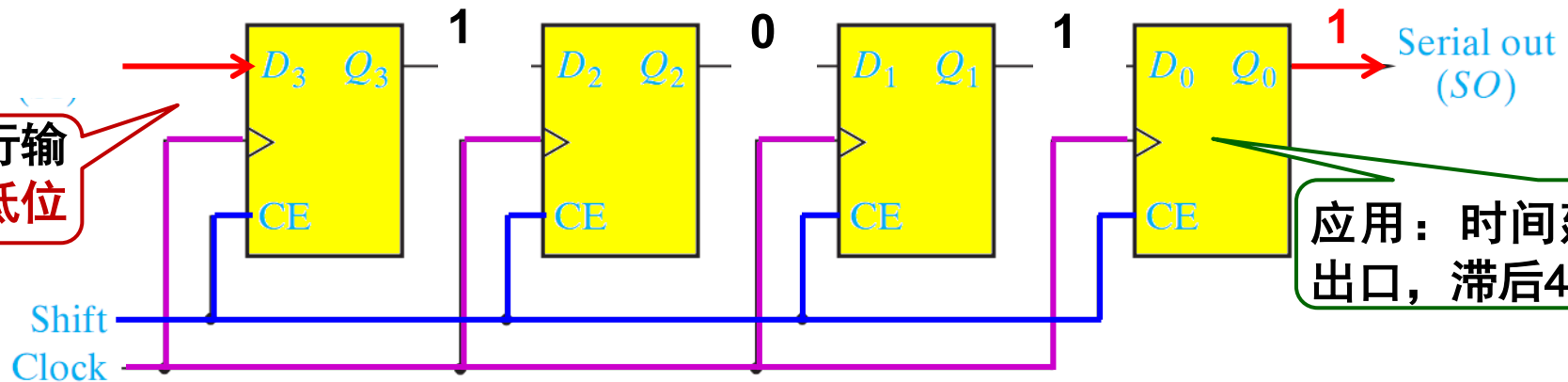


# 右移寄存器 (Right-Shift Register)

## (1). 串行输入/串行输出 ( Serial in / Serial out )

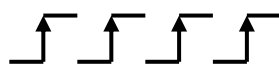
- 串行输出：移位路径上最后一个触发器的输出作为整个电路的输出

右移方式下：数据从串行输入端送入，应该**先送最低位**



应用：时间延迟，入口到出口，滞后4个时钟周期

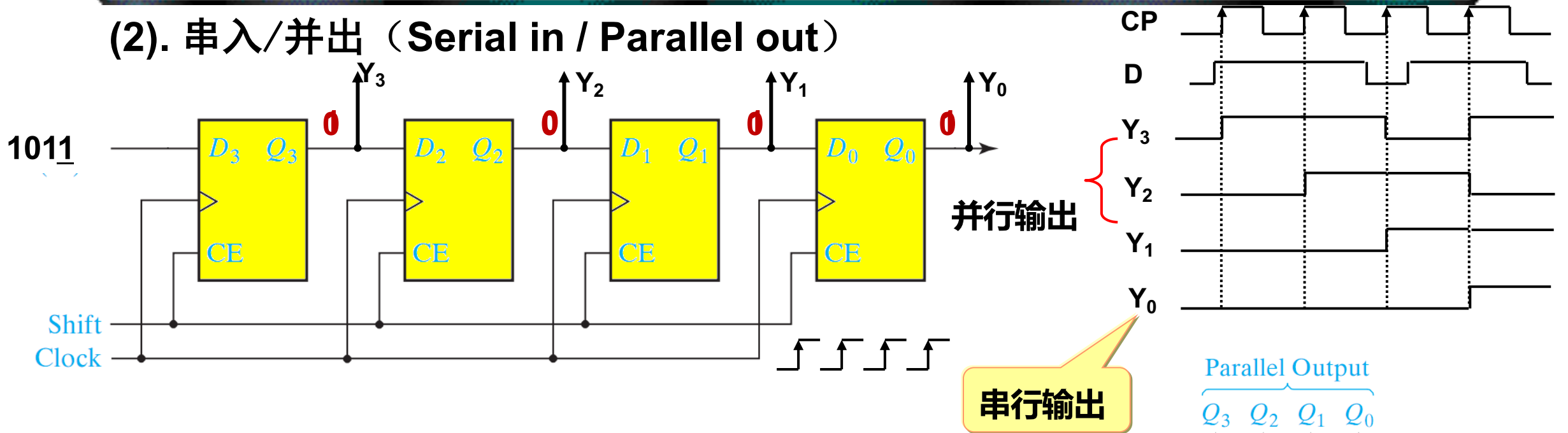
同步时序



思考：左移寄存器如何设计？左移方式下从串行输入端送入数据，应该先送最低位还是最高位？

# 右移寄存器

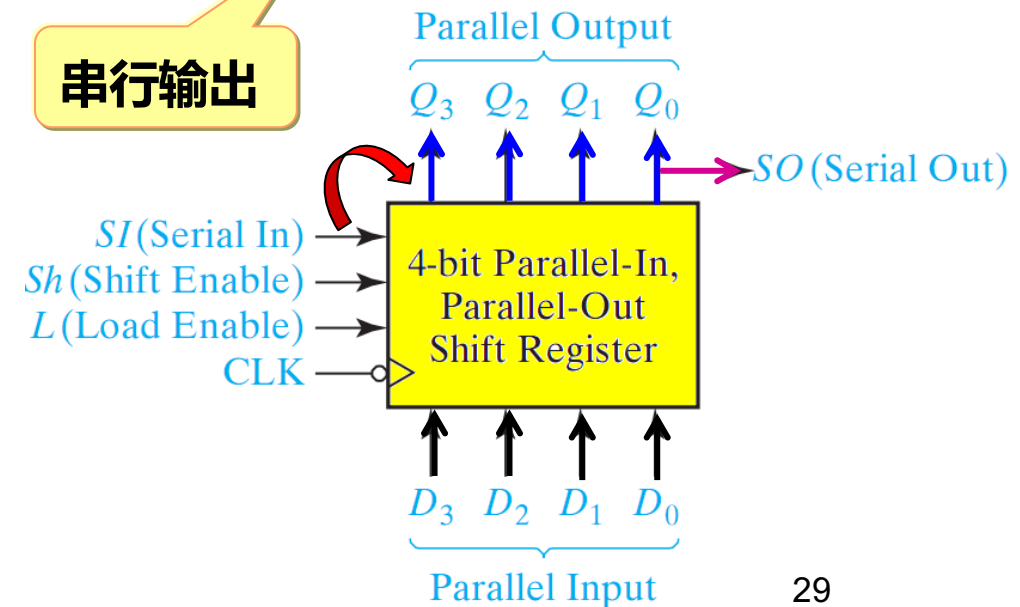
## (2). 串入/并出 (Serial in / Parallel out)



## (3). 并入/并出 (Parallel in / Parallel out)

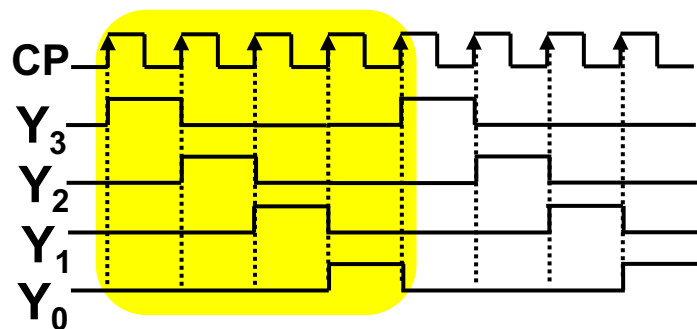
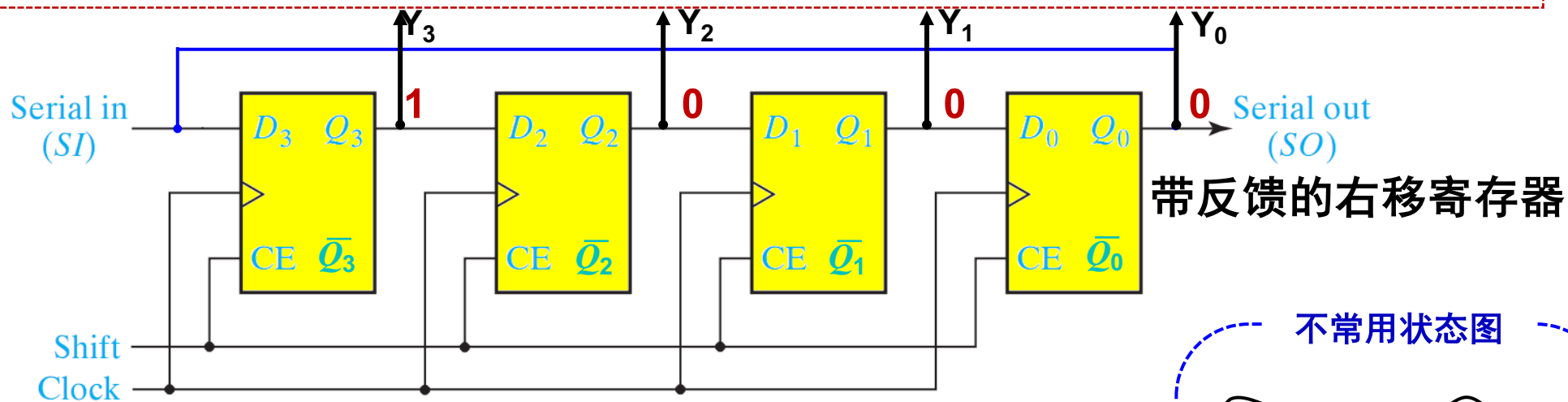
## (4). 并入/串出 (Parallel in / Serial out)

Inputs		Next State				Action
Sh (Shift)	L (Load)	$Q_3^+$	$Q_2^+$	$Q_1^+$	$Q_0^+$	
0	0	$Q_3$	$Q_2$	$Q_1$	$Q_0$	No change
0	1	$D_3$	$D_2$	$D_1$	$D_0$	Load
1	X	SI	$Q_3$	$Q_2$	$Q_1$	Right shift



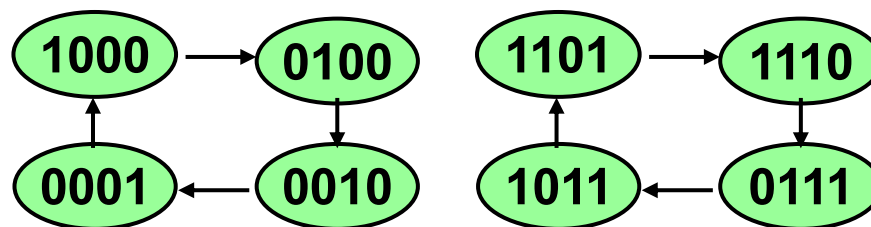
# 右移寄存器的应用——环形计数器

**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。



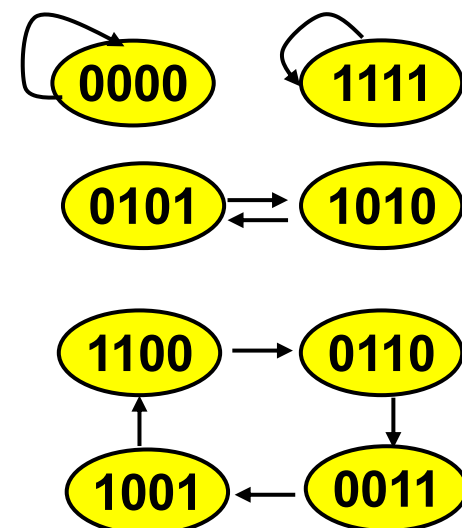
优点：电路简单  
输出具有二进制译码器的特点

常用状态图



缺点： $2^n$ 个状态只用了 $n$ 个；  
不能自启动，需要**预置**

不常用状态图



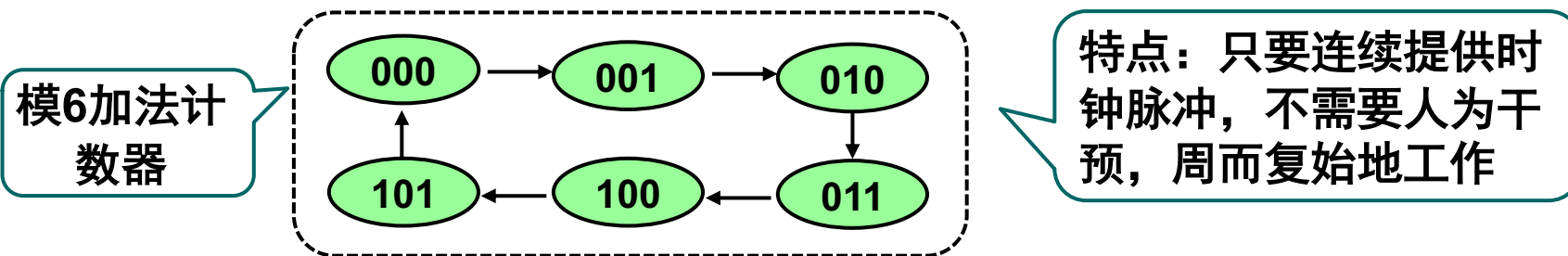
# 寄存器总结

---

- ❑ 寄存器主要功能：存放二进制数据（存储位数由里面触发器的数量决定）
- ❑ 寄存器操作：写入、读出、保持、清零。
- ❑ 移位寄存器还可以：将数据依次左移或右移1位
- ❑ 特点：寄存器的每一个操作（写入、读出、左移、右移）都是在**节拍**的控制下完成的

# 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。



# 计数器

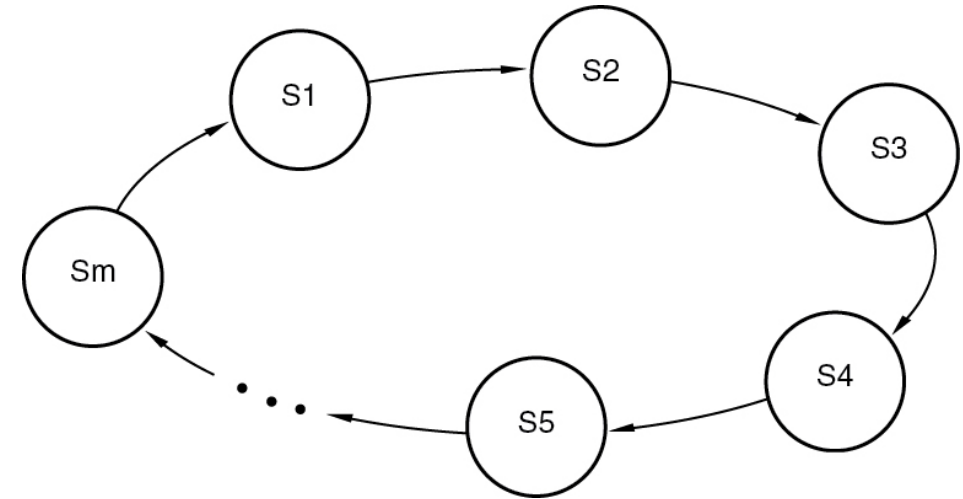
➤ 状态图中包含一个循环的任何时钟时序电路都可称为计数器

➤ 计数器的模是循环中的状态个数

➤ 最常用的是n位二进制计数器

➤ 有n个触发器

➤  $2^n$ 种状态,  $0, 1, 2, \dots, 2^n-1, 0, 1, \dots$



# 计数器的种类

- (1) 按时钟方式分为：同步计数器和异步计数器（行波计数器 ripple counter）；
- (2) 按功能分为：加法计数器、减法计数器和可逆计数器等。
- (3) 按计数方式分为：二进制计数器，十进制计数器，M进制计数器

## 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）

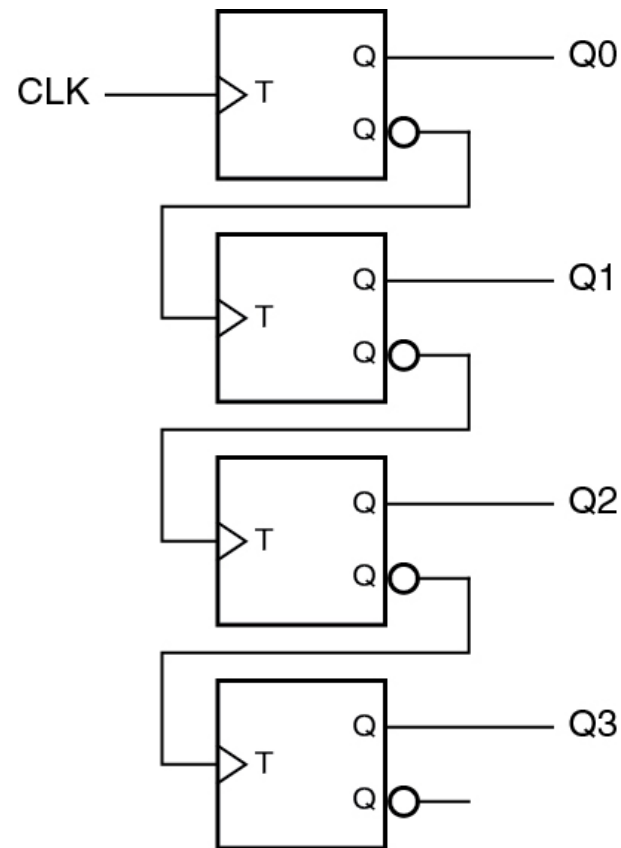
- ① 列驱动方程（控制函数）
- ② 列输出方程（输出函数）——组合逻辑
- ③ 列状态方程（次态方程）——时序逻辑
- ④ 列写状态转换表
- ⑤ 画出状态图
- ⑥ 画出波形图（如必要）

# 行波计数器(异步计数器)

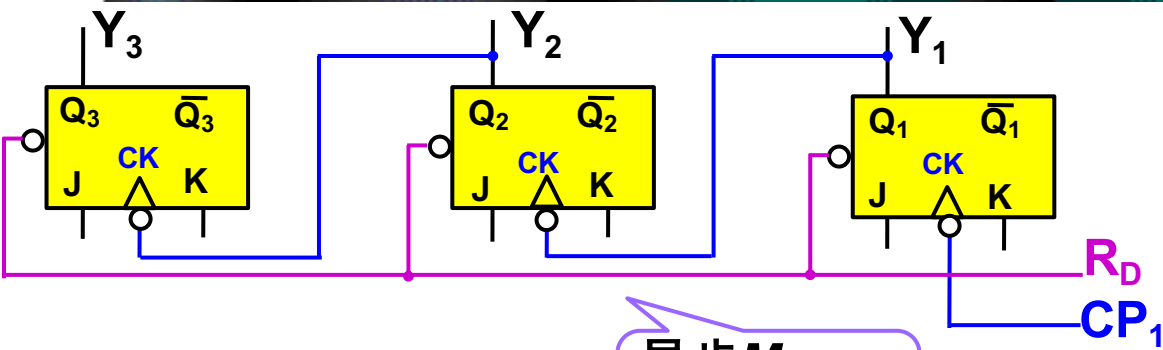
➤ 只用 $n$ 个触发器即可实现 $n$ 位二进制计数器

➤ 各个触发器的时钟信号不同。

➤ T' 触发器每个时钟上升沿都会产生反转  
利用此特性，模拟二进制计数器的进位



# 异步计数器



### ① 输入方程

$J_1 = K_1 = 1 \quad CP_1 \downarrow$   
 $J_2 = K_2 = 1 \quad CP_2 = Y_1 \downarrow$   
 $J_3 = K_3 = 1 \quad CP_3 = Y_2 \downarrow$

### ② 次态方程

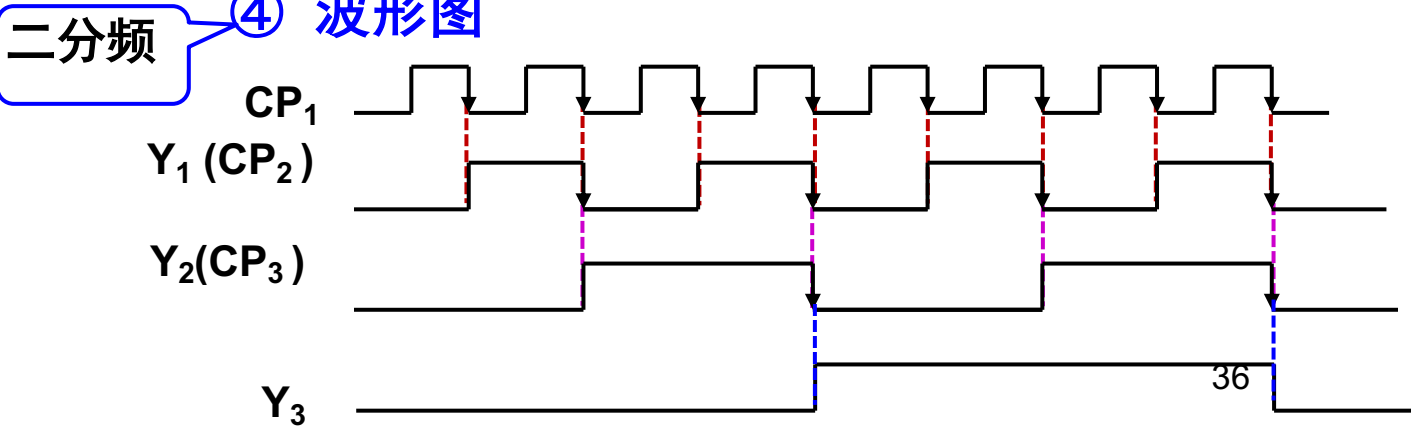
$Y_1^{n+1} = J_1 \bar{Q}_1 + \bar{K}_1 Q_1 = \bar{Y}_1 \quad CP_1 \downarrow$   
 $Y_2^{n+1} = J_2 \bar{Q}_2 + \bar{K}_2 Q_2 = \bar{Y}_2 \quad Y_1 \downarrow$   
 $Y_3^{n+1} = J_3 \bar{Q}_3 + \bar{K}_3 Q_3 = \bar{Y}_3 \quad Y_2 \downarrow$

### ③ 状态转换表

现态			次态			时钟		
$Y_3^n$	$Y_2^n$	$Y_1^n$	$Y_3^{n+1}$	$Y_2^{n+1}$	$Y_1^{n+1}$	$CP_3$	$CP_2$	$CP_1$
0	0	0	0	0	1	无	无	↓
0	0	1	0	1	0	无	↓	↓
0	1	0	0	1	1	无	无	↓
0	1	1	1	0	0	↓	↓	↓
1	0	0	1	0	1	无	无	↓
1	0	1	1	1	0	无	↓	↓
1	1	0	1	1	1	无	无	↓
1	1	1	0	0	0	↓	↓	↓

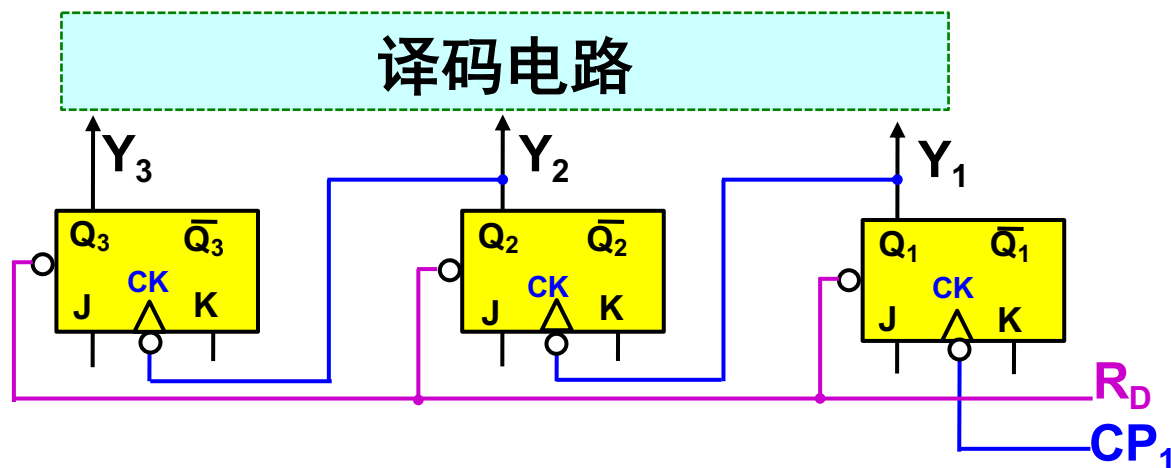
异步模8加法计数器

### ④ 波形图

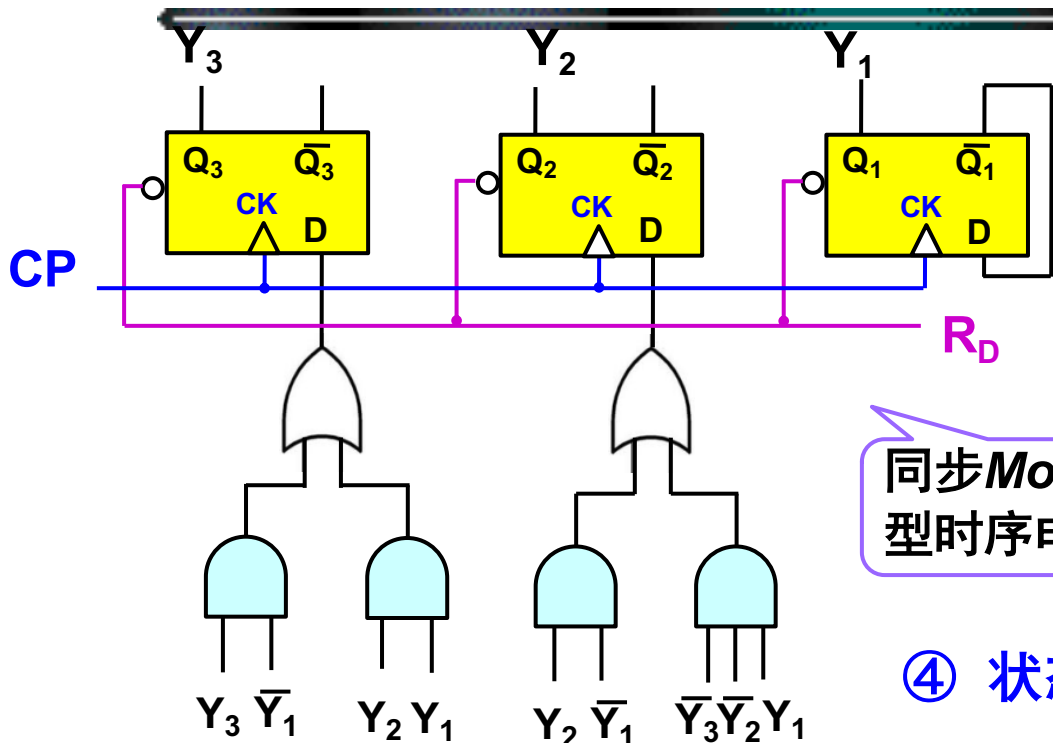


# 异步计数器总结

- 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。



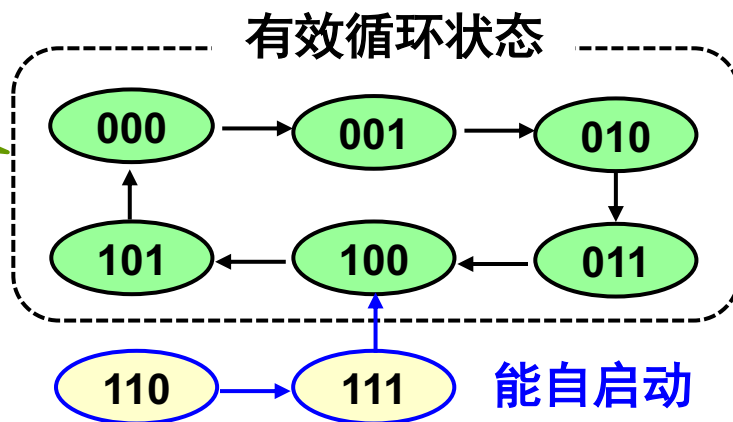
# 同步计数器



同步Moor  
型时序电路

同步模6加  
法计数器

## ④ 状态图



## ① 输入方程

$$D_3 = Y_3 \bar{Y}_1 + Y_2 Y_1$$

$$D_2 = Y_2 \bar{Y}_1 + \bar{Y}_3 \bar{Y}_2 Y_1$$

$$D_1 = \bar{Y}_1$$

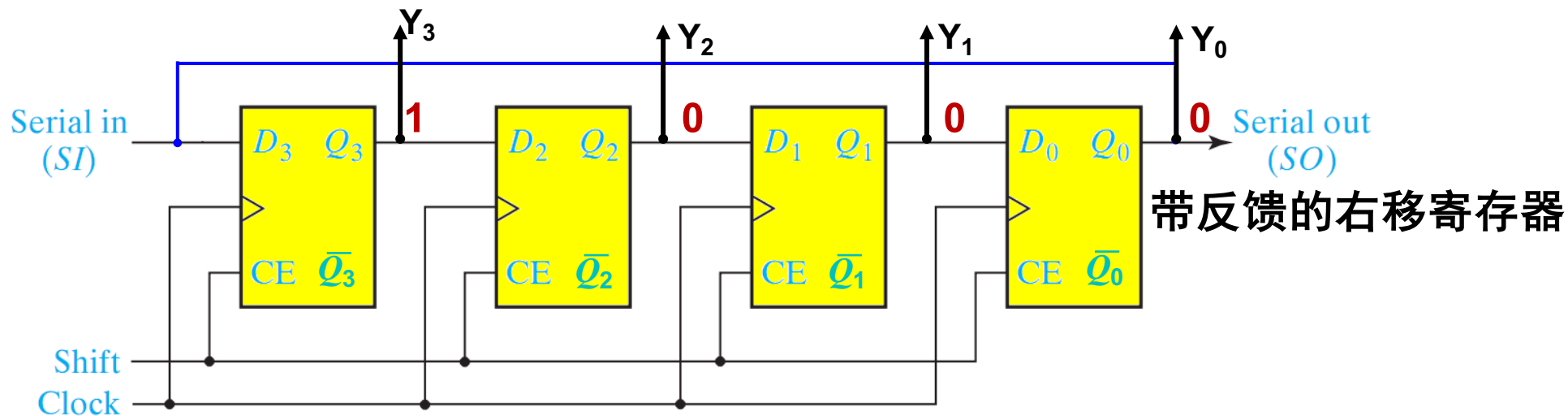
## ② 次态方程

$$\begin{cases} Y_1^{n+1} = D_1 \\ Y_2^{n+1} = D_2 \\ Y_3^{n+1} = D_3 \end{cases}$$

## ③ 状态转换表

现态			次态			时钟
$Y_3^n$	$Y_2^n$	$Y_1^n$	$Y_3^{n+1}$	$Y_2^{n+1}$	$Y_1^{n+1}$	CP
0	0	0	0	0	1	↑
0	0	1	0	1	0	↑
0	1	0	0	1	1	↑
0	1	1	1	0	0	↑
1	0	0	1	0	1	↑
1	0	1	0	0	0	↑
1	1	0	1	1	1	↑
1	1	1	1	0	0	↑

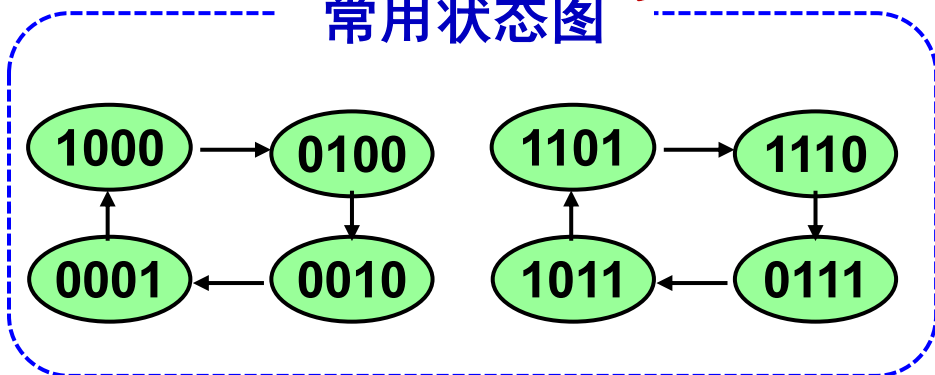
# 同步计数器——环形计数器



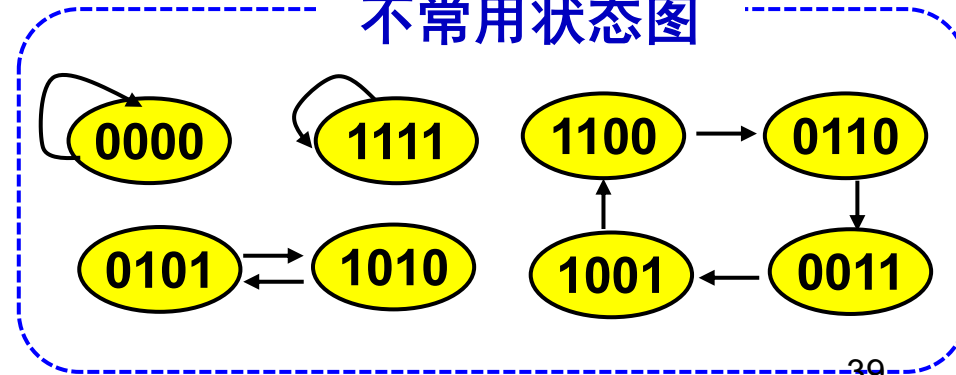
优点：电路简单，  
输出具有二进制译  
码器的特点

缺点： $2^n$ 个状态只使用了 $n$ 个；  
不能自启动，需要预置

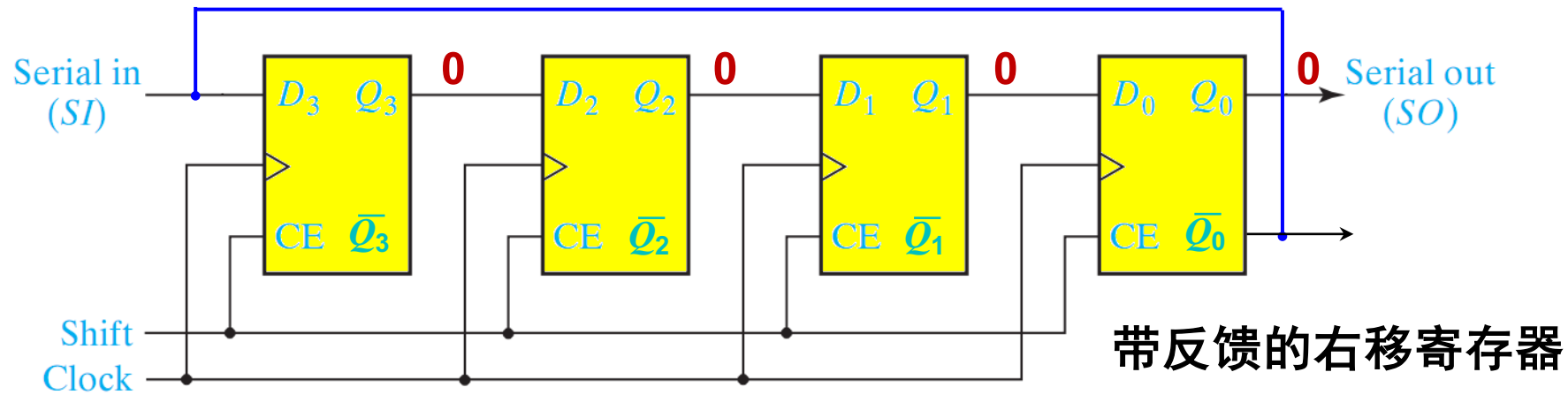
常用状态图



不常用状态图

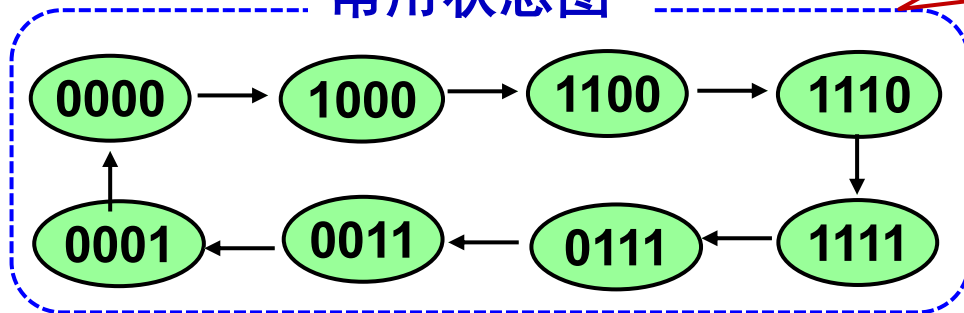


# 同步计数器——扭环形计数器Johnson Counter



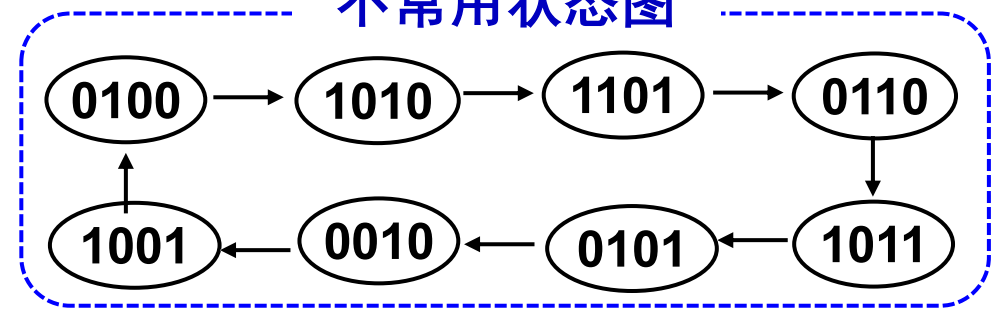
电路具有格雷码输出的特点

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





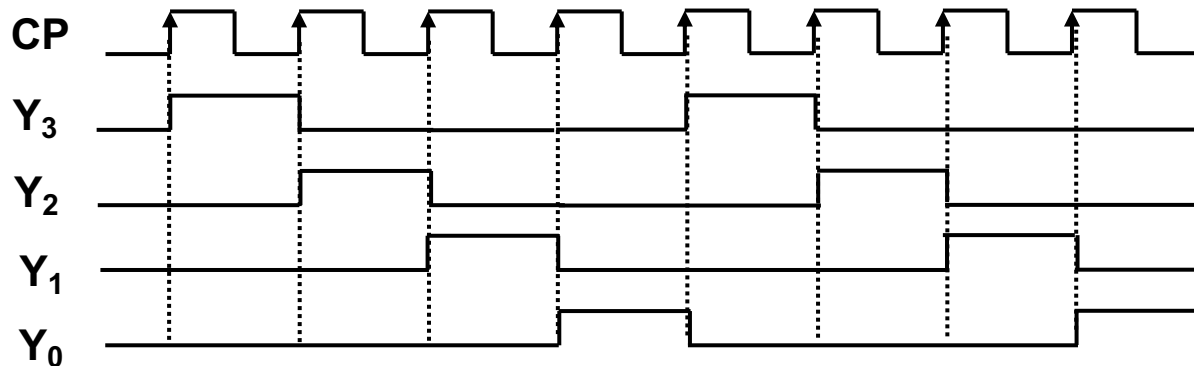
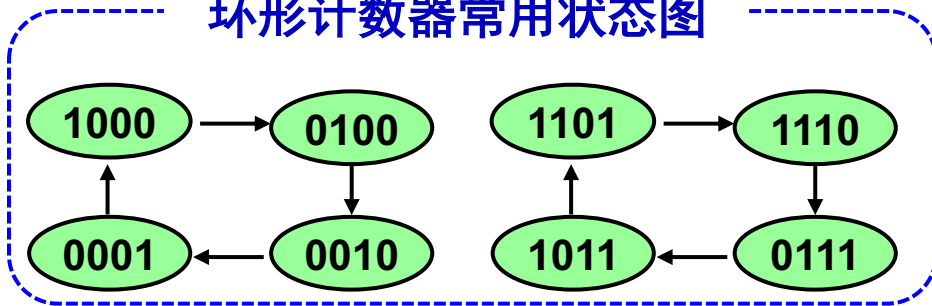
# 环形、扭环形计数器总结

特点：在移位寄存器的基础上，增加反馈逻辑电路组成。

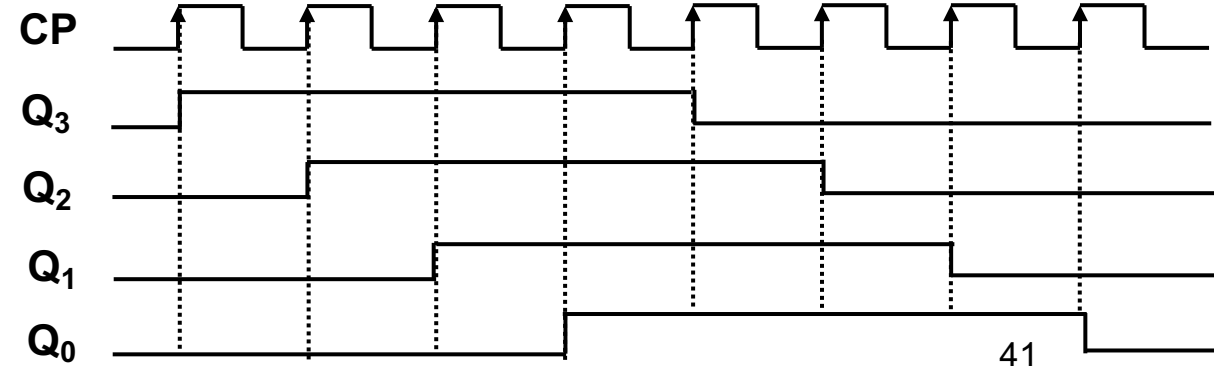
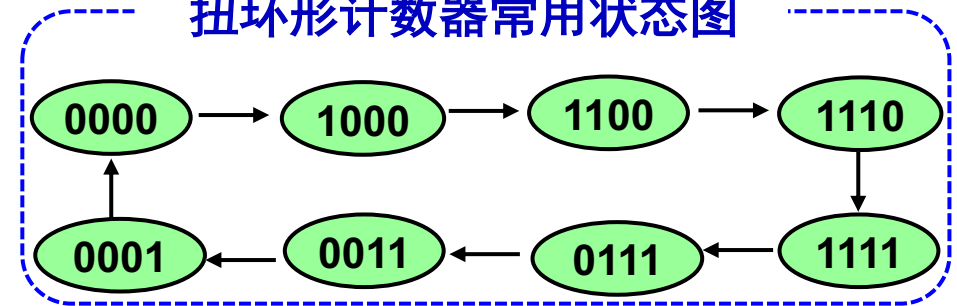
用途：

- 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



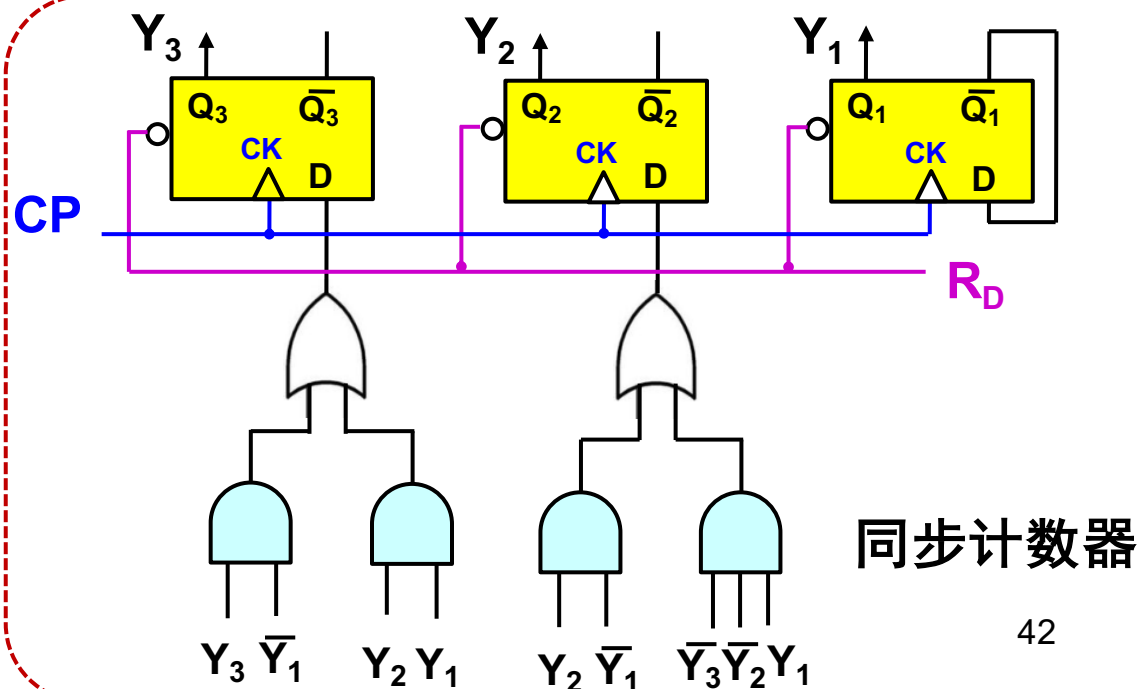
扭环形计数器常用状态图



# 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。

异步计数器



# 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)

# 典型时序逻辑部件——节拍发生器(时序发生器)

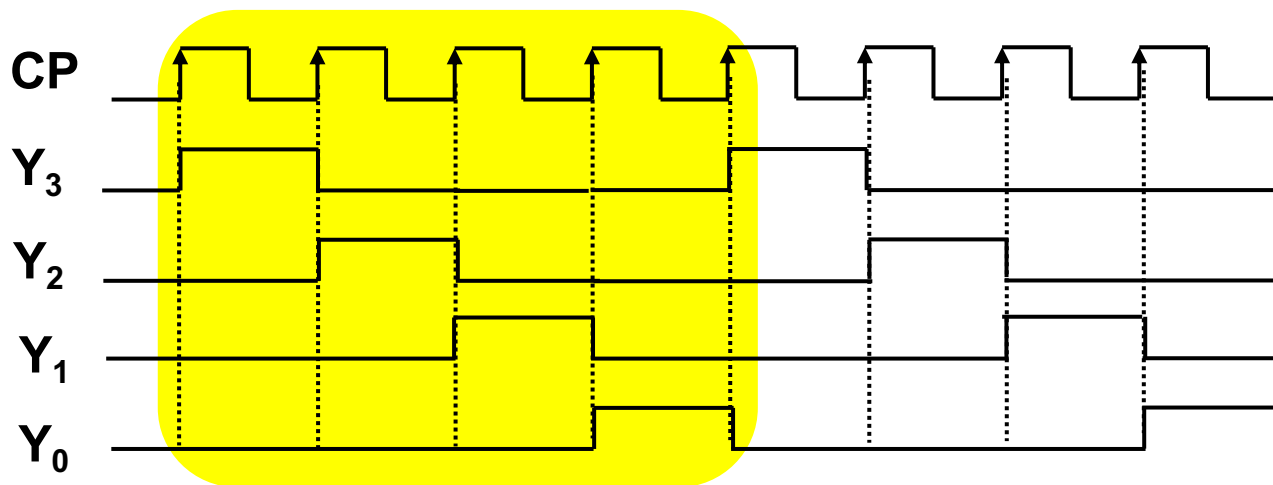
## 定义

每个循环周期内, 在时钟脉冲的作用下, 产生一组在时间上有一定**先后顺序**的脉冲信号

## 作用

数字系统和计算机的控制部件利用顺序脉冲, 形成所需要的各种控制信号, 使某些设备按照事先规定的顺序进行运算或操作

例: 将4位二进制数(如1000)存入某寄存器, 然后将数据右移1位, 之后将数据读走, 再将右移后的数据左移1位。以上操作可以自动循环进行。



- ①执行写入操作: 写入使能有效 (存入1000)
- ②执行右移操作: 右移使能有效 (右移后0100)
- ③执行读出操作: 读出使能有效
- ④执行左移操作: 左移使能有效 (左移后1000)

# 时序逻辑电路分析

## 时序逻辑电路的分析方法

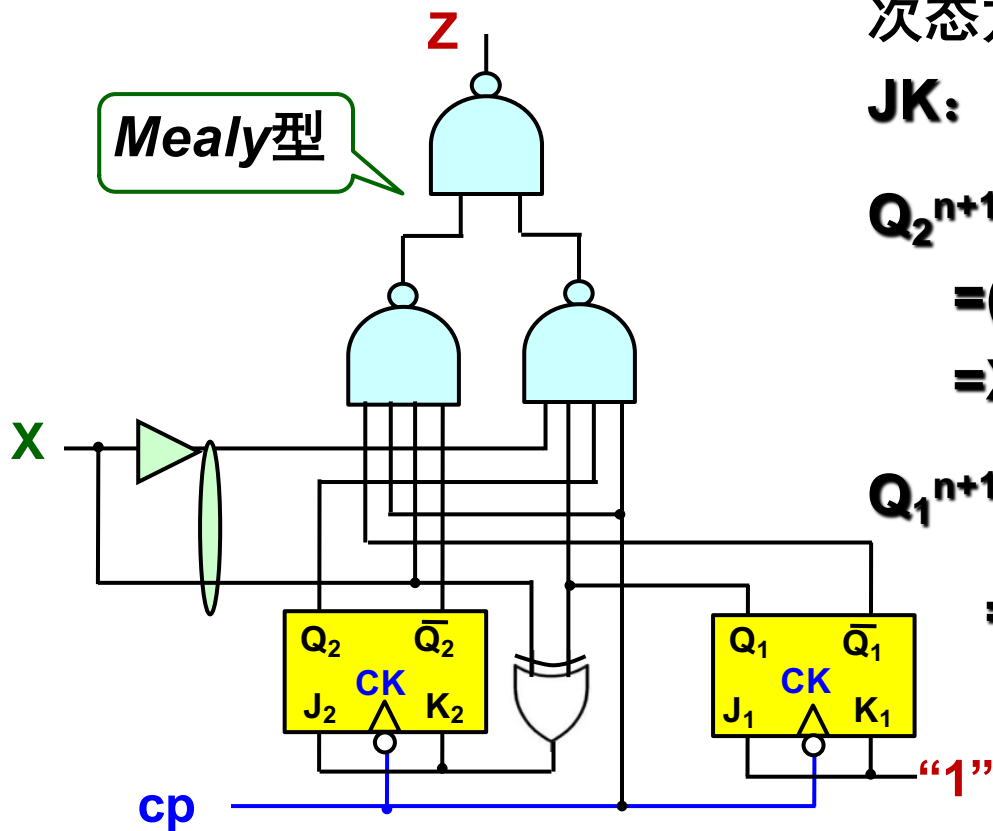
确定系统变量（输入变量、输出变量、状态变量）

- 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——示例1：同步时序



① 输入方程

$$J_1 = K_1 = 1$$

$$J_2 = K_2 = X \oplus Q_1^n$$

② 次态方程

$$Q_2^{n+1} = X \oplus Q_1^n \oplus Q_2^n$$

$$Q_1^{n+1} = \overline{Q_1^n}$$

次态方程：

③ 输出方程

$$JK: Q^{n+1} = JQ^n + K\overline{Q}^n$$

$$\begin{aligned} Q_2^{n+1} &= J_2 \overline{Q_2^n} + K_2 Q_2^n \\ &= (X \oplus Q_1^n) \overline{Q_2^n} + (X \oplus Q_1^n) Q_2^n \\ &= X \oplus Q_1^n \oplus Q_2^n \end{aligned}$$

$$\begin{aligned} Q_1^{n+1} &= J_1 \overline{Q_1^n} + K_1 Q_1^n \\ &= \overline{Q_1^n} \end{aligned}$$

$$\begin{aligned} Z &= \overline{XCPQ_2^n \overline{Q_1^n}} \cdot \overline{XCPQ_2^n Q_1^n} \\ &= XCP\overline{Q_2^n} \overline{Q_1^n} + \overline{XCP} Q_2^n Q_1^n \end{aligned}$$

④ 状态转换表

输入	现态		次态		输出
X	$Q_2^n$	$Q_1^n$	$Q_2^{n+1}$	$Q_1^{n+1}$	Z
0	0	0	0	1	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	0	1	0
1	1	1	1	0	0

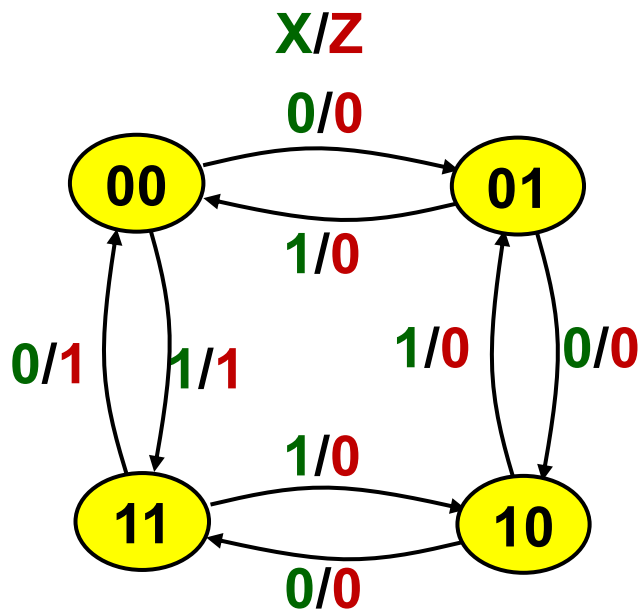
现态 $Q_2^n \quad Q_1^n$	次态 $Q_2^{n+1} \quad Q_1^{n+1} / Z$	
	X=0	X=1
0 0	0 1 / 0	1 1 / 1
0 1	1 0 / 0	0 0 / 0
1 0	1 1 / 0	0 1 / 0
1 1	0 0 / 1	1 0 / 0

# 时序逻辑电路分析——示例1:同步时序

## ④ 状态转换表

现态 $Q_2^n \ Q_1^n$	$Q_2^{n+1} \ Q_1^{n+1} / Z$	
	$X=0$	$X=1$
0 0	0 1 / 0	1 1 / 1
0 1	1 0 / 0	0 0 / 0
1 0	1 1 / 0	0 1 / 0
1 1	0 0 / 1	1 0 / 0

## ⑤ 状态图



结论：模4可逆计数器

- $X=0$ : 加计数

- $X=1$ : 减计数

$Z$ : 进位和借位输出标志

**Mealy型: 输出值画在状态图转换线的旁边**

# 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：

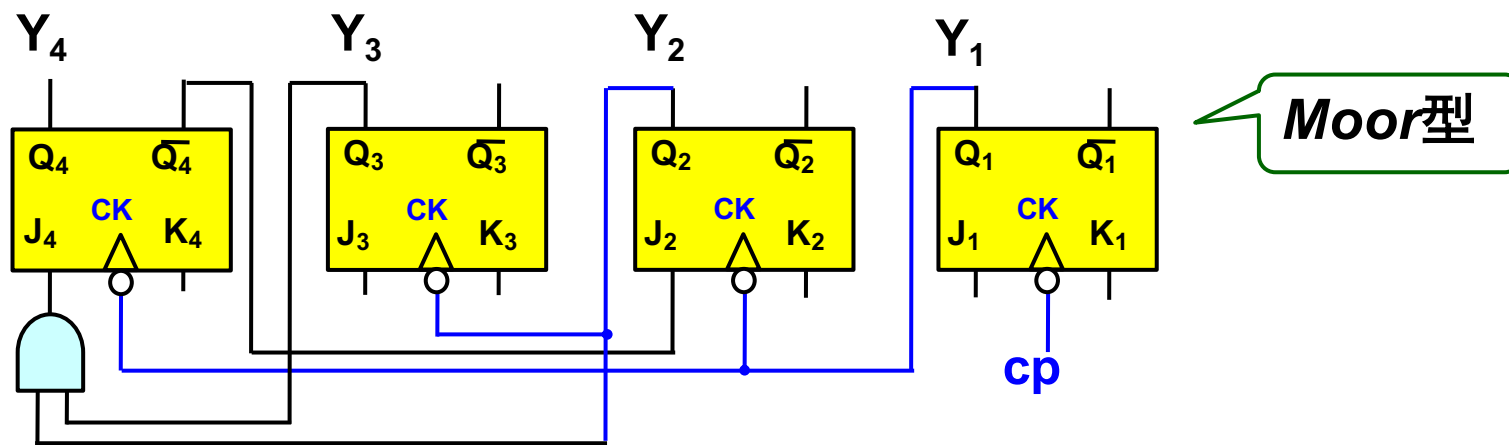
- 写出所有输入及现态的取值组合；
- 将每一种取值组合带入次态方程和输出方程，计算后的得出次态值和输出值；
- 从表中第一行开始，寻找状态转换规律；

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——示例3: 异步时序



## ① 输入方程

$$\begin{cases} J_4 = Y_3^n Y_2^n \\ K_4 = 1 \\ J_3 = K_3 = 1 \\ J_2 = \overline{Y_4^n}, K_2 = 1 \\ J_1 = K_1 = 1 \end{cases}$$

## ② 次态方程

$$\begin{cases} Y_4^{n+1} = J_4 \overline{Y_4^n} + \overline{K_4} Y_4^n = \overline{Y_4^n} Y_3^n Y_2^n & CP_4 = Y_1 \downarrow \\ Y_3^{n+1} = J_3 \overline{Y_3^n} + \overline{K_3} Y_3^n = \overline{Y_3^n} & CP_3 = Y_2 \downarrow \\ Y_2^{n+1} = J_2 \overline{Y_2^n} + \overline{K_2} Y_2^n = \overline{Y_4^n} \overline{Y_2^n} & CP_2 = Y_1 \downarrow \\ Y_1^{n+1} = J_1 \overline{Y_1^n} + \overline{K_1} Y_1^n = \overline{Y_1^n} & CP_1 \downarrow \end{cases}$$

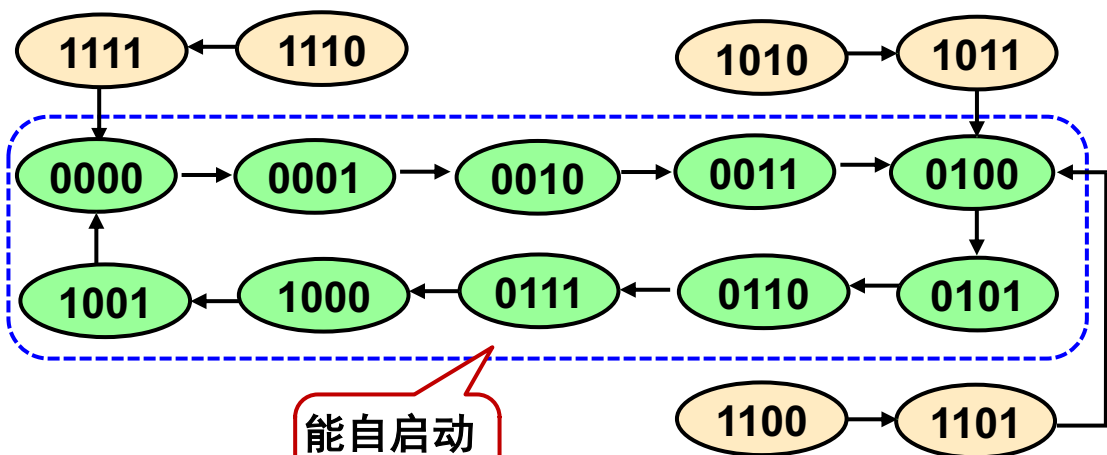
# 时序逻辑电路分析——异步时序示例3

## ② 次态方程

$$\begin{cases} Y_4^{n+1} = J_4 \overline{Y_4^n} + \overline{K_4} Y_4^n = \overline{Y_4^n} Y_3^n Y_2^n & CP_4 = Y_1 \downarrow \\ Y_3^{n+1} = J_3 \overline{Y_3^n} + \overline{K_3} Y_3^n = \overline{Y_3^n} & CP_3 = Y_2 \downarrow \\ Y_2^{n+1} = J_2 \overline{Y_2^n} + \overline{K_2} Y_2^n = \overline{Y_4^n} \overline{Y_2^n} & CP_2 = Y_1 \downarrow \\ Y_1^{n+1} = J_1 \overline{Y_1^n} + \overline{K_1} Y_1^n = \overline{Y_1^n} & CP_1 \downarrow \end{cases}$$

## ④ 状态图

8421 BCD 码异步加法计数器



## ③ 状态转换表

现态				次态				时钟			
$Y_4^n$	$Y_3^n$	$Y_2^n$	$Y_1^n$	$Y_4^{n+1}$	$Y_3^{n+1}$	$Y_2^{n+1}$	$Y_1^{n+1}$	$cp_4$	$cp_3$	$cp_2$	$cp_1$
0	0	0	0	0	0	0	1	无	无	无	↓
0	0	0	1	0	0	1	0	↓	无	↓	↓
0	0	1	0	0	0	1	1	无	无	无	↓
0	0	1	1	0	1	0	0	↓	↓	↓	↓
0	1	0	0	0	1	0	1	无	无	无	↓
0	1	0	1	0	1	1	0	↓	无	↓	↓
0	1	1	0	0	1	1	1	无	无	无	↓
0	1	1	1	1	0	0	0	↓	↓	↓	↓
1	0	0	0	1	0	0	1	无	无	无	↓
1	0	0	1	0	0	0	0	↓	无	↓	↓
1	0	1	0	1	0	1	1	无	无	无	↓
1	0	1	1	0	1	0	0	↓	↓	↓	↓
1	1	0	0	1	1	0	1	无	无	无	↓
1	1	0	1	0	1	0	0	↓	无	↓	↓
1	1	1	0	1	1	1	1	无	无	无	↓
1	1	1	1	0	0	0	0	↓	↓	↓	↓

# 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

# 第10章 同步时序逻辑设计

---

- 状态机基础
- 原始状态图和状态表
- 状态表化简
- 状态分配

# 状态机基础

## 时序电路的状态 (state)

- 是一个状态变量 (state variable) 集合
- 状态变量的值包含决定电路的未来行为的所有信息

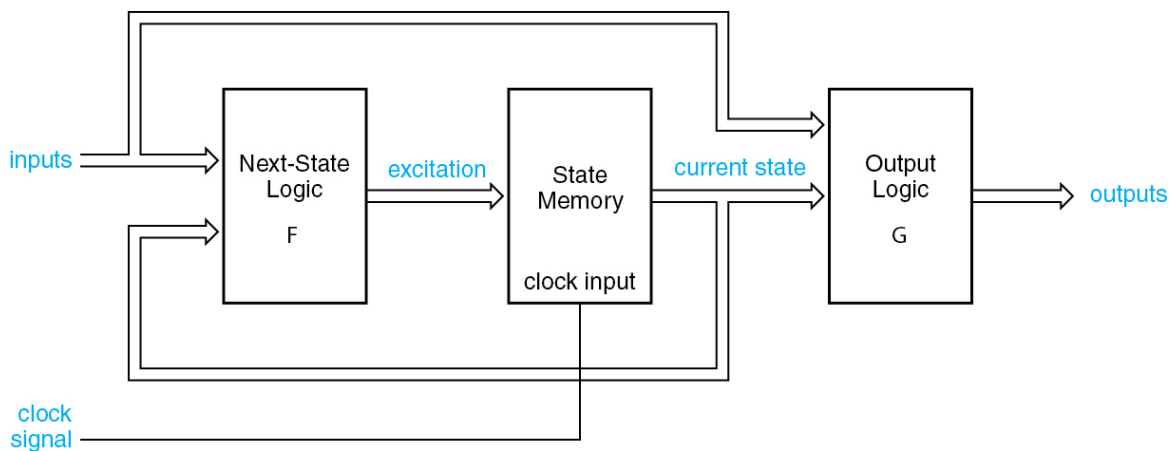
## 状态机

- 具有 $n$ 位二进制状态变量的电路有 $2^n$ 种可能的状态
- 因为时序电路的状态是有限的，所以可将其称为有限状态机 (Finite State Machine)，简称为状态机 (state machine)

# Mealy状态机 vs Moore状态机

## 状态机结构

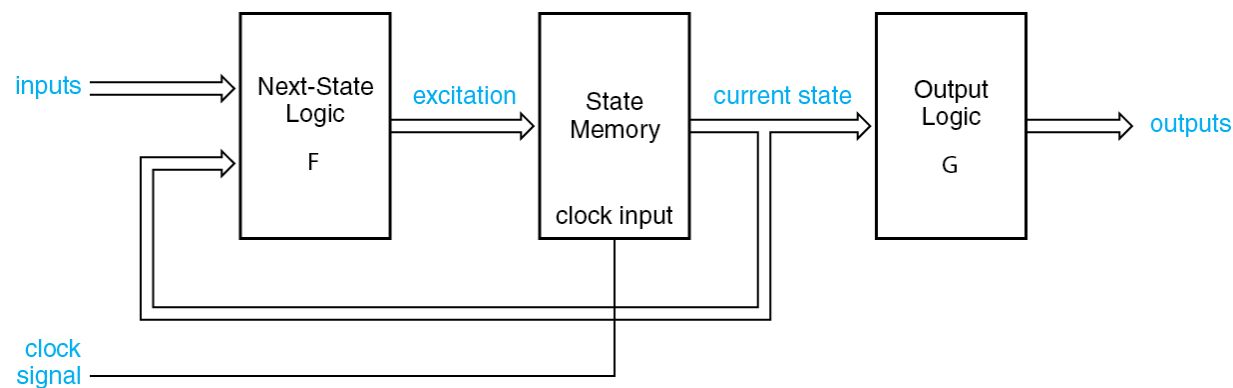
- 状态存储器 (state memory) 是存储状态机现态的一组触发器
- 状态机的次态, 由次态逻辑 (next-state logic)  $F$  确定
- 状态机的输出, 由输出逻辑 (output logic)  $G$  确定



### Mealy状态机

次态 =  $F$ (现态, 输入)

输出 =  $G$ (现态, 输入)



### Moore状态机

输出 =  $G$ (现态)

# 状态机结构和分析

## 状态机的形式定义

次态 =  $F$ (现态, 输入信号)

输出 =  $G$ (现态, 输入信号)

## 进行状态机分析的基本步骤

- 确定次态函数 $F$ 和输出函数 $G$
- 用 $F$ 和 $G$ 构造状态/输出表
- 画出状态图, 表示上述信息

# 同步时序逻辑设计

---

- 状态机基础
- 原始状态图和状态表
- 状态表化简
- 状态分配



# 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

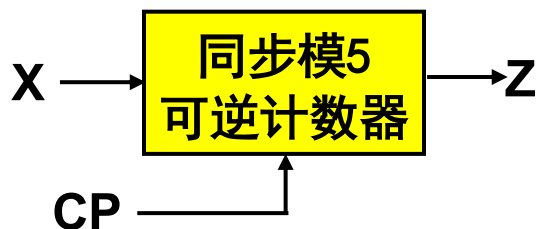
- (1) 根据需求 → 获得原始状态图、状态表
- (2) 最小化状态图、状态表
- (3) 状态编码（分配）→ 获得状态转移表
- (4) 状态转移表  
触发器特征 } → 触发器激励
- (5) 卡诺图化简 → { 激励（输入）函数表达式  
输出函数表达式
- (6) 电路实现      (7) 检查无关项

# 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

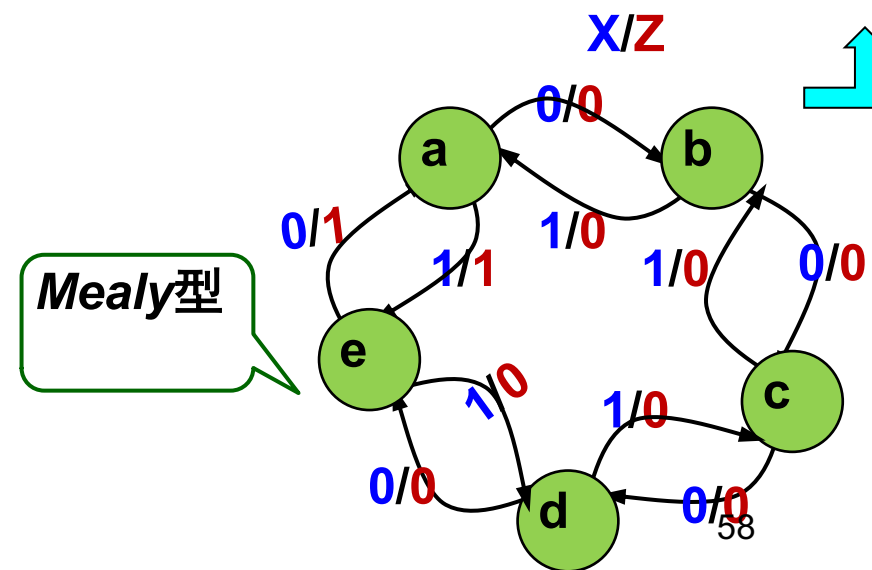


$X=0$  : 加计数

$X=1$  : 减计数

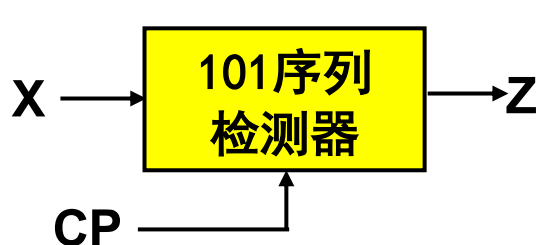
$Z$ : 进位、借位输出标志

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—101序列检测器

例：序列检测——给出同步Mealy型101序列检测器的状态表



X: 0 1 0 1 0 1 1 0 1

Z: 0 0 0 1 0 1 0 0 1

可重叠检测

X: 0 1 0 1 0 1 0 1 1

Z: 0 0 0 1 0 0 0 1 0

不可重叠检测

## (1) 状态设定

$S_0$ ——初始状态，表示收到1位数据：“0”

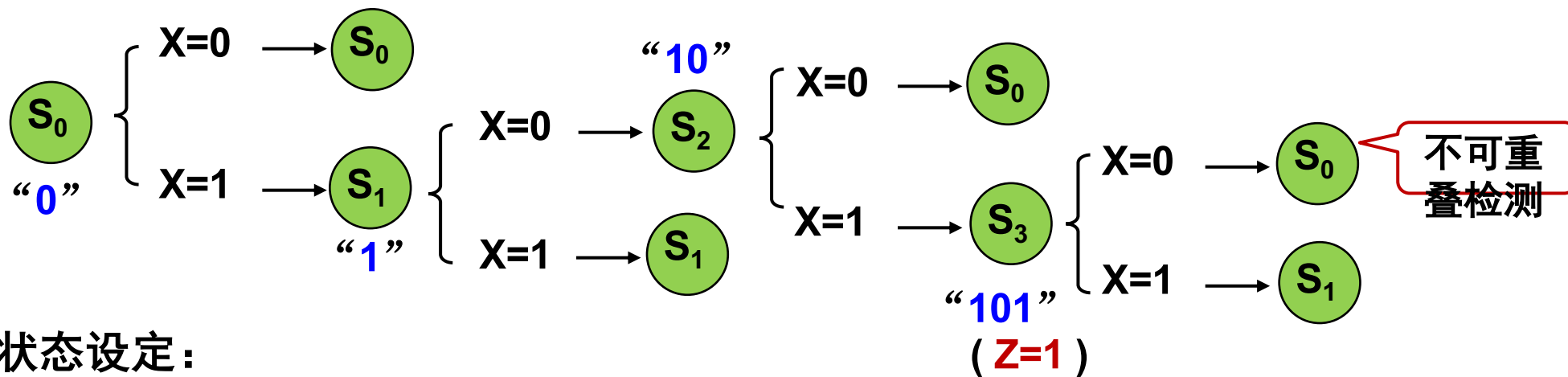
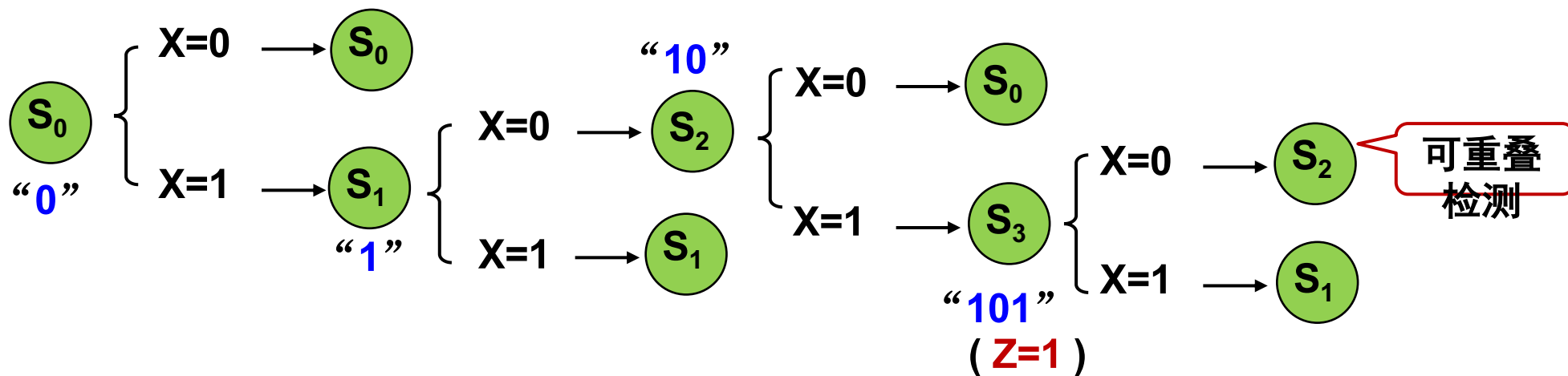
$S_1$ ——表示收到1位数据：“1”

$S_2$ ——表示收到2位数据：“10”

$S_3$ ——表示收到3位数据：“101”，此时输出标志  $Z=1$ .

只标记感兴趣的子串

# 101序列检测器



状态设定:

$S_0$ ——0 ;     $S_1$ —— 1;

$S_2$ —— 10 ;     $S_3$ —— 101 , 且  $Z=1$

# 序列检测电路设计

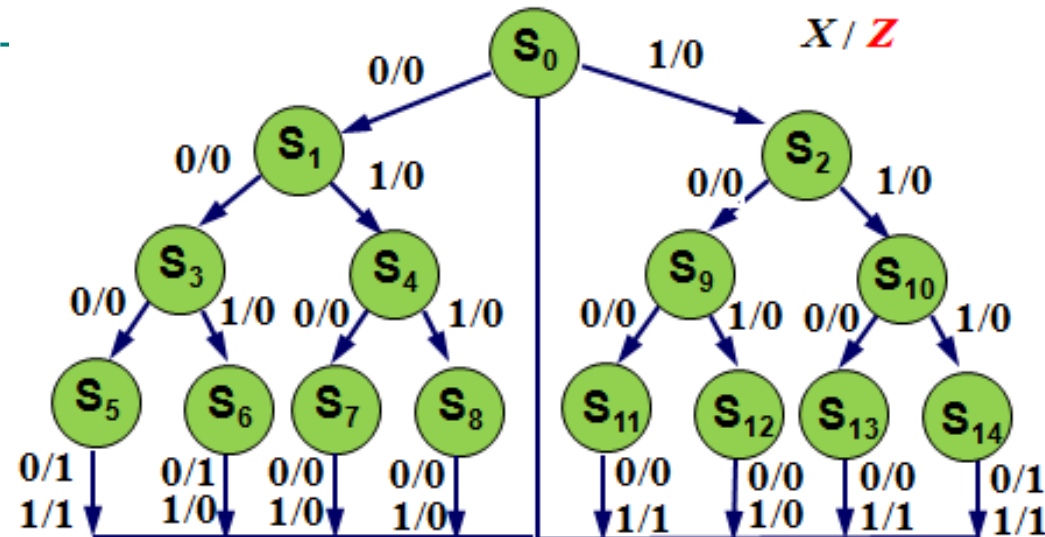
## 序列检测的原始状态图构造方法总结

- (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。



# 同步时序逻辑电路设计方法

## 利用触发器设计时序逻辑的方法

- (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

# 状态表化简

## 等价状态的判定条件

状态表中的任意两个状态  $s_i$  和  $s_j$  同时满足下列两个条件，它们可以合并为一个状态

1. 在所有不同的现输入下，现输出分别相同
2. 在所有不同的现输入下，次态分别为下列情况之一
  - (1) 两个次态完全相同
  - (2) 两个次态为其现态本身或交错
  - (3) 两个次态为状态对封闭链中的一个状态对
  - (4) 两个次态的某一后续状态对可以合并

状态合并的  
必要条件



# 隐含表法化简状态表

## 隐含表(蕴含)法

### 等价状态的判定条件

状态表中的任意两个状态  $S_i$  和  $S_j$  同时满足下列两个条件，它们可以合并为一个状态

1. 在所有不同的现输入下，**现输出**分别相同

状态合并的  
必要条件

2. 在所有不同的现输入下，**次态**分别为下列情况之一

(1) 两个次态完全相同

(2) 两个次态为其现态本身或交错

(3) 两个次态为状态对封闭链中的一个状态对

(4) 两个次态的某一后续状态对可以合并

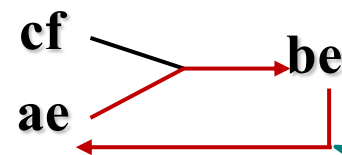
① 建立隐含表

② 比较

③ 追踪

b	cf✓					
c	X	X				
d	X	X	X			
e	be✓	ae✓	X	X		
f	X	X	✓	X	X	
g	X	X	X	<del>cf</del>	X	X
	a	b	c	d	e	f

竖列横排  
掐头去尾



状态对  
封闭连

等价状态对

{a,b}、{a,e}

{b,e}、{c,f}

例1：化简如下状态表

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

# 隐含表法化简状态表

## ④ 获得最大等价状态类

等价状态类的定义——

If:  $S_i \equiv S_j, S_j \equiv S_m$

Then:  $S_i \equiv S_j \equiv S_m$ , 即  $\{S_i, S_j, S_m\}$

最大等价状态类——

某一等价状态类不属于其他任何等价状态类

等价状态对:

$\{a, b\}$ 、 $\{a, e\}$

$\{b, e\}$ 、 $\{c, f\}$

最大等价状态类:

$\{a, b, e\}$ 、 $\{c, f\}$

Let  $\begin{cases} q_1 = \{a, b, e\} \\ q_2 = \{c, f\} \\ q_3 = d \\ q_4 = g \end{cases}$

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

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
$q_1$	$q_2 / 0$	$q_1 / 1$
$q_1$	$q_2 / 0$	$q_1 / 1$
$q_2$	$q_3 / 0$	$q_4 / 0$
$q_3$	$q_3 / 1$	$q_1 / 0$
$q_1$	$q_2 / 0$	$q_1 / 1$
$q_2$	$q_3 / 0$	$q_4 / 0$
$q_4$	$q_2 / 1$	$q_3 / 0$

化简后的状态表

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
$q_1$	$q_2 / 0$	$q_1 / 1$
$q_2$	$q_3 / 0$	$q_4 / 0$
$q_3$	$q_3 / 1$	$q_1 / 0$
$q_4$	$q_2 / 1$	$q_3 / 0$

最小覆盖集:  $\{q_1, q_2, q_3, q_4\}$

# 状态编码（分配）

## 利用触发器设计时序逻辑的方法

- (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) 检查无关状态

# 状态编码（分配）

## 状态分配

### 规则

一种  
经验法

- 1.同一输入下，相同的次态所对应的**现态**应该给予相邻编码
- 2.同一现态在不同输入下所对应的**次态**应给予相邻编码
- 3.给定输入下，输出完全相同，**现态**编码应相邻

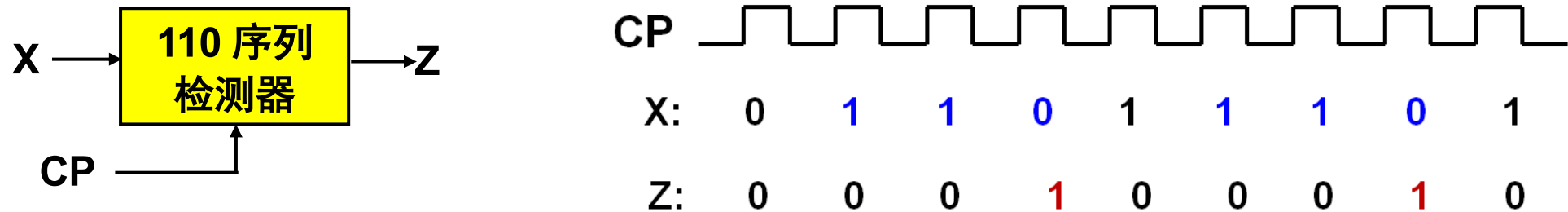
目的：尽量使卡诺图中更多的“1”（或“0”）相邻

注意：

- 初始状态一般可以放在卡诺图的 0号单元格里
- 优先满足规则1和规则2
- 状态编码尽量按照相邻原则给予
- 对于多输出函数，规则3可以适当调高优先级

# 完整电路设计过程示例

例：利用JK触发器设计110序列检测器



## 1. 获得原始状态图和原始状态表

### (1) 状态设定

$S_0$ ——初始状态，表示收到1位数据：“0”

$S_1$ ——表示收到1位数据：“1”

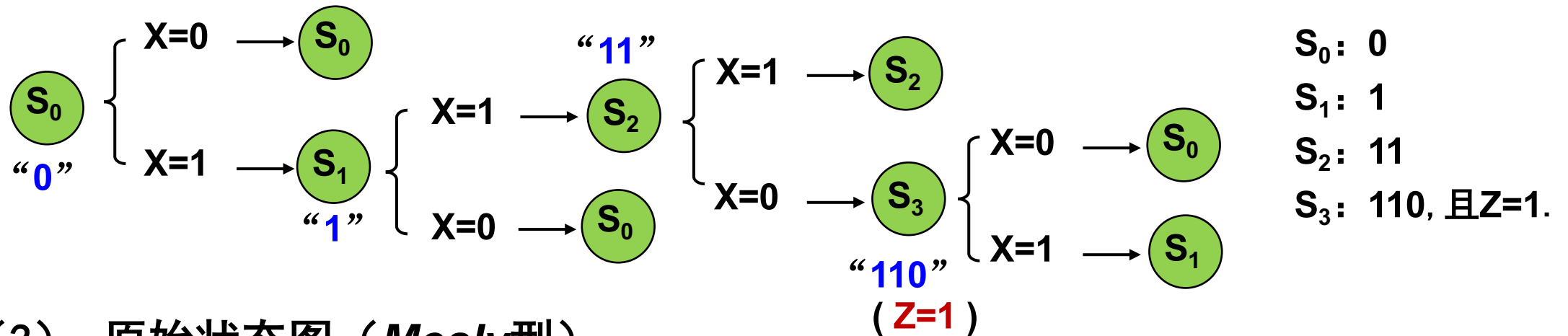
$S_2$ ——表示收到2位数据：“11”

$S_3$ ——表示收到3位数据：“110”，此时输出标志  $Z=1$ .

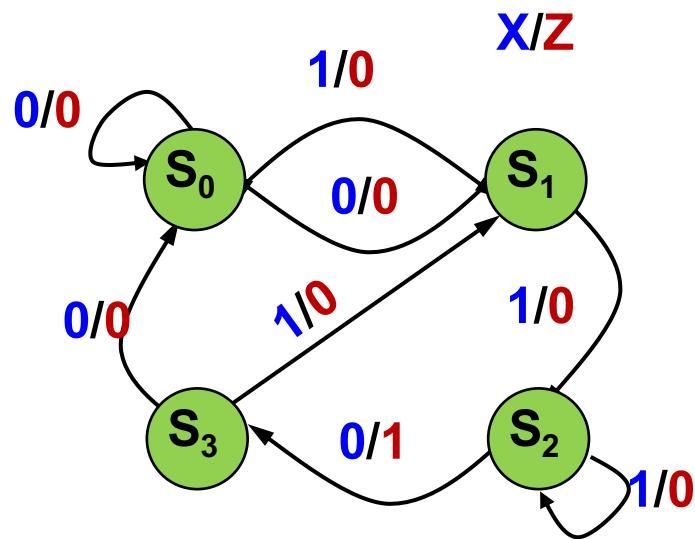
只标记感兴趣的子串

# 利用JK触发器设计110序列检测器

## (2) 分析状态转换情况



## (3) 原始状态图 (Mealy型)



## (4) 原始状态表

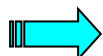
现态 $Q^n$	$Q^{n+1} / Z$	
	$X=0$	$X=1$
$S_0$	$S_0 / 0$	$S_1 / 0$
$S_1$	$S_0 / 0$	$S_2 / 0$
$S_2$	$S_3 / 1$	$S_2 / 0$
$S_3$	$S_0 / 0$	$S_1 / 0$

# 利用JK触发器设计110序列检测器

$J_2 K_2$  : 看  $Q_2^n \rightarrow Q_2^{n+1}$   
 $J_1 K_1$  : 看  $Q_1^n \rightarrow Q_1^{n+1}$

## 2. 状态化简

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
$S_0$	$S_0/0$	$S_1/0$
$S_1$	$S_0/0$	$S_2/0$
$S_2$	$S_3/1$	$S_2/0$
$S_3$	$S_0/0$	$S_1/0$



现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
$S_0$	$S_0/0$	$S_1/0$
$S_1$	$S_0/0$	$S_2/0$
$S_2$	$S_0/1$	$S_2/0$

## 3. 状态分配

使用2个JK触发器

$Y_2 Y_1$

$S_0$  — 00

$S_1$  — 10

$S_2$  — 11

JK触发器驱动表

$Q_n$	$\rightarrow$	$Q_{n+1}$	J	K
0	$\rightarrow$	0	0	X
0	$\rightarrow$	1	1	X
1	$\rightarrow$	0	X	1
1	$\rightarrow$	1	X	0

## 4. 状态转换真值表

输入 现态			次态		触发器				输出
X	$Y_2^n$	$Y_1^n$	$Y_2^{n+1}$	$Y_1^{n+1}$	$J_2$	$K_2$	$J_1$	$k_1$	Z
0	0	0	0	0	0	X	0	X	0
0	1	0	0	0	X	1	0	X	0
0	1	1	0	0	X	1	X	1	1
1	0	0	1	0	1	X	0	X	0
1	1	0	1	1	X	0	1	X	0
1	1	1	1	1	X	0	X	0	0
0	0	1	X	X	X	X	X	X	X
1	0	1	X	X	X	X	X	X	X

规则

1. 同一输入下，相同的次态所对应的**现态**应该给予相邻编码
2. 同一现态在不同输入下所对应的**次态**应给予相邻编码
3. 给定输入下，输出完全相同，**现态**编码应相邻<sup>71</sup>

# 利用JK触发器设计110序列检测器

#### 4. 状态转换真值表

输入 现态			次态		触发器				输出
X	$Y_2^n$	$Y_1^n$	$Y_2^{n+1}$	$Y_1^{n+1}$	$J_2$	$K_2$	$J_1$	$k_1$	Z
0	0	0	0	0	0	X	0	X	0
0	1	0	0	0	X	1	0	X	0
0	1	1	0	0	X	1	X	1	1
1	0	0	1	0	1	X	0	X	0
1	1	0	1	1	X	0	1	X	0
1	1	1	1	1	X	0	X	0	0
0	0	1	X	X	X	X	X	X	X
1	0	1	X	X	X	X	X	X	X

## 5. 卡诺图化简

		$Y_2^n Y_1^n$			
$X$		00	01	11	10
0		0	X	X	X
1		1	X	X	X

**$J_2 = X$**

	$Y_2^n Y_1^n$	00	01	11	10
$X$	0	X	X	1	1
	1	X	X	0	0

$$K_2 = \overline{X}$$

$Y_2^n Y_1^n$ X	00	01	11	10
0	0	X	X	0
1	0	X	X	1

$$J_1 = XY_2^n$$

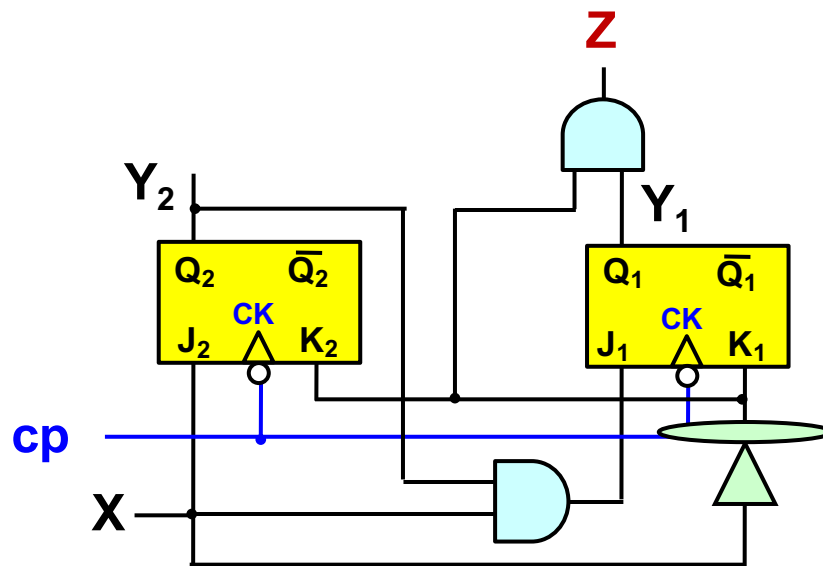
$X \backslash Y_2^n Y_1^n$	00	01	11	10
0	X	X	1	X
1	X	X	0	X

$$K_1 = \overline{X}$$

		$Y_2^n Y_1^n$			
$X$		00	01	11	10
0		0	X	1	0
1		0	X	0	0

$$Z = \overline{X}Y_1^n$$

## 6. 电路实现

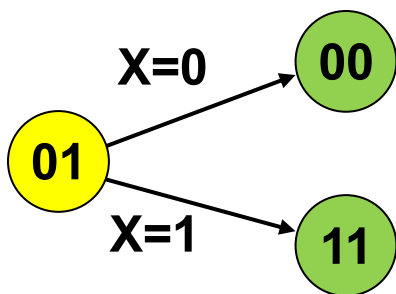




# 利用JK触发器设计110序列检测器

## 7. 检查无关项

$$\left\{ \begin{array}{l} J_1 = XY_2^n \\ K_1 = \bar{X} \\ J_2 = X \\ K_2 = \bar{X} \end{array} \right. \Rightarrow \left\{ \begin{array}{l} Y_1^{n+1} = XY_2^n\bar{Y}_1^n + XY_1^n \\ \quad = X(Y_1^n + Y_2^n) \\ Y_2^{n+1} = X\bar{Y}_2^n + XY_2^n \\ \quad = X \end{array} \right.$$



电路可以自启动

# 用触发器设计同步时序逻辑—实例

---

- 模8可逆计数器
- 自动售卖机
- 时序锁
- 二进制串行加法器
- 串行输入的8421BCD码检测器
- 奇偶校验器
- 更复杂的同步时序逻辑设计

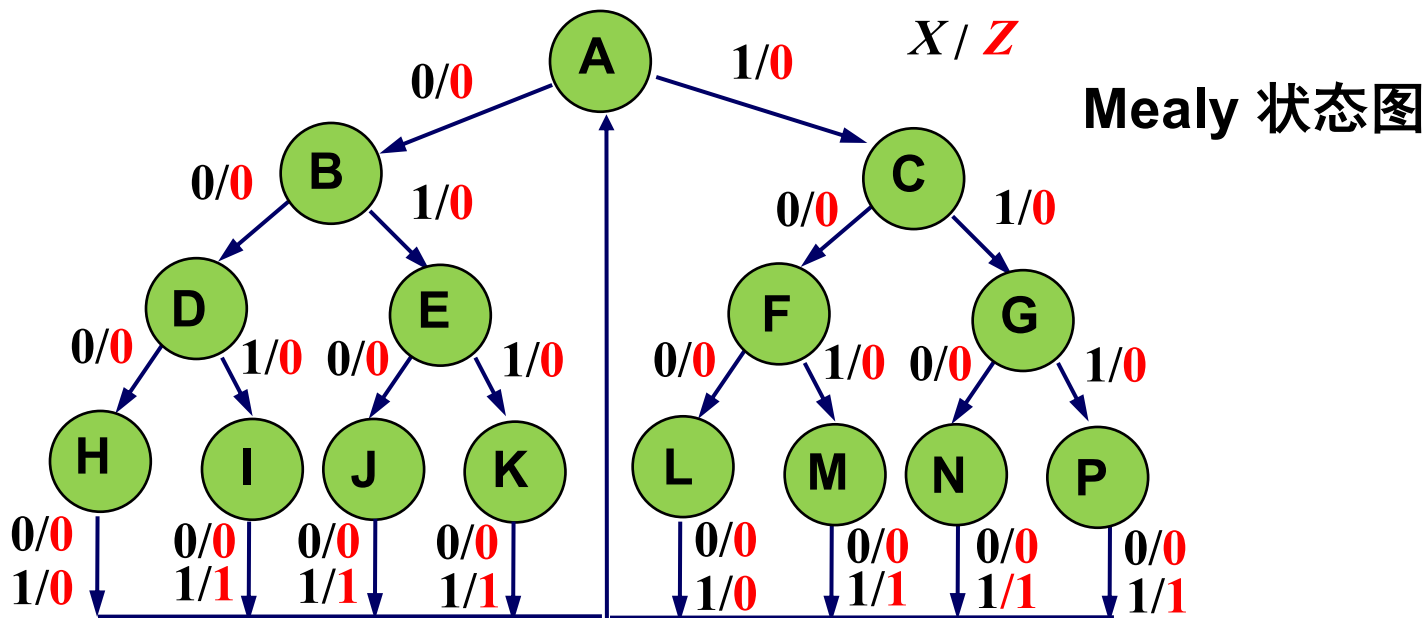
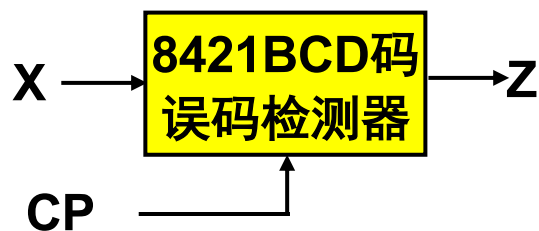
# 利用触发器设计同步时序逻辑\_例5

例：用D触发器设计一个串行输入的8421BCD码误码检测器

要求：

- 8421BCD码低位在前、高位在后串行地加到检测器的输入端。
- 电路每接收一组代码，即在收到第4位代码时判断。若是错误代码，则输出为1，否则输出为0，电路又回到初始状态并开始接收下一组代码。

## 1. 原始状态图及状态表

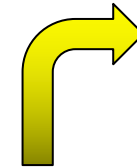


# 利用触发器设计同步时序逻辑\_例5

## 2. 状态化简

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	C / 0
B	D / 0	E / 0
C	F / 0	G / 0
D	H / 0	I / 0
E	J / 0	K / 0
F	L / 0	M / 0
G	N / 0	P / 0
H	A / 0	A / 0
I	A / 0	A / 1
J	A / 0	A / 1
K	A / 0	A / 1
L	A / 0	A / 0
M	A / 0	A / 1
N	A / 0	A / 1
P	A / 0	A / 1

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	C / 0
B	D / 0	E / 0
C	F / 0	G / 0
D	H / 0	I / 0
E	I / 0	I / 0
F	H / 0	I / 0
G	I / 0	I / 0
H	A / 0	A / 0
I	A / 0	A / 1



现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	B / 0
B	D / 0	E / 0
D	H / 0	I / 0
E	I / 0	I / 0
H	A / 0	A / 0
I	A / 0	A / 1

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	C / 0
B	D / 0	E / 0
C	D / 0	E / 0
D	H / 0	I / 0
E	I / 0	I / 0
H	A / 0	A / 0
I	A / 0	A / 1

# 利用触发器设计同步时序逻辑\_例5

## 2. 状态化简

现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	B / 0
B	D / 0	E / 0
D	H / 0	I / 0
E	I / 0	I / 0
H	A / 0	A / 0
I	A / 0	A / 1

## 3. 状态分配

规则1：次态相同，现态编码应相邻


HI, DE 应相邻

规则2：同一现态对应的次态应给予相邻编码

DE, HI 应相邻

规则3：输出相同，现态编码应相邻

ABDEH 应相邻



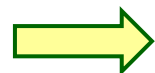
$Q_2^n Q_1^n$		00	01	11	10
$Q_3^n$	0	A	B	D	I
	1			E	H

A: 000;    B: 001  
D: 011;    I: 010  
E: 111;    H: 110

确定 $D_3$ : 看 $Q_3^{n+1}$   
 确定 $D_2$ : 看 $Q_2^{n+1}$   
 确定 $D_1$ : 看 $Q_1^{n+1}$

## 4. 状态转换真值表

$Q_2^n Q_1^n$		$Q_3^n$			
		00	01	11	10
0		A	B	D	I
1				E	H



现态 $Q^n$	$Q^{n+1}/Z$	
	$X=0$	$X=1$
A	B / 0	B / 0
B	D / 0	E / 0
D	H / 0	I / 0
E	I / 0	I / 0
H	A / 0	A / 0
I	A / 0	A / 1

输入及现态				次态			输入 输出			
X	$Q_3^n$	$Q_2^n$	$Q_1^n$	$Q_3^{n+1}$	$Q_2^{n+1}$	$Q_1^{n+1}$	$D_3$	$D_2$	$D_1$	Z
0	0	0	0	0	0	1	0	0	1	0
0	0	0	1	0	1	1	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	1	1	0	0
0	1	0	0	X	X	X	X	X	X	X
0	1	0	1	X	X	X	X	X	X	X
0	1	1	0	0	0	0	0	0	0	0
0	1	1	1	0	1	0	0	1	0	0
1	0	0	0	0	0	1	0	0	1	0
1	0	0	1	1	1	1	1	1	1	0
1	0	1	0	0	0	0	0	0	0	1
1	0	1	1	0	1	0	0	1	0	0
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	0	1	0	0	1	0	0

# 利用触发器设计同步时序逻辑\_例5

## 5. 卡诺图化简

$XQ_3^n$ \ $Q_2^n Q_1^n$	00	01	11	10
00	0	0	1	0
01	X	X	0	0
11	X	X	0	0
10	0	1	0	0

$$D_3 = \overline{Q_3^n Q_2^n Q_1^n} \overline{X} + X \overline{Q_2^n Q_1^n}$$

$XQ_3^n$ \ $Q_2^n Q_1^n$	00	01	11	10
00	0	1	1	0
01	X	X	1	0
11	x	x	1	0
10	0	1	1	0

$$D_2 = Q_1^n$$

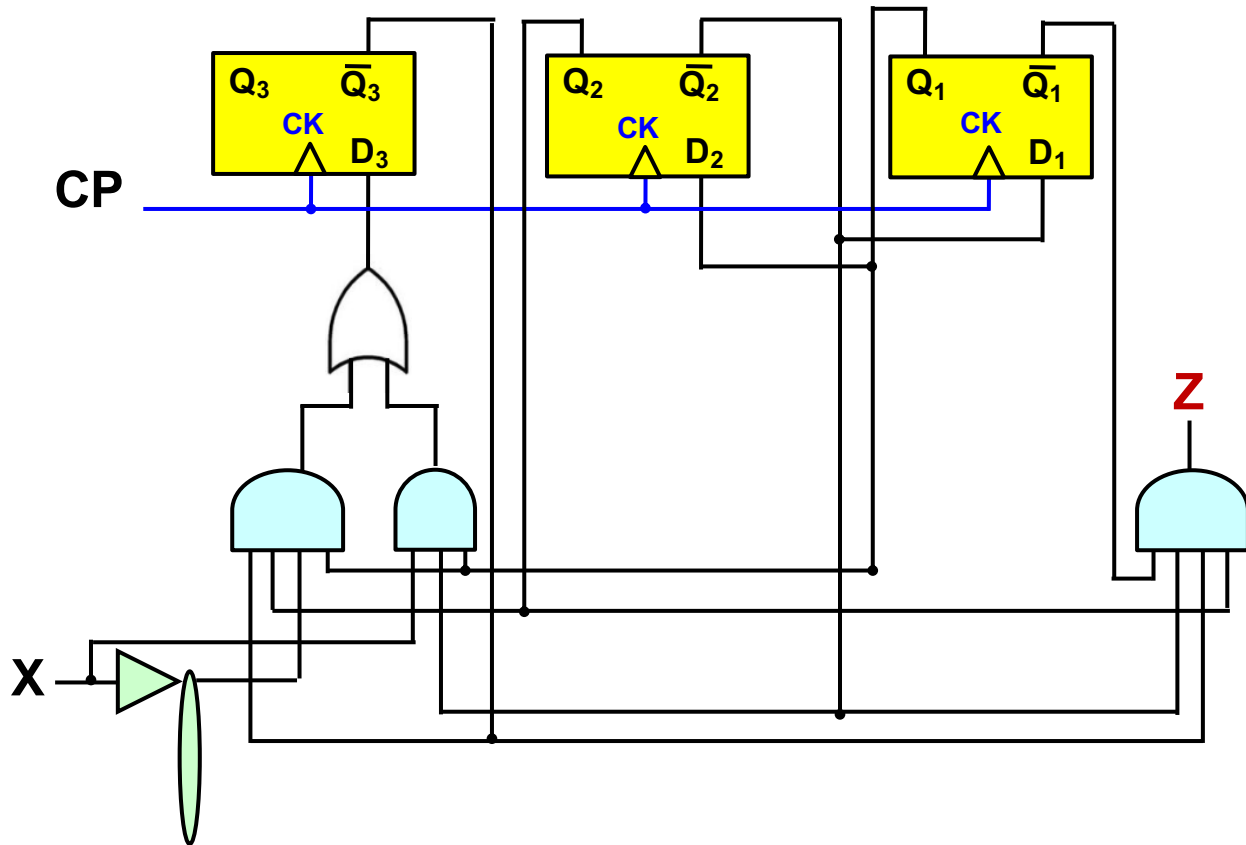
	$Q_2^n Q_1^n$			
$XQ_3^n$	00	01	11	10
00	1	1	0	0
01	x	x	0	0
11	x	x	0	0
10	1	1	0	0

$$D_1 = \overline{Q_2^n}$$

$Q_2^n Q_1^n$ $XQ_3^n$	00	01	11	10
00	0	0	0	0
01	x	x	0	0
11	x	x	0	0
10	0	0	0	1

$$Z = X \overline{Q_3^n} \overline{Q_2^n} \overline{Q_1^n}$$

## 6. 电路实现

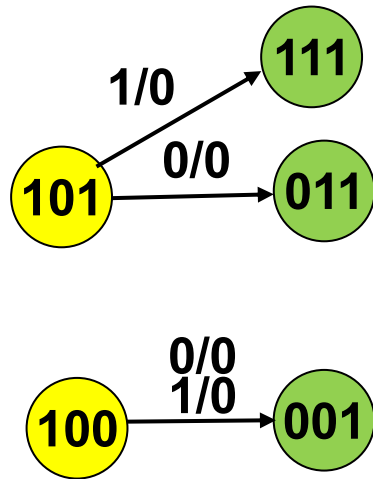


# 利用触发器设计同步时序逻辑\_例5

## 7. 无关项检查

$Q_2^n Q_1^n$					
		00	01	11	10
$Q_3^n$	0	A	B	D	I
	1			E	H

将无关状态 $Q_3^n Q_2^n Q_1^n=101$ 和 $100$ 分别代入次态方程和输出方程计算



电路可以自启动



$$\left\{ \begin{array}{l} Q_i^{n+1} = D_i \\ D_3 = \overline{Q_3^n} Q_2^n Q_1^n \overline{X} + X \overline{Q_2^n} Q_1^n \\ D_2 = Q_1^n \\ D_1 = \overline{Q_2^n} \\ Z = X \overline{Q_3^n} Q_2^n \overline{Q_1^n} \end{array} \right.$$



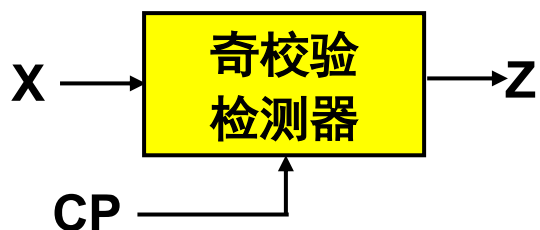
# 利用触发器设计时序逻辑——实例

---

- 模8可逆计数器
- 自动售卖机
- 时序锁
- 二进制串行加法器
- 串行输入的8421BCD码检测器
- 奇偶校验器
- 更复杂的同步时序逻辑设计

# 利用触发器设计同步时序逻辑\_例6

例：利用T触发器设计一个串行输入的奇校验检测器



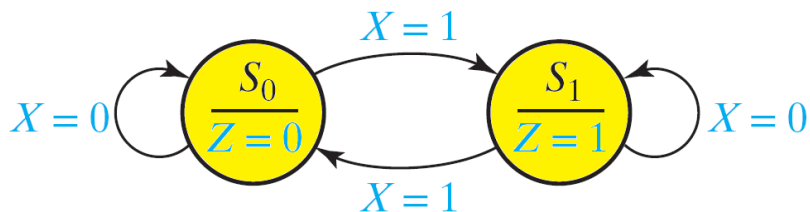
## 1. 原始状态图及状态表

### ① 状态设定

$S_0$ ——表示收到偶数个“1”，初始为0个“1”

$S_1$ ——表示收到奇数个“1”

### ② Moor 状态图



### ③ 状态表

现态	次态 $Q^{n+1}$		输出
$Q^n$	$X=0$	$X=1$	$Z$
$S_0$	$S_0$	$S_1$	0
$S_1$	$S_1$	$S_0$	1

## 2. 状态化简

## 3. 状态分配 $S_0: 0; S_1: 1$

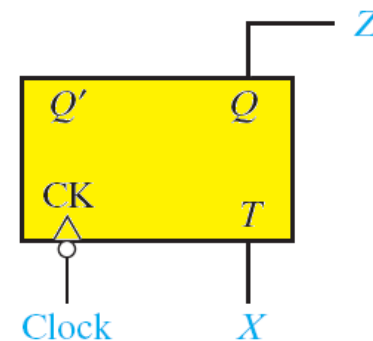
## 4. 状态转换真值表

输入	现态	次态	输入	输出
X	$Q^n$	$Q^{n+1}$	T	Z
0	0	0	0	0
0	1	1	0	1
1	0	1	1	0
1	1	0	1	1

## 5. 卡诺图化简

$$T=X; Z=Q^n$$

## 6. 电路实现



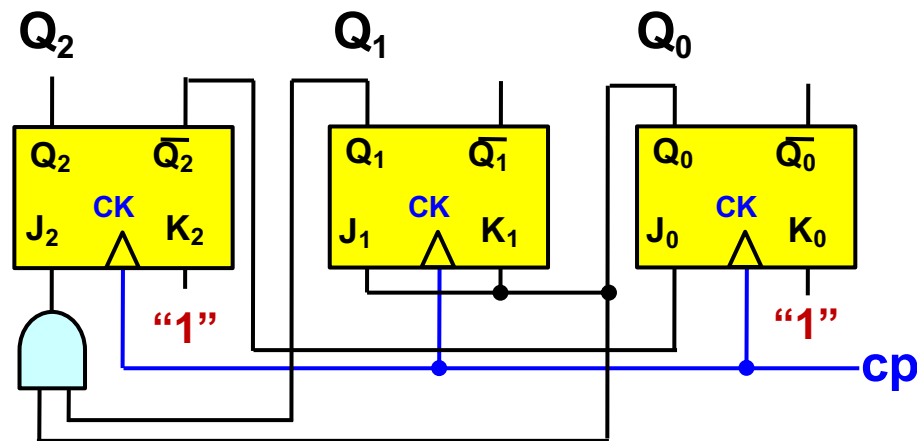
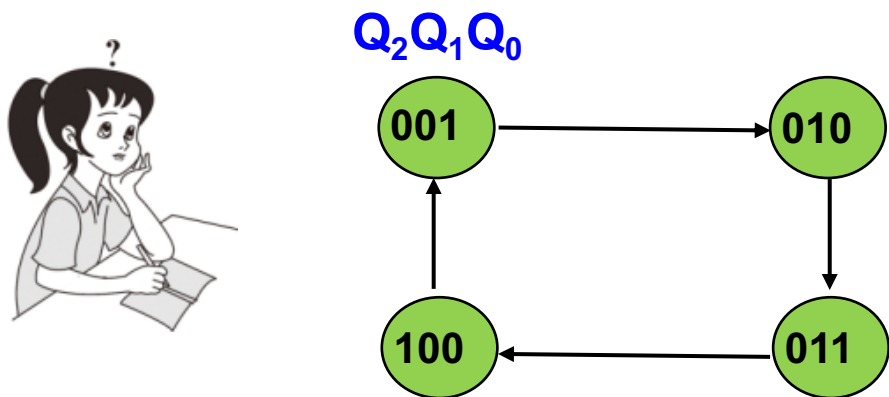
# 用触发器设计同步时序逻辑—实例

---

- 模8可逆计数器
- 自动售卖机
- 时序锁
- 二进制串行加法器
- 串行输入的8421BCD码检测器
- 奇偶校验器
- 更复杂的同步时序逻辑设计

# 更复杂的同步时序设计\_例10

例：某同步时序电路如下所示，按图接线后，试验得到如下的循环状态。经检查：触发器工作正常，试分析故障所在。



## 1. 获得正确状态图

### ① 输入方程

$$J_0 = \overline{Q_2}^n, K_0 = 1$$

$$J_1 = K_1 = Q_0^n$$

$$J_2 = Q_0^n Q_1^n, K_2 = 1$$

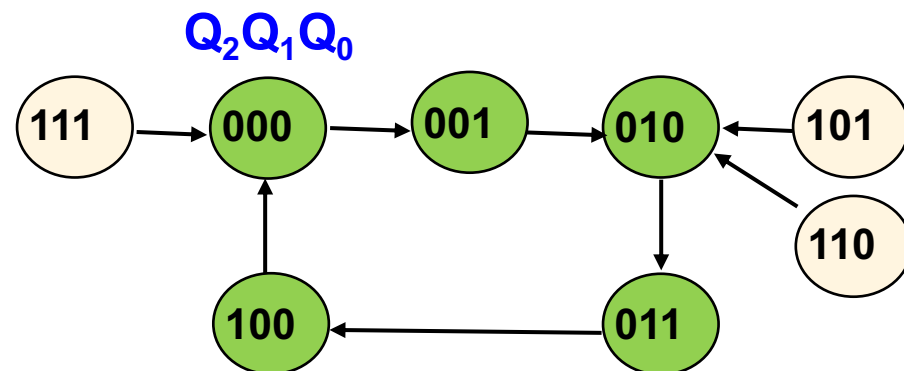
### ② 次态方程

$$Q_0^{n+1} = \overline{Q_0}^n \overline{Q_2}^n$$

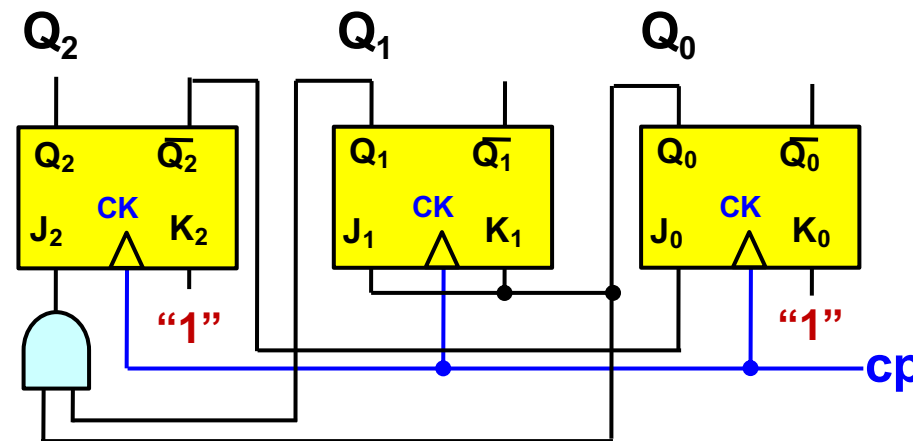
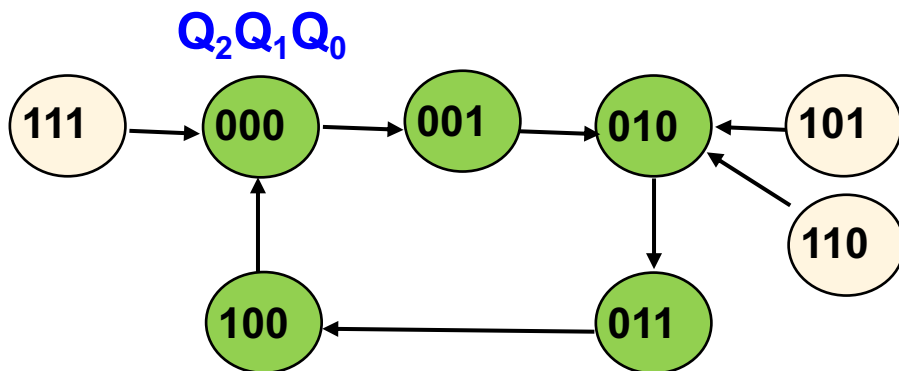
$$Q_1^{n+1} = Q_1^n \oplus Q_0^n$$

$$Q_2^{n+1} = Q_0^n Q_1^n \overline{Q_2}^n$$

### ③ 正确的状态转换图



# 更复杂的同步时序设计\_例10



④ 电路功能：模5加法计数器，可自启动

## 2. 故障分析

① 触发器工作正常：说明——电源和地线接触良好、时钟信号CP正常送入  
故障只可能在进位链或驱动回路中

② 分析各触发器状态：

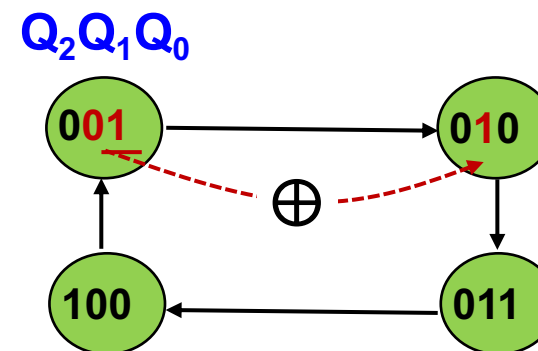
次态方程

$$Q_0^{n+1} = \overline{Q_0}^n \overline{Q_2}^n$$

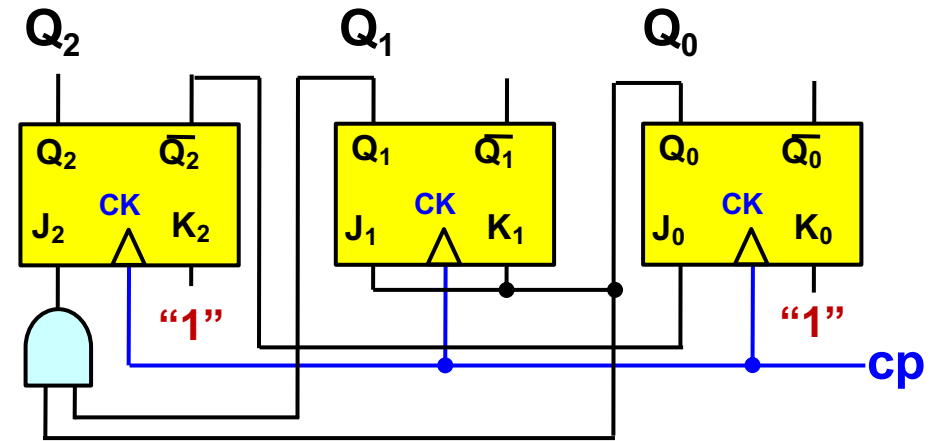
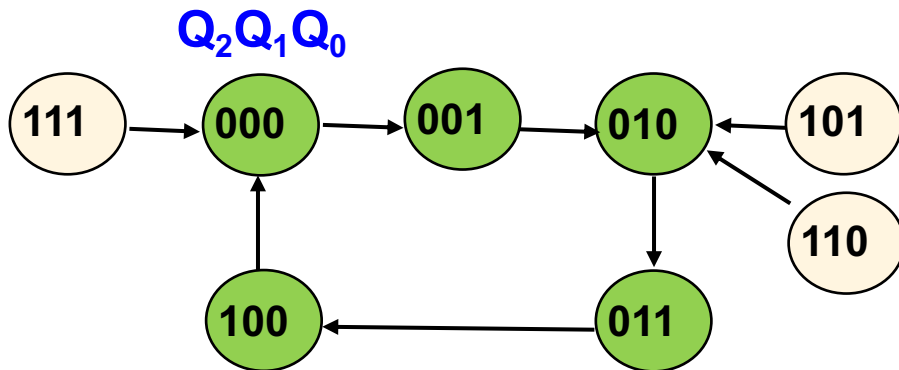
$$Q_1^{n+1} = Q_1^n \oplus Q_0^n$$

$$Q_2^{n+1} = Q_0^n Q_1^n \overline{Q_2}^n$$

触发器FF1  
没有问题

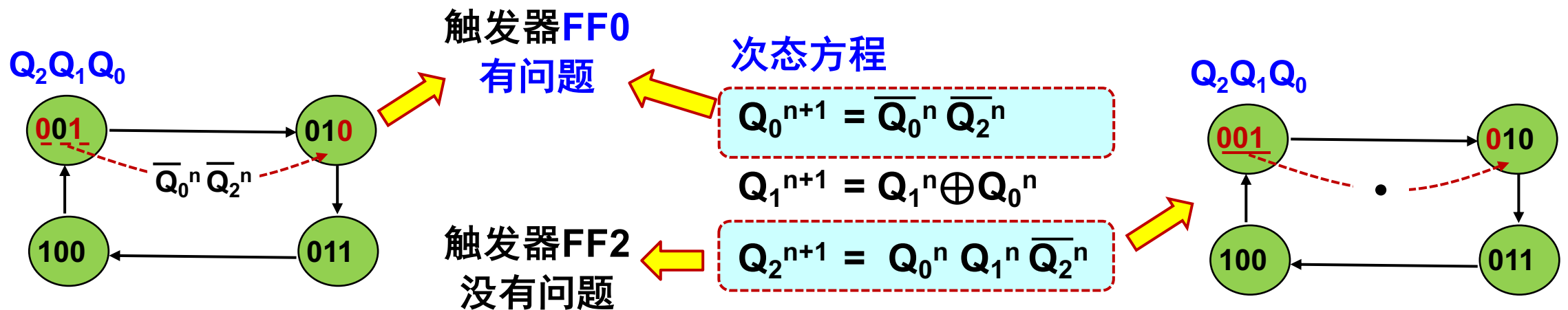


# 更复杂的同步时序设计\_例10

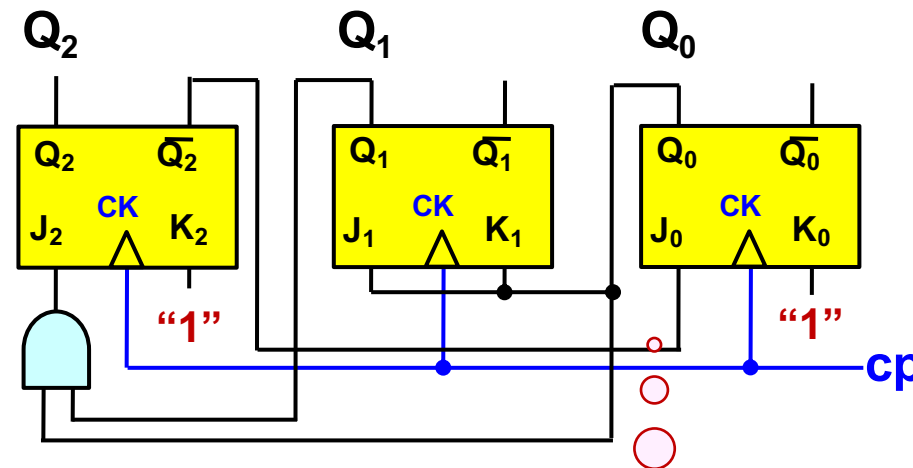
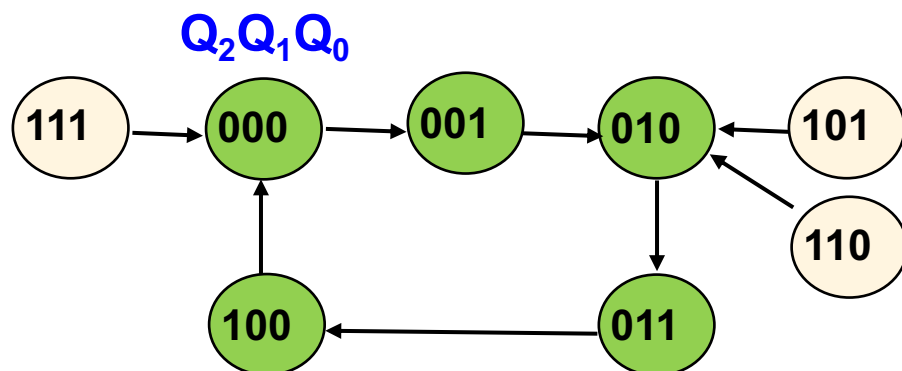


## 2. 故障分析

### ② 分析各触发器状态:



# 更复杂的同步时序设计\_例10



## 2. 故障分析

### ③ 针对触发器0分析:



$K_0$ 接触不良?

$J_0$ 接触不良?

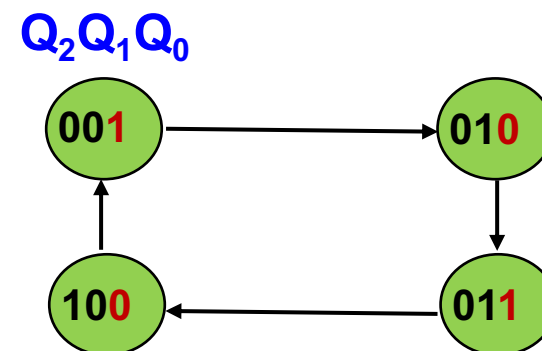
TTL电路管脚悬空等效为高电平1

$\bar{Q}_2$  没有接入,  $J_0$  悬空等效为高电平1

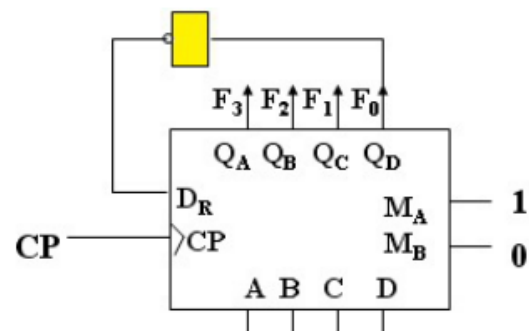
$K_0$ 没问题

触发器变成T', 符合故障现象

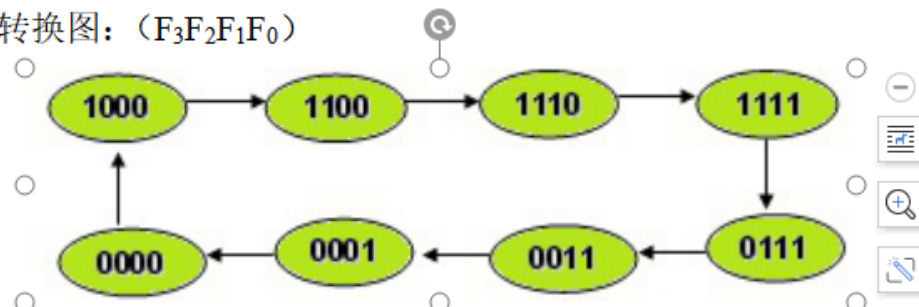
结论:  $\bar{Q}_2$  没有接入,  $J_0$  悬空



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 ( )



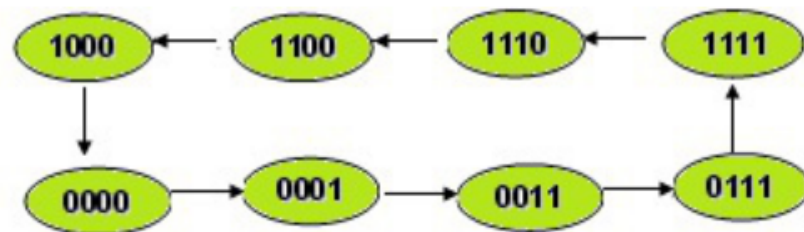
A. 电路的状态转换图：( $F_3F_2F_1F_0$ )



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。

