数字逻辑设计

张春慨

School of Computer Science

ckzhang@hit.edu.cn

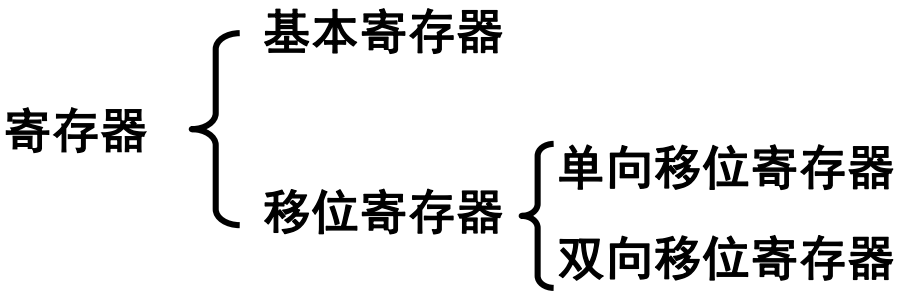
Unit 11 寄存器和计数器

- 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

寄存器——

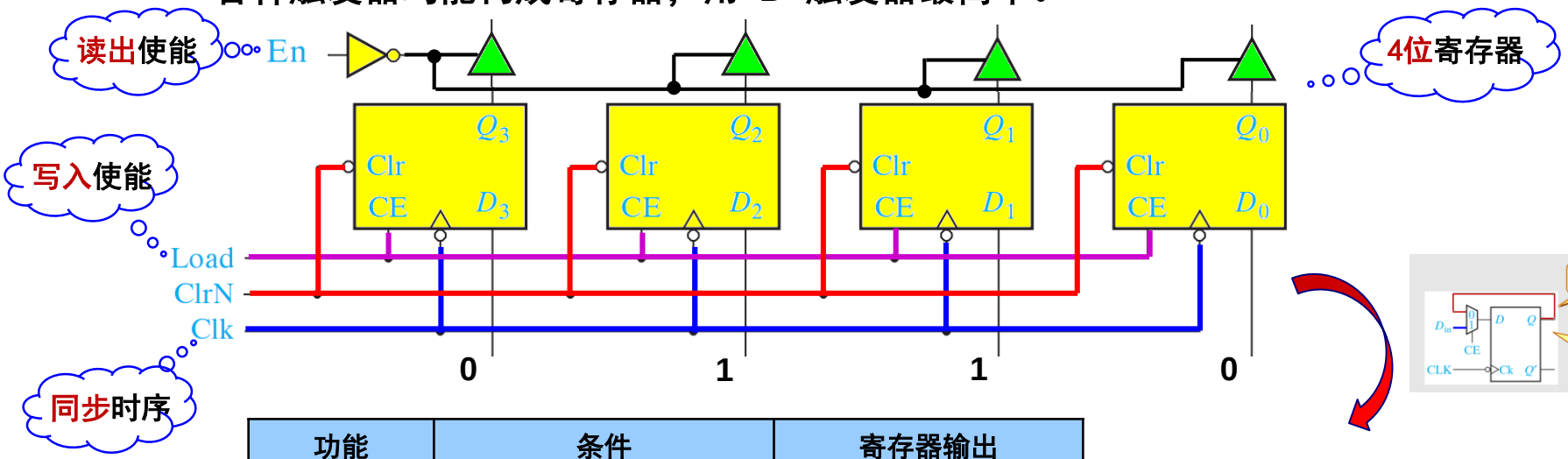
- 是计算机的一个重要部件，用于暂时存放一组二值代码（如参加运算的数据、运算结果、指令等）。
- 由触发器及控制门组成



- 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）

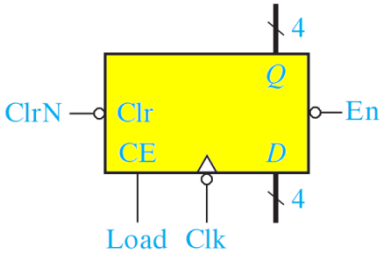
基本寄存器

- 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。



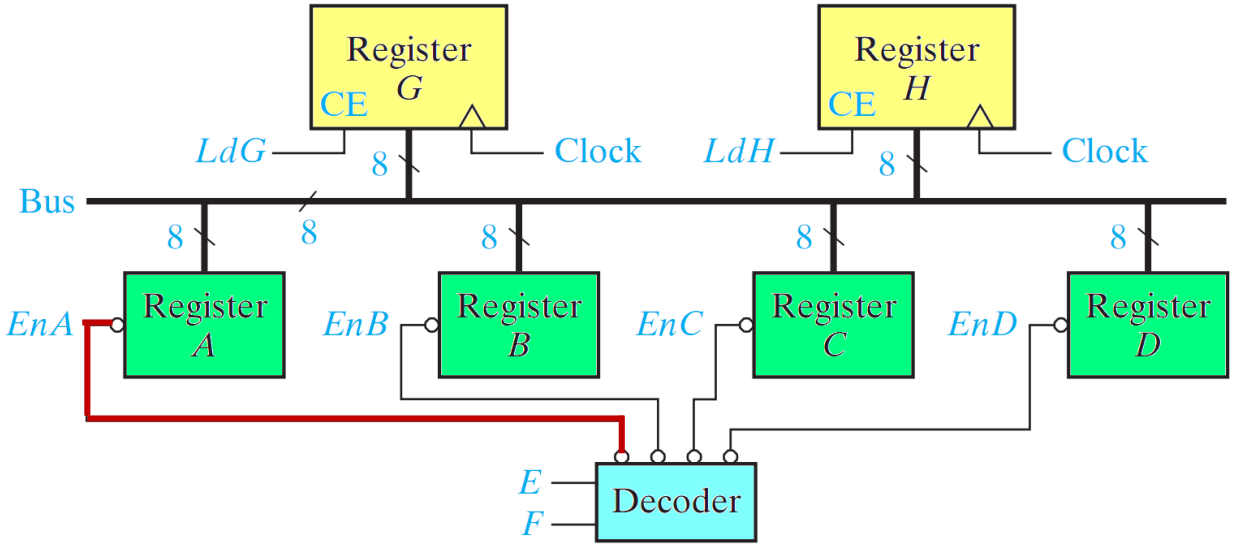
基本寄存器功能表

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	$Q_3Q_2Q_1Q_0$=$D_3D_2D_1D_0$



寄存器的应用

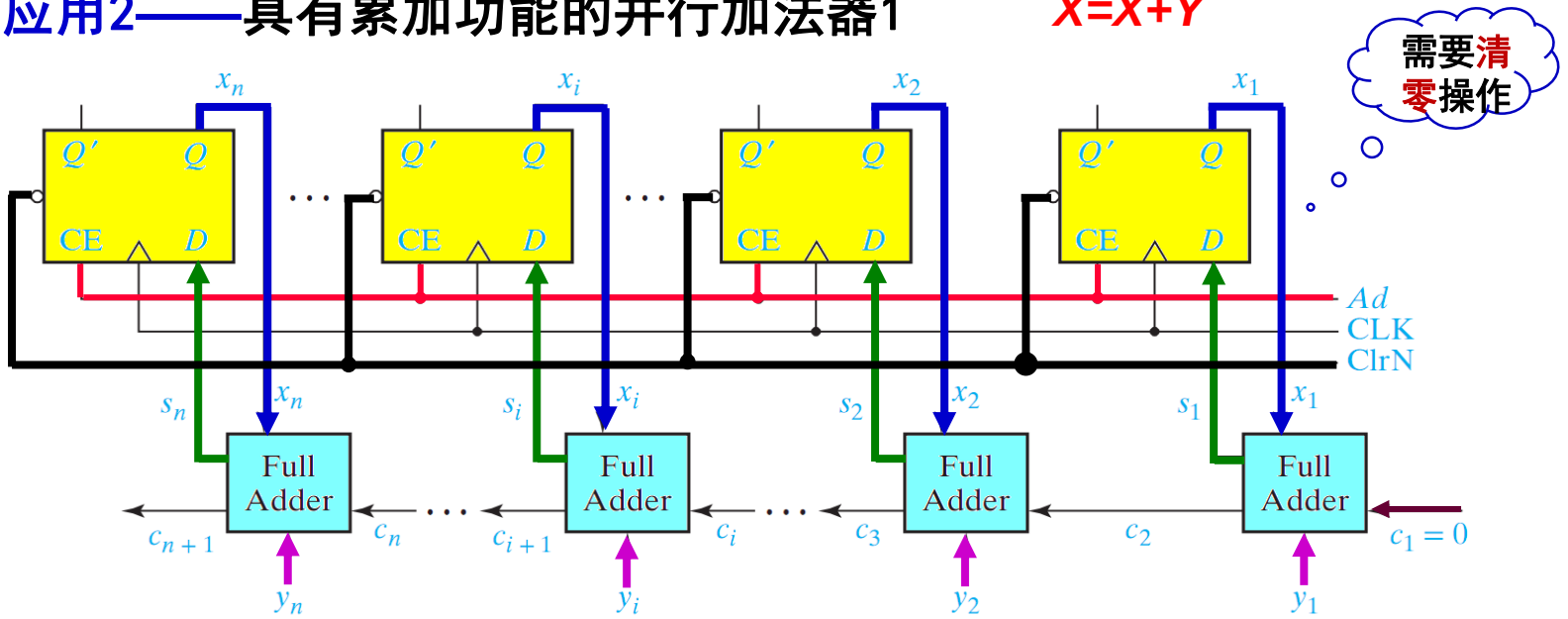
- 应用1——利用三态总线进行数据传送



- **Register A to G: EF=00，且LdG=1,LdH=0, clk ↑**
- **Register B to H: EF=01，且LdG=0,LdH=1, clk ↑**

寄存器的应用

- **应用2——具有累加功能的并行加法器1** $X=X+Y$

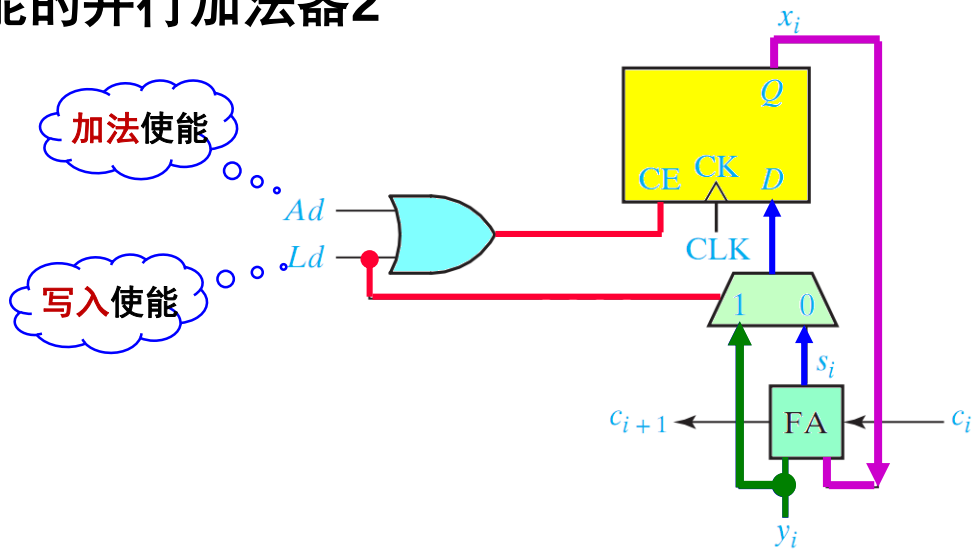


1. 初始化清零：ClrN=0，则Q_n .....Q_0 =0, 即X_n .....X_0 =0
2. ClrN=1, 将y_i送到全加器输入端
3. 执行$S_i = y_i + x_i + C_i$
4. 存储累加和：ClrN=1, Ad=1, CLK ↑ 到来时, 寄存器 $Q_i=S_i$

寄存器的应用

- **应用2——具有累加功能的并行加法器2**

$X=X+Y$



- **初始化：**

  Ld=1, 则CE=1, 当ck↑到来时, $Q_i = y_i$即$y_i \rightarrow x_i$，将x_i送到全加器的另一个输入端

- 送入第二个操作数y_i，执行 $S_i = y_i + x_i + c_i$

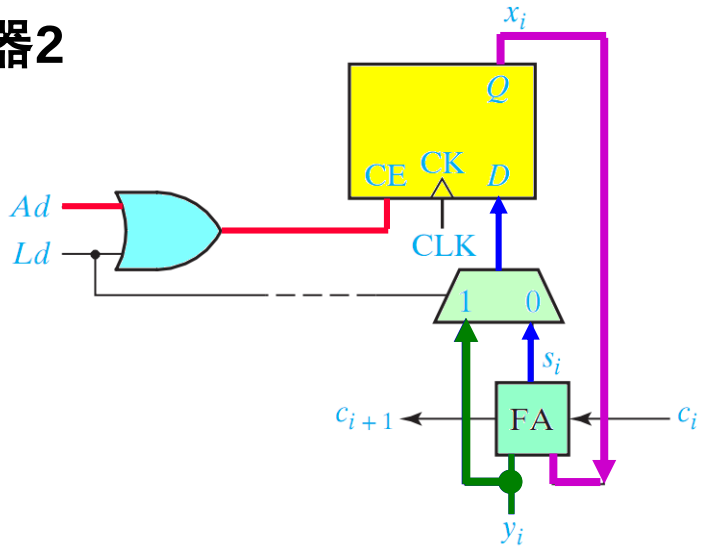
寄存器的应用

■应用2——具有累加功能的并行加法器2

$X=X+Y$

与方案1比较：

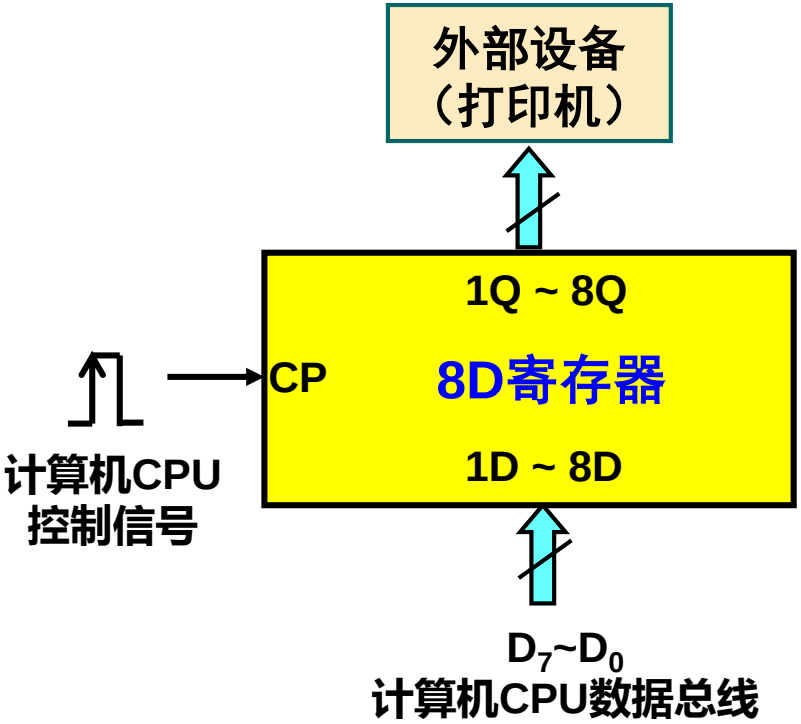
触发器不需要初始清零，通过一个二选一数据选择器，在第一个时钟沿送入一个操作数，之后在每个时钟沿送入累加和



- 初始化：
  Ld=1, 则CE=1, 当ck↑到来时, $Q_i=y_i$即$y_i \rightarrow x_i$，将x_i送到全加器的另一个输入端
- 送入第二个操作数y_i，执行$S_i=y_i+x_i$
- Ld=0, Ad=1, ck↑到来时: $x_i = s_i$
- 保持：Ld=0, Ad=0

寄存器的应用

- 应用3——计算机并行输入/输出接口



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）

移位寄存器

■ 移位寄存器——

- ➢ 每来一个时钟脉冲，寄存器里存储的数据，能依次的左移或右移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

移位寄存器
- 单向移位寄存器
  - 左移寄存器
  - 右移寄存器
- 双向移位寄存器

- ➢ 工作方式

❖数据输入方式
- 串行输入
- 并行输入

＋

❖数据输出方式
- 串行输出
- 并行输出

⇒

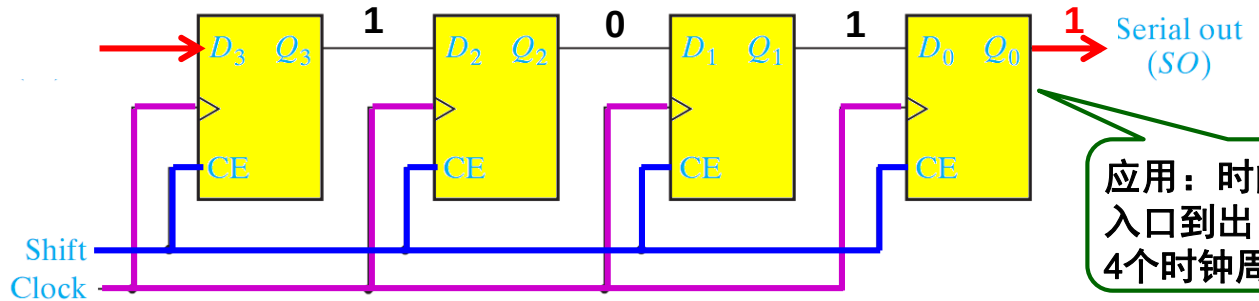
工作方式
- 串入_串出
- 串入_并出
- 并入_串出
- 并入_并出

右移寄存器

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**

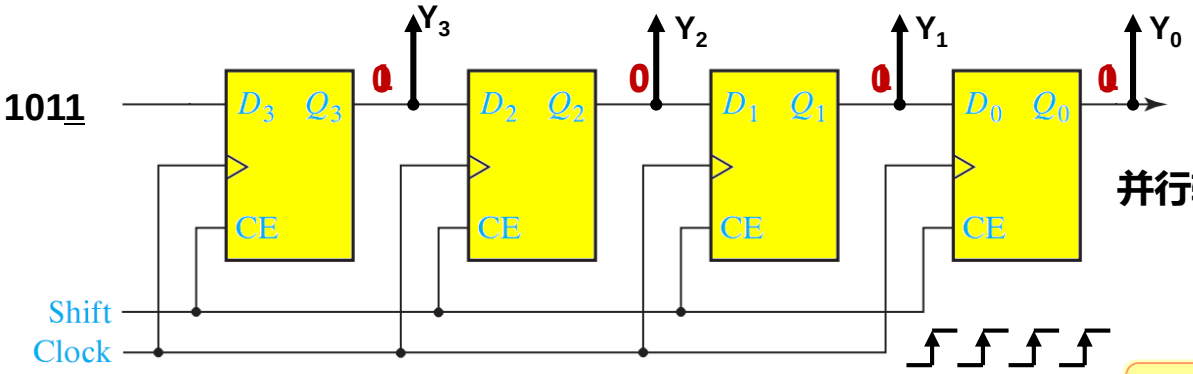
应用：时间延迟，入口到出口，滞后4个时钟周期

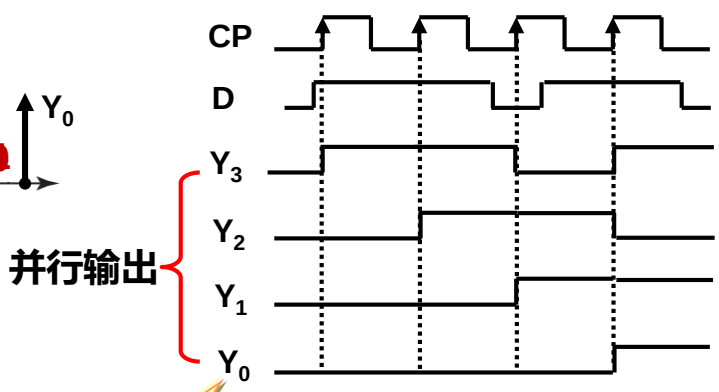
同步时序

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）

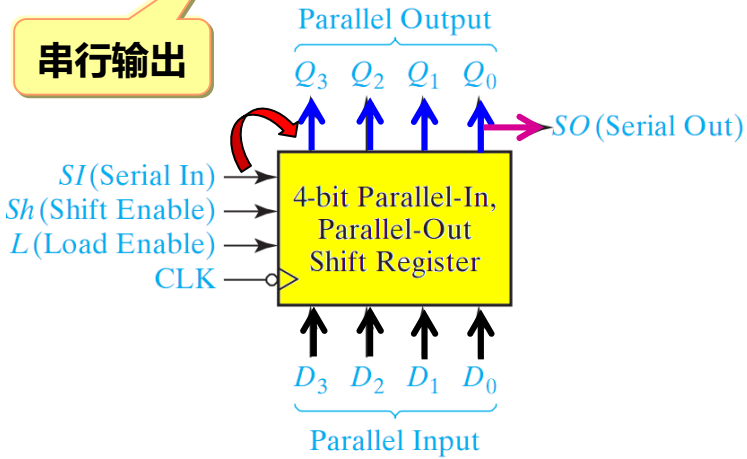




(3). 并入/并出（Parallel in / Parallel out）

(4). 并入/串出（Parallel in / Serial out）

Inputs		Next State				
Sh (Shift)	*L* (Load)	Q_3^+	Q_2^+	Q_1^+	Q_0^+	Action
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

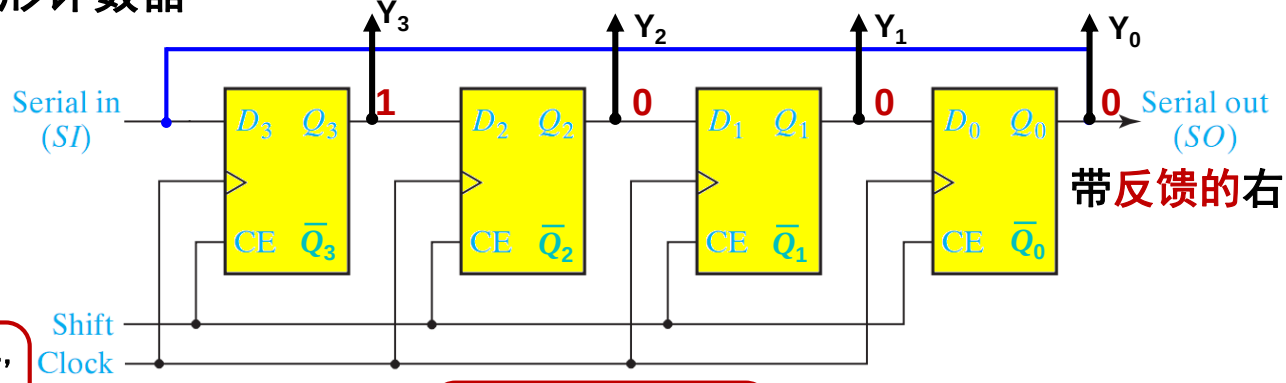


右移寄存器

右移寄存器的应用——

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。

(1).环形计数器

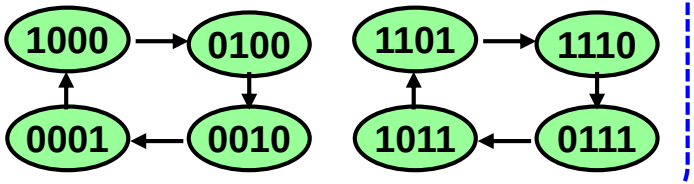


带反馈的右移寄存器

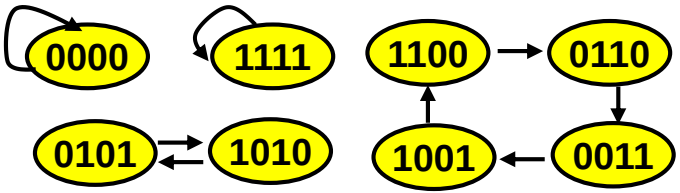
优点：电路简单，输出具有二进制译码器的特点

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



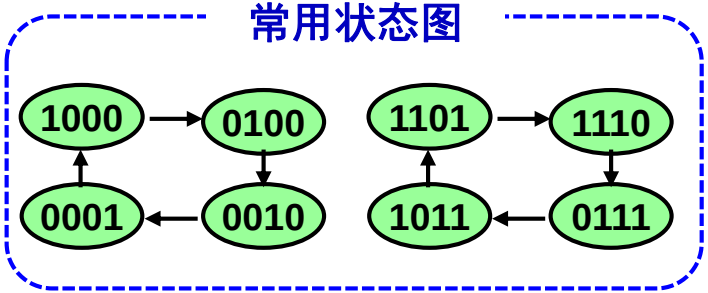
不常用状态图

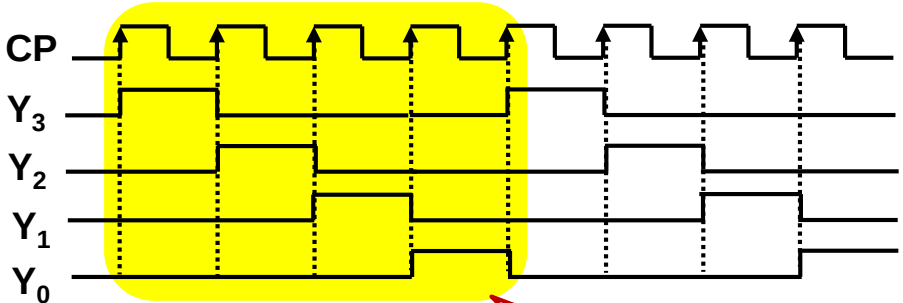


右移寄存器

优点：电路简单，输出具有二进制译码器的特点

常用状态图



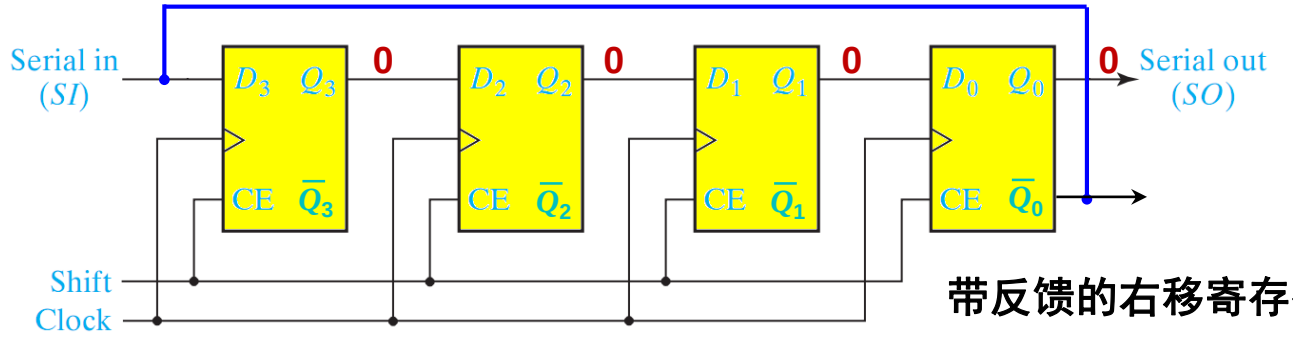


特点：一个周期之内的波形与译码器输出相同

右移寄存器的应用

- **右移寄存器的应用——**

(2).扭环形计数器

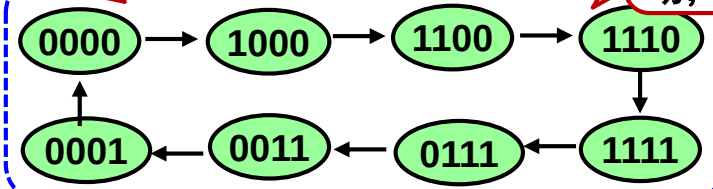


带反馈的右移寄存器

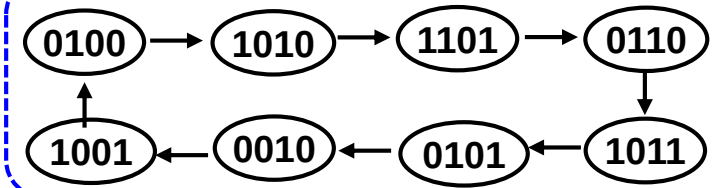
电路具有格雷码输出的特点

2^n个状态使用了$2n$个；不能自启动，需要预置

常用状态图

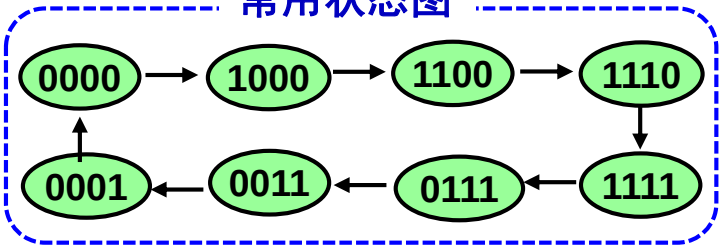


不常用状态图



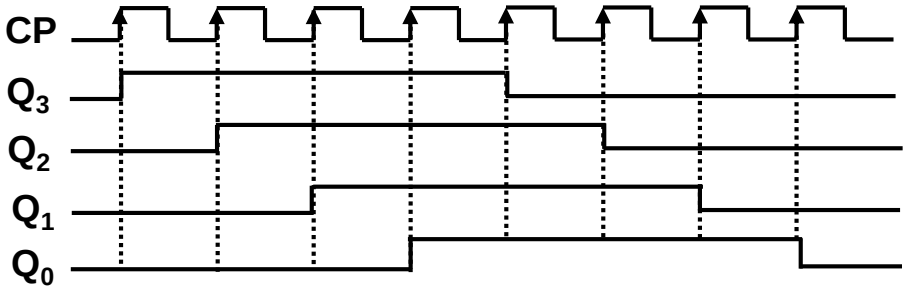
扭环形计数器

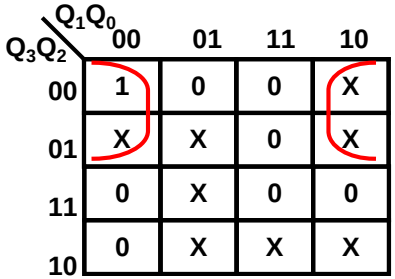
常用状态图



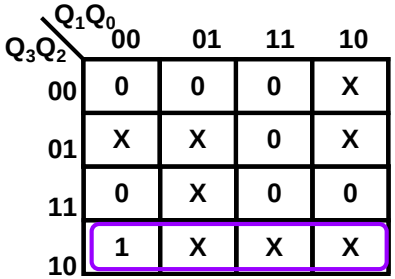
优点：①无险象
②后级每个译码门只需要2个输入端.
③模8计数器

输入				译码输出							
Q_3	Q_2	Q_1	Q_0	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0
1	1	0	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0
1	1	1	1	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	1





$Y_0 = \overline{Q}_3\overline{Q}_0$



$Y_1 = Q_3\overline{Q}_2$

环形、扭环形移位寄存器

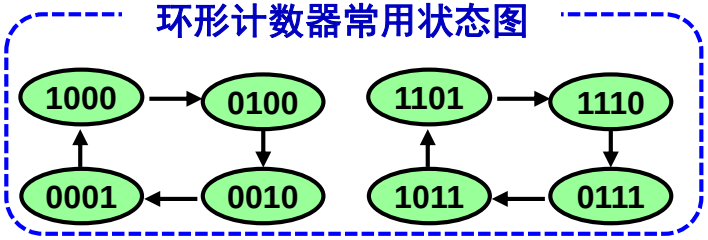
环形、扭环形计数器总结——

特点：在移位寄存器的基础上，增加反馈逻辑电路组成。

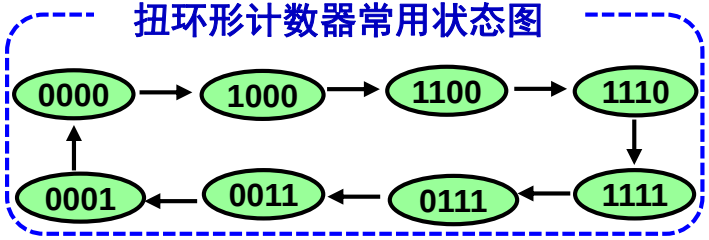
用途：

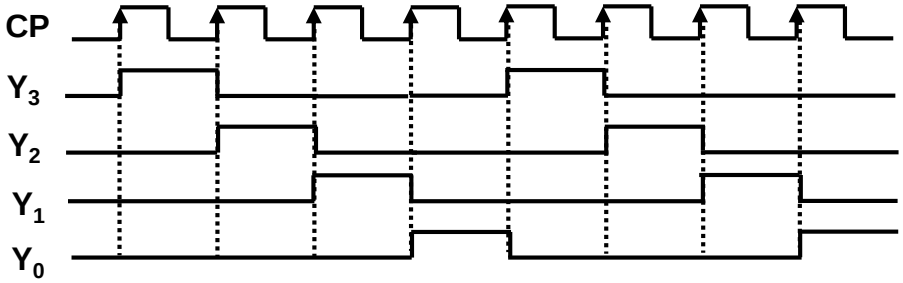
- 构成特殊编码的计数器（非二进制计数器）
- 环形计数器和扭环形计数器在计算机中可用于组成时序信号发生器（节拍发生器）

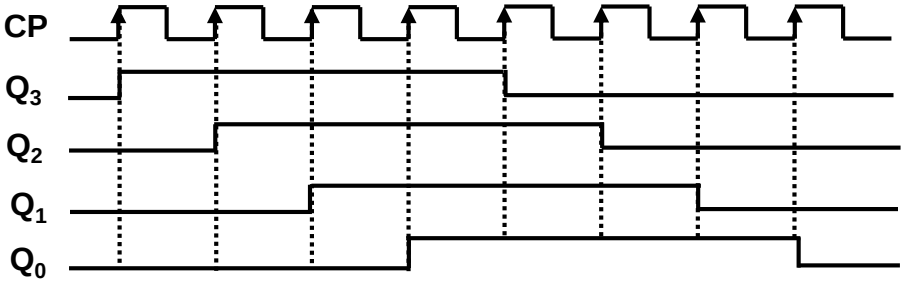
环形计数器常用状态图



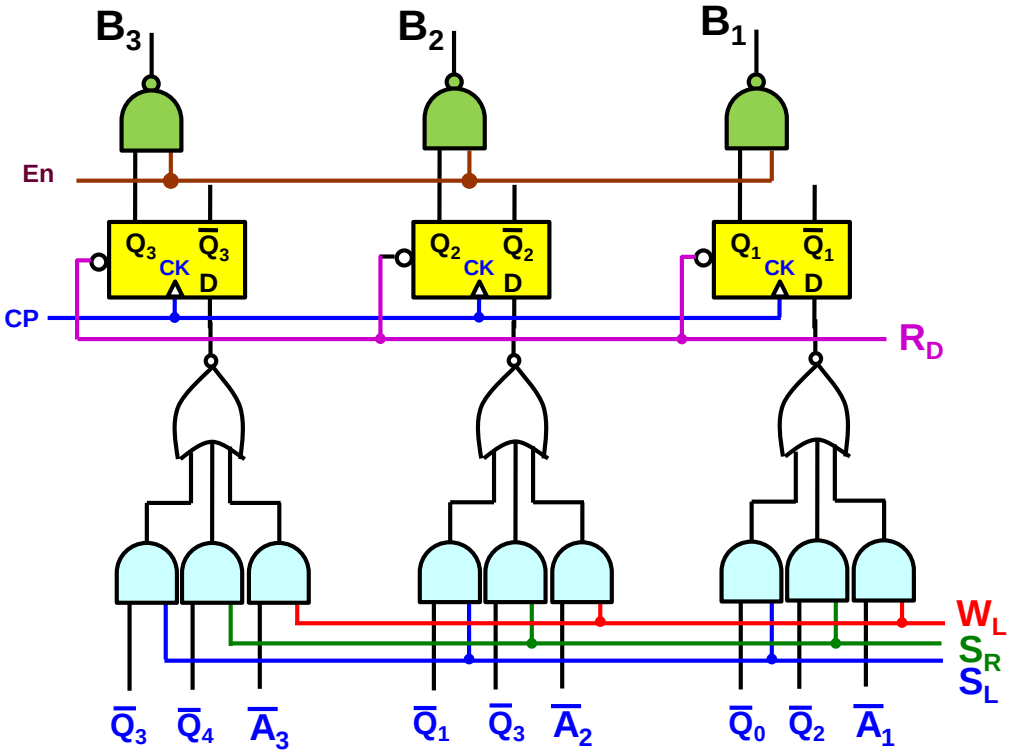
扭环形计数器常用状态图







双向移位寄存器



输入方程

$$\begin{cases} D_3 = \overline{\overline{A}_3 W_L + \overline{Q}_4 S_R + \overline{Q}_2 S_L} \\ D_2 = \overline{\overline{A}_2 W_L + \overline{Q}_3 S_R + \overline{Q}_1 S_L} \\ D_1 = \overline{\overline{A}_1 W_L + \overline{Q}_2 S_R + \overline{Q}_0 S_L} \end{cases}$$

输出方程

$$\begin{cases} B_3 = \overline{Q_3 E_n} \\ B_2 = \overline{Q_2 E_n} \\ B_1 = \overline{Q_1 E_n} \end{cases}$$

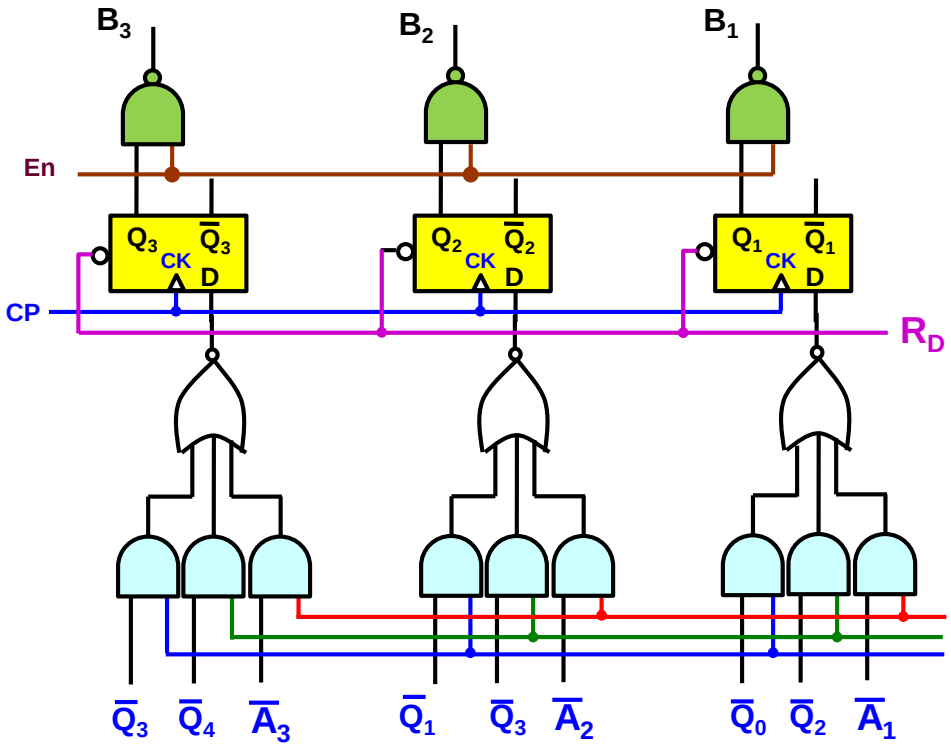
次态方程

$$\begin{cases} Q_3^{n+1} = D_3 \\ Q_2^{n+1} = D_2 \\ Q_1^{n+1} = D_1 \end{cases}$$

R_d——异步清零； W_L——写入使能

S_R——右移使能； S_L——左移使能

En——输出使能



功能——

（1）写入：将 $A_1 \sim A_3$ 存放在寄存器中

Let: $W_L = 1$, $S_R = S_L = 0$

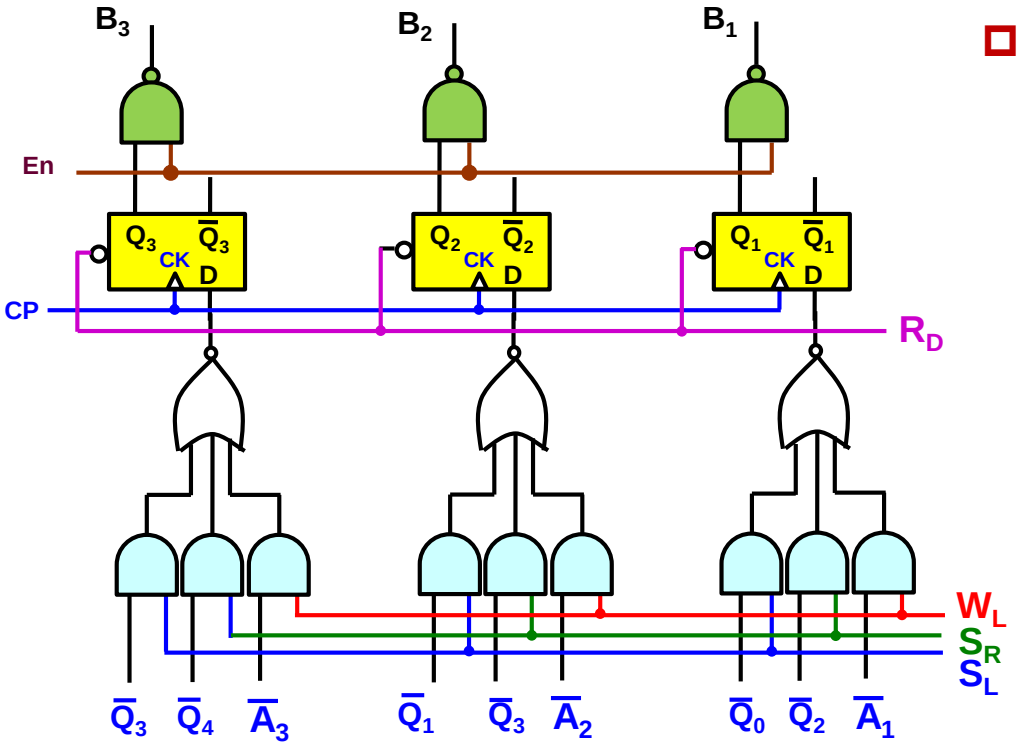
当 cp ↑ 上升沿到来时：

次态方程

$$\begin{cases} Q_3^{n+1} = D_3 = A_3 \\ Q_2^{n+1} = D_2 = A_2 \\ Q_1^{n+1} = D_1 = A_1 \end{cases}$$

输入方程

$$\begin{cases} D_3 = \overline{\bar{A}_3 W_L + \bar{Q}_4 S_R + \bar{Q}_2 S_L} = \overline{A_3 \cdot 1 + Q_4 \cdot 0 + Q_2 \cdot 0} = A_3 \\ D_2 = \overline{\bar{A}_2 W_L + \bar{Q}_3 S_R + \bar{Q}_1 S_L} = \overline{A_2 \cdot 1 + Q_3 \cdot 0 + Q_1 \cdot 0} = A_2 \\ D_1 = \overline{\bar{A}_1 W_L + \bar{Q}_2 S_R + \bar{Q}_0 S_L} = \overline{A_1 \cdot 1 + Q_2 \cdot 0 + Q_0 \cdot 0} = A_1 \end{cases}$$



功能——

（2）右移

Let: $S_R = 1$, $W_L = S_L = 0$

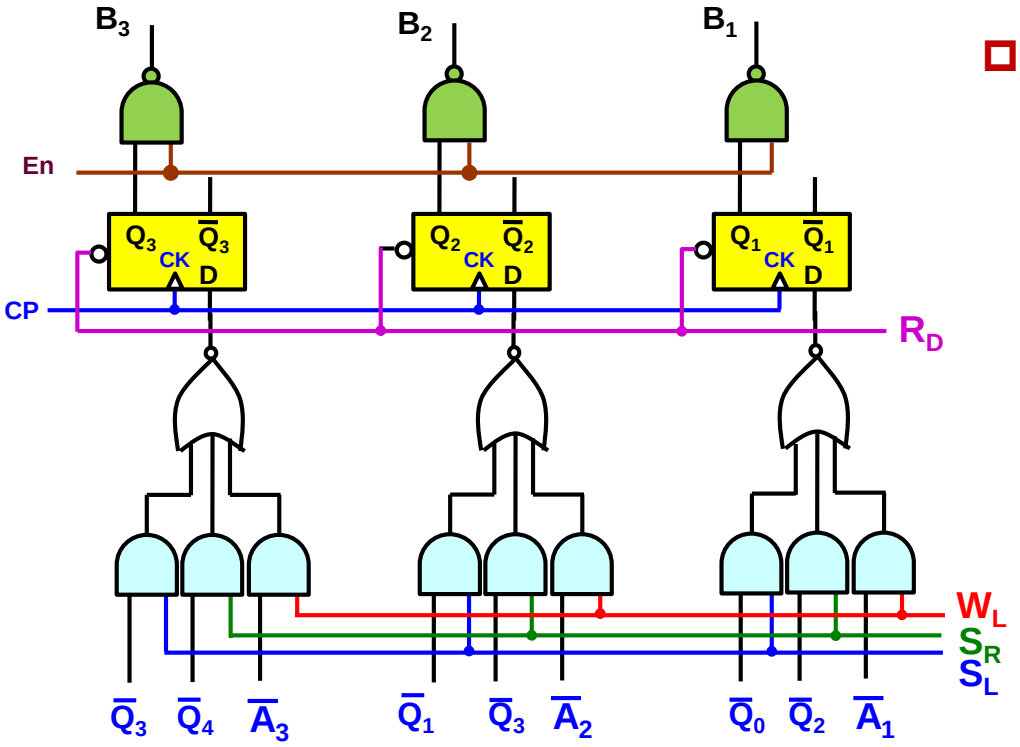
当 cp ↑ 上升沿到来时：

次态方程

$$\begin{cases} Q_3^{n+1} = D_3 = Q_4 \\ Q_2^{n+1} = D_2 = Q_3 \\ Q_1^{n+1} = D_1 = Q_2 \end{cases}$$

输入方程

$$\begin{cases} D_3 = \overline{\bar{A}_3 W_L + \bar{Q}_4 S_R + \bar{Q}_2 S_L} = \overline{A_3 \cdot 0 + Q_4 \cdot 1 + Q_2 \cdot 0} = Q_4 \\ D_2 = \overline{\bar{A}_2 W_L + \bar{Q}_3 S_R + \bar{Q}_1 S_L} = \overline{A_2 \cdot 0 + Q_3 \cdot 1 + Q_1 \cdot 0} = Q_3 \\ D_1 = \overline{\bar{A}_1 W_L + \bar{Q}_2 S_R + \bar{Q}_0 S_L} = \overline{A_1 \cdot 0 + Q_2 \cdot 1 + Q_0 \cdot 0} = Q_2 \end{cases}$$



功能——

（3）左移

Let: $S_L = 1$, $W_L = S_R = 0$

当 cp ↑ 上升沿到来时：

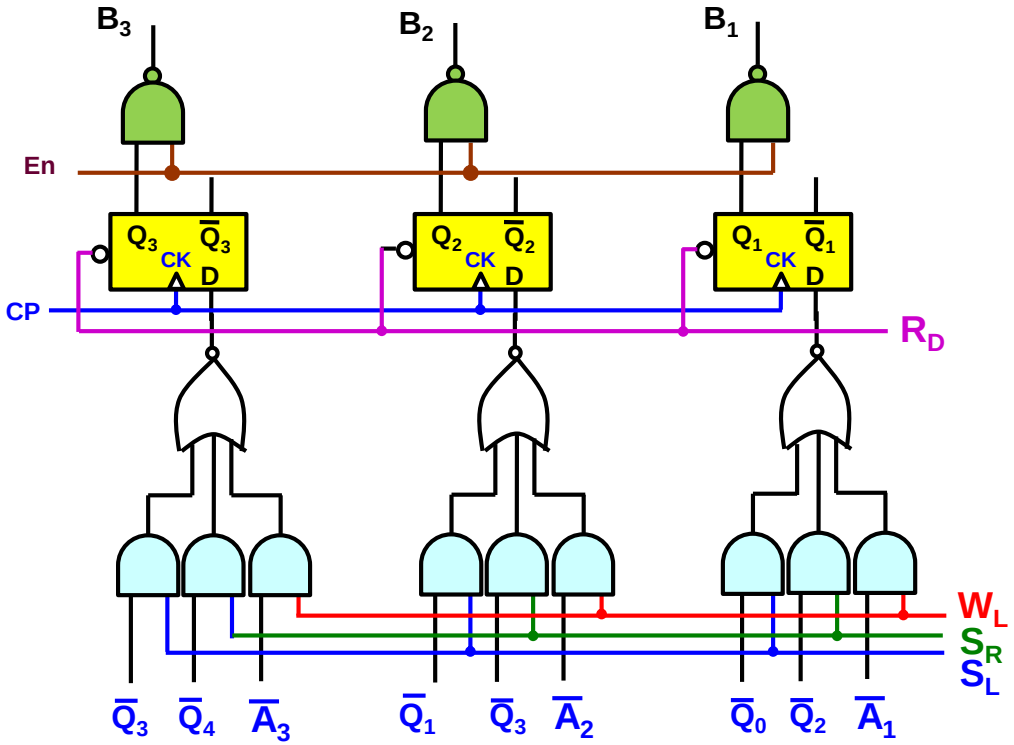
次态方程

$$\begin{cases} Q_3^{n+1} = D_3 = Q_2 \\ Q_2^{n+1} = D_2 = Q_1 \\ Q_1^{n+1} = D_1 = Q_0 \end{cases}$$

输入方程

$$\begin{cases} D_3 = \overline{\overline{A}_3 W_L + \overline{Q}_4 S_R + \overline{Q}_2 S_L} = \overline{A_3 \cdot 0 + Q_4 \cdot 0 + Q_2 \cdot 1} = Q_2 \\ D_2 = \overline{\overline{A}_2 W_L + \overline{Q}_3 S_R + \overline{Q}_1 S_L} = \overline{A_2 \cdot 0 + Q_3 \cdot 0 + Q_1 \cdot 1} = Q_1 \\ D_1 = \overline{\overline{A}_1 W_L + \overline{Q}_2 S_R + \overline{Q}_0 S_L} = \overline{A_1 \cdot 0 + Q_2 \cdot 0 + Q_0 \cdot 1} = Q_0 \end{cases}$$

双向移位寄存器



功能——

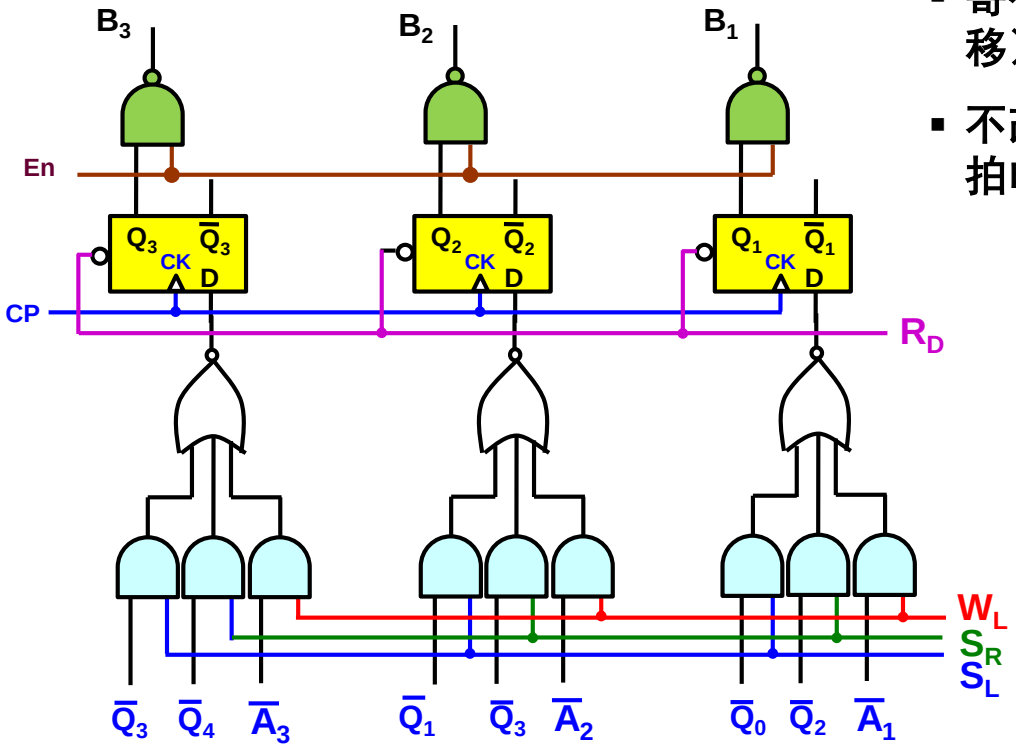
（4）读出

Let：$E_n=1$

输出方程

$$\begin{cases} B_3 = \overline{Q_3 E_n} = \overline{Q}_3 \\ B_2 = \overline{Q_2 E_n} = \overline{Q}_2 \\ B_1 = \overline{Q_1 E_n} = \overline{Q}_1 \end{cases}$$

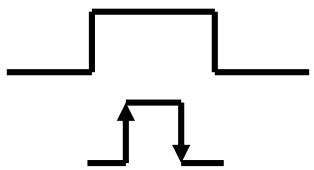
双向移位寄存器



- 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。
- 不改变触发器状态的操作（读出），只需要节拍电位。

必须保证节拍脉冲的边沿被节拍电位的有效电平完全覆盖

节拍：一种控制信号



节拍电位

节拍脉冲

节拍

例如：

- 写入操作，需要 $W_L = 1$，同时CP ↑
- 左移操作，需要 $S_L = 1$，同时CP ↑
- 读出操作，只需要 En=1

移位寄存器

寄存器总结

- 主要功能：存放二进制数据（存储的二进制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

用Verilog实现移位寄存器

- **串入并出8位移位寄存器**

```
module Vr8bitSRparout ( CLK, CLR, SERIN, Q );
    input CLK, CLR, SERIN;
    output reg [WID-1:0] Q;
    parameter WID = 8;

    always @ (posedge CLK)
        if (CLR == 1) Q <= 0;        // 同步清零
        else  Q <= {Q[WID-2:0], SERIN};  // 移位
endmodule
```

用Verilog实现移位寄存器

- **通用4位移位寄存器**

```
module Vrshrg4u ( CLK, CLR, RIN, LIN, S0, S1, A, B, C, D, QA, QB, QC, QD );
    input CLK, CLR, S0, S1, RIN, LIN, A, B, C, D;
    output reg QA, QB, QC, QD;

    always @ (posedge CLK)
        if (CLR == 1'b1) {QA,QB,QC,QD} <= 4'b0;
        else case ({S1,S0})
            2‘b00: ;                          // 保持
            2‘b01: {QA,QB,QC,QD} <= {RIN,QA,QB,QC};   // 右移
            2‘b10: {QA,QB,QC,QD} <= {QB,QC,QD,LIN};   // 左移
            2‘b11: {QA,QB,QC,QD} <= {A,B,C,D};        // 载入
        default: {QA,QB,QC,QD} <= 4‘bx;               // 不会发生
    endcase
endmodule
```

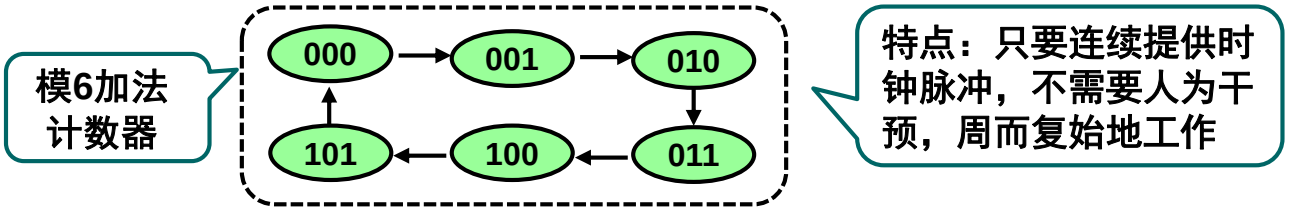
Unit 9 寄存器和计数器

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

典型时序逻辑部件——计数器

计数器

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

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；

(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）

③ 列状态方程（次态方程）

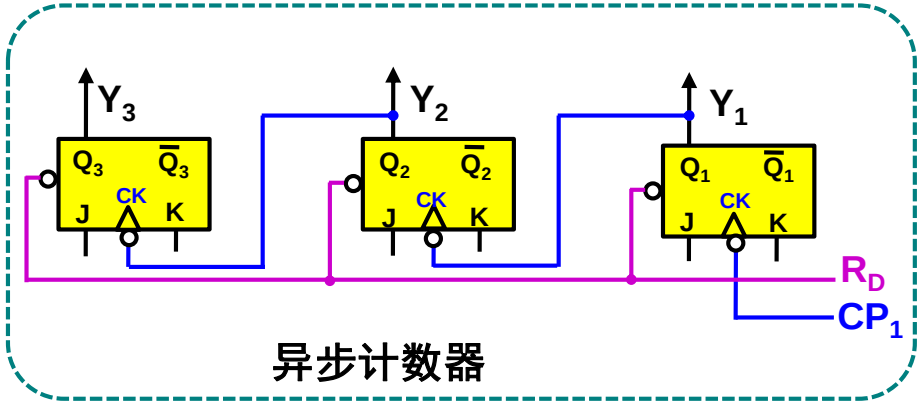
④ 列写状态转换表

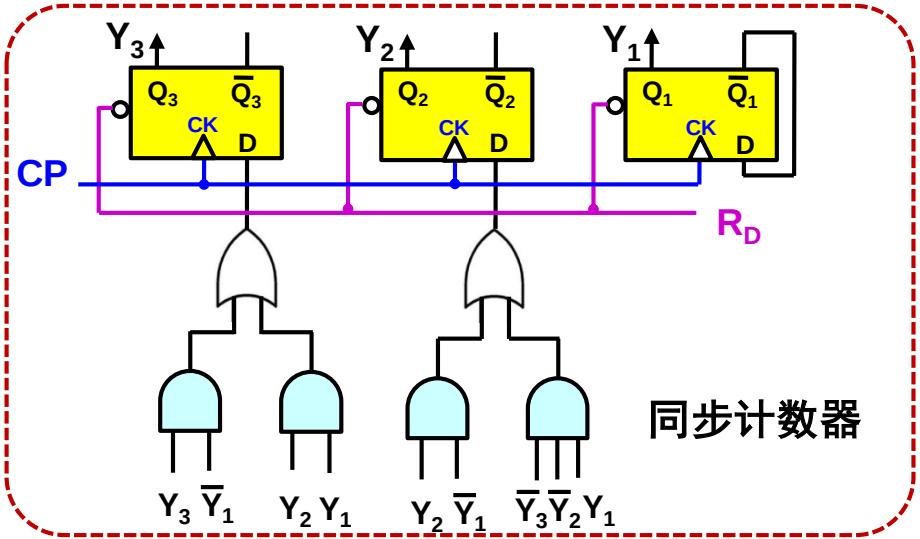
⑤ 画出状态图

⑥ 画出波形图（如必要）

同步计数器

- 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。



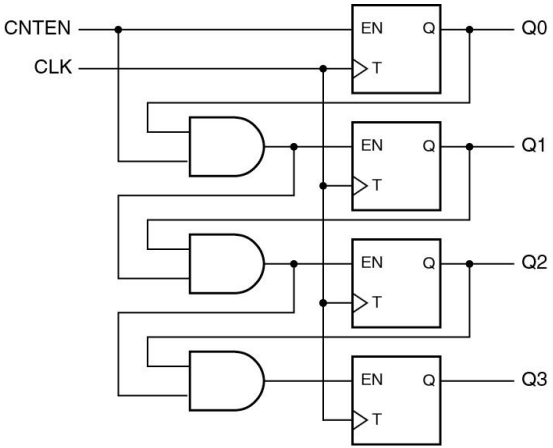


同步计数器

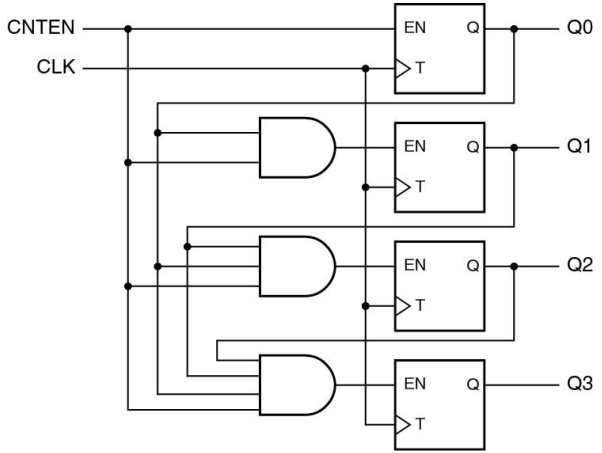
■ 同步计数器——

- 触发器共用时钟信号CLK
- 使能端EN有效时，触发器输出在T信号上升沿反转
- 主计数器使能信号CNTEN

  CNTEN信号有效且所有低阶计数位为1时，每个计数器都翻转。

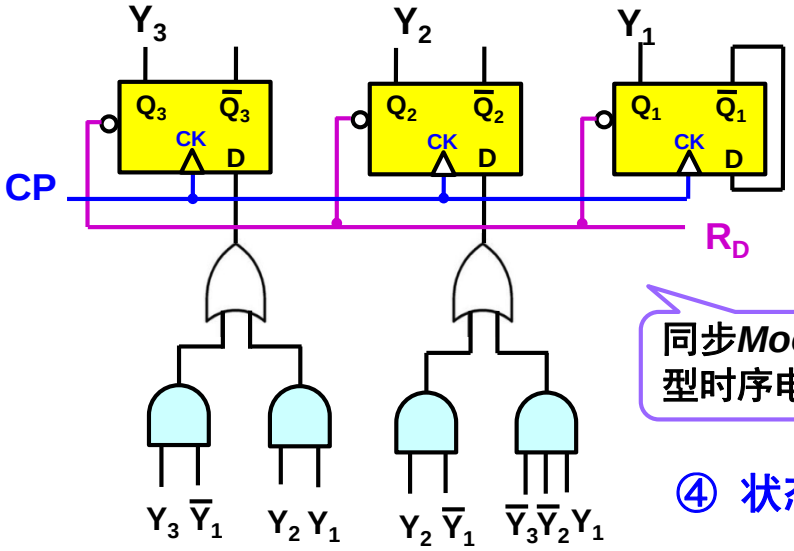


同步串行4位二进制计数器



并行同步4位二进制计数器

同步计数器的时序电路分析



同步*Moor*型时序电路

① 输入方程

$$D_3 = Y_3\overline{Y}_1 + Y_2Y_1$$

$$D_2 = Y_2\overline{Y}_1 + \overline{Y}_3\overline{Y}_2Y_1$$

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

② 次态方程

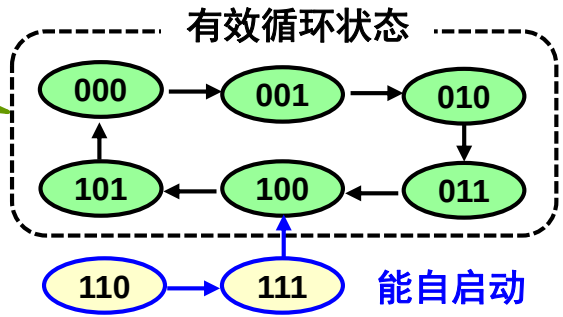
$$\begin{cases} Y_1^{n+1} = D_1 \\ Y_2^{n+1} = D_2 \\ Y_3^{n+1} = D_3 \end{cases}$$

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	↑

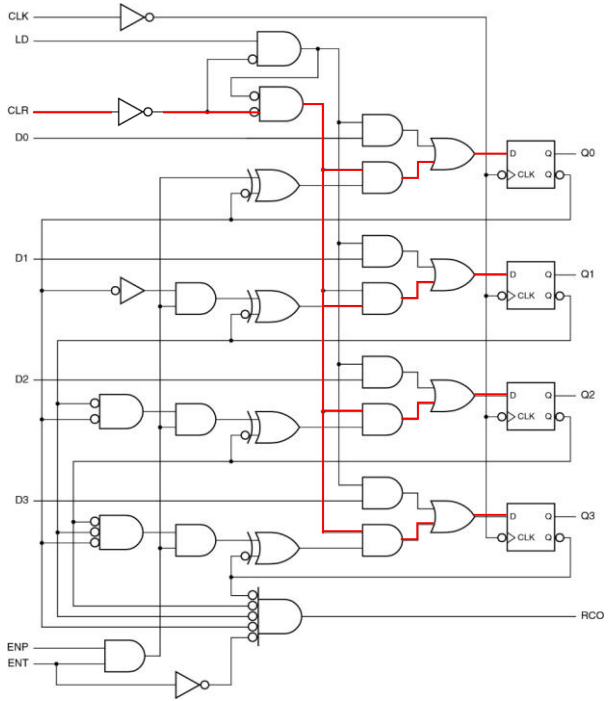
④ 状态图

同步模6加法计数器



同步计数器（带有同步载入端和清零端）

- **4位二进制计数器电路的设计——**
  - 带有同步载入端和清零端——D触发器
  - 2输入多路复用器驱动D触发器的输入
    - CLR有效时，多路复用器输出为0
    - 输入信号LD有效，将输入数据传送到输出端



同步计数器（带有同步载入端和清零端）

- **4位二进制计数器电路的状态表**

输入				输出				次态			
CLR	LD	ENT	ENP	Q3	Q2	Q1	Q0	Q3*	Q2*	Q1*	Q0*
1	x	x	x	x	x	x	x	0	0	0	0
0	1	x	x	x	x	x	x	D3	D2	D1	D0
0	0	0	x	x	x	x	x	Q3	Q2	Q1	Q0
0	0	x	0	x	x	x	x	Q3	Q2	Q1	Q0
0	0	1	1	0	0	0	0	0	0	0	1
0	0	1	1	0	0	0	1	0	0	1	0
…											
0	0	1	1	1	1	0	1	1	1	1	0
0	0	1	1	1	1	1	0	1	1	1	1
0	0	1	1	1	1	1	1	0	0	0	0

用Verilog实现带有同步载入端和清零端计数器

- **4位通用二进制计数器**

```
module Vrcntr4u ( CLK, CLR, LD, ENP, ENT, D, Q, RCO );
   input CLK, CLR, LD, ENP, ENT;
   input [3:0] D;
   output reg [3:0] Q;
   output reg RCO;

   always @ (posedge CLK)  // 创建f-f计数器的特性
      if (CLR)                    Q <= 4'd0;
      else if (LD)                Q <= D;
      else if (ENT && ENP)    Q <= Q + 1;
      //    else                  Q <= Q;

   always @ (Q or ENT)     // 创建组合输出RCO
      if (ENT && (Q == 4'd15))   RCO = 1;
      else                       RCO = 0;
endmodule
```

用Verilog实现带有同步载入端和清零端计数器

- 修改上述程序，实现十进制计数

```
module Vrcntr4udec ( CLK, CLR, LD, ENP, ENT, D, Q, RCO );
   input CLK, CLR, LD, ENP, ENT;
   input [3:0] D;
   output reg [3:0] Q;
   output reg RCO;

   always @ (posedge CLK)  // 创建f-f计数器的特性
      if (CLR)              Q <= 4'd0;
      else if (LD)          Q <= D;
      else if (ENT && ENP && (Q == 4'd9))  Q <= 4'd0;
      else if (ENT && ENP)    Q <= Q + 1;
      else                  Q <= Q;

   always @ (Q or ENT)     // 创建组合输出RCO
      if (ENT && (Q == 4'd9))    RCO = 1;
      else                       RCO = 0;
endmodule
```

用Verilog实现带有同步载入端和清零端计数器

- 修改上述程序，实现余三十进制计数

```
module Vrexcess3 ( CLK, CLR, LD, ENP, ENT, D, Q, RCO );
   input CLK, CLR, LD, ENP, ENT;
   input [3:0] D;
   output reg [3:0] Q;
   output reg RCO;

   always @ (posedge CLK)  // 创建f-f计数器的特性
      if (CLR)                          Q <= 4'd3;
      else if (LD)                      Q <= D;
      else if (ENT && ENP && (Q == 4'd12))  Q <= 4'd3;
      else if (ENT && ENP)              Q <= Q + 1;
      else                              Q <= Q;

   always @ (Q or ENT)     // 创建组合输出RCO
      if (ENT && (Q == 4'd12))    RCO = 1;
      else                        RCO = 0;
endmodule
```

用Verilog实现带有同步载入端和清零端计数器

- 实现4位**递增/递减**的计数器

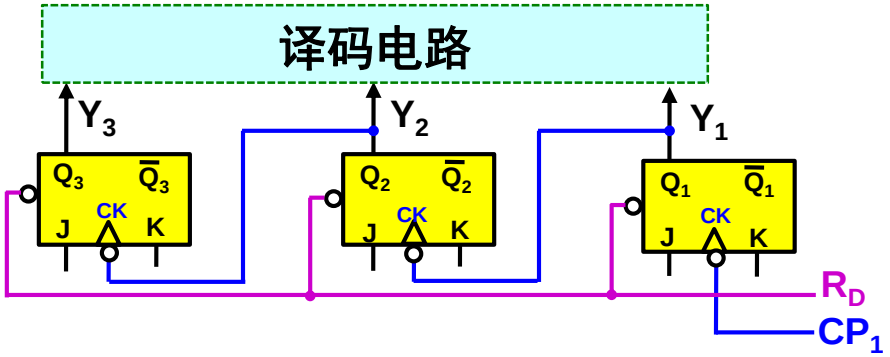
```
module Vrupdn4 ( CLK, CLR, LD, ENP, ENT, UPDN, D, Q, RCO );
   input CLK, CLR, LD, ENP, ENT, UPDN;
   input [3:0] D;
   output reg [3:0] Q;
   output reg RCO;

   always @ (posedge CLK)     //创建f-f计数器的特性
      if (CLR )               Q <= 4'b0;
      else if (LD)            Q <= D;
      else if (ENT && ENP &&  UPDN)          Q <= Q + 1;
      else if (ENT && ENP && !UPDN)          Q <= Q - 1;
      else                    Q <= Q;

   always @ (Q or ENT or UPDN) // 创建组合输出RCO
      if     (ENT &&  UPDN && (Q == 4'd15))    RCO = 1;
      else if (ENT && !UPDN && (Q == 4'd0 ))  RCO = 1;
      else                                    RCO = 0;
endmodule
```

异步计数器

- 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。



行波计数器

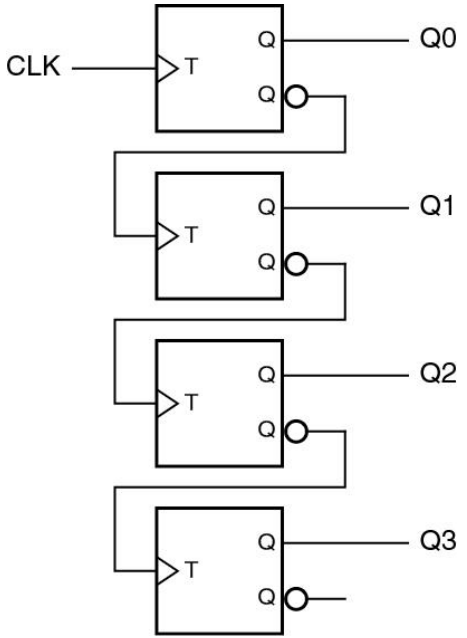
■ 行波计数器——

➢只用n个触发器即可实现n位二进制计数器

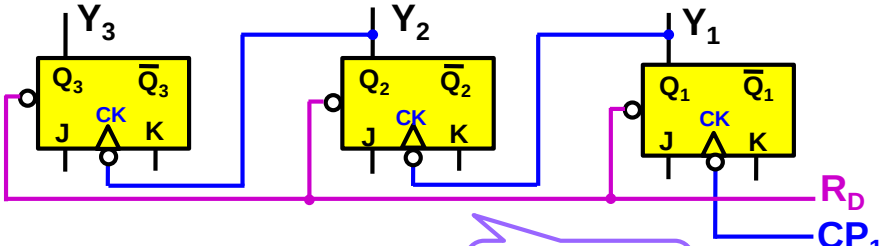
➢T触发器每个时钟上升沿都会产生反转

利用此特性，模拟二进制计数器的进位

➢用触发器来驱动其他触发器的时钟输入端，一般都会存在问题。由于每个寄存器都存在传播延迟，且每一级触发器时钟的延迟会使下一级触发器的输入时钟产生偏移，会造成累积延迟



异步计数器



异步*Moor*型时序电路

① 输入方程

$J_1 = K_1 = 1$　　$CP_1 \downarrow$

$J_2 = K_2 = 1$　　$CP_2 = Y_1 \downarrow$

$J_3 = K_3 = 1$　　$CP_3 = Y_2 \downarrow$

② 次态方程

$$Y_1^{n+1} = J_1\overline{Q}_1 + \overline{K}_1 Q_1 = \overline{Y}_1 \qquad CP_1 \downarrow$$

$$Y_2^{n+1} = J_2\overline{Q}_2 + \overline{K}_2 Q_2 = \overline{Y}_2 \qquad Y_1 \downarrow$$

$$Y_3^{n+1} = J_3\overline{Q}_3 + \overline{K}_3 Q_3 = \overline{Y}_3 \qquad Y_2 \downarrow$$

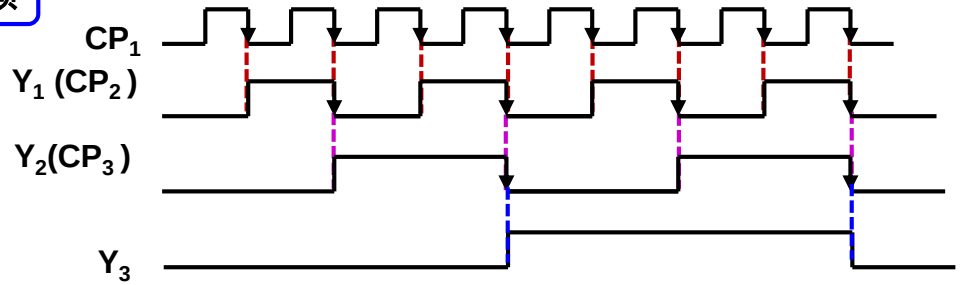
③ 状态转换表

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

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）

典型时序逻辑部件——节拍发生器

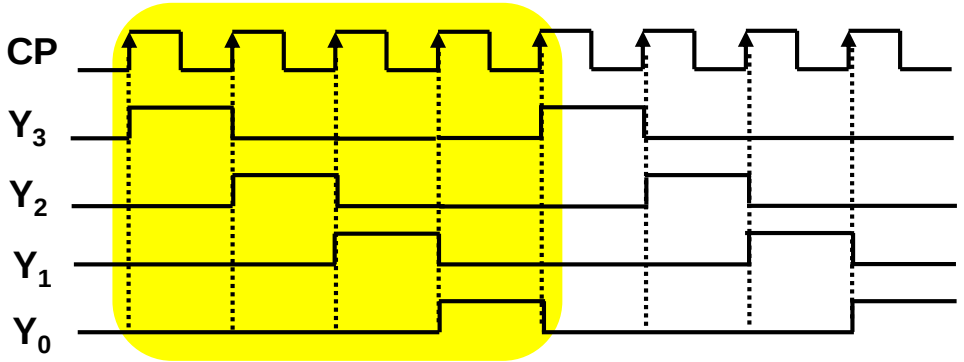
- 节拍发生器（顺序脉冲发生器）——

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。

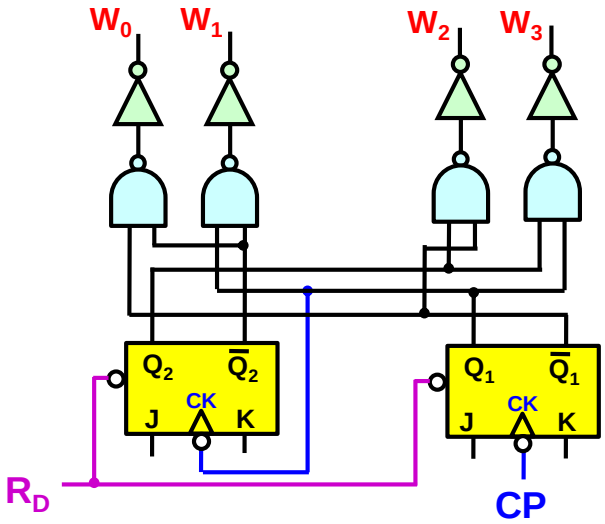
①执行写入操作：写入使能有效（存入1000）

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）

节拍发生器1



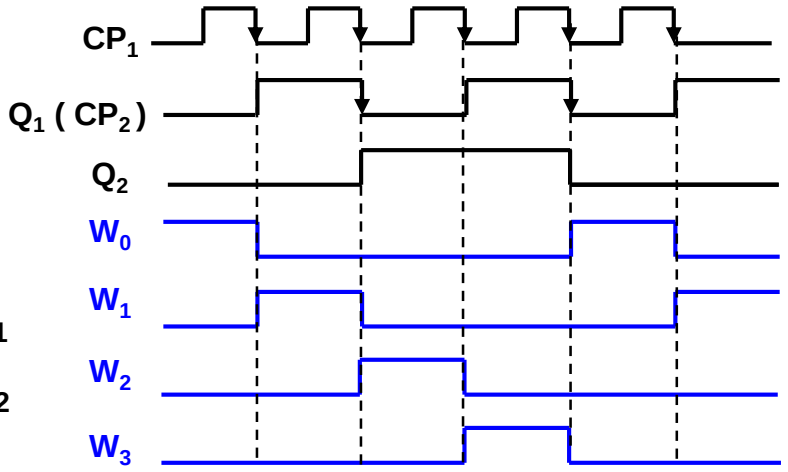
③ 输出方程

$$\begin{cases} W_0 = \overline{Q}_2\,\overline{Q}_1 \\ W_1 = \overline{Q}_2\,Q_1 \\ W_2 = Q_2\,\overline{Q}_1 \\ W_3 = Q_2\,Q_1 \end{cases}$$

④ 状态转换表

现态		次态		时钟	
Q_2^n	Q_1^n	Q_2^{n+1}	Q_1^{n+1}	CP_2	CP_1
0	0	0	1	无	↓
0	1	1	0	↓	↓
1	0	1	1	无	↓
1	1	0	0	↓	↓

结论：4-节拍发生器（$W_0 \sim W_3$）



① 输入方程

$J_1 = K_1 = 1$, CP_1 ↓

$J_2 = K_2 = 1$, $CP_2 = Q_1$ ↓

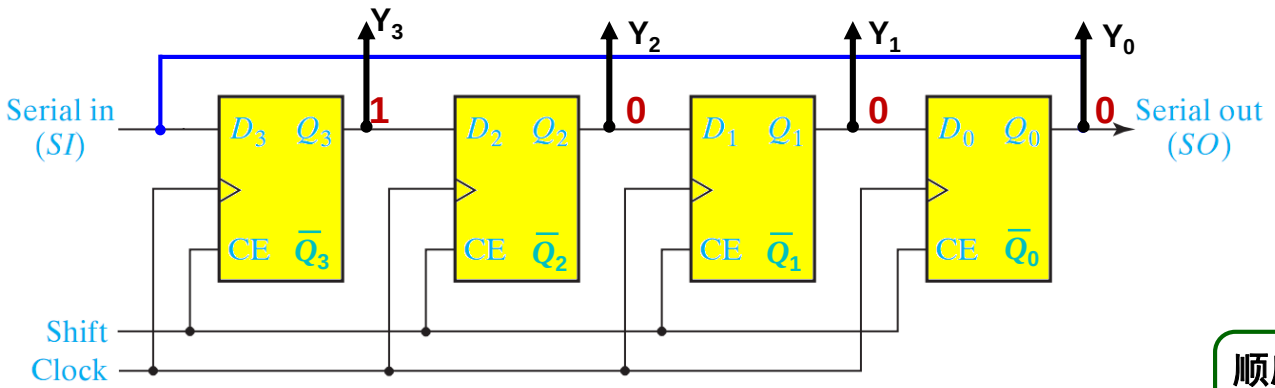
② 次态方程

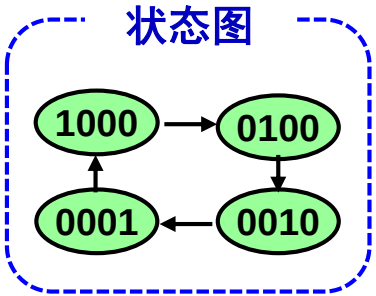
$$Q_1^{n+1} = J_1\overline{Q}_1 + K_1\overline{Q}_1 = \overline{Q}_1$$

$$Q_2^{n+1} = J_2\overline{Q}_2 + K_2\overline{Q}_2 = \overline{Q}_2$$

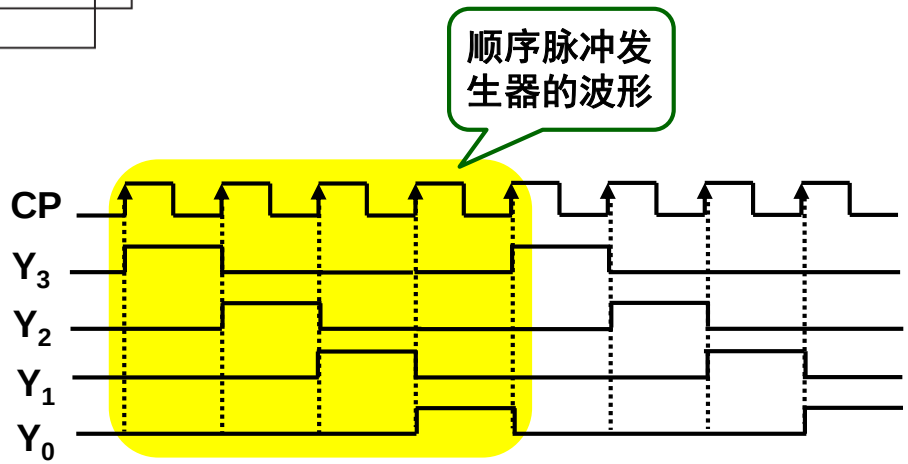
节拍发生器1

回顾：环形计数器

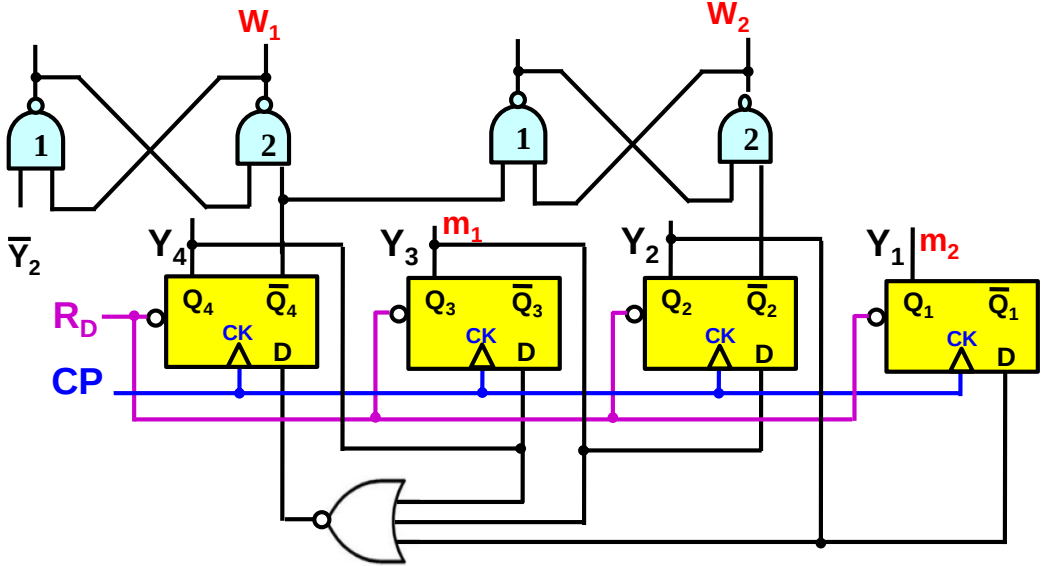


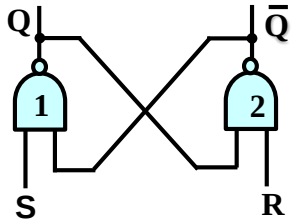


- 应用：电话响铃控制
- 用4位顺序脉冲发生器的某一个输出作为响铃控制信号，若时钟CP周期为1秒，电话铃声就是响1秒停3秒的节奏。

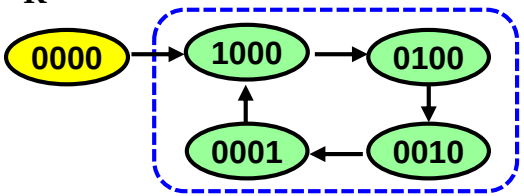


节拍发生器2





④ 状态图



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
0	0	0	0	1	0	0	0	↑
1	0	0	0	0	1	0	0	↑
0	1	0	0	0	0	1	0	↑
0	0	1	0	0	0	0	1	↑
0	0	0	1	1	0	0	0	↑

① 输入方程

$$\begin{cases} D_1 = Y_2 \\ D_2 = Y_3 \\ D_3 = Y_4 \\ D_4 = \overline{Y_4+Y_3+Y_2} \end{cases}$$

② 次态方程

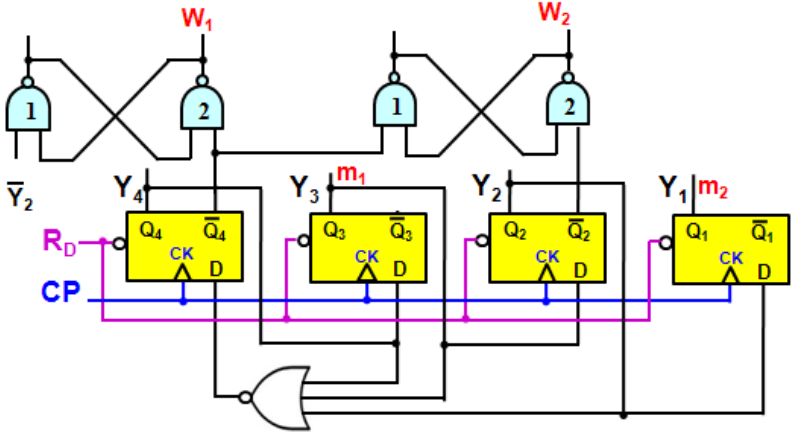
$$Y_1^{n+1} = Y_2$$

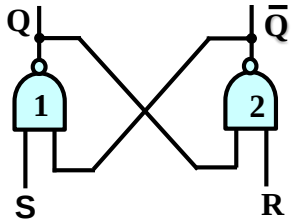
$$Y_2^{n+1} = Y_3$$

$$Y_3^{n+1} = Y_4$$

$$Y_4^{n+1} = \overline{Y_4 + Y_3 + Y_2}$$

节拍发生器2





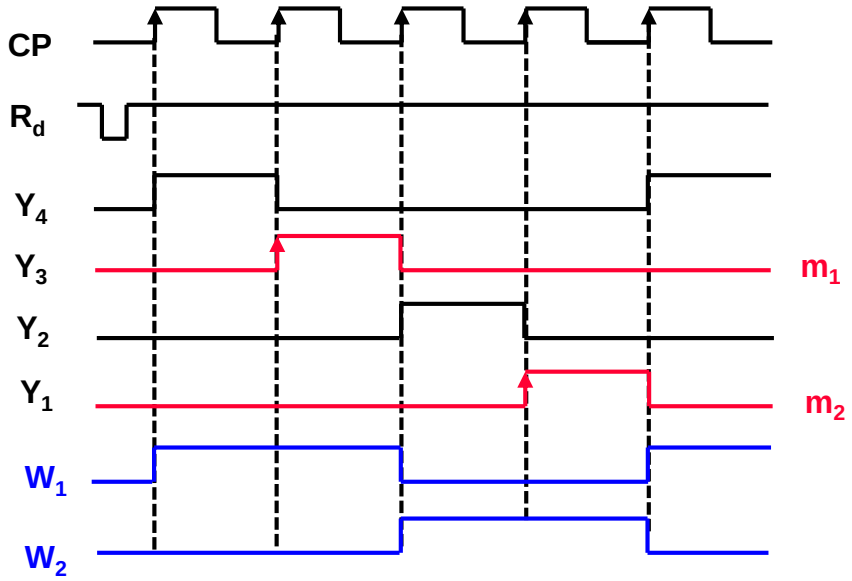
结论：2-节拍发生器

- W_1_m_1：节拍电位_节拍脉冲
- W_2_m_2：节拍电位_节拍脉冲

⑤ 确定输出

R	S	Q_{n+1}	$\overline{Q}_{n+1}$
$\overline{Y}_4$	$\overline{Y}_2$	($W_1=\overline{Q}$)	
1	1	Q_n	$\overline{Q}_n$
0	1	0	1
1	0	1	0
0	0	_	_

R	S	Q_{n+1}	$\overline{Q}_{n+1}$
$\overline{Y}_2$	$\overline{Y}_4$	($W_2=\overline{Q}$)	
1	1	Q_n	$\overline{Q}_n$
0	1	0	1
1	0	1	0
0	0	_	_



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）