



# Ch8 Sequential Logic Design Practices

## Main contents:

(1) Timing diagram时序图

(2) Registers寄存器

(3) Counters计数器 (重点)

以74x163为代表的电路分析与应用，包含：电路功能分析；任意模m计数器设计；序列发生器；控制各种不同器件

(4) Shift registers移位计数器 (重点)

以74x194为代表的电路分析与应用，包含：

8.5.1 Shift-Register Structure    8.5.4 Ring Counters

8.5.2 MSI Shift Registers        8.5.5 Johnson Counters

8.5.3 Shift-Register Counters

8.5.6 Linear Feedback Shift-Register Counters(LFSR)



# 8.1 Typical Synchronous Sequential Logic Circuit Timing Diagrams

## 正确理解现态和次态

### (1) 现态与次态的基本概念

在同步时序电路中，状态存储元件由时钟控制触发器组成，时钟脉冲信号加在每个触发器的时钟输入端，只有当时钟信号到来时，电路的**当前状态**才会发生改变，而且只改变一次。如果时钟触发沿没有到来，即使输入发生改变，电路的状态也不会变化。通常将时钟**触发沿到来之前**的电路状态称为**现态**，记为**S**；将时钟**触发沿到来之后**的电路状态称为**次态**，记为 **$S^{n+1}$ (或 **$S^*$** )**。在这里，时钟信号起到**同步**的作用。

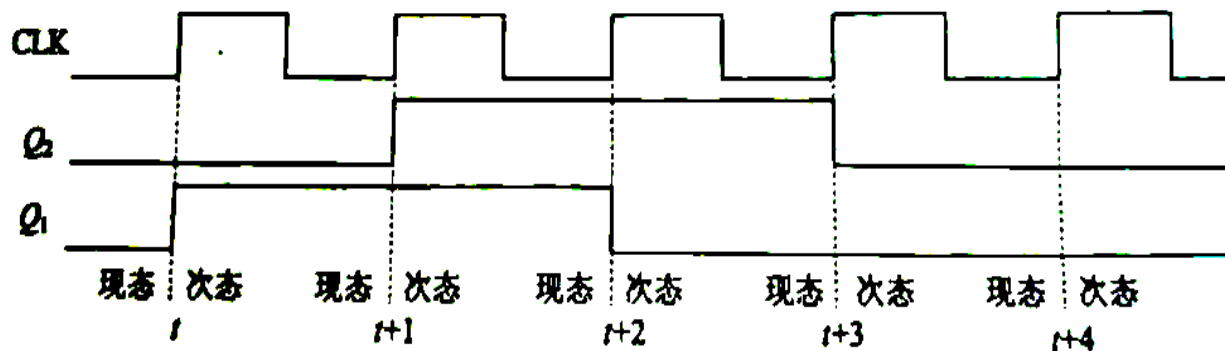


# 8.1 Typical Synchronous Sequential Logic Circuit Timing Diagrams

## 正确理解现态和次态

### (2) 现态与次态的时序关系

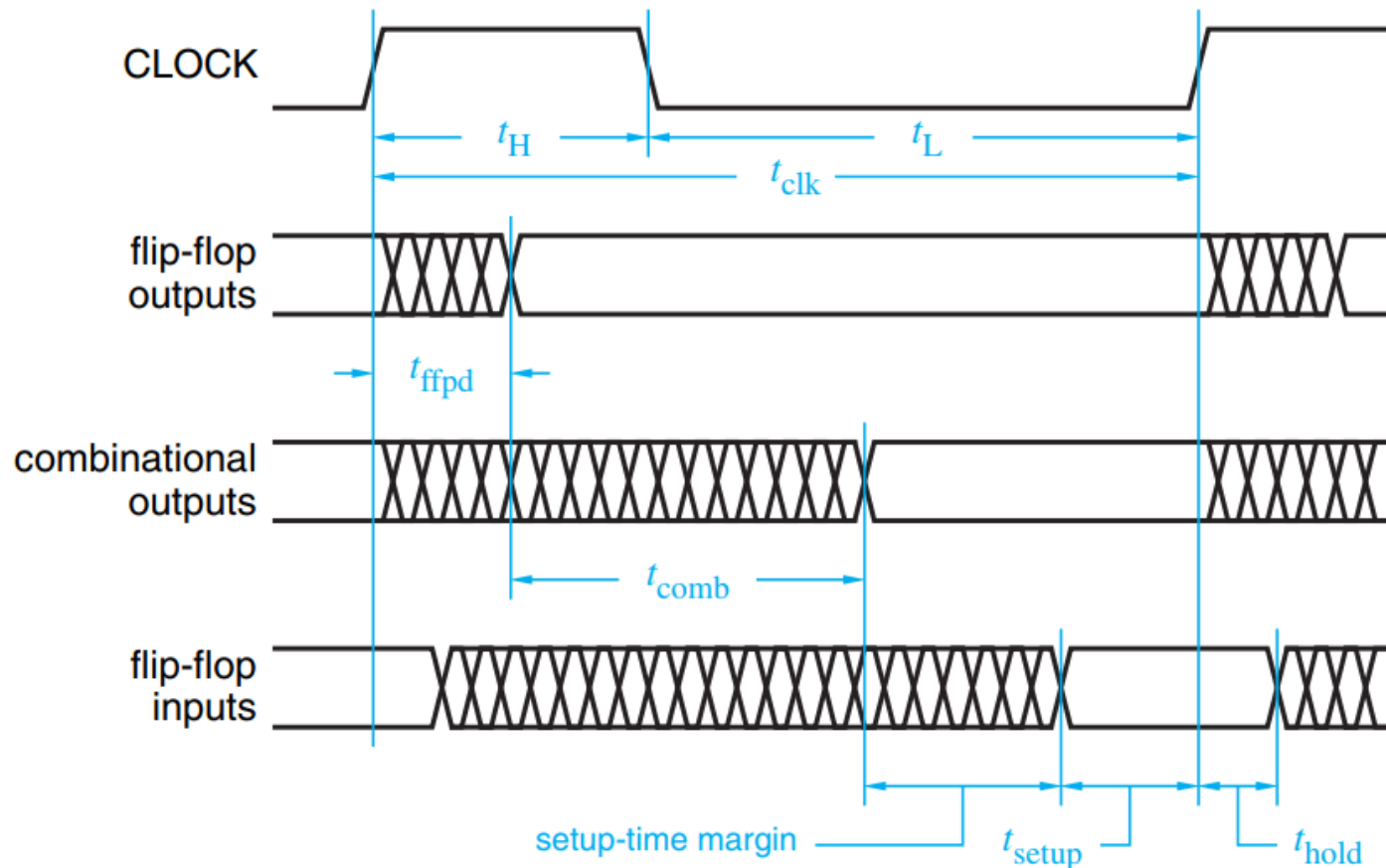
在下图所示的时序图中，状态变量 $Q_1$ 和 $Q_2$ 仅在时钟的**上升沿**到来时变化，这两个状态变量有4种组合形式，可表示4种状态： $Q_2Q_1=00, 01, 10, 11$ 。每一种状态在某个时钟触发沿之前称为**现态**，并在该时钟触发沿之后进入**次态**。比如，在时刻 $t$ 之前，现态为 $Q_2Q_1=00$ ，次态为01；在时刻 $t$ ，状态发生改变，而后现态变为01，并一直保持，直到 $t+1$ 时刻状态变为11；……因此，现态和次态是相对于时钟触发沿而言的，随着时刻的不同，次态可以变为现态。



现态与次态之间的相对关系



# 8.1 Typical Synchronous Sequential Logic Circuit Timing Diagrams



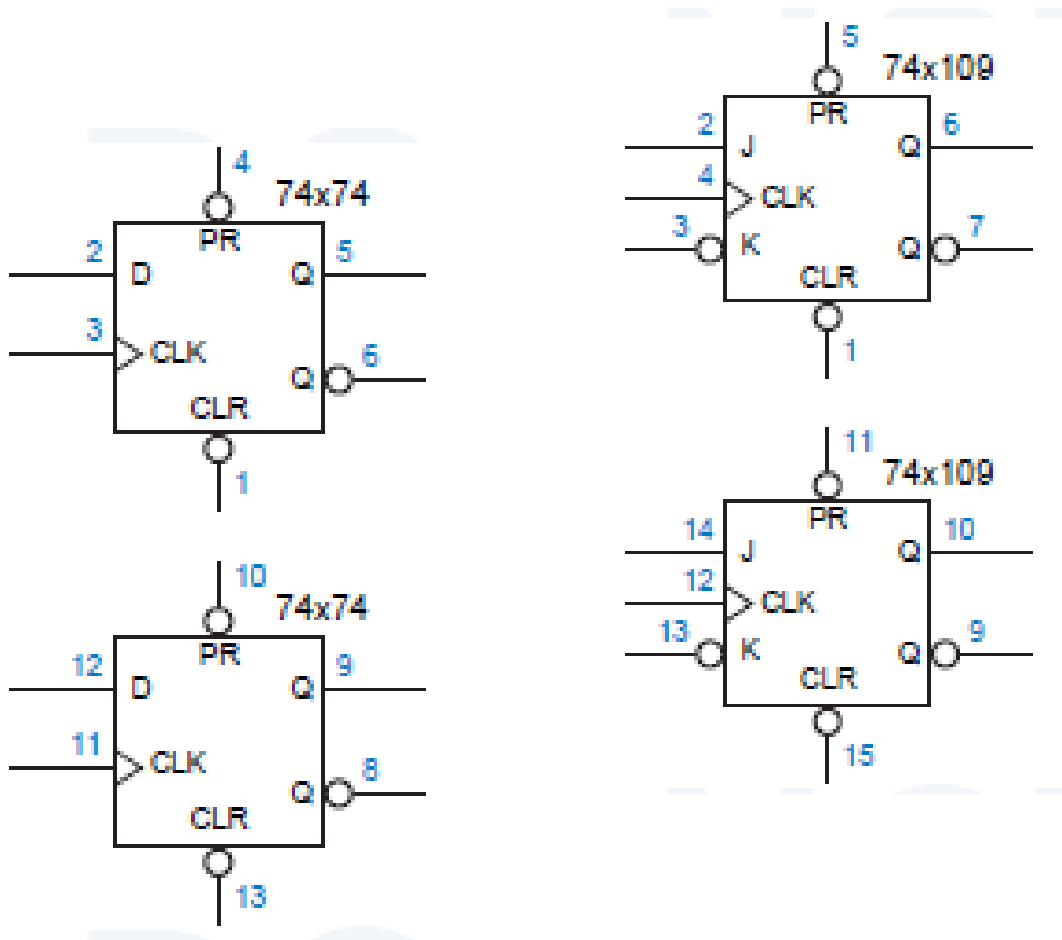
典型同步时序电路的传输延迟、建立和保持时间等的详细说明



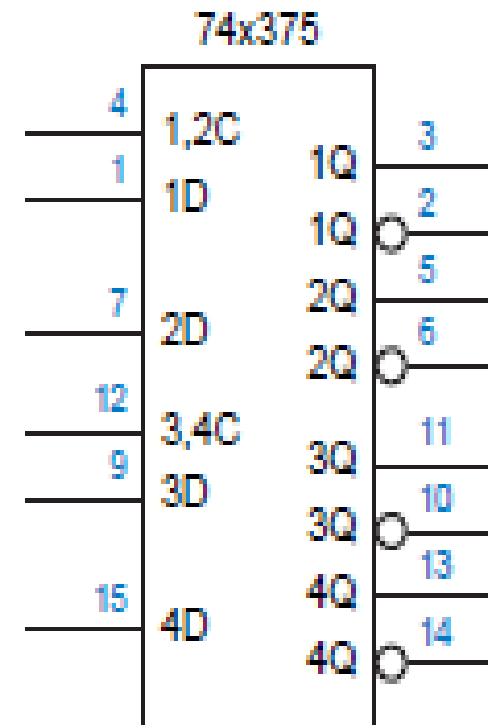
## 8.2 Latches and Flip-flops

### 8.2.1 SSI latches and flip-flops

2个独立带异步置位和清零的正边沿触发JK'-FF



2个独立带异步置位和清零的正边沿触发D-FF



4个D-FF构成的锁存器



## ☀ 8.2.5 Multibit registers and latches

**---Register: a collection of two or more D flip-flops  
with a common clock input**

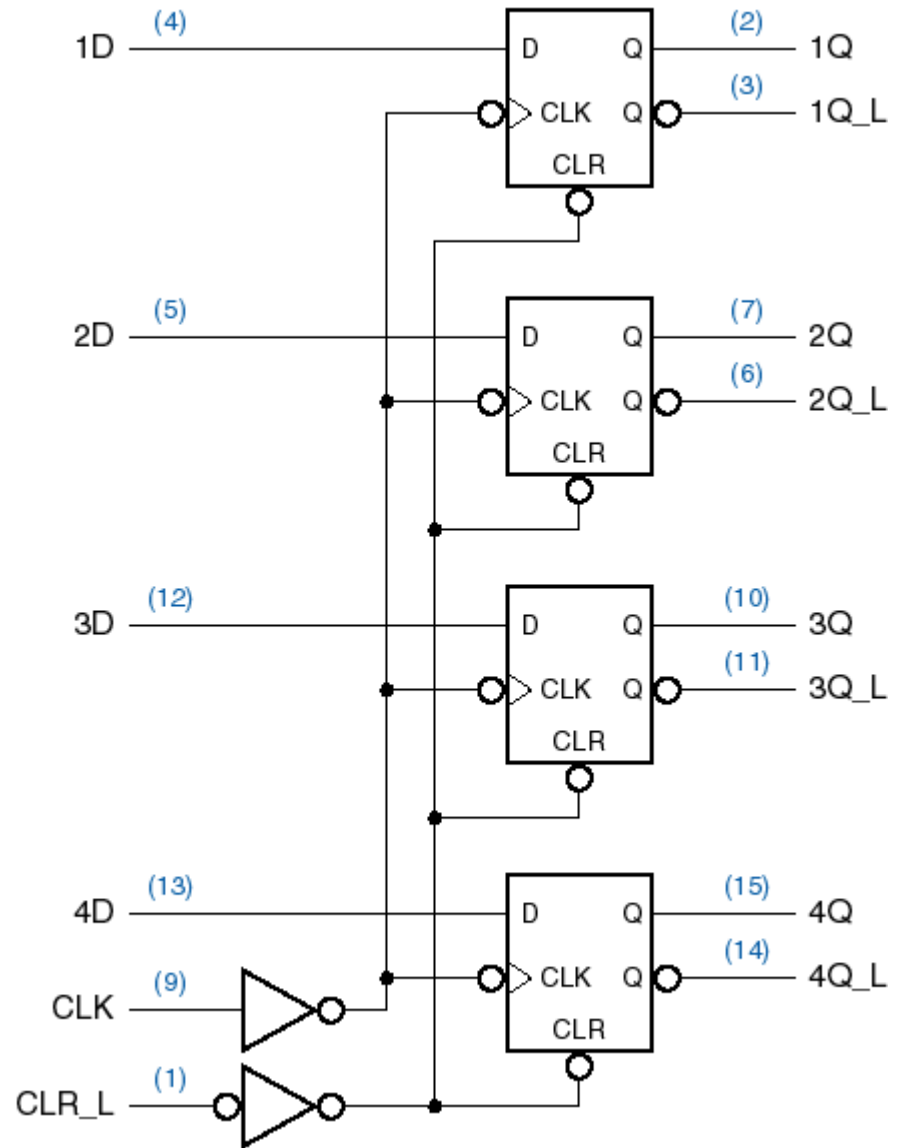
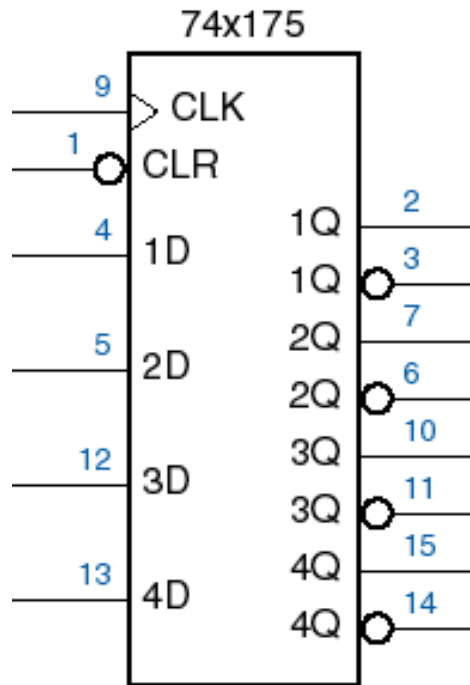
**---Register vs. latch, what's the difference**

- ☀ **Register: edge-triggered behavior**

- ☀ **Latch: output follows inputs when enable  
input is asserted**



# 1) 4-bit register 74x175

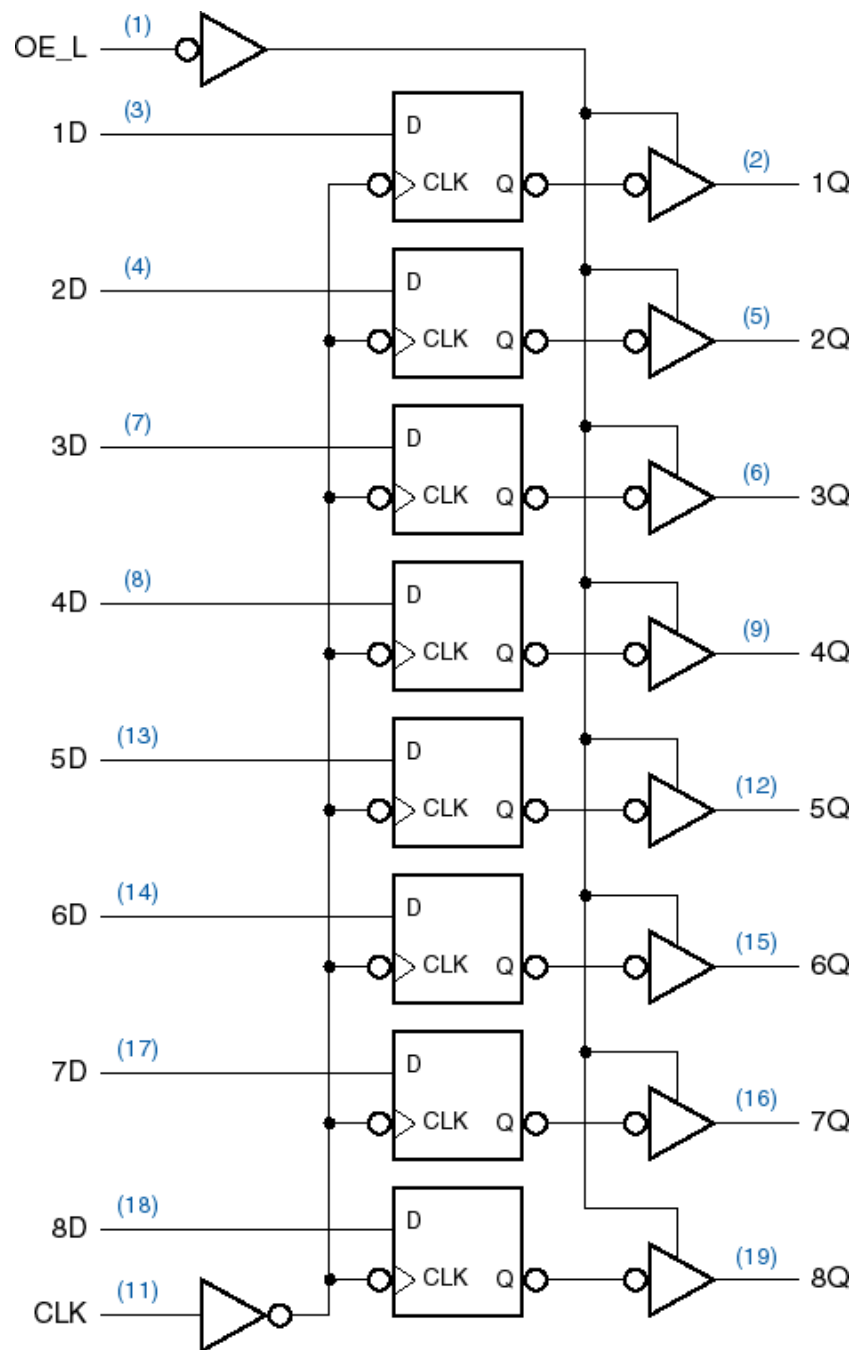
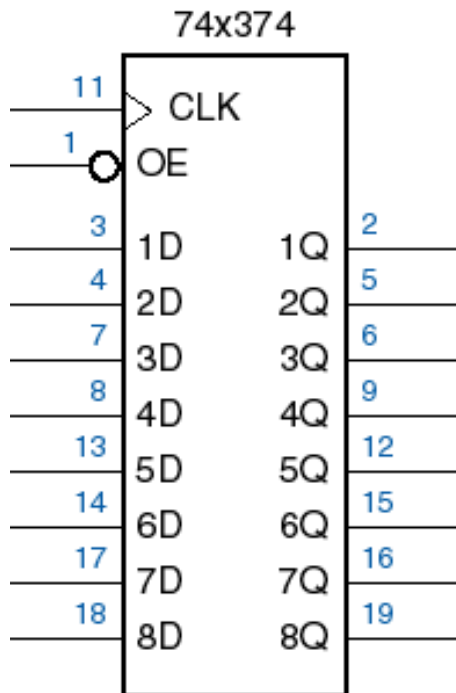




## 2) 8-bit (octal) register

☀ 74x374

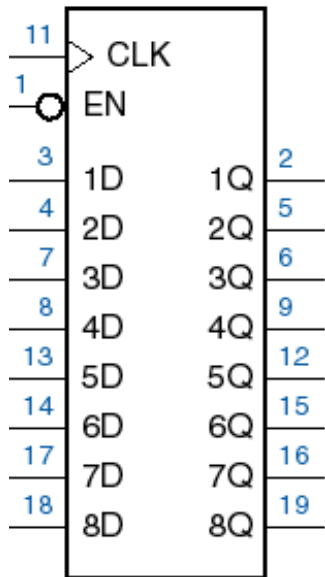
☀ 3-state  
output







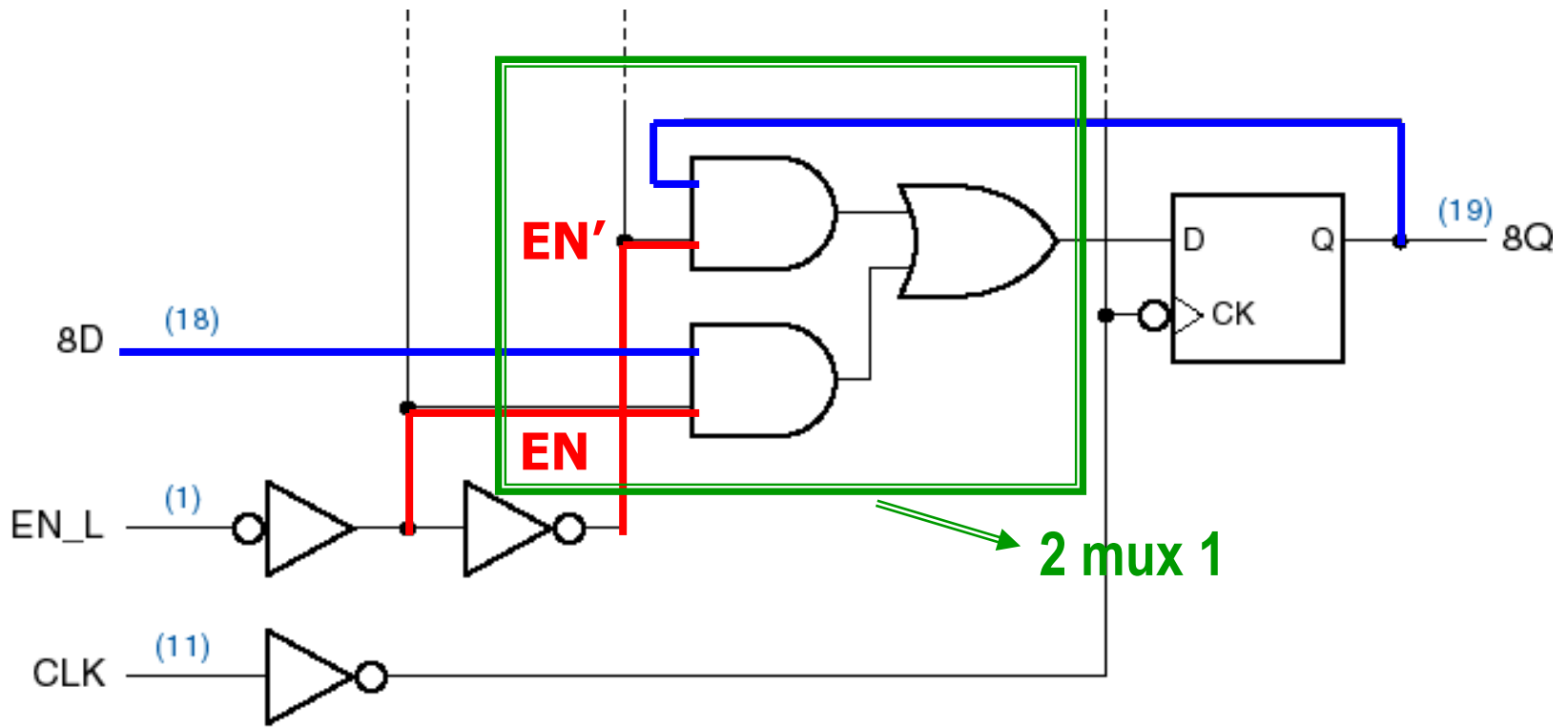
74x377



☀ 74x377

☀ clock enable

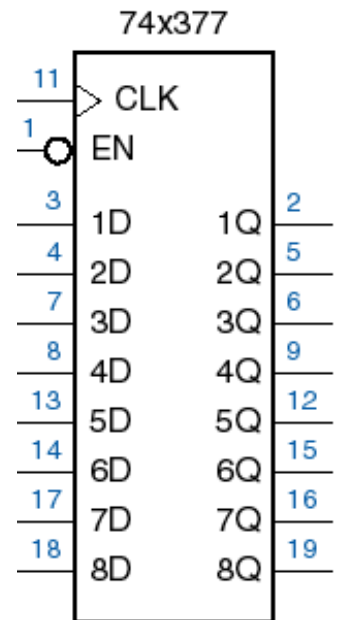
☀ no tristate-buffer



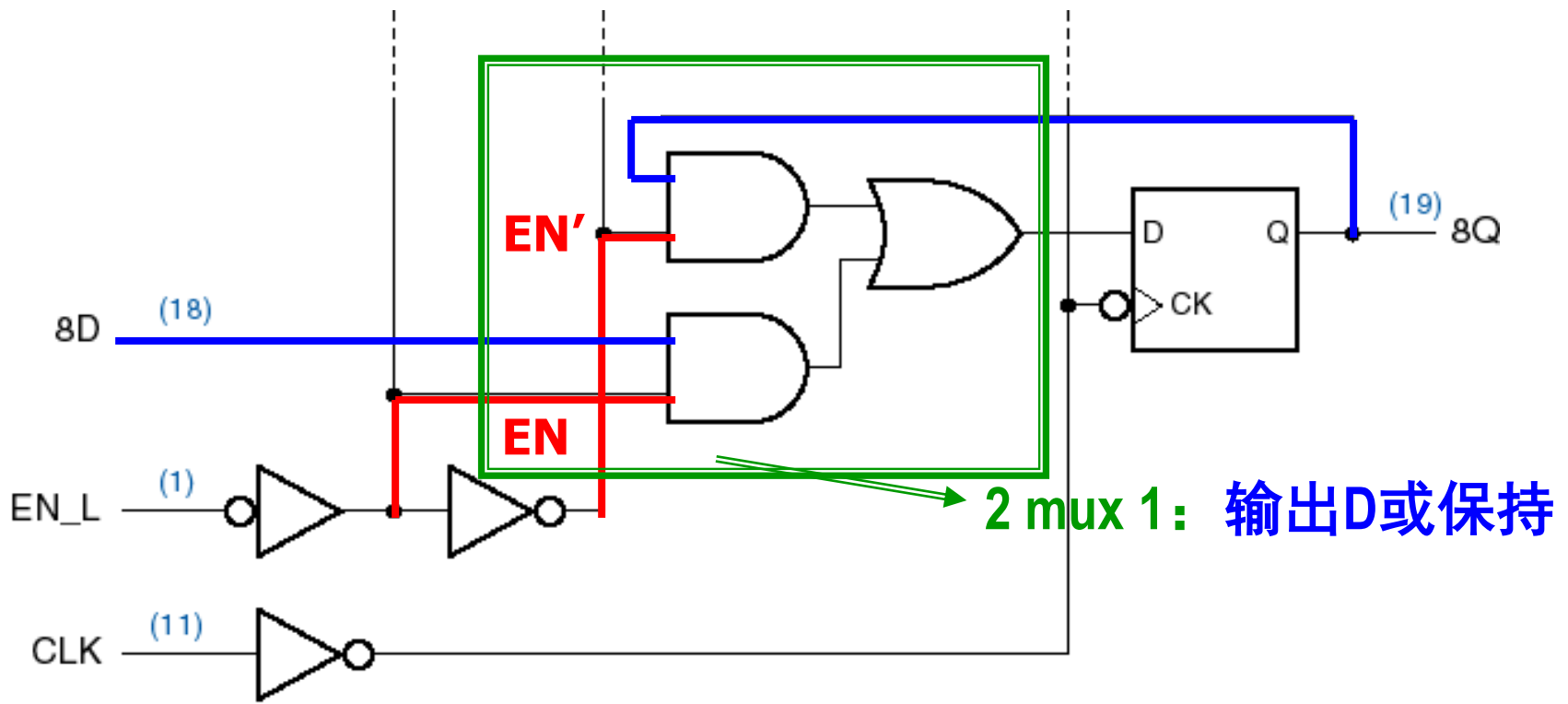


## ☀ 74x377 (与74x374类似)

- ☀ clock enable
- ☀ no tristate-buffer



具有选通时钟的1位逻辑特性



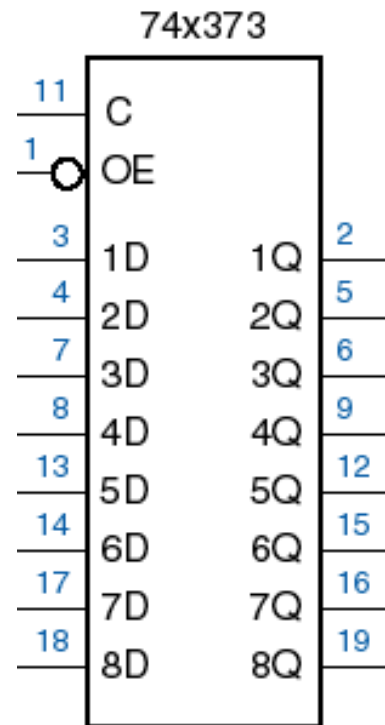


### 3) 8-bit(Octal) latches

☀ 74x373(是74x374的一个变种)

即，用D锁存器代替D-FF

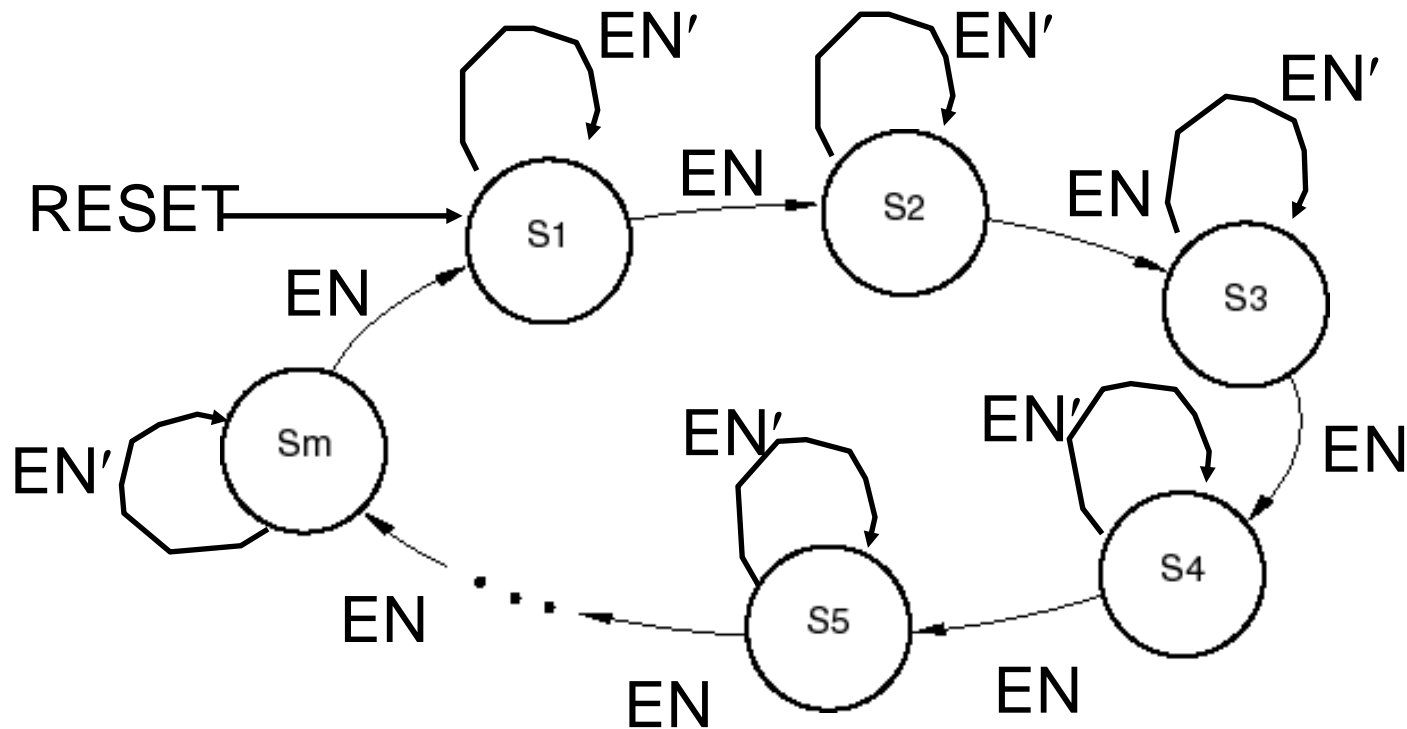
- ☀ Output enable
- ☀ Latch-enable input “C”  
or “G”





## 8.4 Counters

- Any sequential circuit whose state diagram is a single cycle. 状态图中包含有单一循环的任何时序电路



- Modulus (模) : number of states
- Modulo-m counter (divide-by-m counter)
- Most commonly used counter--- n-bit binary counter



## 8.4.1 ripple counter行波计数器-仅需n个触发器

考虑二进制计数顺序：只有当第  $i-1$  位由  $1 \rightarrow 0$  时，第  $i$  位才翻转。

When  $(i-1)_{th}$  bit is change from  $1 \rightarrow 0$ ,  $i_{th}$  bit toggles

工作原理：

(1) 利用一般T触发器实现

Q3 Q2 Q1 Q0

0 0 0 0

0 0 0 1

0 0 1 0

0 0 1 1

0 1 0 0

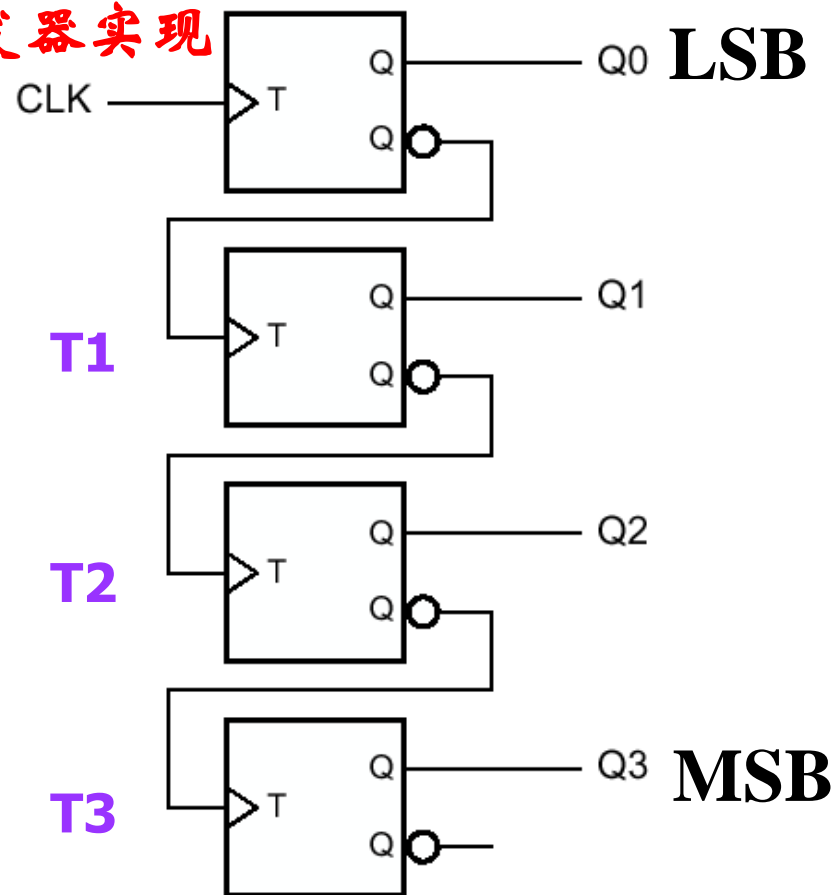
0 1 0 1

0 1 1 0

0 1 1 1

1 0 0 0

⋮  
⋮

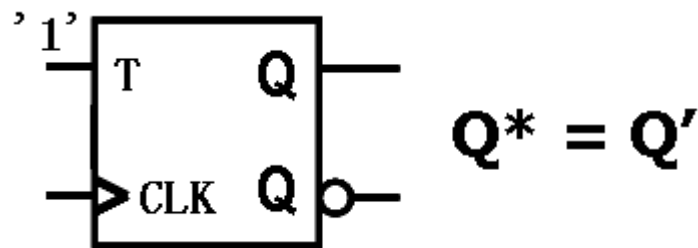


A 4-Bit binary Ripple counter



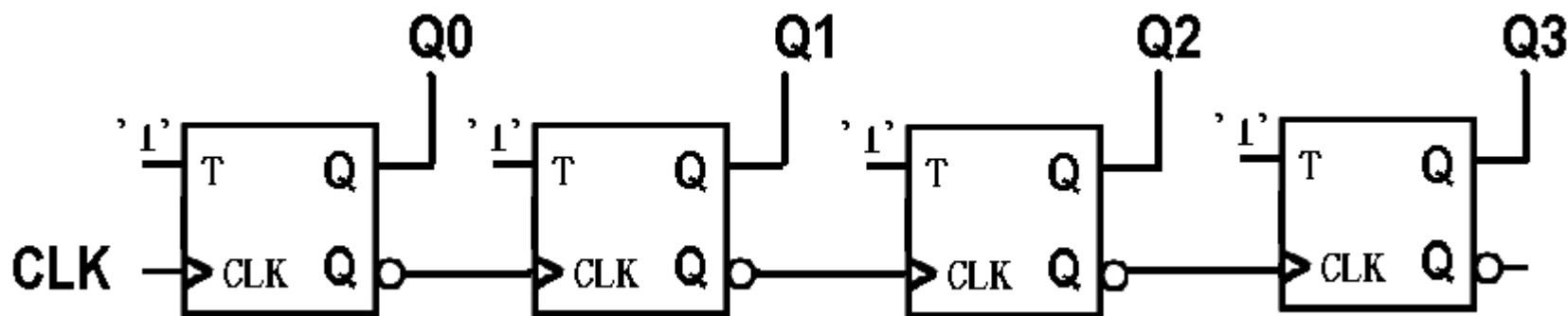
# 行波计数器 (ripple counter)

考虑二进制计数顺序： (2) 利用使能端的T触发器实现：  
只有当第  $i-1$  位由  $1 \rightarrow 0$  时，  
第  $i$  位才翻转。



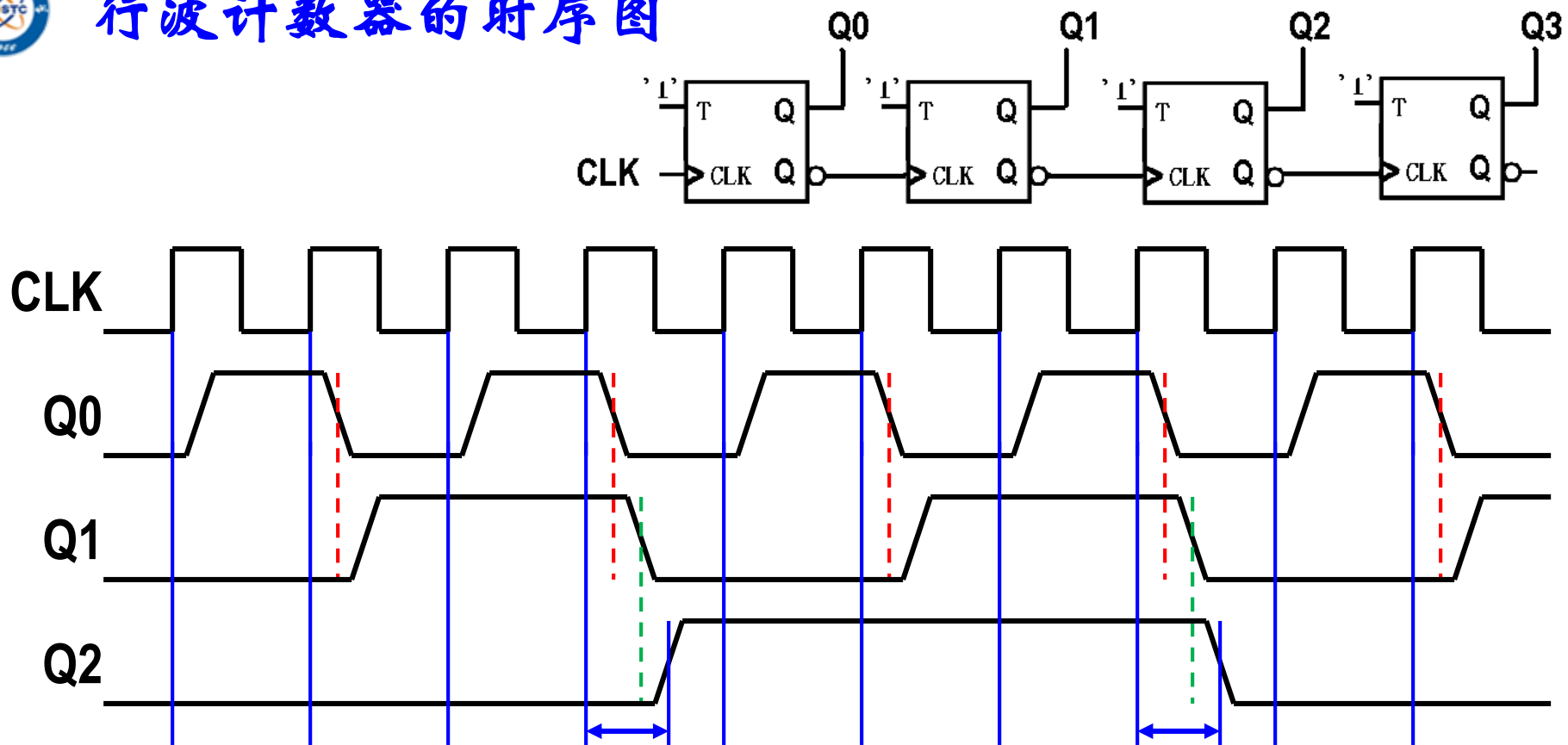
$$T=1 \text{ 代入 } Q^* = T \cdot Q' + T' \cdot Q$$

得  $Q^* = Q'$





## 行波计数器的时序图



States change at different time

—— 异步计数器(Asynchronous Counters)

结论：速度慢，最坏情况，第 $n$ 位要经过  $n \times t_{TQ}$  的延迟时间

行波计数器特点：优点—结构最简单；缺点—速度最慢



## 8.4.2 Synchronous counter 同步计数器

0 1 1  
↓  
1 0 0

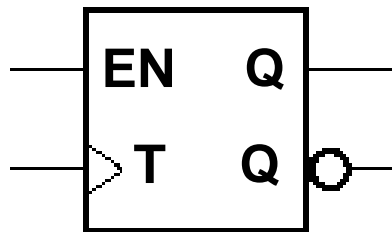
工作原理: When the bits **below** the  $i_{th}$  bit are **all 1**, then the  $i_{th}$  bit change its value

以多位二进制数加法为例阐述上述原理

$$\begin{array}{r} 1011011 \\ + \quad \quad 1 \\ \hline 1011100 \end{array}$$

在多位二进制数的**末位加 1**,  
仅当第  $i$  位以下的各位都为 **1** 时,  
第  $i$  位的状态才会改变。  
最低位的状态每次加**1**都要改变。

□ 利用有使能端的 T 触发器实现:



$$Q^* = EN \cdot Q' + EN' \cdot Q = EN \oplus Q$$

When toggle is needed, make **EN = 1**

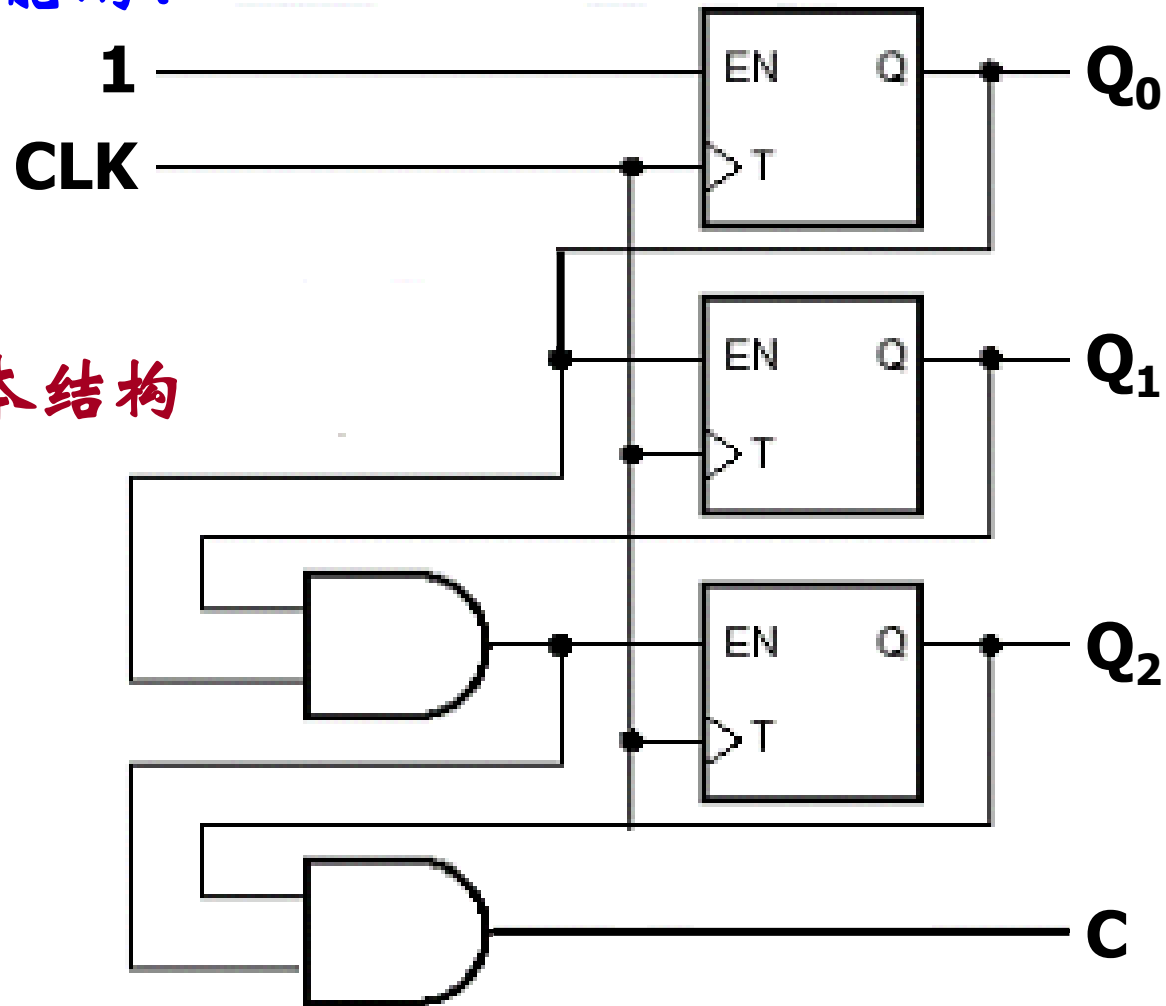
$$EN_i = Q_{i-1} \cdot Q_{i-2} \cdot \dots \cdot Q_1 \cdot Q_0$$

$$EN_0 = 1$$





如何加入使能端？



Synchronous  
counter的基本结构

“modulo-8” counter



LSB

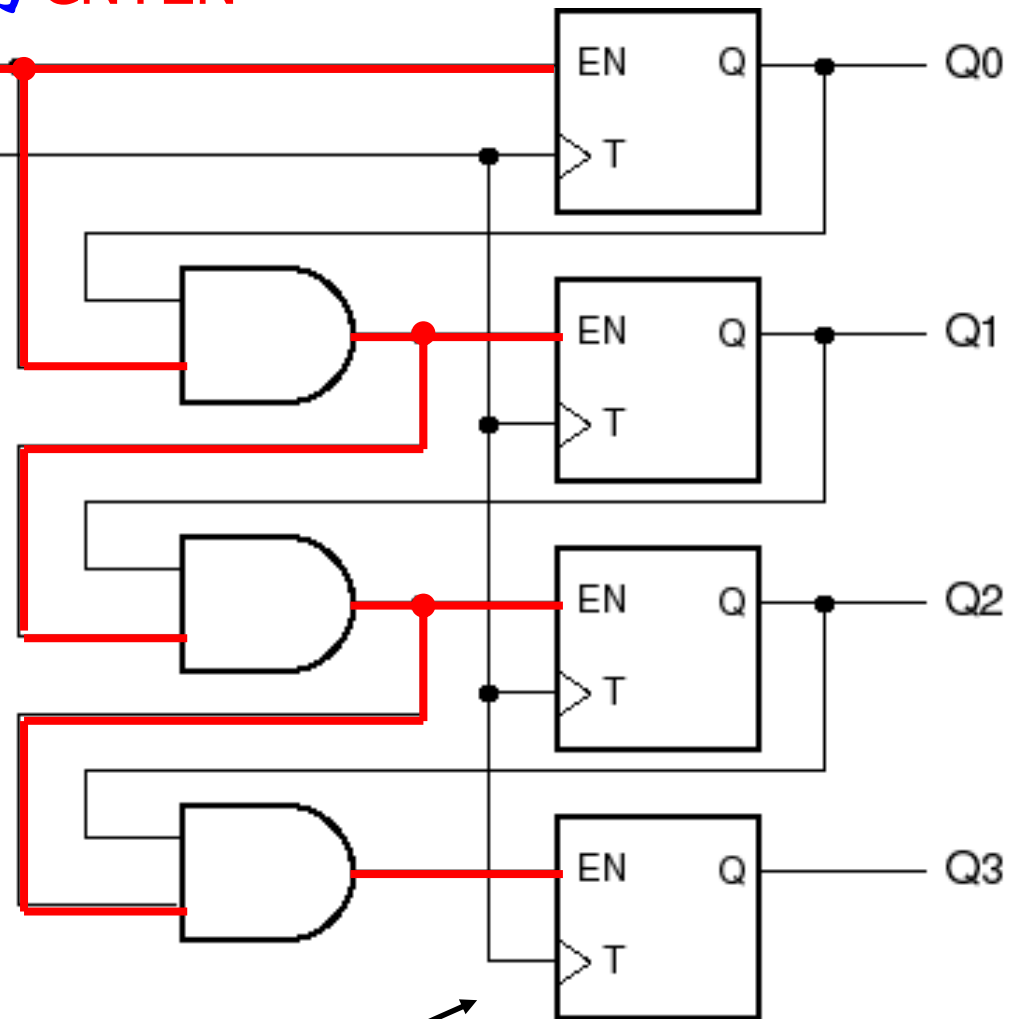
将  $EN_0 = 1$  修改为主计数使能信号 **CNTEN**

**CNTEN**

CLK

**Synchronous**  
**counter with enable**

**Serial enable logic**

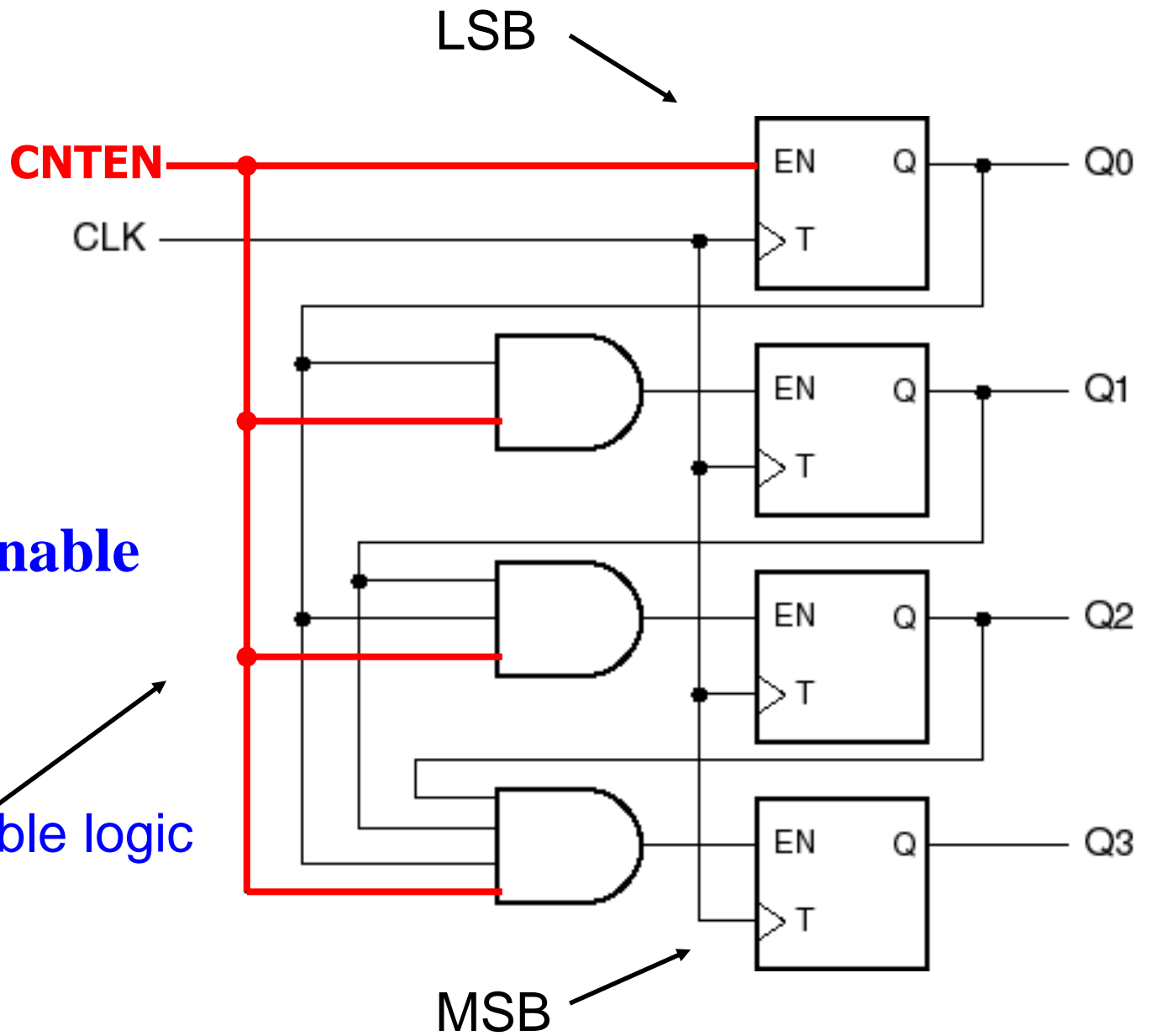


MSB



# Synchronous counter with enable

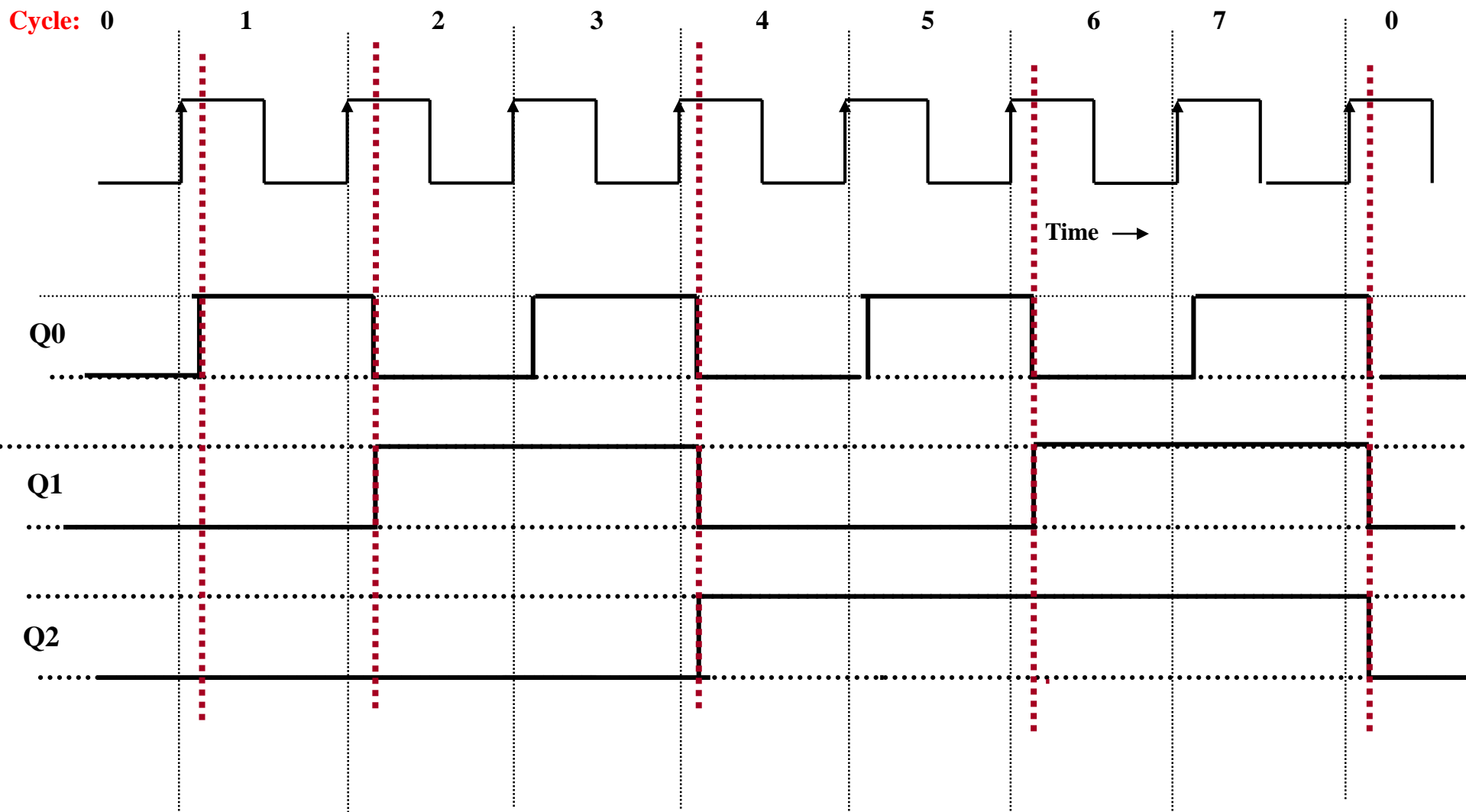
Parallel enable logic





# 同步计数器的典型时序图

Clock

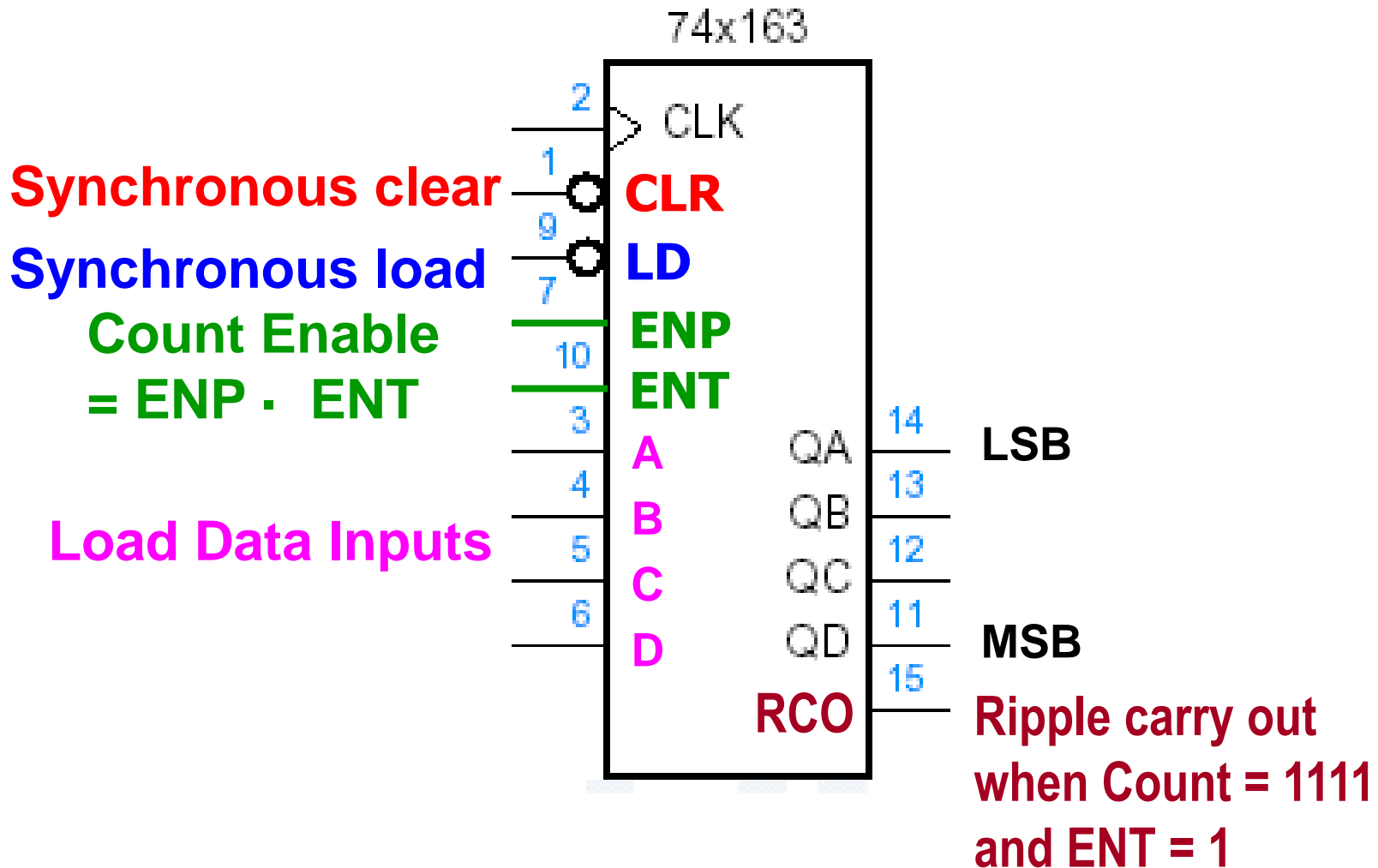


States change at the same time

—同步计数器



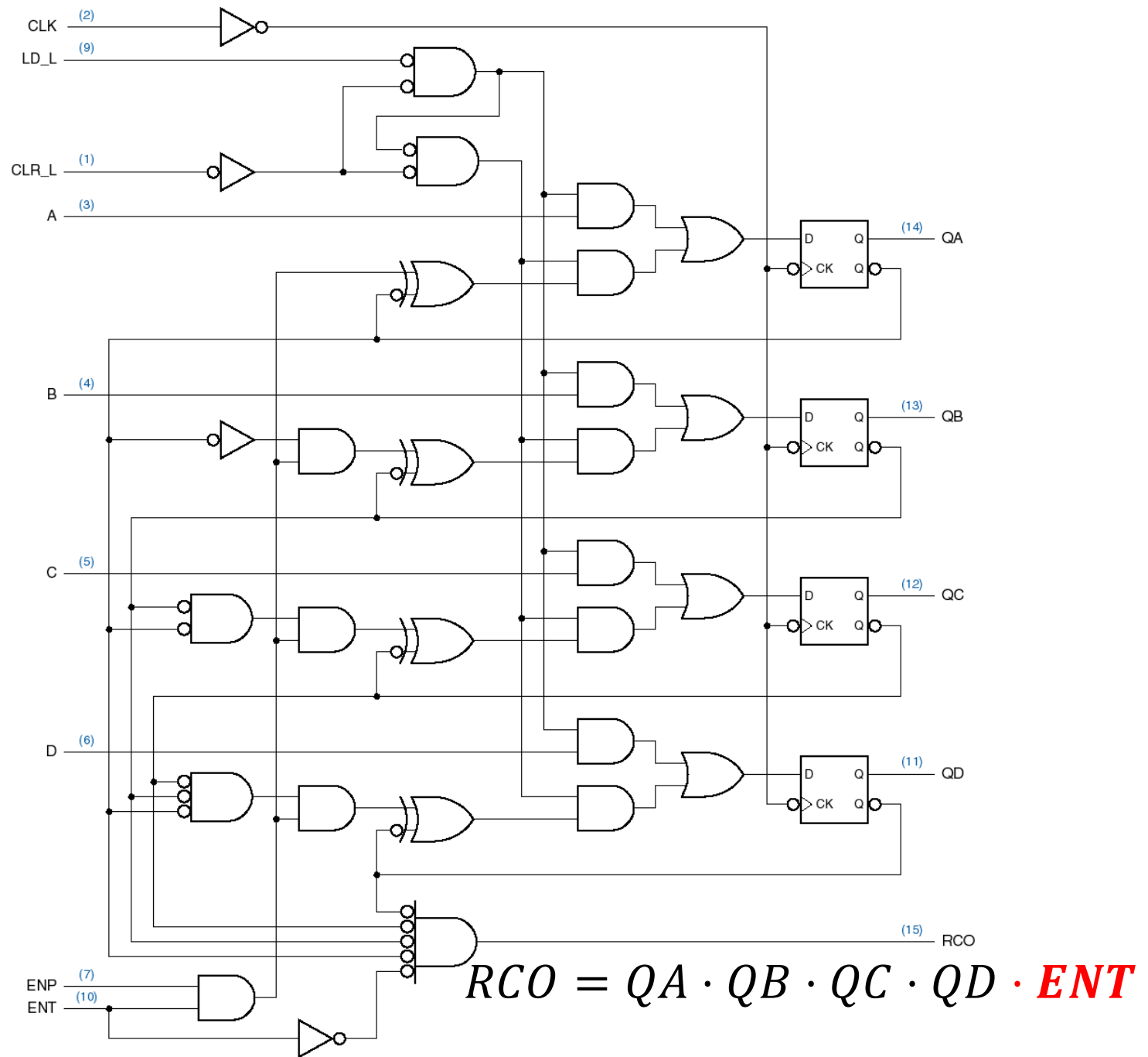
## 8.4.3 MSI counter: 4-bit counter 74x163



$$RCO = QA \cdot QB \cdot QC \cdot QD \cdot ENT$$



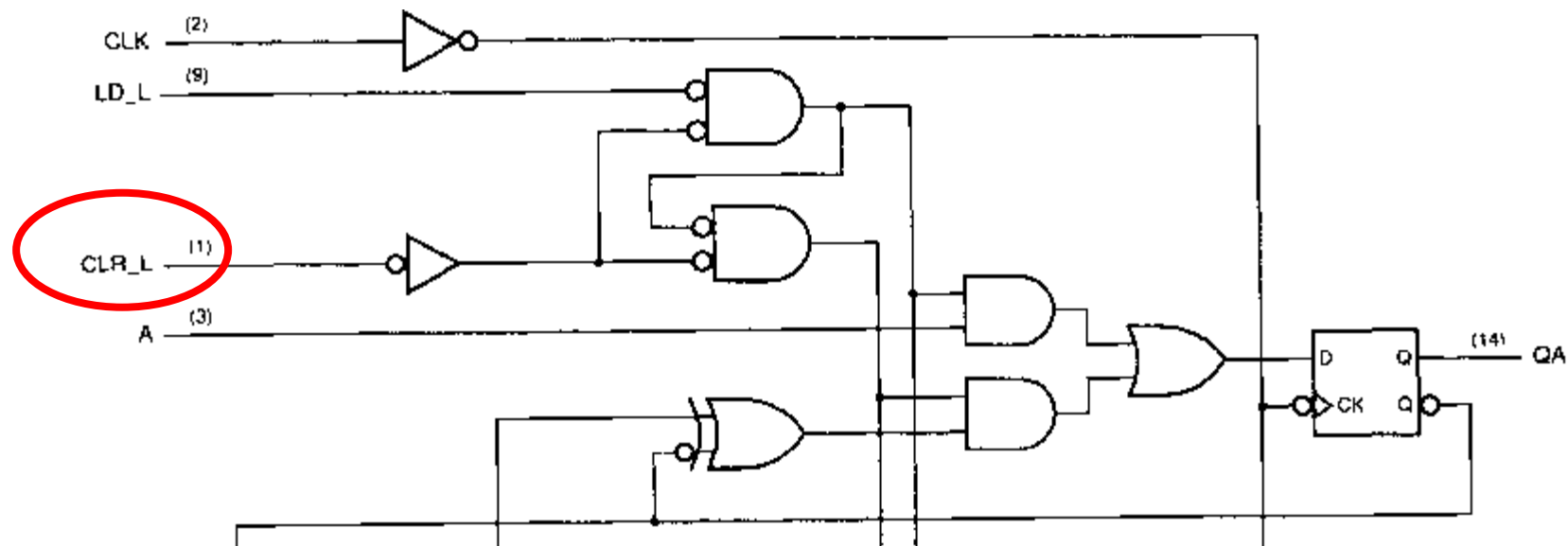
# 74x163 internal logic diagram



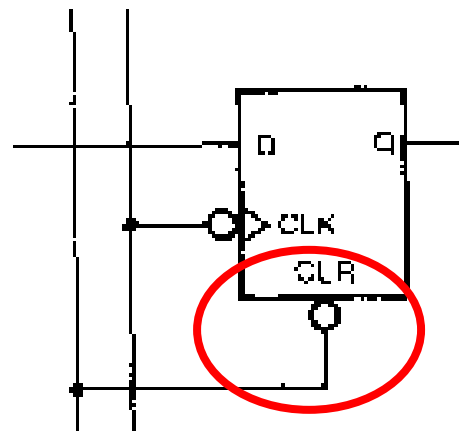


# 同步清零和异步清零

## ☀ 74x163 (同步清零)

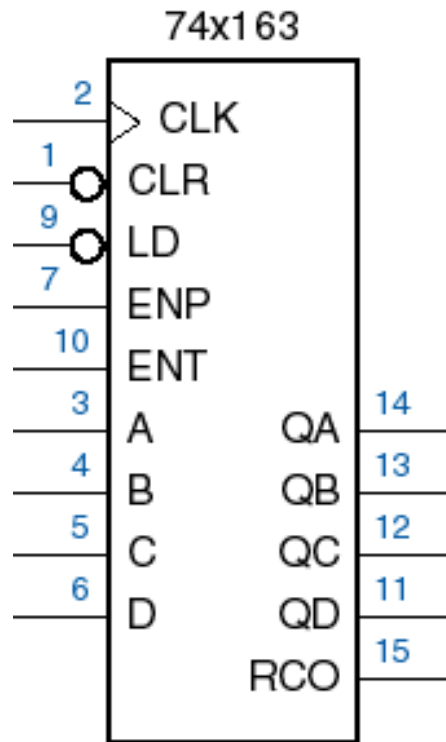


## ☀ 74x161 (异步清零)



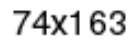





# 74x163 function table



Inputs				Current State				Next State			
CLR_L	LD_L	ENT	ENP	QD	QC	QB	QA	QD*	QC*	QB*	QA*
0	x	x	x	x	x	x	x	0	0	0	0
1	0	x	x	x	x	x	x	D	C	B	A
1	1	0	x	x	x	x	x	QD	QC	QB	QA
1	1	x	0	x	x	x	x	QD	QC	QB	QA
1	1	1	1	0	0	0	0	0	0	0	1
1	1	1	1	0	0	0	1	0	0	1	0
1	1	1	1	0	0	1	0	0	0	1	1
1	1	1	1	0	0	1	1	0	1	0	0
1	1	1	1	0	1	0	0	0	1	0	1
1	1	1	1	0	1	0	1	0	1	1	0
1	1	1	1	0	1	1	0	0	1	1	1
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	1	0	0	0	1	0	0	1
1	1	1	1	1	0	0	1	1	0	1	0
1	1	1	1	1	0	1	0	1	0	1	1
1	1	1	1	1	0	1	1	1	1	0	0
1	1	1	1	1	1	0	0	1	1	0	1
1	1	1	1	1	1	0	1	1	1	1	0
1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0

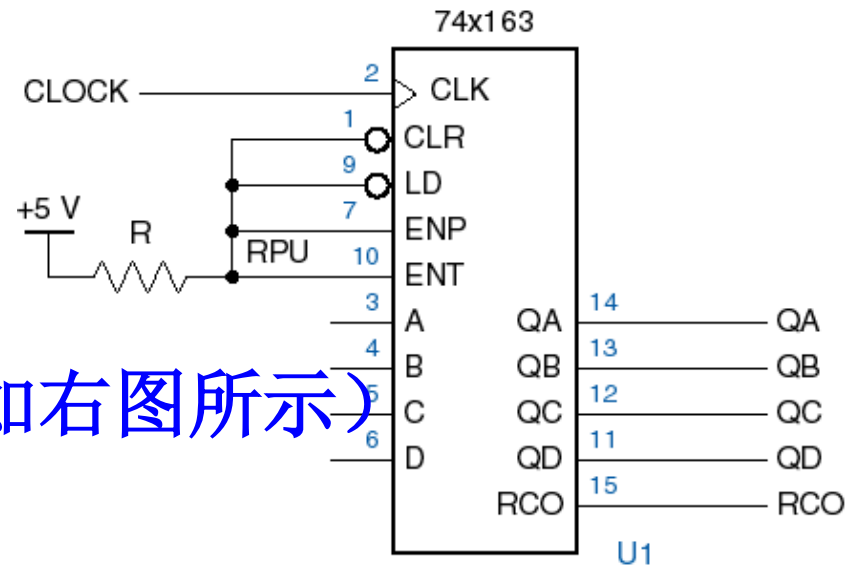




CLK	CLR_L	LD_L	ENP	ENT	operations
	<b>0</b>	×	×	×	<b>Synchronous clear</b>
	<b>1</b>	<b>0</b>	×	×	<b>Synchronous load</b>
×	<b>1</b>	<b>1</b>	<b>0</b>	×	<b>Keep state</b>
×	<b>1</b>	<b>1</b>	×	<b>0</b>	<b>Keep state</b>
	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>Free running</b>



# Counter operation

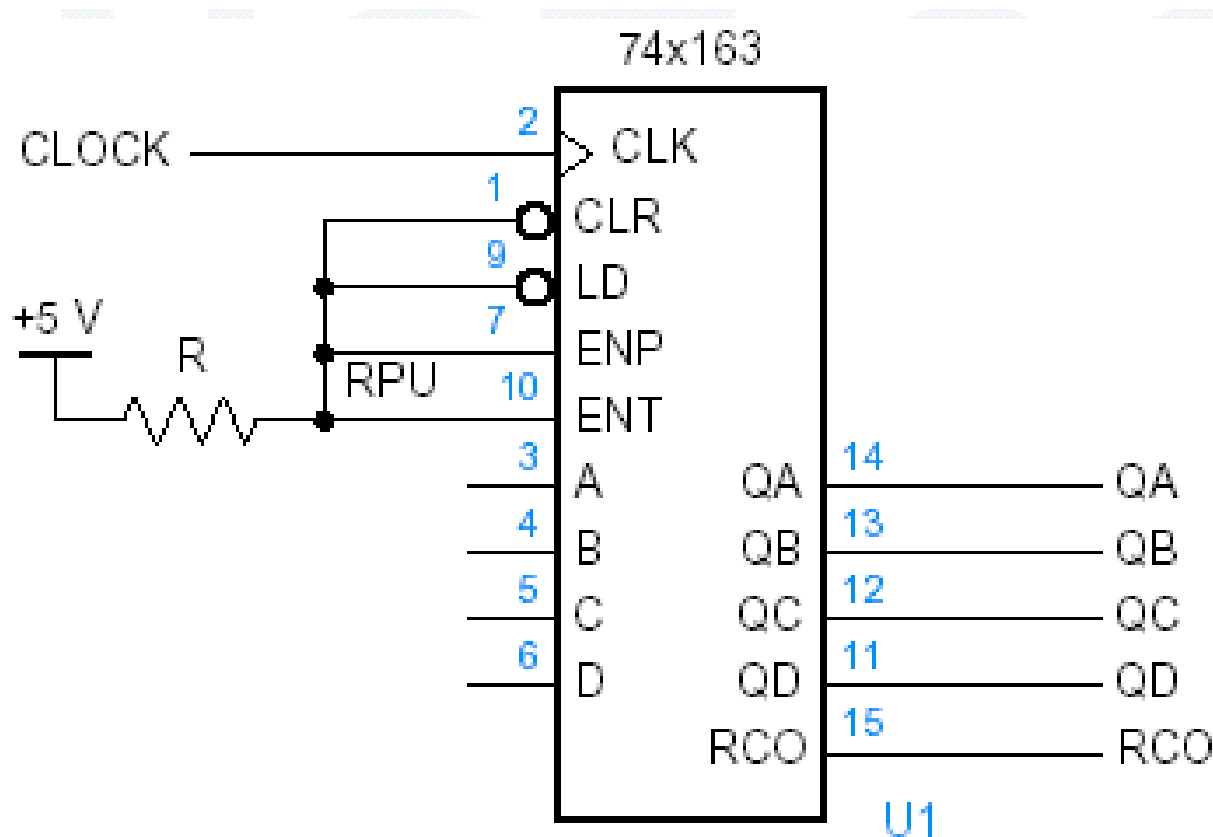


- ☀ **Free-running  $\div 16$**   
(自由运行模式的接线方法，如右图所示)
- ☀ **Count if ENP and ENT both asserted.**
- ☀ **Load if LD is asserted (overrides counting).**
- ☀ **Clear if CLR is asserted (overrides loading and counting).**
- ☀ **All operations take place on rising CLK edge.**
- ☀ **RCO is asserted if ENT is asserted and Count = 15.**



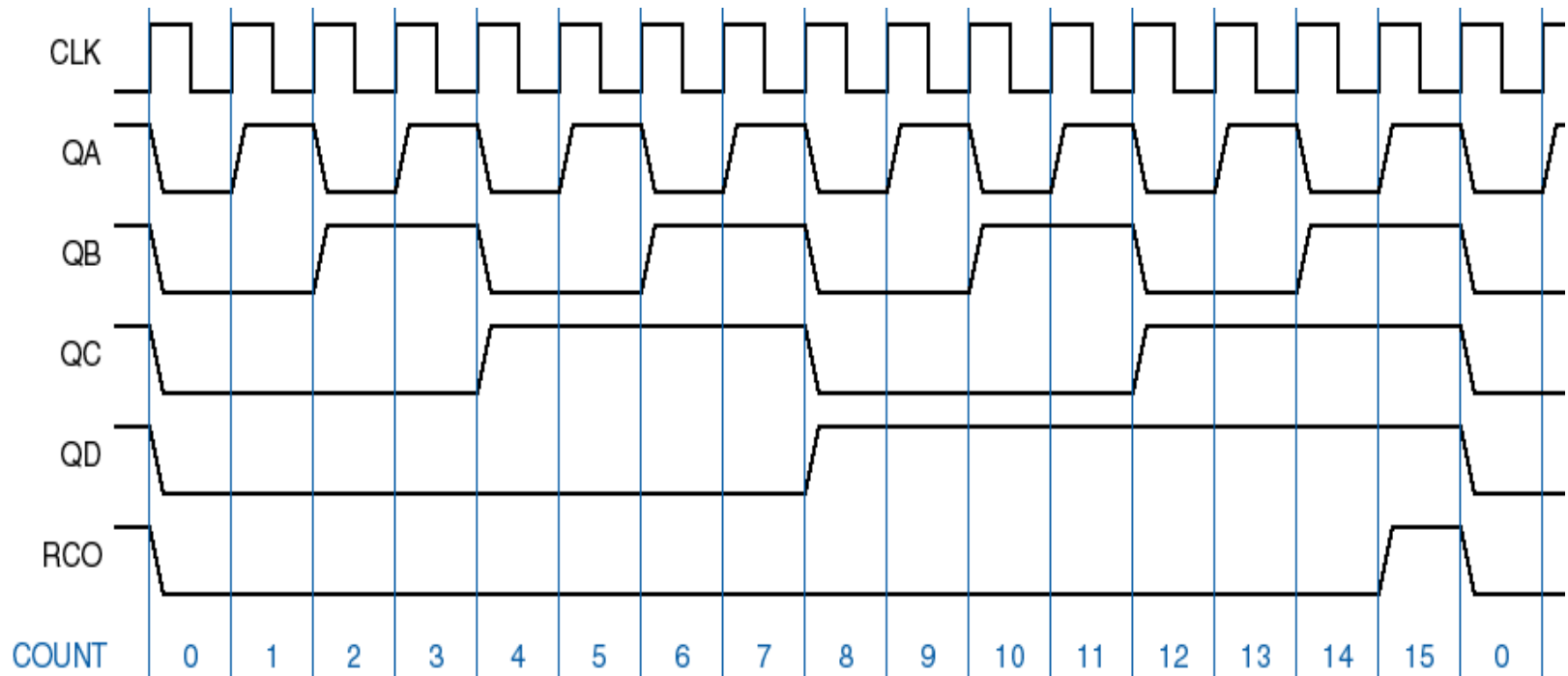
# Redrawing: Free-running 4-bit counter

## 74x163工作于自由运行模式时的接线方法





# Free-running 4-bit '163 counter



✳ “modulo-16” or “divide-by-16” counter

自由运行的'163可以用作2 (QA)、4 (QB)、  
8 (QC)和16 (QD)分频计数器



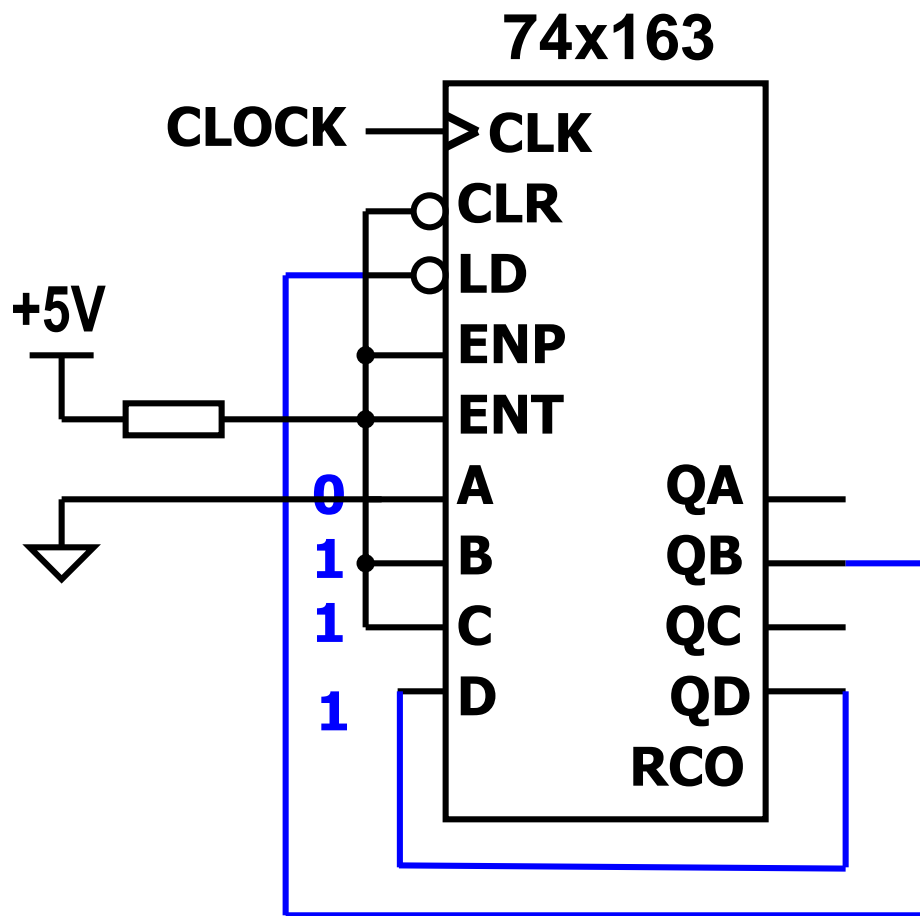
# Analyze the counter(1)

分析下面电路的模为多少？

其它状态？

思考：QB 多少分频？ QD 呢？

QB 是 3 分频， 占空比 2/3； QD 是 6 分频， 占空比 50%



分析如下：

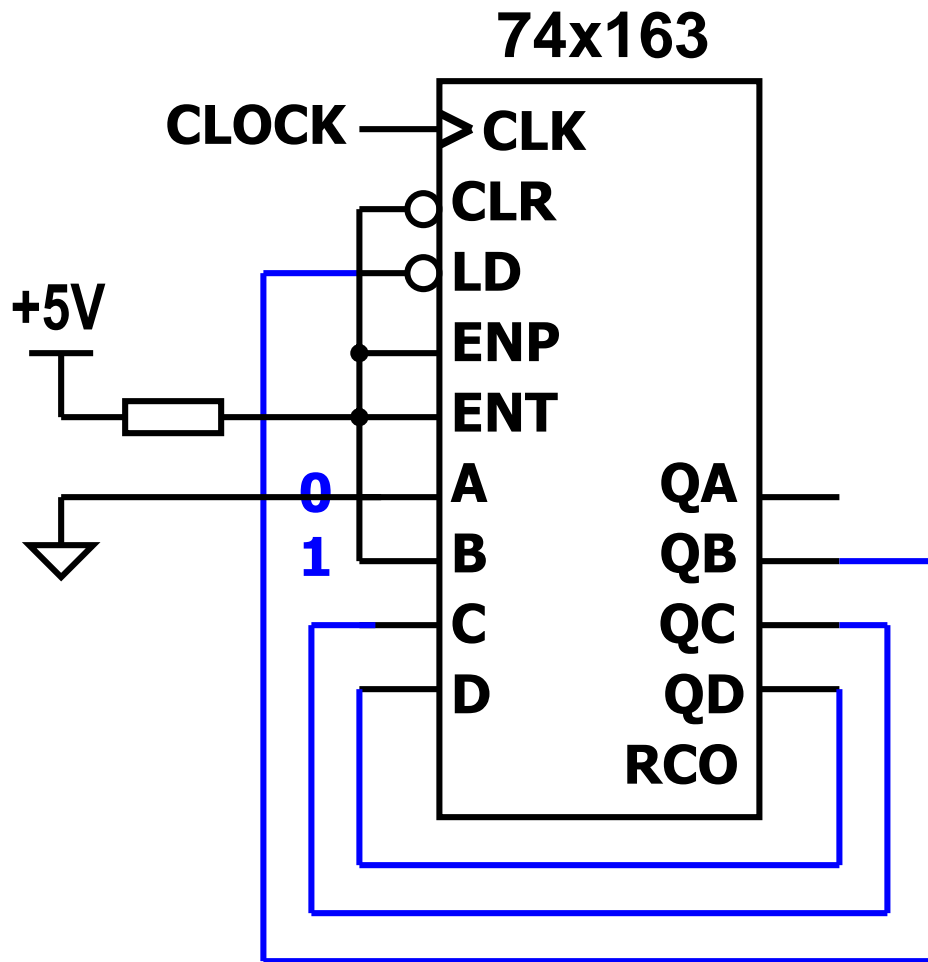
QD	QC	QB	QA
0	0	0	0
0	1	1	0
0	1	1	1
1	0	0	0
1	1	1	0
1	1	1	1

“modulo-6” counter



## Analyze the counter(2)

分析下面电路的模为多少？



**Solution:** “modulo-12” counter

分析如下：

QD	QC	QB	QA
0	0	0	0
0	0	1	0
0	0	1	1
0	1	0	0
0	1	1	0
0	1	1	1
1	0	0	0
1	0	1	0
1	0	1	1
1	1	0	0
1	1	1	0
1	1	1	1



## Other MSI counters

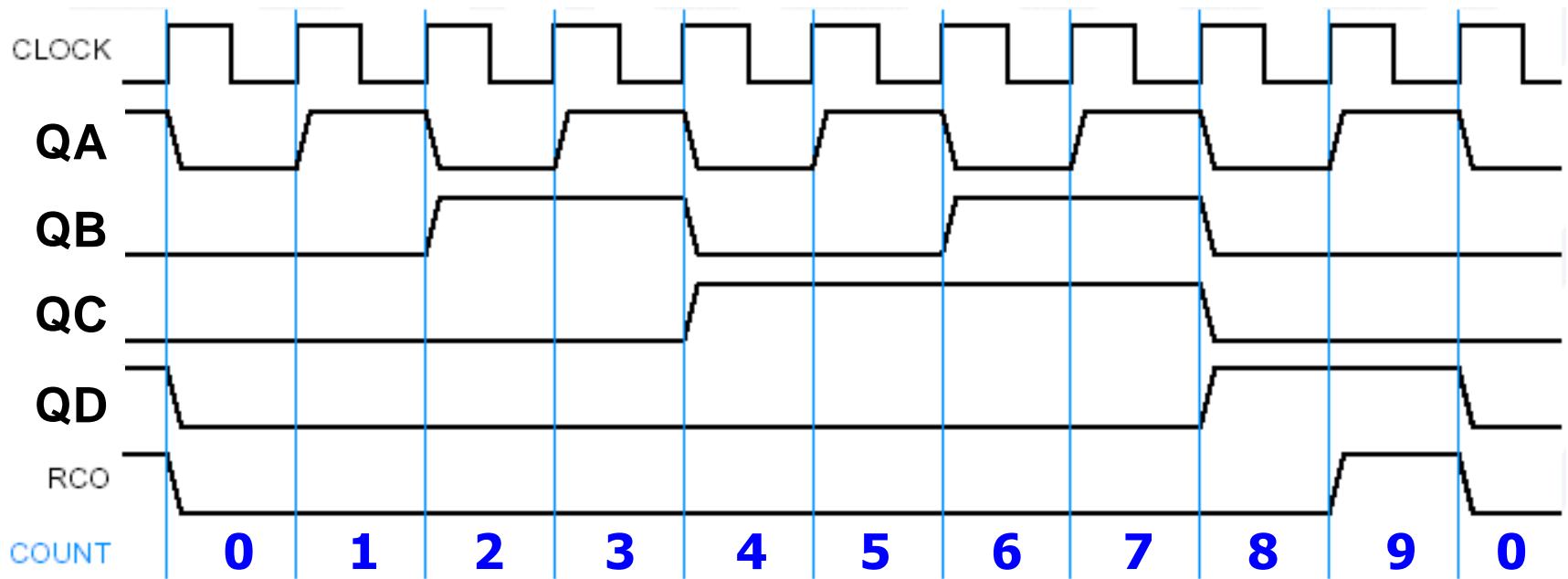
★ **74x163: 同步清零; 74x161:异步清零**

★ **74x160、74x162**

**1-bit decimal counter (BCD)**

**74x160 :asynchronous clear**

**74x162: synchronous clear**





## ★ 74x169 UP/DOWN COUNTER

UP/DN = 1 = up

UP/DN = 0 = down

down

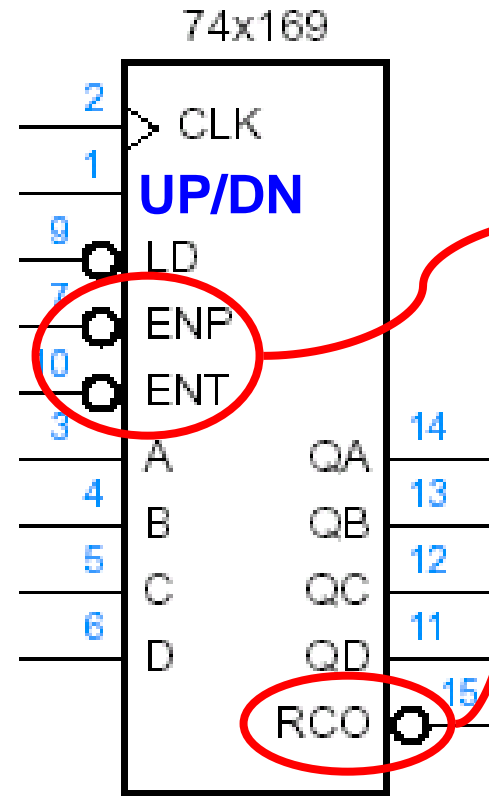
up

Ex: 1,0,15,14,

15,0,1,2

RCO\_L=0

RCO\_L=0



Enable input

Carry out

Active low





# Counter application(1)

## -----Generating arbitrary modulo-m counter

Considering two cases (using n-bit binary counter devices ):

- ★  $m < 2^n$        $\rightarrow$  Clear、 Load
- ★  $m > 2^n$        $\rightarrow$  Cascading counters

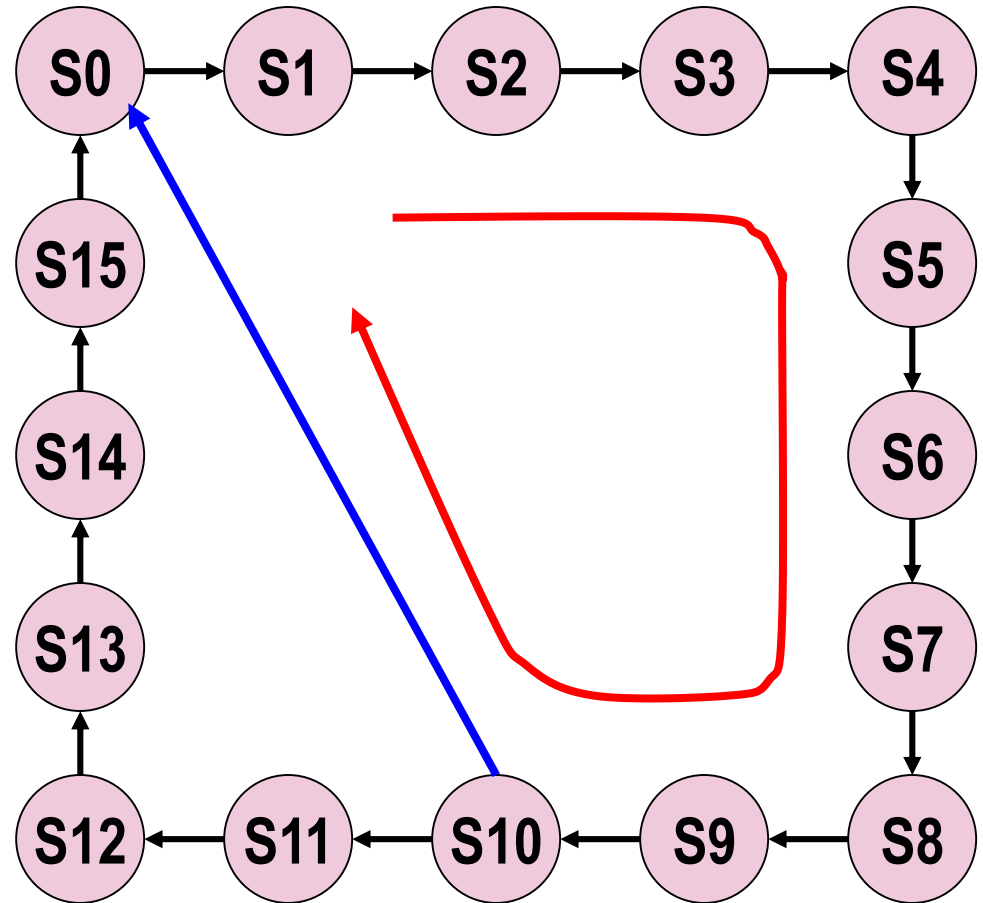


# Example 1: modulo-11 counter using 74x163

$$m < 2^n$$

□ Clear

**When counts to 1010,  
CLR input is asserted,  
Counter outputs are  
forced to 0000.**





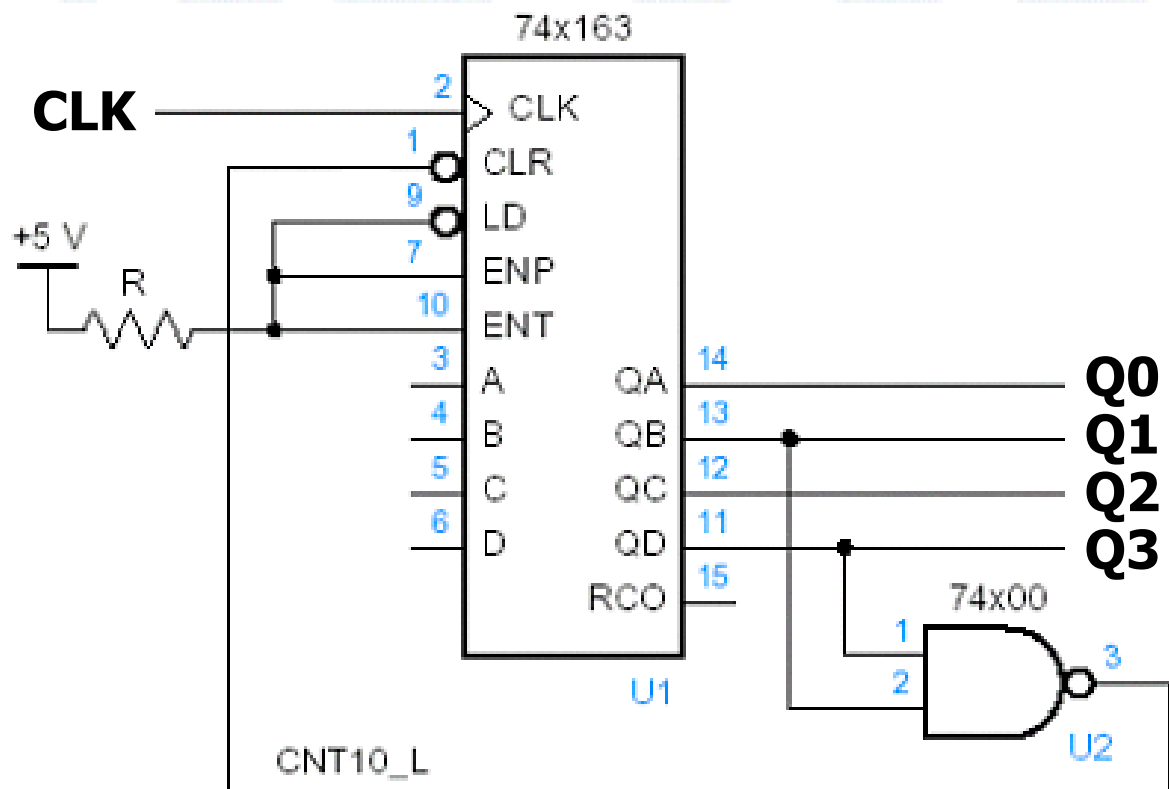
# Example 1: modulo-11 counter using 74x163

□ Clear

$$m < 2^n$$

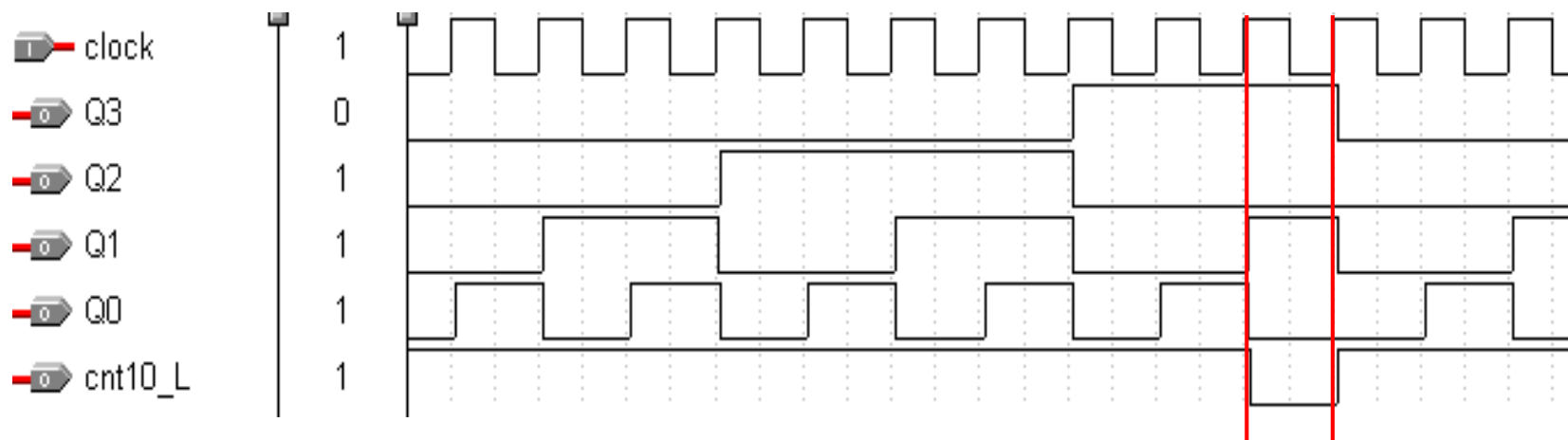
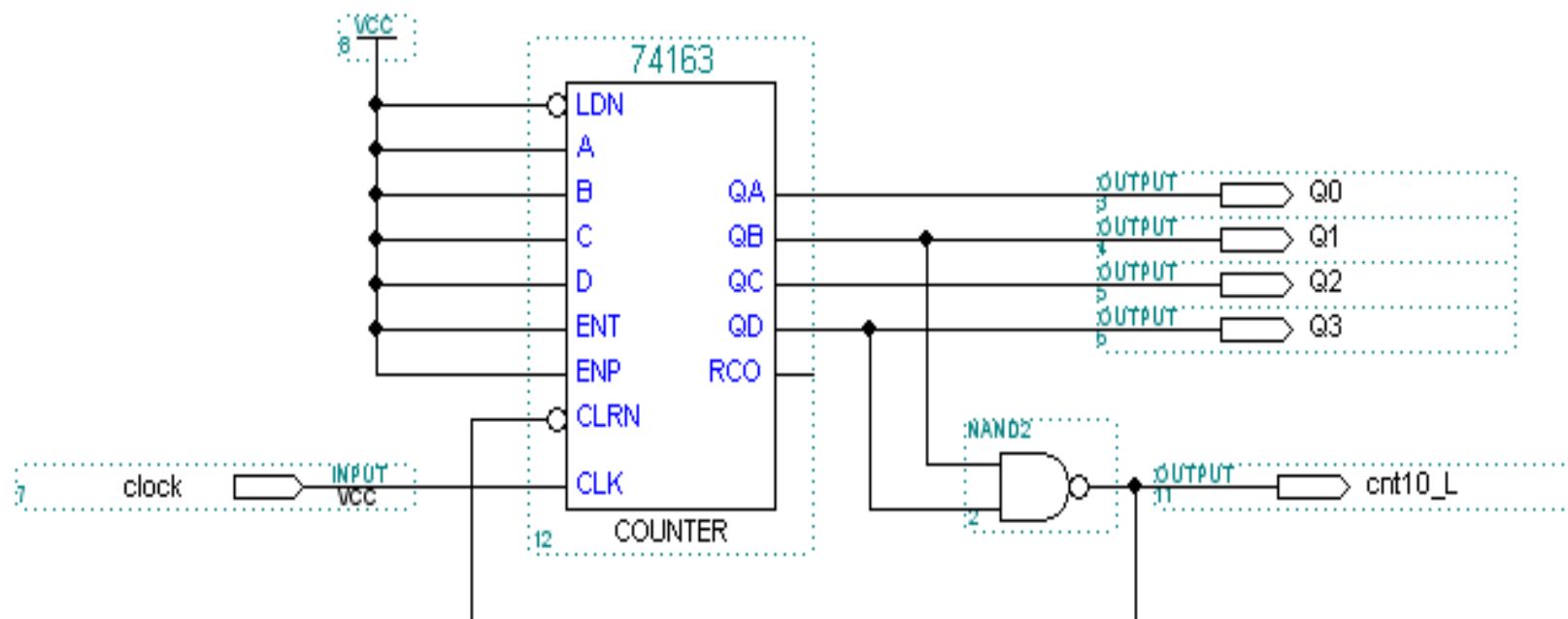
计数到**1010**时，  
利用同步清零端  
强制为**0000**。

思考：  
如果是**74x161**  
(异步清零)  
怎么连接？



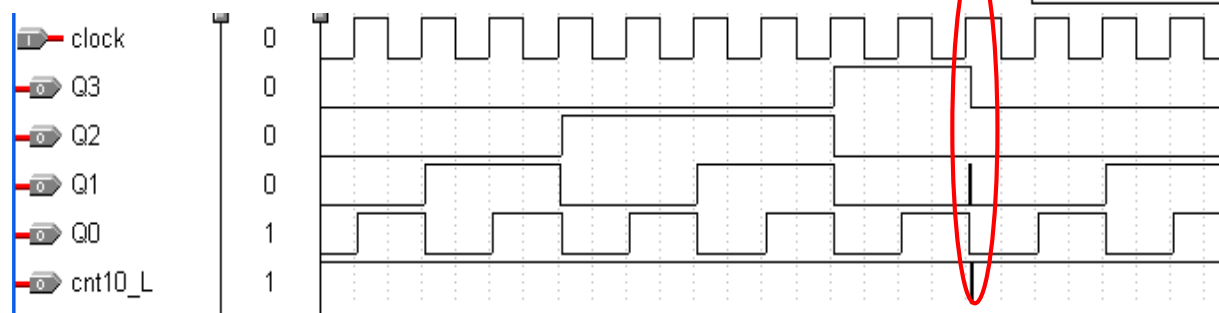
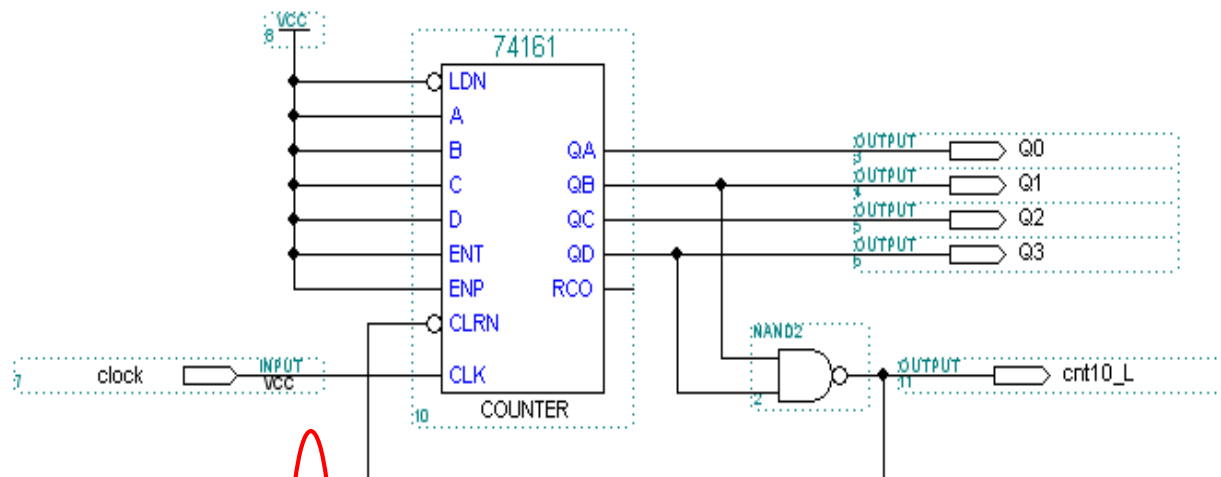
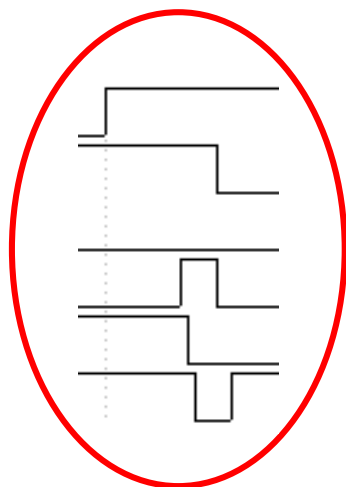


## 74x163实现模11计数器的仿真分析





# 74x161实现模10计数器的仿真分析



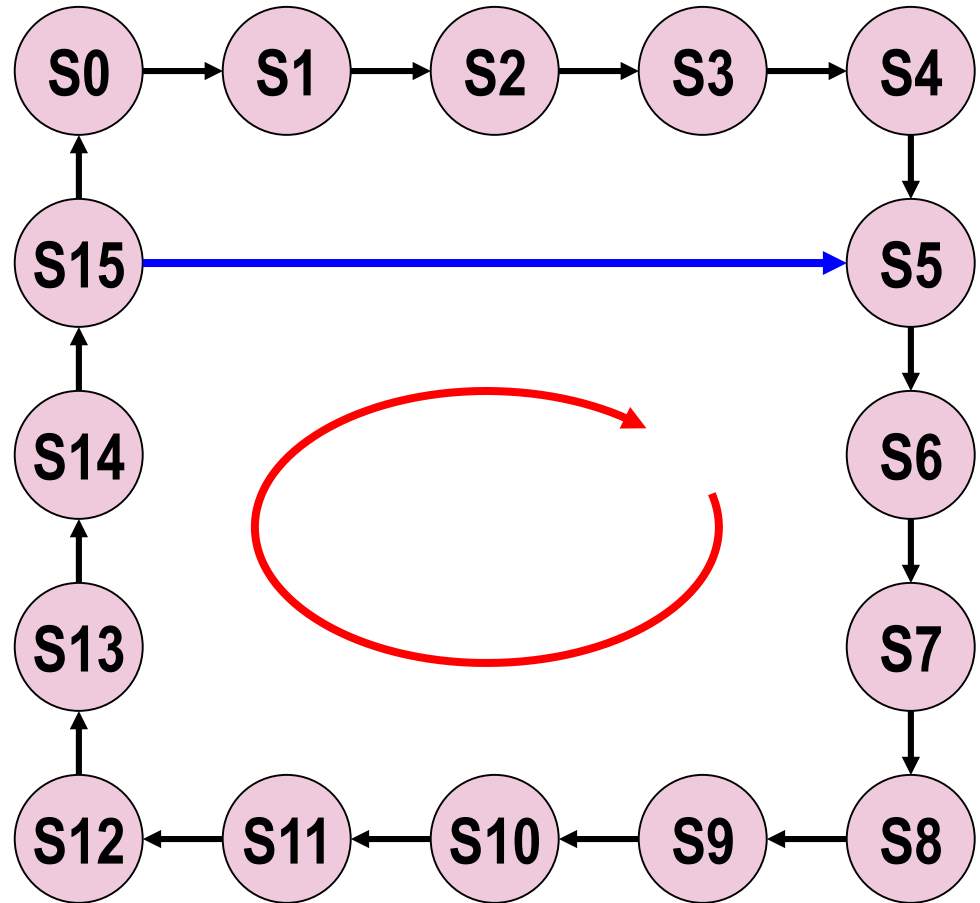
## 模10计数器

**说明：**74x161是异步清零、同步计数，区别于74x163同步清零、同步计数。本题中74x161从0000状态开始计数，在其输出1010（10）时产生清零信号。因为是异步清零，因此，清零信号马上起作用，计数器输出立刻变为0000。所以，1010状态持续时间很短，与0000同在一个时钟周期内。因此，该计数器为10进制。



## ❑ Load

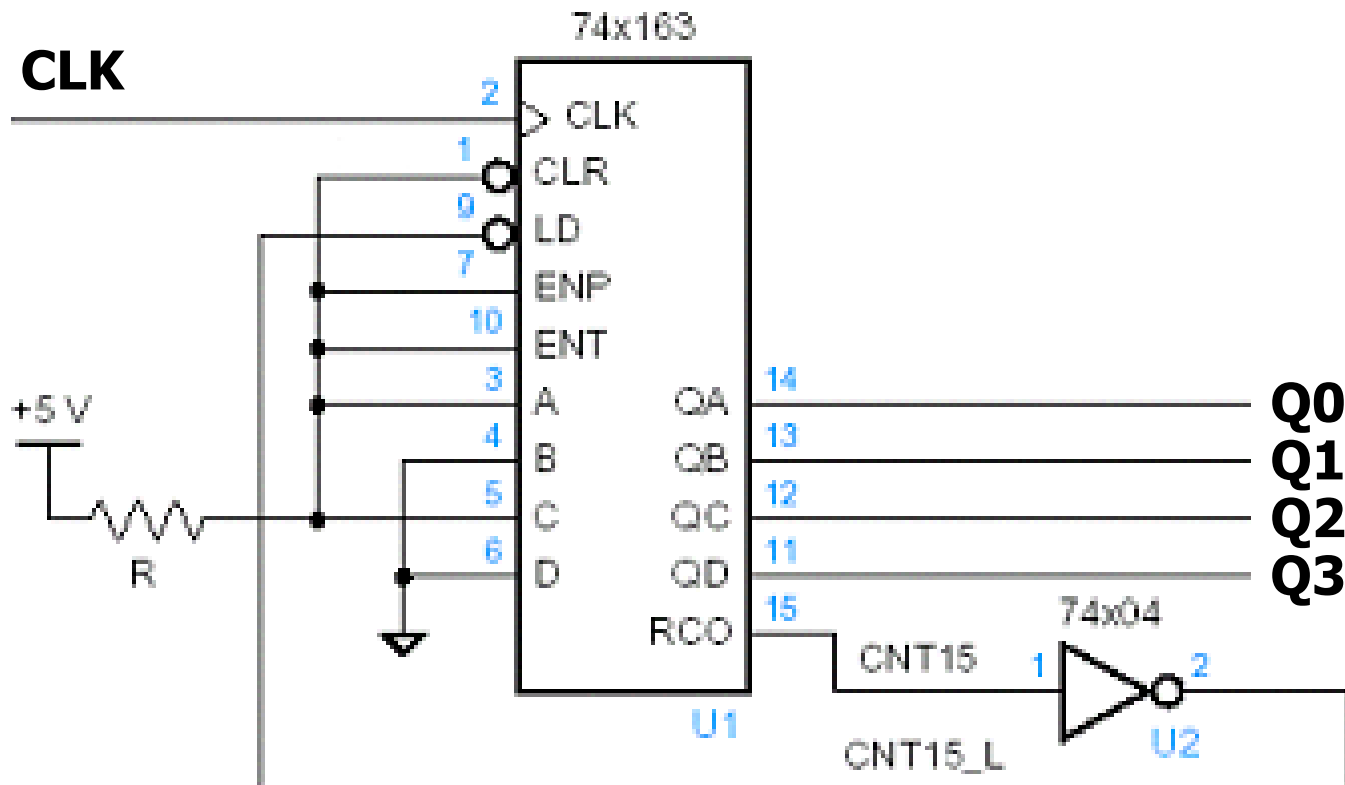
**When counts to 1111,  
LD input is asserted,  
Counter outputs are  
loaded to 0101.**





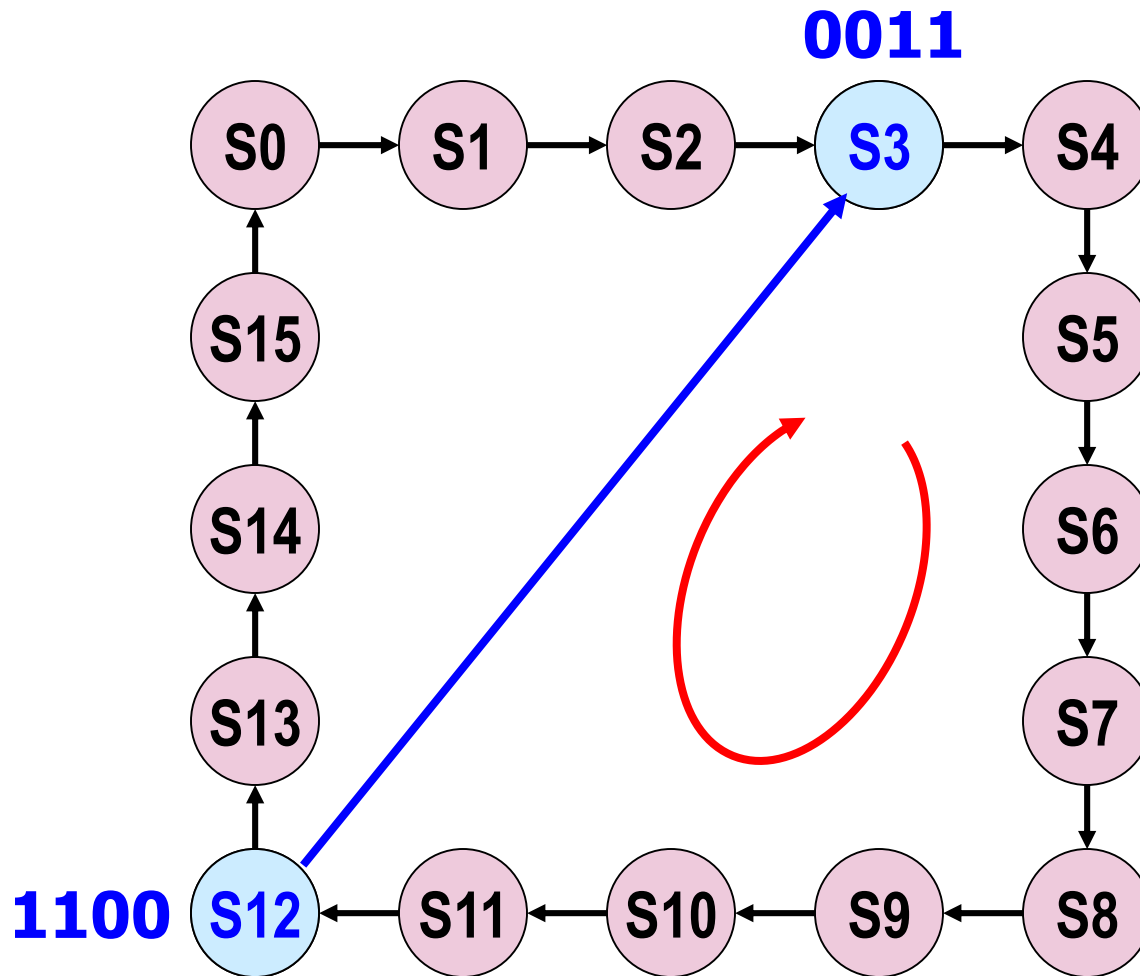
□ Load

**When counts to 1111,  
LD input is asserted,  
Counter outputs are  
loaded to 0101.**

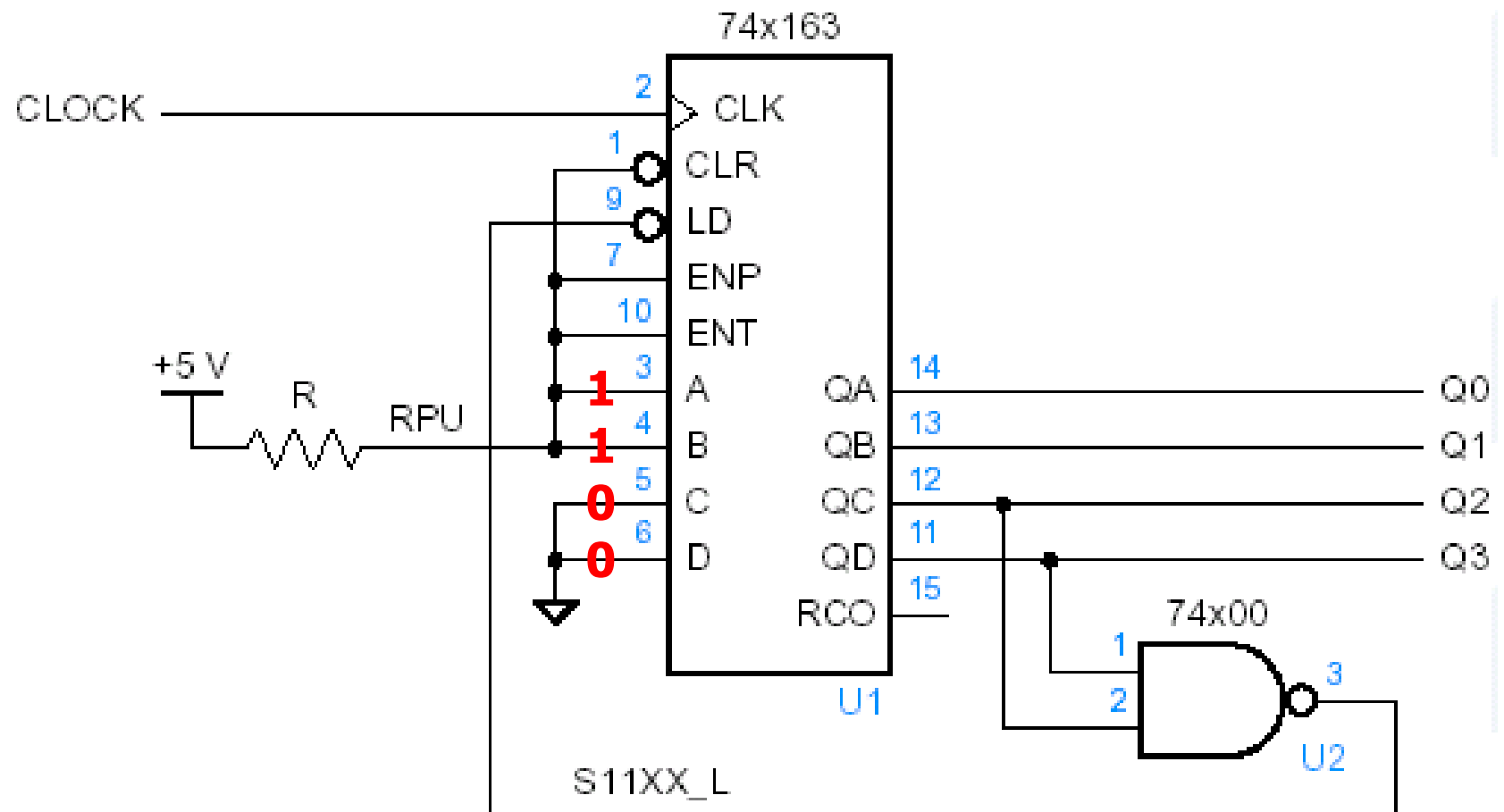


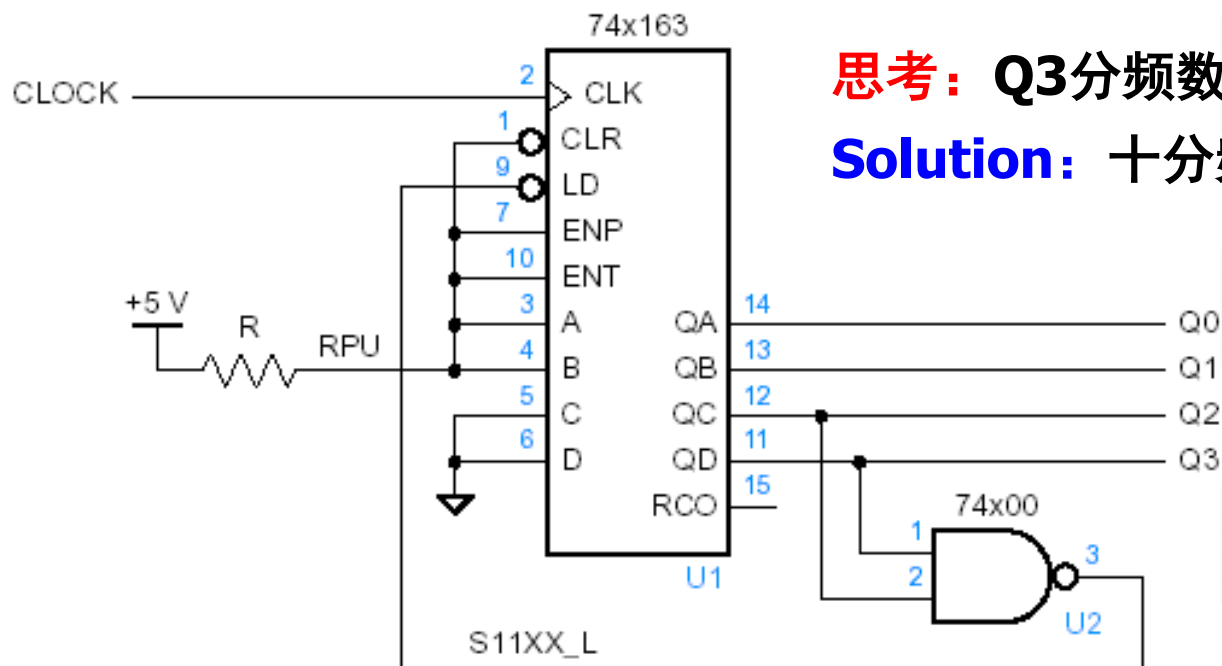


## Example 2: excess-3 code counter using 74x163



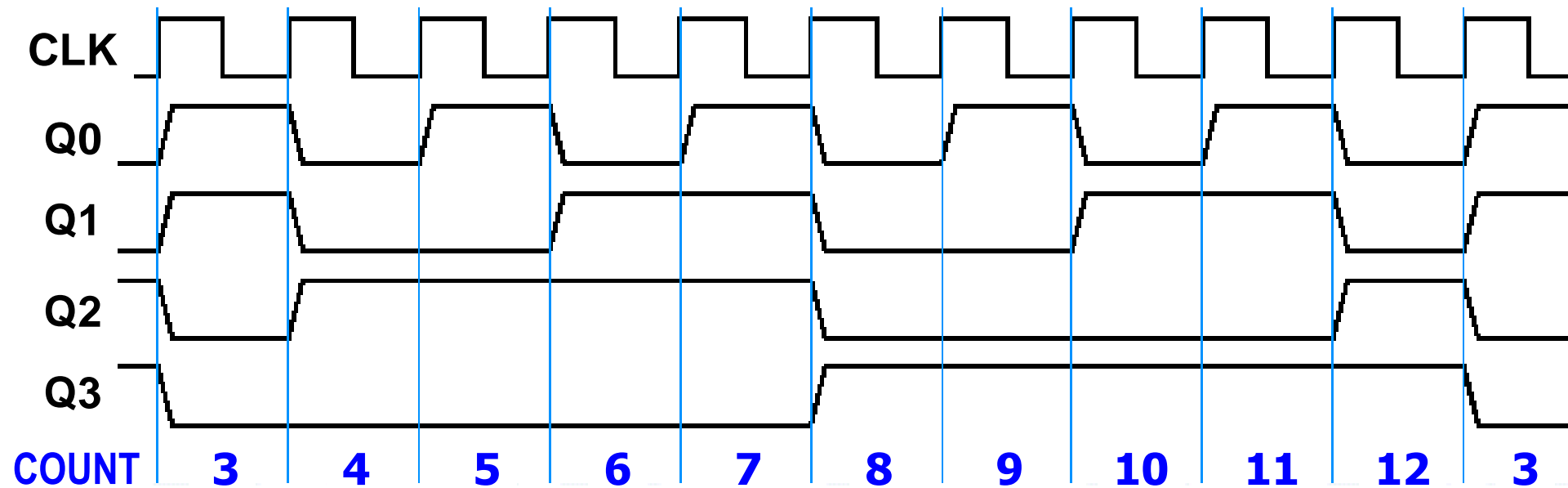






**思考：**Q3分频数？占空比？

**Solution：**十分频，占空比50%





# 思考：利用74x163实现2421码计数器？

十进制数	2421码
0	0000
1	0001
2	0010
3	0011
4	0100
5	1011
6	1100
7	1101
8	1110
9	1111

