CS2022: 數位系統設計

Registers and Counters

Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

Registers

- **Clocked sequential circuits**
  - A group of flip-flops and combinational gates
  - Connected to form a feedback path
  - Flip-flops + Combinational gates
    (essential) (optional)
- **Register**
  - A group of flip-flops and gates
    - » Flip-flops store binary data
    - » Gates determine how the information is transferred into the register
- **Counter**
  - A register that goes through a predetermined sequence of states

Registers

- **A *n*-bit register**
  - ***n* flip-flops capable of storing *n* bits of binary information**
  - **4-bit register is shown in Fig. 1**

Clear_b = 0 (active low): $A_x = 0$

Clear_b = 1 (normal operation)

Clock = ↑: $A_x = I_x$

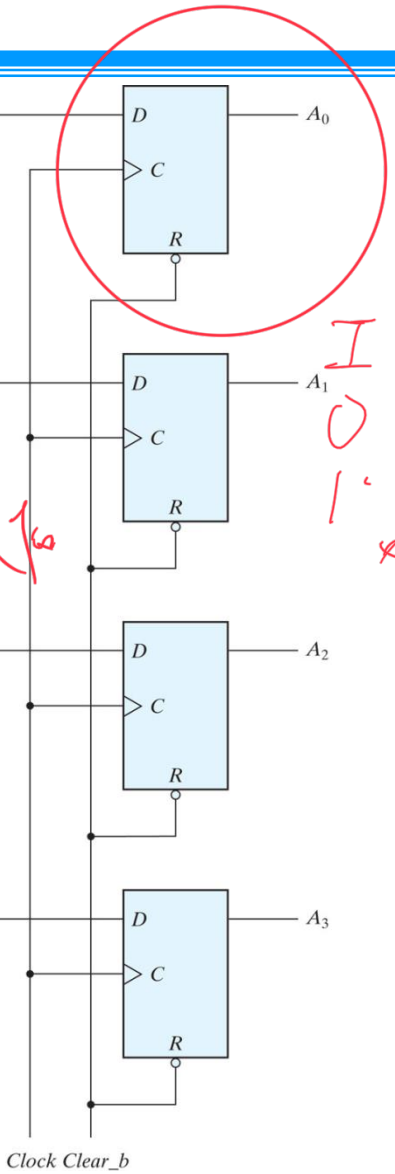


Fig. 1 Four-bit register

4-bit Register with Parallel Load

4位元暫存器

Load = 1: Parallel load

Load = 0: No change

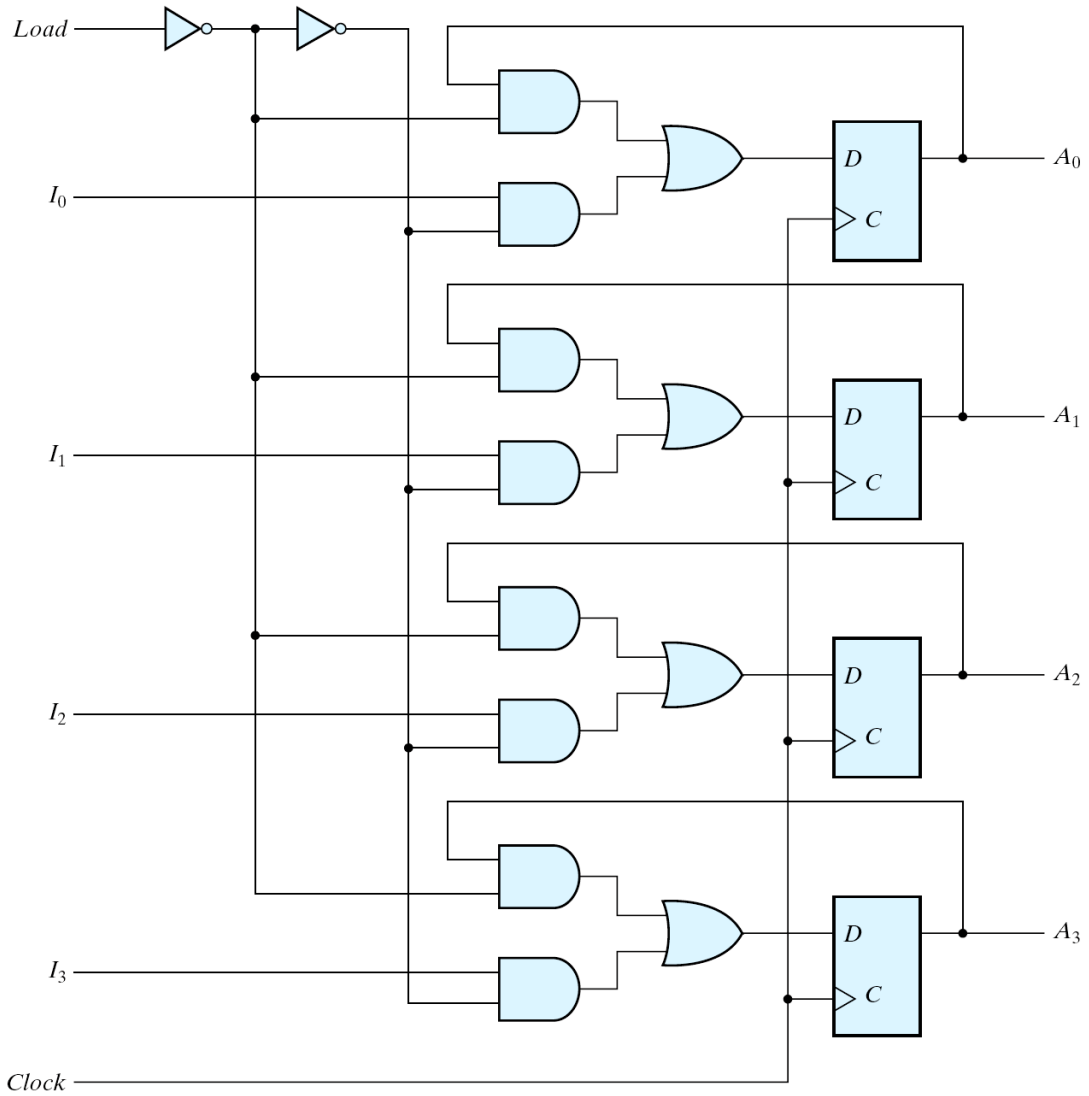


Fig. 2 Four-bit register with parallel load

Shift Registers

- **Shift register**
  - **A register capable of shifting its binary information in one or both directions**
- **Simplest shift register**

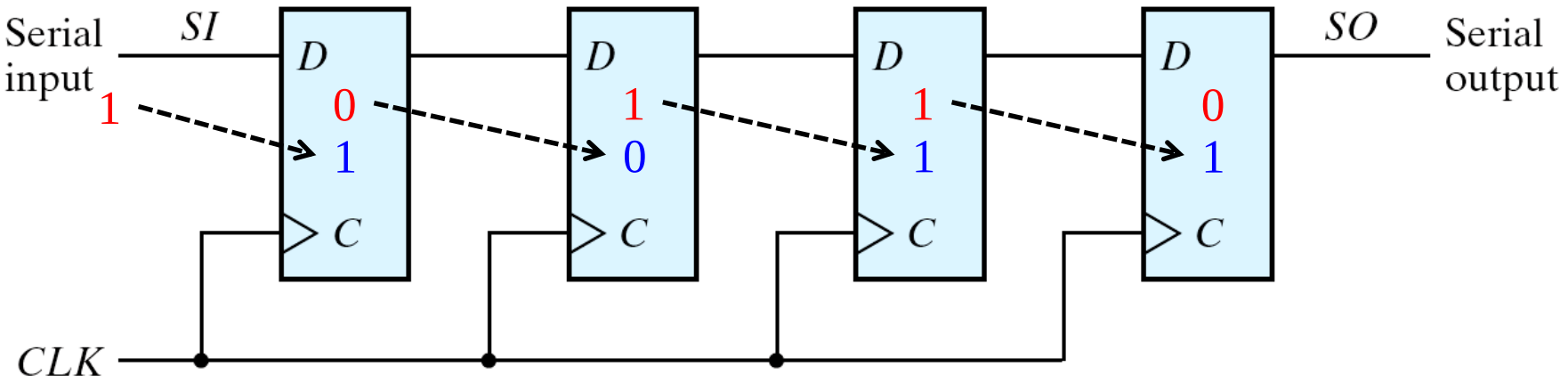


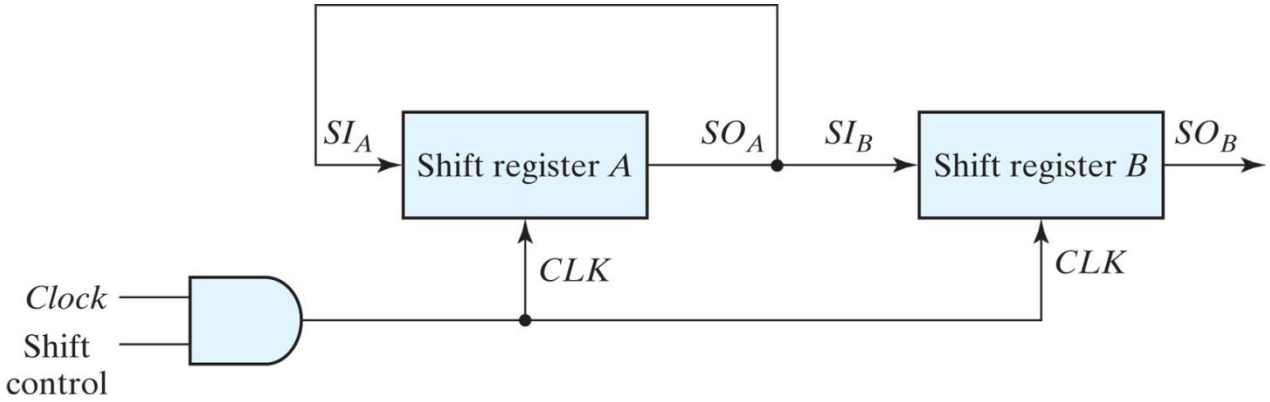
Fig. 3 Four-bit shift register

Data Transfer

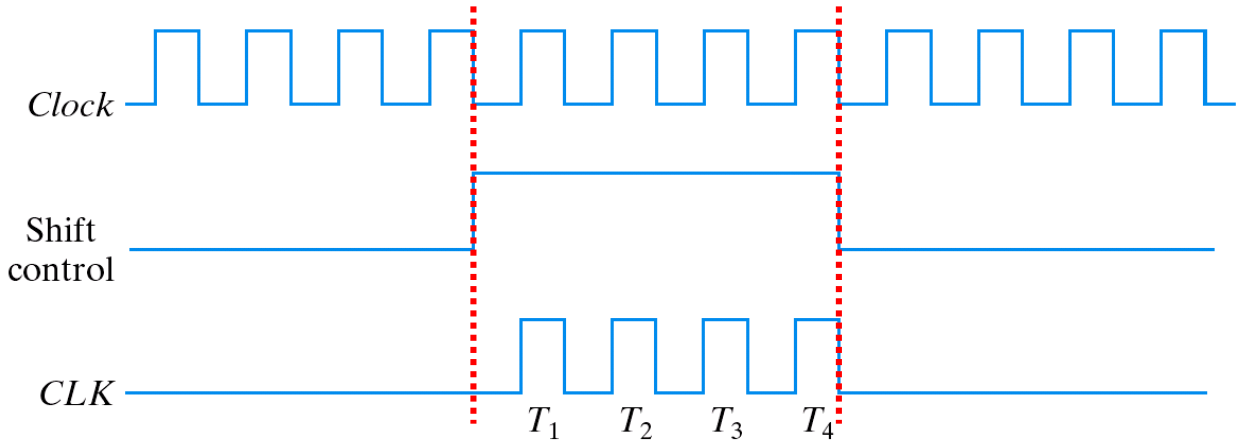
- **Serial transfer vs. Parallel transfer**
  - **Serial transfer**
    - **Information is transferred one bit at a time**
    - **Shifts the bits out of the source register into the destination register**
  - **Parallel transfer**
    - **All the bits of the register are transferred at the same time**

Serial Transfer (1/2)

- **Example: Serial transfer from register *A* to register *B***



(a) Block diagram

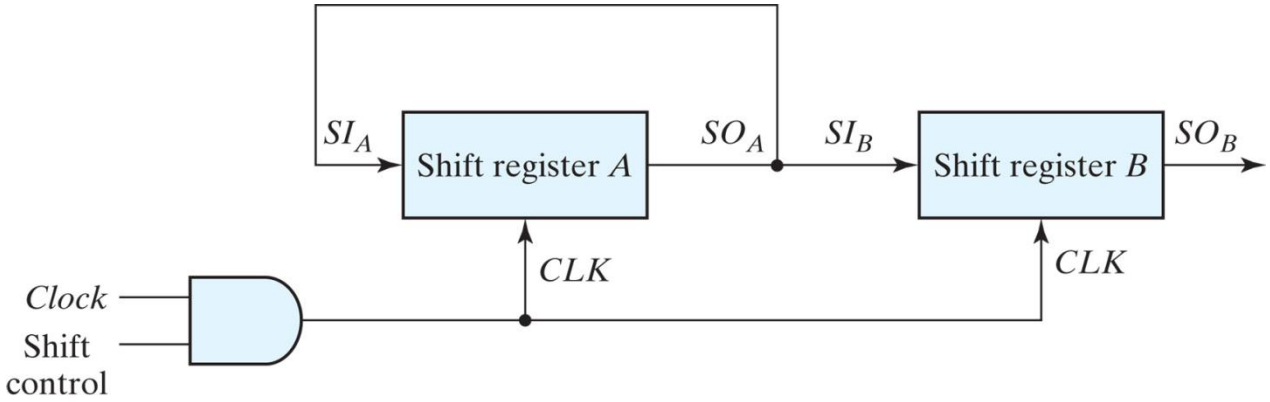


(b) Timing diagram

Fig. 4 Serial transfer from 4-bit register *A* to register *B*

Serial Transfer (2/2)

- **Example: Serial transfer from register *A* to register *B***



(a) Block diagram

Table 6.1
Serial-Transfer Example

Timing Pulse	Shift Register *A*				Shift Register *B*			
Initial value	1	0	1	1	0	0	1	0
After T_1	1	1	0	1	1	0	0	1
After T_2	1	1	1	0	1	1	0	0
After T_3	0	1	1	1	0	1	1	0
After T_4	1	0	1	1	1	0	1	1

Serial Addition Using D Flip-Flops

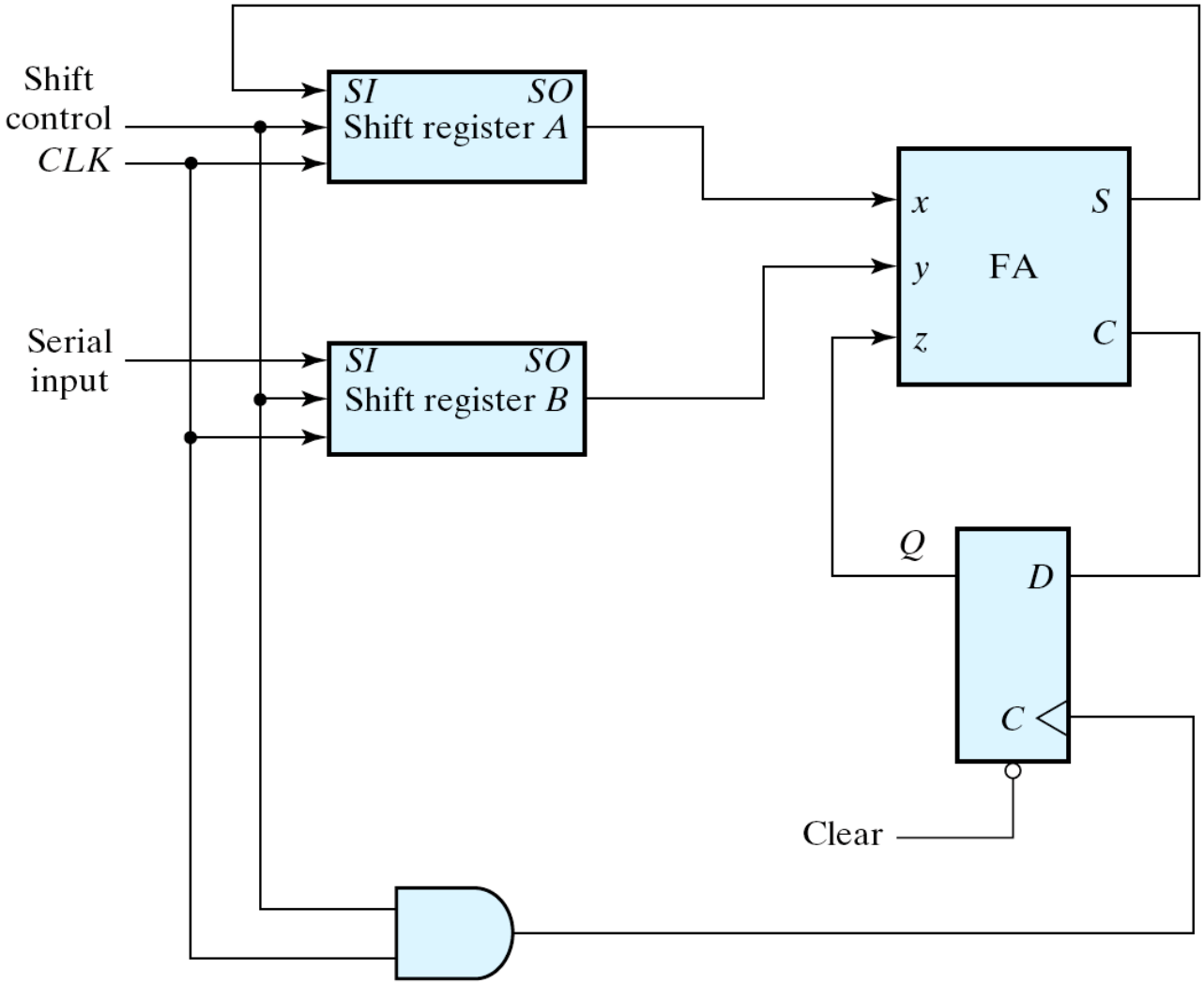


Fig. 5 Serial adder

Serial Adder Using JK FFs (1/2)

- Serial adder using *JK* flip-flops

Flip-Flop Excitation Table

Q(t)	Q(t + 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

JK

Table 6.2
State Table for Serial Adder

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q (C_{in})	x	y	Q (C_{out})	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

Serial Adder Using JK FFs (2/2)

- **Circuit diagram**
  - $J_Q = xy$
  - $K_Q = x'y' = (x+y)'$
  - $S = x \oplus y \oplus Q$

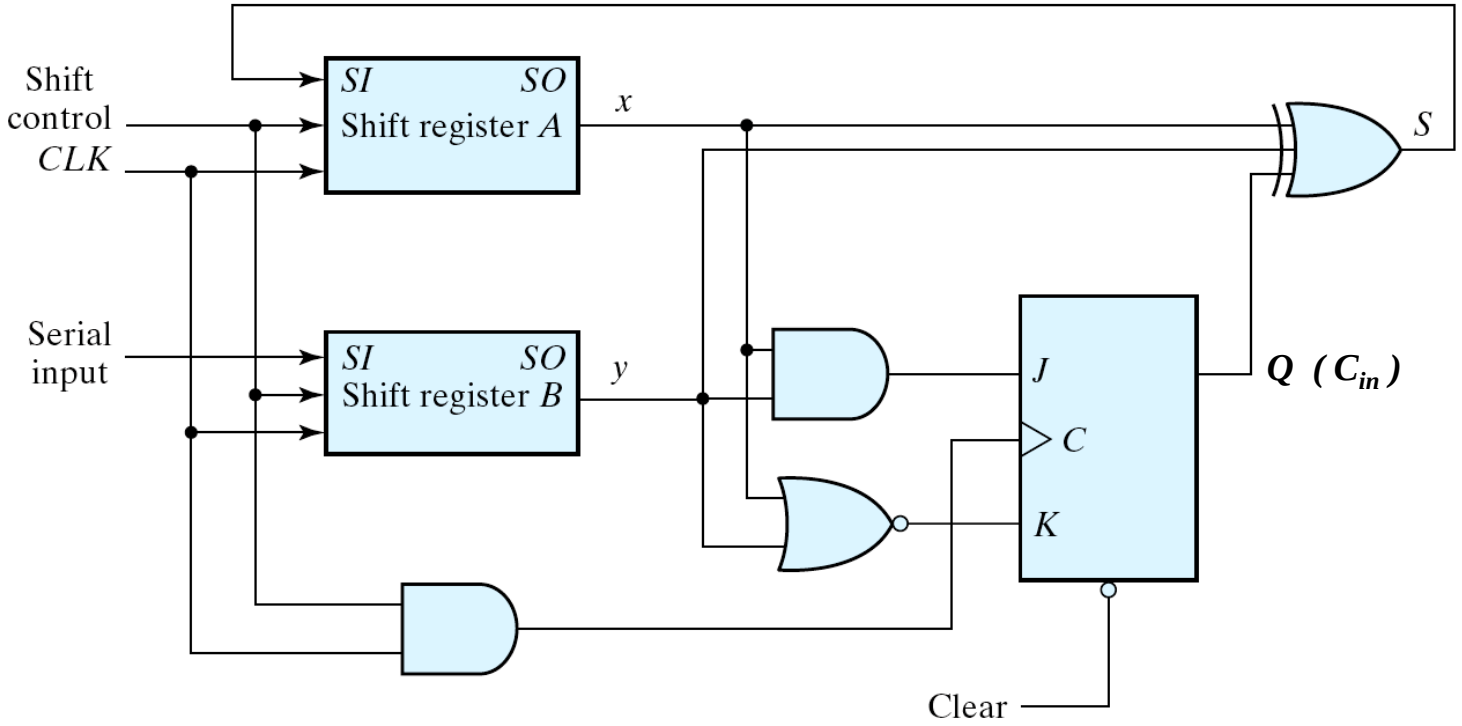


Fig. 6 Second form of serial adder

Universal Shift Register (1/5)

- **Three types of shift registers**
  - **Unidirectional shift register**
    - **A register capable of shifting in one direction**
  - **Bidirectional shift register**
    - **A register can shift in both directions**
  - **Universal shift register**
    - **Support both direction shifts & parallel load/out**

Universal Shift Register (2/5)

- **Capability of a universal shift register:**
  1. **A clear control to clear the register to 0**
  2. **A clock input to synchronize the operations**
  3. **A shift-right control to enable the shift right operation and the serial input and output lines associated w/ the shift right**
  4. **A shift-left control to enable the shift left operation and the serial input and output lines associated w/ the shift left**
  5. **A parallel-load control to enable a parallel transfer and the *n* parallel input lines associated w/ the parallel transfer**
  6. ***n* parallel output lines**
  7. **A control state that leaves the information in the register unchanged in the presence of the clock**

Universal Shift Register (3/5)

- **Example: 4-bit universal shift register**

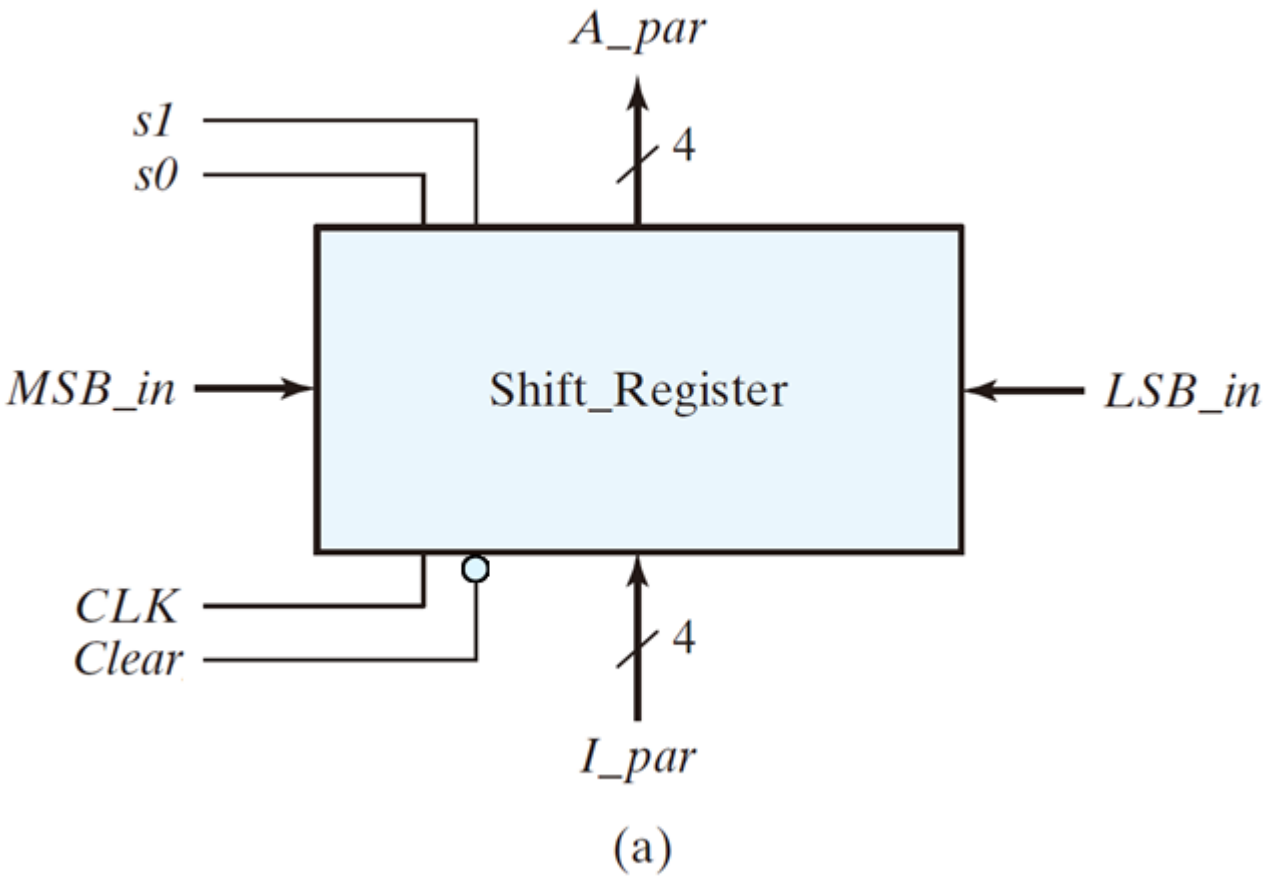
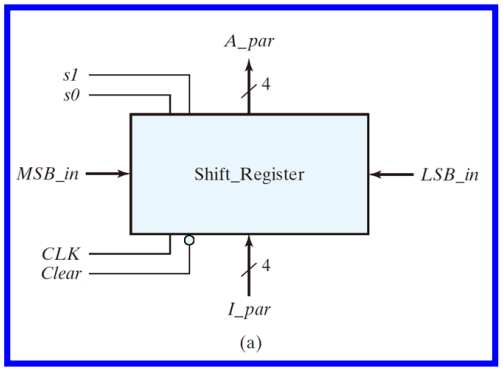


Fig. 7 Four-bit universal shift register

Universal Shift Register (4/5)



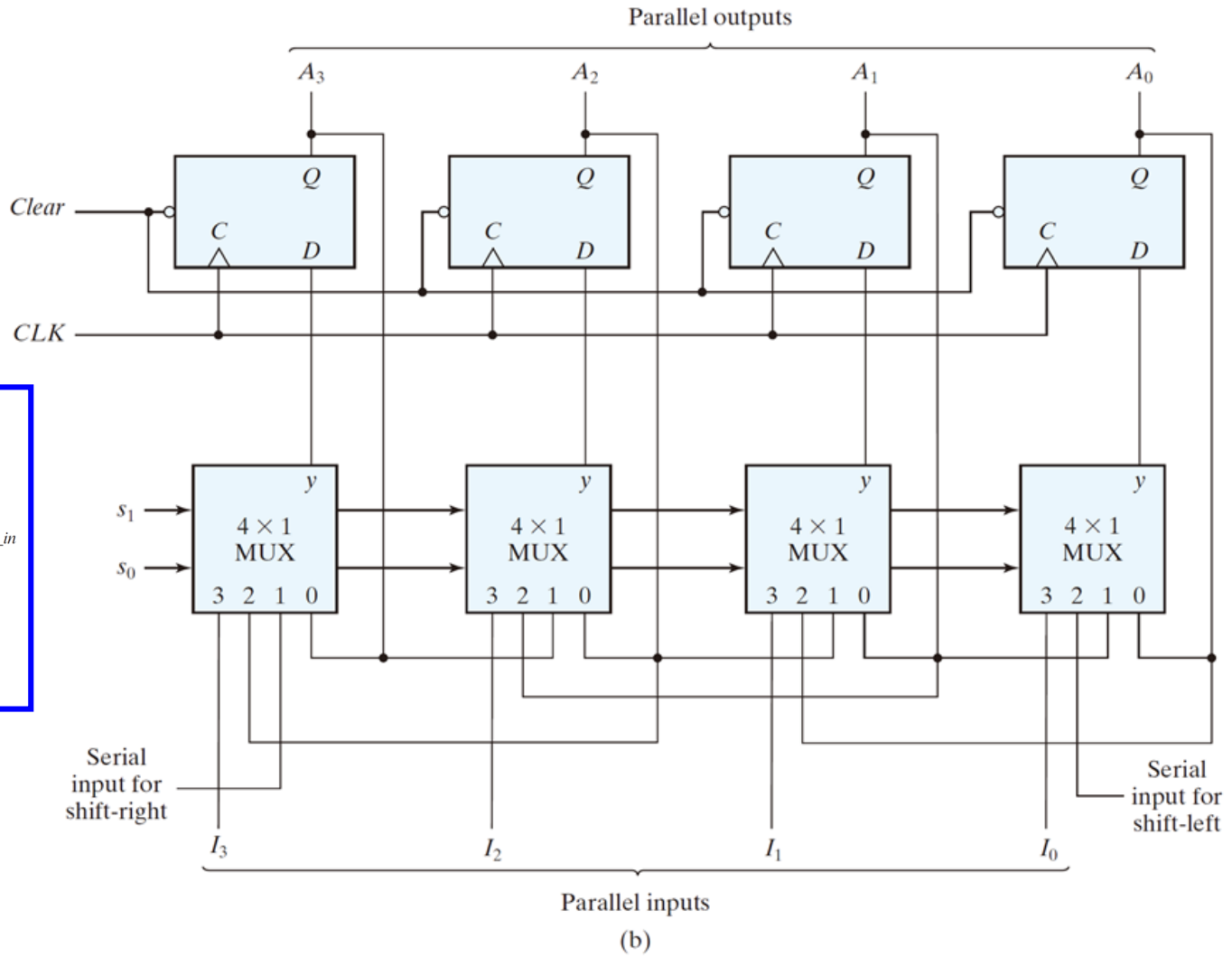


Fig. 7 Four-bit universal shift register

Universal Shift Register (5/5)

Function Table

Table 6.3
Function Table for the Register of Fig. 6.7

Mode Control		
s_1	s_0	Register Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

Clear	S1	S0	A3+	A2+	A1+	A0+	(operation)
0	×	×	0	0	0	0	Clear
1	0	0	A3	A2	A1	A0	No change
1	0	1	sri	A3	A2	A1	Shift right
1	1	0	A2	A1	A0	sli	Shift left
1	1	1	I3	I2	I1	I0	Parallel load

Ripple Counters

- **Counter**
  - **A register that goes through a prescribed sequence of states**
  - **Upon the application of input pulses**
    - **Input pulses: may be clock pulses or originate from some external source**
    - **The sequence of states: may follow the binary number sequence (⇒ binary counter) or any other sequence of states**
    - **A *n*-bit binary counter → *n* FFs → count from 0 to 2^n-1**

Counters

Categories of counters

1. **Ripple counters**
   - The flip-flop output transition serves as a source to trigger other flip-flops
   - No common clock pulse (not synchronous)
2. **Synchronous counters**
   - The CLK inputs of all flip-flops receive a common clock

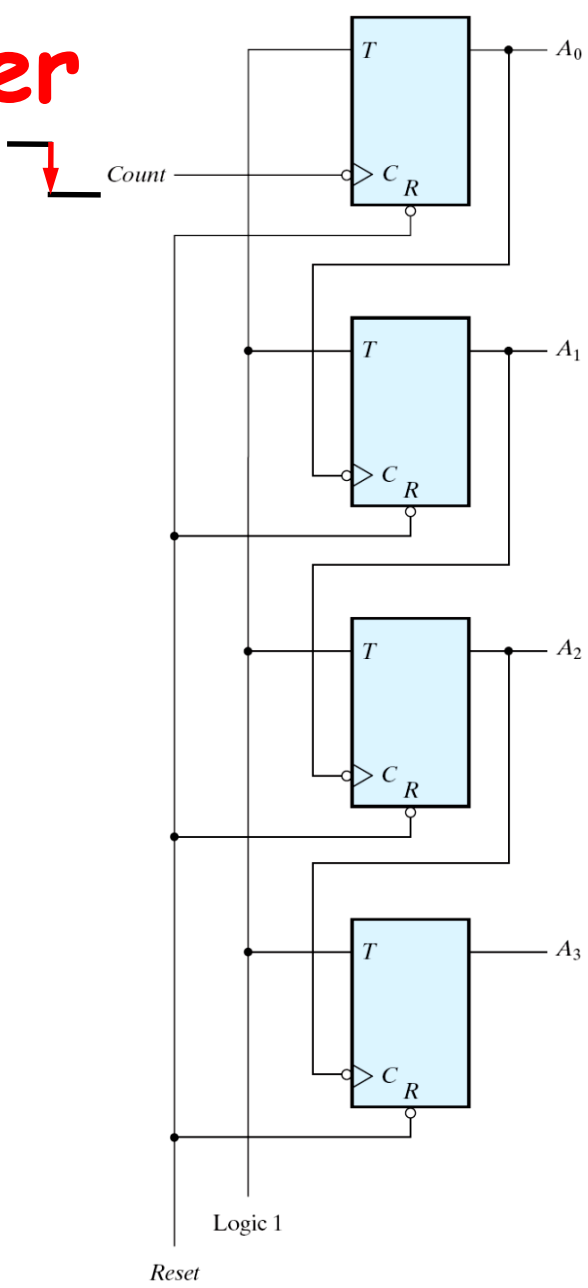
4-bit Binary Count Sequence

- **Binary count sequence: 4-bit**

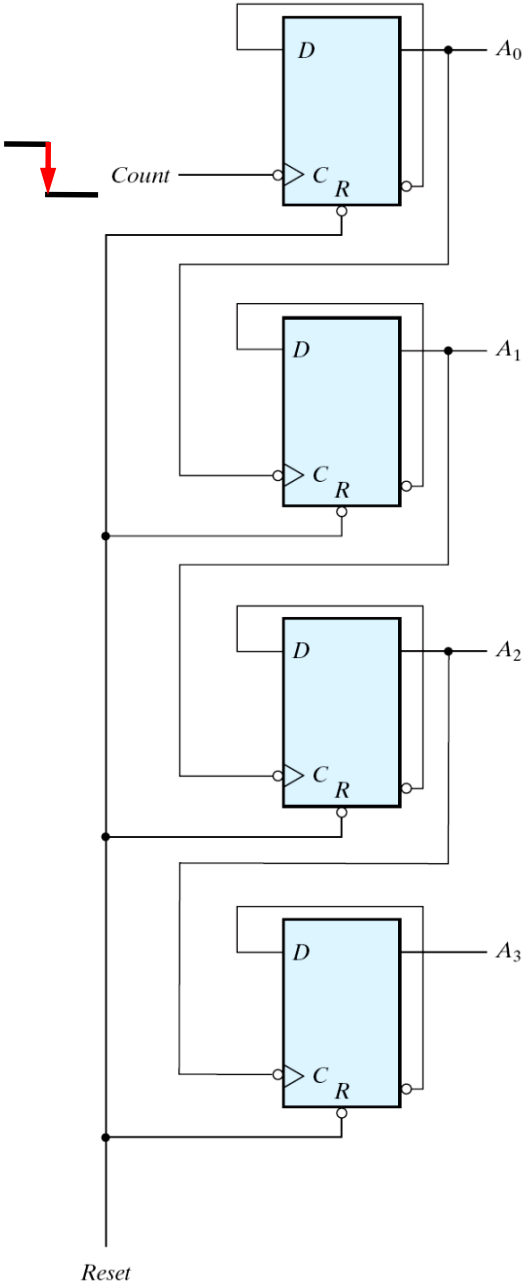
A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Ripple Counter

A_3	A_2	A_1	A_0
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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1



(a) With *T* flip-flops



(b) With *D* flip-flops

Fig. 8 Four-bit binary ripple counter

BCD Ripple Counter

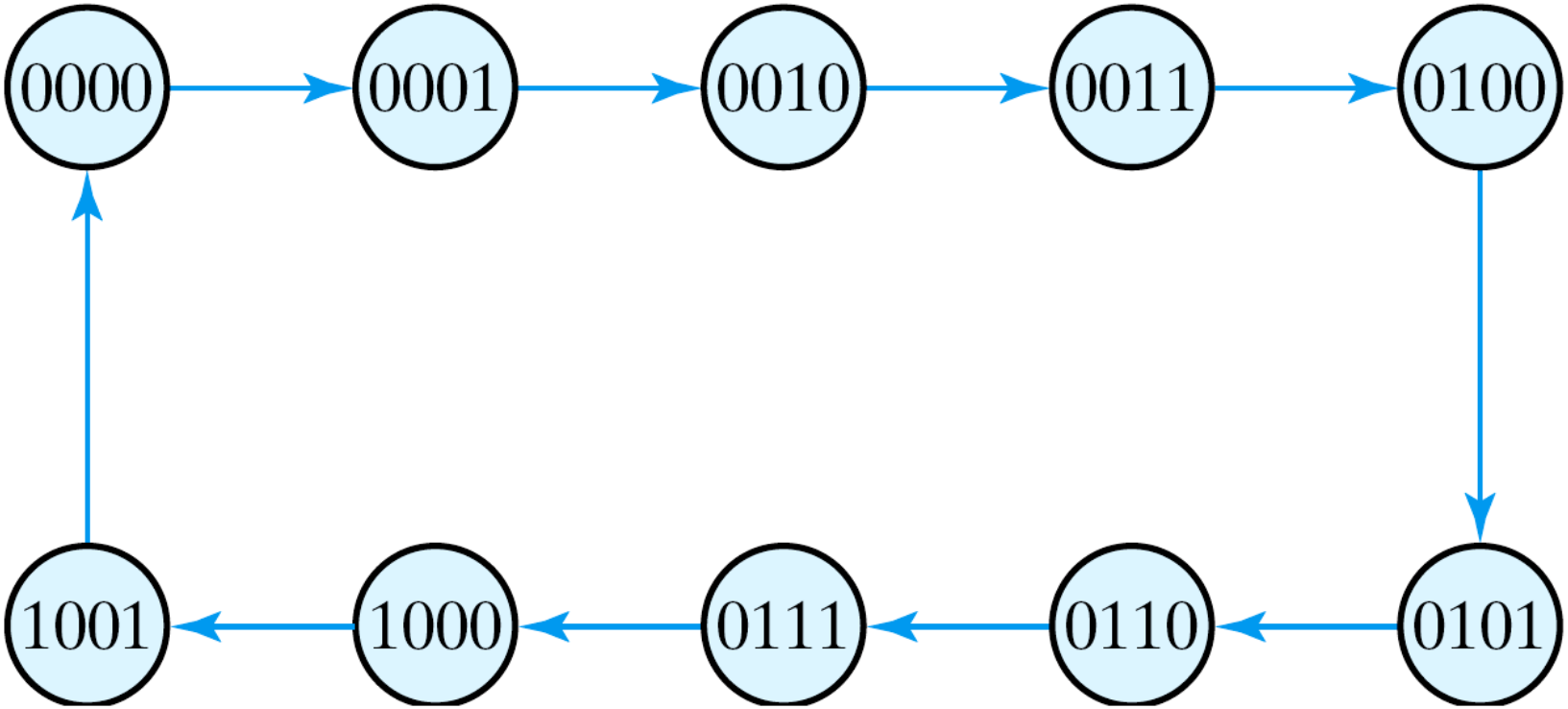
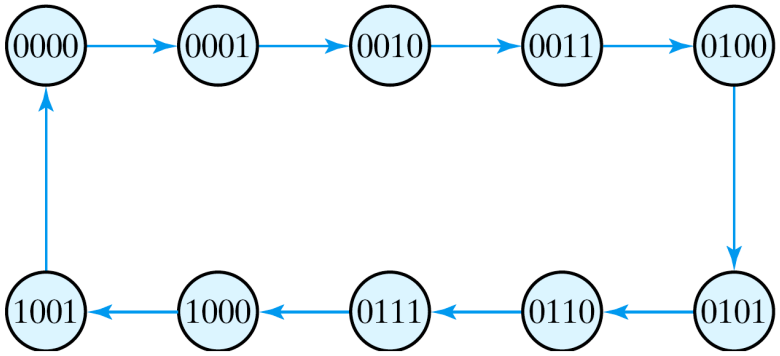


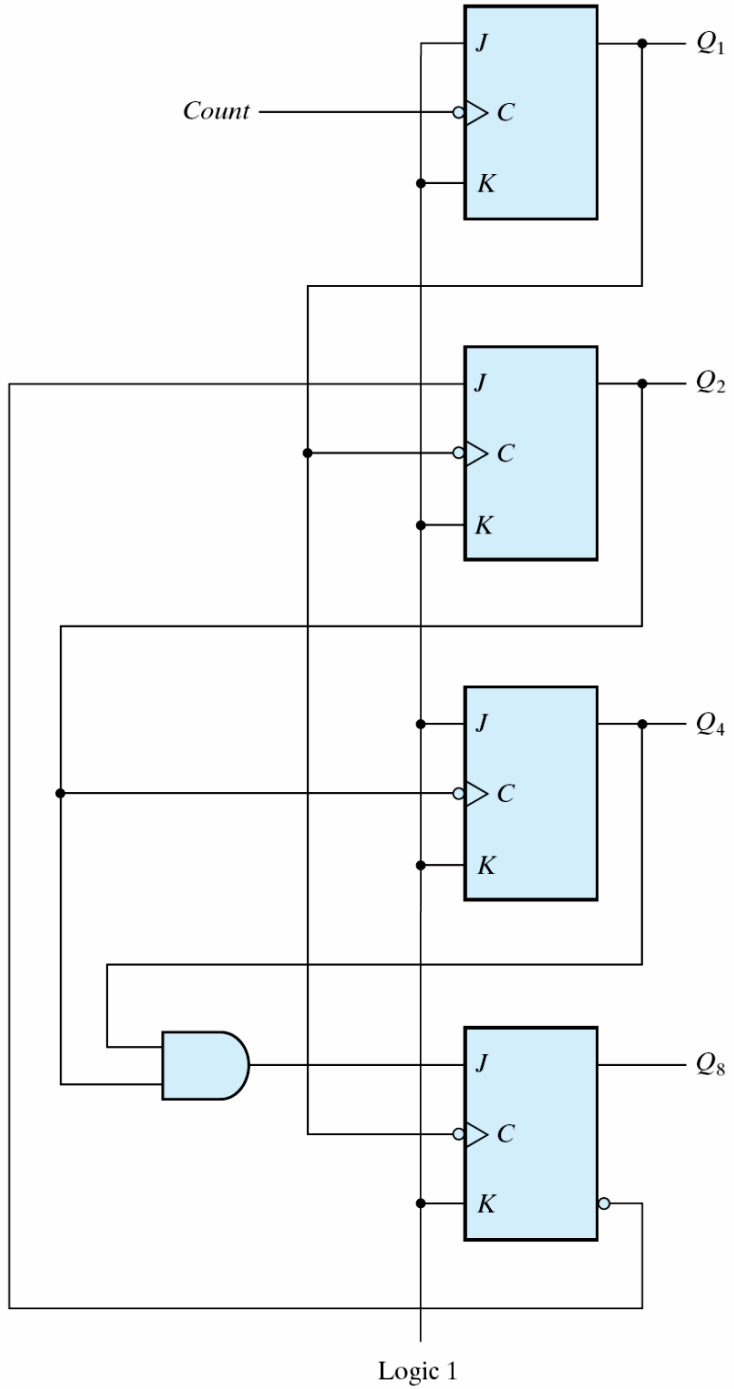
Fig. 9 State diagram of a decimal BCD counter

BCD Ripple Counter



Q_8	Q_4	Q_2	Q_1
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
1	0	0	1

Fig. 10 BCD ripple counter



Decade Counter

- **Three-decade BCD counter**

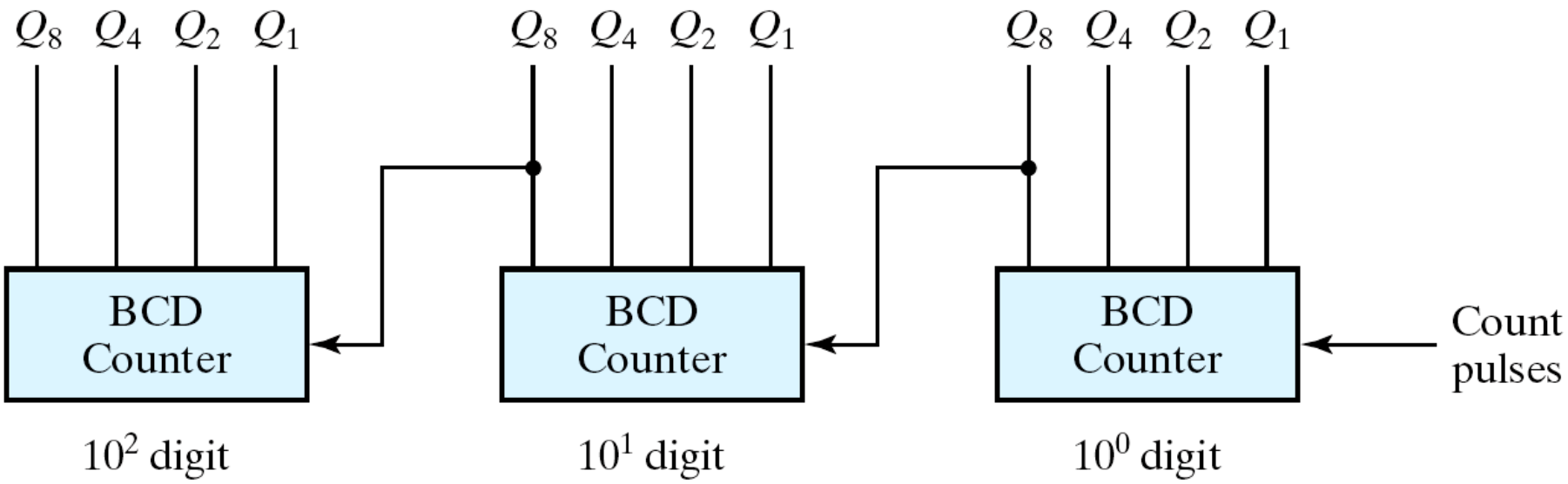
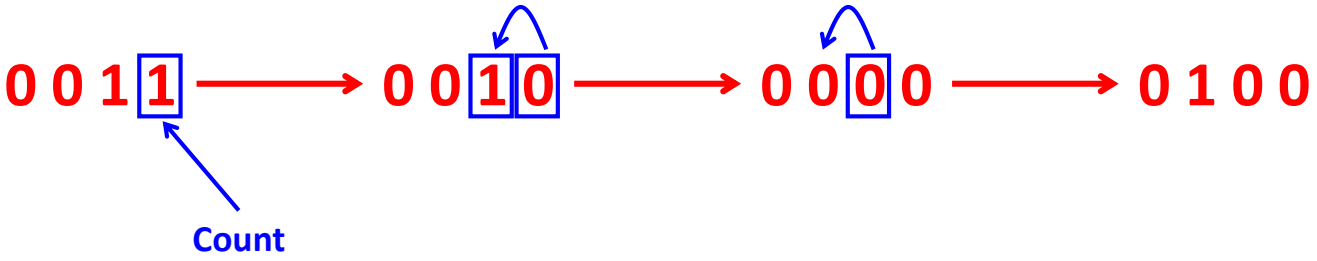


Fig. 11 Block diagram of a three-decade decimal BCD counter

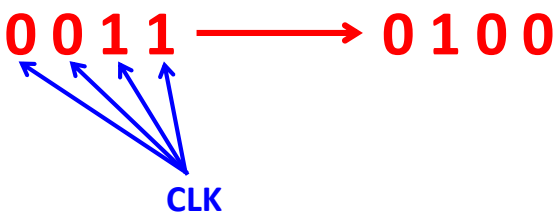
Synchronous Counters (1/2)

Review of counters

1. Ripple counters
   - The flip-flop output transition serves as a source to trigger other flip-flop
   - ⇒ No common clock pulse (not synchronous)



2. Synchronous counters
   - The CLK inputs of all flip-flops receive a common clock



Synchronous Counters (2/2)

- **Synchronous counter**
  - A common clock triggers all flip-flops simultaneously
- **Design procedure**
  - Apply the same procedure of sync. sequential circuits (Chap. 5)
  - Sync. counter is simpler than general sync. sequential circuits
- ***T* and *JK* FFs**
  - *T*=0 or *J*=*K*=0: no change
  - *T*=1 or *J*=*K*=1: complement

Sync. Counters using JK FFs

4-bit binary counter

A_3	A_2	A_1	A_0
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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

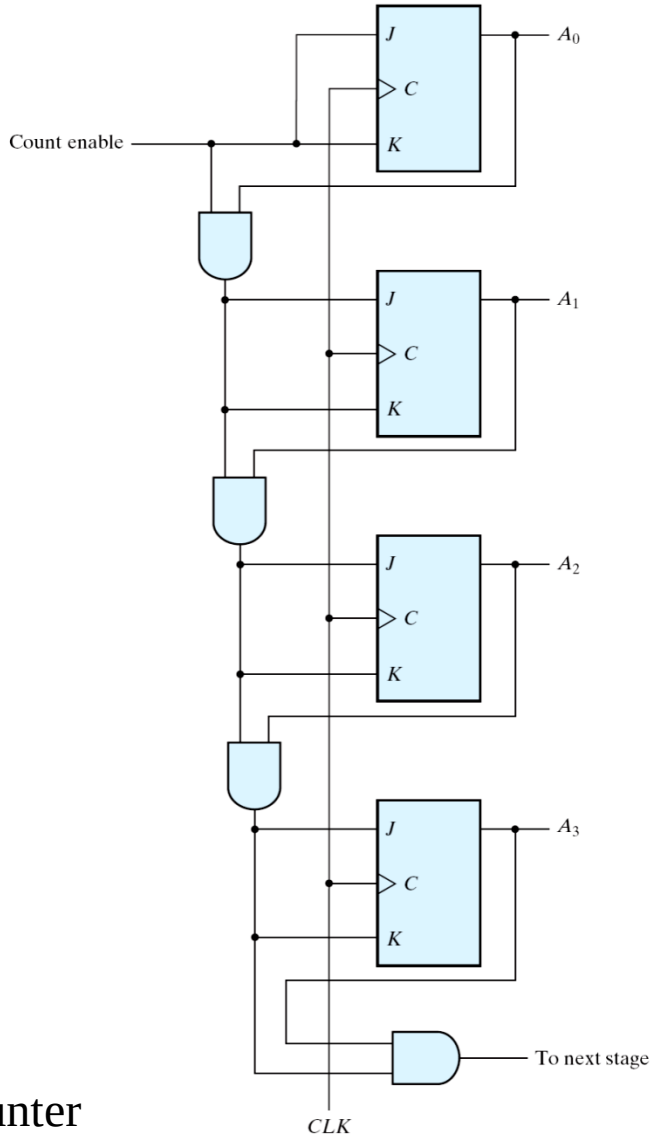


Fig. 12 Four-bit synchronous binary counter

4-bit Up/Down Binary Counter

Up	Down	Function
0	0	No change
0	1	Down Count
1	0	Up Count
1	1	Up Count

A_3	A_2	A_1	A_0
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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

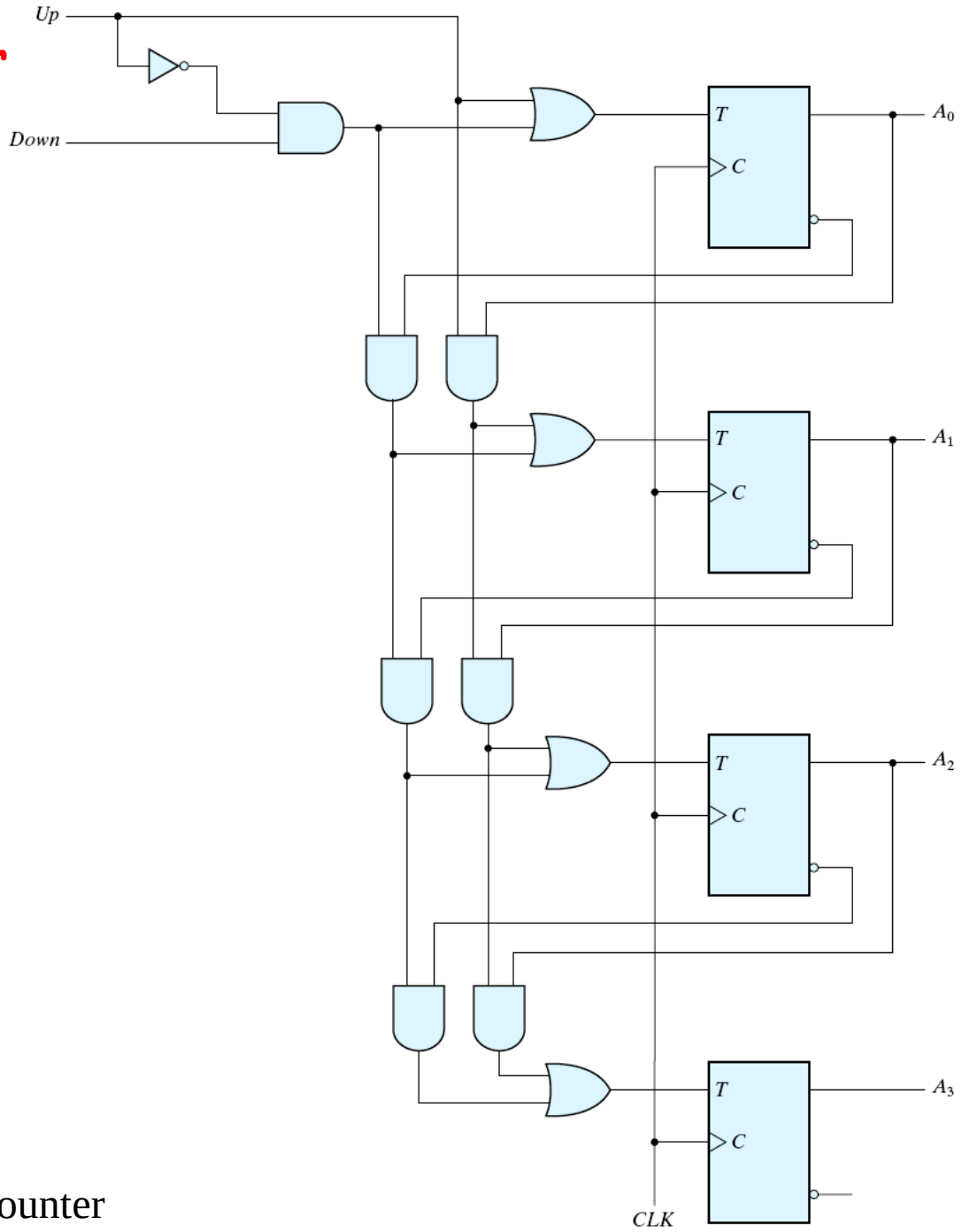


Fig. 13 Four-bit up-down binary counter

Sync. BCD Counters

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

- **Simplified functions**

$$T_{Q1} = 1$$
$$T_{Q2} = Q'_8 Q_1$$

$$T_{Q4} = Q_2 Q_1$$
$$T_{Q8} = Q_8 Q_1 + Q_4 Q_2 Q_1$$
$$y = Q_8 Q_1$$

Binary Counter with Parallel Load

- **4-bit binary counter with parallel load**

Table 6.6

Function Table for the Counter of Fig. 6.14

Clear	CLK	Load	Count	Function
0	X	X	X	Clear to 0
1	↑	1	X	Load inputs
1	↑	0	1	Count next binary state
1	↑	0	0	No change

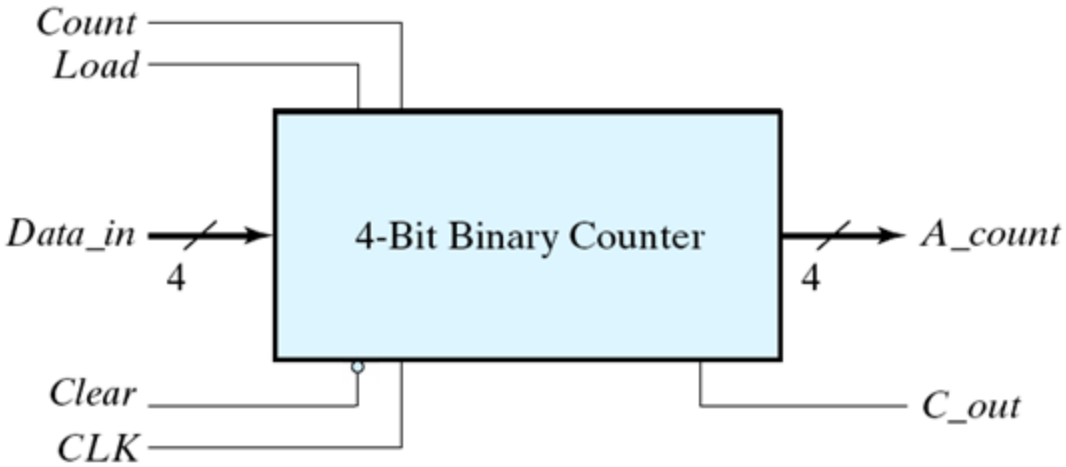


Fig. 14 Four-bit binary counter with parallel load

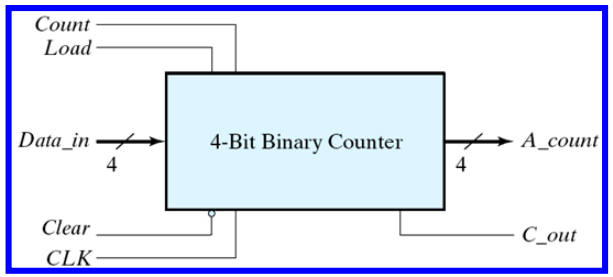


Fig. 14 Four-bit binary counter with parallel load

Extensions of Parallel Load Counter

- **Other BCD counter implementations**

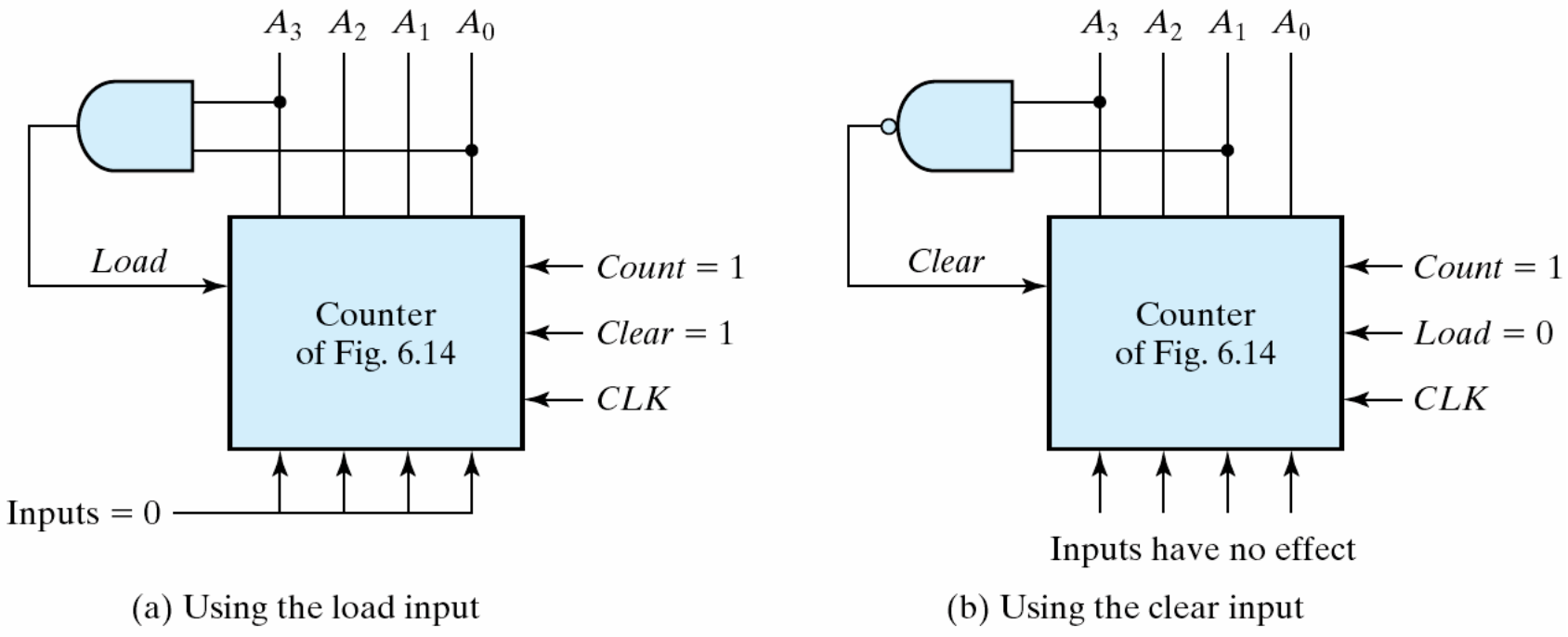


Fig. 15 Two ways to achieve a BCD counter using a counter with parallel load

Other Counters

- **Counters**
  - **Can be designed to generate any desired sequence of states**
- **Divide-by-*N* counter (modulo-N counter)**
  - **A counter that goes through a repeated sequence of *N* states**
  - **The sequence may follow the binary count or may be any other arbitrary sequence**

Counter with Unused States

- ***n*** **flip-flops ⇒ 2^n binary states**
- **Unused states**
  - **States that are not used in specifying the FSM**
  - **May be treated as don't-care conditions or may be assigned specific next states**
- **Self-correcting counter**
  - **Ensure that when a circuit enter one of its unused states, it eventually goes into one of the valid states after one or more clock pulses so it can resume normal operation**

    ⇒ Analyze the circuit to determine the next state from an unused state after it is designed

Self-Correcting Counter (1/2)

- An example

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

- Two unused states: 011 & 111
- The simplified flip-flop input equations:
  - $J_A = B$, $K_A = B$
  - $J_B = C$, $K_B = 1$
  - $J_C = B'$, $K_C = 1$

Self-Correcting Counter (2/2)

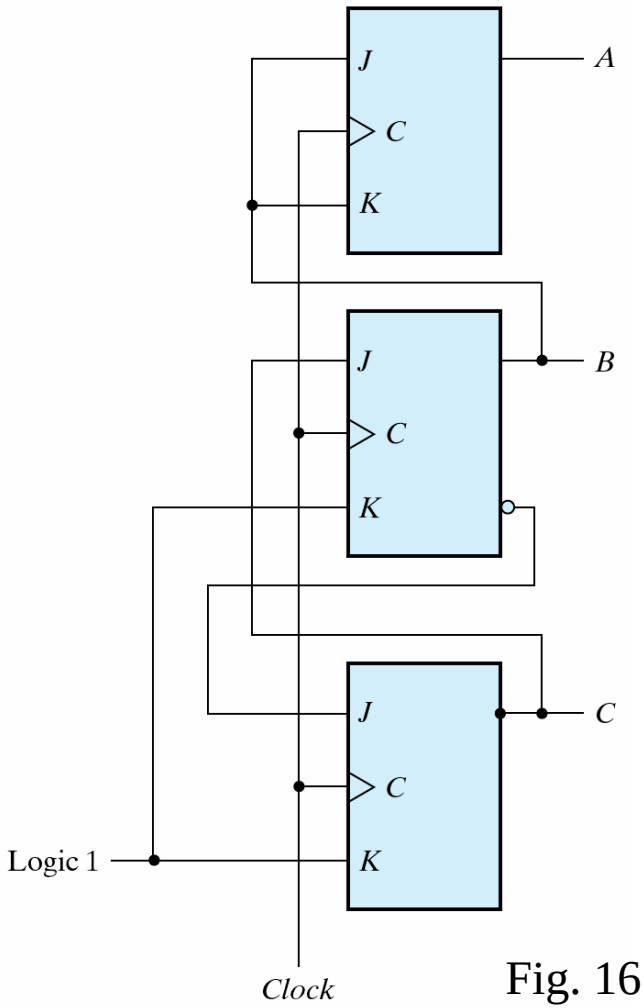
- **The logic diagram & state diagram of the circuit**

The simplified flip-flop input equations:

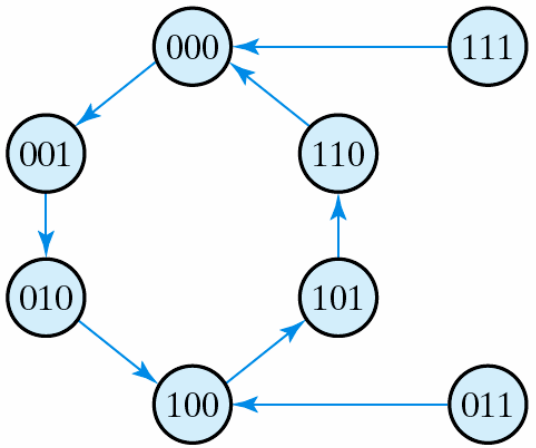
$J_A = B,\ K_A = B$

$J_B = C,\ K_B = 1$

$J_C = B',\ K_C = 1$



(a) Logic diagram



(b) State diagram

Fig. 16 Counter with unused states

Ring Counter (1/4)

- **Ring counter**
  - **A circular shift register with only one flip-flop being set at any particular time, all others are cleared (initial value = 1 0 0 … 0)**
  - **The single bit is shifted from one flip-flop to the next to produce the sequence of timing signals**

Ring Counter (2/4)

- **A 4-bit ring counter**

T_0	T_1	T_2	T_3
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1
1	0	0	0

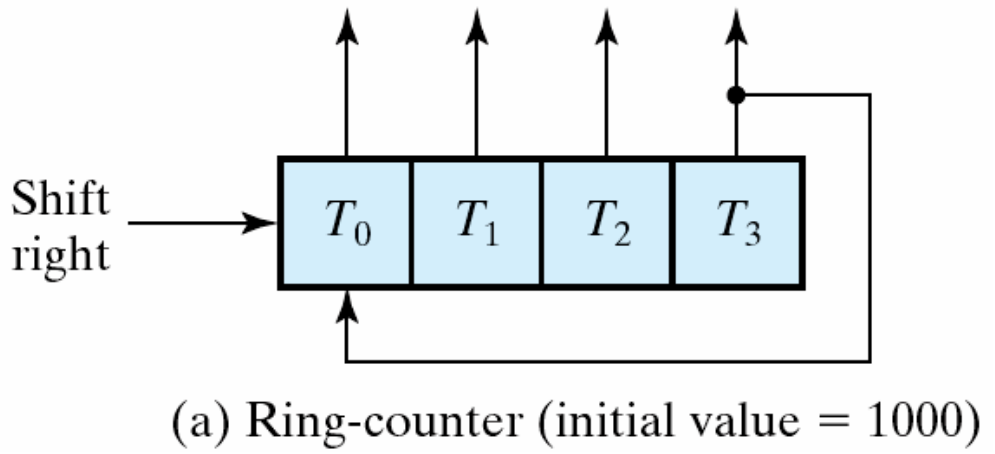


Fig. 17 Generation of timing signals

Ring Counter (3/4)

- **Application of counters**
  - **Counters may be used to generate timing signals to control the sequence of operations in a digital system**
- **Approaches for generation of 2^n timing signals**
  1. **A shift register with 2^n flip-flops**
  2. **An *n*-bit binary counter together with an *n*-to-2^n-line decoder**

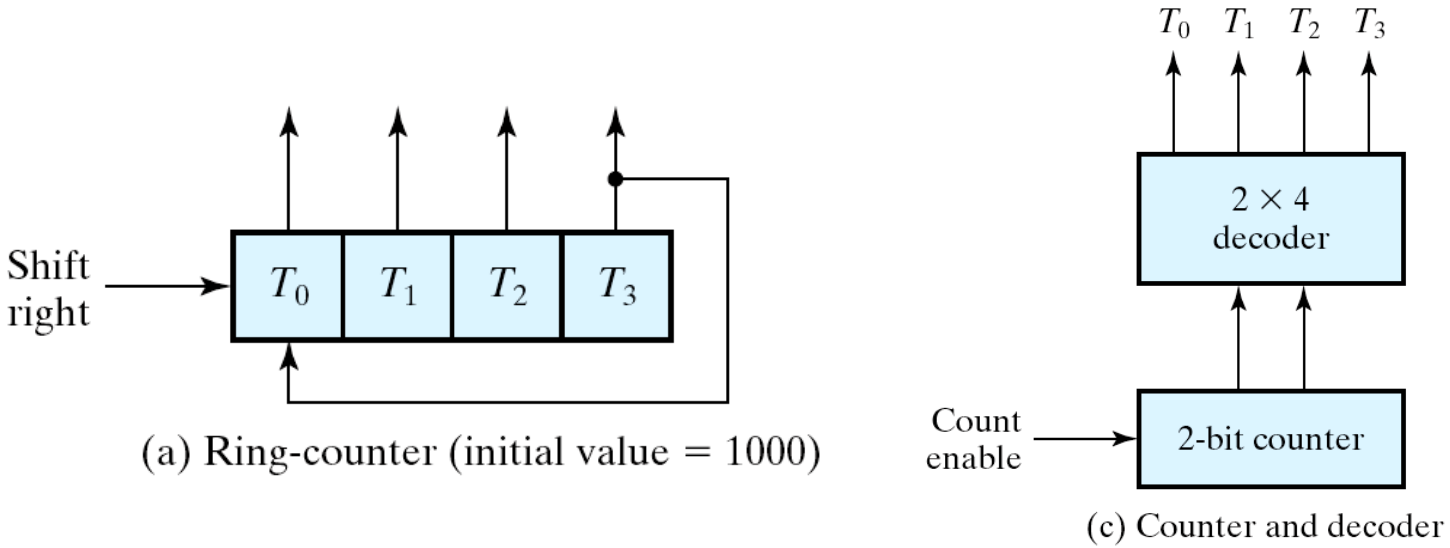
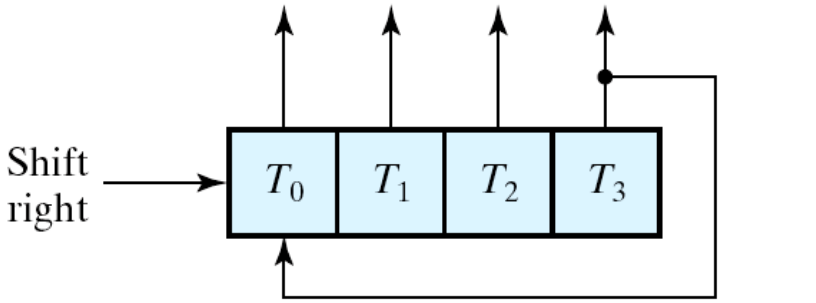
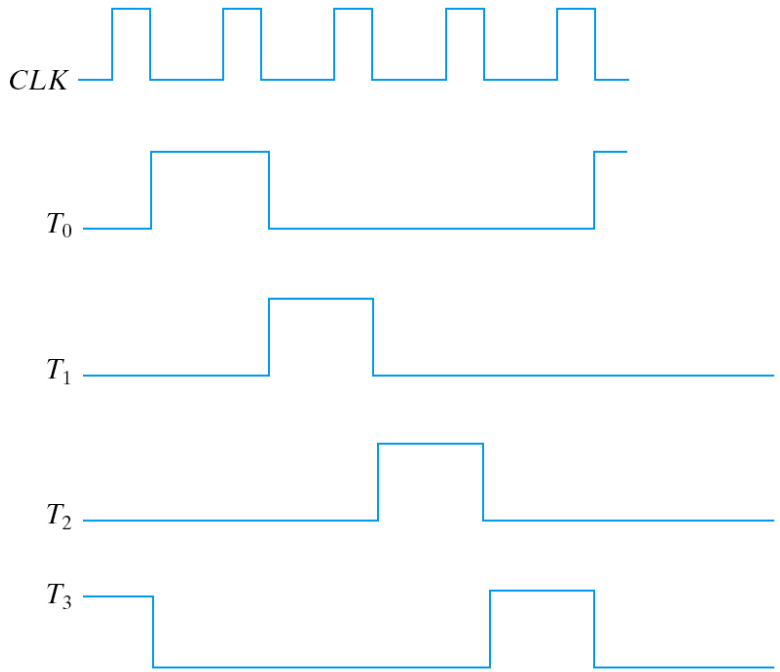


Fig. 17 Generation of timing signals

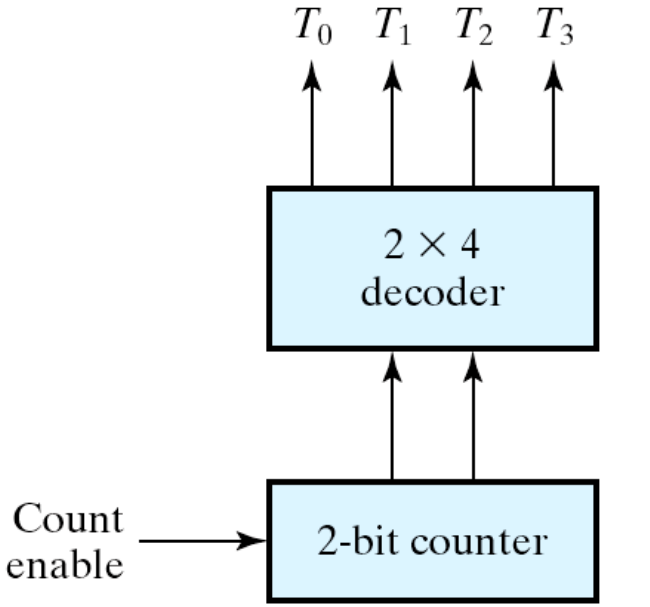
Ring Counter (4/4)



(a) Ring-counter (initial value = 1000)



(b) Sequence of four timing signals



(c) Counter and decoder

Fig. 17 Generation of timing signals

Johnson Counter (1/4)

Ring counter vs. Switch-tail ring counter

- **Ring counter**
  - A *k*-bit ring counter circulates a single bit among the flip-flops to provide *k* distinguishable states (initial value = 1 0 … 0)

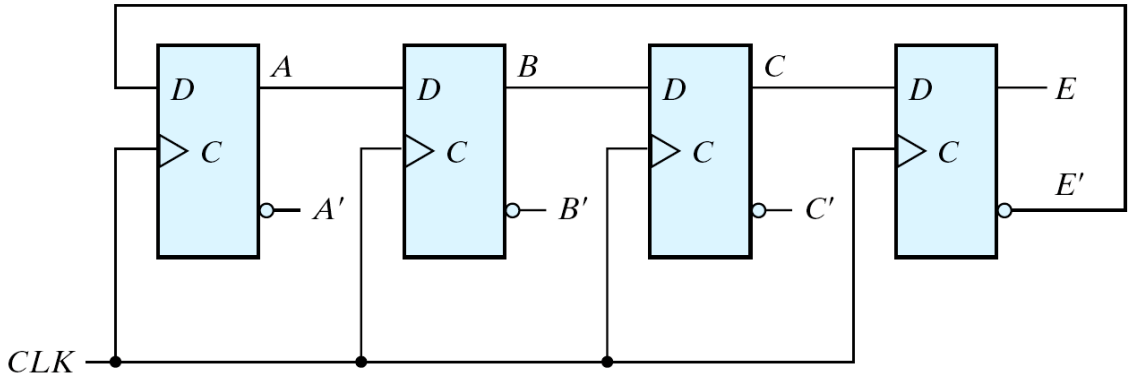
Straight ring/*Overbeck counter*				
State	Q0	Q1	Q2	Q3
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
0	1	0	0	0

- **Switch-tail ring counter**
  - It is a circular shift register with the complement output of the last flip-flop connected to the input of the first flip-flop
  - A *k*-bit switch-tail ring counter will go through a sequence of 2*k* distinguishable states (initial value = 0 0 … 0)

Twisted ring/*Johnson counter*				
State	Q0	Q1	Q2	Q3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
0	0	0	0	0

Johnson Counter (2/4)

An example: Switch-tail ring counter



(a) Four-stage switch-tail ring counter

Sequence number	Flip-flop outputs				AND gate required for output
	A	B	C	E	
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. 18 Construction of a Johnson counter

Johnson Counter (3/4)

- **Johnson counter**
  - **A *k*-bit switch-tail ring counter + 2*k* decoding gates**
  - **Provide outputs for 2*k* timing signals**
    - **E.g.: 4-bit Johnson counter**

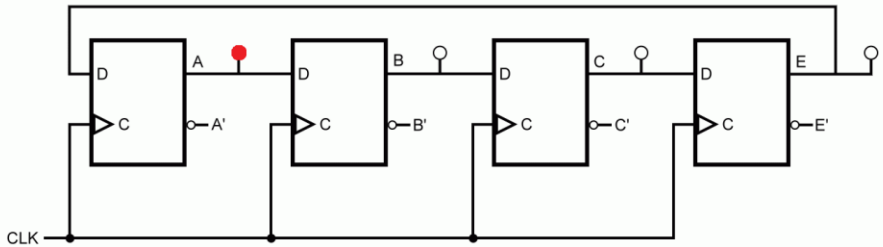
Sequence number	Flip-flop outputs				AND gate required for output
	A	B	C	E	
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

(b) Count sequence and required decoding

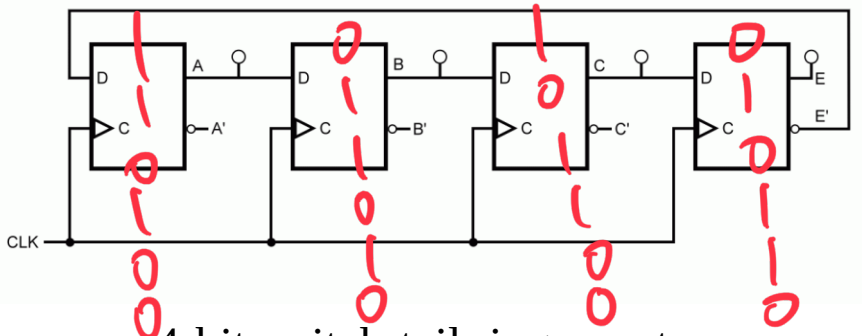
  - **The decoding follows a regular pattern**
    - **2 inputs per decoding gate**

Johnson Counter (4/4)

- **Disadvantage of the switch-tail ring counter**
  - If it finds itself in an unused state, it will persist to circulate in the invalid states and never find its way to a valid state
  - One correcting procedure: $D_C = (A + C)\,B$
- **Summary**
  - Johnson counters can be constructed for any number of timing sequences
    - » Number of flip-flops = 1/2 (the number of timing signals)
    - » Number of decoding gates = number of timing signals (2-input per gate)



4-bit ring counter



4-bit switch-tail ring counter