

数字逻辑设计

Digital Logic Design


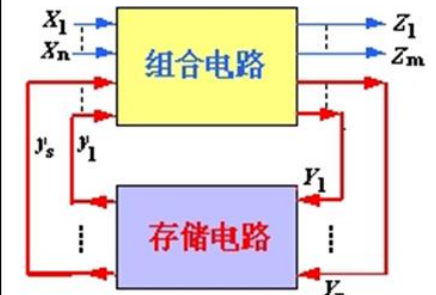
张春慨

计算机科学与技术学院

ckzhang@hit.edu.cn

组合逻辑电路 vs 时序逻辑电路

- 时序逻辑元件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">■ 仅与当前时刻的输入有关 $Z_m = f_m(x_1, \dots, x_n)$	不包含存储电路		$Z_m = f_m(x_1, \dots, x_n)$
	时序逻辑电路	<p>任意时刻的输出与以下有关:</p> <ul style="list-style-type: none">■ 当前时刻的输入■ 电路过去（上一个时刻）的工作状态 $Z_m = f_m(x_1, \dots, x_n, y_1, \dots, y_s)$	包含存储电路		<p>输出方程, 驱动方程, 状态方程:</p> $Z_m = f_m(x_1, \dots, x_n, y_1^n, \dots, y_s^n)$ $Y_r = g_r(x_1, \dots, x_n, y_1, \dots, y_s)$ $Y_s^{n+1} = q_s(x_1, \dots, x_n, Y_1^n, \dots, Y_s^n)$

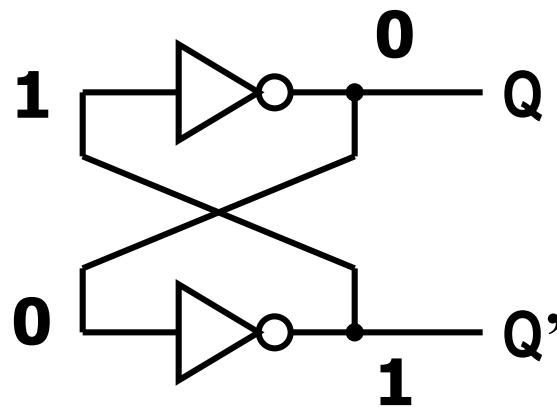
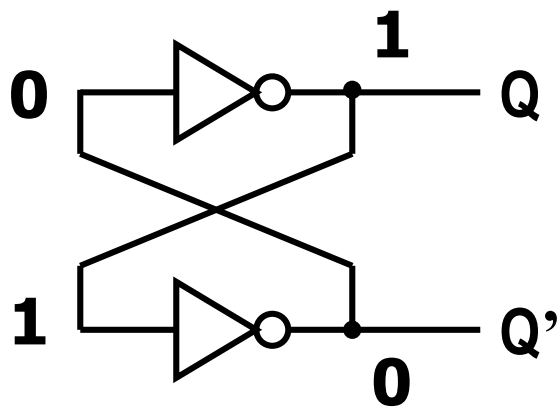
时序逻辑元件

- 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
- 时序逻辑的Verilog描述

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
 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(激励), 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。

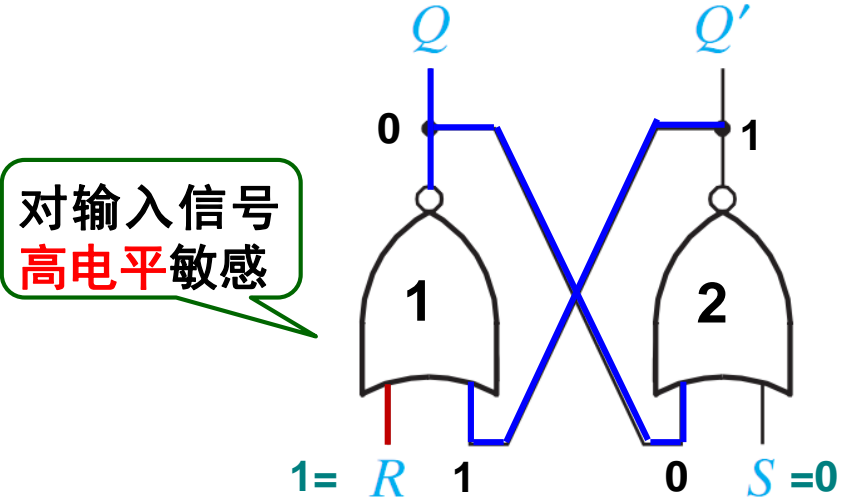
双稳态



- 有反馈，不同于普通组合逻辑电路
- 两个输出互补 (Q , Q')
- 无输入，无实际用途

基本SR锁存器

(1) 电路构成（或非门）



$Q (Q_n)$ ——**现态**
 $Q^+ (Q_{n+1})$ ——**次态**(Next state)
 $Q = 0$ ($\underline{Q} = 1$) : state 0
 $Q = 1$ ($\underline{Q} = 0$) : 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

清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

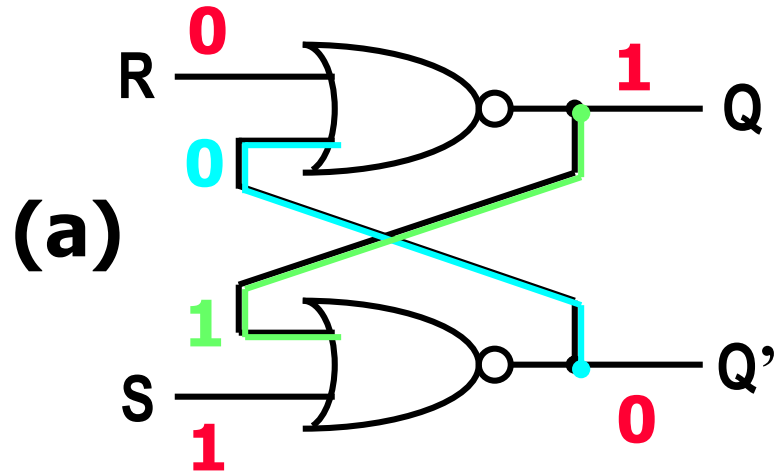
置 0

不允许

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锁存器置1功能分析(S=1, R=0)



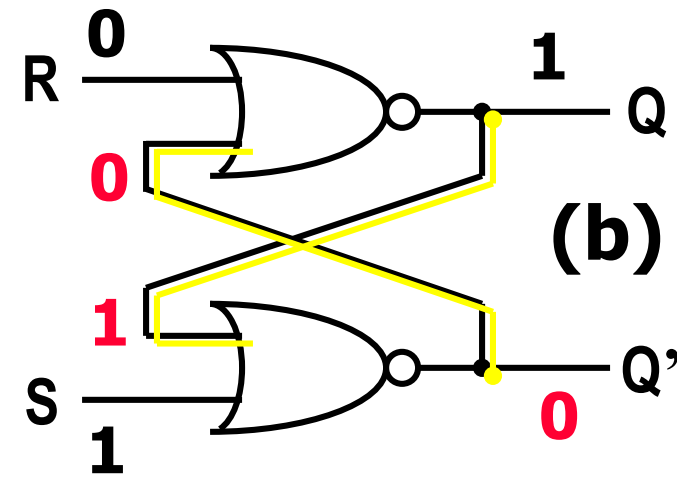
置位: $Q_{n+1}=1$ $Q'_{n+1}=0$

a. $Q_n = 1, Q'_n = 0$

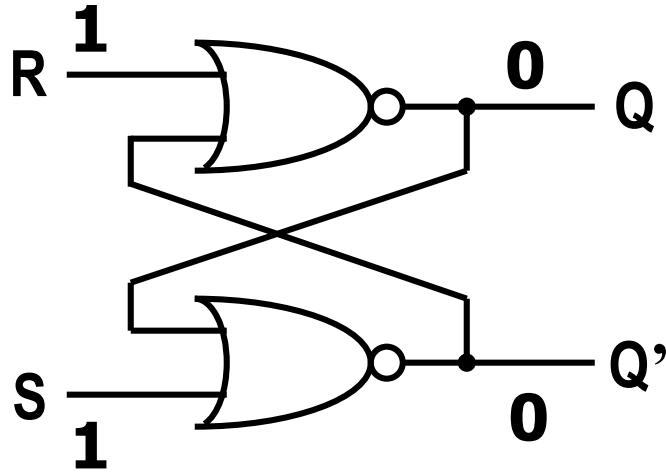
$Q_{n+1}=1, Q'_{n+1}=0$

b. $Q_n = 0, Q'_n = 1$

$Q_{n+1}=1, Q'_{n+1}=0$



基本SR锁存器功能分析(S=1, R=1)



$Q_{n+1} = Q'_{n+1} = 0$
两个输出不互补，不允许

如果输入从S=1, R=1变化为S=0, R=0, (考虑门延迟)
输出如何变化?

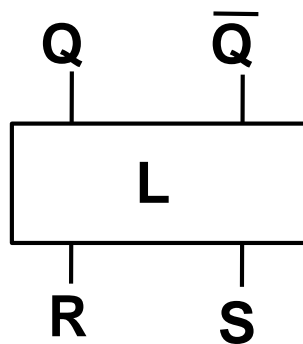
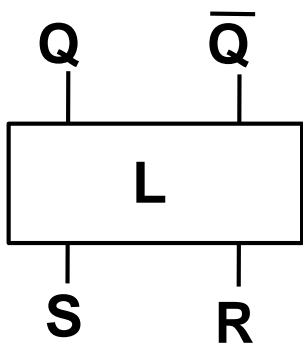
基本SR锁存器次态方程、逻辑符号等

(3) 次态方程 $Q_{n+1} = f(\text{输入}, Q_n)$

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

约束条件: $(SR = 0)$

(4) 逻辑符号



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

功能表

清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

用于时序
电路设计

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

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

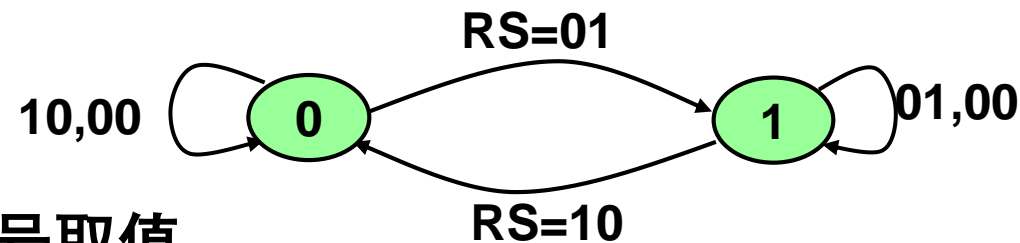
(6) 状态图

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

圆圈：表示电路的状态

有向线段：表示状态的转换关系

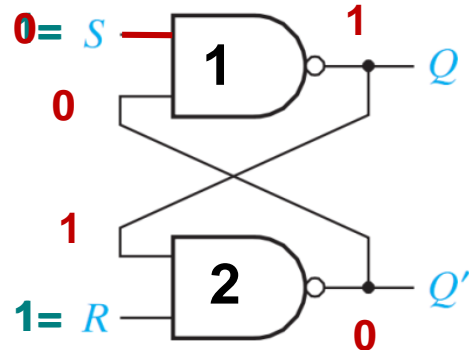
有向线段旁的文字：表示转换条件，即输入信号取值



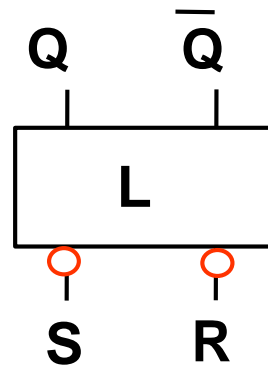
另一种形式的基本SR锁存器（与非门）

写出下面的锁存器的功能表

对输入信号
低电平敏感



逻辑符号:



功能表

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

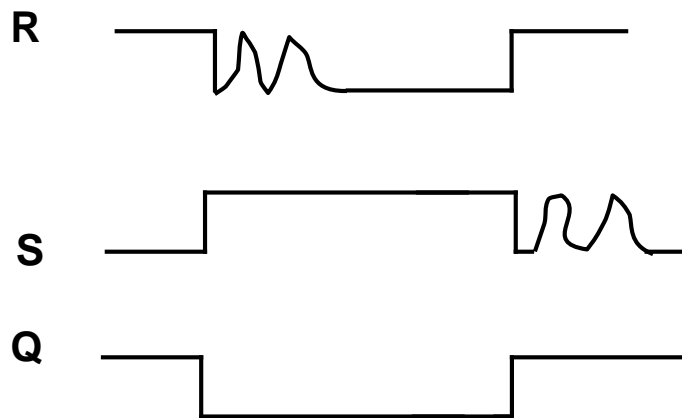
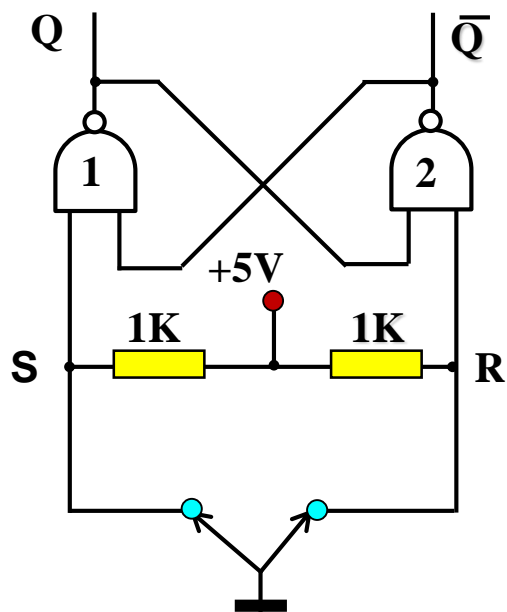
保持

置 1

置 0

× 不允许

锁存器的应用——开关去抖



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

- ❖ 由于机械弹性作用, 机械式开关在使用中, 通常伴随有一定时间的触点机械抖动。
- ❖ 触点抖动可能导致判断出错 (一次按下或释放被错误地认为是多次操作)

(7) 典型芯片

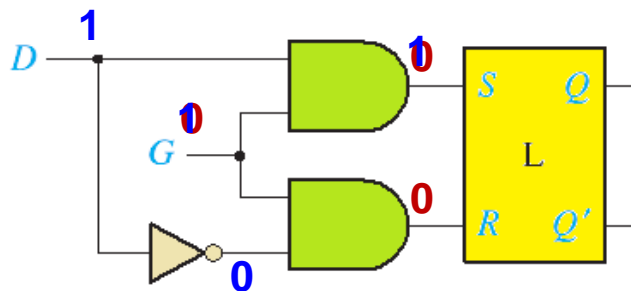
74LS279: 4 R-S latches

基本SR锁存器小结

- 优点：结构简单
- 缺点：
 - 输入存在约束，使用不便；
 - 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

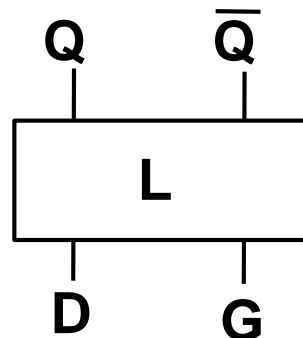
(3) 次态方程

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

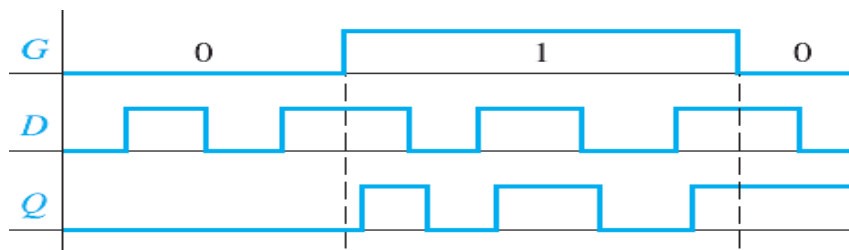
$$Q_{n+1} = GD + \bar{G}Q_n$$

在G为高电平期间, Q端的输出直接复制D端波形

(4) 逻辑符号



(5) 时序分析



(6) 典型芯片

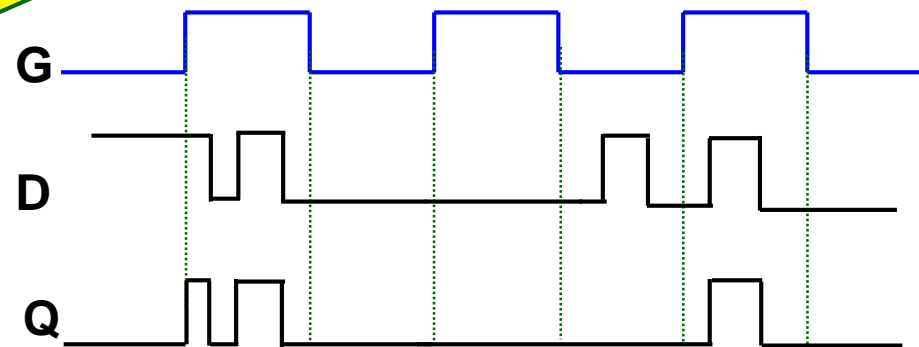
74LS373: 8D锁存器

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
- ◆ 缺点：使能端G有效期间，只要输入信号D改变（有时是干扰信号），Q也跟着改变（“空翻”现象）

锁存器的使能端有效

一个时钟内，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，改变状态

什么样的Verilog描述会生成锁存器

只发生在组合逻辑电路中

1、if...else...语句没有else

```
always @ (*) begin
    if (d_en) q = d;
end
```

2、case语句没有default

锁存器的危害:

- 使静态时序分析变得非常复杂
- 对毛刺敏感，不能异步复位，上电后处于不确定的状态

```
reg q;
always @ (*) begin
    if (d_en) q2 = d;
    else      q2 = 1'h0;
end

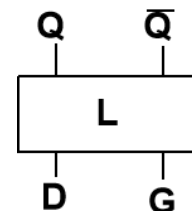
always @ (*) begin
    case (cnt[1:0])
        2'b00 : q = d1;
        2'b01 : q = d2;
        2'b10 : q = d3;
    endcase
end
```


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
- 时序逻辑的Verilog描述

(时钟)触发器

- 受时钟脉冲控制的一种时序逻辑元件，也称时钟触发器。
- 时钟也称同步信号，由时钟脉冲确定状态转换的时刻(何时转换?)
- 由输入信号确定触发器状态转换的方向(如何转换?)
- 将多个触发器的时钟端相连，可以使它们同时动作。



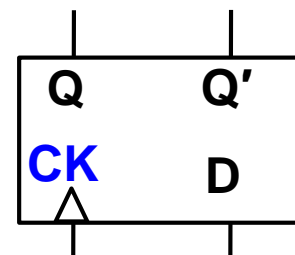
时钟触发器分类

按逻辑功能

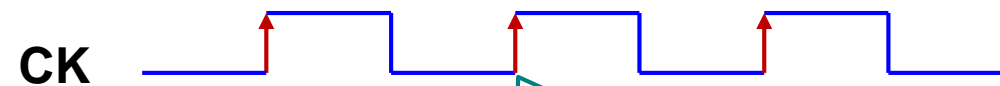
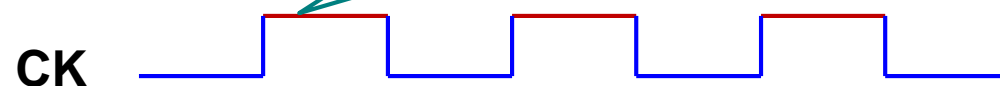
SR触发器
D触发器
JK触发器
T触发器
T'触发器

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，触发器状态翻转（改变）



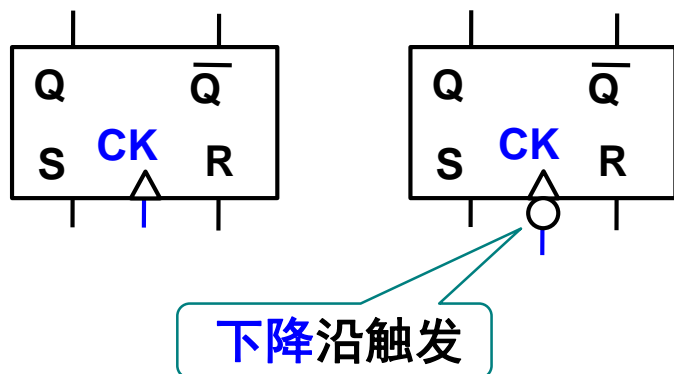
边沿触发方式：时钟上升沿到来时刻，触发器状态翻转

边沿触发器

- SR触发器
- D触发器
- JK触发器
- T触发器
- 带附加输入端的触发器

边沿触发器——SR触发器

(1) 逻辑符号



(3) 次态方程

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

$SR = 0$ (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

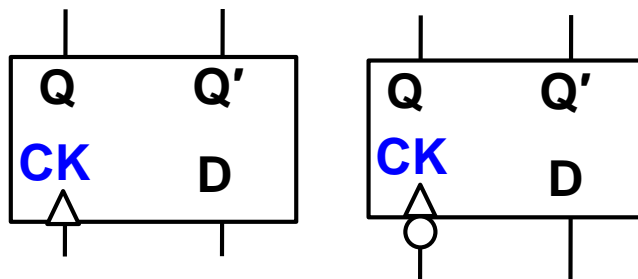
Q_n	→	Q_{n+1}	R	S
0	→	0	X	0
0	→	1	0	1
1	→	0	1	0
1	→	1	0	X

可以从功能表推导出来

SR触发器：输入存在约束

边沿触发器——D触发器

(1) 逻辑符号



(2) 功能表（上升沿为例）

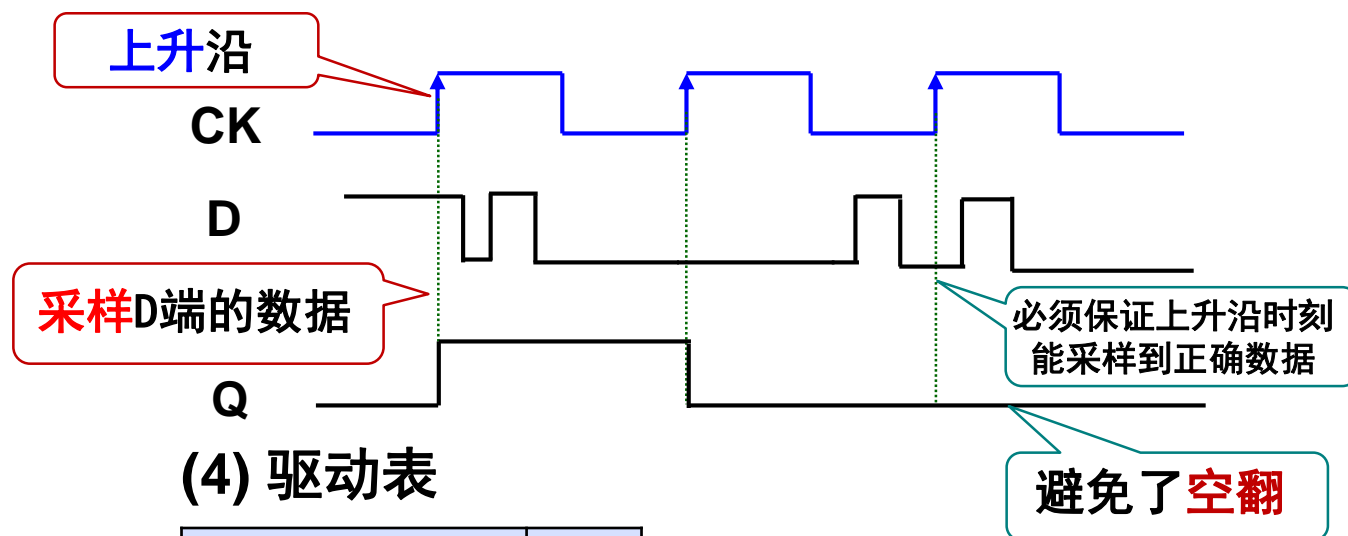
时钟端 CK	输入端 D	现态 Q_n	次态 Q_{n+1}
↑	0	0	0
↑	0	1	0
↑	1	0	1
↑	1	1	1

(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？）



(4) 驱动表

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

D触发器的特点：

输入无约束

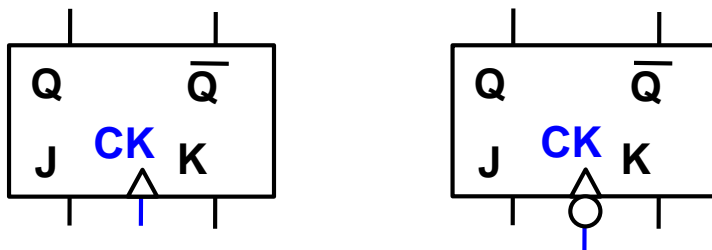
最简单，应用最广

用Verilog实现D触发器

```
module VrDff(input CLK, input D, output reg Q);  
    always @ (posedge CLK)  
        Q <= D;  
endmodule
```

边沿触发器——JK触发器

(1) 逻辑符号



(3) 次态方程 $Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n$

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

(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

保持

置0

置1

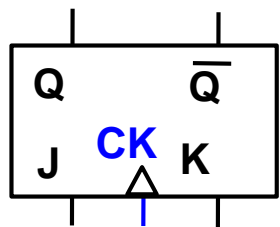
翻转

(4) 驱动表

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

$Q_n \rightarrow Q_{n+1}$	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

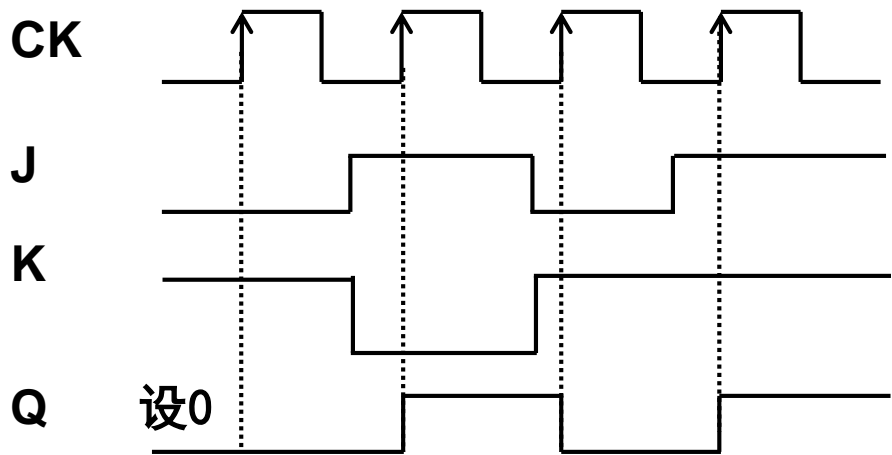


时钟边沿触发器

- ◆ 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

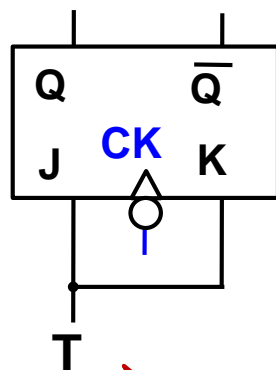
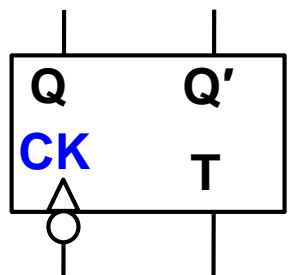
输入端		次态
J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	\bar{Q}_n

设初态 $Q=0$ ，请画出 Q 的波形



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

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

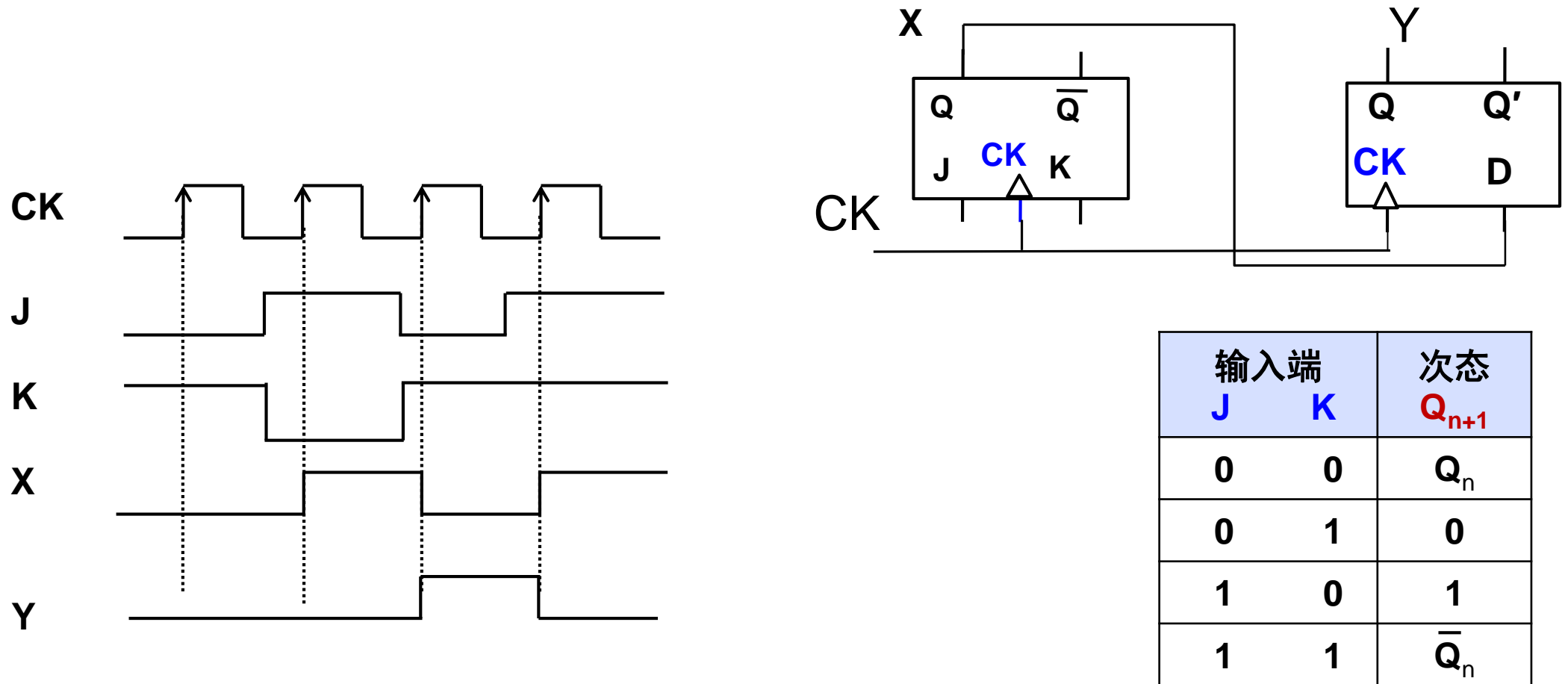
IF $J=K=T$



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

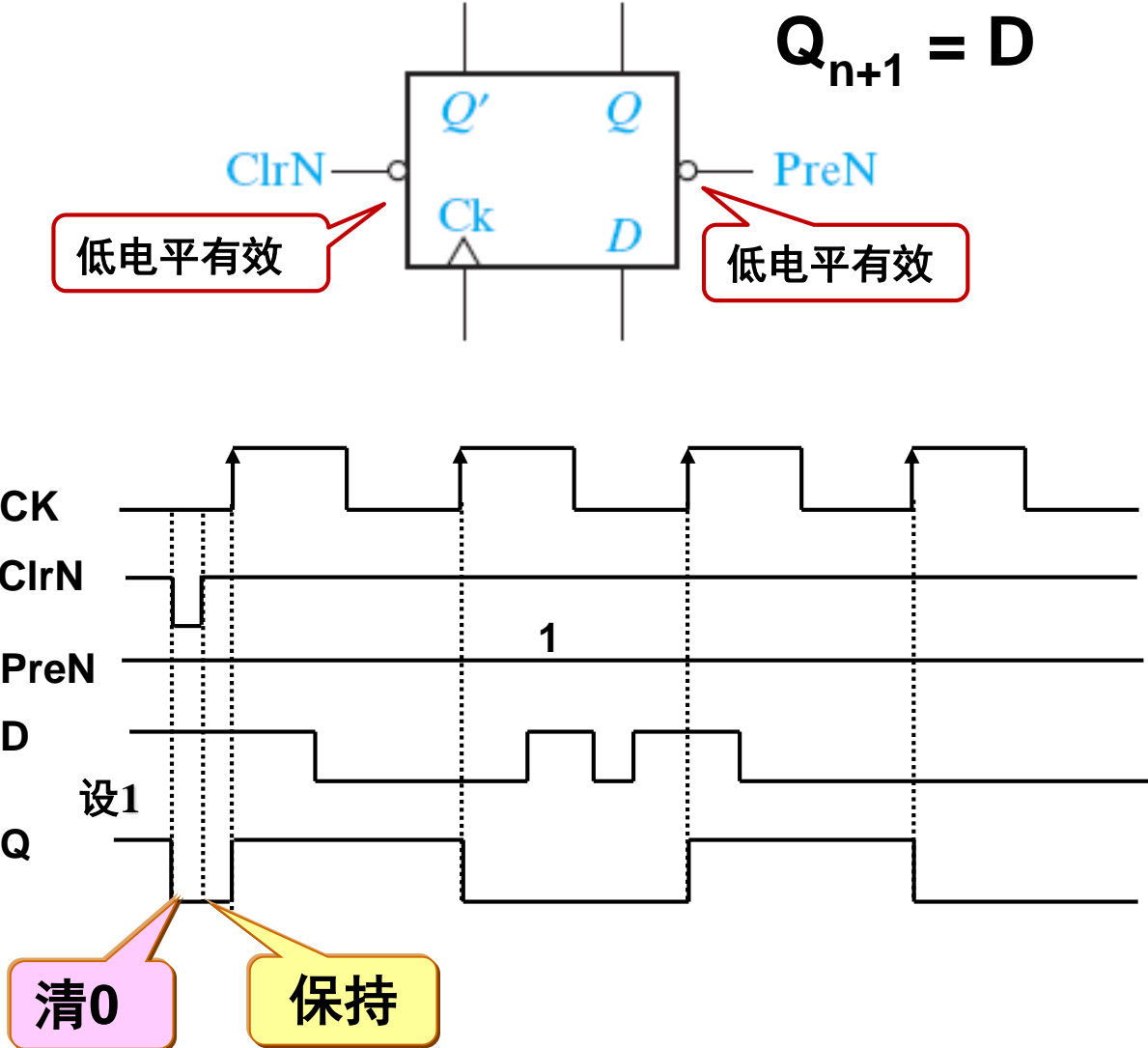
简单时序电路分析

设初态 $X=0$, $Y=0$, 画出 X,Y 的波形



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

用Verilog实现带异步清零端的D触发器

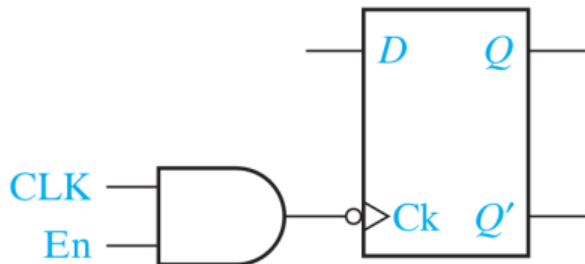
```
module VrDffC(input CLK, input CLRN, input D,  
              output reg Q);  
  
  always @ (posedge CLK or negedge CLRN)  
    if (CLRN==1) Q <= 0;  
    else Q <= D;  
  
endmodule
```

带附加输入端的边沿触发器—续

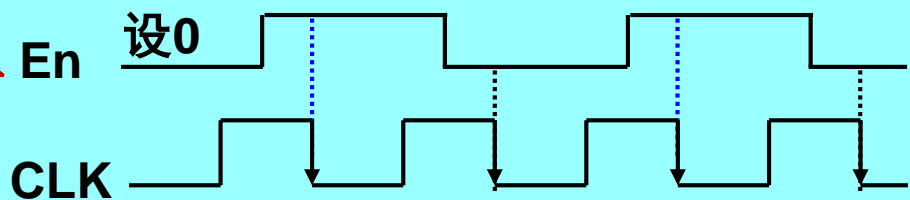
■ 带时钟使能端

解决方案:

使能端不与时钟端CLK捆绑使用

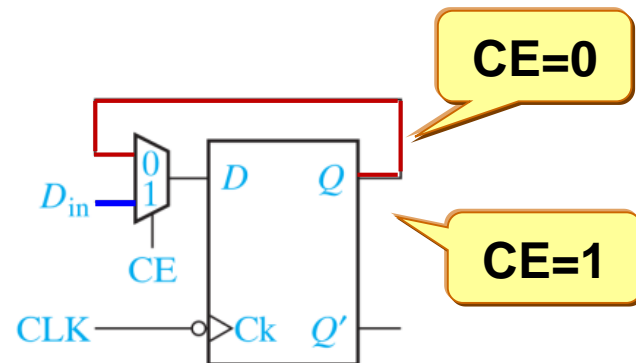
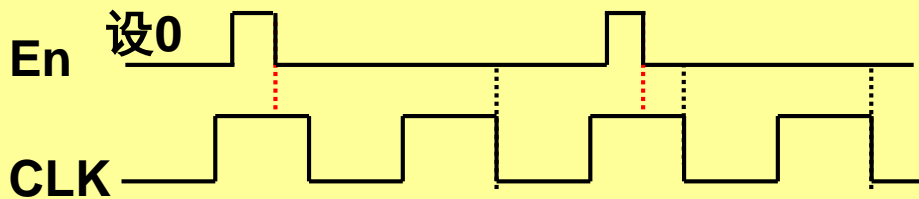


信号给定正确



后果: 失去同步性!

信号给定错误



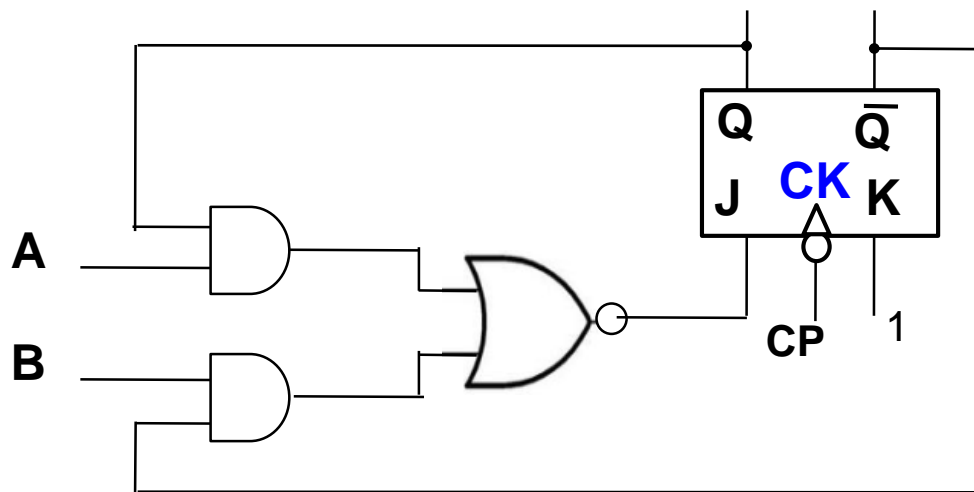
$$Q_{n+1} = Q_n \cdot CE' + D \cdot CE$$

带时钟使能端的D触发器

```
module VrDffSE(input CLK, input CE,  
    input D, output reg Q);  
  
    always @ (posedge CLK)  
        if (CE==1) Q <= D; //时钟使能有效  
        else Q <= 0;  
  
endmodule
```

JK触发器的分析实例1

写出下图所示电路的次态方程

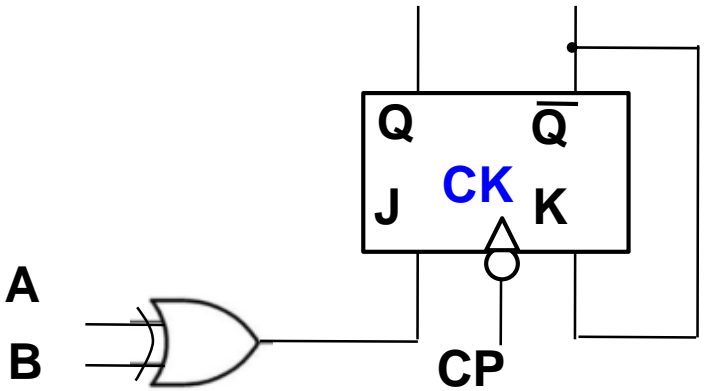


$$\begin{aligned}Q_{n+1} &= J \bar{Q}_n + \bar{K} Q_n \\&= J \bar{Q}_n \\&= \overline{A Q_n + B \bar{Q}_n \bar{Q}_n} \\&= \overline{A Q_n} \cdot \overline{B \bar{Q}_n \bar{Q}_n} \\&= (\bar{A} + \bar{Q}_n) (\bar{B} + Q_n) \bar{Q}_n \\&= \bar{A} \bar{B} \bar{Q}_n + \bar{B} \bar{Q}_n \\&= \bar{B} \bar{Q}_n\end{aligned}$$

JK触发器的分析实例2

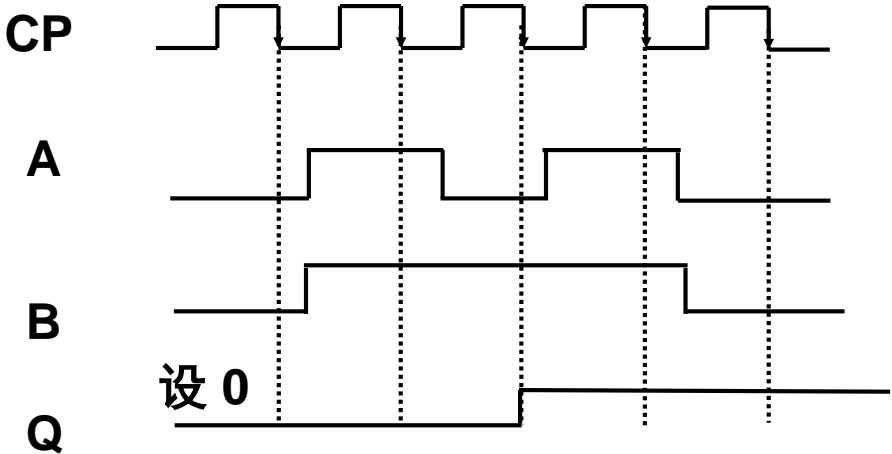
画出Q端波形图

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

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

三人表决器设计

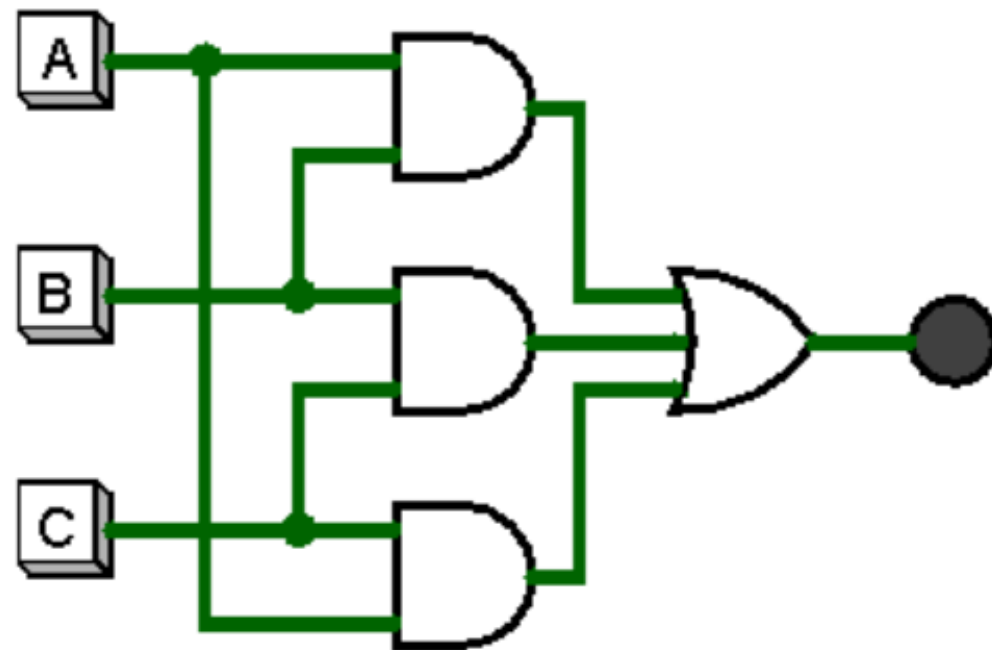
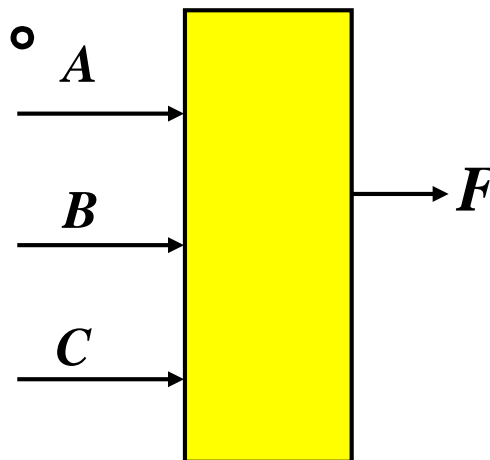
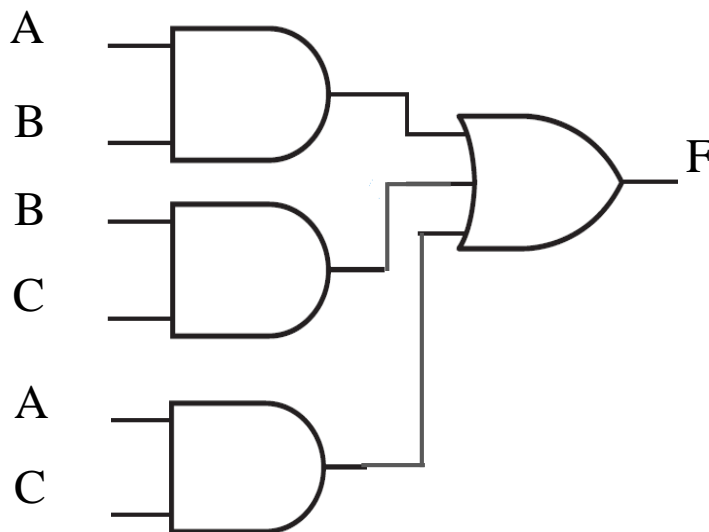
■少数服从多数，结果为多数人的选择。

真值表

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

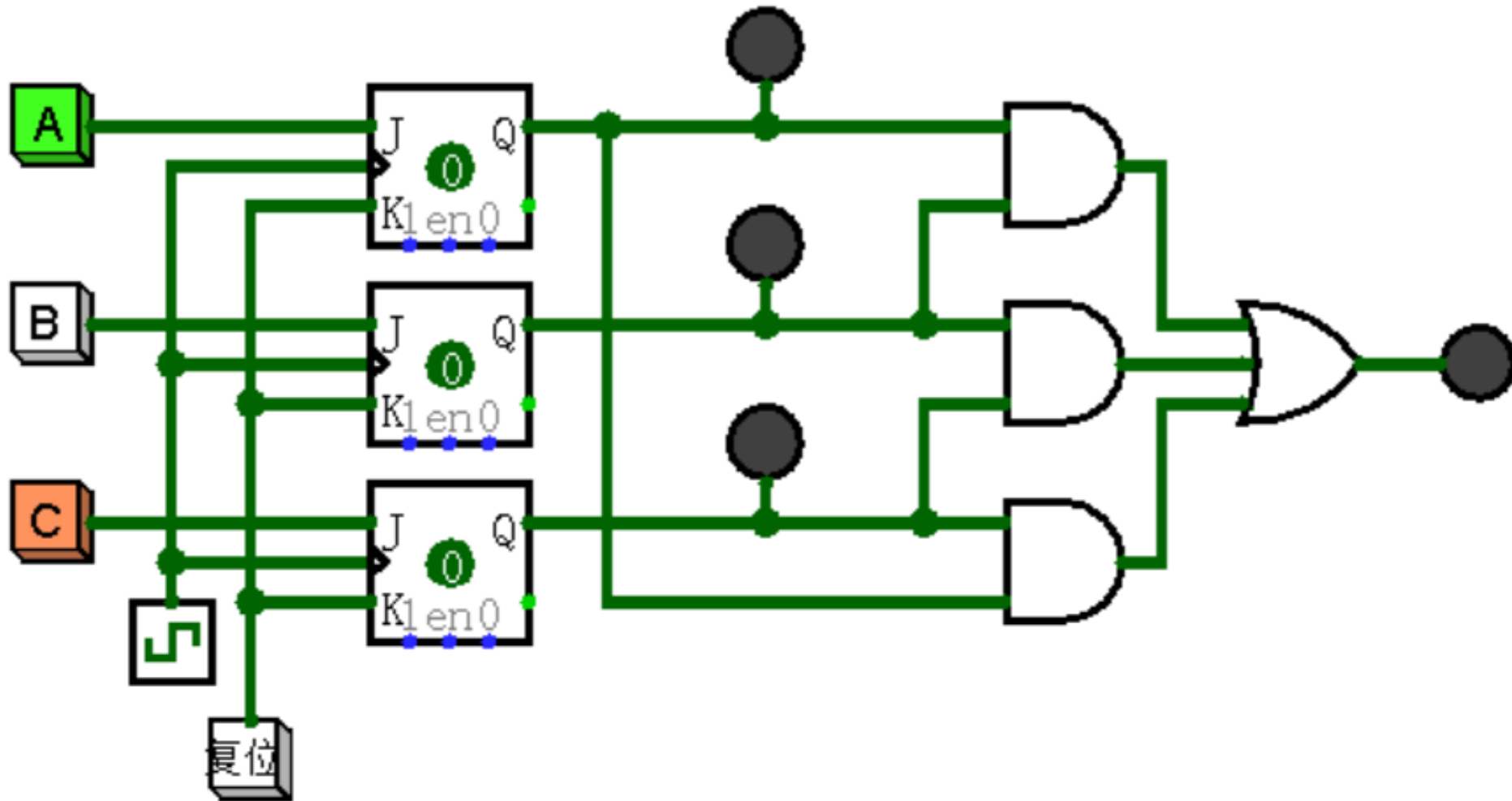
A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$F=AB+AC+BC$$



触发器的应用——存储功能

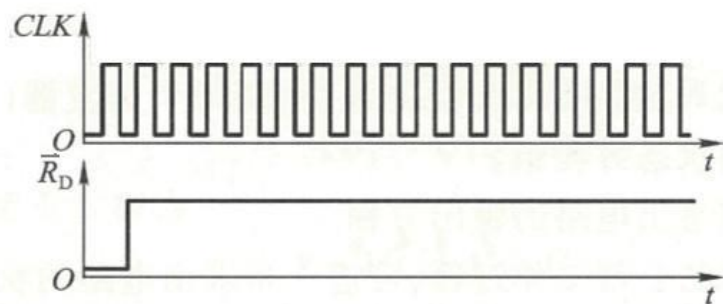
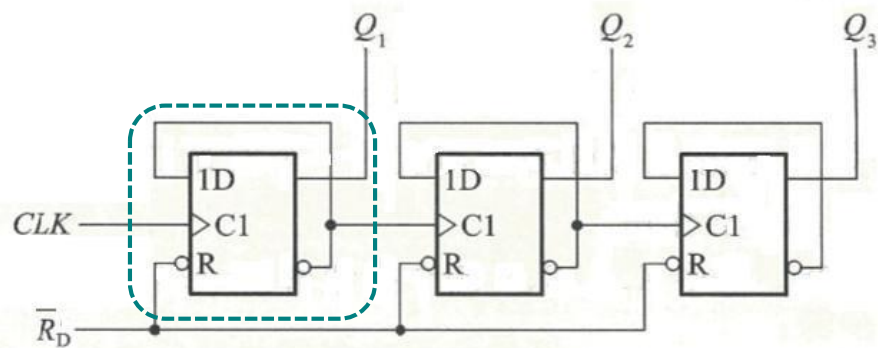
保存瞬态信号，直到清除为止



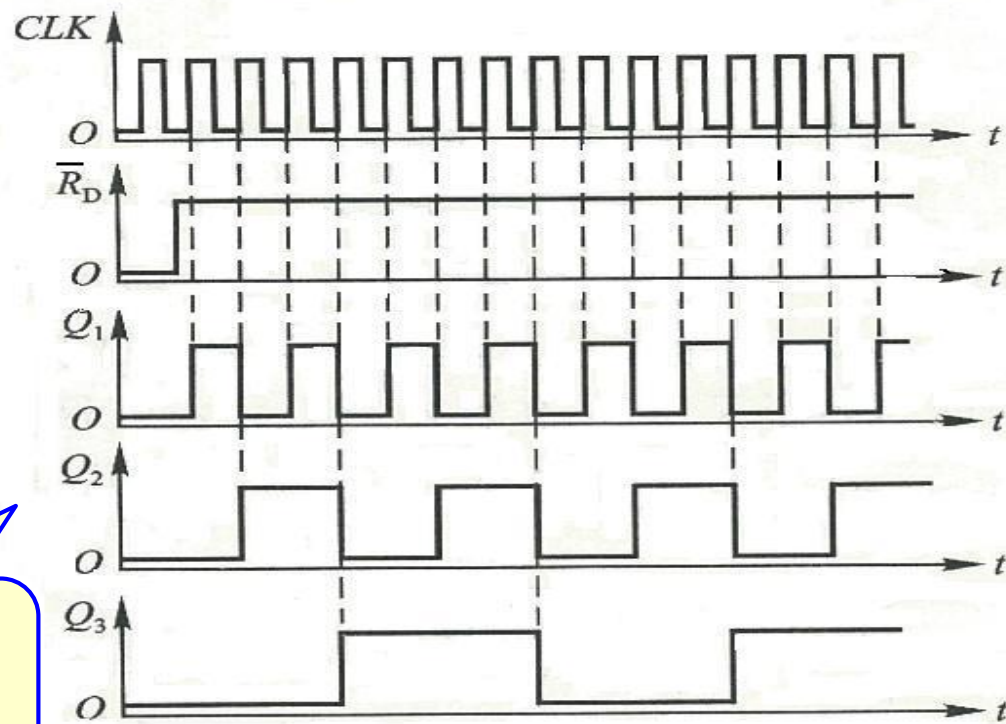
触发器的应用——分频/计数

- 利用触发器的置0、置1功能，由多个触发器组成分频电路，对输入的时钟信号进行分频。

例：分析输出 Q_1 、 Q_2 、 Q_3 与时钟CLK之间的频率关系（R为清零端）



Q_1 对CLK二分频
 Q_2 对CLK四分频
 Q_3 对CLK八分频



Q_1 、 Q_2 、 Q_3 输出波形

边沿触发器

- D触发器
- SR触发器
- JK触发器
- T触发器
- 带附加输入端的触发器

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

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
- 时序逻辑的Verilog描述

触发器类型转换——代数法

- 用得最多的是JK触发器和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）

■代数法

转换方法

- 代数法
- 卡诺图法

从次态方程入手

1. JK → D、T、SR

(1) JK → D

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

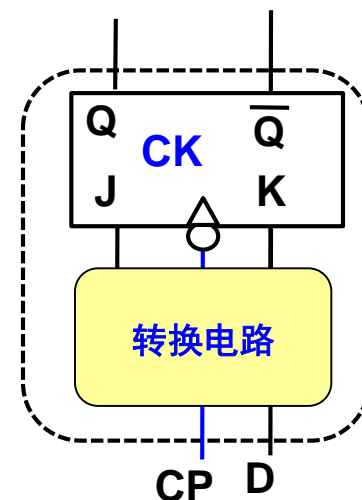
$$\text{D: } Q_{n+1} = D$$

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

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

$$J = f_1(D, Q)$$

$$K = f_2(D, Q)$$



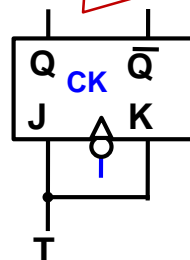
$$\begin{cases} J=D \\ \bar{K}=D \quad (K=\bar{D}) \end{cases}$$

触发器类型转换——JK转其他

(2) JK \rightarrow T

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

JK触发器的特例



(3) JK \rightarrow SR

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

$$\begin{aligned} \text{SR: } Q_{n+1} &= S + \bar{R} Q_n = S(Q_n + \bar{Q}_n) + \bar{R} Q_n \\ &= S Q_n + S \bar{Q}_n + \bar{R} Q_n \\ &= S Q_n (R + \bar{R}) + S \bar{Q}_n + \bar{R} Q_n \\ &= R S Q_n + \bar{R} S Q_n + S \bar{Q}_n + \bar{R} Q_n \\ &= \underline{R S Q_n} + S \bar{Q}_n + \bar{R} Q_n \end{aligned}$$

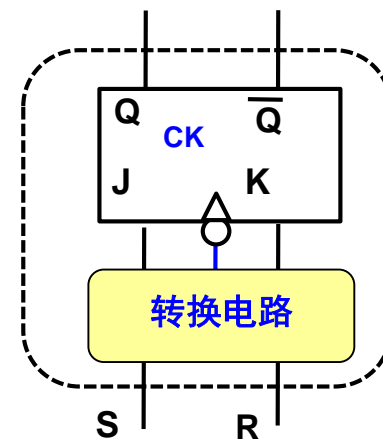
RS=0

$$J = f_1(R, S, Q)$$

$$K = f_2(R, S, Q)$$

$$J=S$$

$$K=R$$



触发器类型转换——D转其他

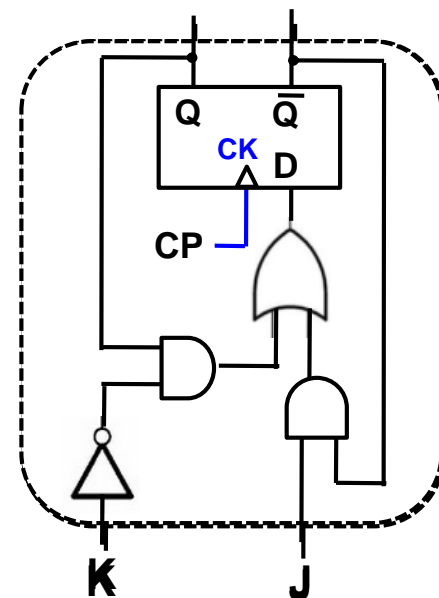
2. $D \rightarrow JK$ 、 T 、 SR

(1) $D \rightarrow JK$

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

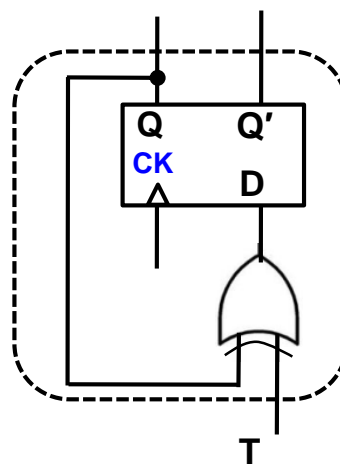
$$D = f(J, K, Q)$$

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



(2) $D \rightarrow T$

$$\left. \begin{array}{l} \text{T: } Q_{n+1} = T \oplus Q_n \\ \text{D: } Q_{n+1} = D \end{array} \right\} D = T \oplus Q_n$$



(3) $D \rightarrow SR$?

触发器类型转换——卡诺图法

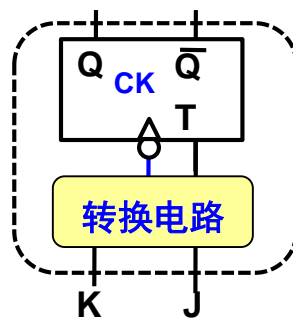
□ 卡诺图法

1. $T \rightarrow JK$ 、 D 、 SR

(1) $T \rightarrow JK$

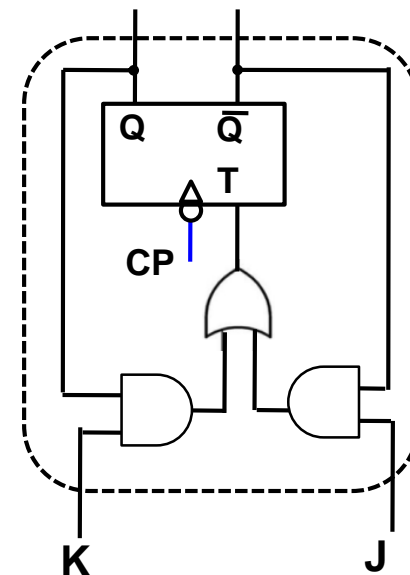
$Q_n \rightarrow Q_{n+1}$	T	J	K
0 \rightarrow 0	0	0	X
0 \rightarrow 1	1	1	X
1 \rightarrow 0	1	X	1
1 \rightarrow 1	0	X	0

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



$$T = J\bar{Q}_n + KQ_n \rightarrow$$

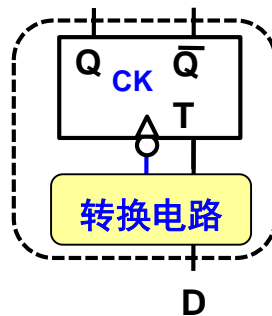
Q_n	JK			
	00	01	11	10
0	0	0	1	1
1	0	1	1	0



(2) $T \rightarrow D$

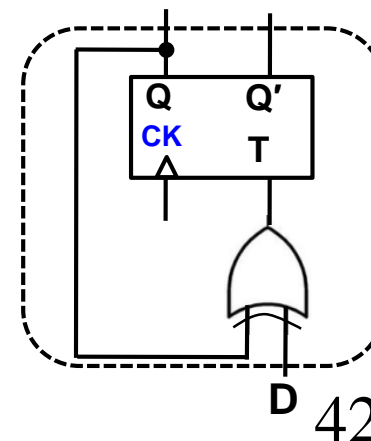
$Q_n \rightarrow Q_{n+1}$	T	D
0 \rightarrow 0	0	0
0 \rightarrow 1	1	1
1 \rightarrow 0	1	0
1 \rightarrow 1	0	0

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



$$T = D \oplus Q_n \rightarrow$$

Q_n	0	1
	D	D
0	0	1
1	1	0



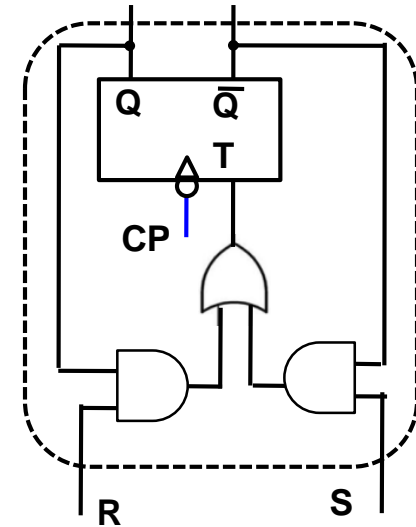
触发器类型转换——卡诺图法

(3) T → SR

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

$$T = S\bar{Q}_n + RQ_n$$

RS				
	00	01	11	10
0	0	1	X	0
1	0	0	X	1



2. SR → JK、D、T

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
- 时序逻辑的Verilog描述

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**输出不仅取决于当时的**输入信号**，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**。

时钟沿触发的always块描述

```
always @ (<敏感信号列表>)  
begin  
    //过程赋值  
    //if-else、case选择语句  
    //for、while等循环块  
end
```

边沿触发：时钟处在上升沿或下降沿时，语句被执行。

always @(posedge clk) ——时钟上升沿触发

always @(negedge clk) ——时钟下降沿触发

always @(posedge clk or negedge rst_n) 带异步复位的
时钟上升沿触发

非阻塞赋值

块内的赋值语句同时进行：先同时采样，最后一起更新

时序电路特点：输出不会随输入变化而立即变化

■ 非阻塞赋值

```
always @ ( posedge clk )  
begin  
    c<= b;  
    b<= a;  
    a<= d;  
End
```

结果：
a=2, b=5
c=3, d=2

初值：
a=5, b=3
c=10, d=2

■ 非阻塞赋值

```
always @ ( posedge clk )  
begin  
    a<= d;  
    b<= a;  
    c<= b;  
End
```

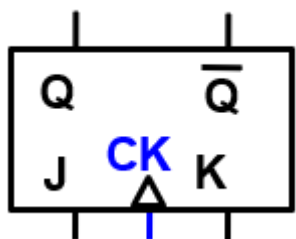
结果：
a=2, b=5
c=3, d=2

结果与书写的顺序无关
(原因：同步更新)

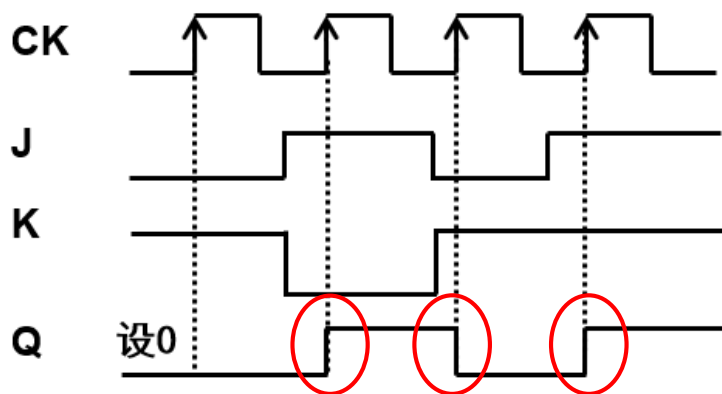
本质上，在一个时钟沿触发里，a得到d的值，但b得到的是a的旧值，c得到的是b的旧值

JK触发器的Verilog描述

- 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**。



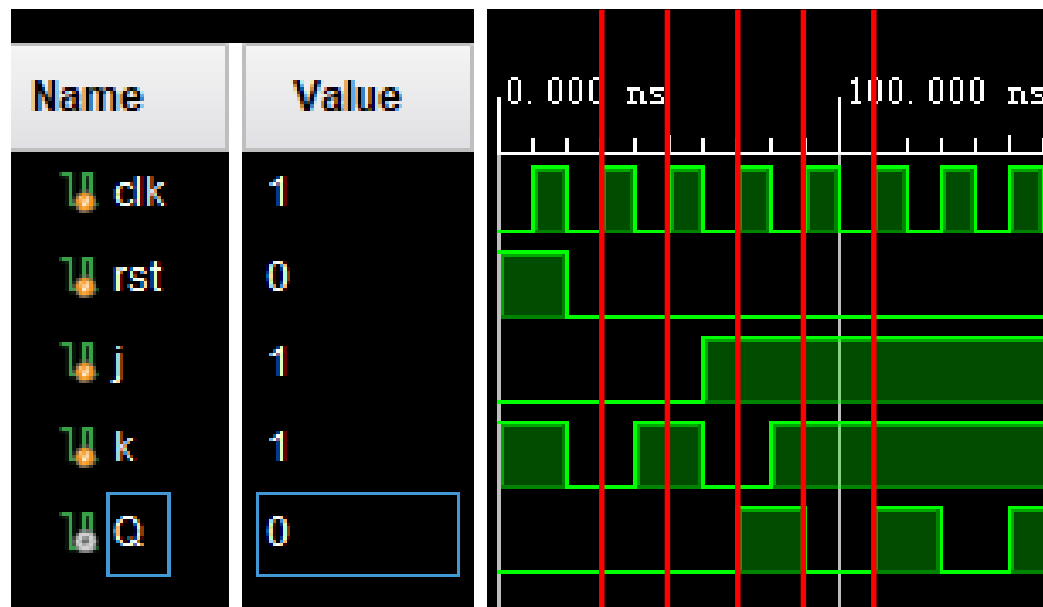
$$Q_{n+1} = J \bar{Q}_n + \bar{K} Q_n$$



```
module J_K (  
    input    wire clk,  
    input    wire j,  
    input    wire k,  
    input    wire rst,  
    output   reg  Q  
);  
  
    wire rst_n;  
  
    assign rst_n=~rst;  
  
    always@(posedge clk or negedge rst_n )begin  
        if (~rst_n)  
            Q <= 1'b0;  
        else  
            Q <= j && (~Q) || (~k) && Q;  
        end  
  
    endmodule
```

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

Testbench-时序逻辑



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

```
`timescale 1ns/1ps
```

```
module J_K_sim();
```

```
reg clk;
```

```
reg rst;
```

```
reg j,k;
```

```
wire Q;
```

```
J_K u_J_K(.clk(clk),.rst(rst),.j(j),.k(k),.Q(Q));
```

```
initial begin
```

```
clk=1'b0;j=1'b0;k=1'b1;rst=1'b1;
```

```
#20 rst=1'b0;j=1'b0;k=1'b0;
```

```
#20 j=1'b0;k=1'b1;
```

```
#20 j=1'b1;k=1'b0;
```

```
#20 j=1'b1;k=1'b1;
```

```
end
```

```
always #10 clk=~clk;
```

```
endmodule
```

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
- 时序逻辑的Verilog描述