6 Registers and Counters

6-1 REGISTERS

Register- a group of binary cells suitable for holding binary information.

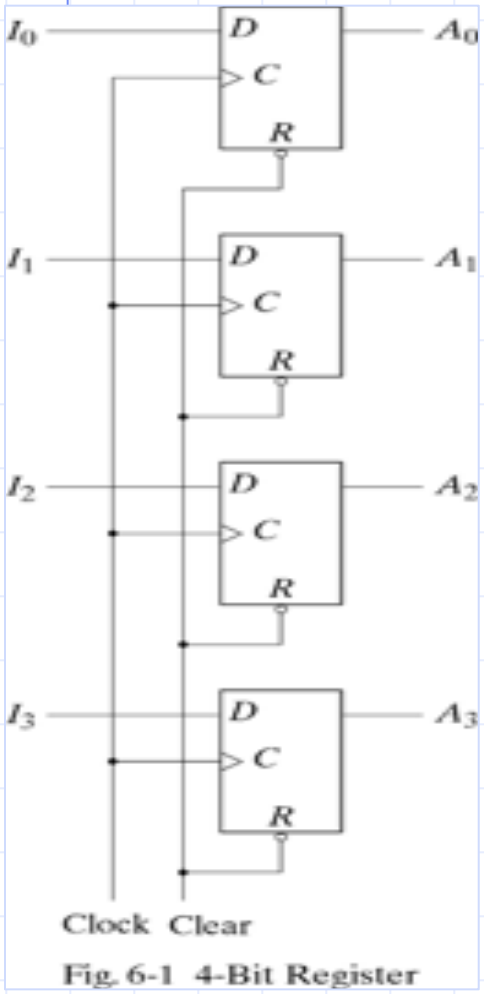


Fig. 6-1 4-Bit Register

- Clock=0 -> 1 : input information transferred
- Clock=0 : unchanged
- Clear=0 : clearing the register to all 0's prior to its clocked operation.

Register with Parallel Load

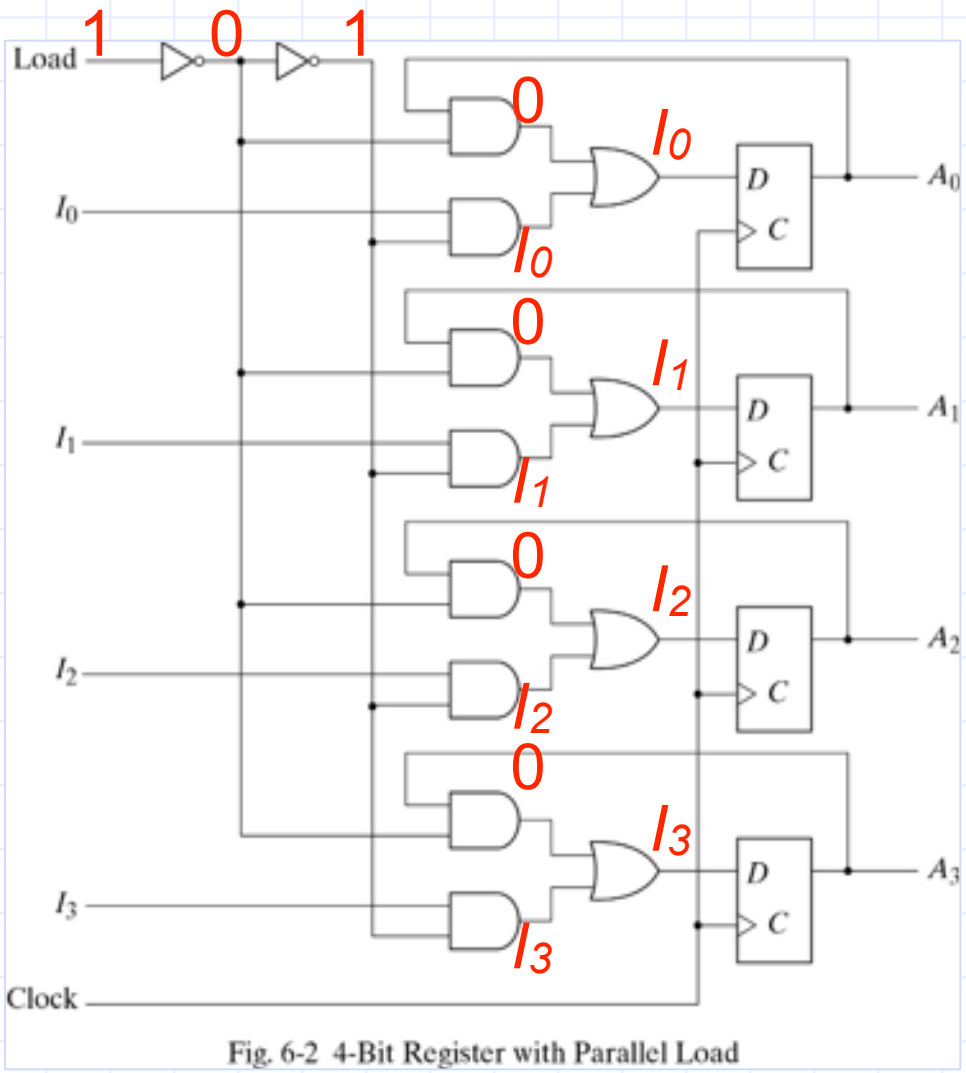


Fig. 6-2 4-Bit Register with Parallel Load

- Clock=0->1 : input information
  -> loading
- Clock=0 or 0->1 or 1: the content of the register -> unchanged
- Load input=1 : the I inputs are transferred into the register

Register with Parallel Load

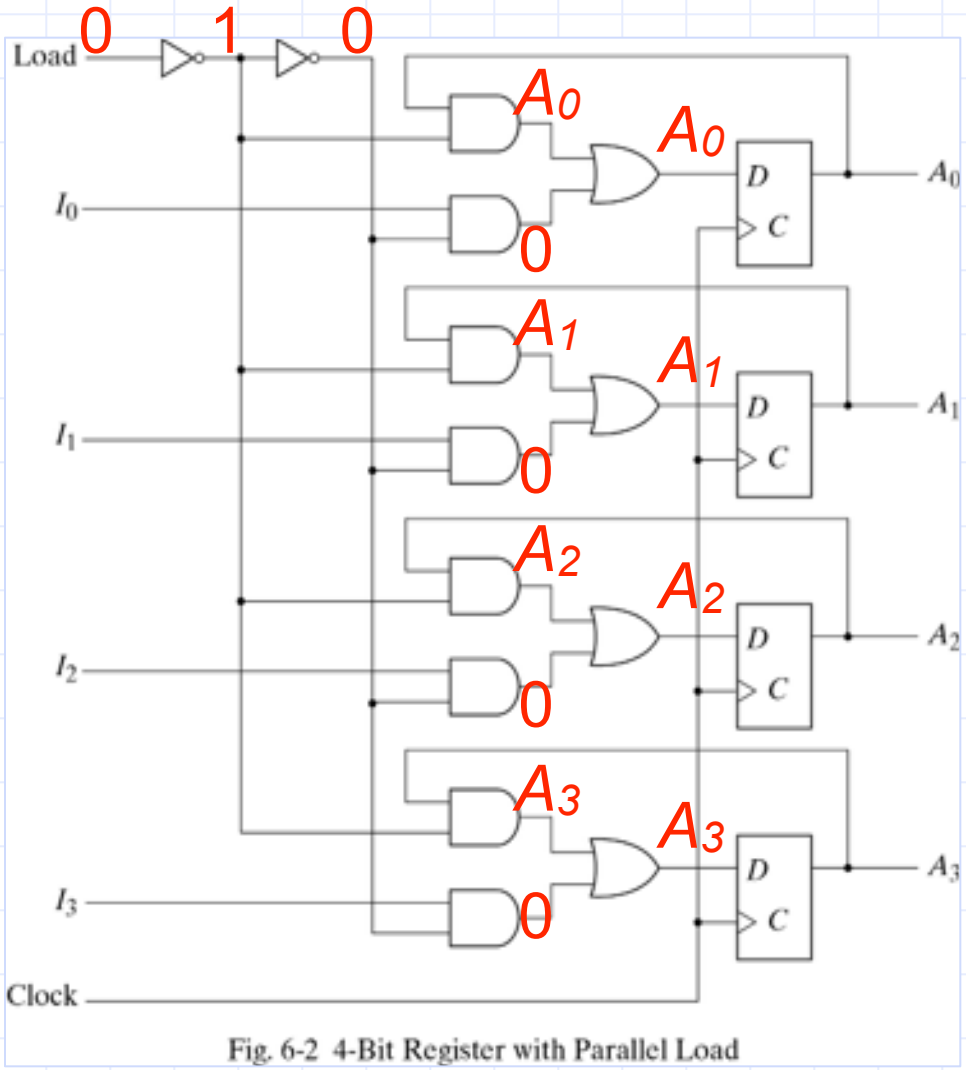


Fig. 6-2 4-Bit Register with Parallel Load

- Load input=0 ; maintain the content of the register

6-2 SHIFT REGISTERS

Shift register-capable of shifting its binary information in one or both directions

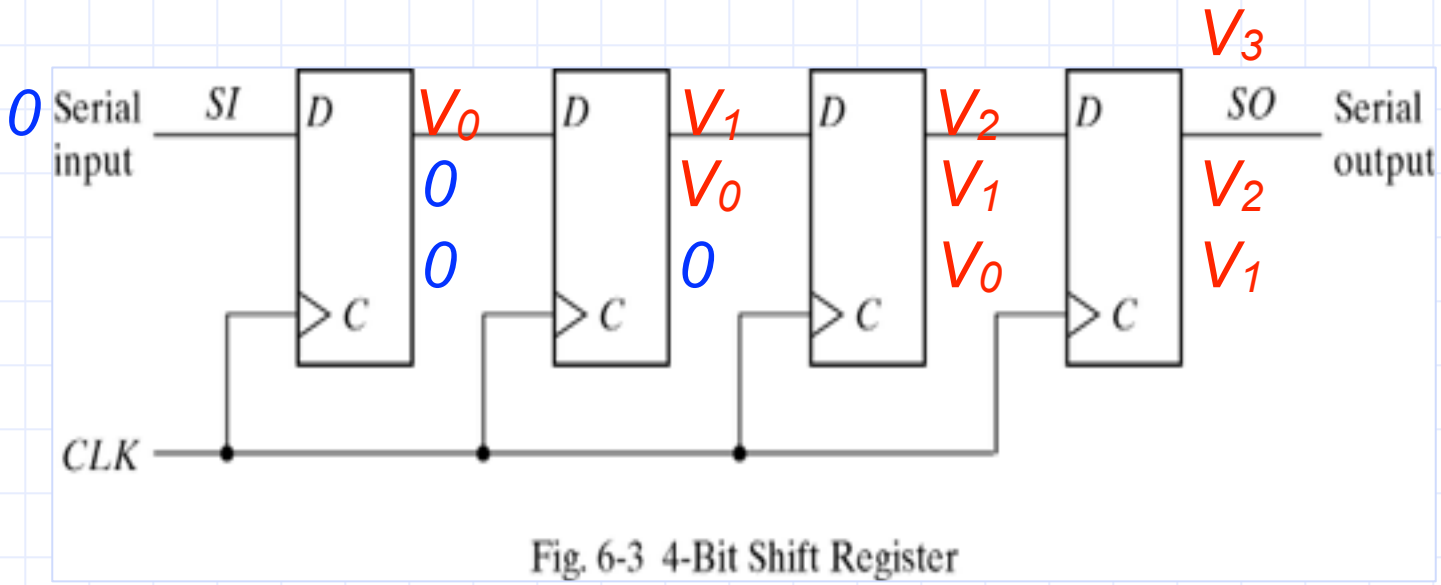
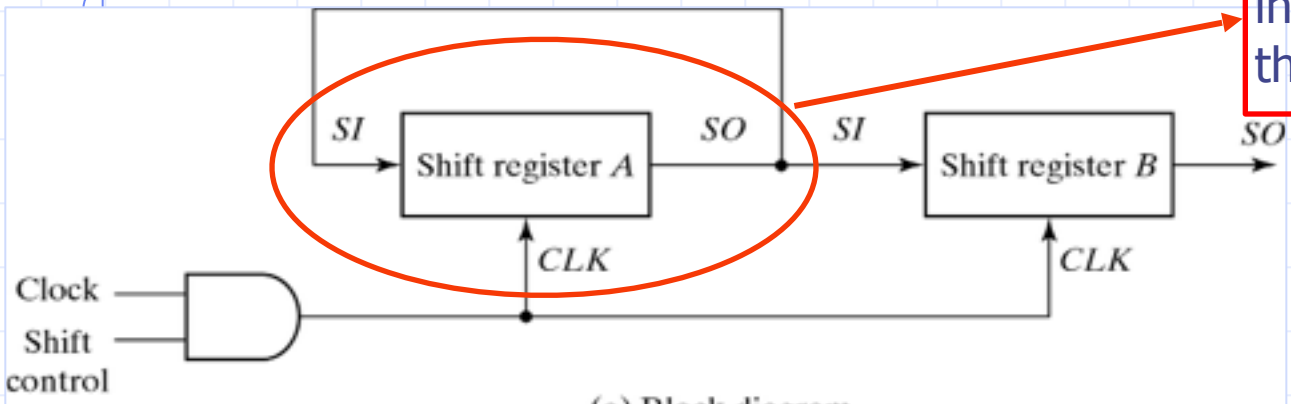


Fig. 6-3 4-Bit Shift Register

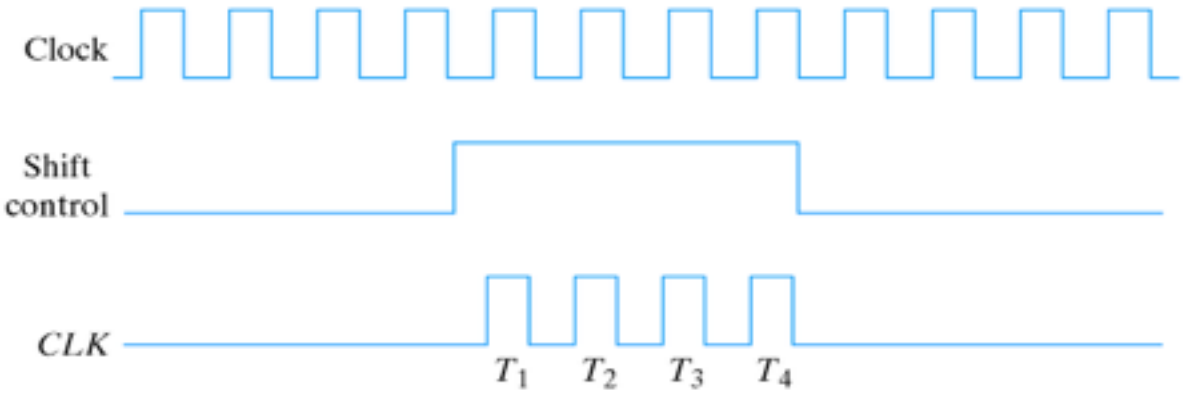
The simplest shift register

Serial Transfer

To prevent the loss of information stored in the source register



(a) Block diagram

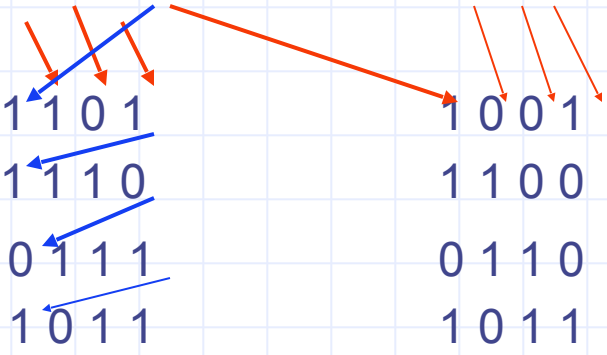


(b) Timing diagram

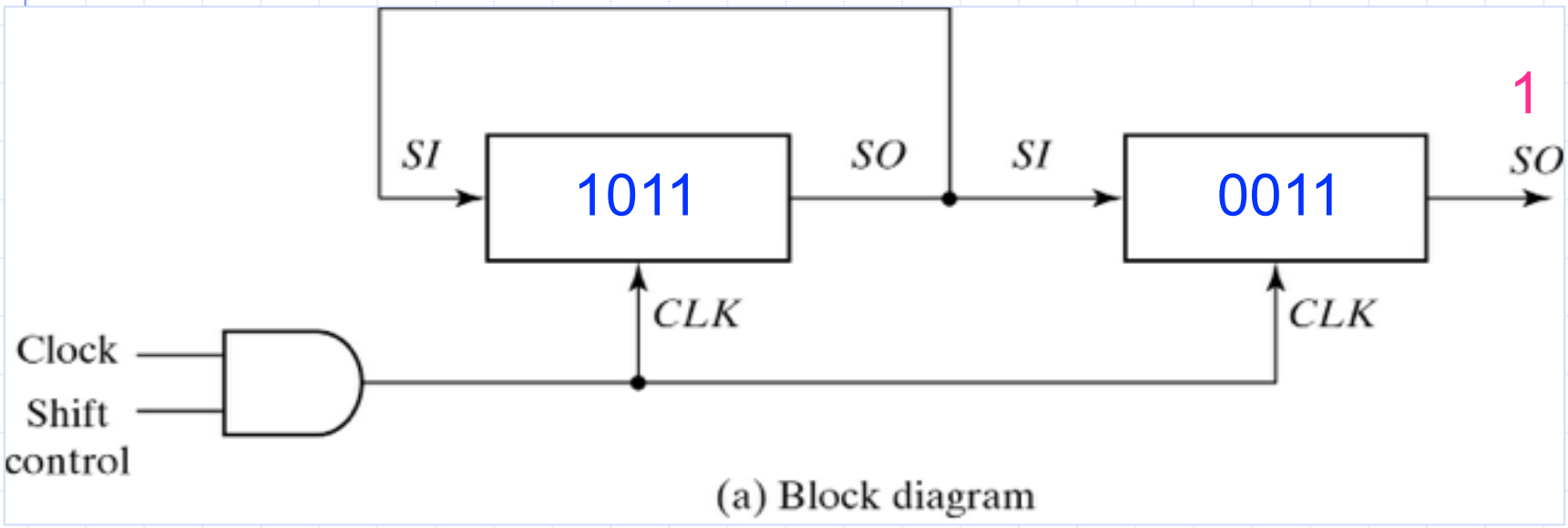
Fig. 6-4 Serial Transfer from Register *A* to register *B*

Serial-Transfer Example

Timing pulse	Shift register A	Shift register B	Serial output of B
Initial value	1 0 1 1	0 0 1 0	0
After T_1	1 1 0 1	1 0 0 1	1
After T_2	1 1 1 0	1 1 0 0	0
After T_3	0 1 1 1	0 1 1 0	0
After T_4	1 0 1 1	1 0 1 1	1

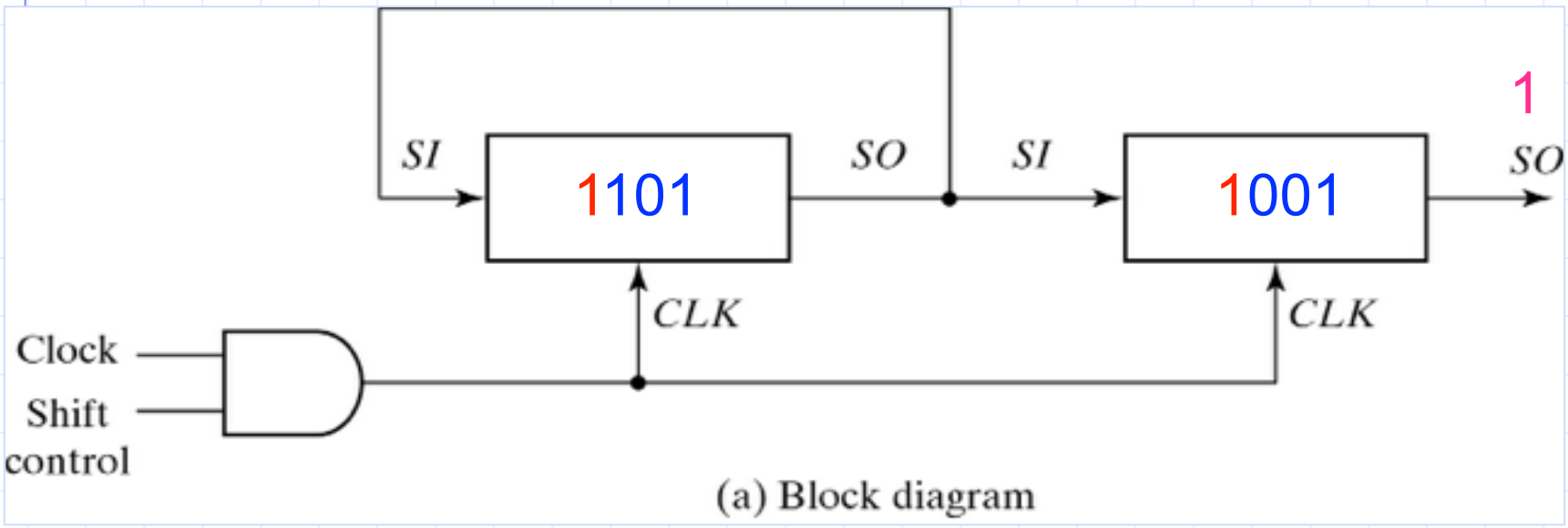


Serial Transfer



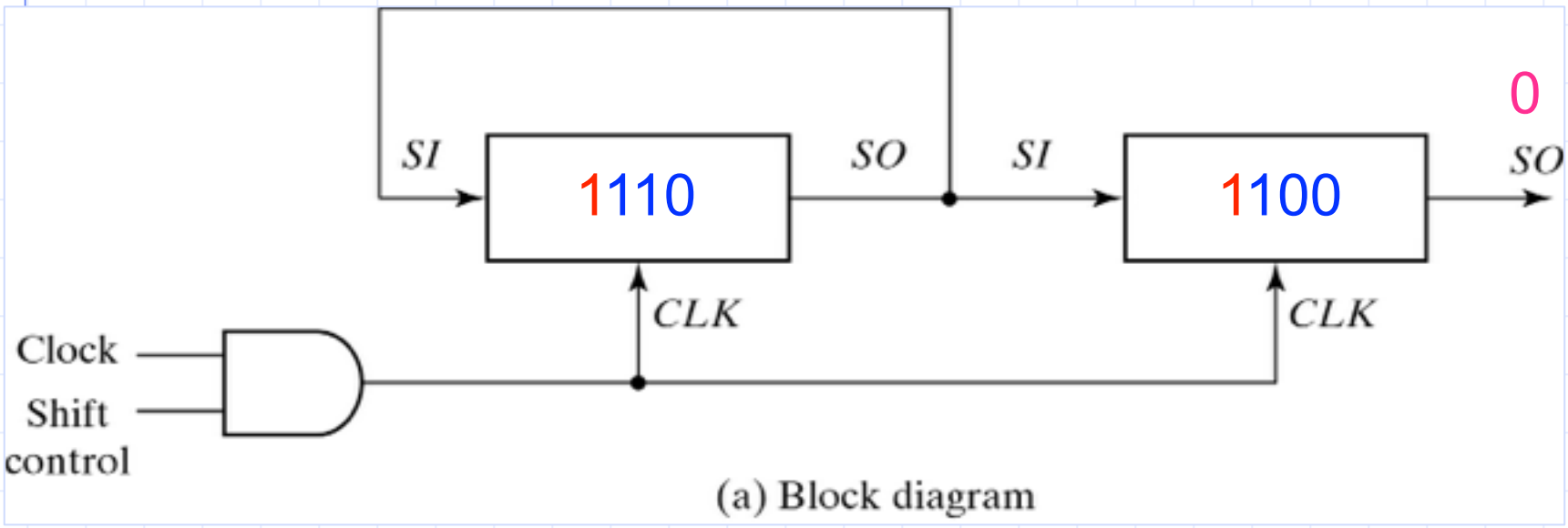
(a) Block diagram

Serial Transfer



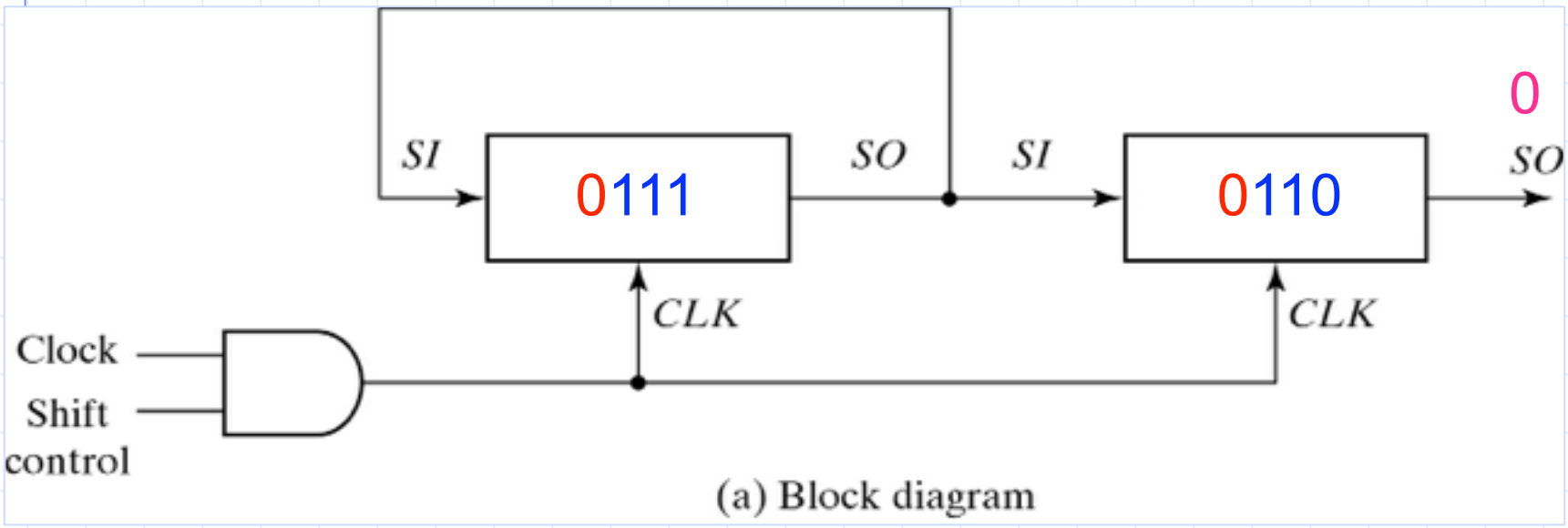
(a) Block diagram

Serial Transfer



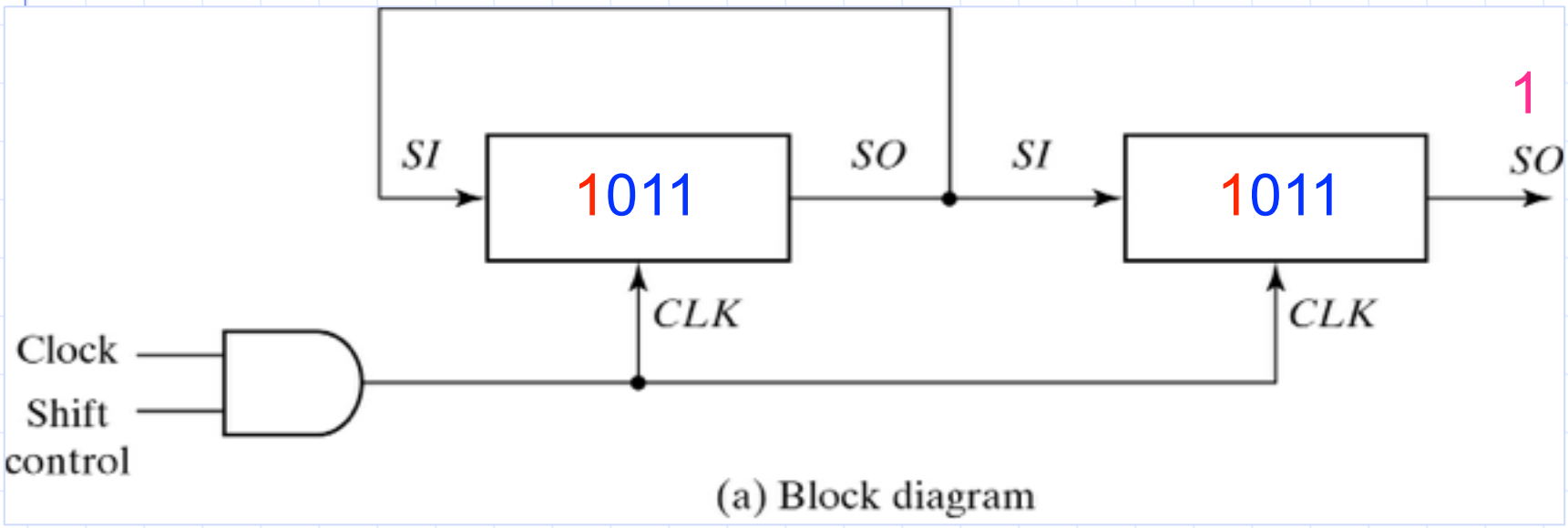
(a) Block diagram

Serial Transfer



(a) Block diagram

Serial Transfer



(a) Block diagram

Serial Addition

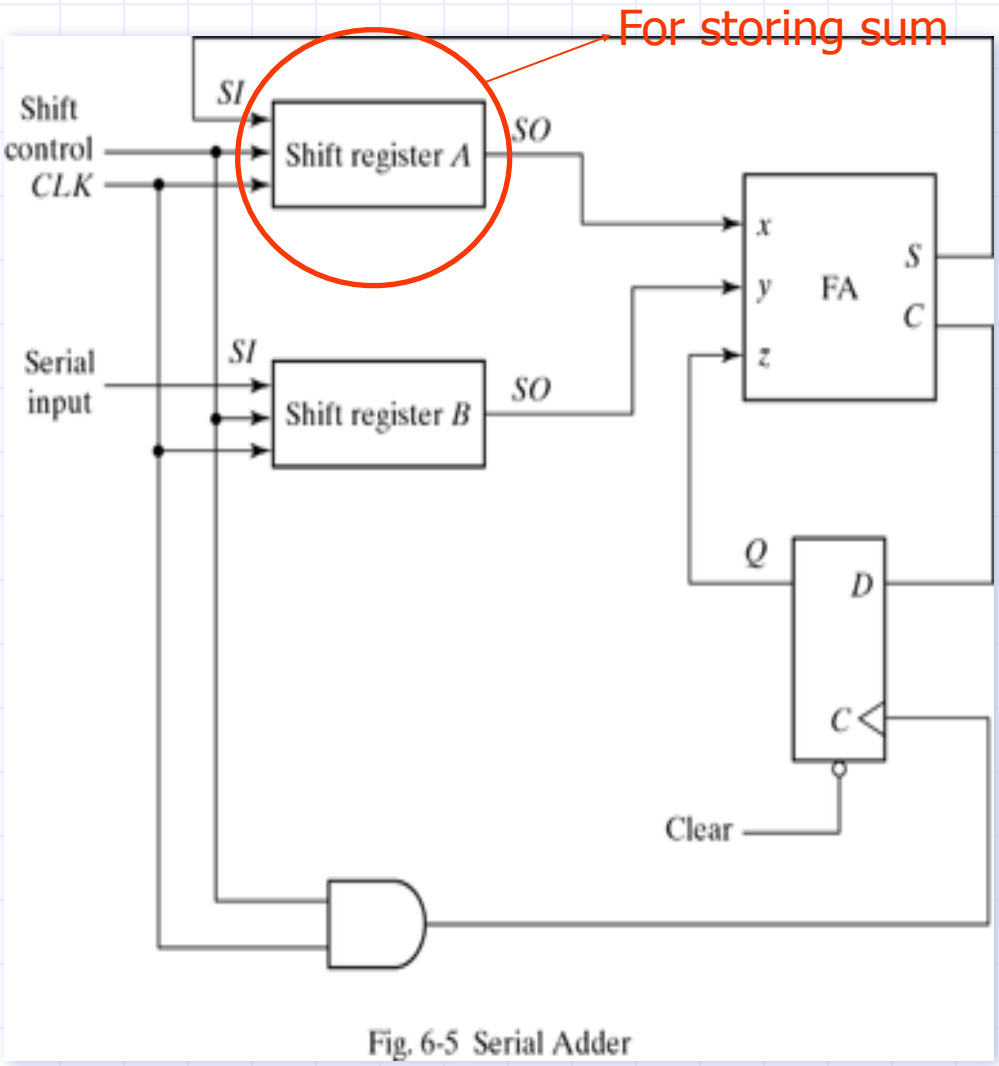


Fig. 6-5 Serial Adder

Operation

- calculate **A**+**B**
- The **SO** of **A** and **B** provide a pair of significant bits for the **FA**
- Output **Q** gives the input carry at **z**
- The shift-right control enables both registers and the carry flip-flop.
- The sum bit from **S** enters the leftmost flip-flop of **A**

Serial Addition

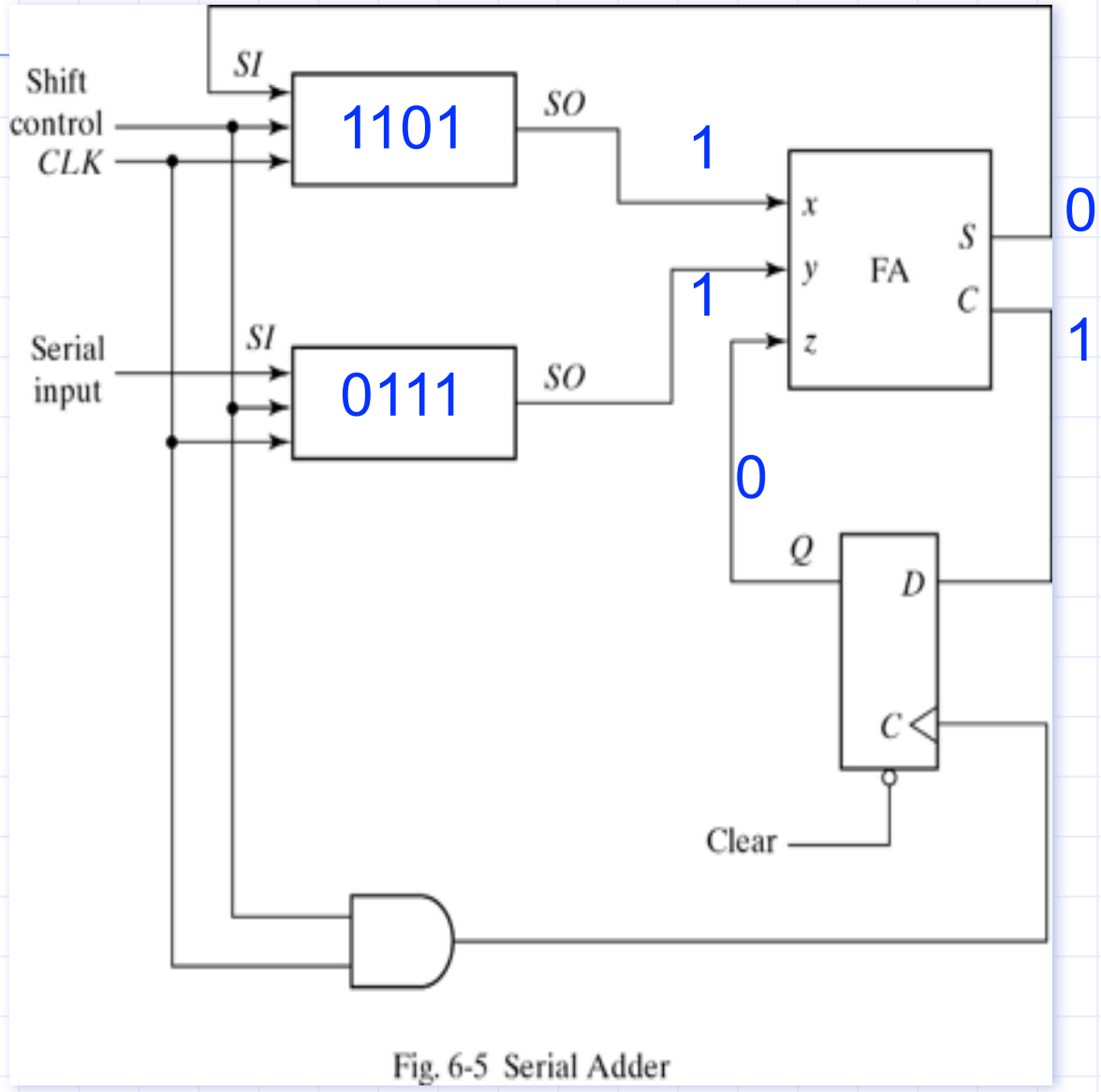


Fig. 6-5 Serial Adder

Serial Addition

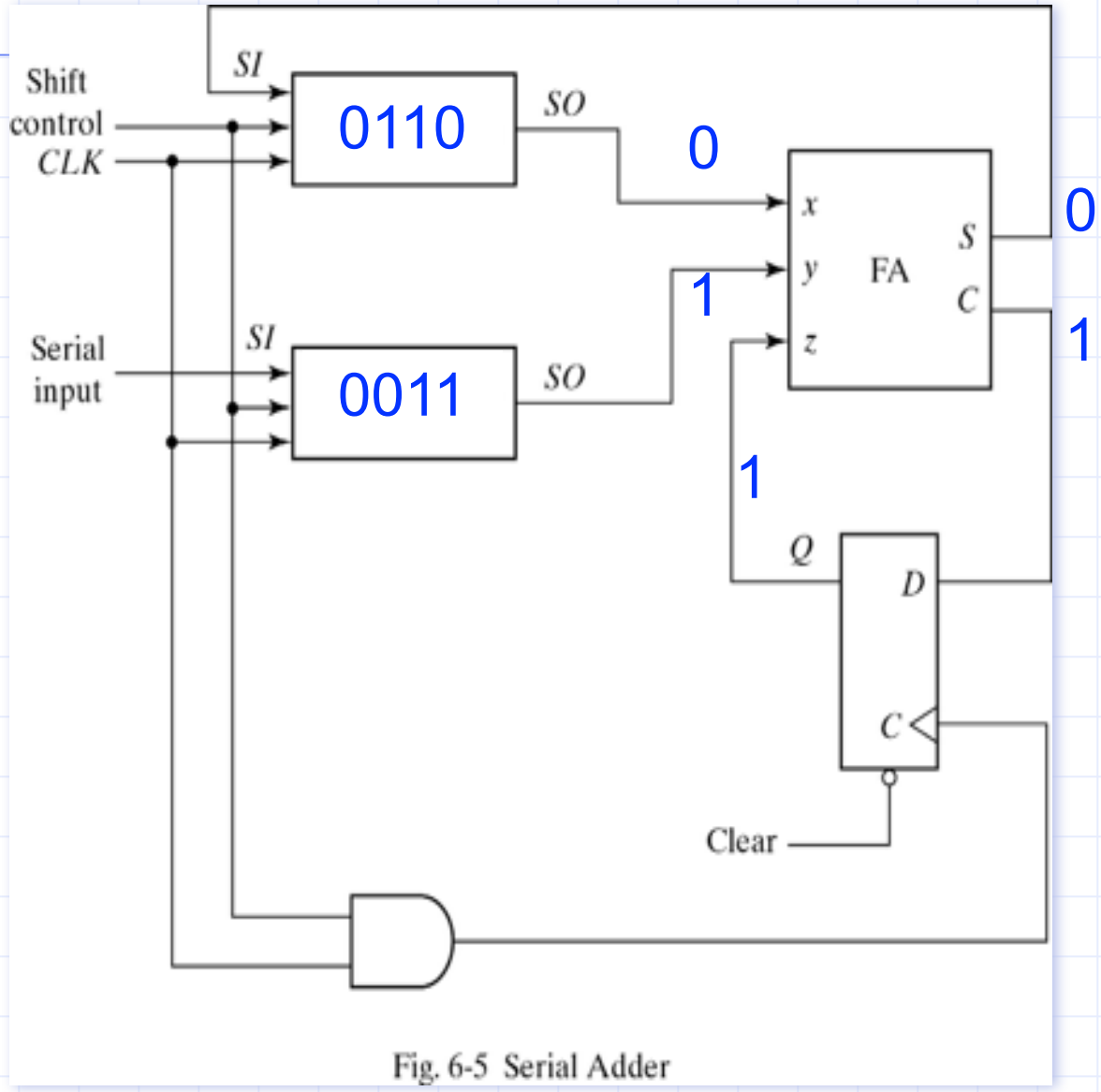


Fig. 6-5 Serial Adder

Serial Addition

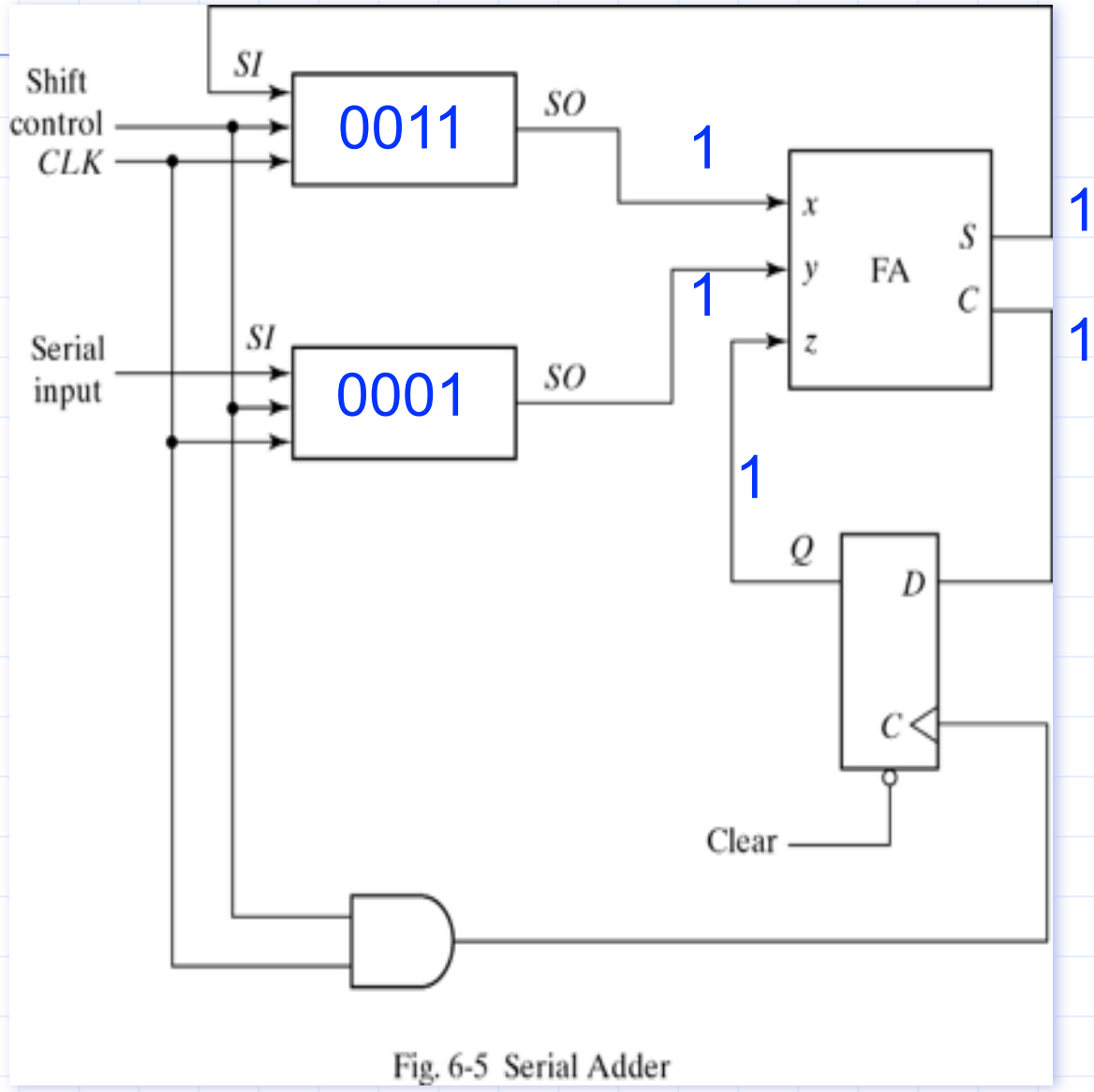


Fig. 6-5 Serial Adder

Serial Addition

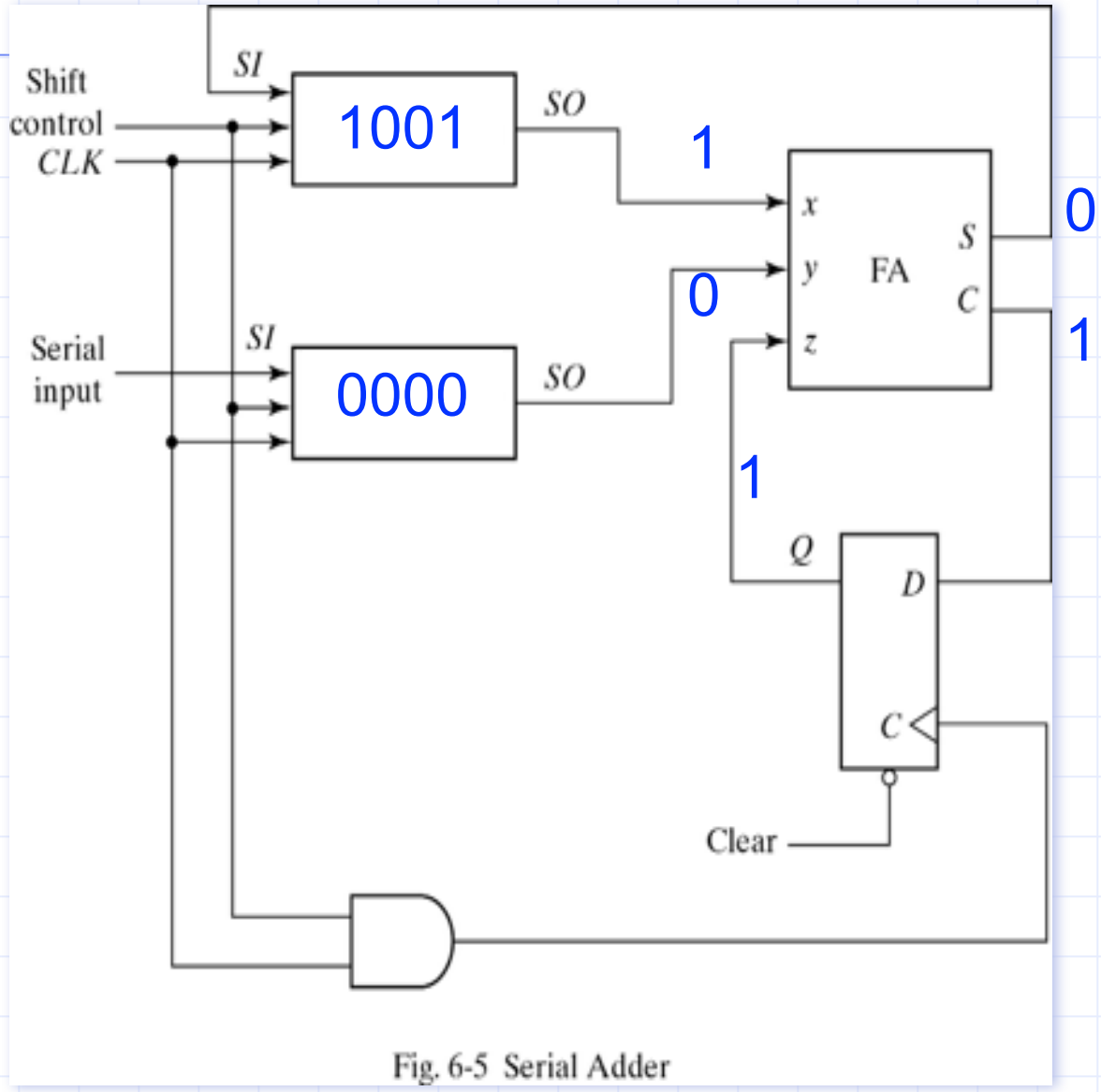


Fig. 6-5 Serial Adder

Serial Addition

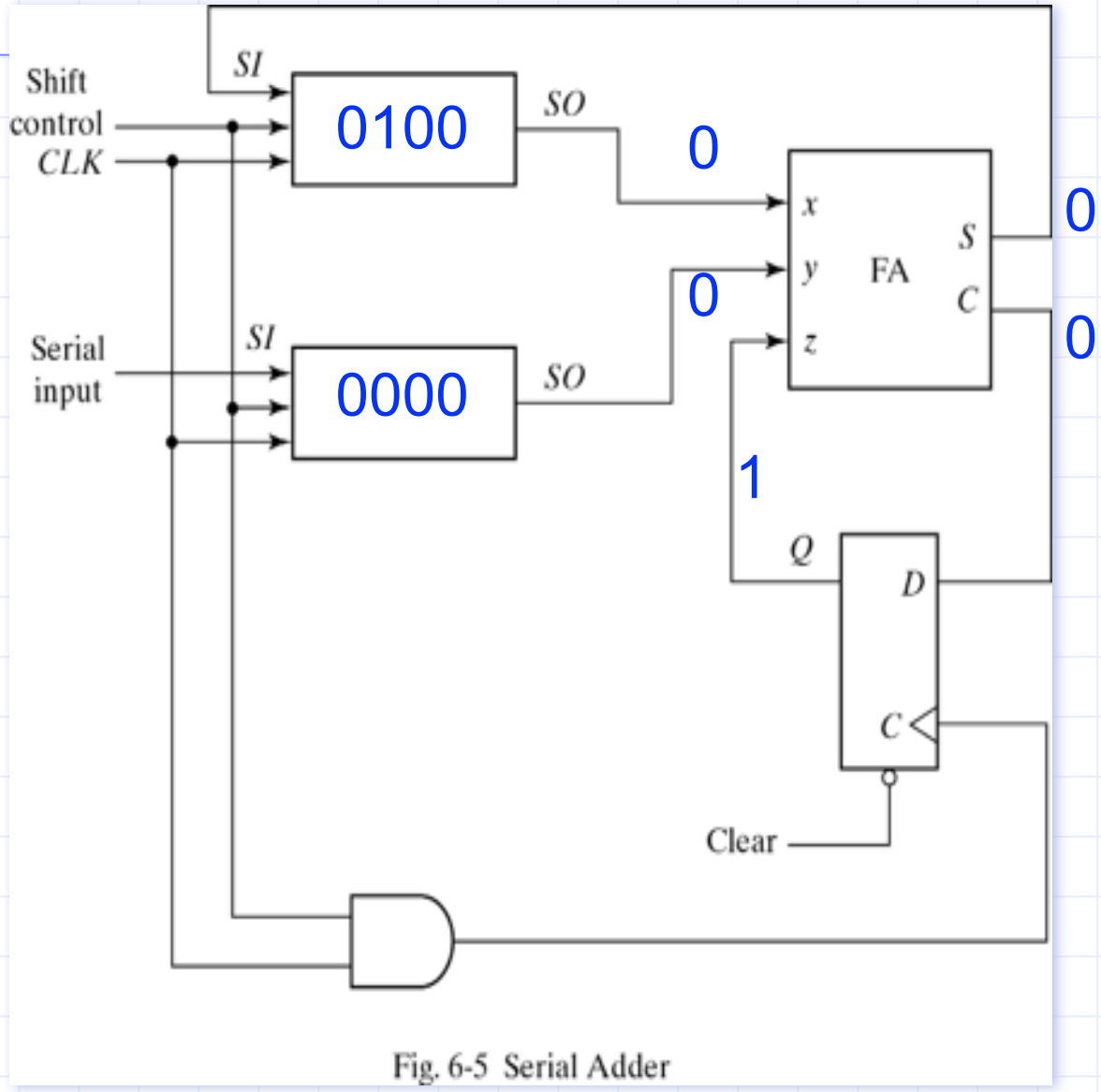


Fig. 6-5 Serial Adder

State Table for Serial Adder

Present value of carry

Output carry

Table 6-2
State Table for Serial Adder

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	X	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

$$J_Q = xy$$
$$K_Q = x'y' = (x+y)'$$
$$S = x \oplus y \oplus Q$$

By k-map

Second form of Serial Adder

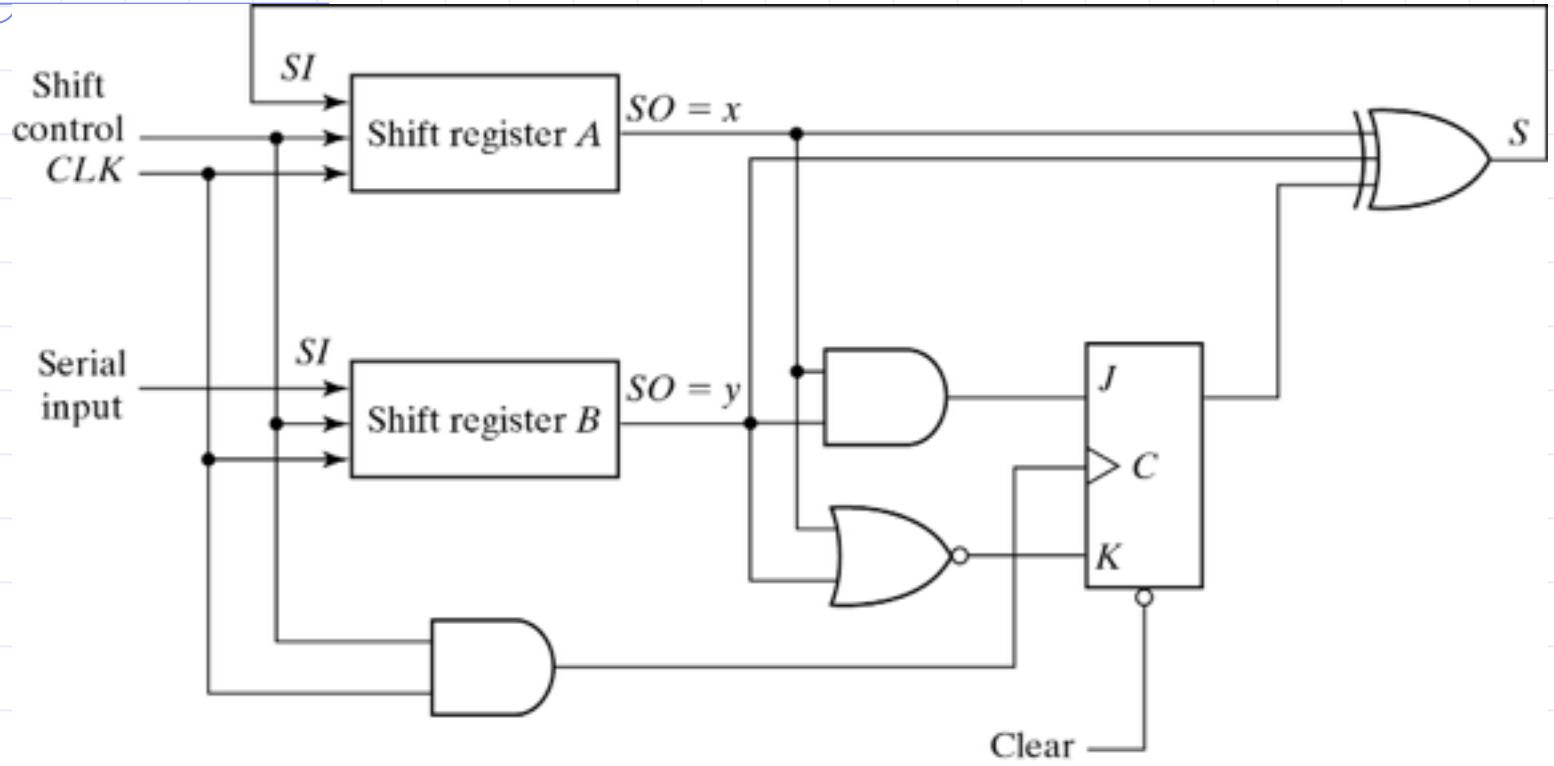


Fig. 6-6 Second form of Serial Adder

Universal Shift Register

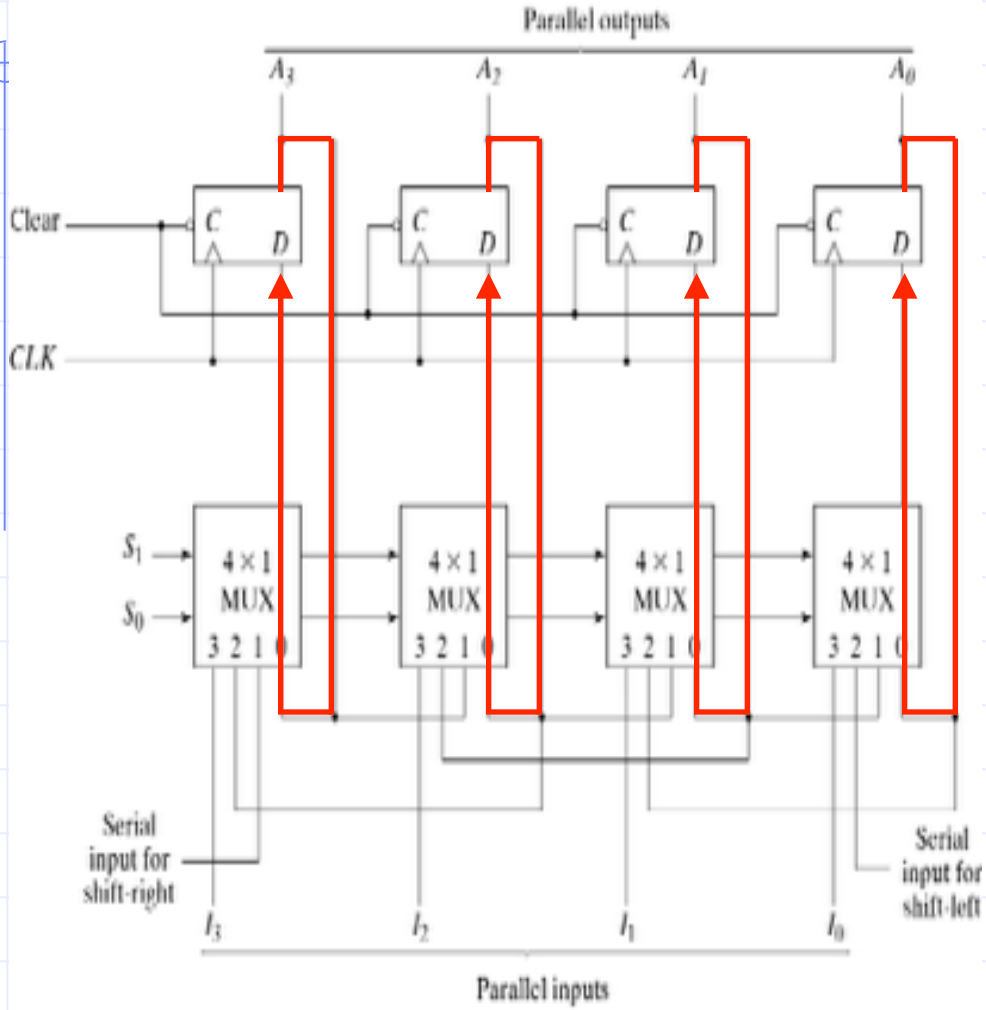


Fig. 6-7 4-Bit Universal Shift Register

- **S_1, S_0** -> 0, 0 **;No change**

Universal Shift Register

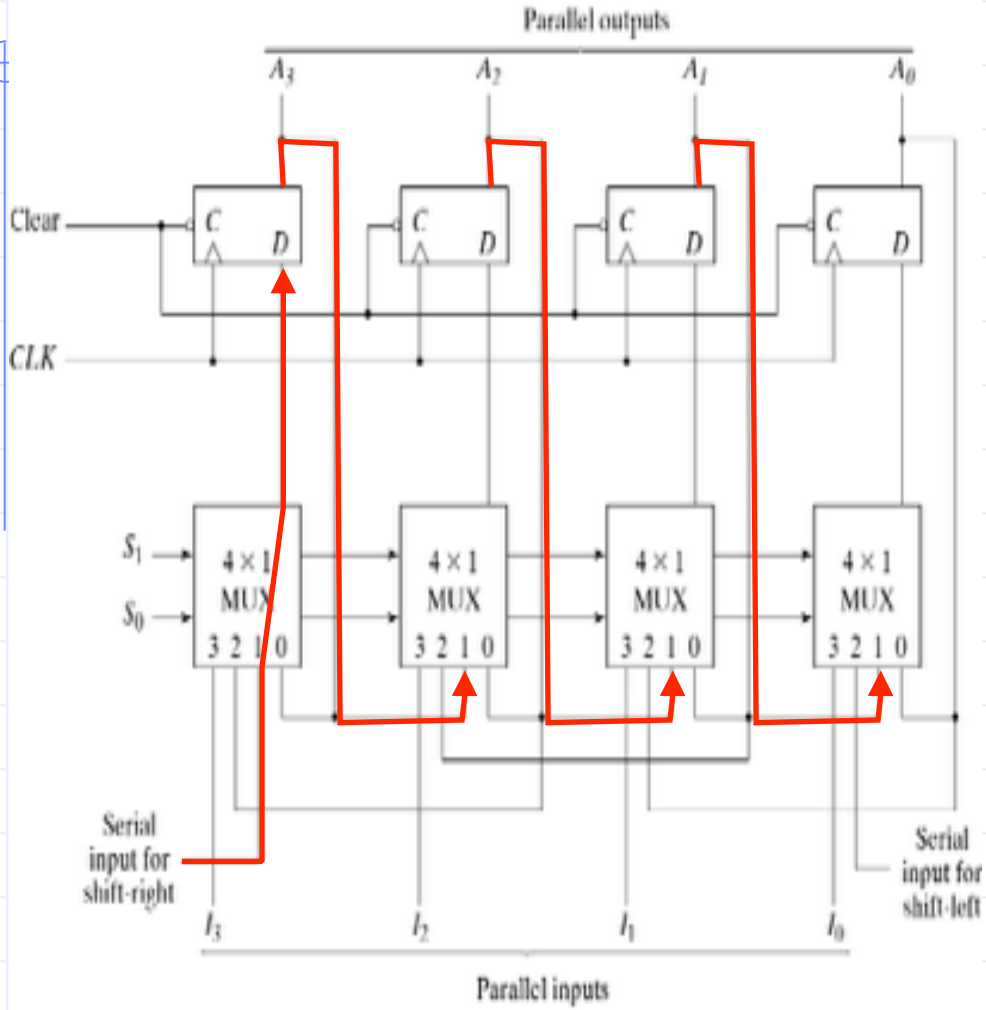


Fig. 6-7 4-Bit Universal Shift Register

- **S_1, S_0** -> 0, 1 **;Shift right**

Universal Shift Register

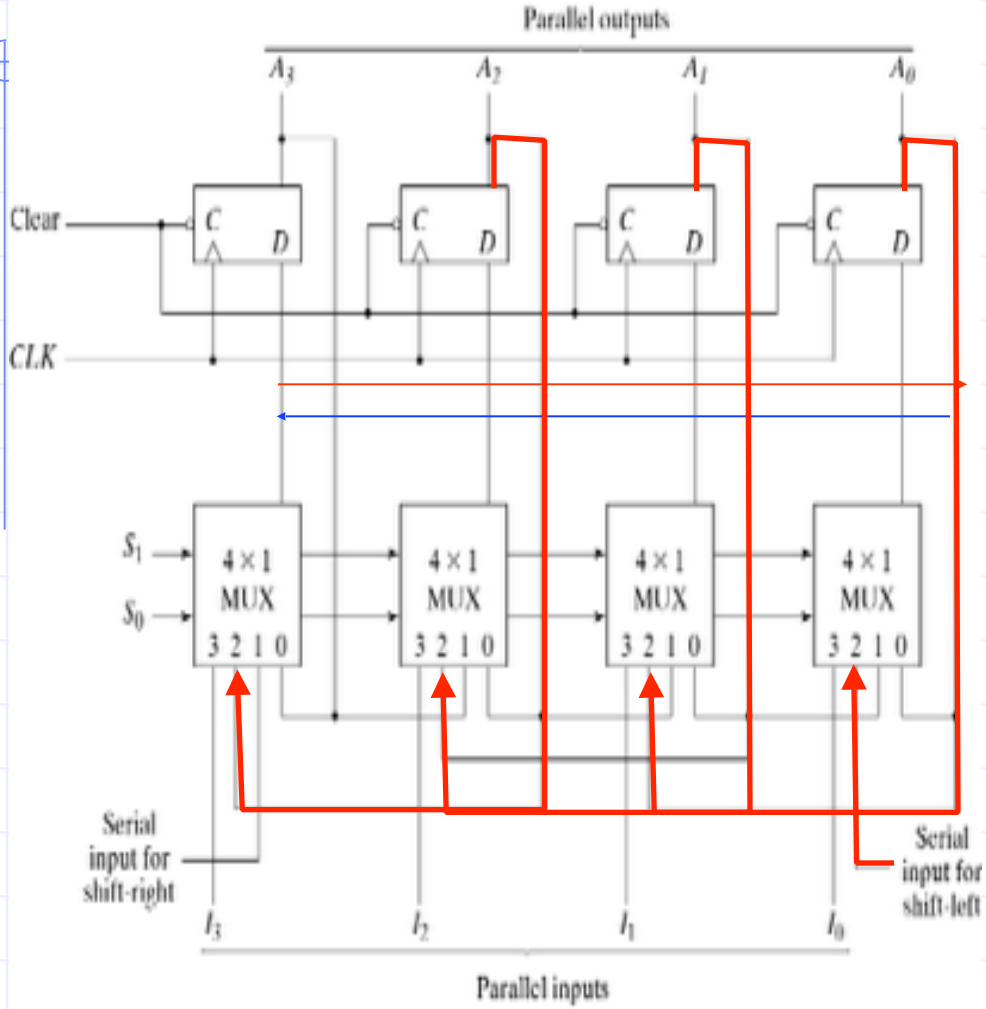


Fig. 6-7 4-Bit Universal Shift Register

- **S_1, S_0 -> 1, 0 ;Shift left**

Universal Shift Register

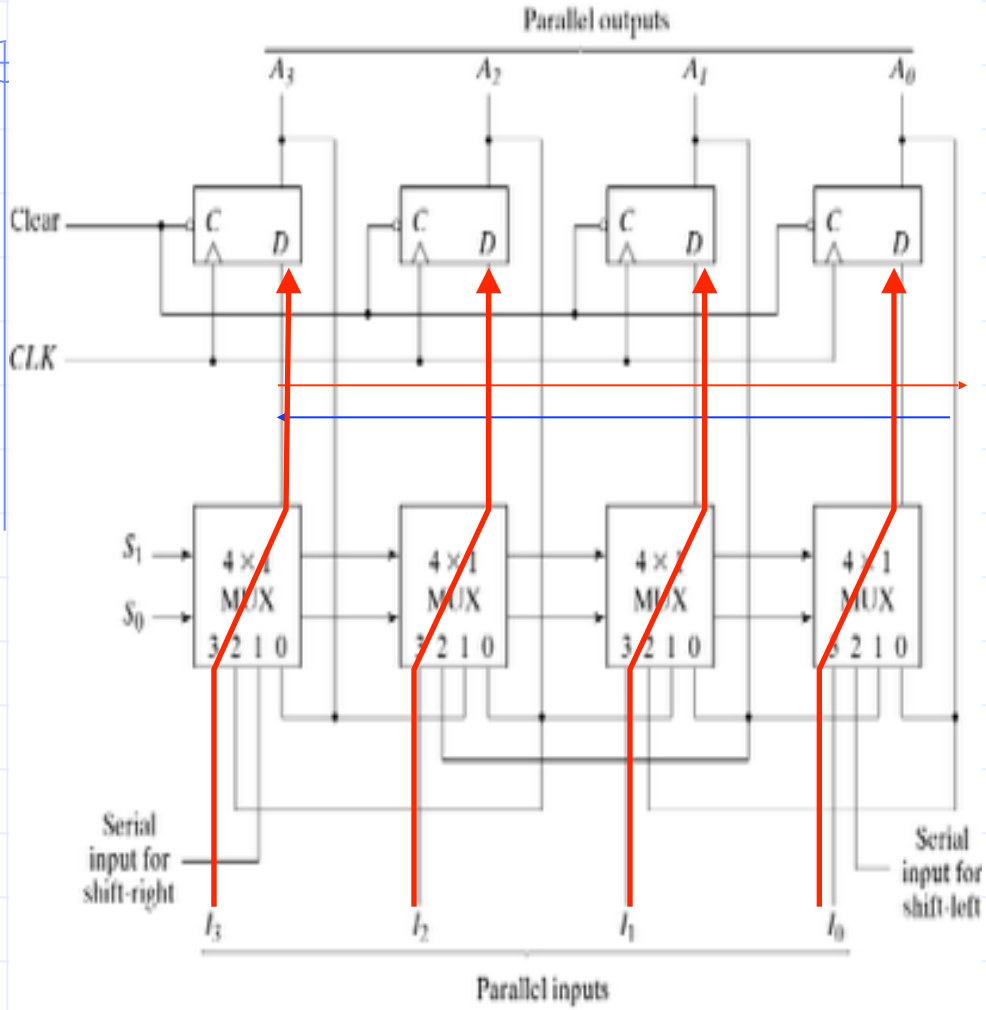
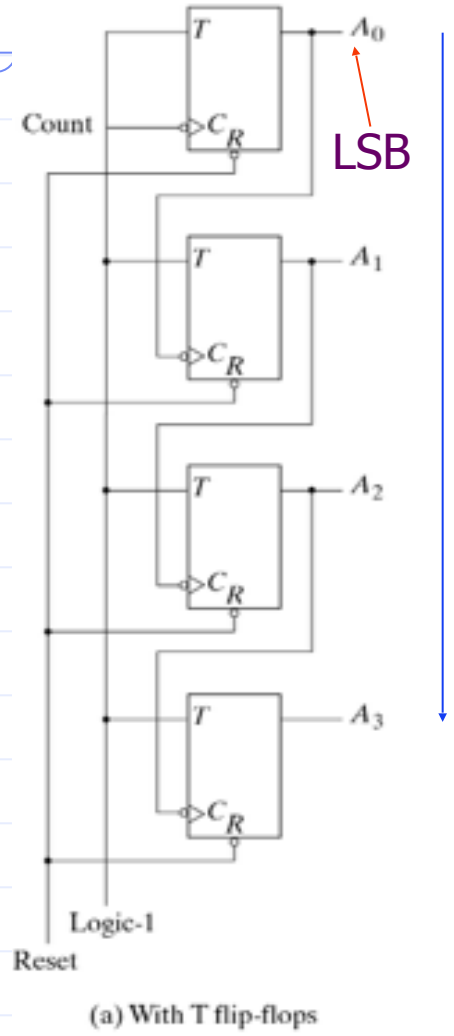


Fig. 6-7 4-Bit Universal Shift Register

- **S_1, S_0 -> 1, 1 ;Parallel load**

6-3 RIPPLE COUNTERS



(a) With T flip-flops

0000 → 0001: Count: 1 -> 0

0001 → 0010: Count: 1 -> 0, A_0: 1->0

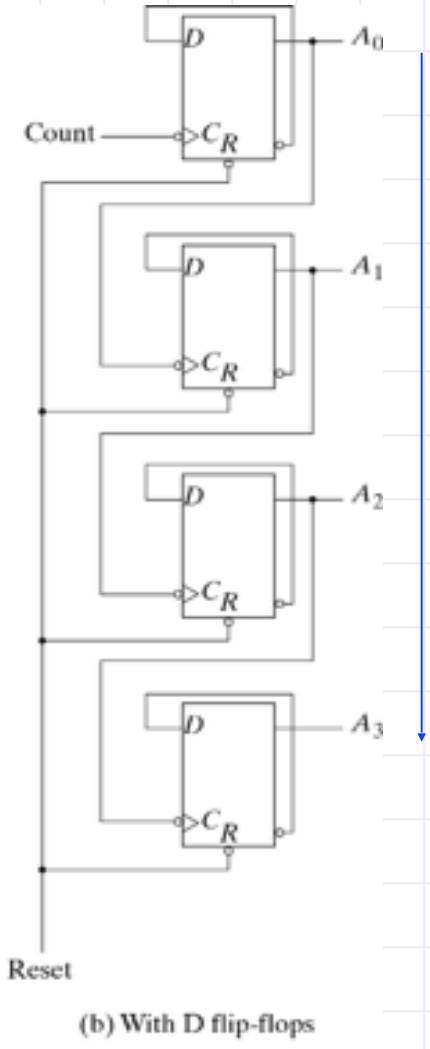
0010 → 0011: Count: 1 -> 0

0011 → 0100: Count: 1 -> 0, A_0: 1->0, A_1: 1->0

…

1111 → 0000: Count: 1 -> 0,
A_0: 1->0, A_1: 1->0,
A_2: 1->0, A_3: 1->0

6-3 RIPPLE COUNTERS



(b) With D flip-flops

0000 → 0001: Count: 1 -> 0

0001 → 0010: Count: 1 -> 0, A_0: 1->0

0010 → 0011: Count: 1 -> 0

0011 → 0100: Count: 1 -> 0, A_0: 1->0, A_1: 1->0

…

1111 → 0000: Count: 1 -> 0,
A_0: 1->0, A_1: 1->0,
A_2: 1->0, A_3: 1->0

Count sequence for a binary Counter

Count sequence A_3 A_2 A_1 A_0		Conditions for Complementing
0 0 0 0	Complement A_0	
0 0 0 1	Complement A_0	A_0 will go from 1 to 0 and complement A_1
0 0 1 0	Complement A_0	
0 0 1 1	Complement A_0	A_0 will go from 1 to 0 and complement A_1 ;
0 1 0 0	Complement A_0	A_1will go from 1 to 0 and complement A_2
0 1 0 1	Complement A_0	
0 1 1 0	Complement A_0	A_0 will go from 1 to 0 and complement A_1
0 1 1 1	Complement A_0	
.....		
1 0 0 0	and so on…	

BCD Ripple Counter

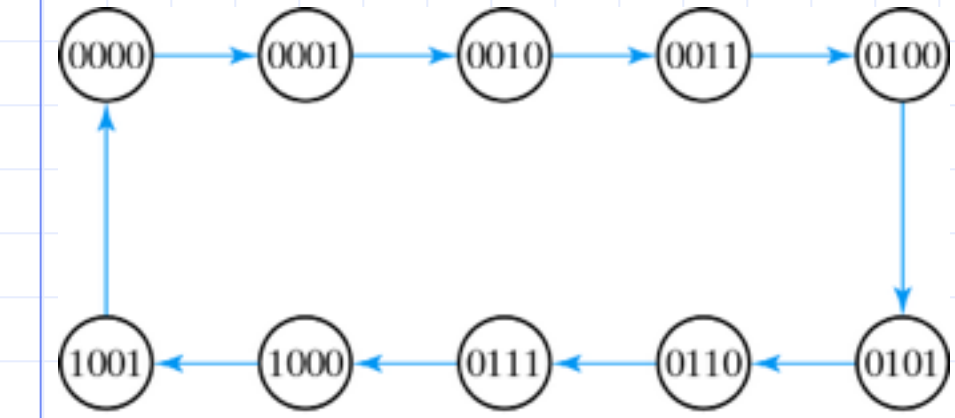


Fig. 6-9 State Diagram of a Decimal BCD-Counter

Count: 1 -> 0

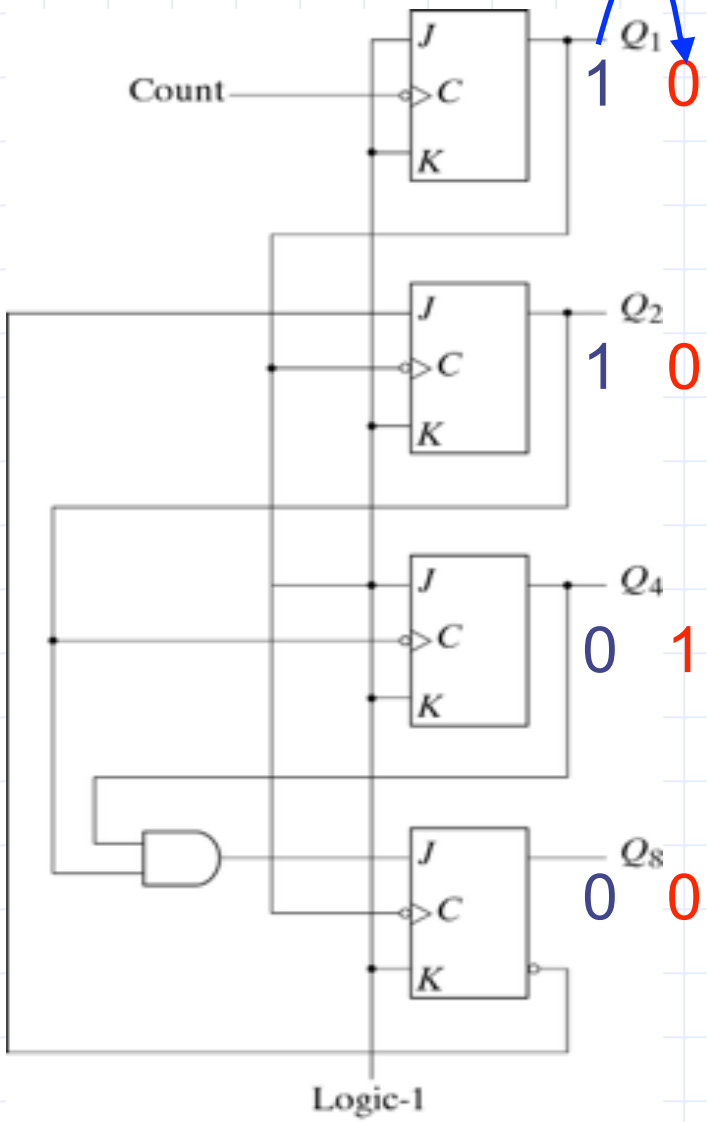


Fig. 6-10 BCD Ripple Counter

BCD Ripple Counter

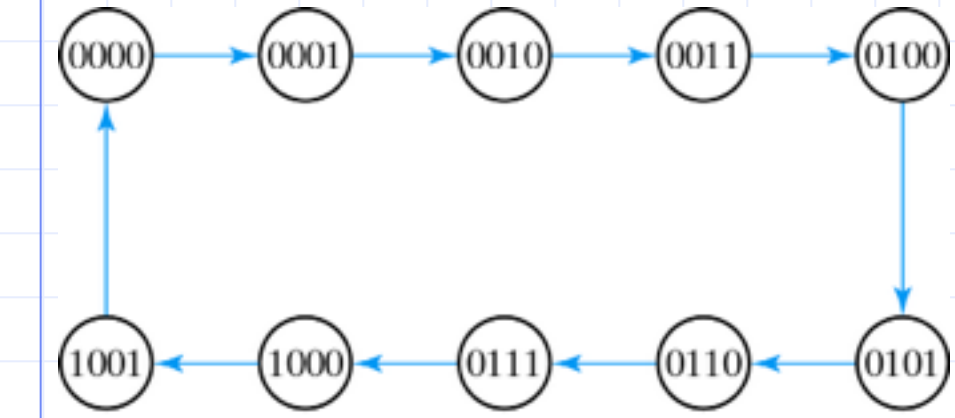


Fig. 6-9 State Diagram of a Decimal BCD-Counter

Count: 1 -> 0

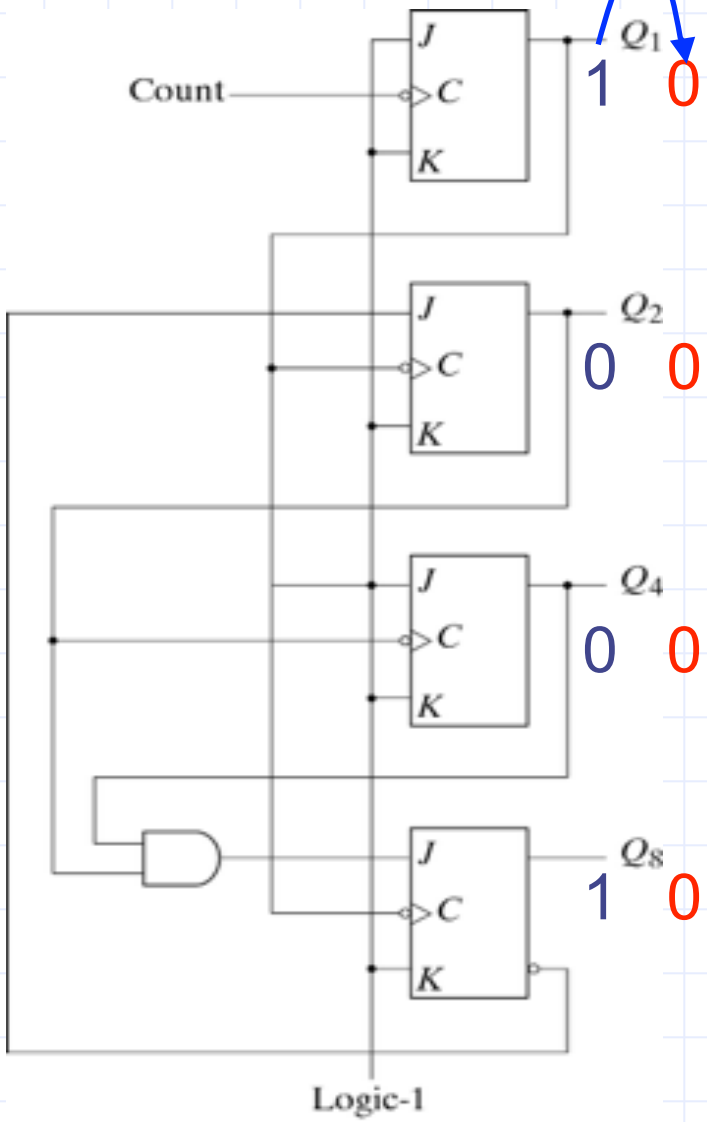


Fig. 6-10 BCD Ripple Counter

BCD Ripple Counter

operation

1. Q_1 is complemented on the negative edge of every count pulse.

2. Q_2 is complemented if $Q_8=0$ and Q_1 goes from 1 to 0. Q_2 is cleared if $Q_8=1$ and Q_1 goes from 1 to 0.

3. Q_4 is complemented when Q_2 goes from 1 to 0.

4 .Q_8 is complemented when $Q_4Q_2=11$ and Q_1 goes from 1 to 0. Q_8 is cleared if either Q_4 or Q_2 is 0 and Q_1 goes from 1 to 0

Three-Decade Decimal BCD Counter

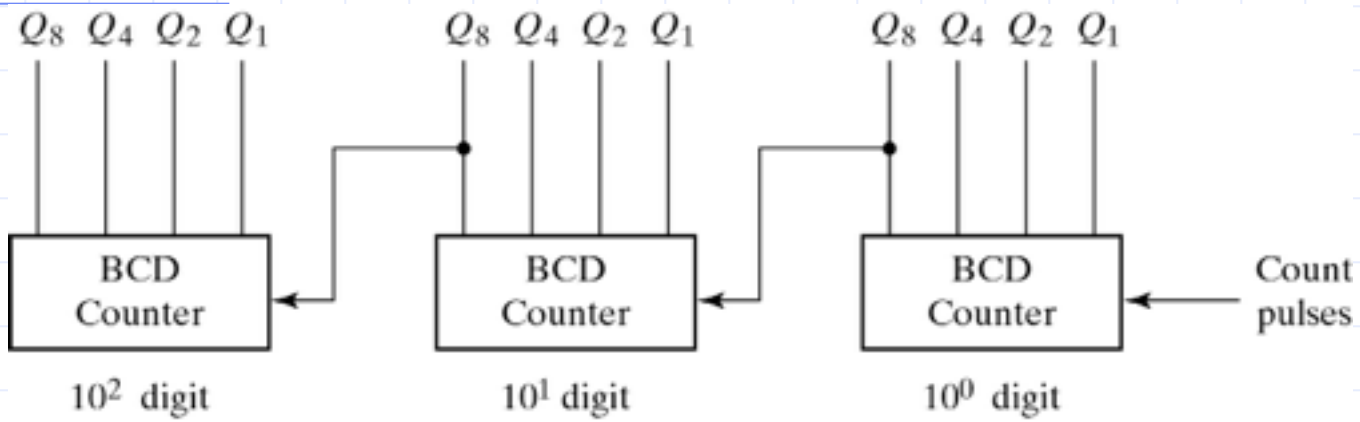


Fig. 6-11 Block Diagram of a Three-Decade Decimal BCD Counter

- To count from **0** to **999**, We need a three-decade counter.

6-4 SYNCHRONOUS COUNTERS

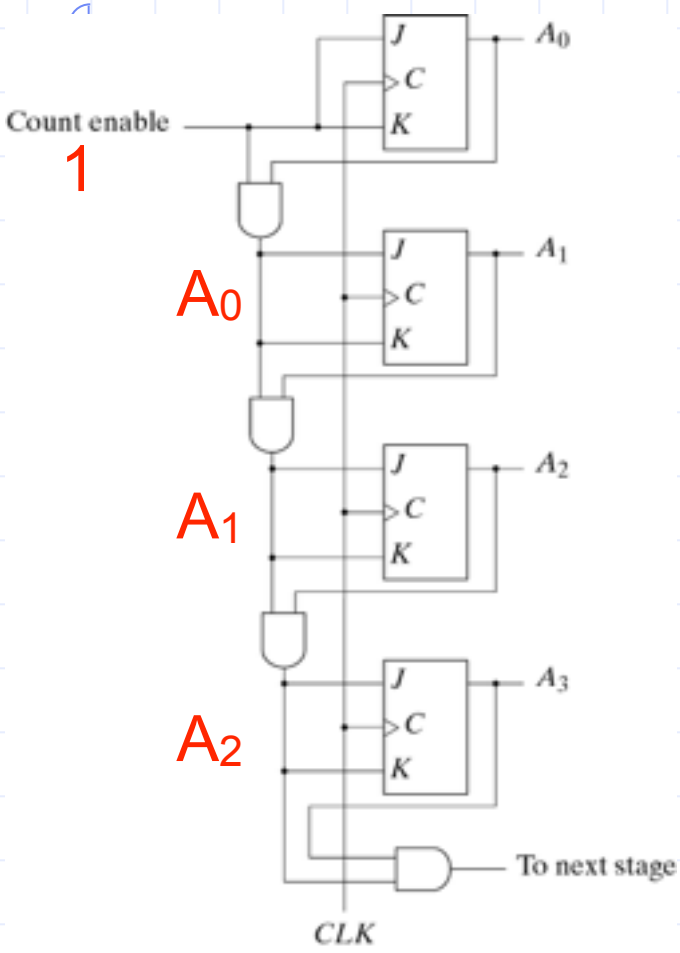


Fig. 6-12 4-Bit Synchronous Binary Counter

- The first stage **A_0** has its **J** and **K** equal to **1** if the counter is enabled .
- The other **J** and **K** inputs are equal to **1** if all previous low-order bits are equal to **1** and the count is enabled.
- **A_0** = 1이면, 클럭의 상승에지가 발생할 때 마다, **A_1**의 값은 반전된다.

Up-Down Binary Counter

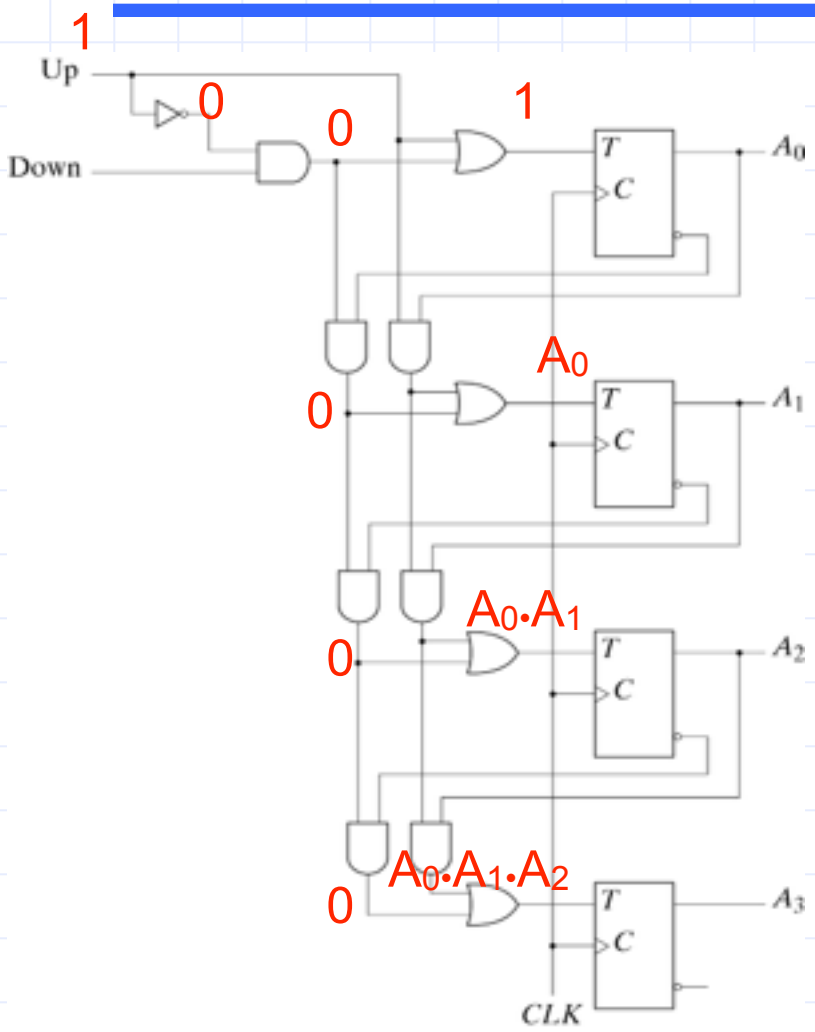


Fig. 6-13 4-Bit Up-Down Binary Counter

- **Up** input control=**1** ;count up (the **T** inputs receive their signals from the values of the previous normal outputs of the flip-flops.)

Up-Down Binary Counter

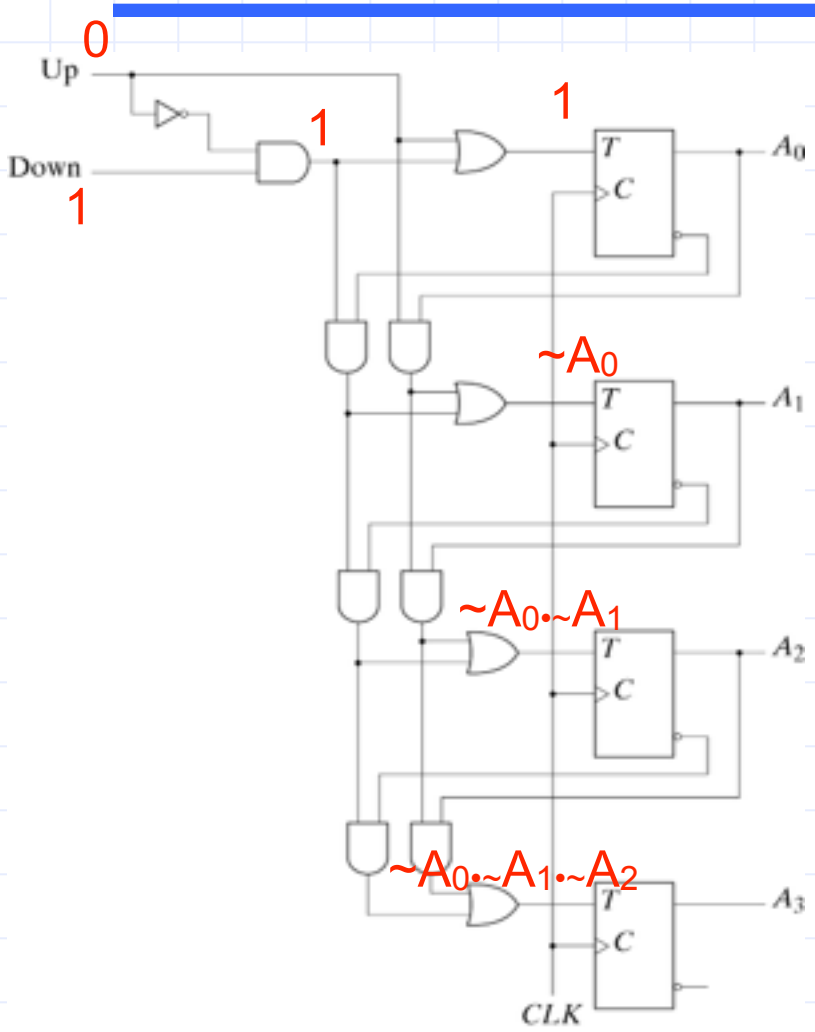
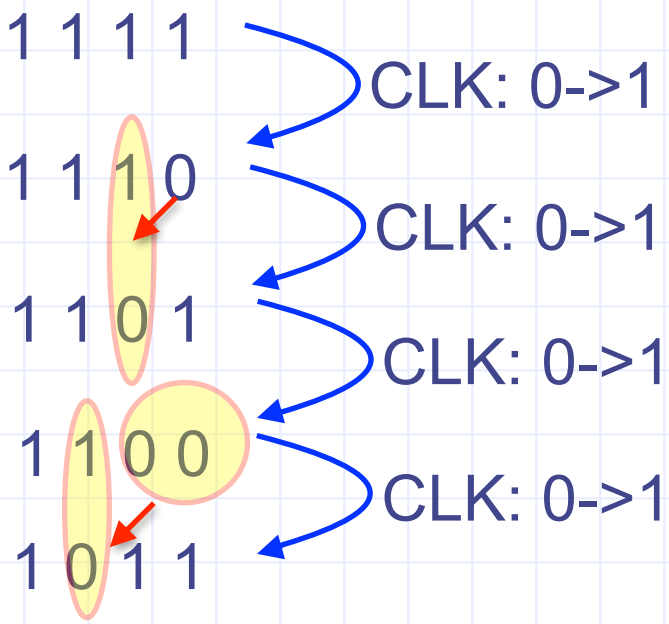


Fig. 6-13 4-Bit Up-Down Binary Counter

❑**Down** input control=**1**, **up** input control=**0** ; count down



Up-Down Binary Counter

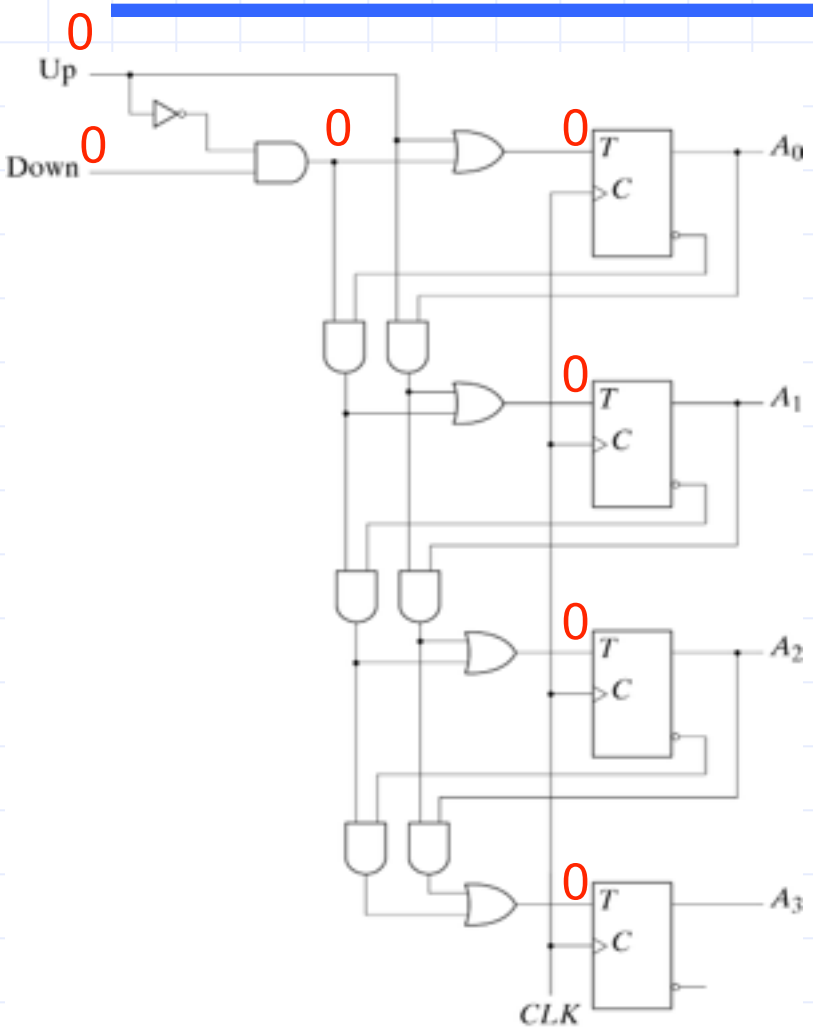


Fig. 6-13 4-Bit Up-Down Binary Counter

- **Up**=**down**=0 ;unchanged state

Up-Down Binary Counter

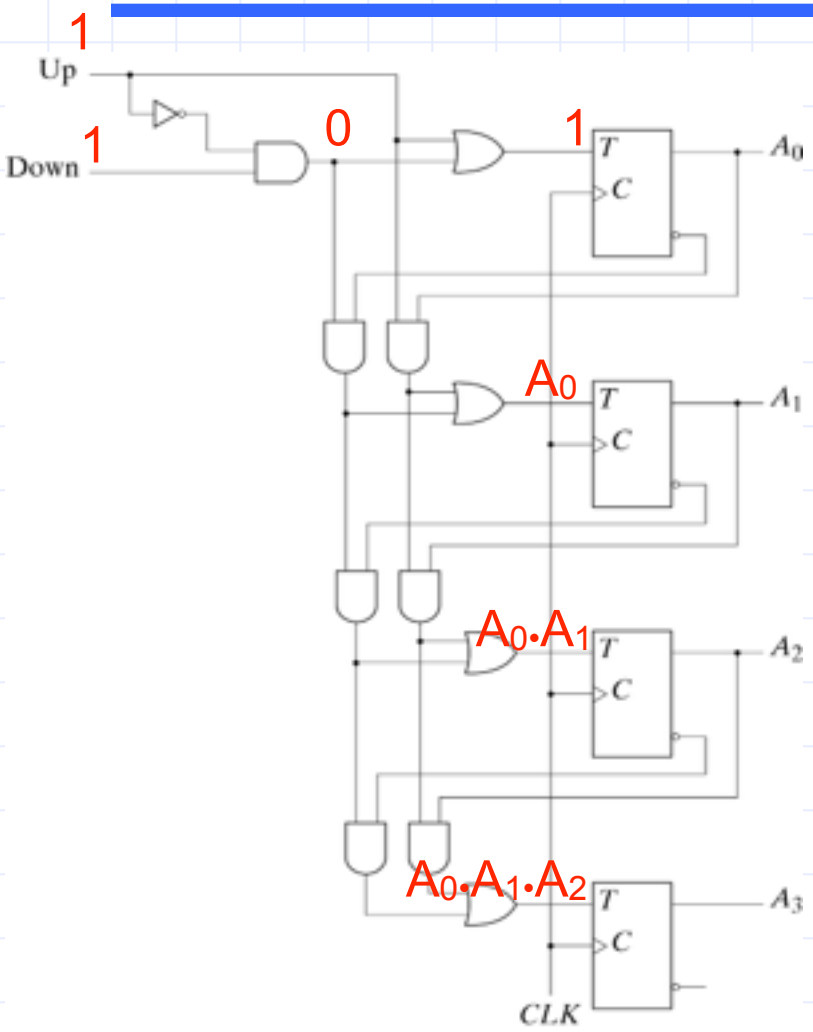


Fig. 6-13 4-Bit Up-Down Binary Counter

- **Up**=**down**=1 ;count up

BCD Counter

Table 6-5
State Table for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

$$T_{Q1} = 1$$
$$T_{Q2} = Q'_8 Q_1$$
$$T_{Q4} = Q_2 Q_1$$
$$T_{Q8} = Q_8 Q_1 + Q_4 Q_2 Q_1$$
$$y = Q_8 Q_1$$

Binary Counter with Parallel Load

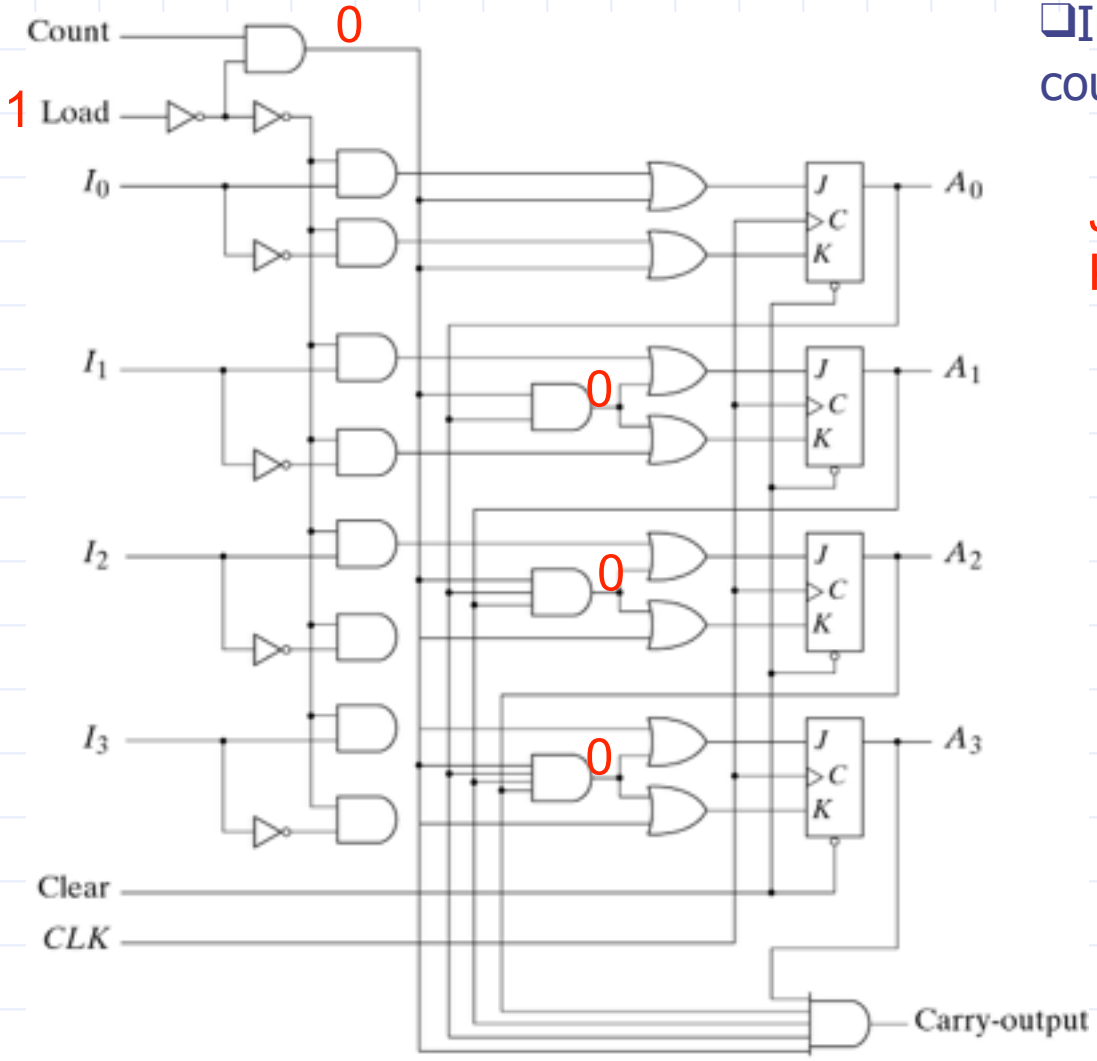


Fig. 6-14 4-Bit Binary Counter with Parallel Load

- Input **load** control=1 ; disables the count sequence ,data transfer

$J_i = I_i$
$K_i = \sim I_i$
$A_i = I_i$

Binary Counter with Parallel Load

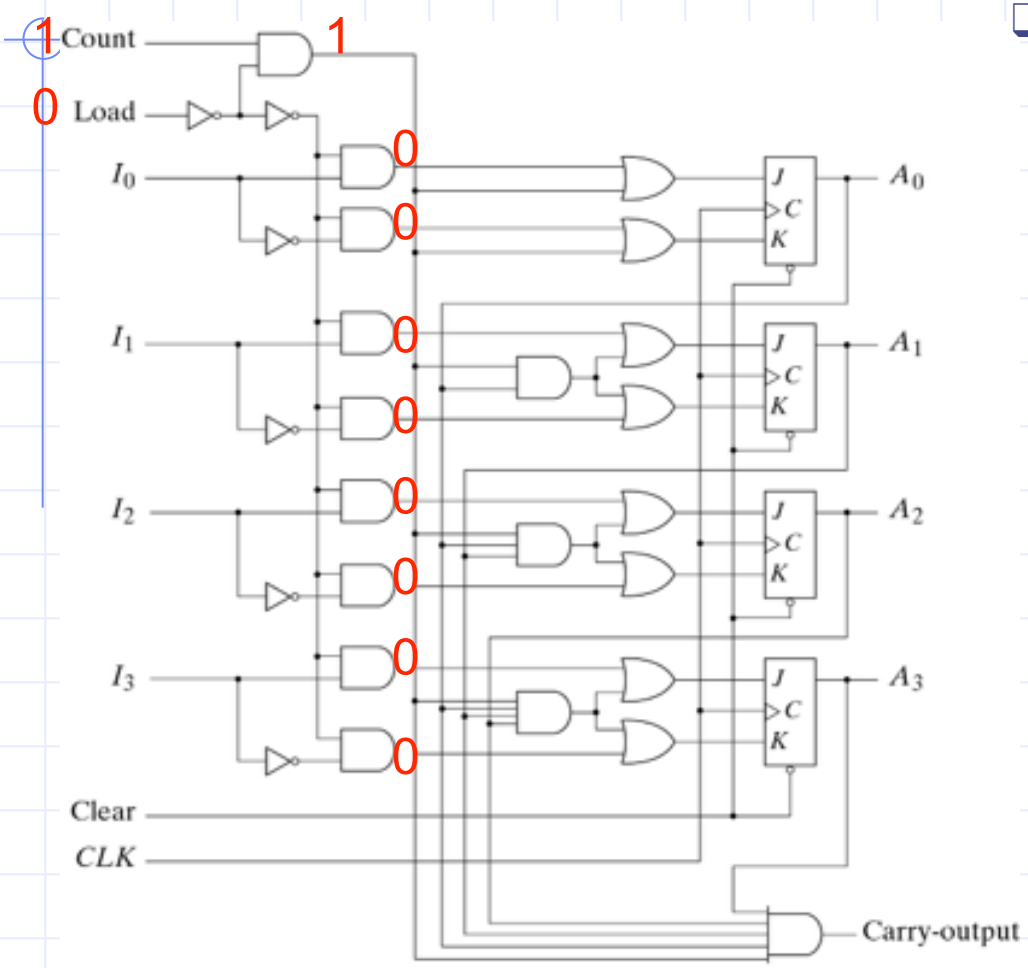


Fig. 6-14 4-Bit Binary Counter with Parallel Load

- **Load** =0 and **count**=1 ;count

$J_0=1$
$K_0 = 1$

$J_1=A_0$
$K_1 = A_0$

$J_2=A_0 \cdot A_1$
$K_2 = A_0 \cdot A_1$

$J_3=A_0 \cdot A_1 \cdot A_2$
$K_3 = A_0 \cdot A_1 \cdot A_2$

0 0 0 0 → CLK: 0->1
0 0 0 1 → CLK: 0->1
0 0 1 0 → CLK: 0->1
0 0 1 1 → CLK: 0->1
0 1 0 0

Binary Counter with Parallel Load

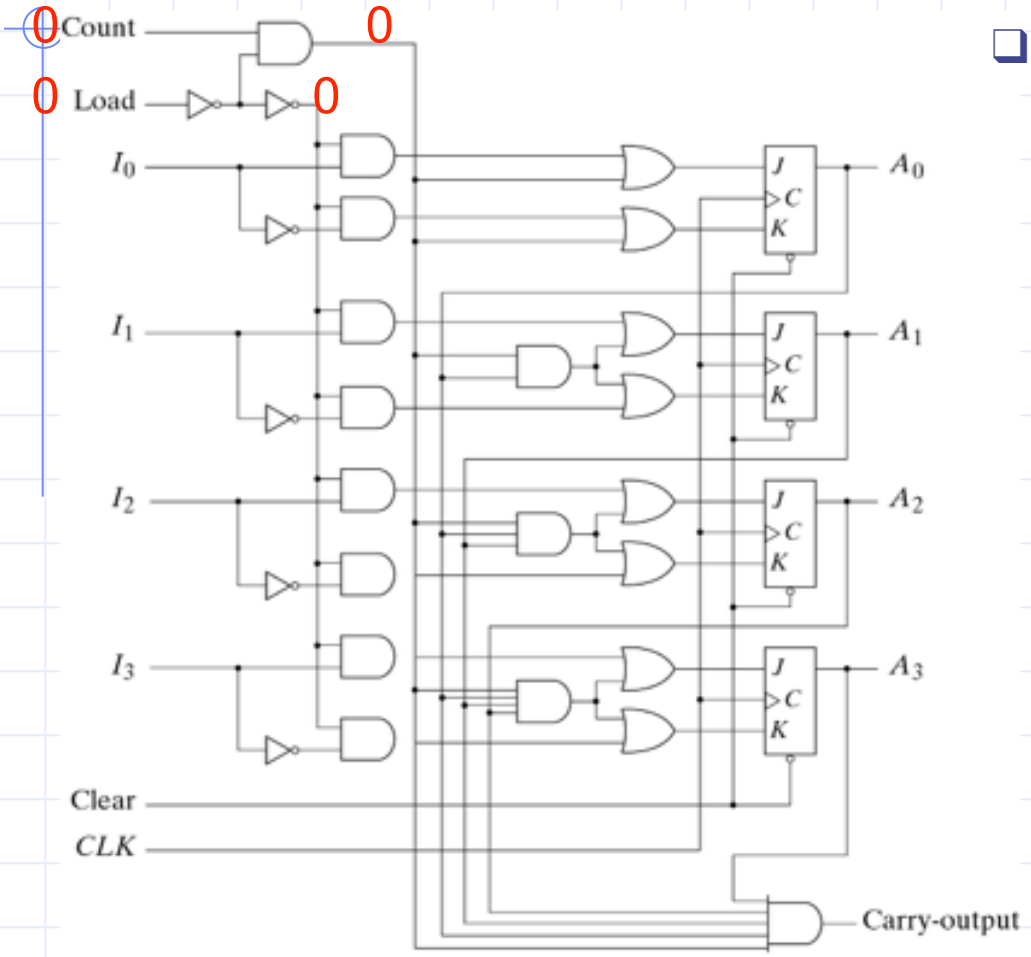


Fig. 6-14 4-Bit Binary Counter with Parallel Load

- **Load**=0 and **count**=0 ;unchanged

$J_i = 0$

$K_i = 0$

Binary Counter with Parallel Load

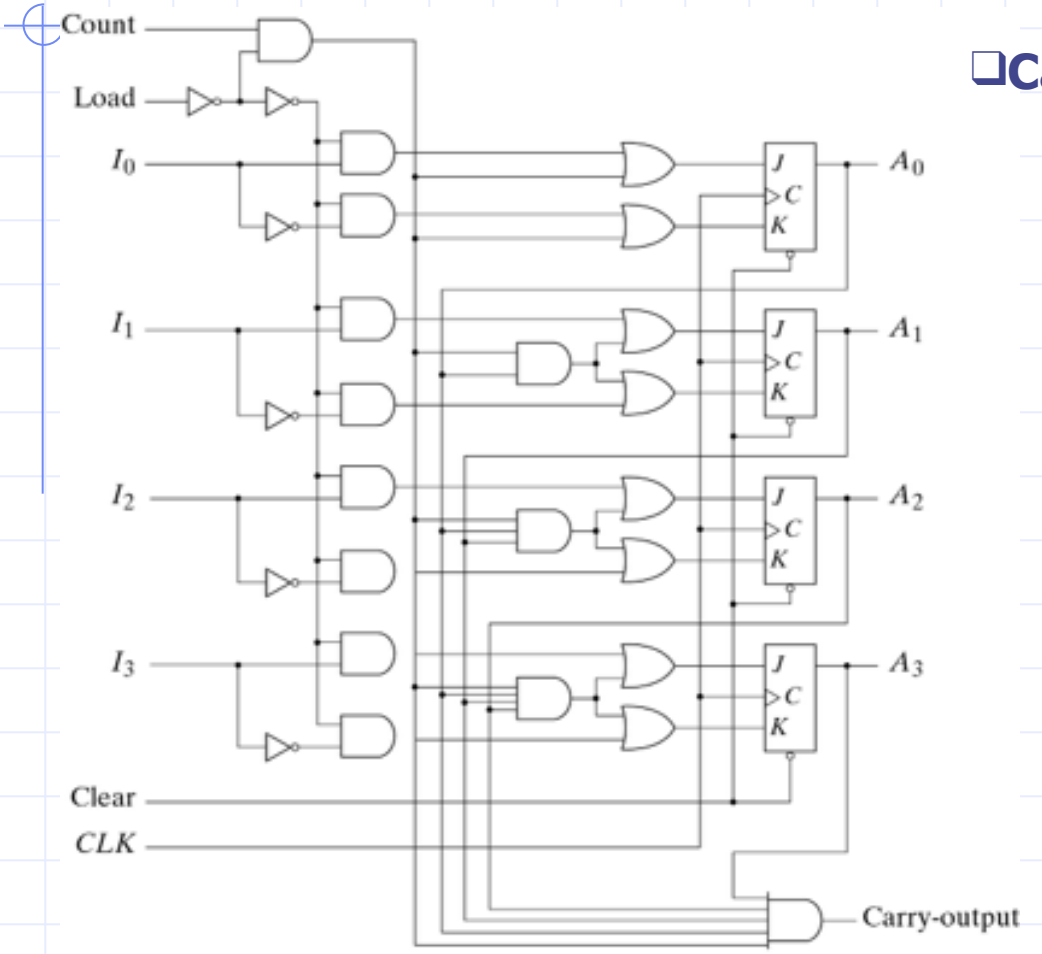


Fig. 6-14 4-Bit Binary Counter with Parallel Load

- **Carry out**=1(all flip-flop=1)

BCD COUNTER using Binary Counter with Parallel Load

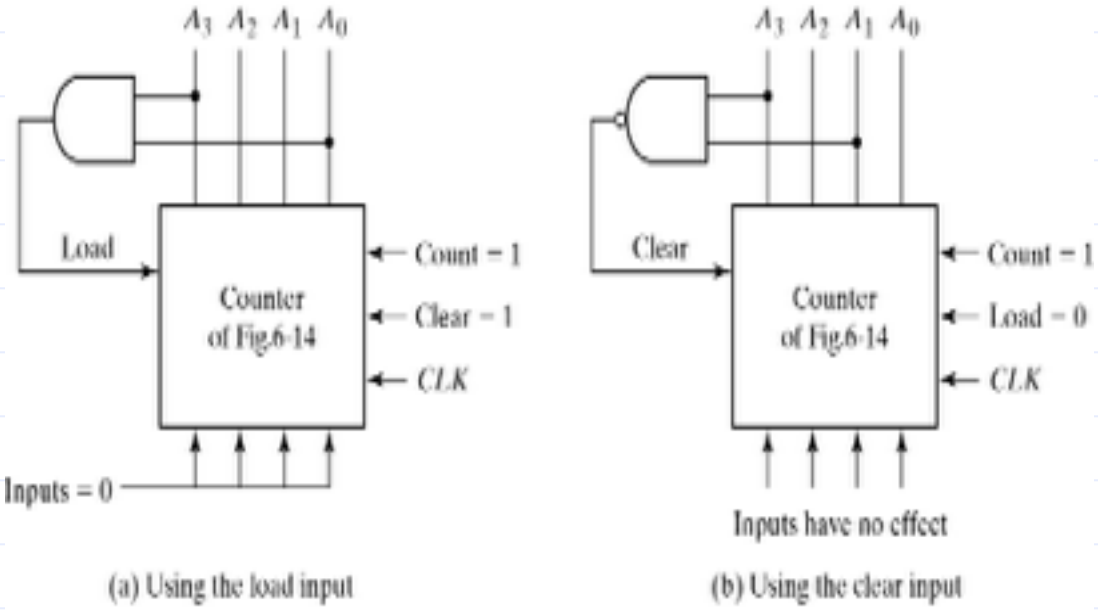
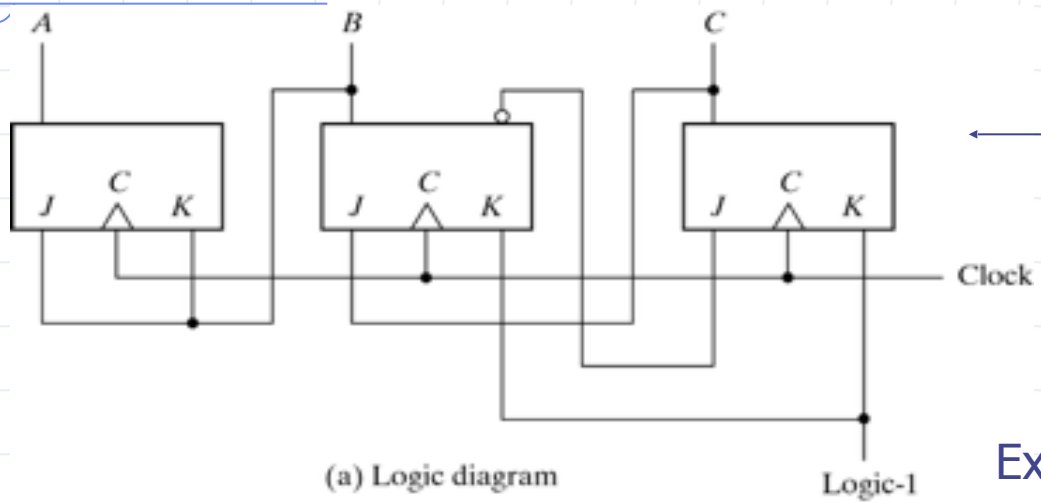


Fig. 6-15 Two ways to Achieve a BCD Counter Using a Counter with Parallel Load

- The **AND** gate detects the occurrence of state **1001(9)** in the output. In this state, the load input is enabled and all-**0**'s input is loaded into register.
- The **NAND** gate detects the count of **1010(10)** , as soon as this count occurs the register is **cleared.**
- A momentary **spike** occurs in output A2 as the count goes from **1001** to **1010** and immediately to **0000**

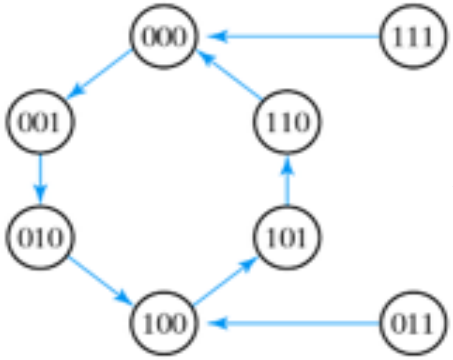
6-5 OTHER COUNTERS



(a) Logic diagram

$$J_A = B \qquad K_A = B$$
$$J_B = C \qquad K_B = 1$$
$$J_C = B' \qquad K_C = 1$$

Except **011 ,111**



(b) State diagram

Fig. 6-16 Counter with Unused States

Table 6-7
State Table for Counter

Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X

Ring Counter

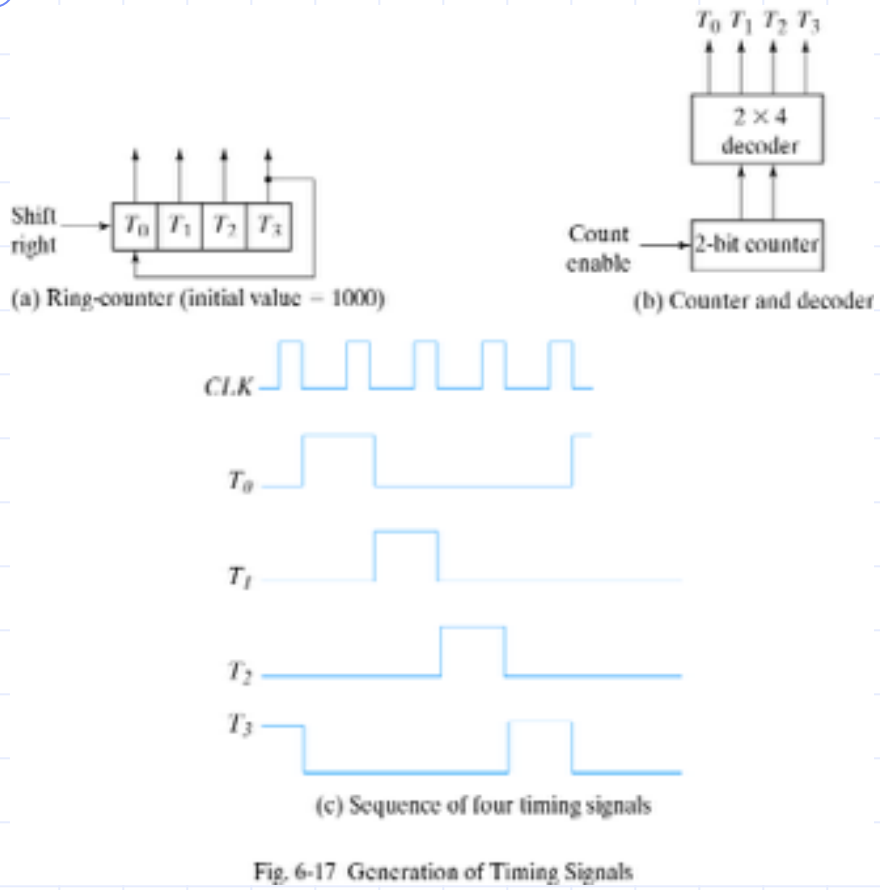
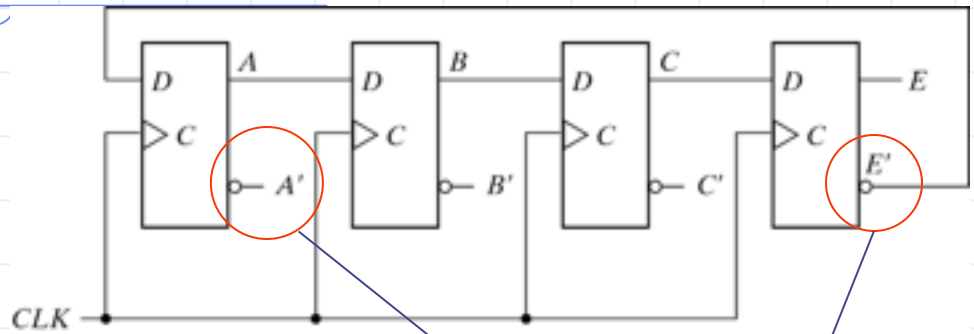


Fig. 6-17 Generation of Timing Signals

- A circular sift register with only one flip-flop being set at any **particular time**. ; all others are cleared.
- **The single bit** is shifted from one flip-flop to the other.

Johnson Counter



(a) Four-stage switch-tail ring counter

Sequence number	Flip-flop outputs A	B	C	E	AND gate required for output
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

(b) Count sequence and required decoding

Fig. 6-18 Construction of a Johnson Counter

- A circular shift register with **the complement output of the last flip-flop** connected to the input of the first flip-flop.