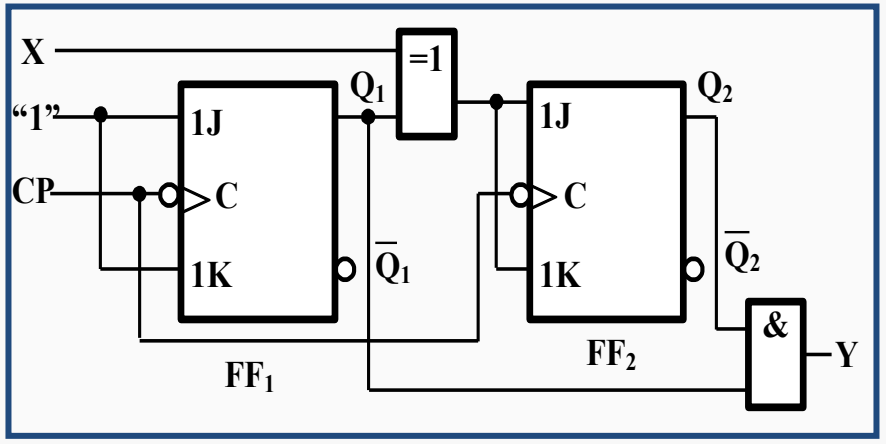
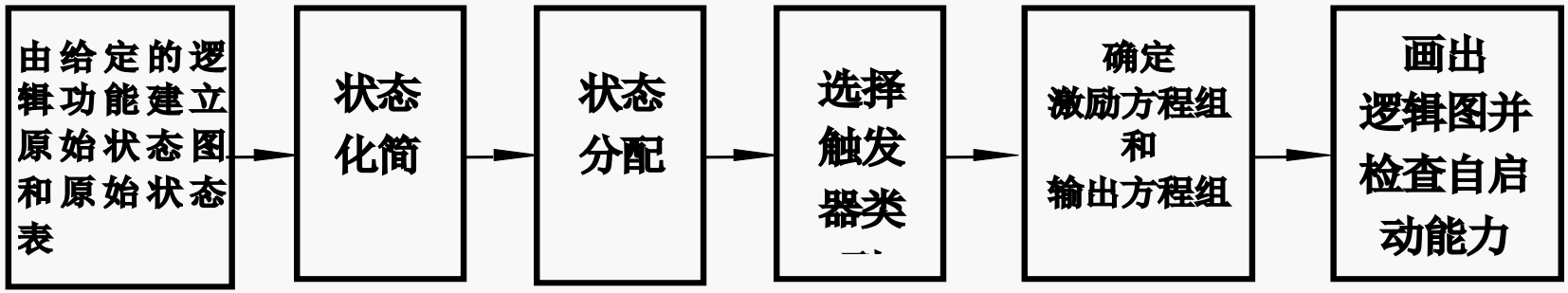
# 6 时序逻辑电路

# 6.3 时序逻辑电路的设计

同步时序逻辑电路的设计是分析的逆过程，其任务是根据实际逻辑问题的要求，设计出能实现给定逻辑功能的电路。




## 设计同步时序逻辑电路的一般步骤



# 6.3 时序逻辑电路的设计

1、**逻辑抽象：**建立原始状态图和原始状态表

（1）确定电路的输入变量、输出变量和电路的状态，并定义各变量和状态的含义；

（2）确定所有状态在不同输入条件下的转换关系，建立原始状态图；

（3）根据原始状态图建立原始状态表。

2、**状态化简：**合并等价状态，得到最简状态转换图

**等价状态：**如果两个状态，在**相同的输入**下有**相同的输出**，并转换到**相同的次态**，则它们为等价状态。（可以引申）

3、**状态分配：**也称为状态编码，给每个状态赋以二进制码。

(1) Highly Encoded：$2^{n-1} < M \leq 2^n$，（状态数M，触发器数n）

(2) One Hot Encoded：每个状态对应一个触发器为1，$M = n$

**Note：**(1) 以降低电路实现复杂度和提高时钟频率为目标；

(2) 触发器的数量少并不一定代表电路复杂度低；

(3) 组合电路的复杂度决定了时钟频率；

(4) 状态分配影响电路复杂度和是种速率，但不影响电路功能。

4、**触发器选型：**JK触发器、D触发器，影响电路结构

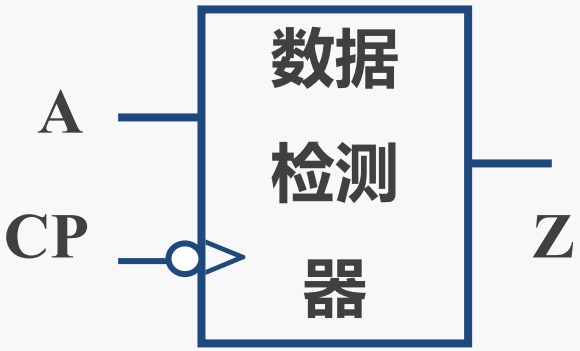
5、**求电路激励方程和输出方程**

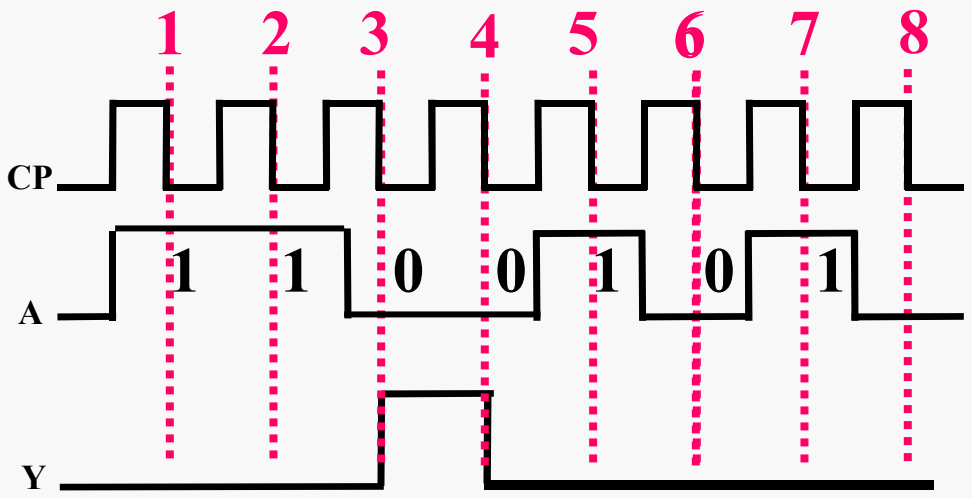
6、**自启动检查，画出逻辑电路图**

# 6.3 时序逻辑电路的设计

例 设计一个串行数据检测器。电路的输入信号A是与时钟脉冲同步的串行数据，其时序关系如下图所示。

输出信号为Y：要求电路在A信号输入出现“110”序列时，输出信号Y为1，否则为0。



# 6.3 时序逻辑电路的设计

1、逻辑抽象建立原始状态图或状态表.

（1）确定输入、输出变量及电路的状态数:

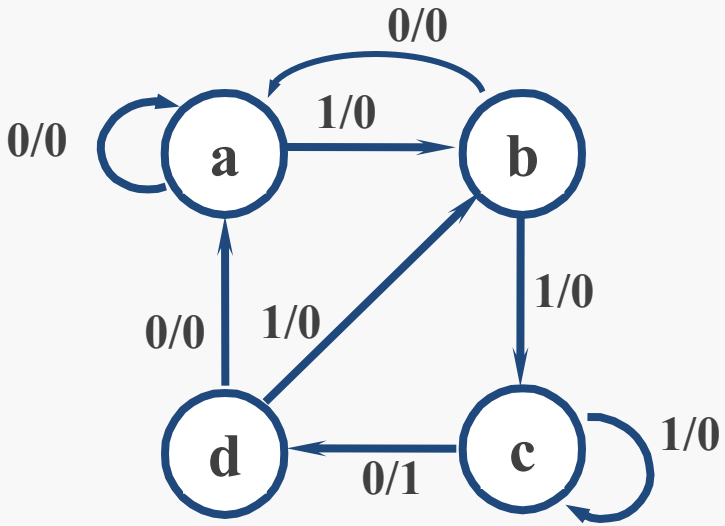
输入变量：A　输出变量：Y　状态数：4个

（2）定义输入 输出逻辑状态和每个电路状态的含义

a —— 初始状态;　b —— A输入1后;

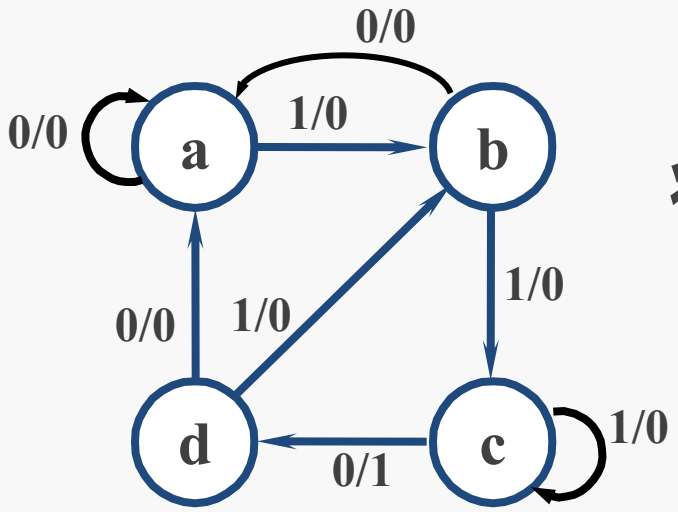
c —— A输入11后;　d —— A输入110后。

（3）按题意画出状态转换图。




原始状态图

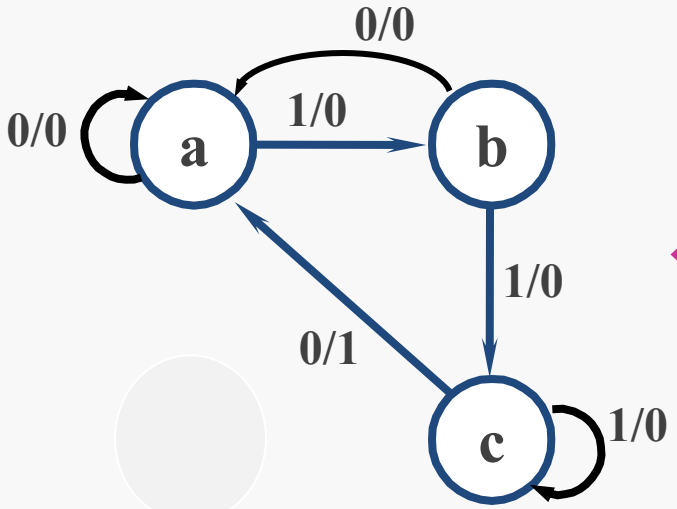
# 6.3 时序逻辑电路的设计




列出原始状态转换表

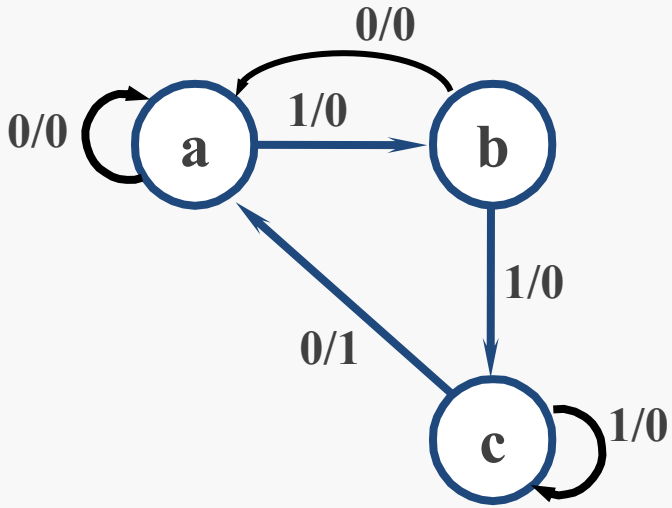
| 现态 | 次态/输出 | |
|---|---|---|
| | A=0 | A=1 |
| a | a/0 | b/0 |
| b | a/0 | c/0 |
| c | d/1 | c/0 |
| d | a/ 0 | b/0 |

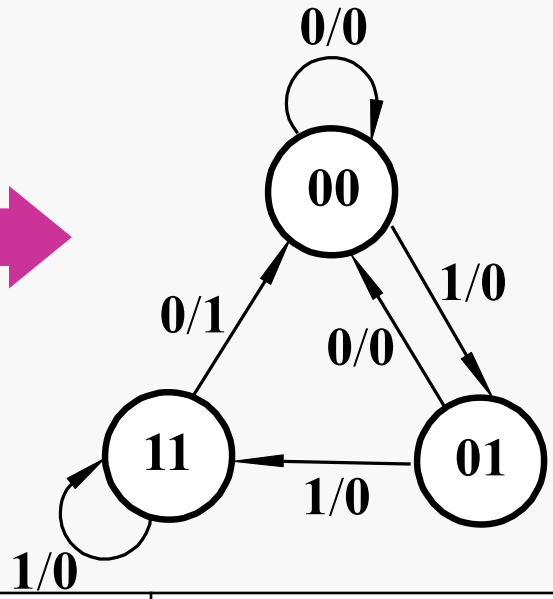
## 2. 状态化简




| 现态 | 次态 / 输出 | |
|---|---|---|
| | A=0 | A=1 |
| a | a/0 | b/0 |
| b | a/0 | c/0 |
| c | a/1 | c /0 |

3、状态分配　　令 $a=00$，$b=01$，$c=11$，



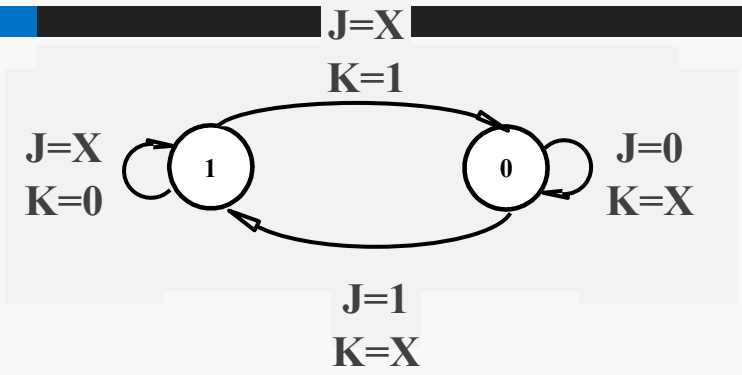

4、选择触发器的类型

触发器个数: 2

触发器类型：采用对 CP 下降沿敏感的JK 触发器

| 现态 $Q_1Q_0$ | $Q_1^{n+1}Q_0^{n+1}$ / Y | |
|---|---|---|
| | A=0 | A=1 |
| 00 | 00 / 0 | 01 /0 |
| 01 | 00 / 0 | 11 /0 |
| 11 | 00 / 1 | 11 /0 |

# 5. 求激励方程和输出方程



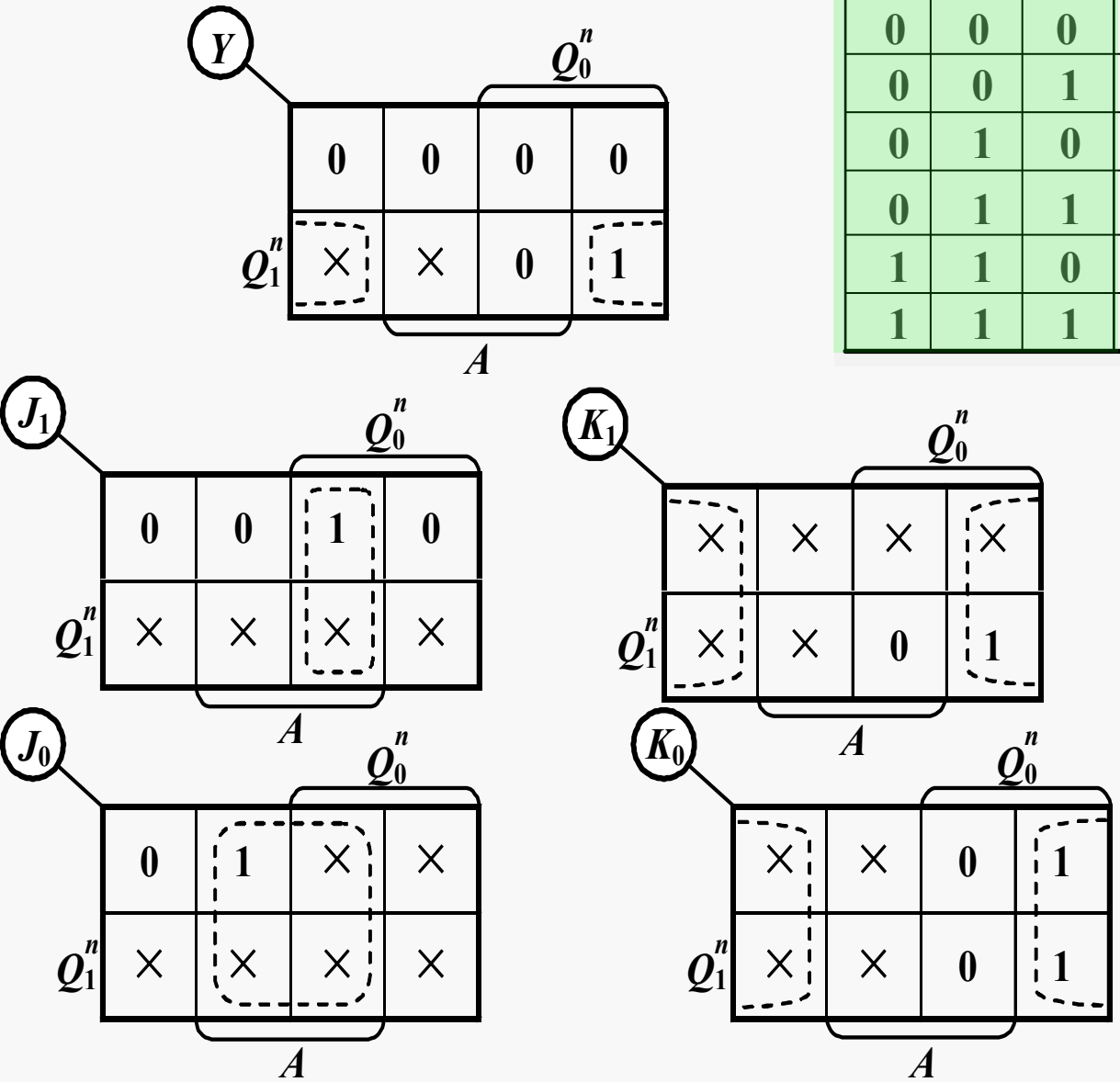

| 现态 $Q_1Q_0$ | $Q_1^{n+1}Q_0^{n+1}$ / Y | |
|---|---|---|
| | A=0 | A=1 |
| 00 | 00 / 0 | 01 /0 |
| 01 | 00 / 0 | 11 /0 |
| 11 | 00 / 1 | 11 /0 |

## 状态转换真值表及激励信号

| $Q_1^n$ | $Q_0^n$ | $A$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | $Y$ | 激励信号 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | $J_1$ | $K_1$ | $J_0$ | $K_0$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | × | 0 | × |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | × | 1 | × |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | × | × | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | × | 0 | × |
| 1 | 1 | 0 | 0 | 0 | 1 | × | 1 | × | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | × | 0 | × | 0 |

# 5、求激励方程和输出方程

| $Q_1^n$ | $Q_0^n$ | $A$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | $Y$ | 激励信号 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | $J_1$ | $K_1$ | $J_0$ | $K_0$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | × | 0 | × |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | × | 1 | × |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | × | × | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | × | 0 | × |
| 1 | 1 | 0 | 0 | 0 | 1 | × | 1 | × | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | × | 0 | × | 0 |




卡诺图化简得

输出方程

$$Y = Q_1\overline{A}$$

激励方程

$$J_1 = Q_0 A \quad K_1 = \overline{A}$$

$$J_0 = A \quad K_0 = \overline{A}$$

# 6. 画出逻辑电路图，并检查自启动能力

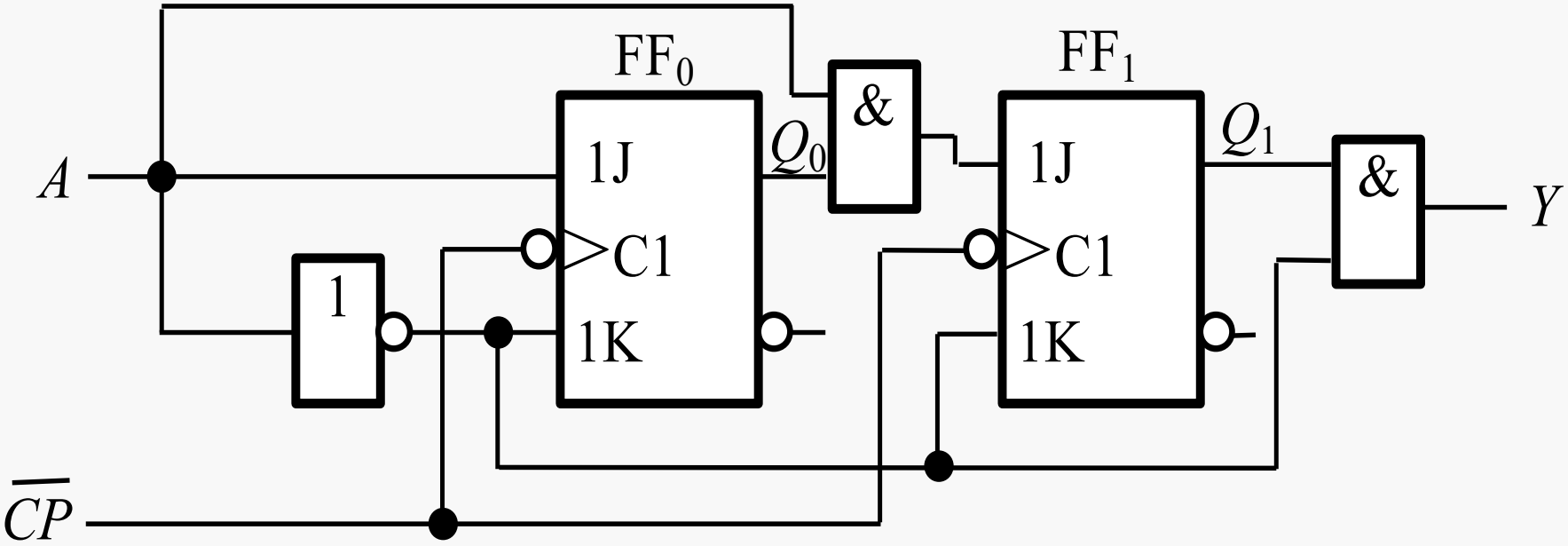
**激励方程**

$$J_1 = Q_0 A \qquad K_1 = \overline{A}$$

$$J_0 = A \qquad K_0 = \overline{A}$$

**输出方程**

$$Y = Q_1 \overline{A}$$



# 6. 画出逻辑电路图，并检查自启动能力

$J_1 = Q_0 A \qquad K_1 = A$

$J_0 = A \qquad K_0 = \overline{A}$

当 $Q_1 Q_0 = 10$时

A=0 $Y = 1$

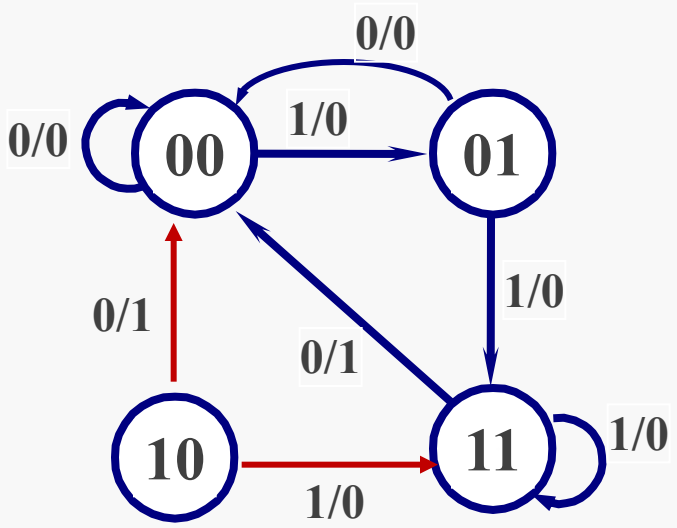
$J_1 = 0 \quad K_1 = 1 \quad Q_1^{n+1} = 0$

$J_0 = 0 \quad K_0 = 1 \quad Q_0^{n+1} = 0$

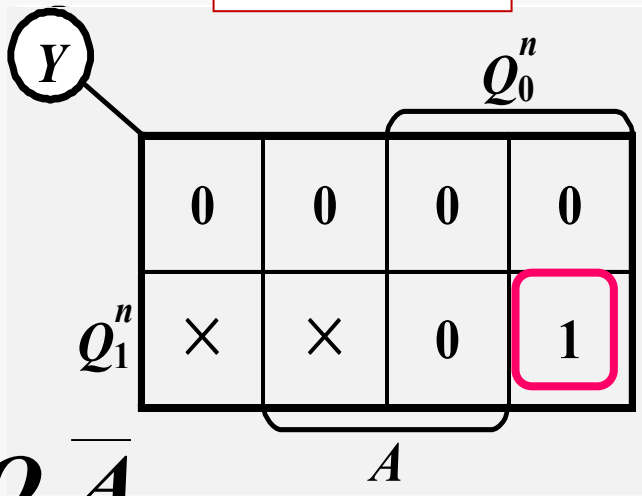
A=1 $Y = 0$

$J_1 = 0 \quad K_1 = 0 \quad Q_1^{n+1} = 1$

$J_0 = 1 \quad K_0 = 0 \quad Q_0^{n+1} = 1$

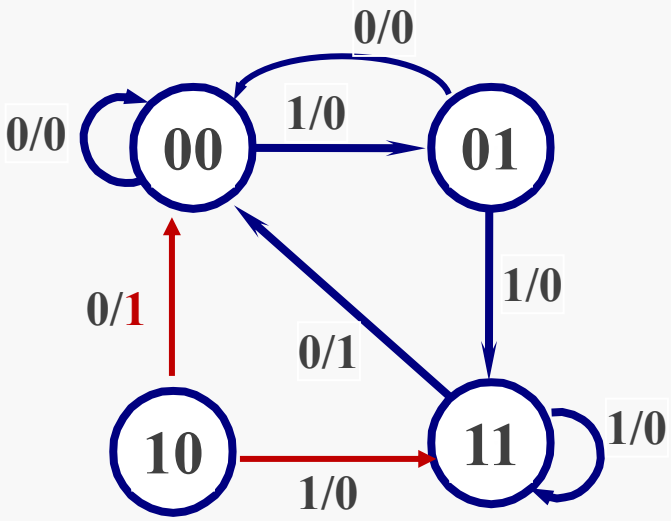



能自启动



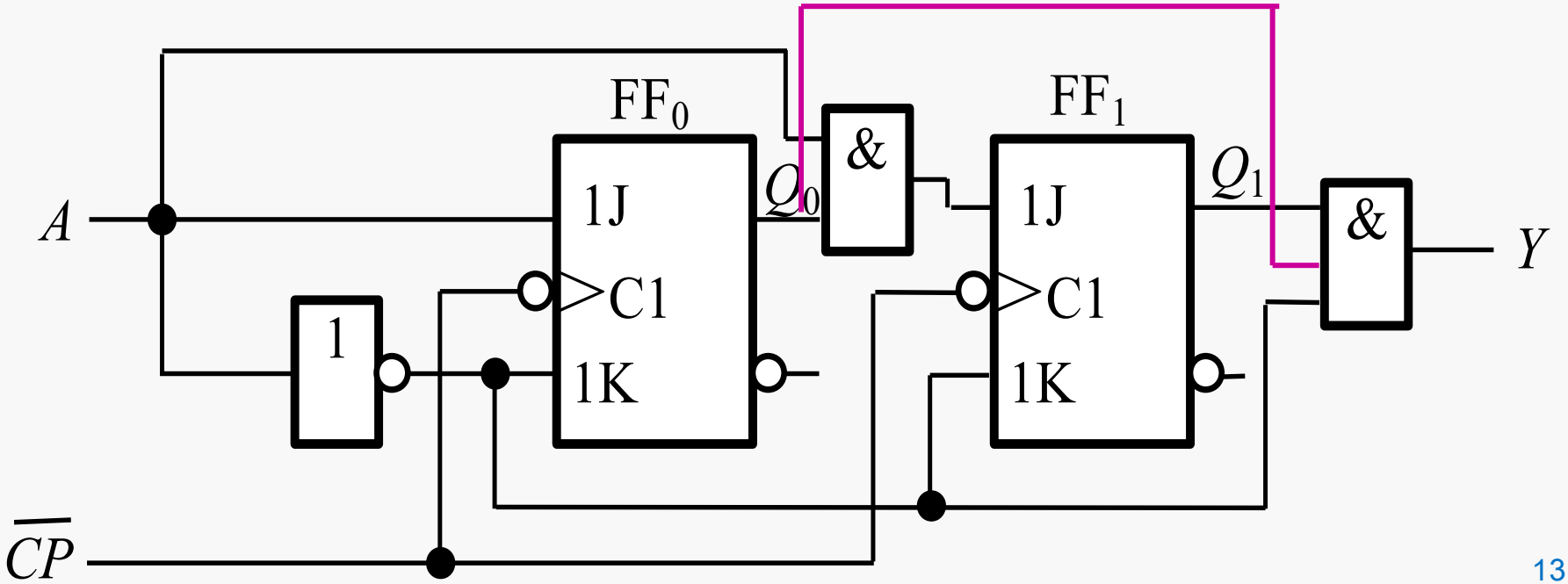

输出方程 $Y = Q_1 \overline{A}$ ➡ $Y = Q_1 Q_0 \overline{A}$

# 7. 几点讨论




1、自启动吗？？？彻底解决自启动问题的方法？？？

2、同步米里型电路，波形是否正确？？？该怎么办？



# 6.3 时序逻辑电路的设计（Verilog描述1）

**Highly Encoded**

```
reg[1:0] state;
parameter
S0=2'b00,
S1=2'b01,
S2=2'b11;
```

**1、摩尔或米里？**

**2、如果要设计米里型电路，该如何修改？**

```
always@(negedge CP or negedge rst) begin
If (!rst) begin state = S0; Y=0; end
else
   case(state)
   S0:  if(!A) begin    state = S0; Y = 0; end
        else begin      state = S1; Y = 0; end
   S1:  if(!A) begin    state = S0; Y = 0; end
        else begin      state = S2; Y = 0; end
   S2:  if(!A) begin    state = S0; Y = 1; end
        else begin      state = S2; Y = 0; end
   default : begin      state = S0; Y = 0; end
   endcase
end
```

# 6.3 时序逻辑电路的设计（Verilog描述2）

**One Hot Encoded**

```
reg[2:0] state;
parameter
S0=3'b001,
S1=3'b010,
S2=3'b100;
```

```
always@(negedge CP or negedge rst) begin
If (!rst) begin state = S0; Y=0; end
else
   case(state)
   S0:  if(!A) begin    state = S0; Y = 0; end
        else begin      state = S1; Y = 0; end
   S1:  if(!A) begin    state = S0; Y = 0; end
        else begin      state = S2; Y = 0; end
   S2:  if(!A) begin    state = S0; Y = 1; end
        else begin      state = S2; Y = 0; end
   default : begin      state = S0; Y = 0; end
   endcase
end
```

# 6.3 时序逻辑电路的设计（Verilog描述3）

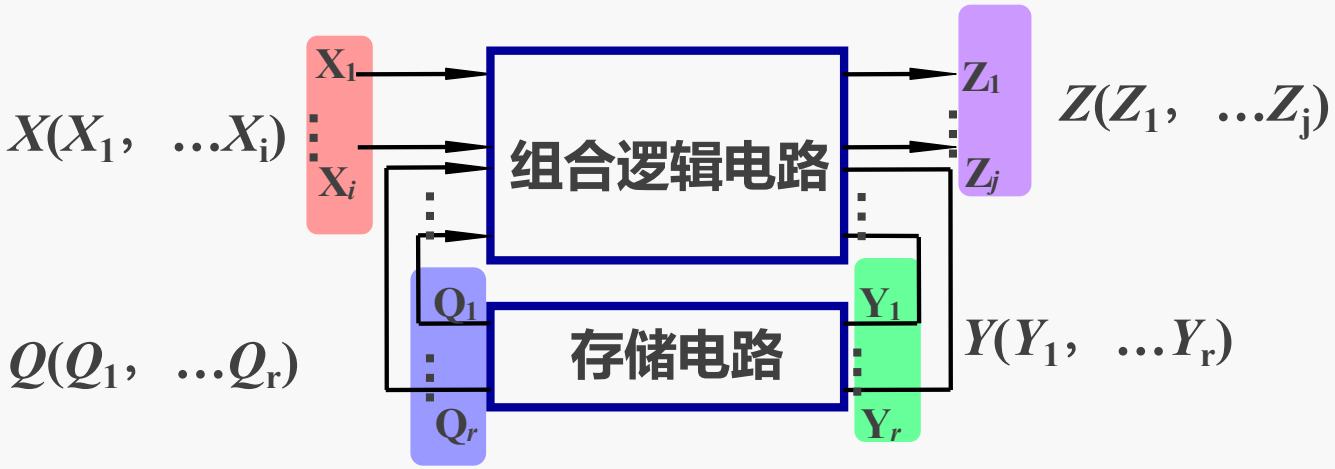
```
reg[2:0] state;
reg[2:0] new_state;
reg new_Y;
parameter S0=3'b001, S1=3'b010, S2=3'b100;
```

```
always@(A) begin
   case(state)
   S0:  if(!A) begin
            new_state = S0; new_Y= 0; end
        else begin
            new_state = S1; new_Y = 0; end
   ……
   default : begin
            new_state = S0; new_Y= 0; end
   endcase
end
```

# 6.3 时序逻辑电路的设计（Verilog描述3）

```
always@(negedge CP or negedge rst)
if(!rst)     begin
   state = S0;
   Y = 0;    end
else         begin
   state = new_state;
   Y = new_Y;      end
```

**如何理解此Verilog 描述状态机的方式？**



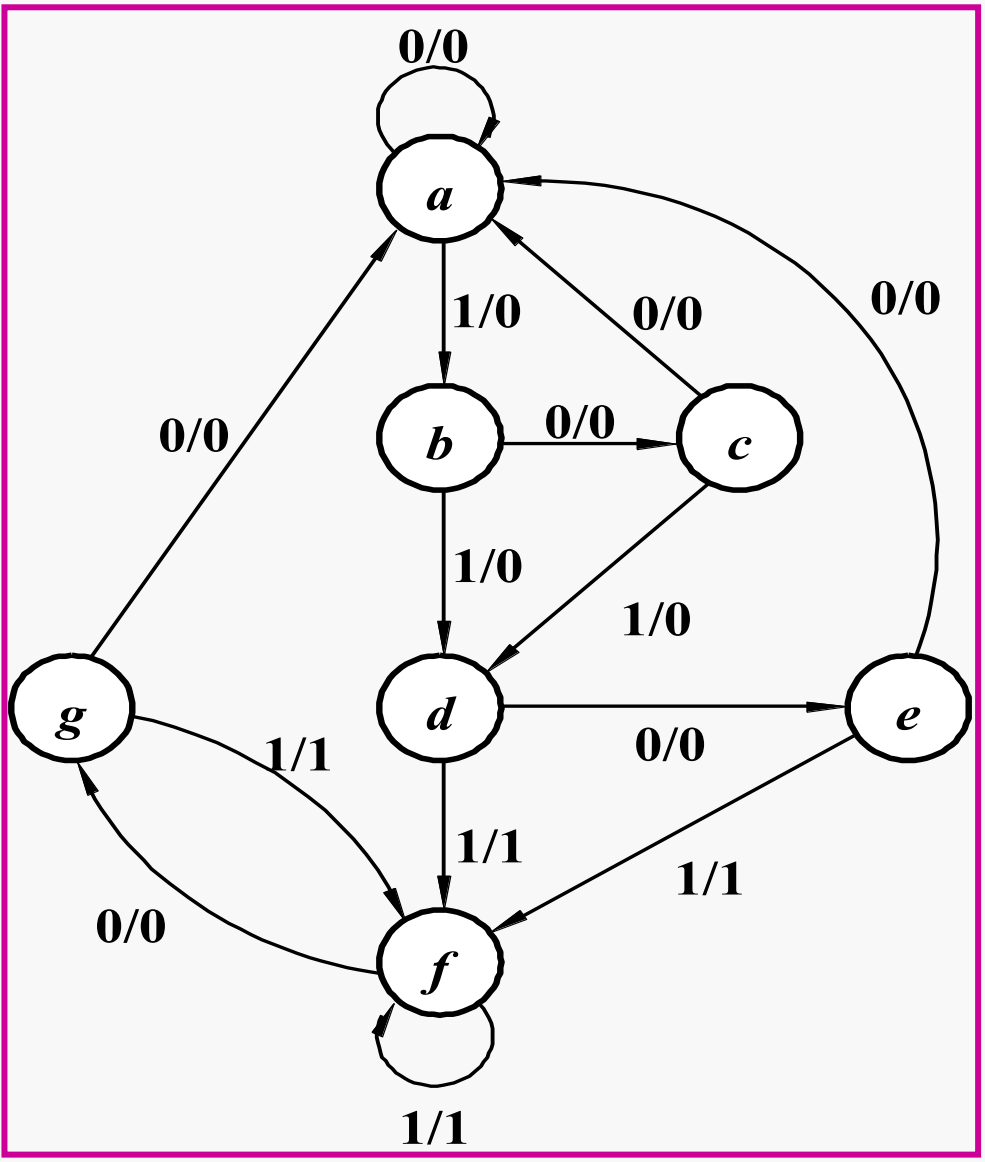
例 设计一个串行数据检测器。电路的输入信号X是与时钟脉冲同步的串行数据，输出信号为Z；要求电路在X信号输入出现101序列时，输出信号Z为1，否则为0。要求以下列3种方式画出状态转换图。

1、可重叠

2、不可重叠

3、以3个数据为一组

# 6.3 时序逻辑电路的设计（状态化简）




## 原始状态表

| 现态 ($S^n$) | 次态/输出 ($S^{n+1}/Y$) | |
|---|---|---|
| | $A$=0 | $A$=1 |
| $a$ | $a$ / 0 | $b$ / 0 |
| $b$ | $c$ / 0 | $d$ / 0 |
| $c$ | $a$ / 0 | $d$ / 0 |
| $d$ | $e$ / 0 | $f$ / 1 |
| $e$ | $a$ / 0 | $f$ / 1 |
| $f$ | $g$ / 0 | $f$ / 1 |
| $g$ | $a$ / 0 | $f$ / 1 |

# 6.3 时序逻辑电路的设计（状态化简）

| 现态 ($S^n$) | 次态/输出 ($S^{n+1}/Y$) | |
|---|---|---|
| | $A$=0 | $A$=1 |
| $a$ | $a$ / 0 | $b$ / 0 |
| $b$ | $c$ / 0 | $d$ / 0 |
| $c$ | $a$ / 0 | $d$ / 0 |
| $d$ | $e$ / 0 | $f$ / 1 |
| $e$ | $a$ / 0 | $f$ / 1 |
| $f$ | $g$ / 0 | $f$ / 1 |

| 现态 ($S^n$) | 次态/输出 ($S^{n+1}/Y$) | |
|---|---|---|
| | $A$=0 | $A$=1 |
| $a$ | $a$ / 0 | $b$ / 0 |
| $b$ | $c$ / 0 | $d$ / 0 |
| $c$ | $a$ / 0 | $d$ / 0 |
| $d$ | $e$ / 0 | $f$ / 1 |
| $e$ | $a$ / 0 | $f$ / 1 |
| $f$ | $e$ / 0 | $f$ / 1 |

# 6.3 时序逻辑电路的设计（状态化简）

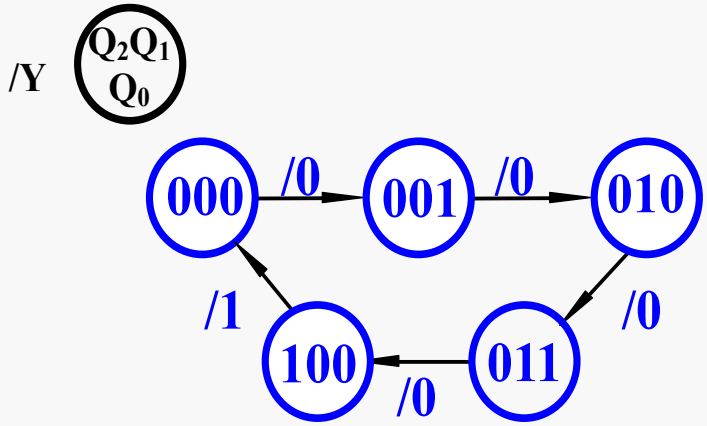
| 现态 ($S^n$) | 次态/输出 ($S^{n+1}/Y$) | |
|---|---|---|
| | $A$=0 | $A$=1 |
| *a* | *a* / 0 | *b* / 0 |
| *b* | *c* / 0 | *d* / 0 |
| *c* | *a* / 0 | *d* / 0 |
| *d* | *e* / 0 | *f* / 1 |
| *e* | *a* / 0 | *f* / 1 |

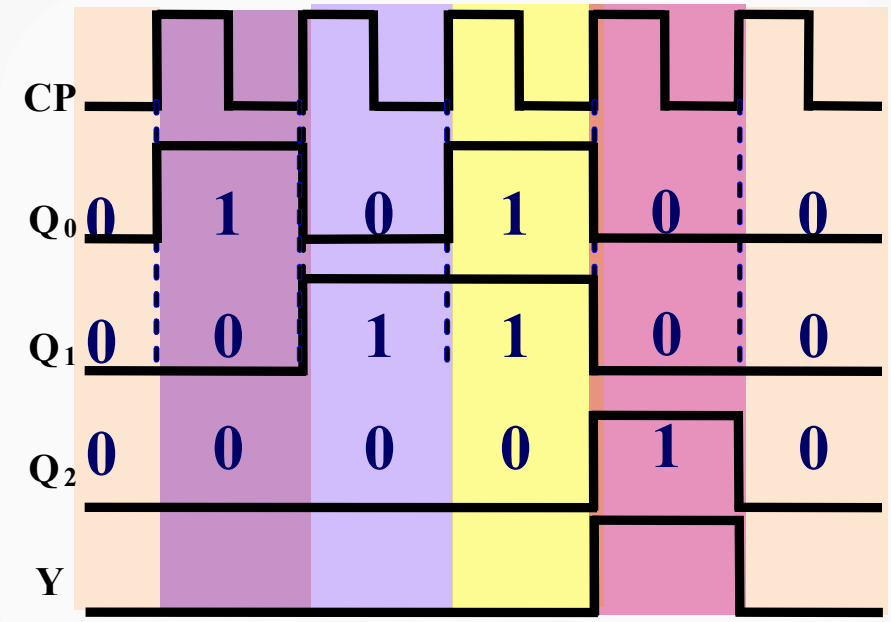
最简的化简状态表

| 现态 ($S^n$) | 次态/输出 ($S^{n+1}/Y$) | |
|---|---|---|
| | $A$=0 | $A$=1 |
| *a* | *a* / 0 | *b* / 0 |
| *b* | *c* / 0 | *d* / 0 |
| *c* | *a* / 0 | *d* / 0 |
| *d* | *e* / 0 | *d* / 1 |
| *e* | *a* / 0 | *d* / 1 |

例 试设计一个同步时序电路，要求电路中触发器$Q_0$、$Q_1$、$Q_2$及输出Y端的信号与CP时钟脉冲信号波形满足下图所示的时序关系。

解：1、可直接由波形图画出电路状态图




2、确定触发器的类型和个数

触发器个数： 3个

触发器类型：上升沿触发的JK触发器

3、求出电路的激励方程和输出方程

| $Q_2^n$ | $Q_1^n$ | $Q_0^n$ | $Q_2^{n+1}$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | Y | $J_2$ | $K_2$ | $J_1$ | $K_1$ | $J_0$ | $K_0$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | X | 0 | X | 1 | X |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | X | 1 | X | X | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | X | X | 0 | 1 | X |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | X | X | 1 | X | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | X | 1 | 0 | X | 0 | X |

$J_2$

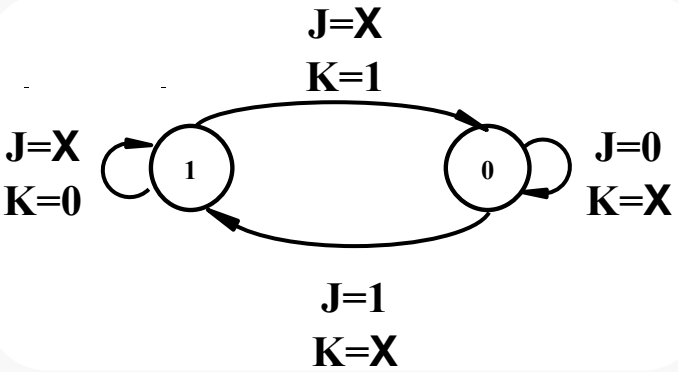
| $Q_2^n$ \ $Q_1^nQ_0^n$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 1 | X | X | X | X |

$$J_2 = Q_0^n Q_1^n$$

$K_2$

| $Q_2^n$ \ $Q_1^nQ_0^n$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | X | X | X | X |
| 1 | 1 | X | X | X |

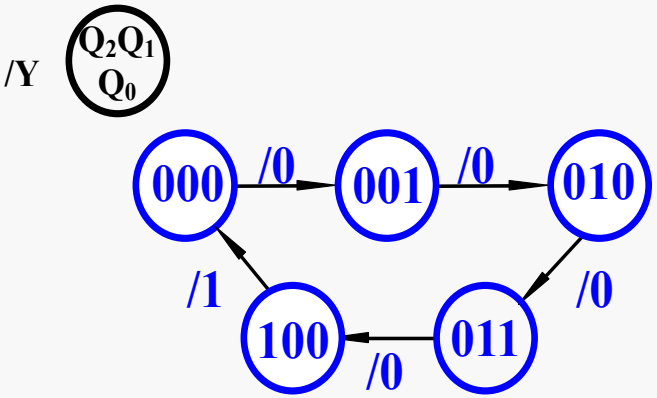
$$K_2 = 1$$



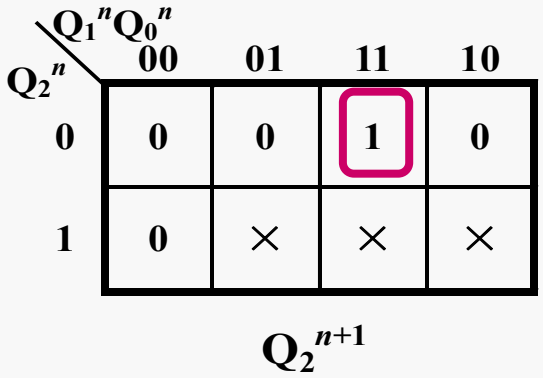

$$J_1 = Q_0^n \quad K_1 = Q_0^n$$

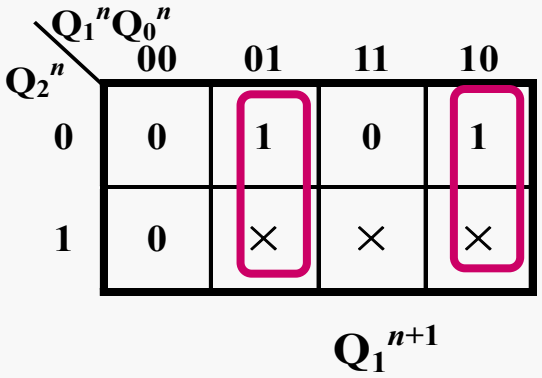
$$J_0 = \overline{Q_2^n} \quad K_0 = 1$$

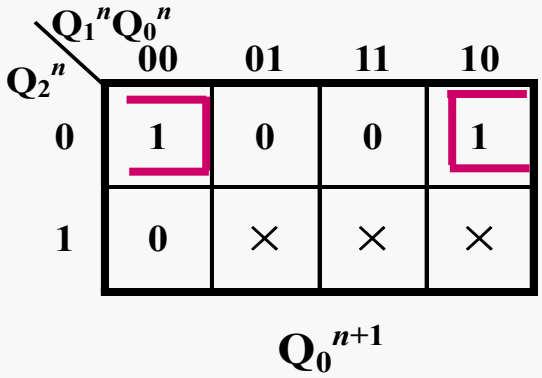
# 求激励方程的第二种方法




| $Q_2^n$ | $Q_1^n$ | $Q_0^n$ | $Q_2^{n+1}$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | Y |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |




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

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

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

$$\begin{cases} Q_0^{n+1} = J_0\overline{Q_0^n} + \overline{K_0}Q_0^n \\ Q_0^{n+1} = \overline{Q_2^n}\cdot\overline{Q_0^n} + 0.Q_0^n \end{cases}$$

$J_0 = \overline{Q_2^n}$　　$K_0=1$

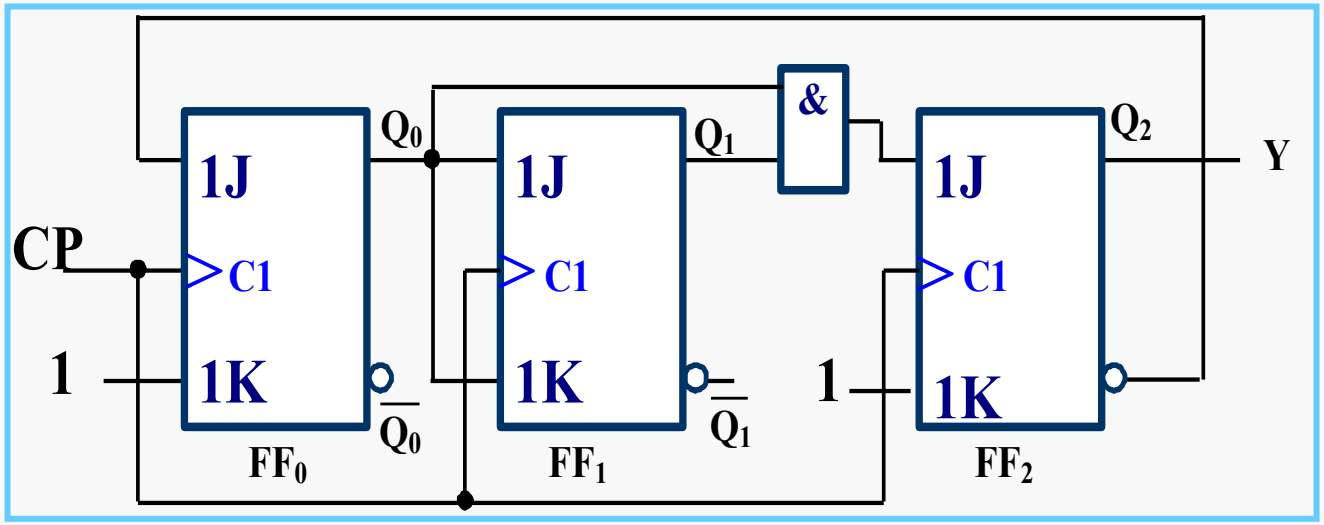
$$\begin{cases} Q_1^{n+1} = J_1\overline{Q_1^n} + \overline{K_1}Q_1^n \\ Q_1^{n+1} = Q_0^n\overline{Q_1^n} + \overline{Q_0^n}Q_1^n \end{cases}$$

$J_1=Q_0^n$　　$K_1=Q_0^n$

$$\begin{cases} Q_2^{n+1} = Q_0^nQ_1^n\overline{Q_2^n} = Q_0^nQ_1^n\overline{Q_2^n} + 0\cdot Q_2^n \\ Q_2^{n+1} = J_2\overline{Q_2^n} + \overline{K_2}Q_2^n \end{cases}$$

$J_2 = Q_0^nQ_1^n$　　$K_2 = 1$

**输出方程：** $Y = Q_2^n$



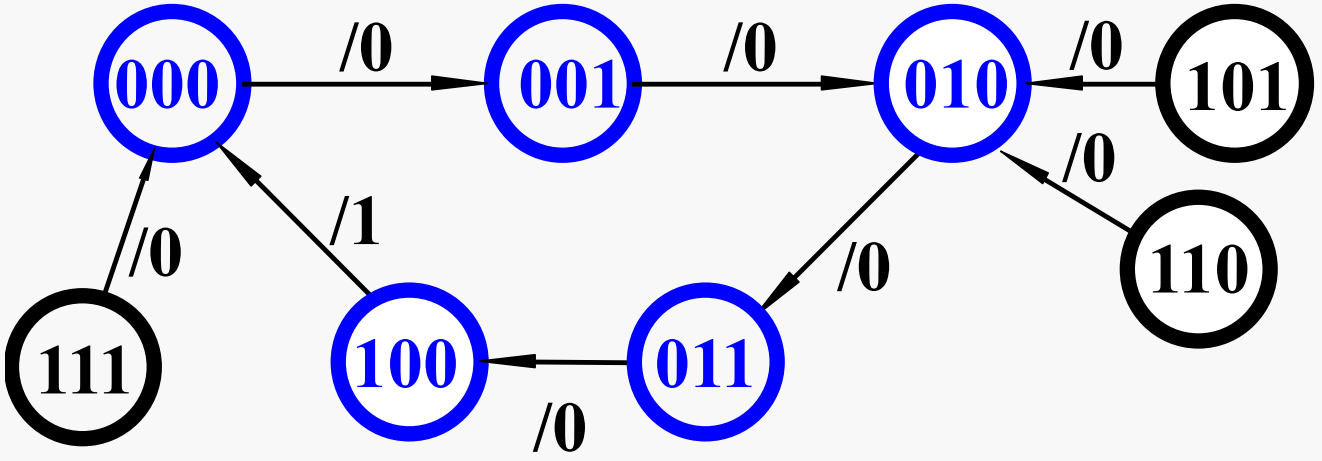
# 4、检查自启动能力

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

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

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

| $Q_2^n$ | $Q_1^n$ | $Q_0^n$ | $Q_2^{n+1}$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | Y |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |




**电路具备自启动能力**

# 4、检查自启动能力

| $Q_2^n$ | $Q_1^n$ | $Q_0^n$ | $Q_2^{n+1}$ | $Q_1^{n+1}$ | $Q_0^{n+1}$ | Y |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 |

输出方程： $Y = Q_2^n$ ⟹ $Y = Q_2^n \cdot \overline{Q_1^n} \cdot \overline{Q_0^n}$

# 4、Verilog设计

```
reg[2:0] state;
parameter
S0=3'b000,
S1=3'b001,
S2=3'b010,
S3=3’b011,
S4=3’b100;
```

**如果计数器的模很大该怎么办？**

```
always@(posedge CP) begin
   case(state)
   S0:  begin
        state = S1; Y = 0; end
   S1:  begin
        state = S2; Y = 0; end
   S2:  begin
        state = S3; Y = 0; end
   S3:  begin
        state = S4; Y = 0; end
   S4:  begin
        state = S0; Y = 1; end
   default : begin
        state = S0; Y = 0; end
   endcase
end
```