

SKEE1223: Digital Electronics

12 – Counters and Registers

Dr Michael Tan Loong Peng
PhD (Cambridge)
Senior Lecturer
Faculty of Electrical Engineering
Universiti Teknologi Malaysia

Flip-Flop Applications

- ◆ Applications of Flip-Flop:-

- Frequency divider

- Counters

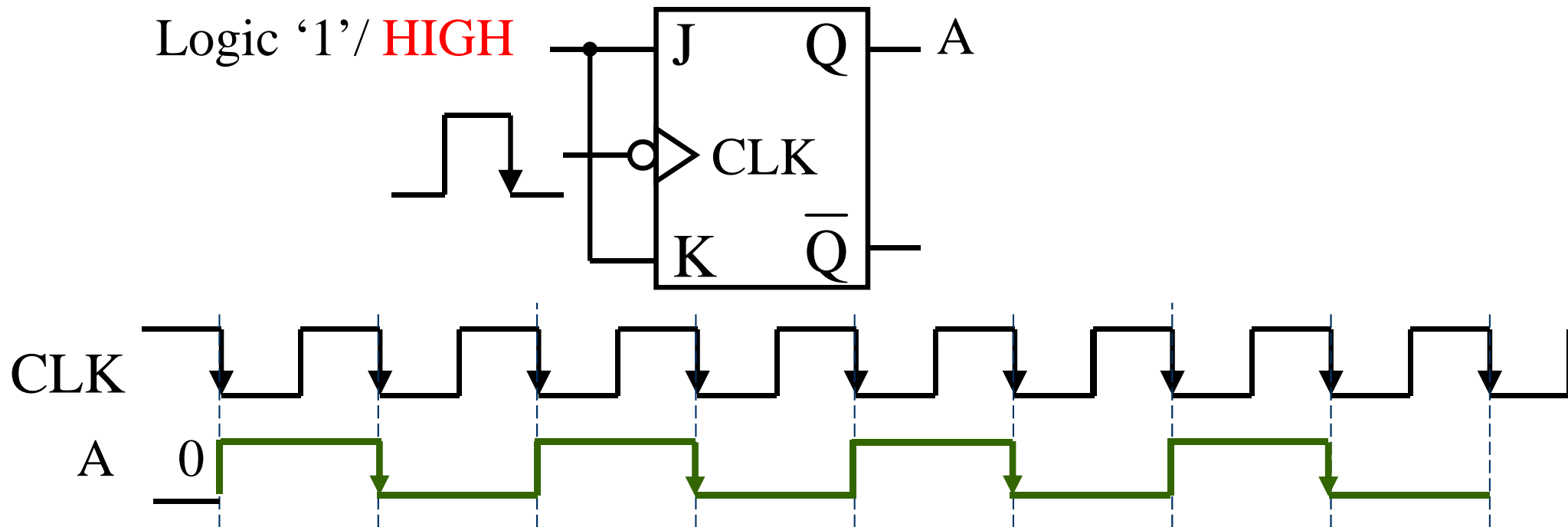
- Asynchronous Counter

- Synchronous Counter

- Register

Frequency Divider

- ◆ One J-K Flip-Flop is used with both inputs J and K are connected and set at Logic '1' or **HIGH** level

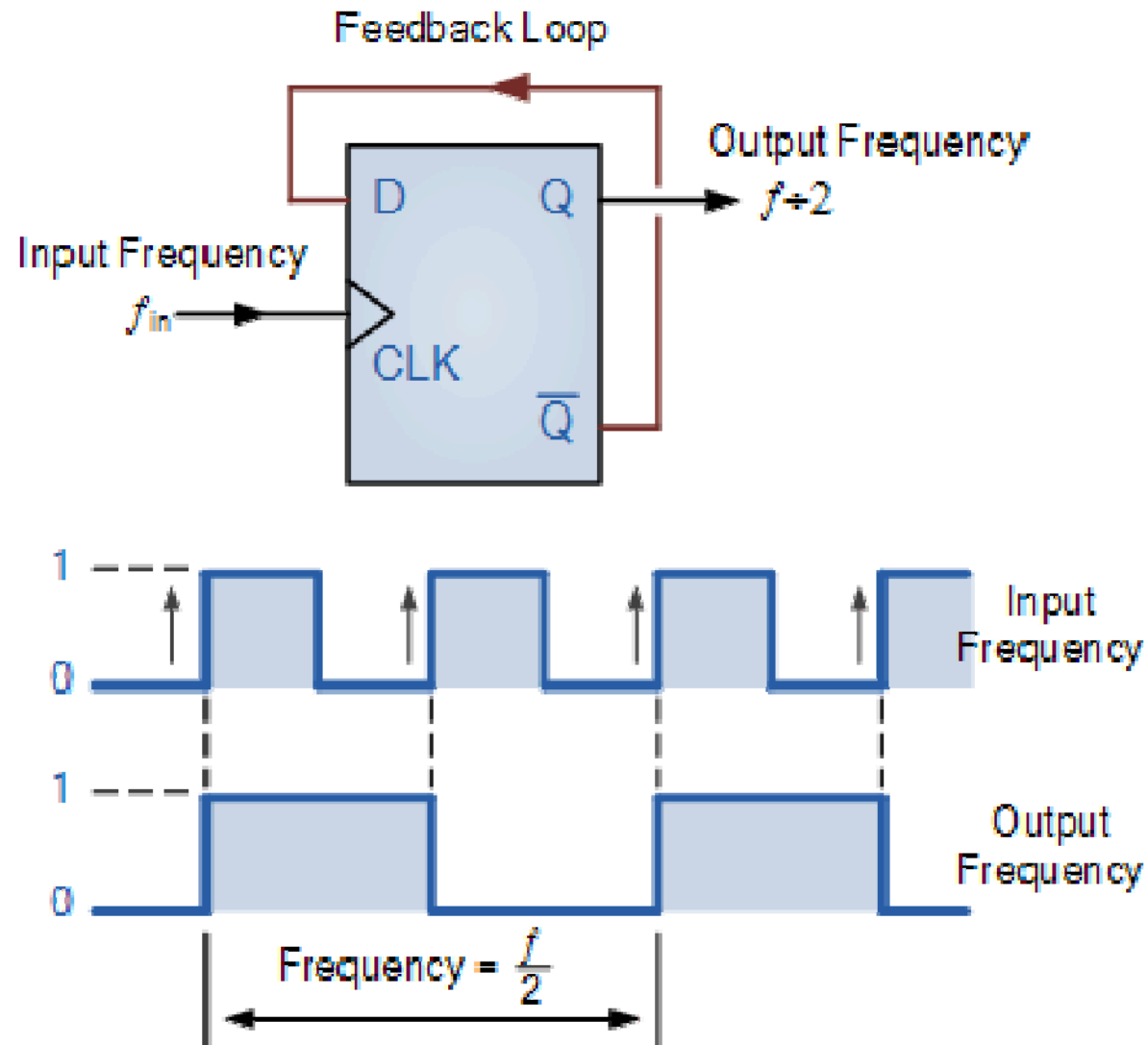


$$T_A = 2T_{CLK}$$

thus,

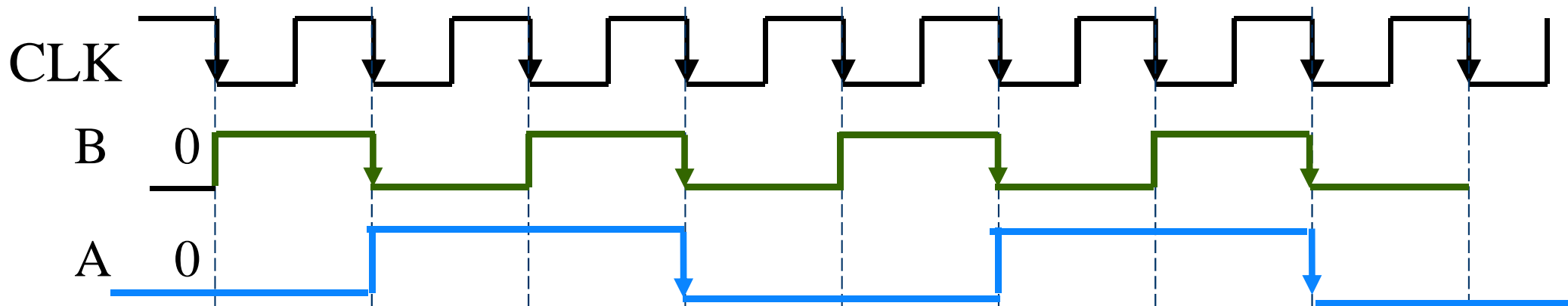
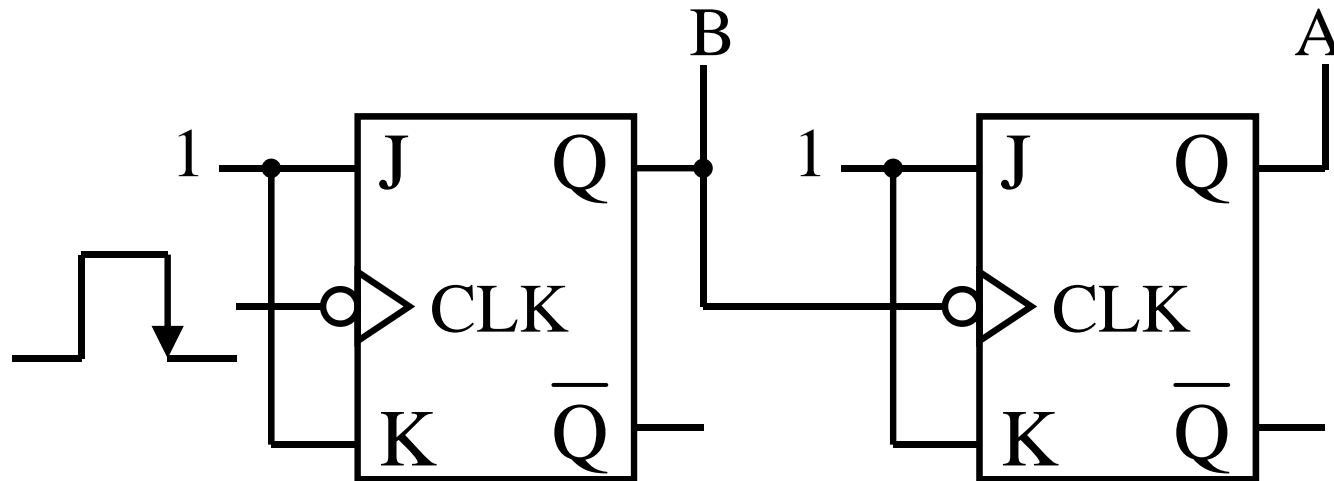
$$f_A = f_{CLK}/2$$

Frequency Divider (continue)



Frequency Divider (continue)

- ◆ When two J-K flip-flop is used



What are the relations between f_{CLK} , f_B and f_A ?

Counters

- ◆ A *counter* is a register that goes through a predetermined sequence of states upon the application of clock pulses.
 - Asynchronous counters
 - Synchronous counters
- ◆ **Async. counters** (or *ripple counters*), the clock signal (CLK) is only used to clock the first FF. Each successive FF is clocked by the preceding FF.
- ◆ **Sync. counters**, the clock signal (CLK) is applied to all FF, which means that all FF shares the same clock signal, thus the output will change at the same time.

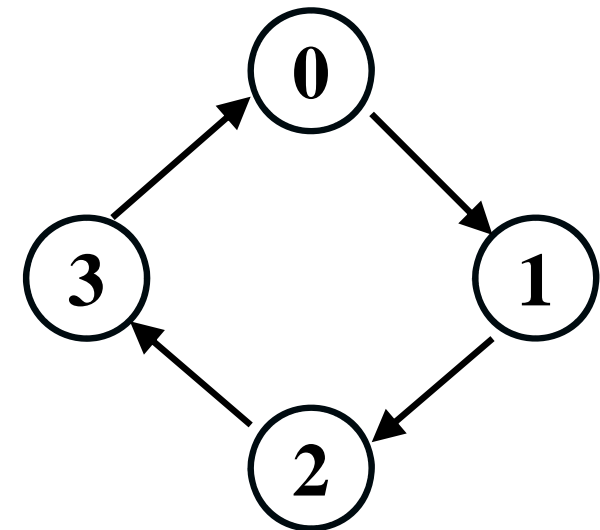
Asynchronous counters

- ◆ The async counter that counts 4 number $(00,01,10,11,00)_2$ starting from and back to $(00)_2$ is called **MOD-4 ripple up-counter**.
- ◆ **MOD number** is generally equal to the number of state it counts in a complete cycle before it goes back to the initial state.
- ◆ Thus, the number of flip-flop used depend on the MOD of the counter and also the number of bit used (ie; **MOD-4** use **2 FF (2-bit)**, **MOD-8** use **3 FF (3-bit)**, etc..)

Asynchronous Counters (continue)

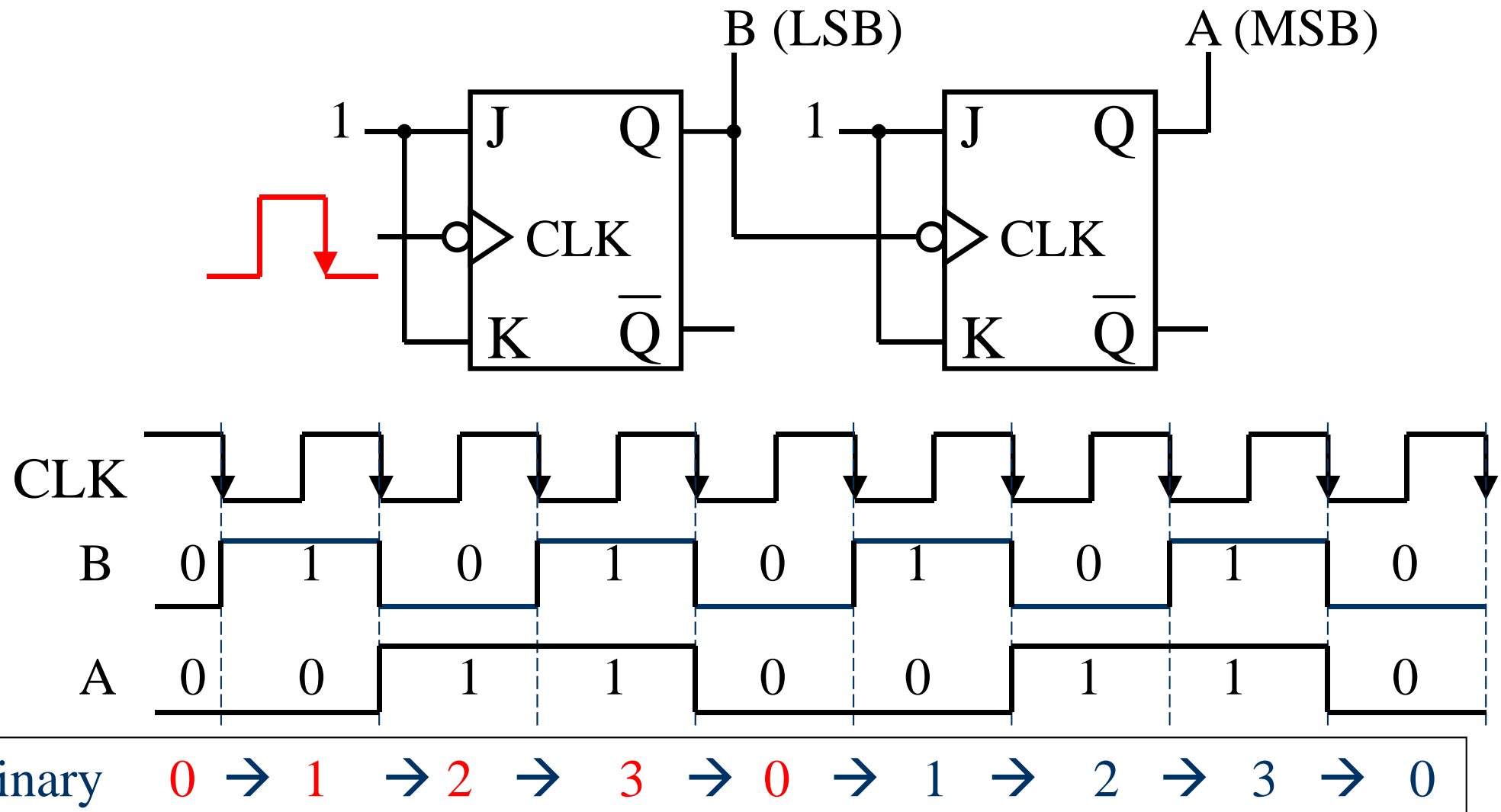
- ◆ **State sequence table and state transition diagram for a *MOD-4 Asynchronous up-counter***

Present State	CLK	Next State
AB	↓	AB
00	1	01
01	2	10
10	3	11
11	4	00



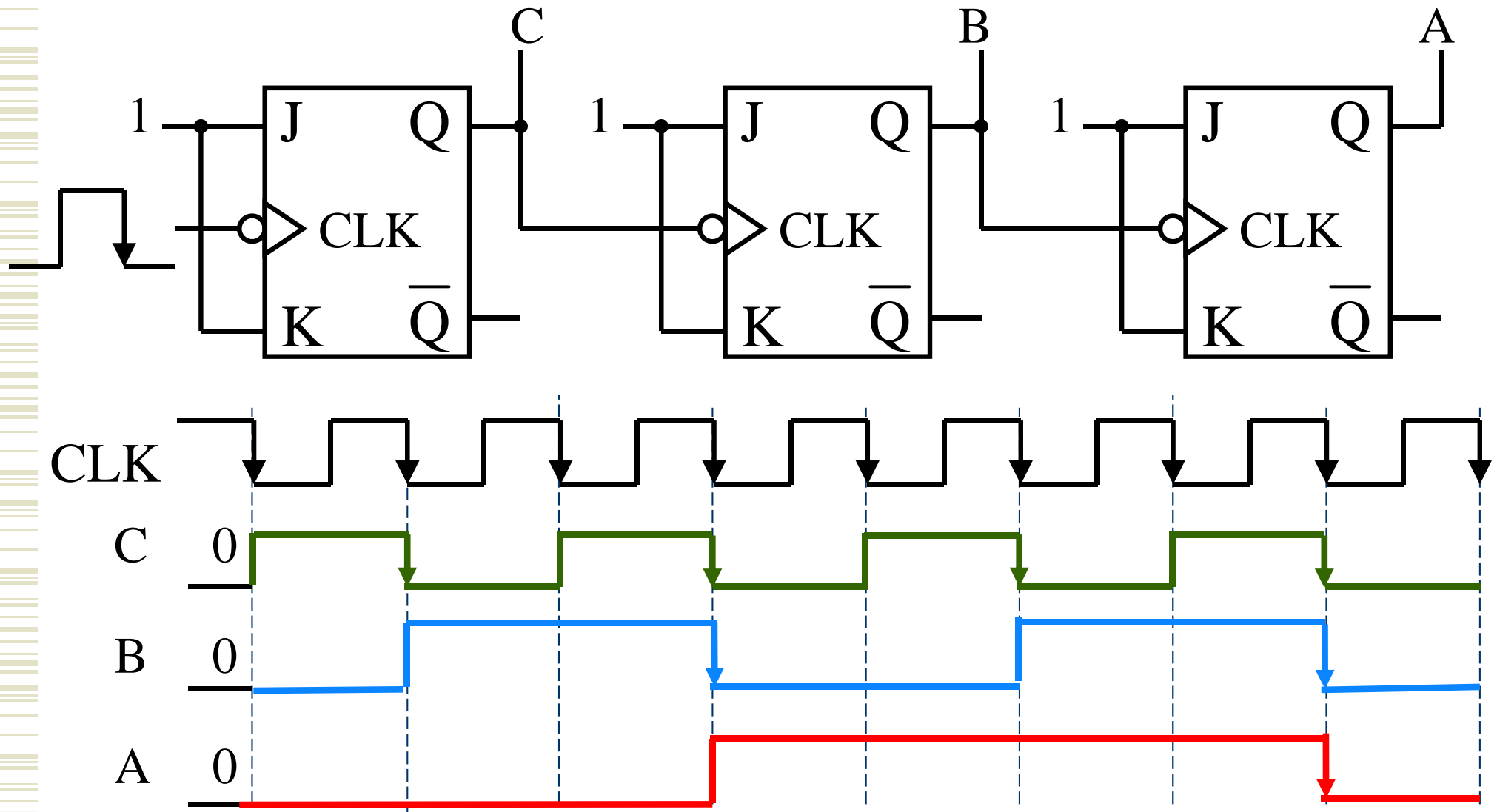
Asynchronous Counters (continue)

- ♦ MOD-4 Asynchronous up-counter



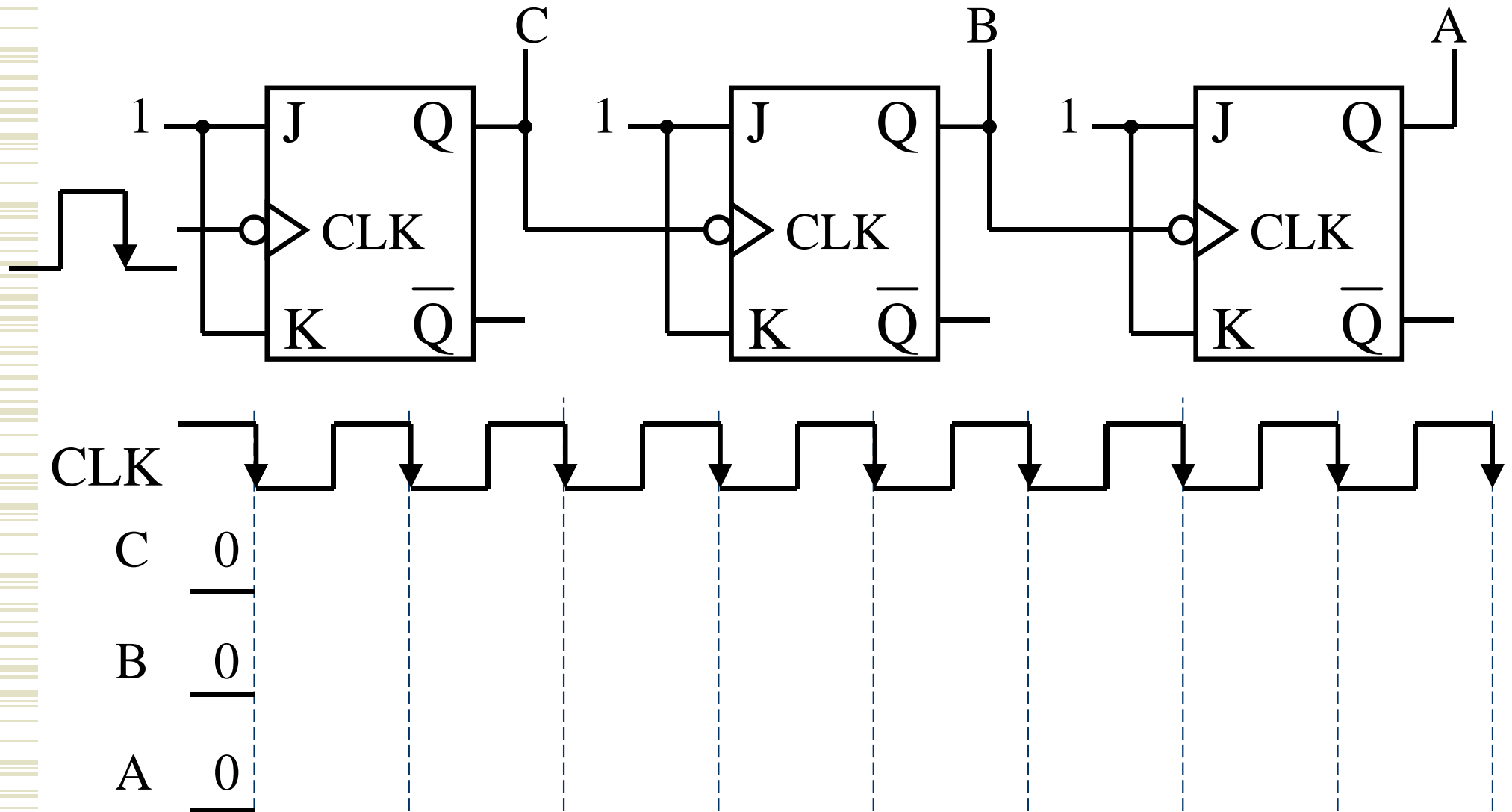
Asynchronous Counters (continue)

- ♦ MOD-8 Asynchronous up-counter



Asynchronous Counters (continue)

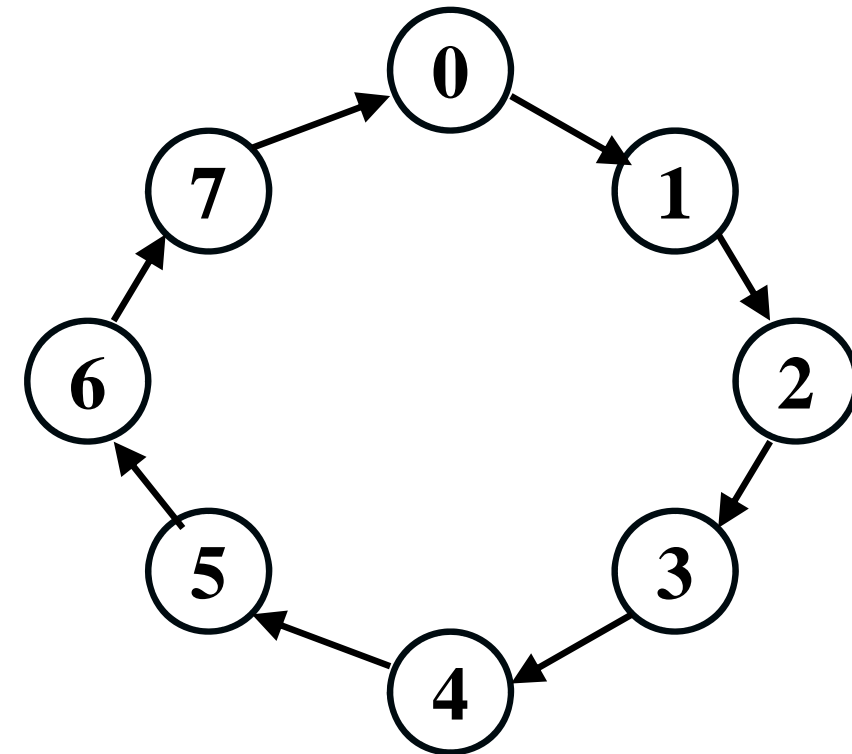
- ♦ MOD-8 Asynchronous up-counter



Asynchronous Counters (continue)

- ◆ State sequence table and state transition diagram

Present St.	CLK	Next St.
ABC		ABC
000	1	001
001	2	010
010	3	011
011	4	100
100	5	101
101	6	110
110	7	111
111	8	000



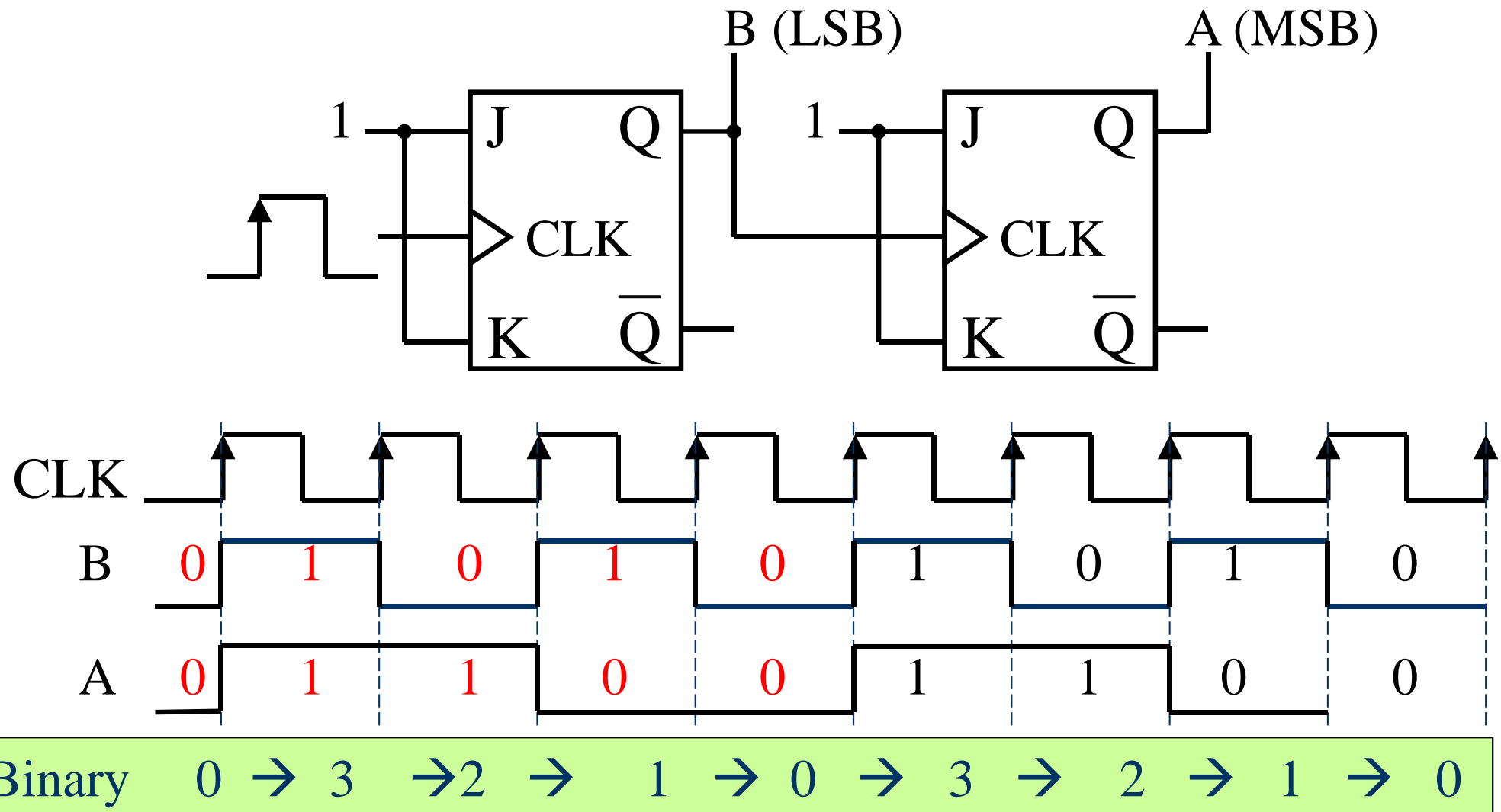
Asynchronous Counters (continue)

◆ Exercise 1:

- Design a MOD-16 ripple up-counter
- Design a MOD-4 ripple down-counter
- Design a MOD-8 ripple down counter
- Design a MOD-16 ripple down counter

Asynchronous Counters (continue)

- ◆ **2-bit** Asynchronous Binary counter



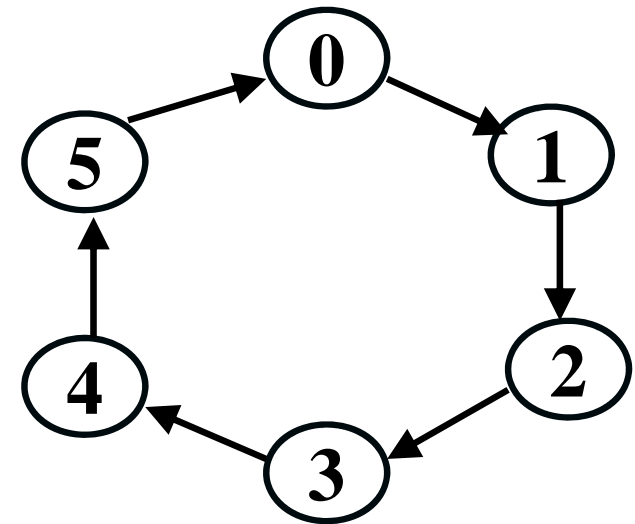
Asynchronous Counters (continue)

- ◆ So far, we have design the counters with MOD number equal to 2^N , where **N** is the number of **bit** ($N = 1,2,3,4\dots$) (also correspond to number of **FF**)
- ◆ Thus, the counters are limited on for counting **MOD-2**, **MOD-4**, **MOD-8**, **MOD-16** etc..
- ◆ The question is how to design a **MOD-5**, **MOD-6**, **MOD-7**, **MOD-9** which is not a **MOD- 2^N** (**MOD $\neq 2^N$**) ?
- ◆ MOD-6 counters will count from 0_{10} (000_2) to 5_{10} (101_2) and after that will recount back to 0_{10} (000_2) continuously.

Asynchronous Counters

- ♦ MOD-6 ripple **up-counter** ($\text{MOD} \neq 2^N$)

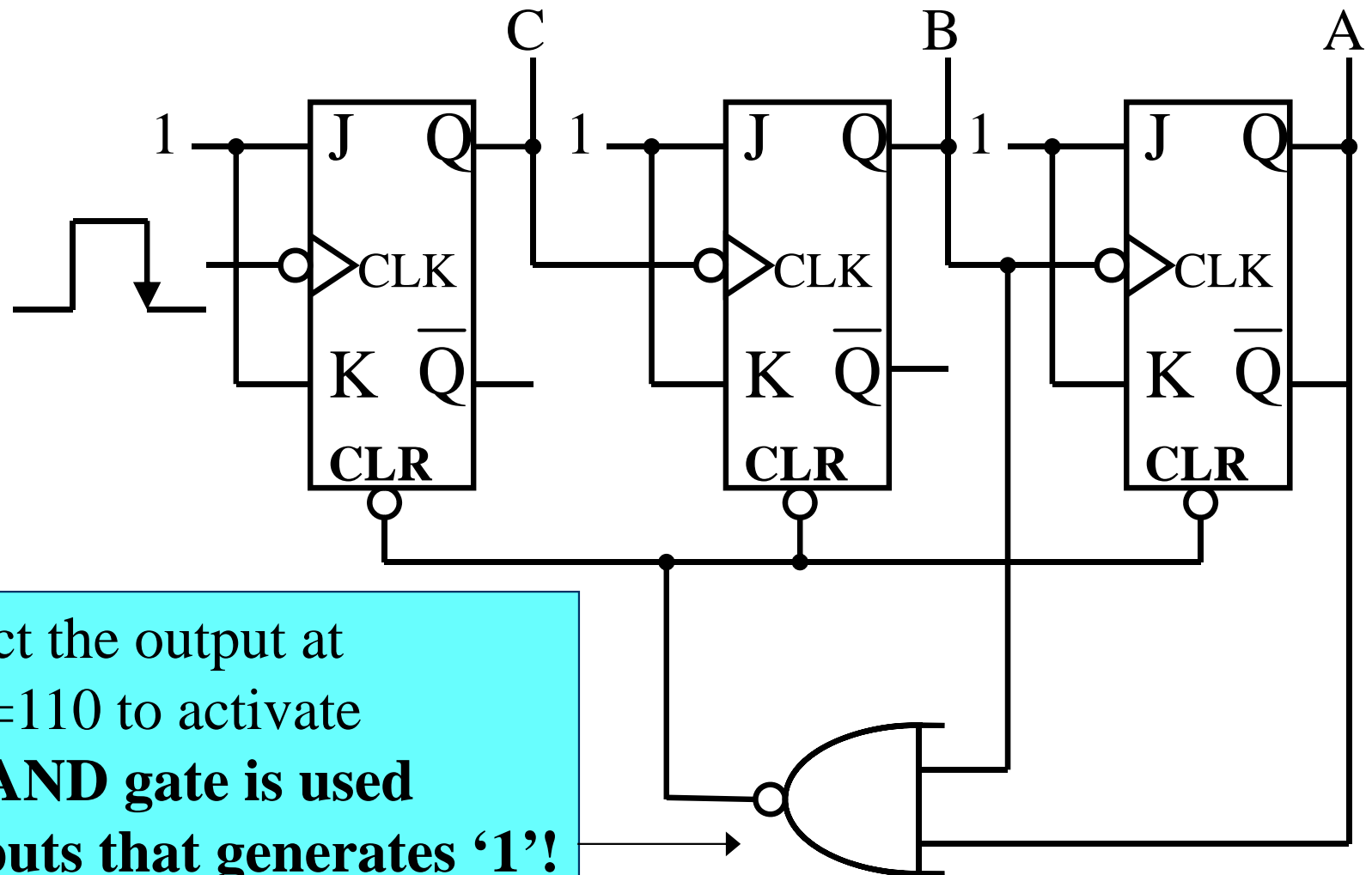
Present St.	CLK	Next St.
ABC	↓	ABC
000	1	001
001	2	010
010	3	011
011	4	100
100	5	101
101	6	000(110)



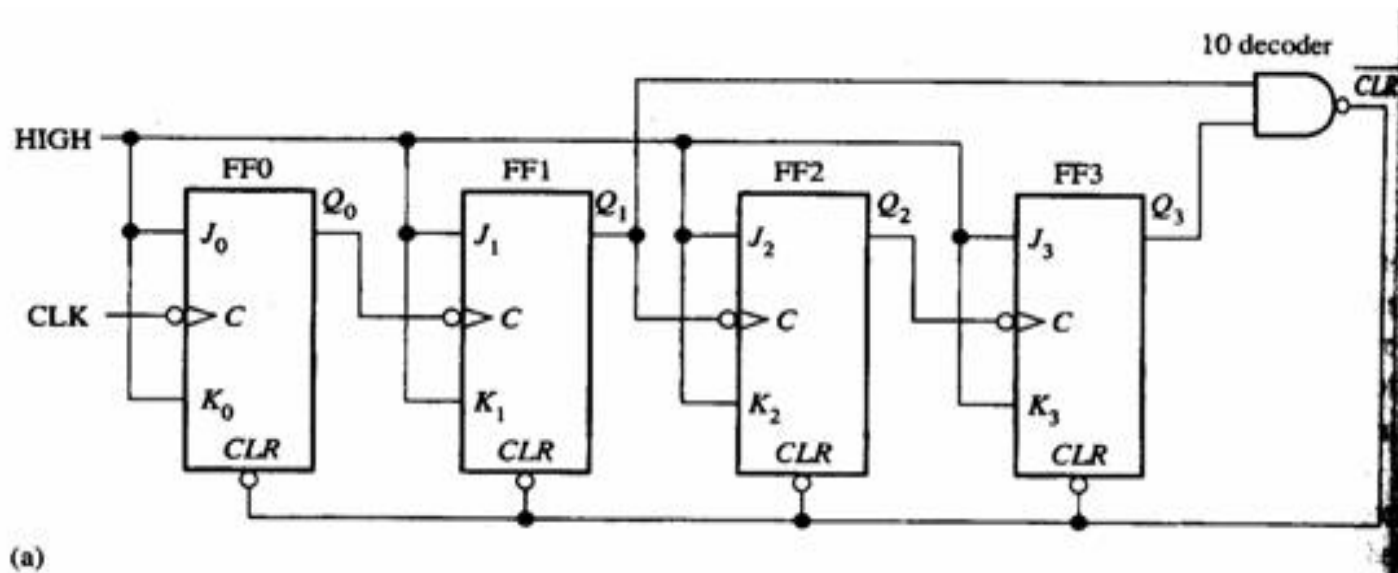
Reset the state to 000_2
when 110_2 is detected

Asynchronous Counters (continue)

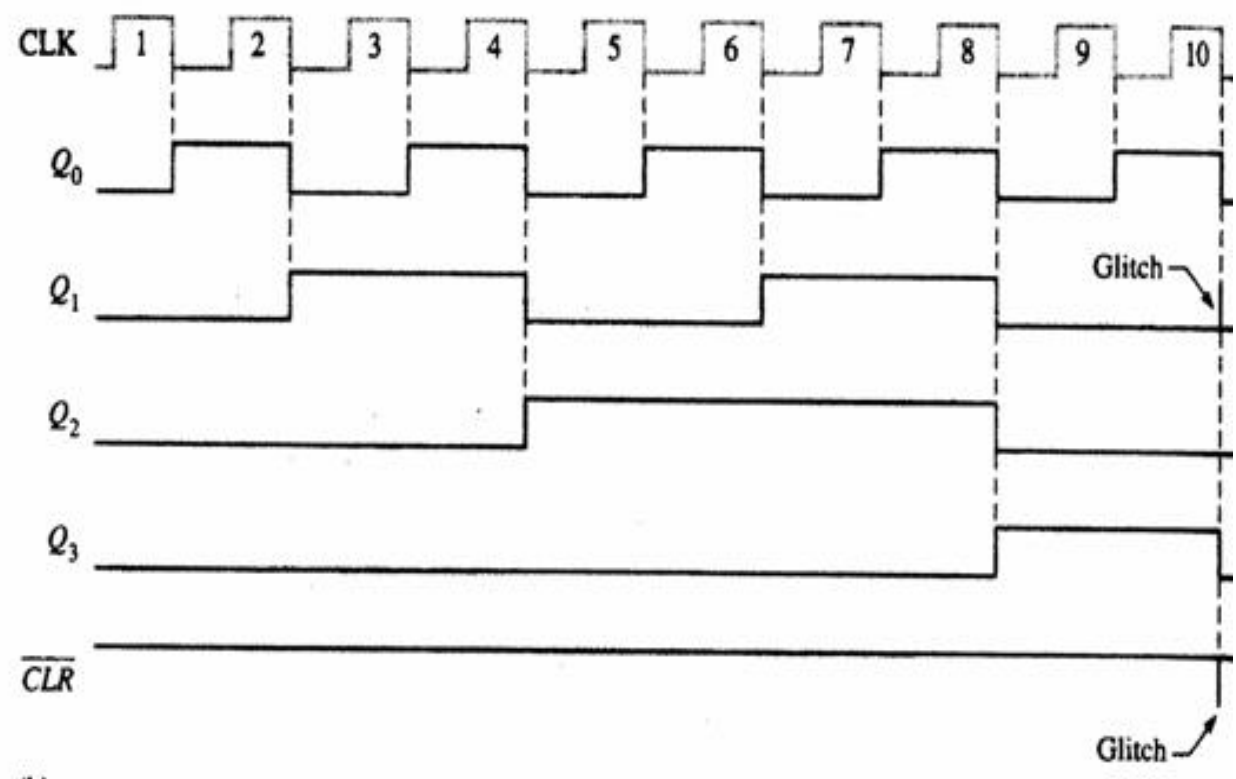
- ◆ Circuit diagram for MOD-6 ripple up-counter (MOD $\neq 2^N$)



Detect the output at ABC=110 to activate CLR. NAND gate is used to detect outputs that generates '1'!



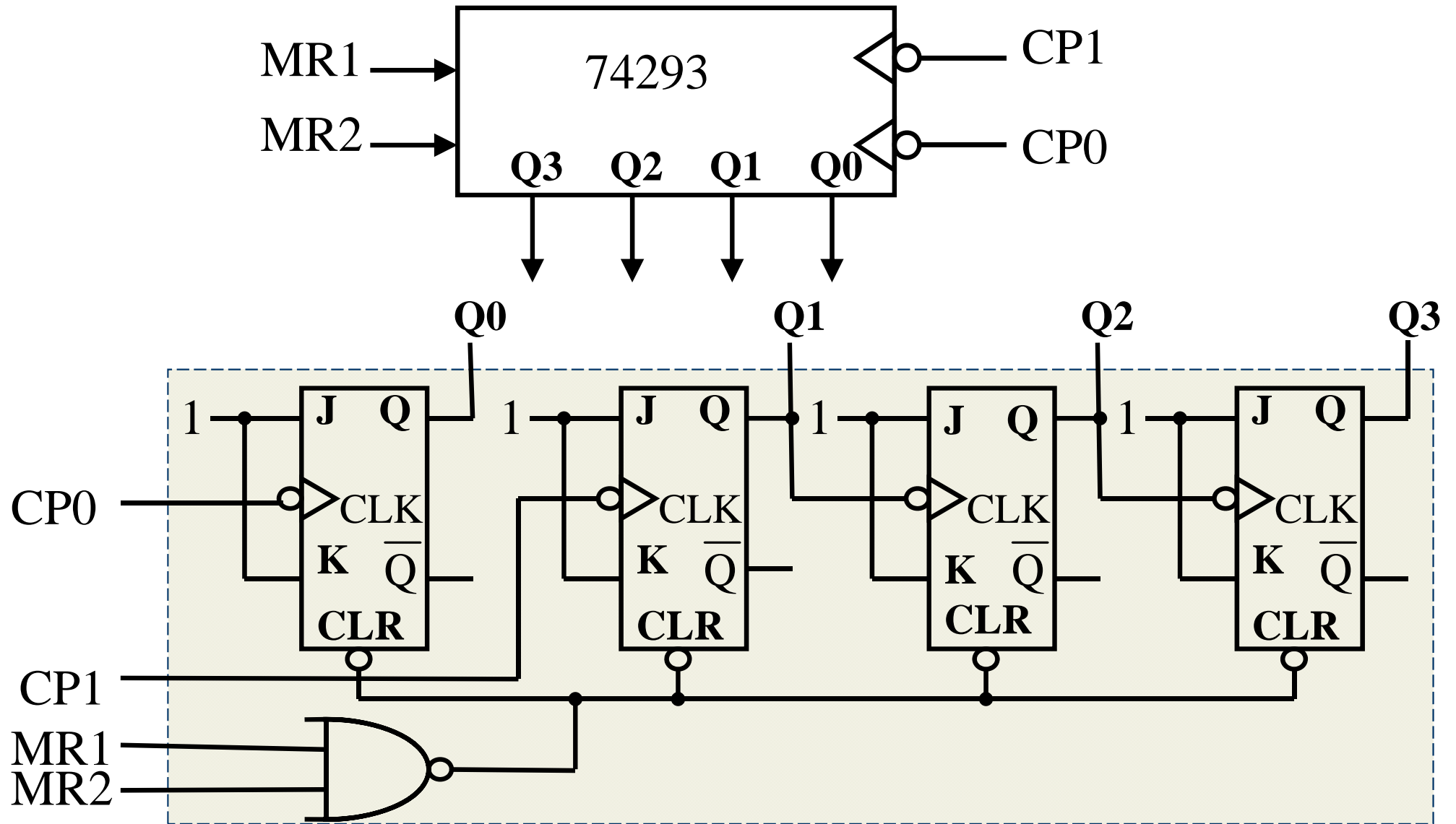
(a)



(b)

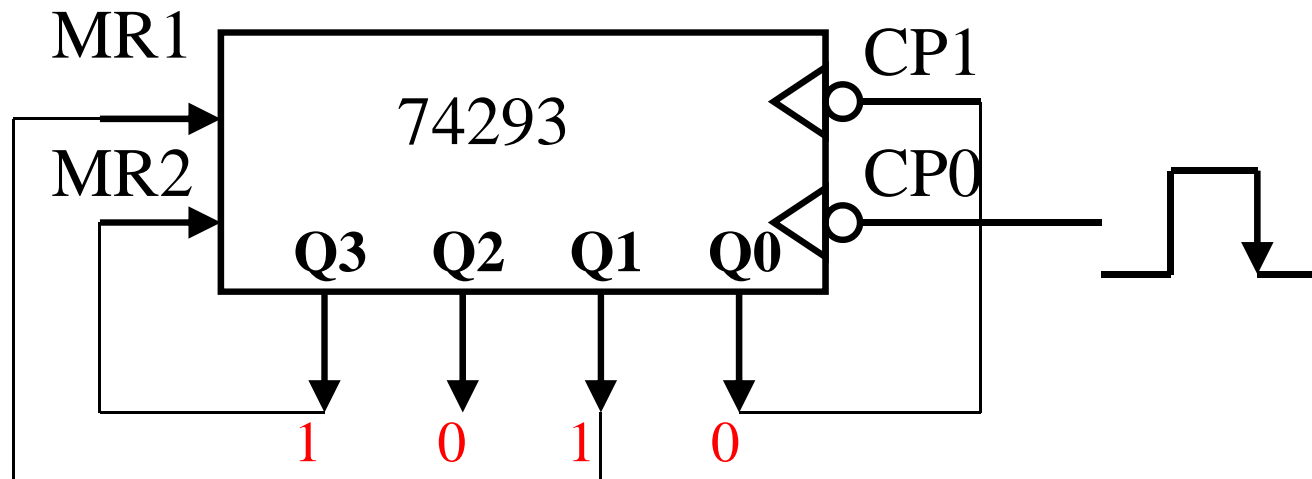
Chip for Asynchronous counters

- ◆ 74293 IC for Asynchronous counter with Reset (MR1 and MR2)



Chip for Asynchronous counters (continue)

- ◆ Using 74293 IC to design $\text{MOD} \leq 16$ asynchronous up-counter!
- ◆ **Exercise 2:**
 - use 74293 IC to design MOD-10 ripple up-counter



Asynchronous counters

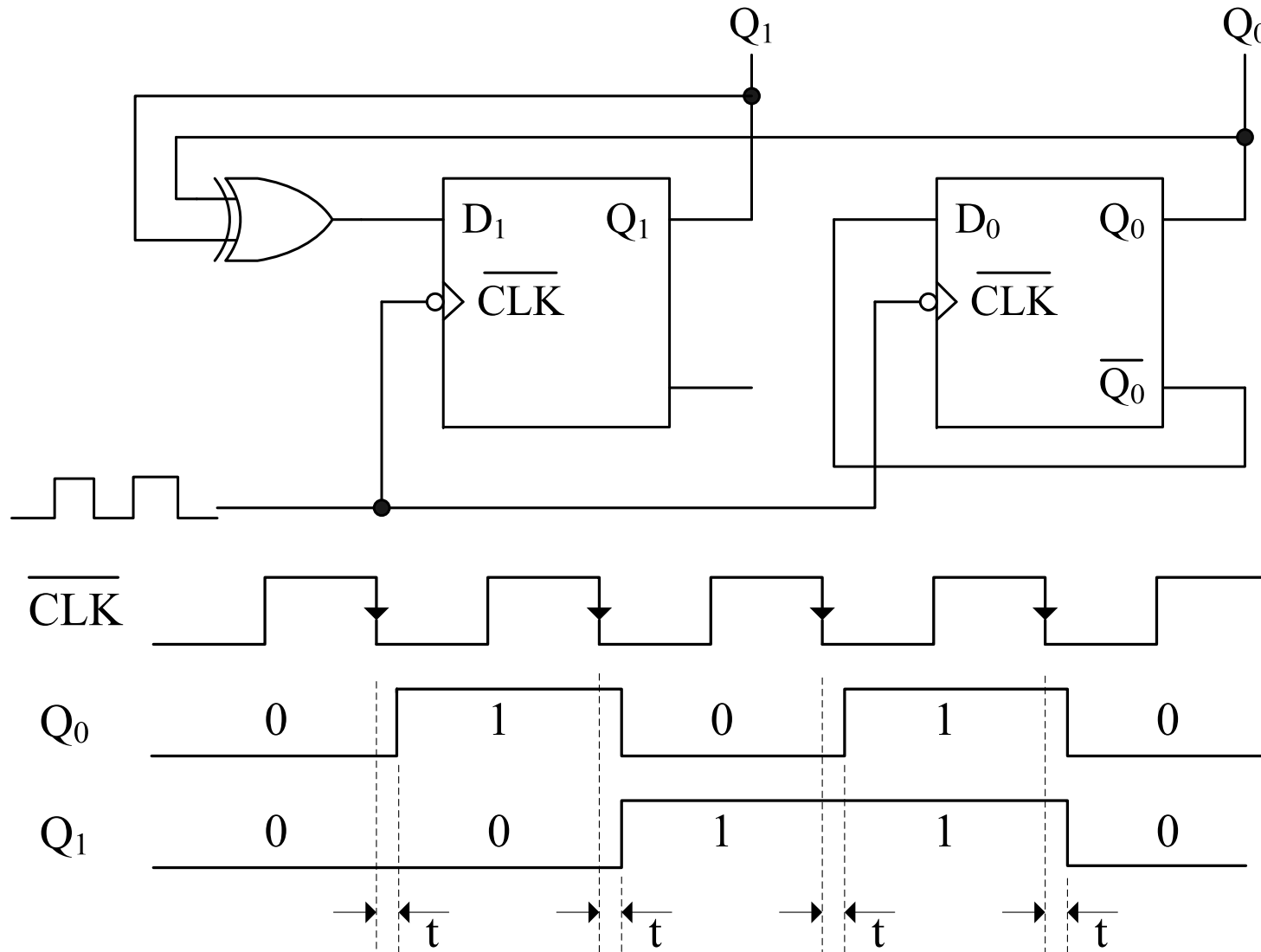
- ◆ Disadvantages of Asynchronous Counters:-
 - **Propagation delay** is severe for larger MOD of counters, especially at the MSB.
 - Existence of '**glitch**' is inevitable for $\text{MOD} \neq 2^N$ counters.
 - Difficult to design random counters (i.e:- to design circuit that counts numbers in these sequence
 $5 \rightarrow 6 \rightarrow 7 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 5 \rightarrow 6 \dots$)
- ◆ Solution, use *SYNCHRONOUS COUNTERS*.

Synchronous counters

- ◆ For synchronous counters, all the flip-flops are using the same CLOCK signal. Thus, the output would change synchronously.
- ◆ Procedure to design any counter circuit is clearly stated.
- ◆ Procedure to design synchronous counter are as follows:-
 - Obtain the **State Transition Diagram**.
 - Obtain the **Excitation Table** using state transition table for any particular FF. Determine number of FF used.
 - Obtain and simplify the function of each FF input using **K-Map**.
 - Draw the **circuit**.

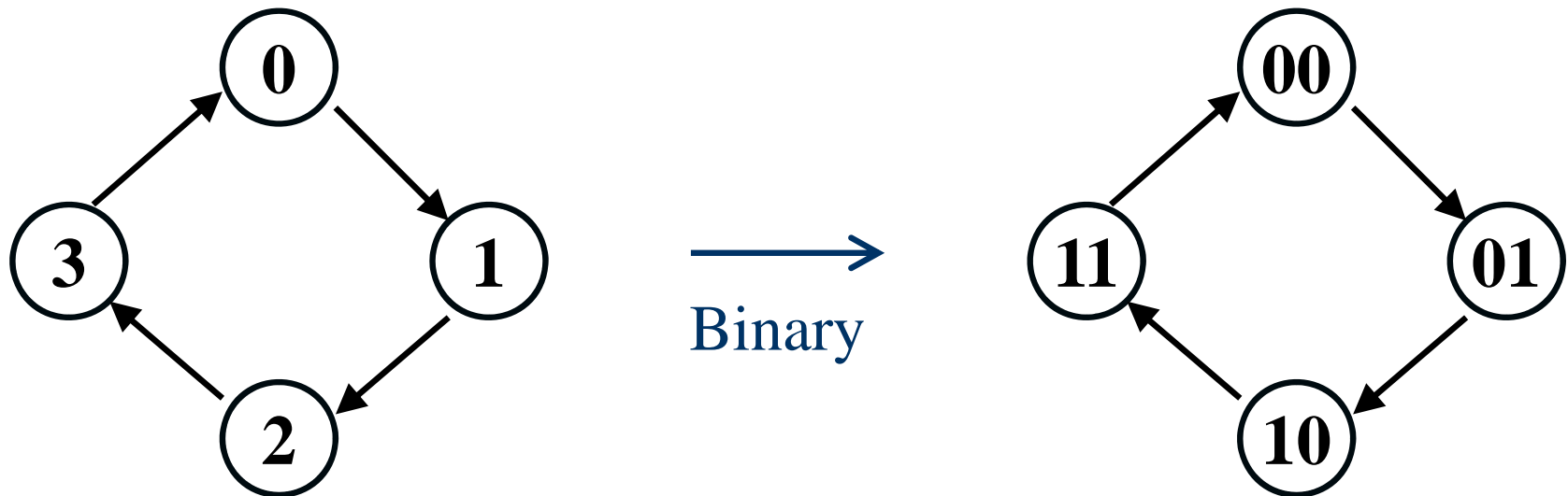
Synchronous counters

- ◆ Example : Sketch the output waveforms of Q0 and Q1.



Synchronous counters

- ◆ Example : Design a **MOD-4** sync **up-counter**, using JK FF?
 - Obtain the State transition Diagram



Synchronous counters

- Obtain the Excitation table. Two JK FF are used.

OUTPUT TRANSITION			FF INPUT	
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

Present St.	Next St.		
A B	A B	$J_A K_A$	$J_B K_B$
0 0	0 1	0 X	1 X
0 1	1 0	1 X	X 1
1 0	1 1	X 0	1 X
1 1	0 0	X 1	X 1

Synchronous counters

- Obtain the simplified function using K-Map

		B	
		0	1
A	0	0	1
	1	X	X

$J_A = B$

		B	
		0	1
A	0	X	X
	1	0	1

$K_A = B$

		B	
		0	1
A	0	1	X
	1	1	X

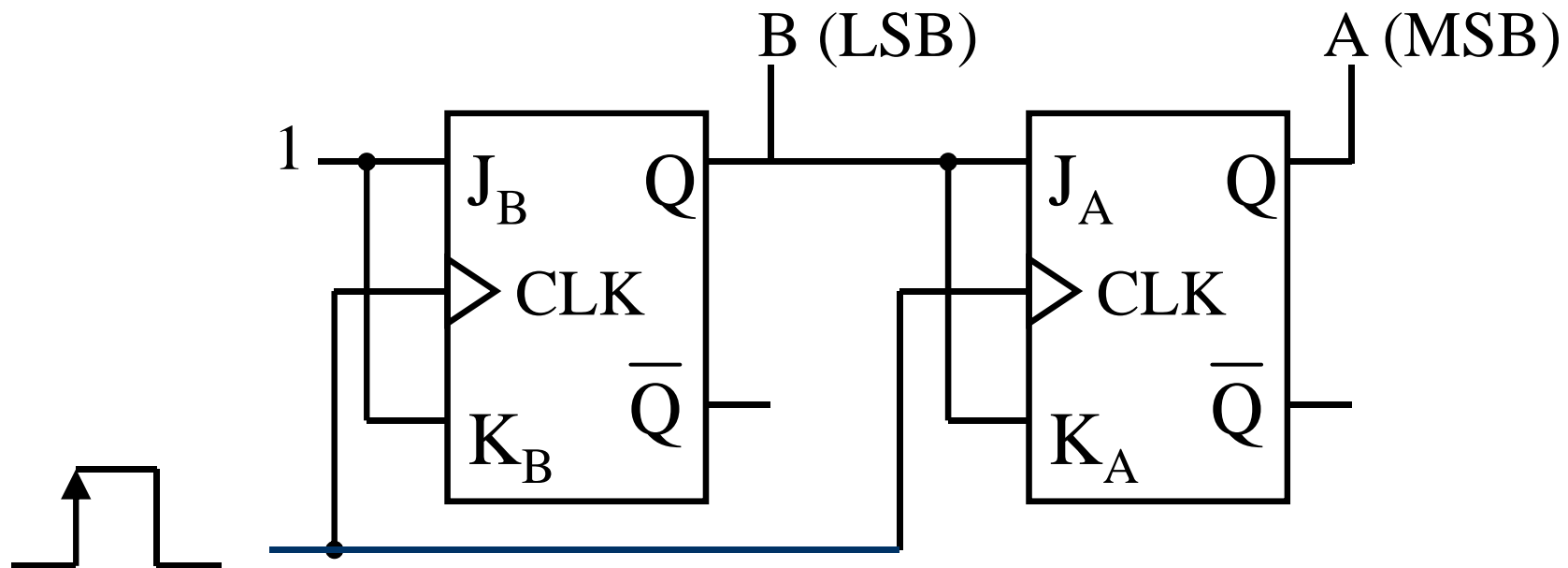
$J_B = 1$

		B	
		0	1
A	0	X	1
	1	X	1

$K_B = 1$

Synchronous counters

- Draw the circuit diagram



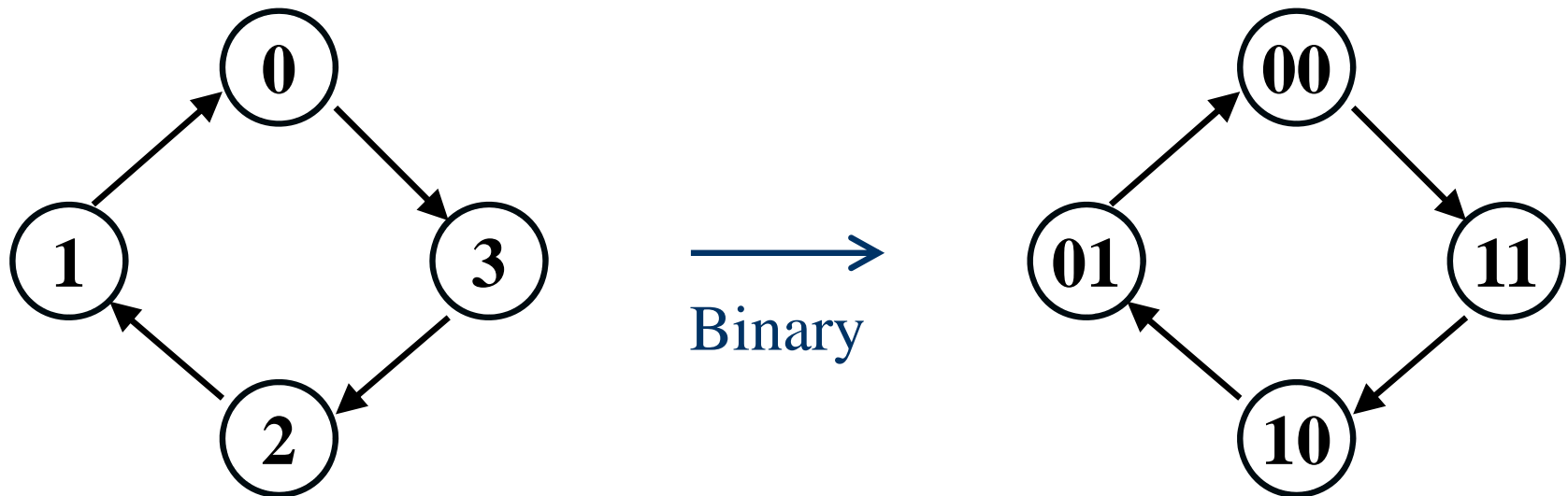
Synchronous counters

◆ Exercise 4:-

- Design MOD-4 sync up-counter using D flip-flop.
- Design MOD-8 sync up-counter using D flip-flop.
- Design MOD-8 sync up-counter. Use T FF for MSB, D FF for second bit and JK FF for LSB.
- Design MOD-16 sync up-counter using T flip-flop.

Synchronous counters

- ◆ **Exercise 5:** Design a MOD-4 sync down-counter, using JK FF?
 - Obtain the State transition Diagram



Synchronous counters

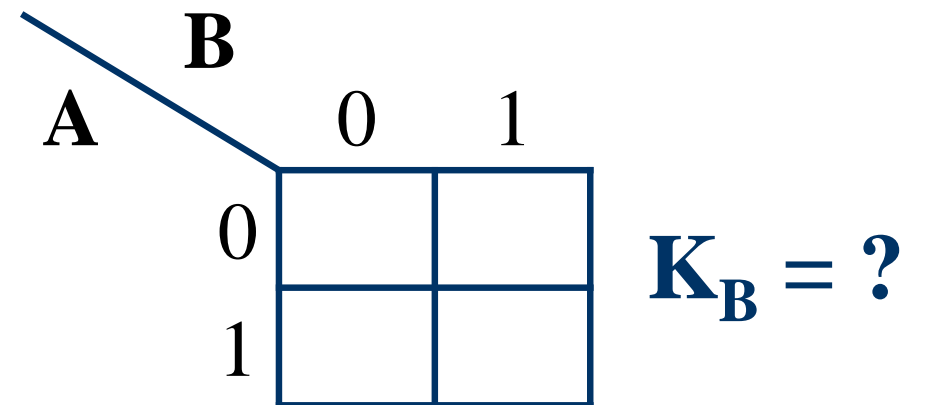
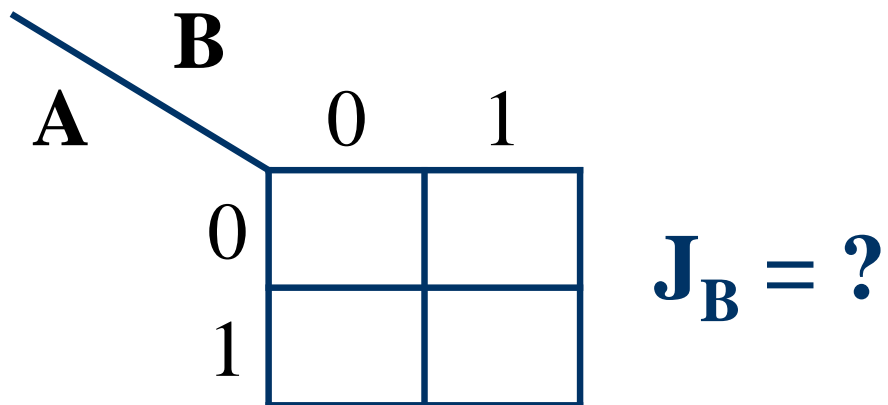
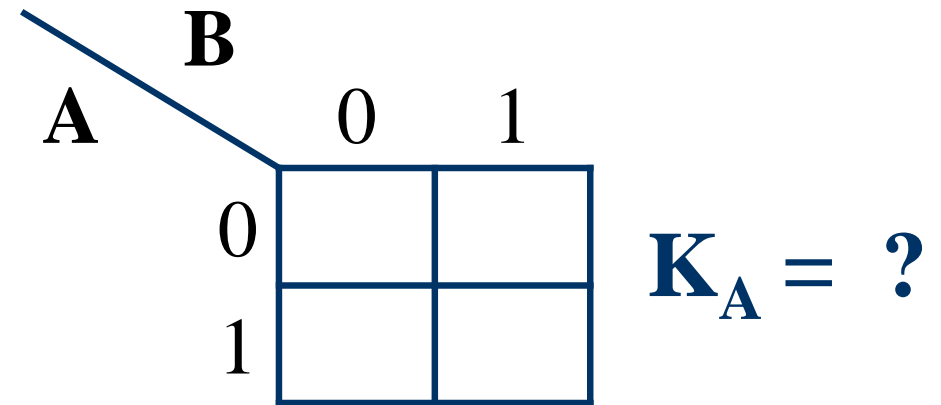
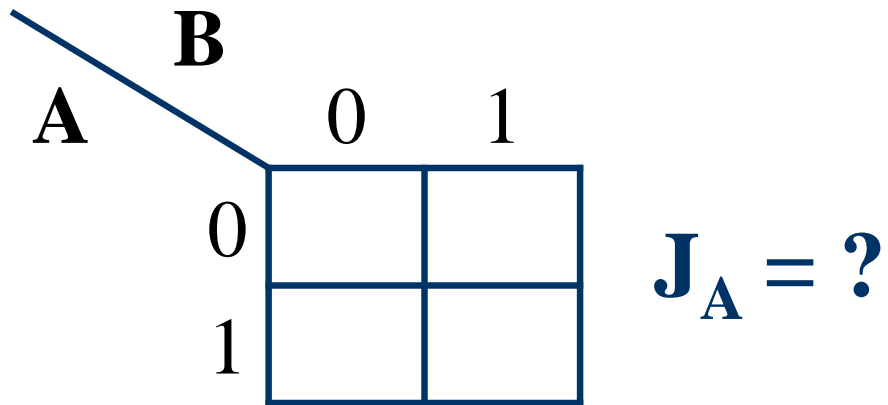
- Obtain the Excitation table. Two JK FF are used.

OUTPUT TRANSITION		FF INPUT	
Q_N	Q_{N+1}	J	K
0	→ 0	0	X
0	→ 1	1	X
1	→ 0	X	1
1	→ 1	X	0

Present St.	Next St.	Flip Flop	
A B	A B	$J_A K_A$	$J_B K_B$
0 0	1 1	1 1	X X
0 1	0 0	0 X	X 1
1 0	0 1	X 1	1 X
1 1	1 0	X X	0 1

Synchronous counters

- Obtain the simplified function using K-Map



Synchronous counters

- Obtain the simplified function using K-Map

		B	
		0	1
A	0	1	0
	1	X	X

$$J_A = B'$$

		B	
		0	1
A	0	1	X
	1	1	X

$$K_A = 1$$

		B	
		0	1
A	0	X	X
	1	1	0

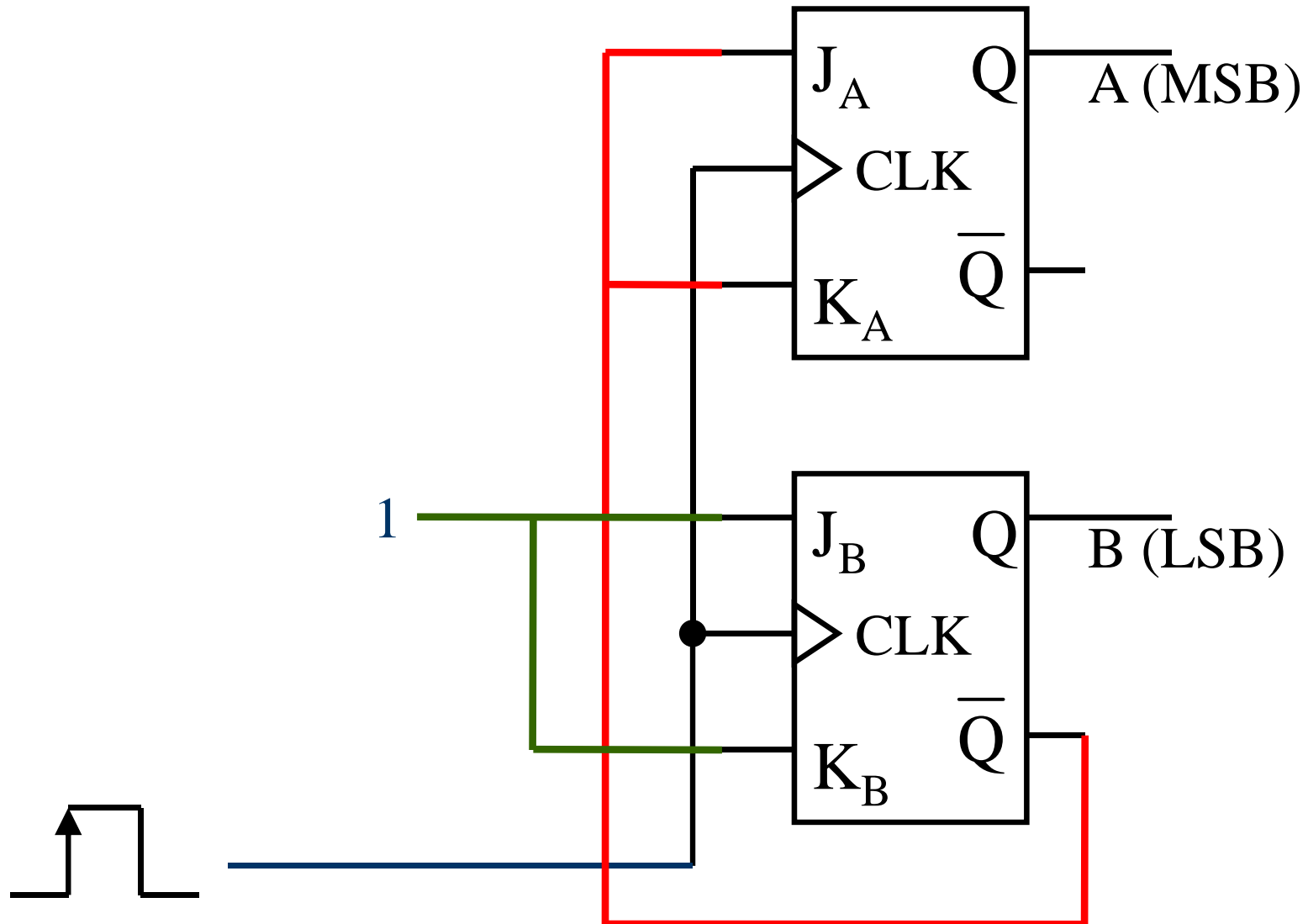
$$J_B = B'$$

		B	
		0	1
A	0	X	1
	1	X	1

$$K_B = 1$$

Synchronous counters

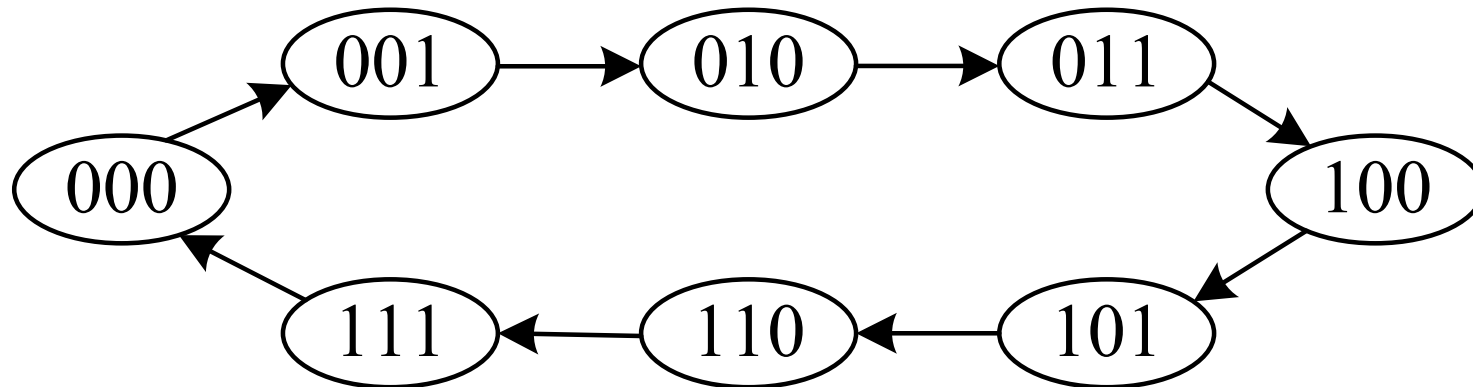
- Draw the circuit diagram



Synchronous counters

◆ Exercise 6 (Part 1):-

■ Design a 3-bit up counter using D flip-flop.

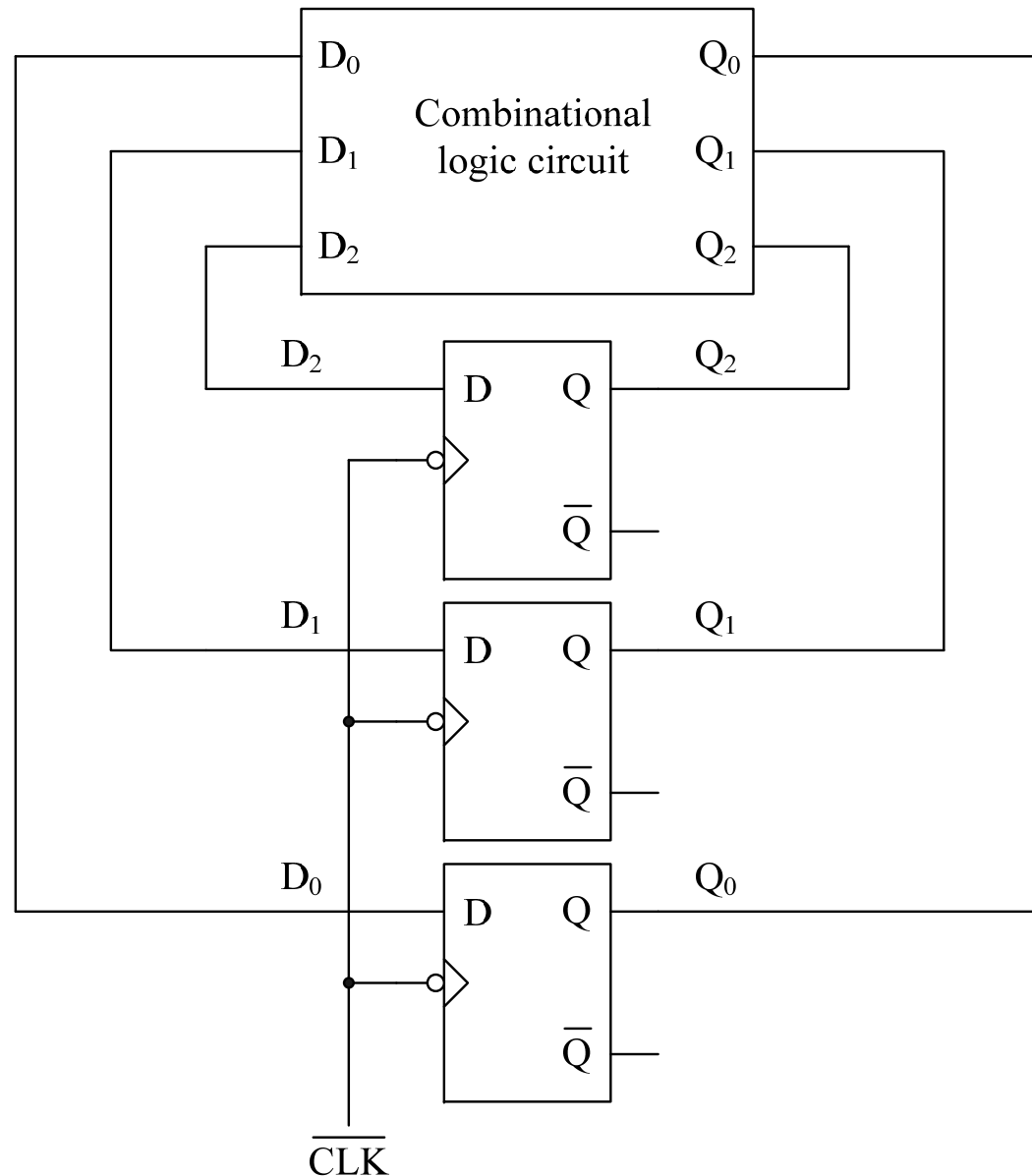


Synchronous counters

- ◆ **Solution Step 1:-**
- ◆ *How to design an automatic counter?*
- ◆ The outputs of the flip-flops are connected to a combinational logic circuit where the outputs of this combinational logic circuit ($D_2D_1D_0$) will give the next output values

Synchronous counters

◆ Solution Step 1:-



Synchronous counters

◆ Solution Step 2:-

Combinational logic circuit Truth-table

Present State			Next State			Output		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀	D ₂	D ₁	D ₀
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0

Synchronous counters

- ◆ **Solution Step 3:-** From the truth table you can use either way to solve:
 - ◆ (1) Use K-map, get simplified expression, draw circuit for this combinational logic
 - ◆ (2) Use MSI (decoder / multiplexers) to implement the output of this truth table.

Synchronous counters

- ◆ **Solution Step 4:-**
- ◆ **Draw the FULL circuit!**

Synchronous counters

◆ Exercise 7:-

- Design MOD-4 sync down-counter using D flip-flop.
- Design MOD-8 sync down-counter using D flip-flop.
- Design MOD-8 sync down-counter. Use T FF for MSB, D FF for second bit and JK FF for LSB.
- Design MOD-16 sync down-counter using T flip-flop.

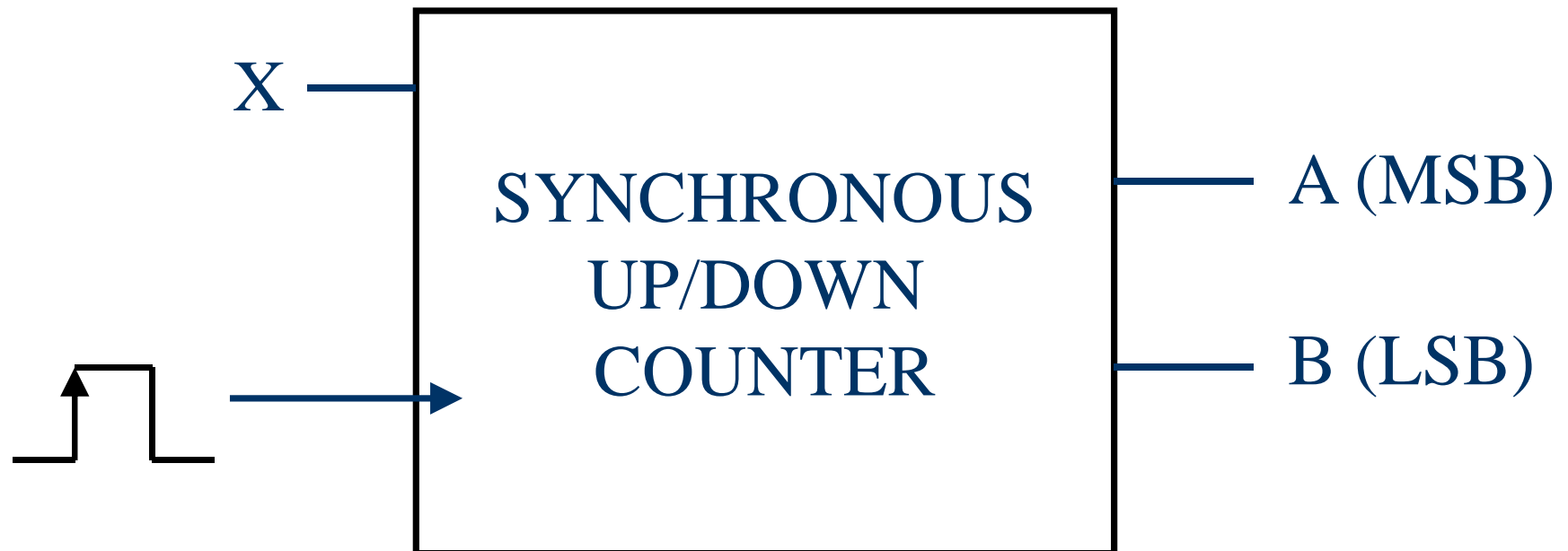
Synchronous counters

- ◆ Example: Design a MOD-4 sync UP/DOWN-counter, using D FF?

- Block diagram of the circuit.

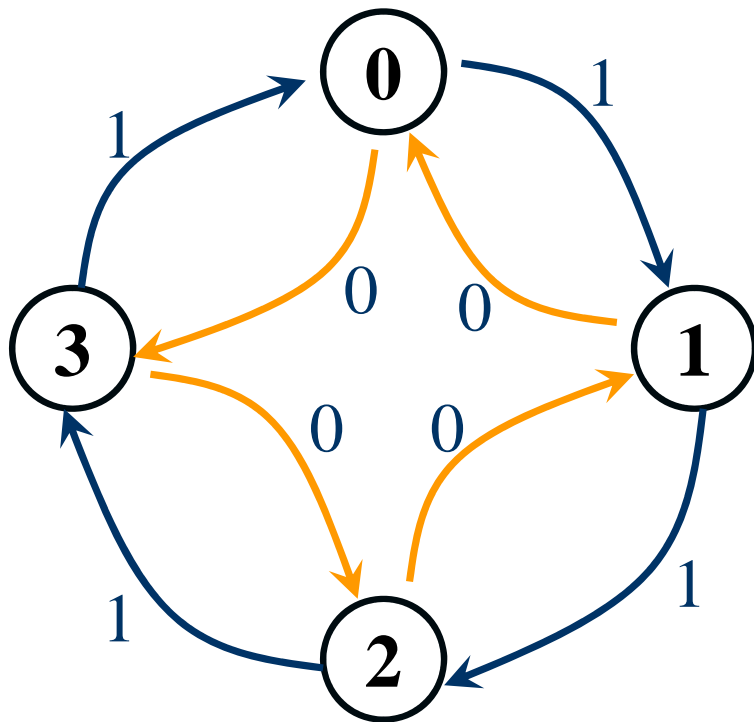
#when $X=0$, AB will COUNT-DOWN

#when $X=1$, AB will COUNT-UP

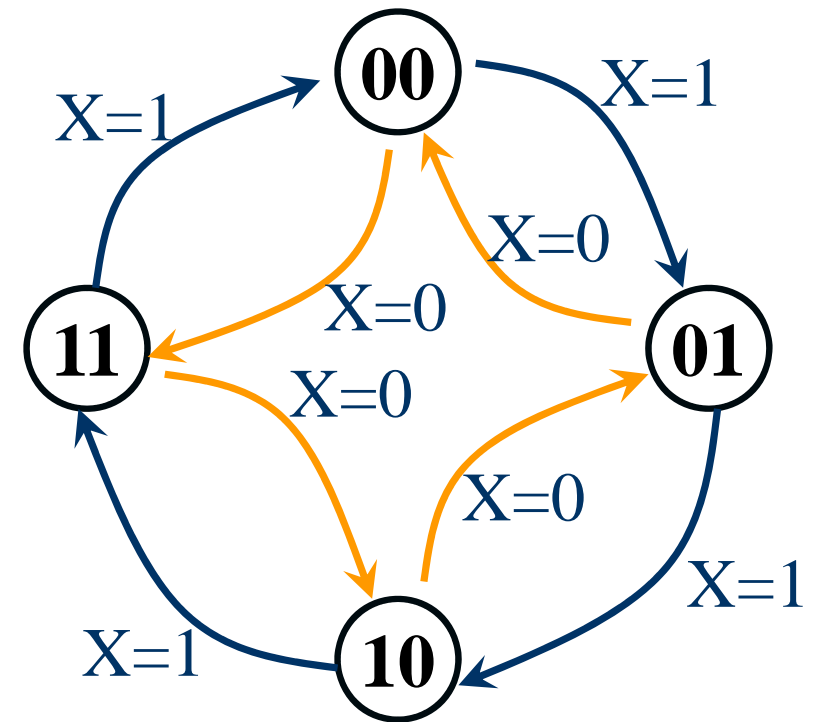


Synchronous counters

- Obtain the State Transition Diagram



Binary



Synchronous counters

- Obtain the Excitation table. Two D FF are used.

X	Present St.		→	Next St.		D_A	D_B
	A	B		A	B		
0	0	0	→	1	1	1	1
0	0	1	→	0	0	0	0
0	1	0	→	0	1	0	1
0	1	1	→	1	0	1	0
1	0	0	→	0	1	0	1
1	0	1	→	1	0	1	0
1	1	0	→	1	1	1	1
1	1	1	→	0	0	0	0

Synchronous counters

- Obtain the simplified function using K-Map

X \ AB	00	01	11	10
0	1	0	1	0
1	0	1	0	1

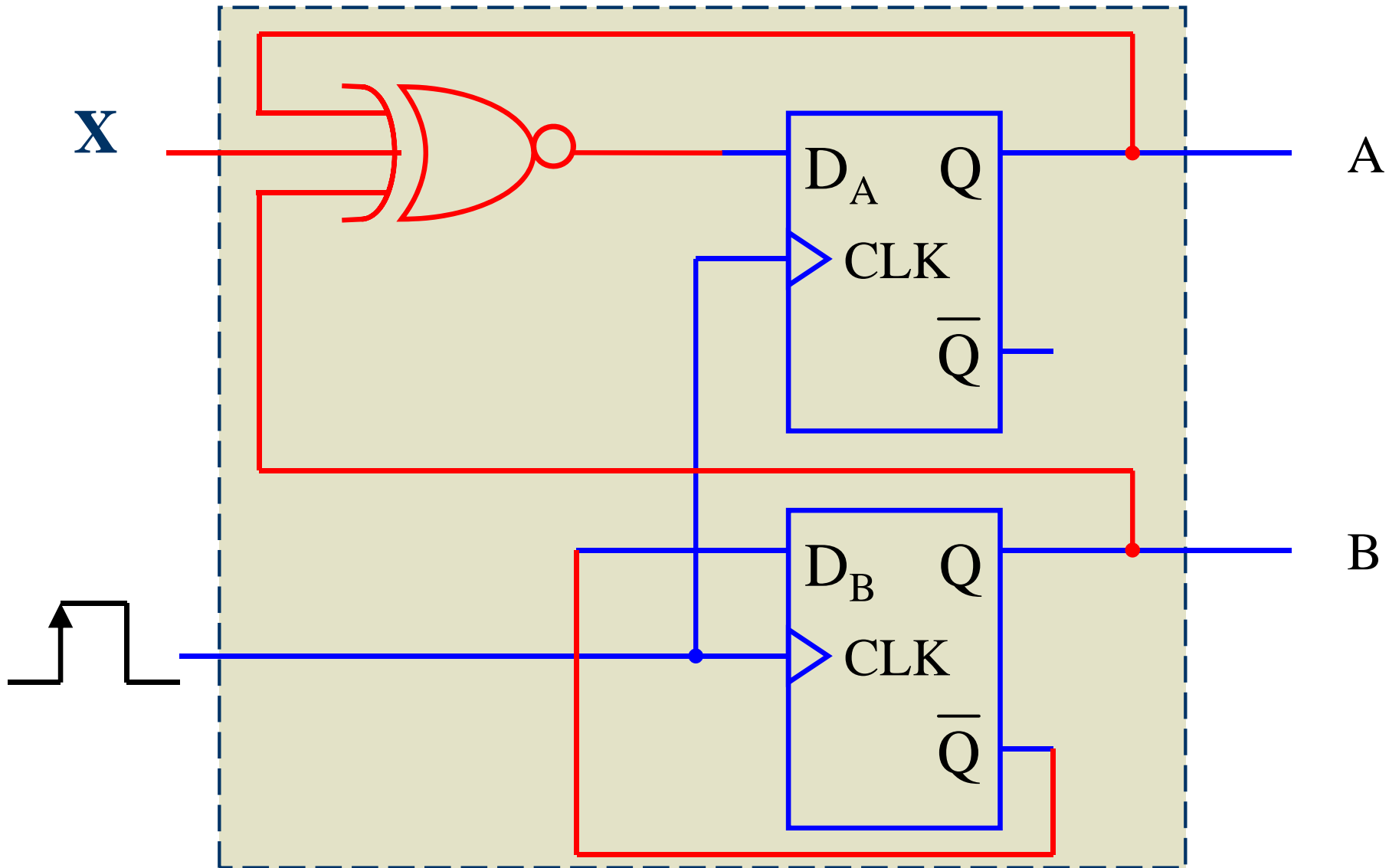
$$D_A = \overline{X \oplus A \oplus B}$$

X \ AB	00	01	11	10
0	1	0	0	1
1	1	0	0	1

$$D_B = \bar{B}$$

Synchronous counters

- Draw the circuit diagram



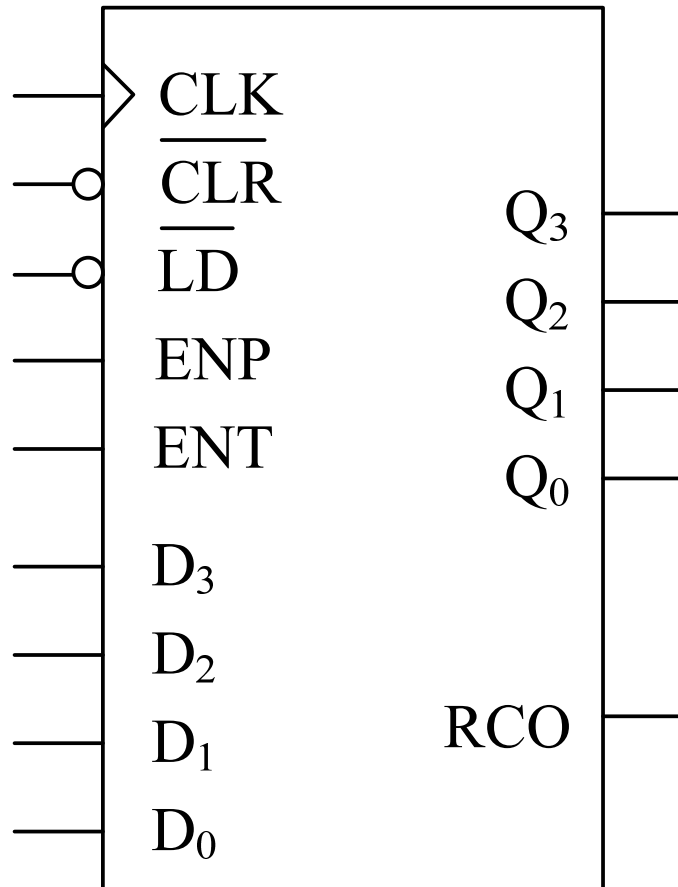
Synchronous counters

◆ Exercise 8:-

- Design **MOD-4** sync UP/DOWN-counter using **JK flip-flop**.
- Design **MOD-8** sync UP/DOWN-counter using **D flip-flop**.
- Design **MOD-8** sync UP/DOWN-counter. Use **T FF** for MSB, **D FF** for second bit and **JK FF** for LSB.
- Design a circuit that counts numbers in these sequence
5 → **6** → **7** → **2** → **3** → **1** → **5** → **6** → **7** → **2** → **3** → **1** → **5** → **6**.....

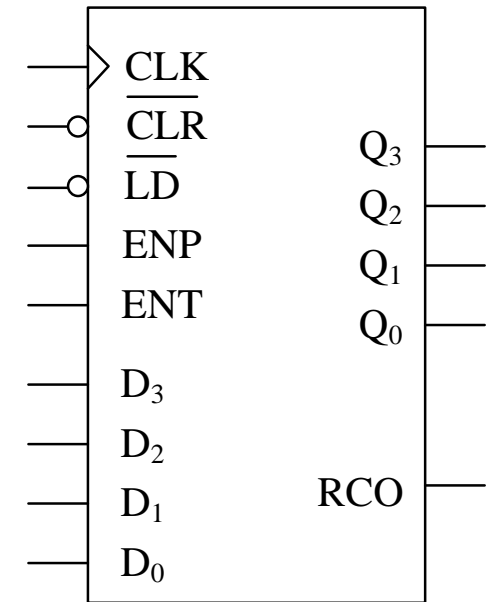
Chip for Synchronous counters

- ◆ 74163 IC for Synchronous counter with Load, CLR and ripple-carry output



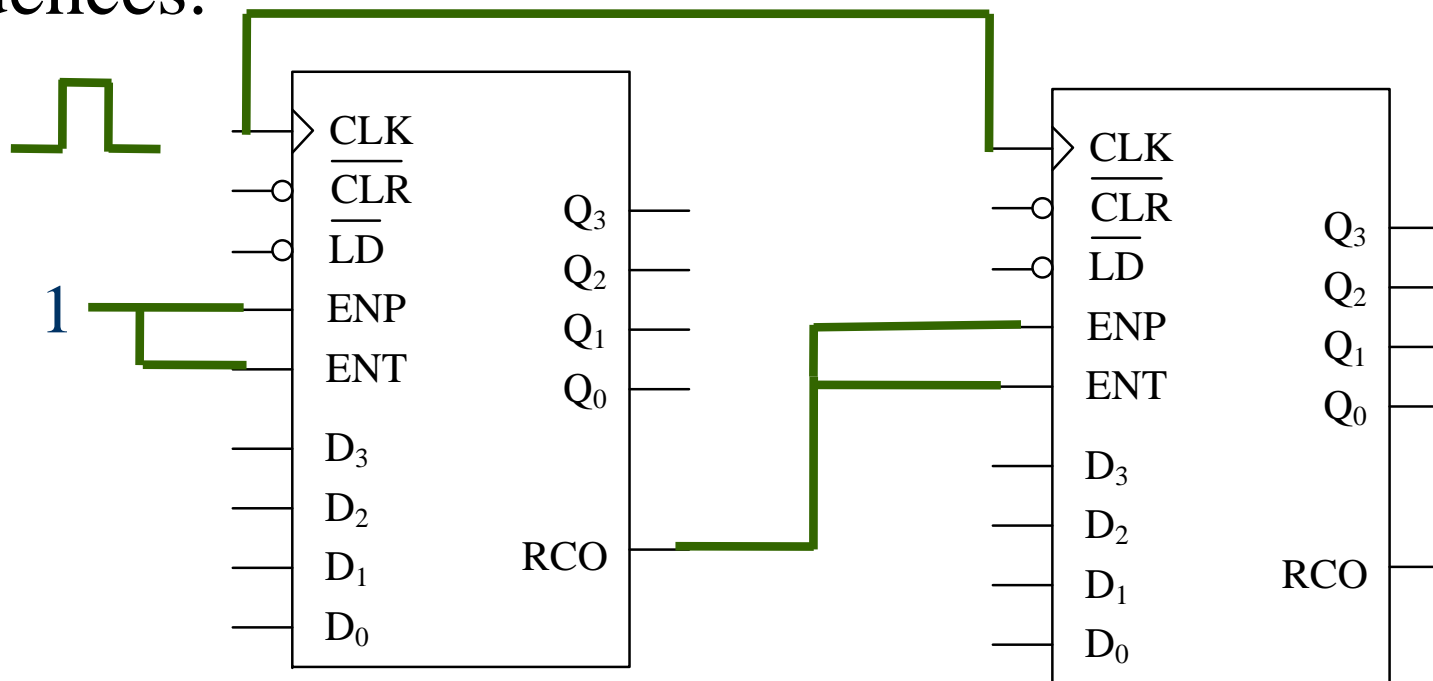
Chip for Synchronous counters

- ◆ ENP and ENT must both be HIGH to ENABLE the chip.
- ◆ CLK is positive-edge triggered.
- ◆ $\overline{\text{LD}}$ is the LOAD (active-LOW). If $= 0$, D_3 to D_0 will be loaded to output O_3 to O_0 .
- ◆ If $\overline{\text{CLR}} = 0$, all the outputs will become 0.
- ◆ $\overline{\text{LD}}$ and $\overline{\text{CLR}}$ are synchronized inputs and will not be activated immediately but only on the next positive clock transition.



Chip for Synchronous counters

- ◆ RCO (*ripple carry output*) is 1 when $Q_3Q_2Q_1Q_0 = 1111$.
- ◆ RCO can be used in conjunction with the enable inputs allows these counters to be cascaded for higher count sequences.



Chip for Synchronous counters (continue)

◆ Exercise 9:

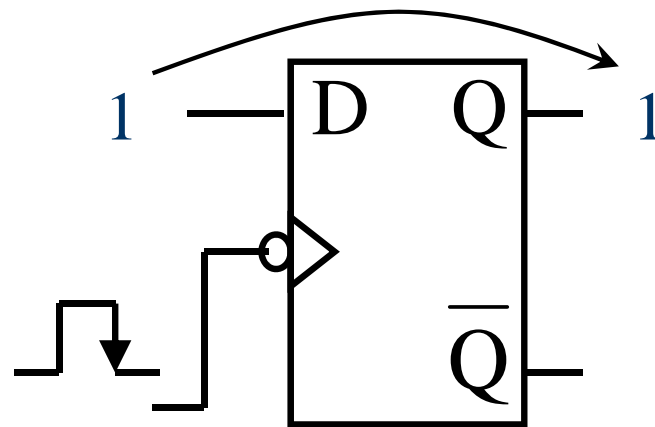
- use 74163 IC to design counter with this sequence:
- 0,1,2,11,0, 1,2,11, 0...

◆ Exercise 10:

- Show how the 74163 ICs can be cascaded to count from 0 -56 then back to 0 – 56.

Shift Register

- ◆ Shift registers are constructed using several flip-flop, connected in such a way to **STORE** and **TRANSFER** digital data.
- ◆ Basically, D flip-flop is used. The input data (either '0' or '1') is applied to the D terminal and the data will be stored at Q during positive/negative-edge transition of the clock pulse.



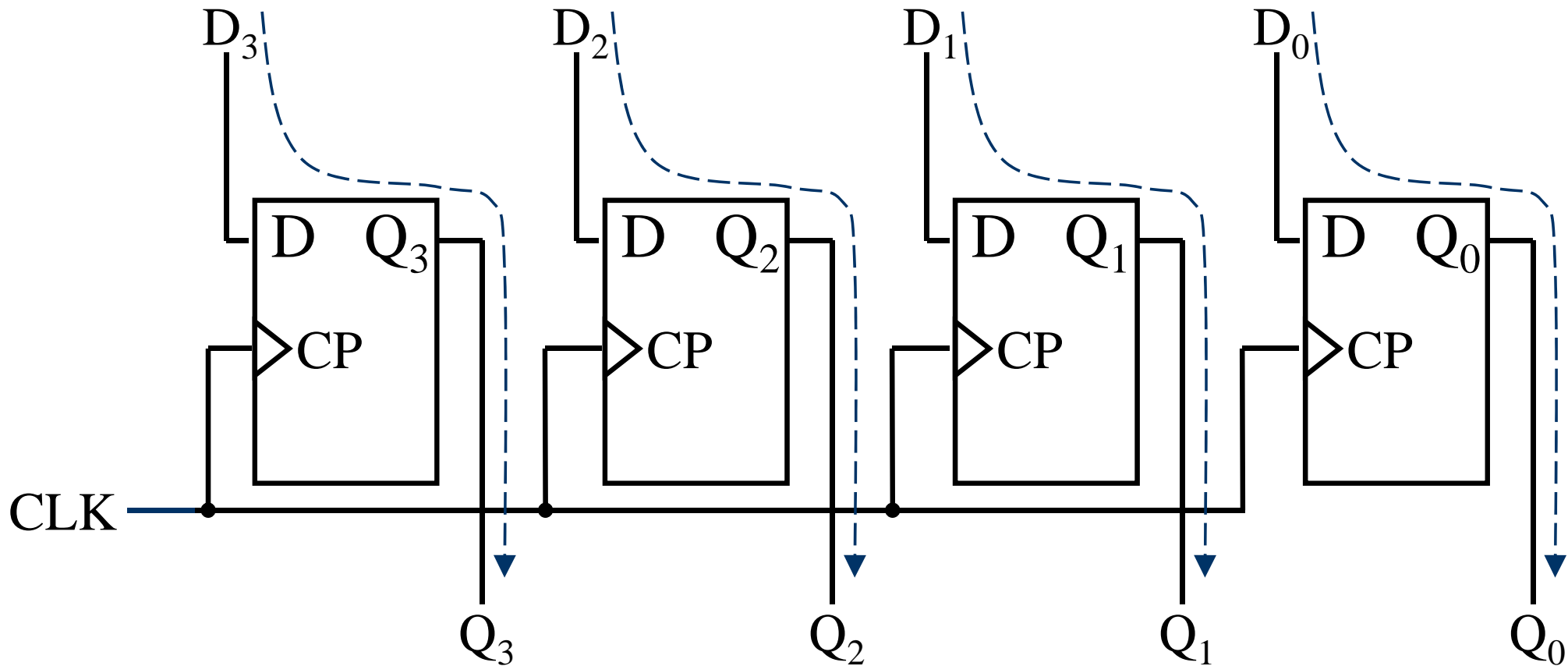
negative edge
transition of CLK

Shift Register

- ◆ One D FF is used to store 1-bit of data. Thus, the number of flip-flop used is the same with the number of bit stored.
- ◆ Shift register mean that the data in each FF can be transferred/move to other FF upon edge triggering of the clock signal.
- ◆ Four types of data movement in shift register are:-
 - ◆ *Parallel in / parallel out (PIPO)*
 - ◆ *Serial in / serial out (SISO)*
 - ◆ *Parallel in / serial out (PISO)*
 - ◆ *Serial in / parallel out (SIPO)*

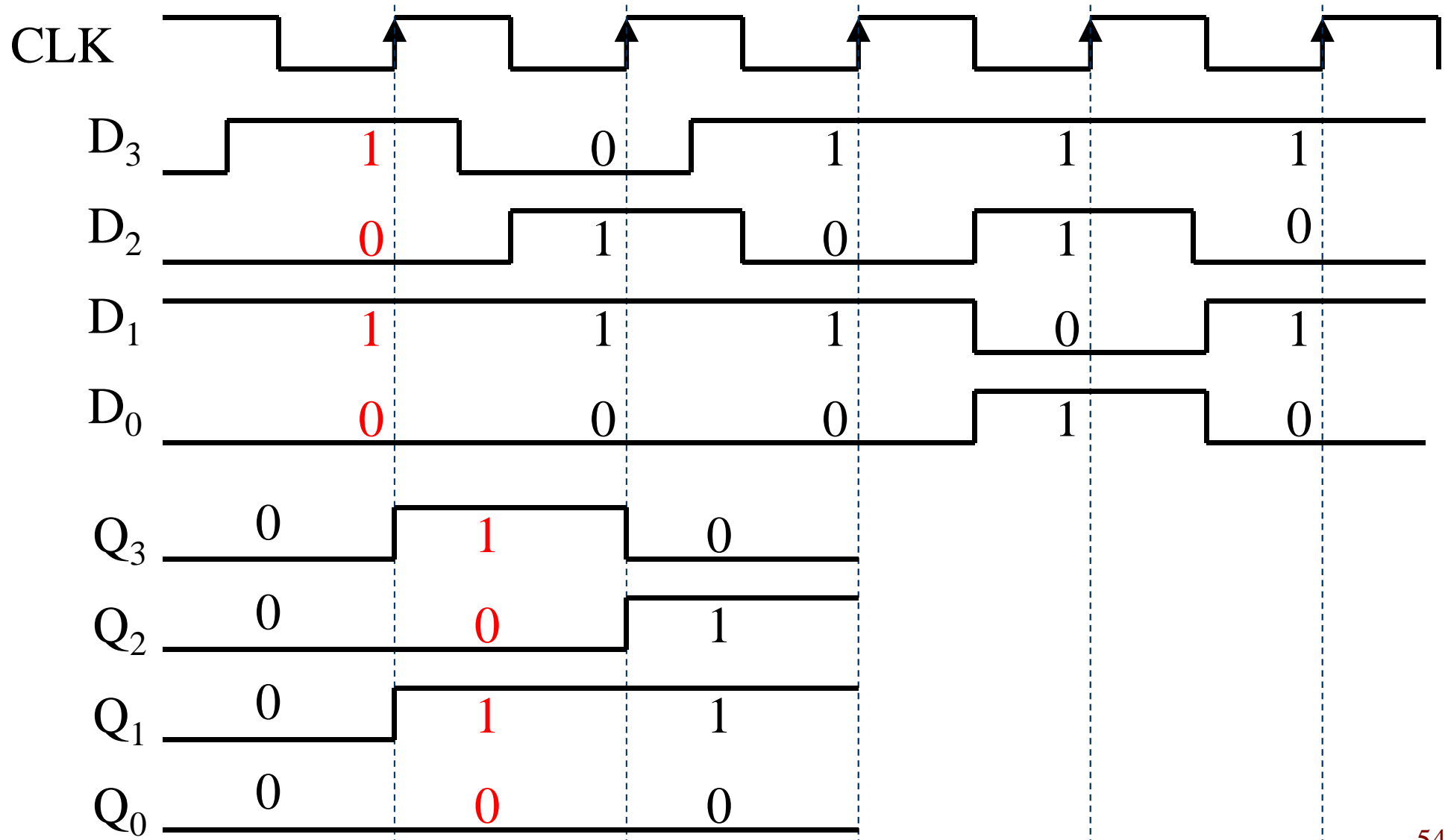
Parallel in / parallel out (PIPO)

- ◆ Flip-flop configuration for PIPO register.



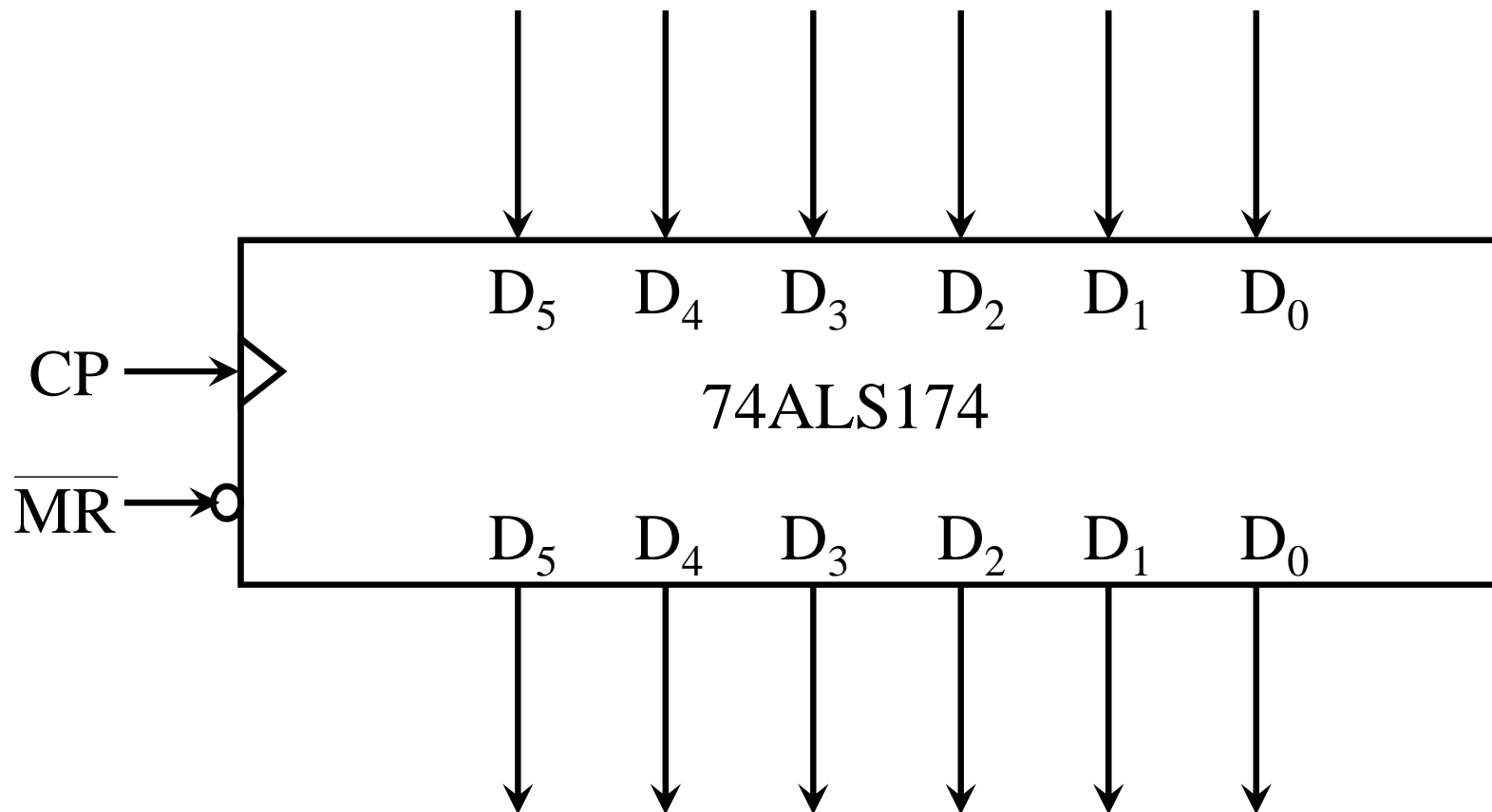
Parallel in / parallel out (PIPO)

- ◆ PIPO data movement.



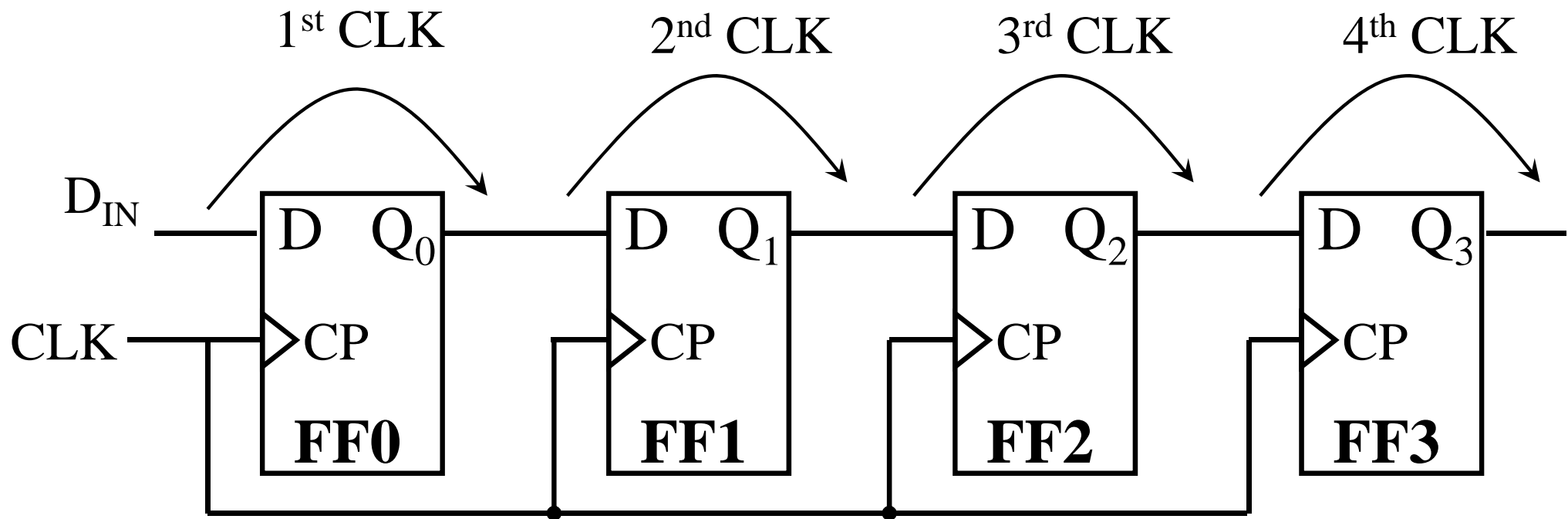
Parallel in / parallel out (PIPO)

- ◆ 74ALS147 chip for PIPO register (6-bit):-



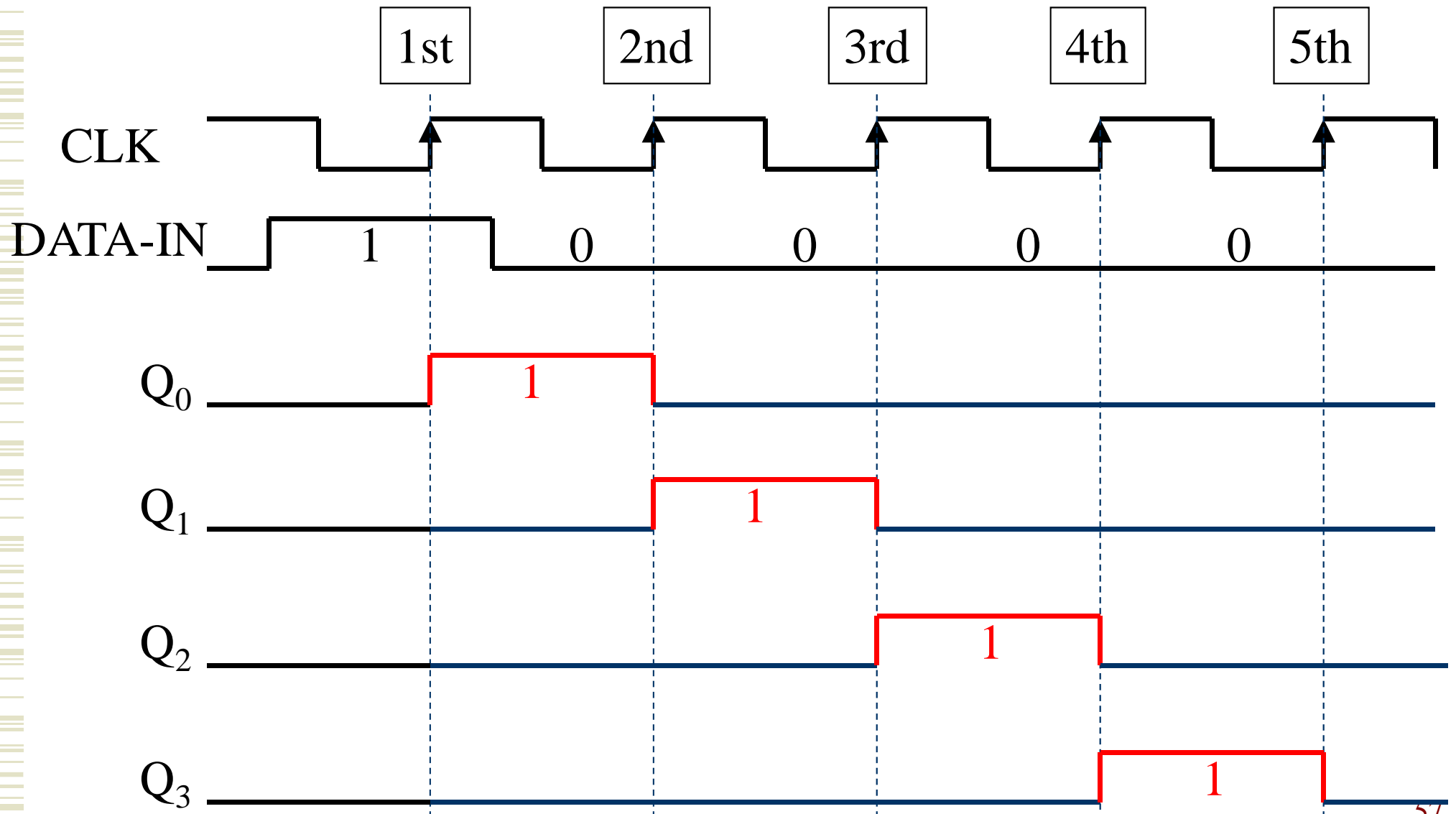
Serial in / serial out (SISO)

- ◆ Flip-flop connection for SISO.



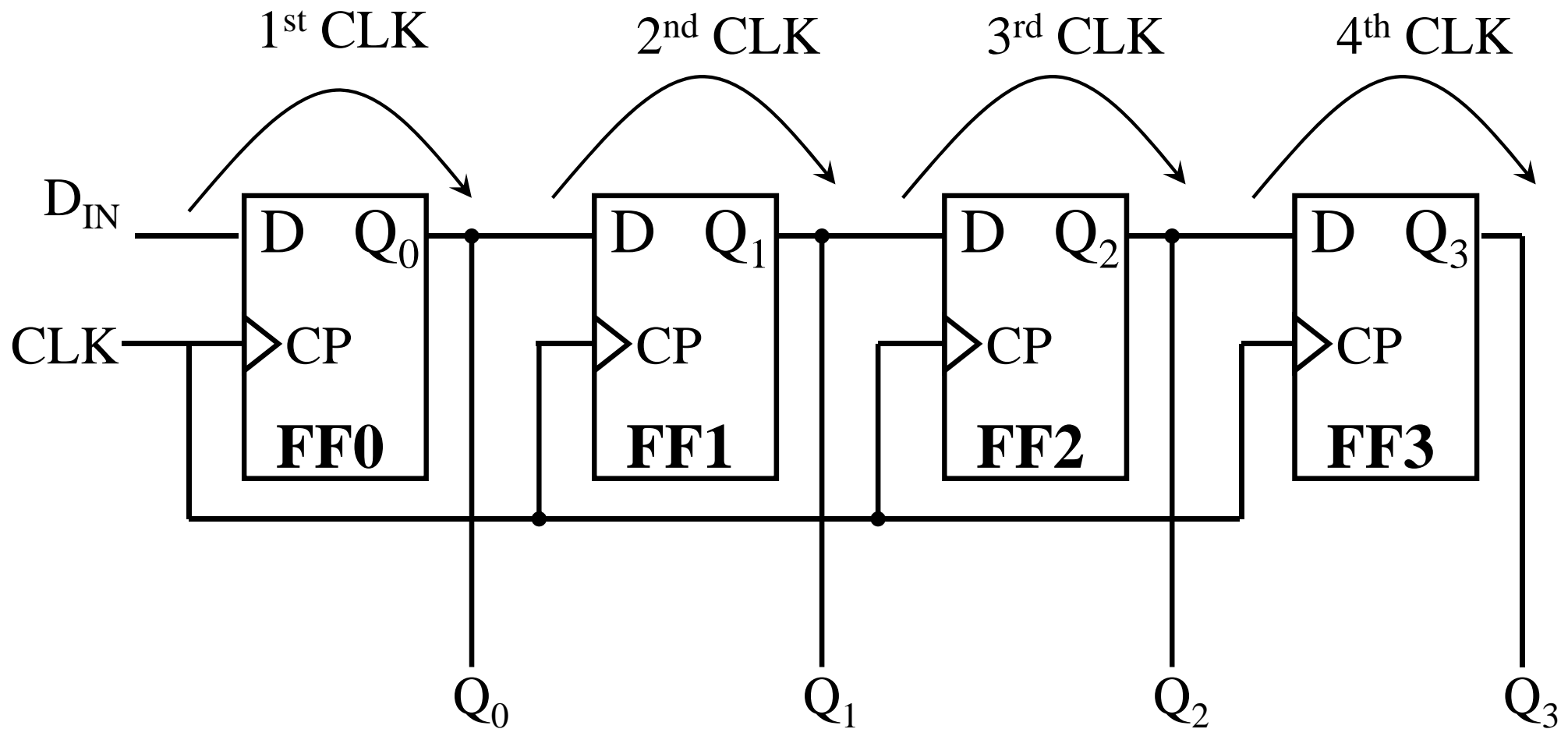
Serial in / serial out (SISO)

- ◆ SISO data movement. Binary data 10000 is transferred!



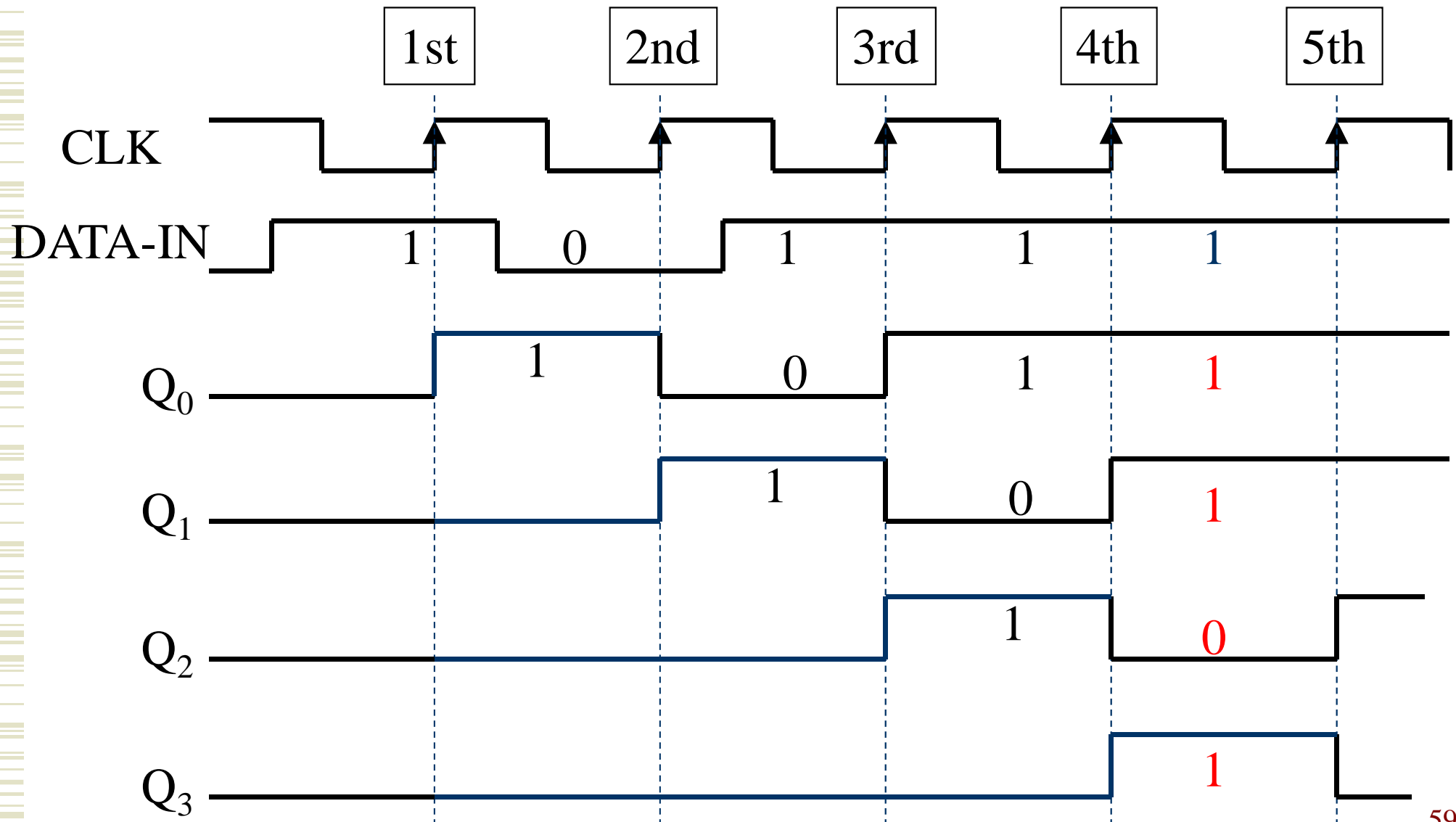
Serial in / parallel out (SIPO)

- ◆ Flip-flop connection for SIPO.



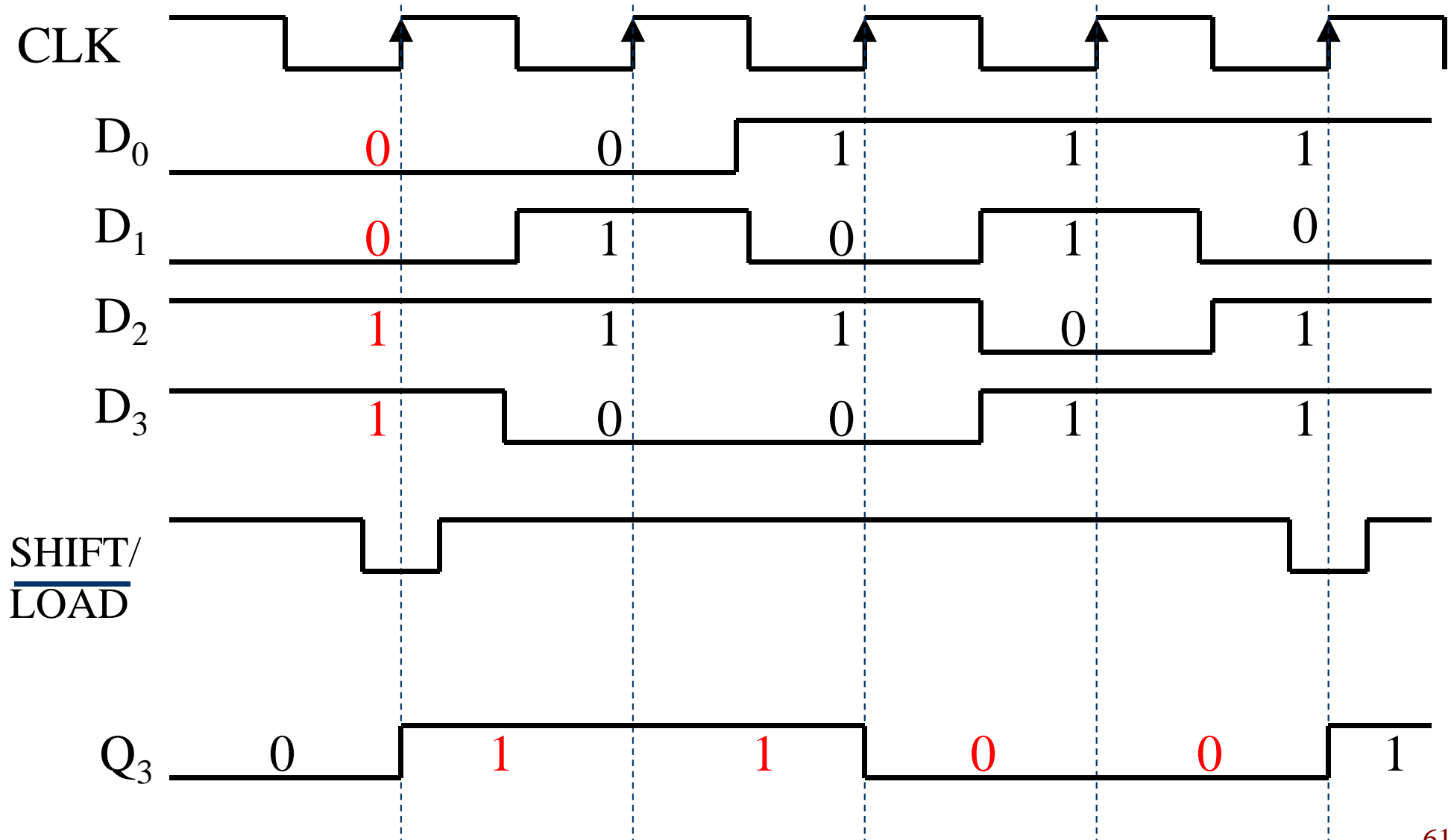
Serial in / parallel out (SIPO)

- ◆ SIPO data movement. Binary data 1011 is transferred!



Parallel in / serial out (PISO)

- ◆ PISO data movement.



Shift Register

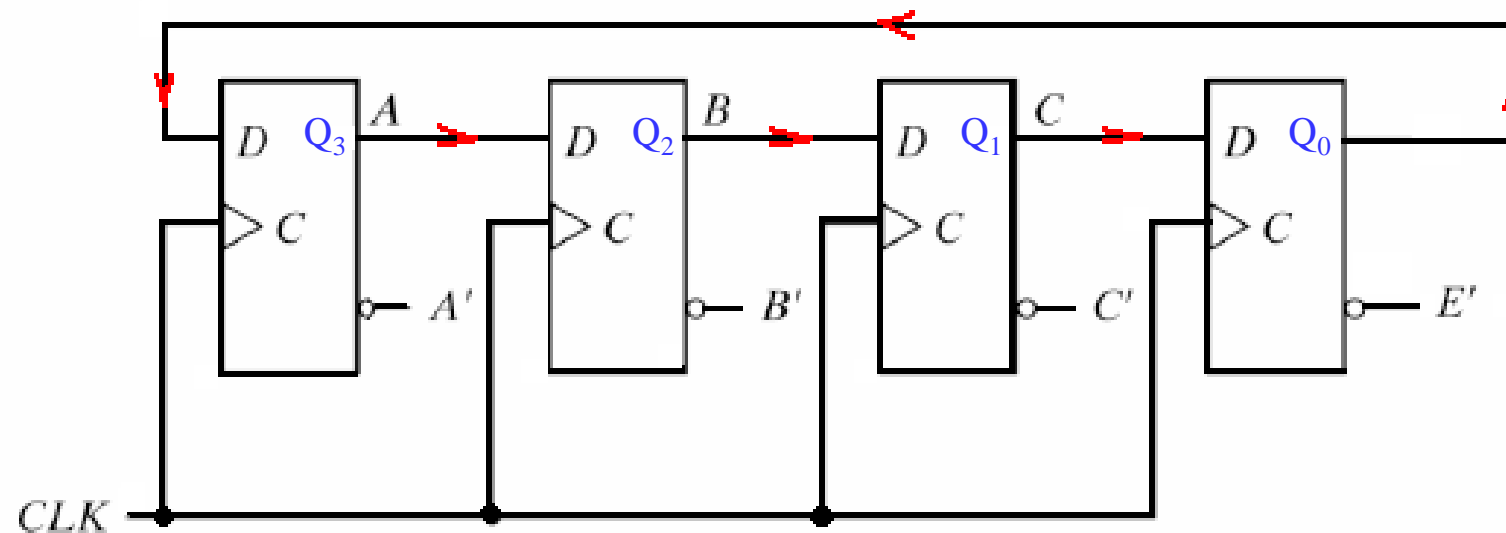
◆ *Serial Data VS Parallel Data* movement

Serial	Parallel
<ul style="list-style-type: none">• Movement of N-bit data require N number of CLK pulses. Thus, the operation is slow.• Only one FF is required to be connected at the output terminal, thus only one wire is required.	<ul style="list-style-type: none">• Require only one CLK pulse to transfer all N-bit of data. Thus, operation is faster than serial.• Required N number of connection to the output terminal, which is proportional to the number of bit. Thus, too many connection is required.

Shift Register Counters

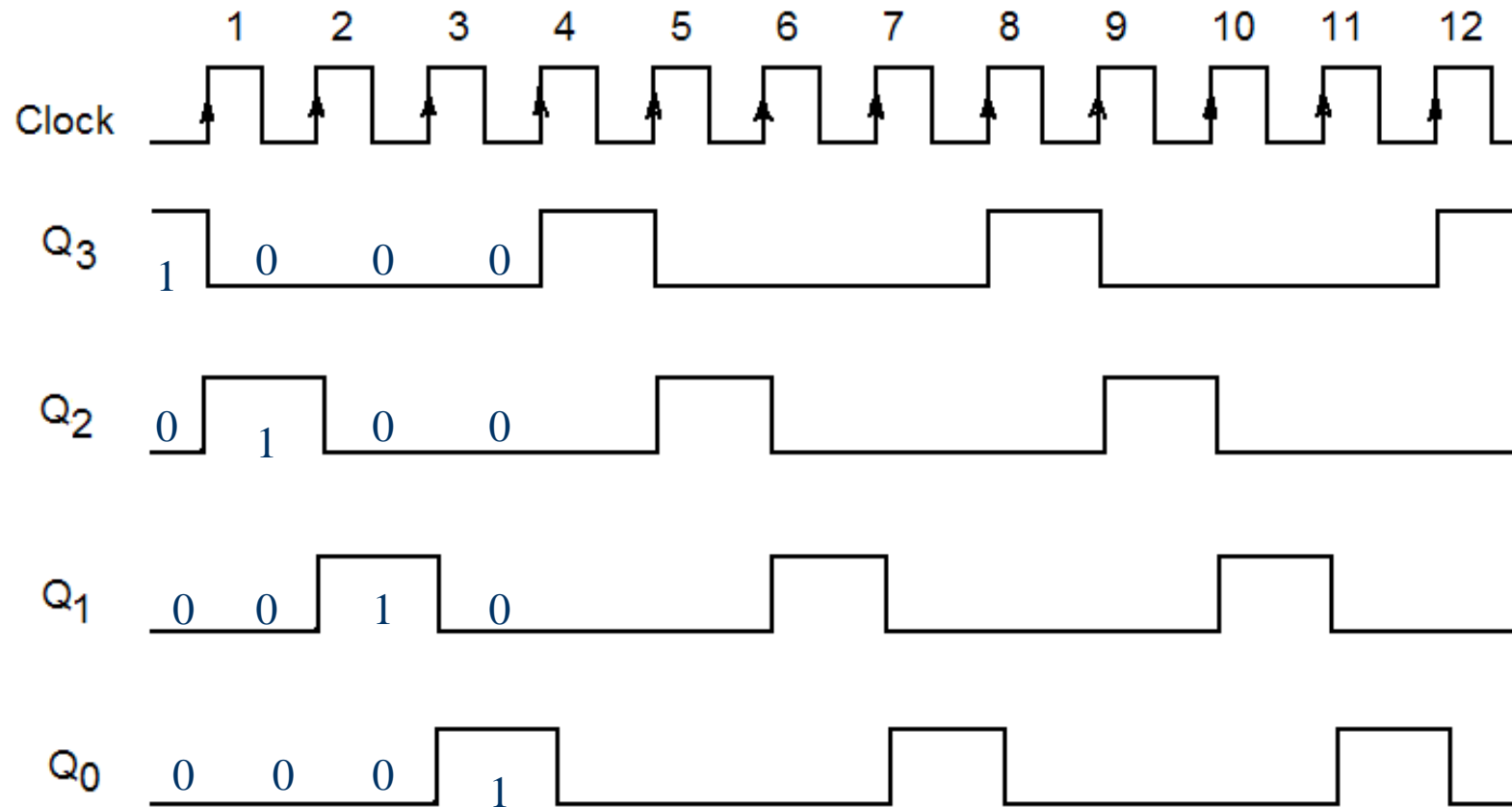
- ◆ A *shift register counter* is a shift register which output being fed back (connected back) to the serial input. This shift register would count the state in a unique sequence!
- ◆ Two types of shift register counter:-
 - ◆ *The Ring counter*
 - ◆ *The Johnson counter*

Ring Counter



a) Four-bit ring counter

Ring Counter (continue)



b) Four-bit ring counter waveform

Assuming the counter is initialized to $Q3..Q0 = 1000$, then the counting will be as in the waveform given above

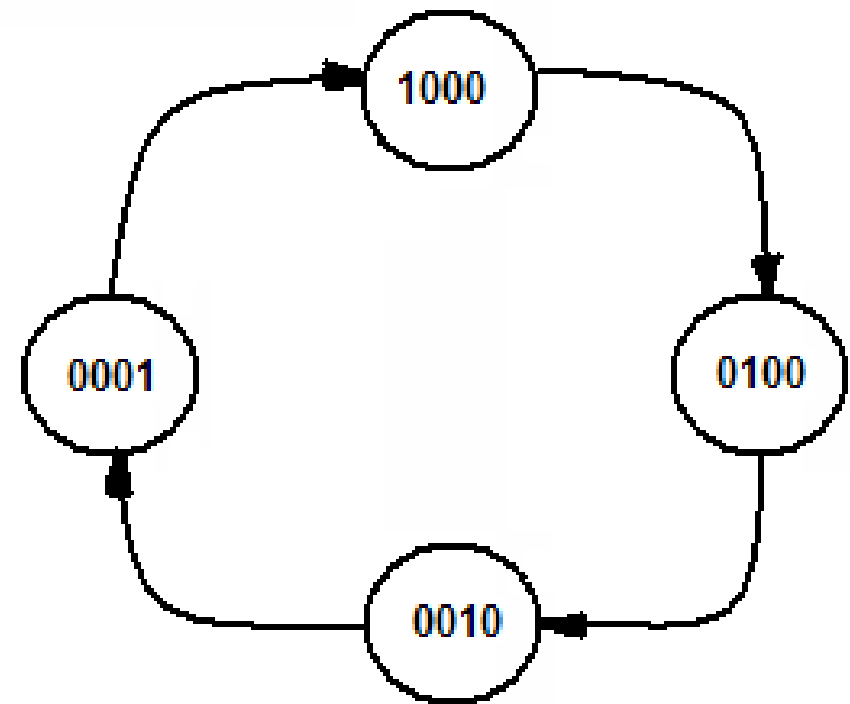
Ring Counter (continue)

Sequence Number	Flip-Flop outputs			
	A	B	C	E
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

c) Sequence table

Ring Counter (continue)

- ❑ Ring counters are used to construct “One-Hot” counters
- ❑ It can be constructed for any desired MOD number
- ❑ A MOD-N ring counter uses N flip-flops connected in the arrangement as shown in fig. a)
- ❑ In general ring-counter will require more flip-flops than a binary counter for the same MOD number

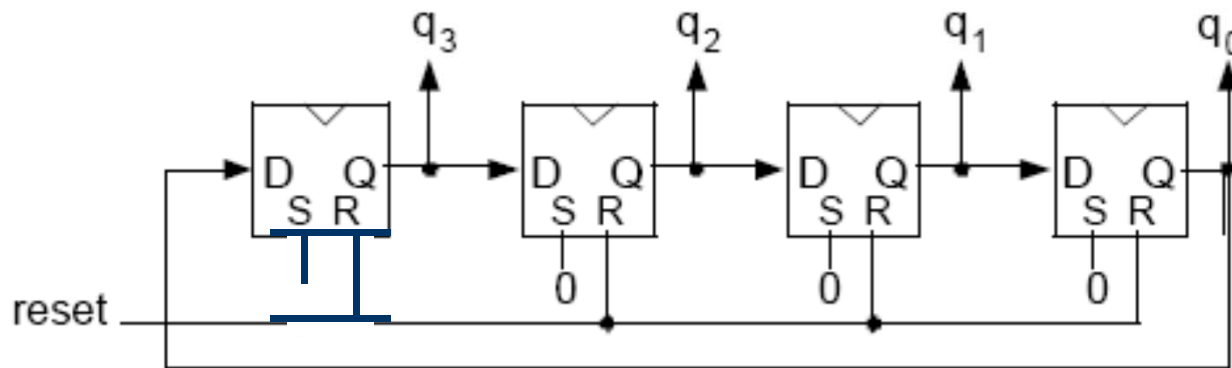


d) State Diagram

Ring Counter (continue)

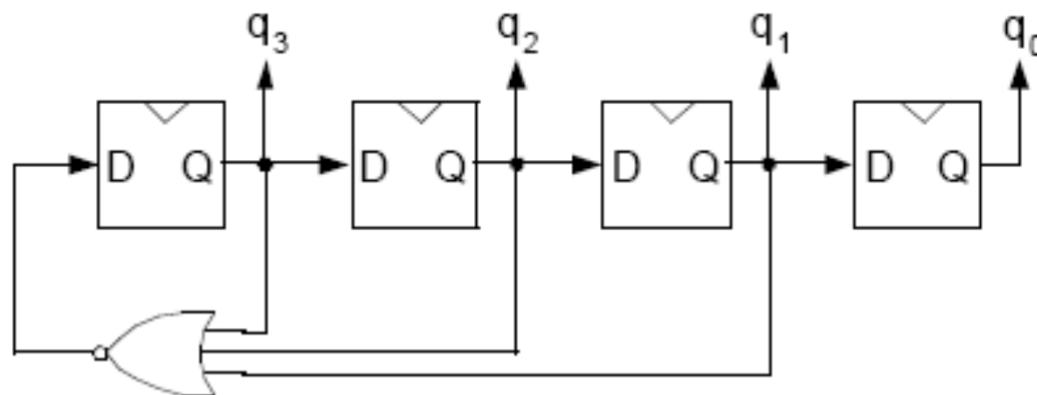
One-hot counter

- What are these good for?



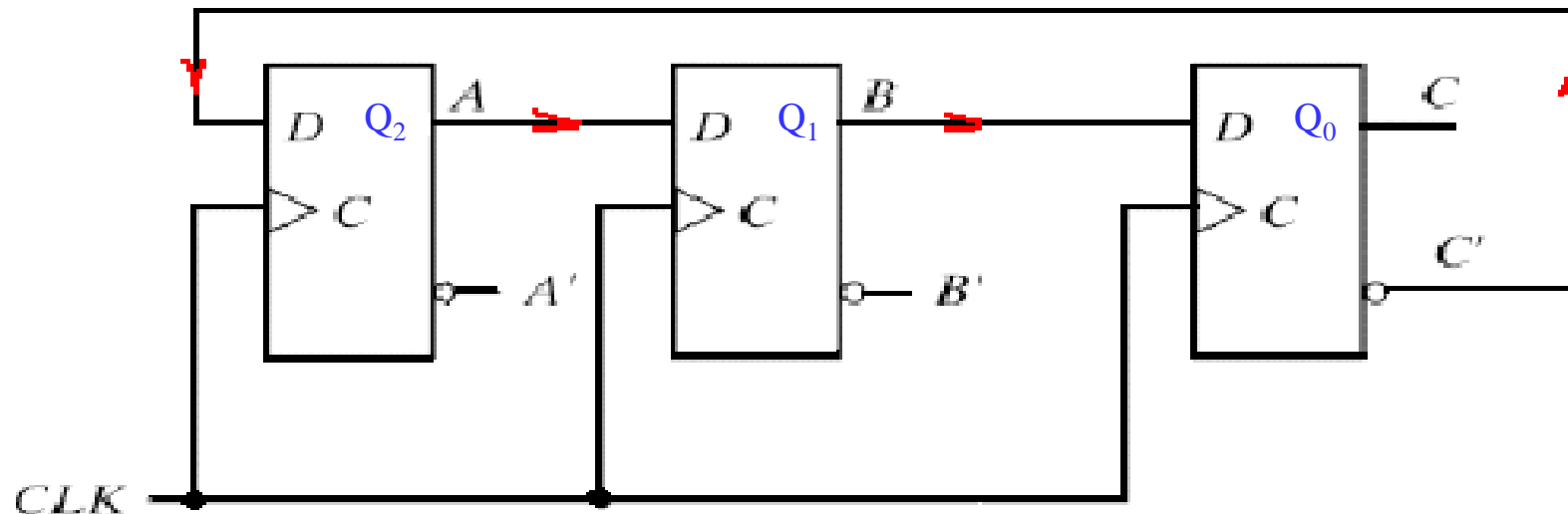
Assuming the counter is initialized to $Q_3..Q_0 = 1000$, then the counting will be as given above

“Self-starting” version:



Johnson Counter

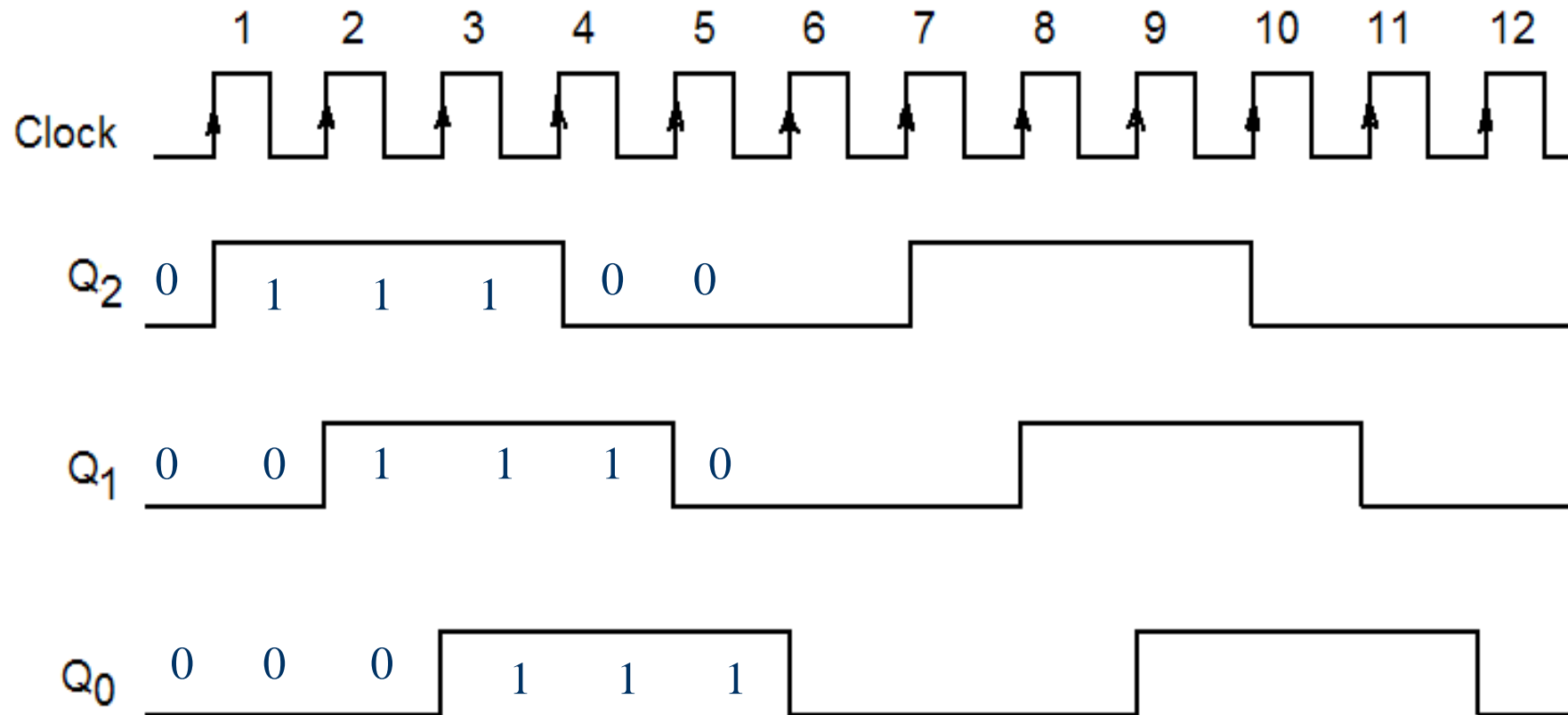
Or Twisted-ring counter



a) MOD-6 Johnson counter

- Johnson counter constructed exactly like a normal ring counter except that the inverted output of the last flip-flop is fed back to first flip-flop

Johnson Counter (Continue)

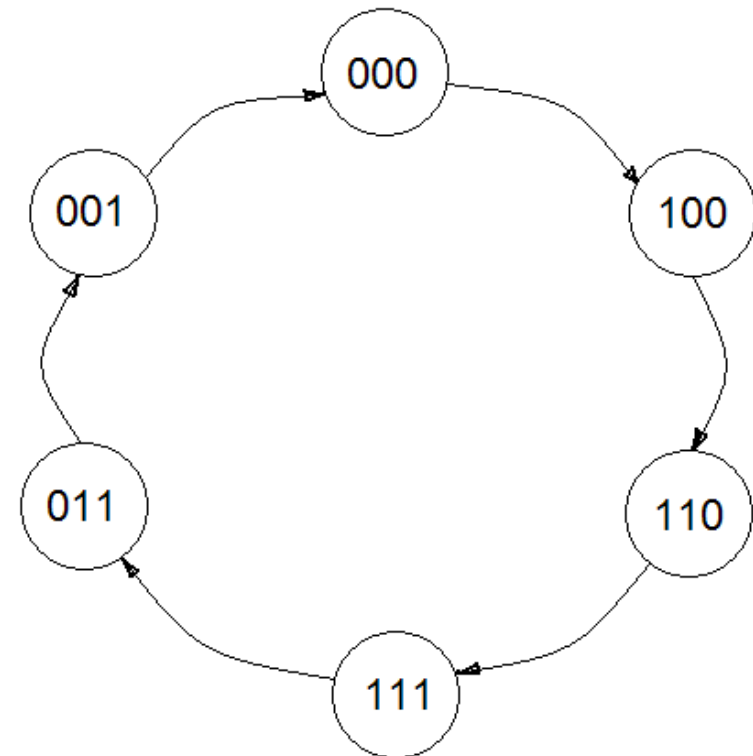


b) MOD-6 Johnson counter waveform

Johnson Counter (Continue)

Sequence Number	Flip-Flop outputs		
	A	B	C
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1
6	0	0	0
7	1	0	0
8	1	1	0
▪	▪	▪	▪
▪	▪	▪	▪
▪	▪	▪	▪

c) Johnson Counter sequence table



d) Johnson counter state diagram