ITI1100 Section Z
Digital Systems I

Chapter 6:

Registers and Counters (1)

Prof. Mohammad Alja'Afreh

Summer 2019

Overview

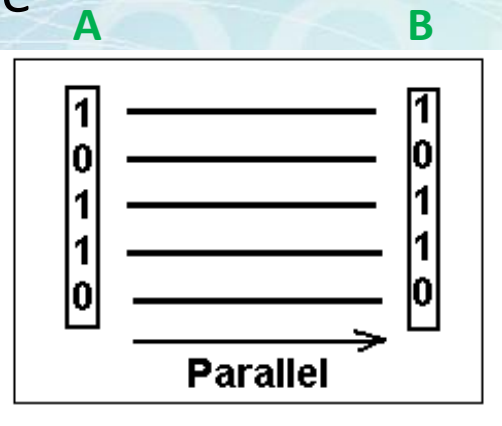
- **Registers** and **counters** are **sequential circuits**
- **Registers** and **counters** are the **basic building blocks** from which **more complex digital systems** are **constructed**

Registers

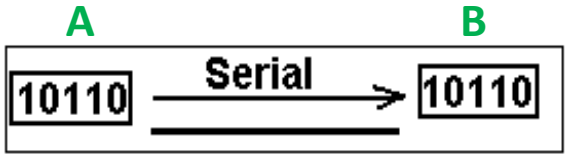
- **Multiple flip-flops sharing** the same **clock** can be combined to form a **data register**
- **Registers** allow for **parallel transfer** of data
  - Many bits transferred at the same time
- **Shift registers** also allow **data** to be **transported one bit at a time**

Parallel versus Serial

- **Parallel communication**
  - Provides binary numbers through multiple data lines at the same time

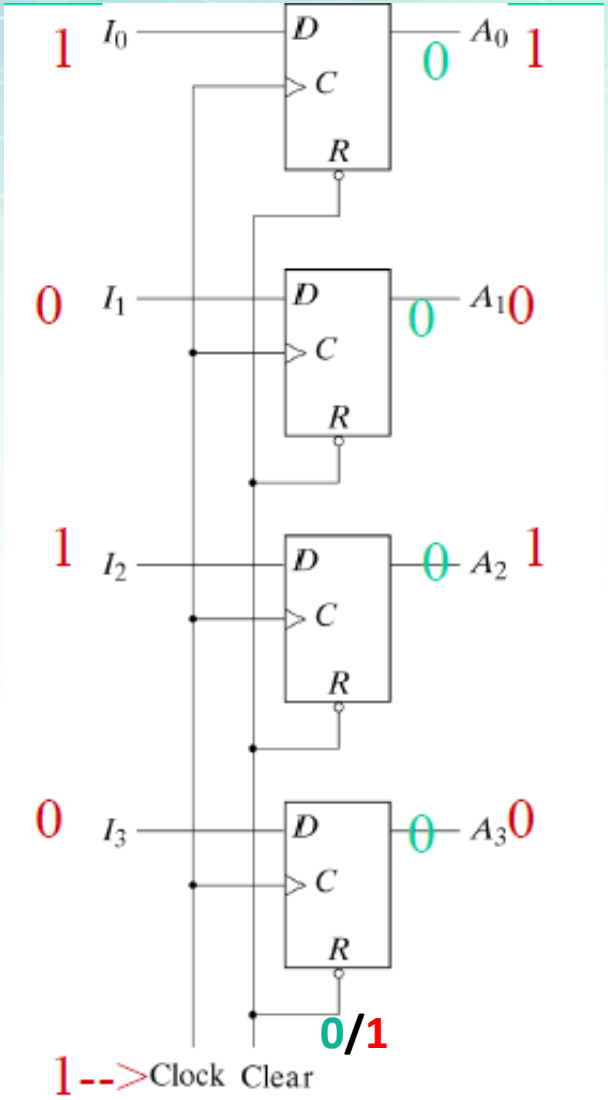


- **Serial communication**
  - Provides binary numbers as a sequence of binary digits, one after another, through one data line

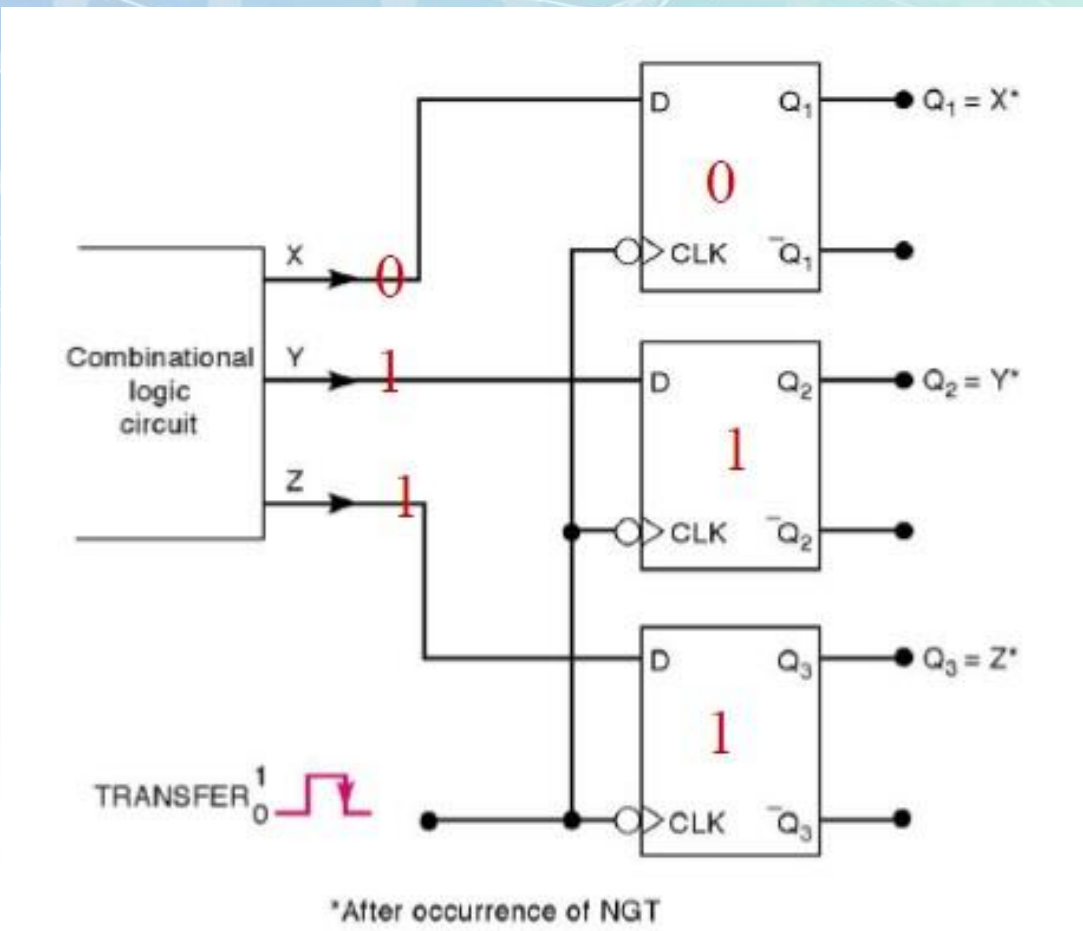


Register with Parallel Load

- **Register: Group** of **flip-flops** (D flip-flops in this example) **sharing** the **same clock**
- Holds 4 bits of Data
- **Loads in Parallel on Clock Transition**
- **Asynchronous Clear (Reset)**



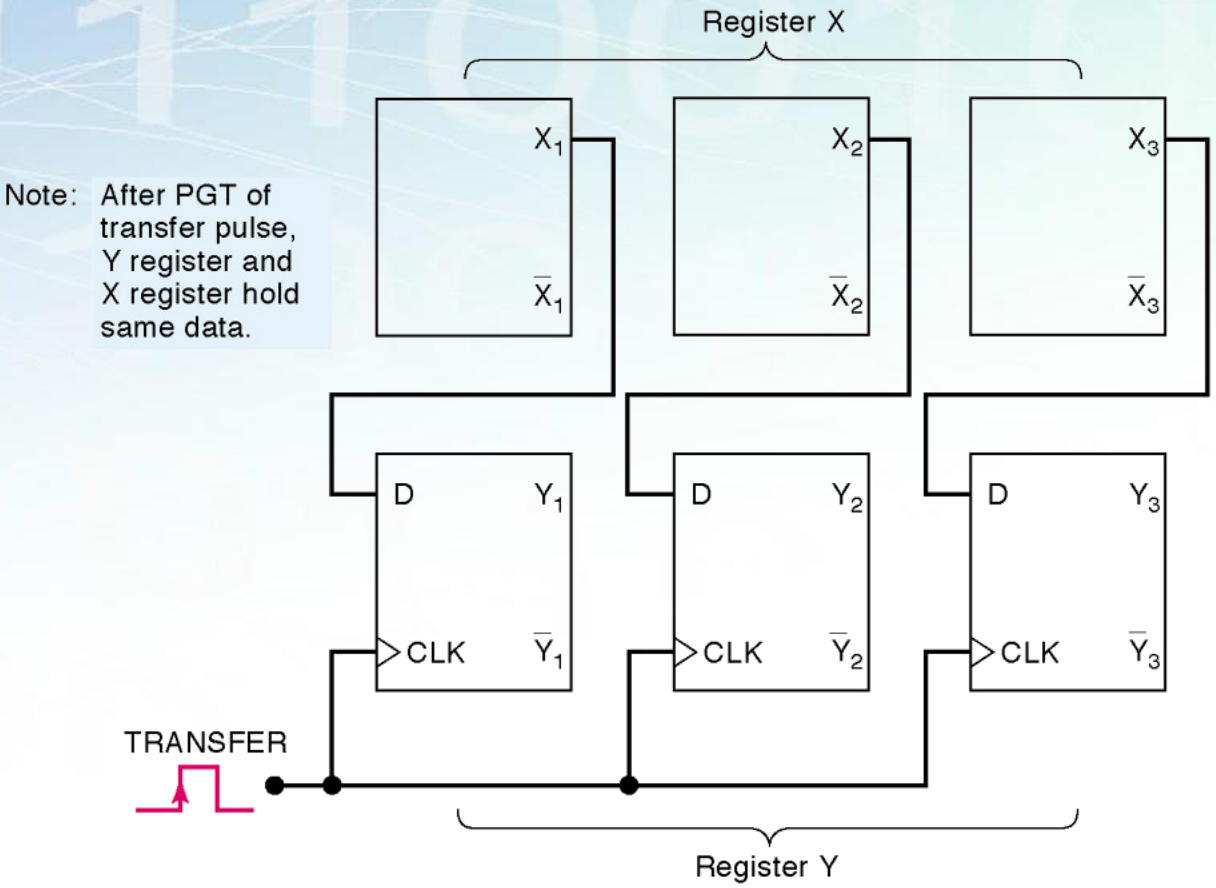
Register with Parallel Load (cont'd)



*After occurrence of NGT

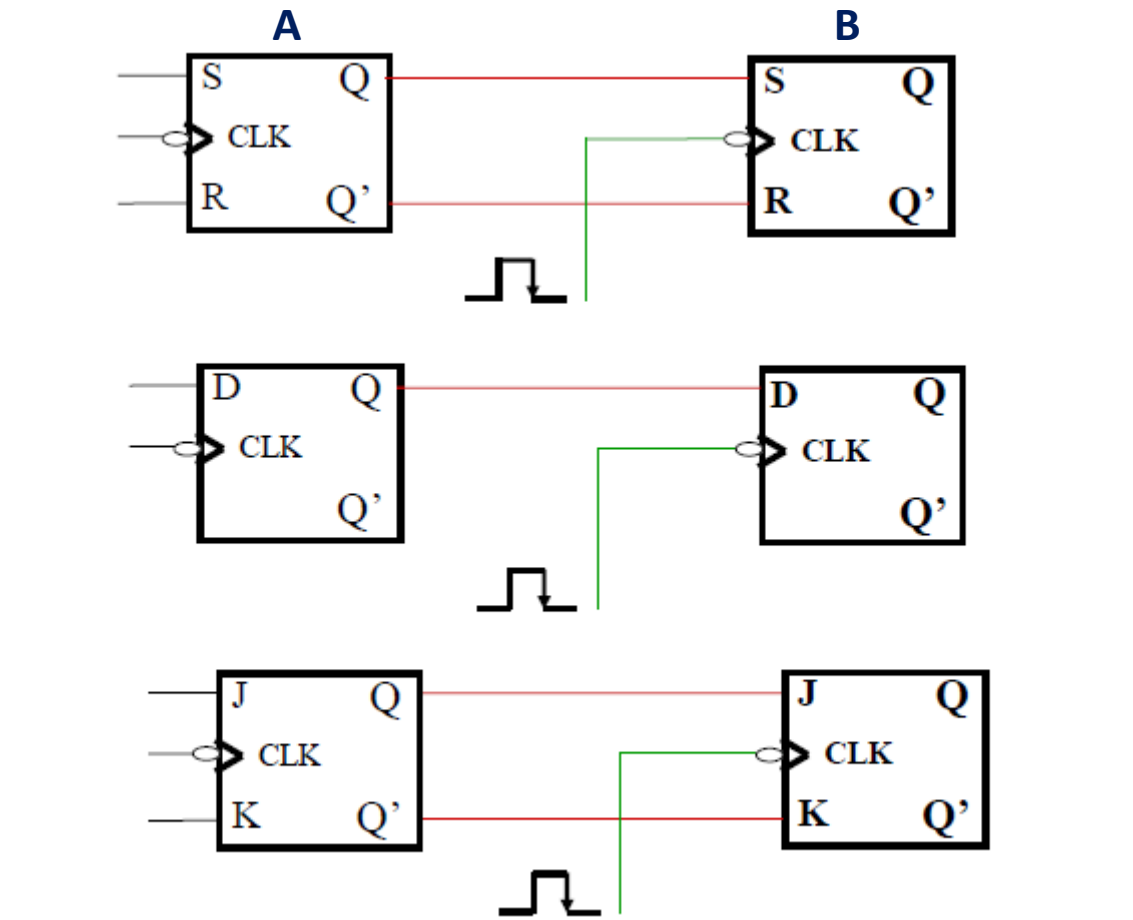
Parallel Data Transfer

- **All data transfers** on **rising clock edge** (in this example)
- **Data clocked** into **register Y**



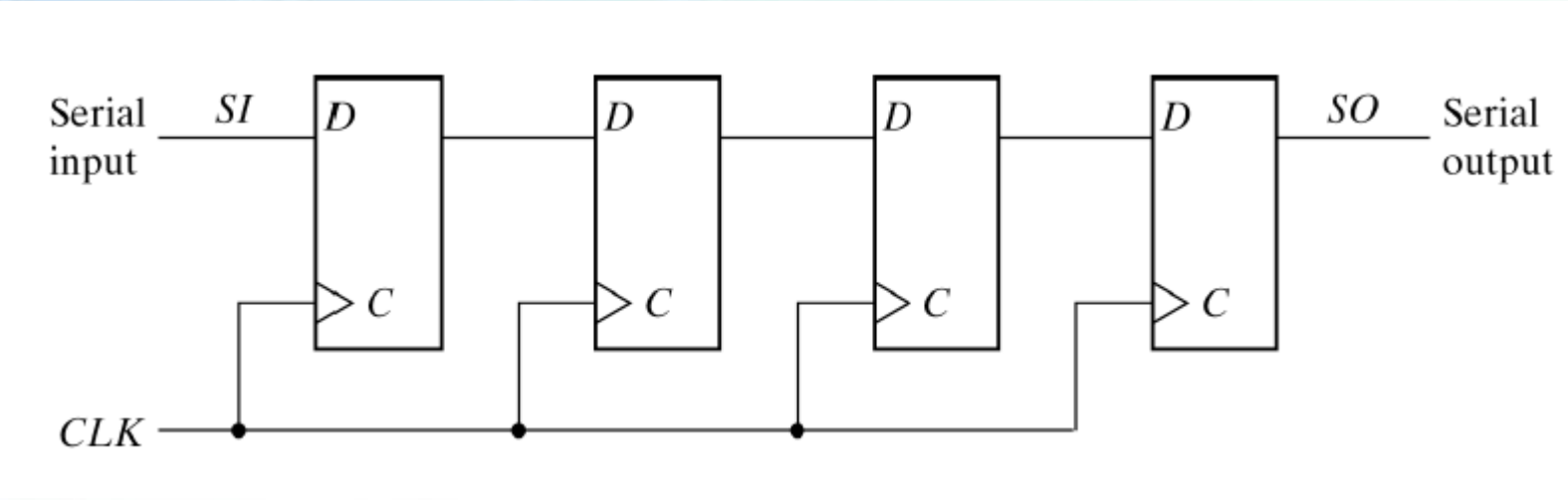
Shift Registers

- Transfer of data from **flip-flop A** to **flip-flop B**



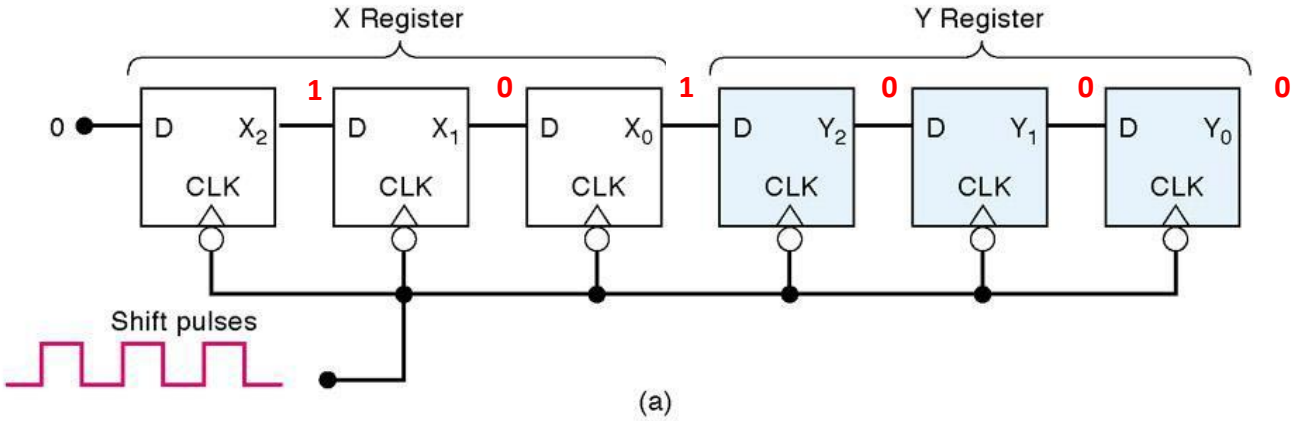
Shift Registers (cont'd)

- **Shift register** is a **cascade chain** of **flip-flops** that share the **same clock**
- **Bits travel** on **positive edges** (in this example)
- **Serial in** (SI) – **Serial out** (SO)



Serial Transfer of Data

- **Transfer** from **register X** to **register Y one bit at a time** (negative clock edges for this example)



(a)

X_2	X_1	X_0	Y_2	Y_1	Y_0	
1	0	1	0	0	0	← Before pulses applied
0	1	0	1	0	0	← After first pulse
0	0	1	0	1	0	← After second pulse
0	0	0	1	0	1	← After third pulse

(b)

Serial Transfer of Data (cont'd)

- **Transfer** from **register A** to **register B**
- **Data loopback** for **register A**

Time	Reg A	Reg B
T0	1011	0011
T1	1101	1001
T2	1110	1100
T3	0111	0110
T4	1011	1011

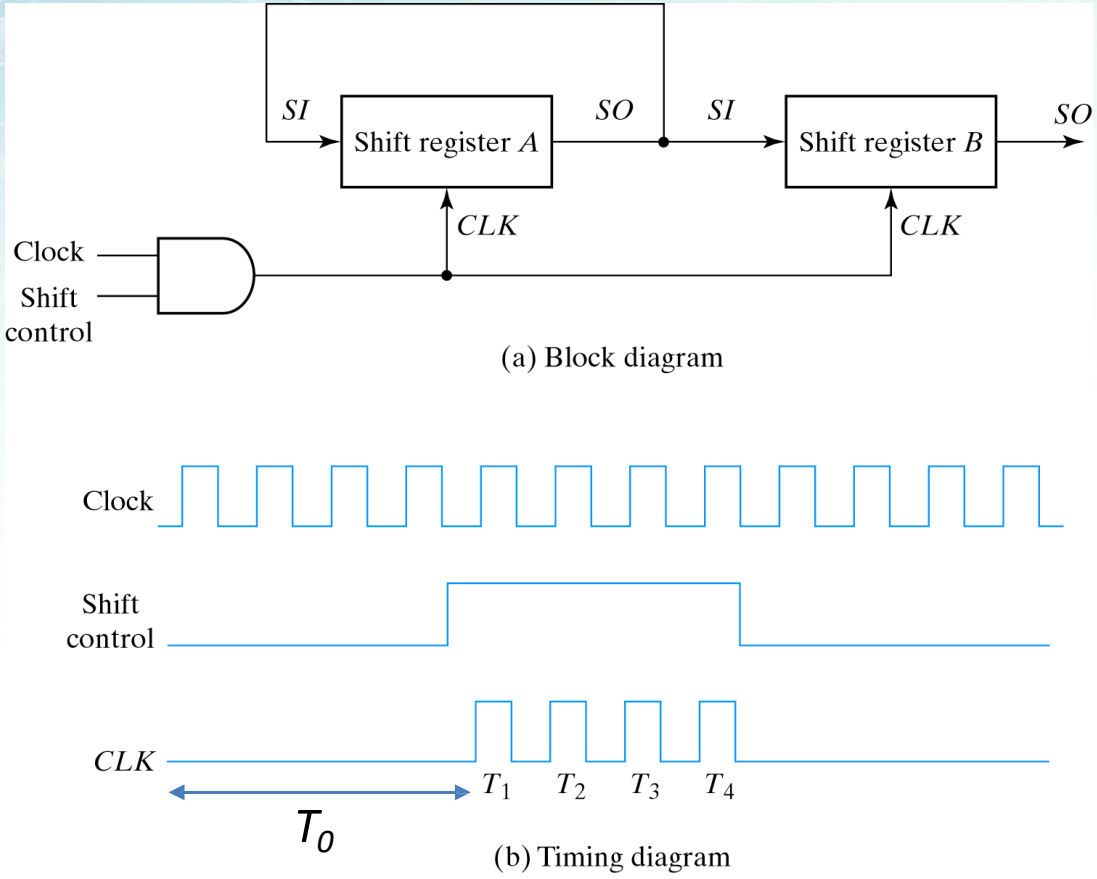
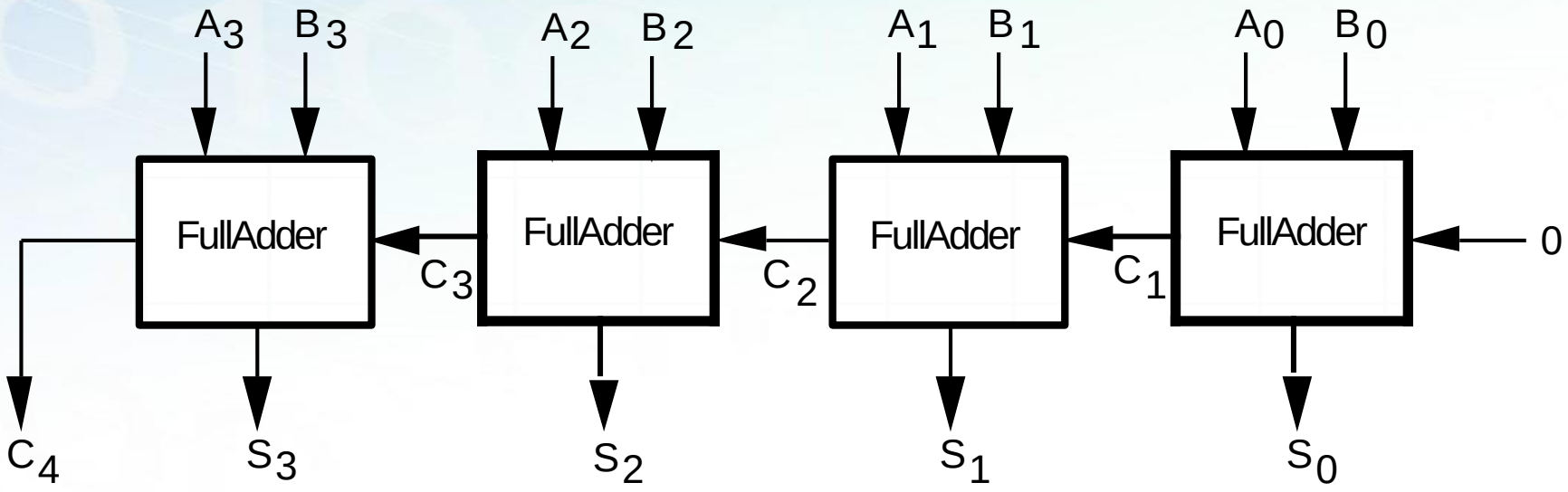


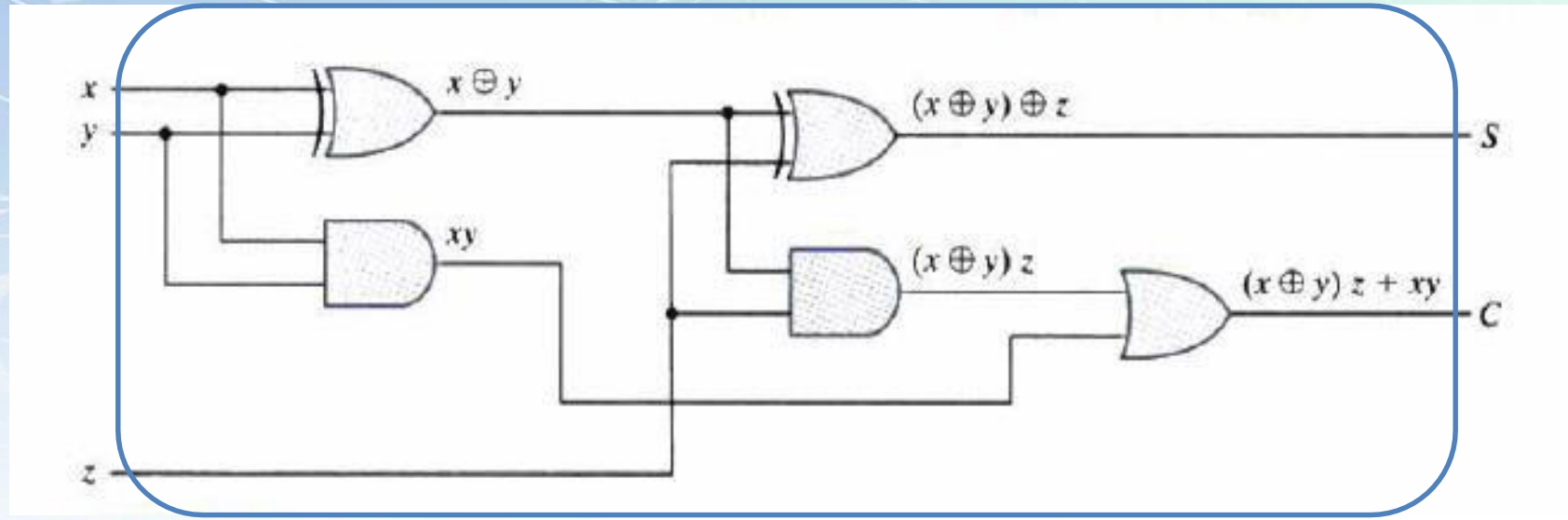
Fig. 6-4 Serial Transfer from Register *A* to register *B*

Parallel Addition (Chapter 4)

- **Add two n-bit numbers** together, **n full adders** should be **cascaded**
- **Each full-adder** represents a **column** in the long addition.
- The **carry signals** '**ripple**' through the adders from **right to left**



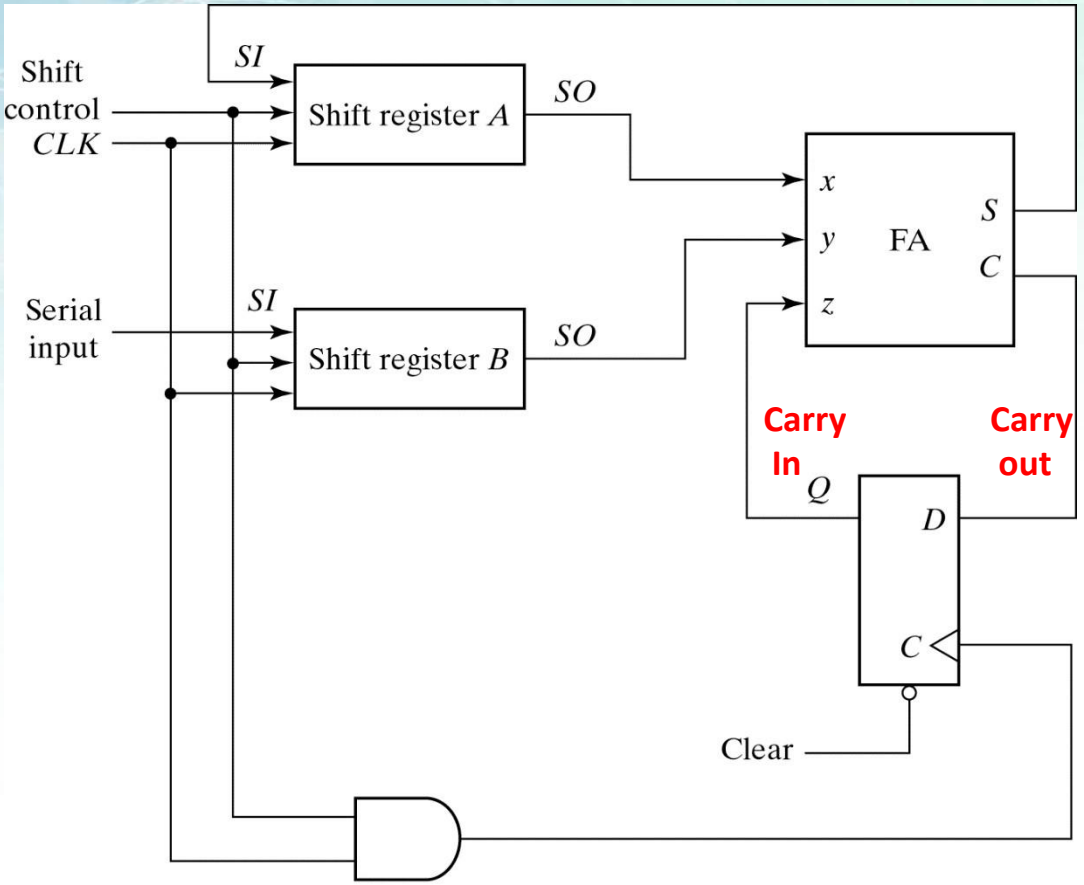
Full Adder (Chapter 4)



The 3-level logic circuit with XOR gates

Serial Addition

- **Slower** than parallel but has **lower cost**
- **Only one full adder** reused for **each bit**
- **Start** with **low-order bit** addition
- **Carry (Q)** is **saved**
- **Sum**, one bit at a time, is **transferred** into **register A**



Serial adder with D flip-flop

Fig. 6-5 Serial Adder

Designing a JK Serial Adder

Carry In **Carry out**

Present State Q	Inputs x	y	Next State Q	Output S	Flip_Flop Inputs 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

State table and JK flip-flop inputs

JK flip-flop Excitation Table

Q	Q_{next}	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

$$J_Q = x \cdot y$$

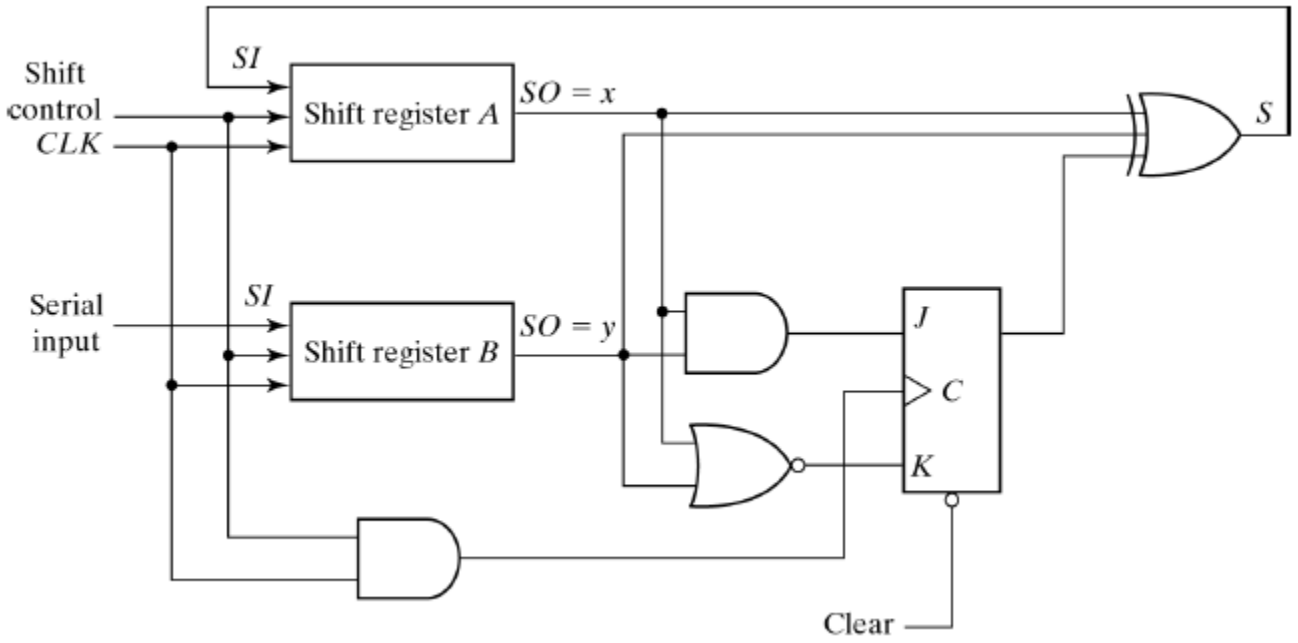
$$K_Q = x' \cdot y' = (x+y)'$$

FF input equations

$$S = x \oplus y \oplus Q$$

Output equation

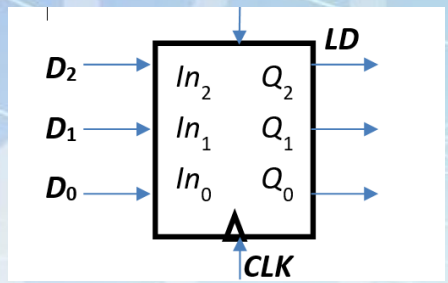
Designing a JK Serial Adder (cont'd)



Serial adder with JK flip-flop

Exercise

Design a 3-bit parallel loading register using T flip-flops, as described in the following functional table. Draw the logic diagram of the circuit that you designed; explain your work.



LD	Q_i^{n+1} (next state)
0	D_i^n (data input)
1	Q_i^n (present state)

$i = \{0, 1, 2\}$

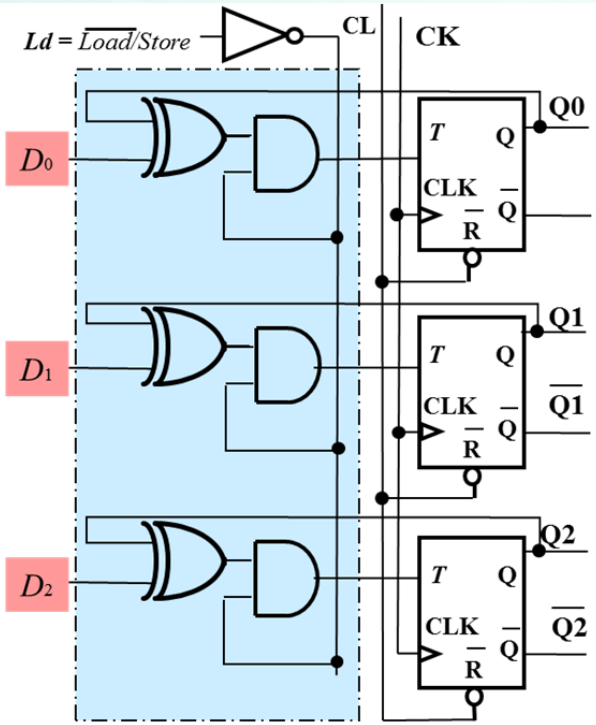
Solution

Full excitation table solution:

LD	Q_i^n	D_i^n	Q_i^{n+1}	T_i
0	0	0	0	0
0	0	1	1	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	0

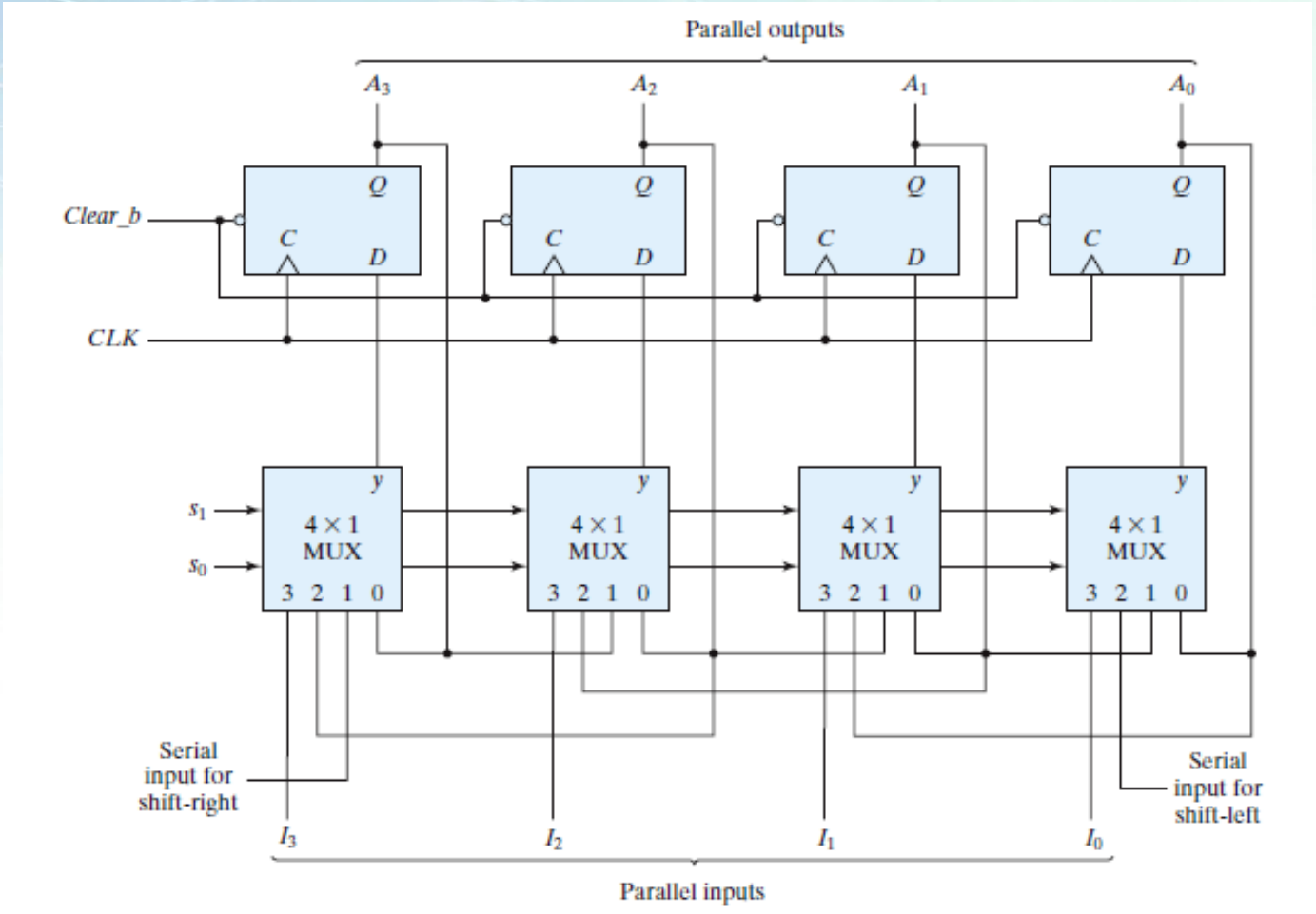
$\Rightarrow T_i = LD' (Q_i D_i' + Q_i' D_i)$, $i = \{0, 1, 2\}$

The circuit can be implemented with gates only



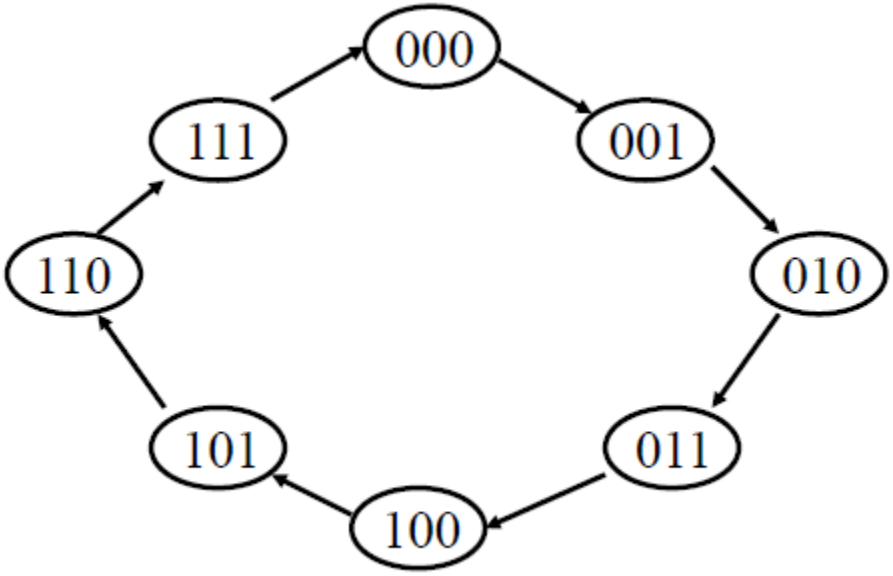
Draw the logic diagram of a 4-bit register with mode selection inputs S_1 and S_0. The register is to be operated according to the function table. Note, allow a serial input for both shift operations.

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



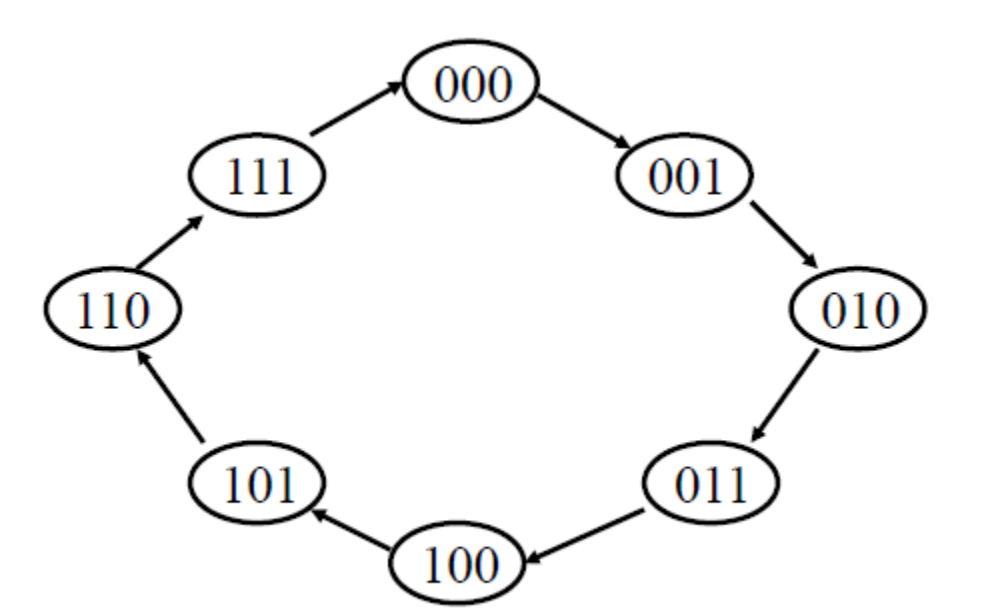
Binary Counters

- **Counter** is a **register** that goes through a **prescribed** or **fixed sequence** of **binary digits** or **states**
- **Counters** are normally **designed** to **recycle** or **restart** the **sequence**
- **Example:** 3-bit counter



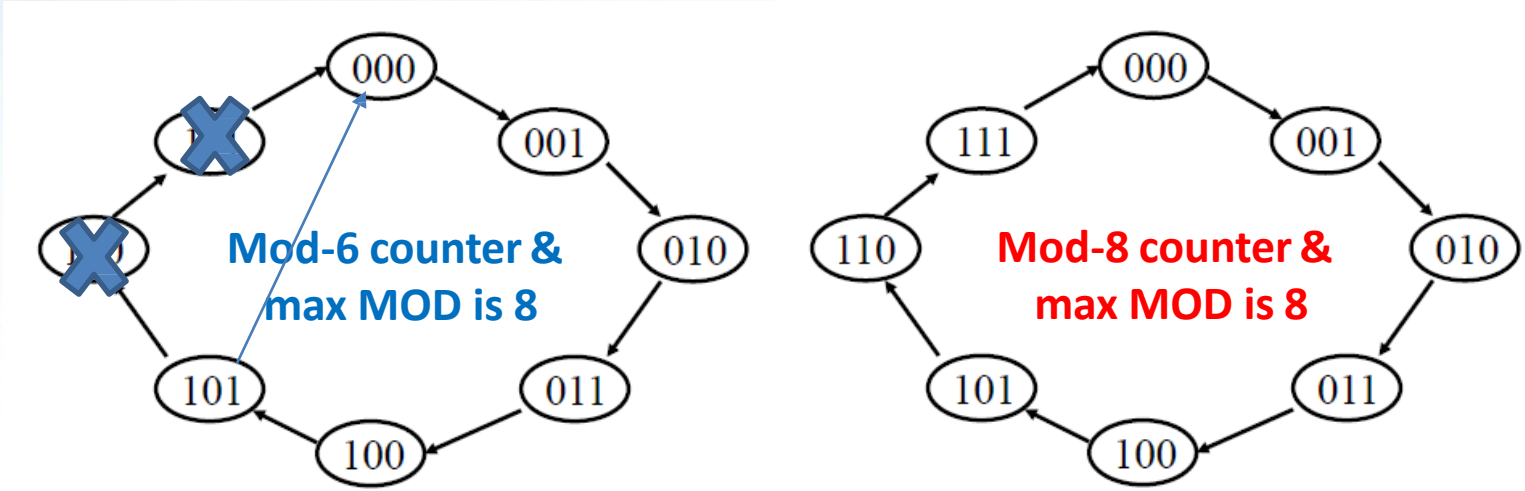
Binary Counters: Modulus

- The **number** of **output states** is called the **Modulus** or **MOD**
  - For example, a **mod-8 counter** has **8 unique output states**



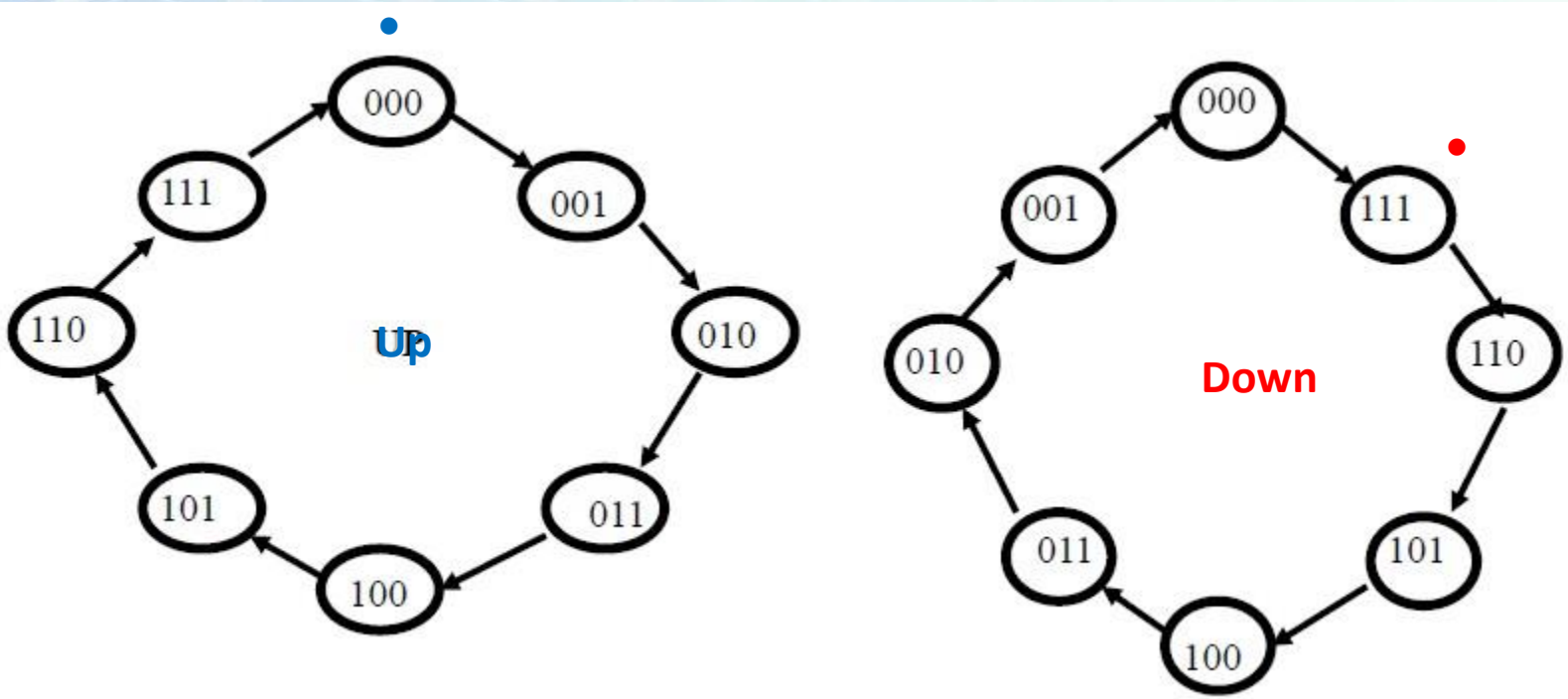
Binary Counters: Sequences

- **Full Sequence Count**
  - Refers to a natural count that **includes all possible binary numbers**. **Its modulus** is the **same** as its **maximum modulus**
- **A Truncated Sequence Count**
  - When the **modulus** is **less than its maximum**, or where **less than all possible binary numbers** are **used**

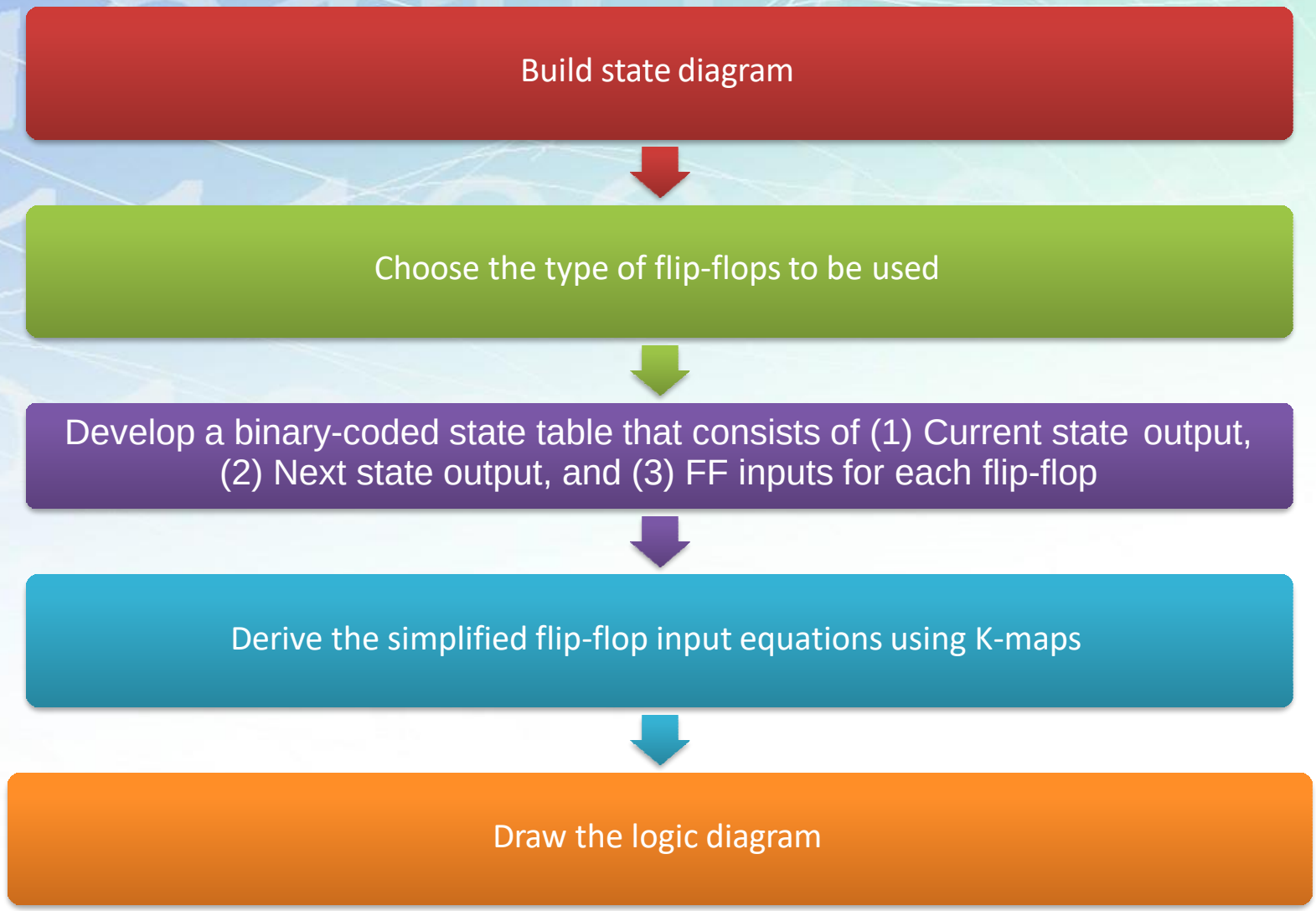


Binary Counters: Up/Down

- A **sequential count** refers to a **natural numerical count**
  - The **sequence** can be **Up** or **Down**

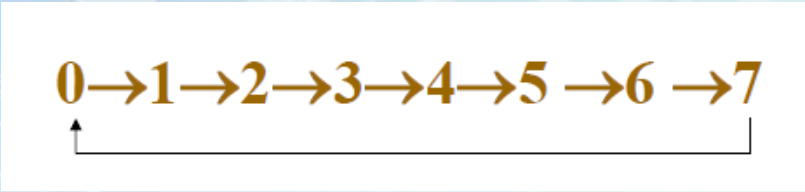


Synchronous Counters: Design Procedure

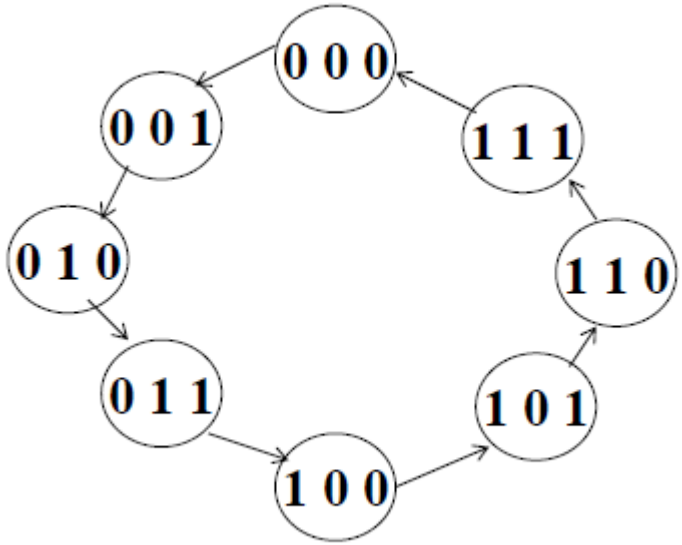


Synchronous Counters with JK Flip-Flop

- Example **without missing states (full sequence count)**



- 3-bit binary up counter
- 3 JK flip-flops are needed
- Current state and next state outputs are 3 bits each
- 3 pairs of JK inputs



Synchronous Counters with JK Flip-Flop (cont'd)

Present state			Next state			JK 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	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
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	1	1	1	X	0	X	0	1	X
1	1	1	0	0	0	X	1	X	1	X	1

State table and JK flip-flop inputs

JK flip-flop Excitation Table or Transition Table

Q	Q_{next}	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Synchronous Counters with JK Flip-Flop (cont'd)

Use K-maps to simplify expressions for JK inputs

X= d

A \ BC	00	01	11	10
0	0	0	1	0
1	d	d	d	d

$J_A = B\,C$

A \ BC	00	01	11	10
0	d	d	d	d
1	0	0	1	0

$K_A = B\,C$

A \ BC	00	01	11	10
0	0	1	d	d
1	0	1	d	d

$J_B = C$

A \ BC	00	01	11	10
0	d	d	1	0
1	d	d	1	0

$K_B = C$

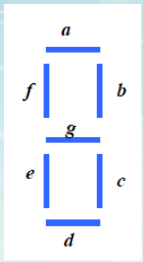
A \ BC	00	01	11	10
0	1	d	d	1
1	1	d	d	1

$J_C = 1$

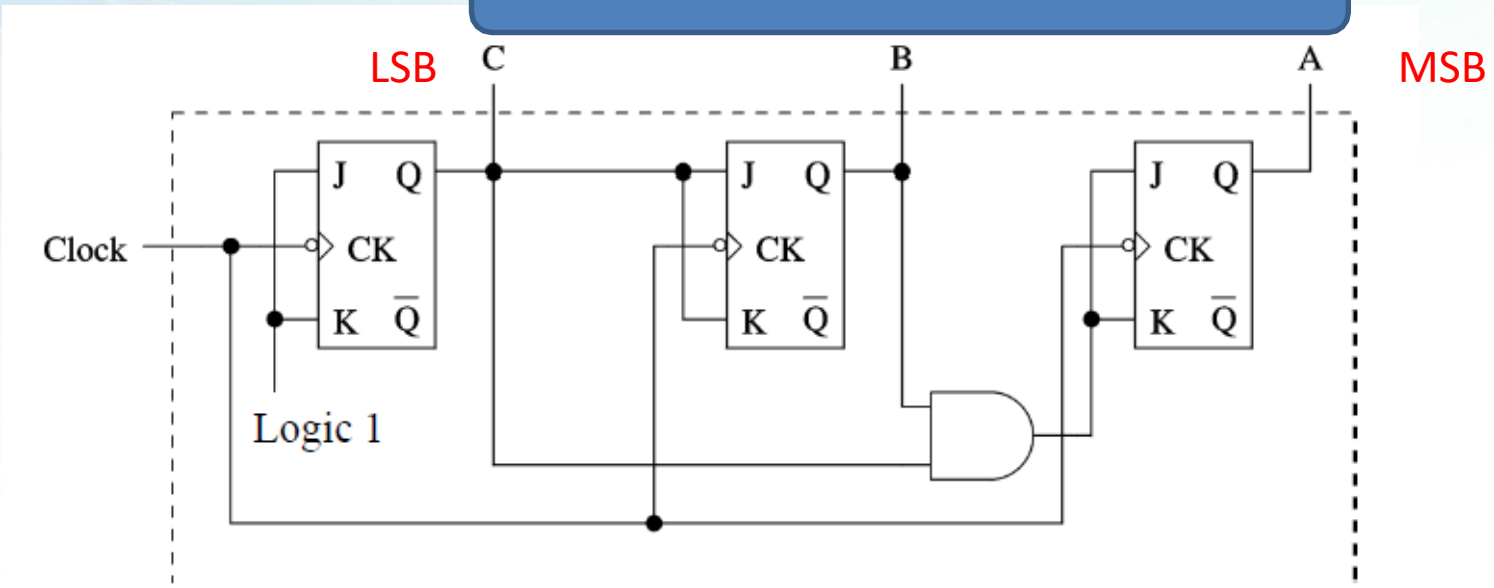
A \ BC	00	01	11	10
0	d	1	1	d
1	d	1	1	d

$K_C = 1$

Synchronous Counters with JK Flip-Flop (cont'd)



BCD to 7 Segment Display Decoder

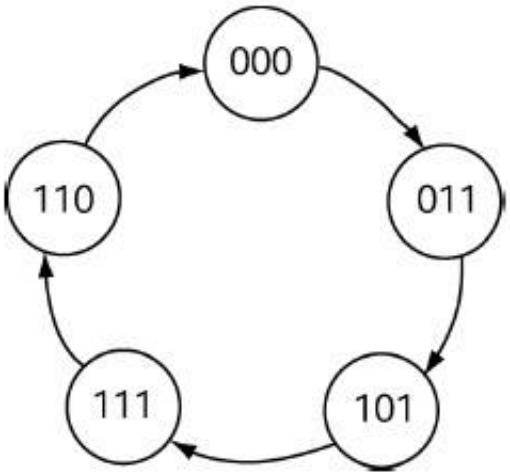


Logic diagram of Mod-8 synchronous, full-sequence, 3-bit, up counter

Synchronous Counters with JK Flip-Flop (cont'd)

- Example **with missing states (truncated sequence count)**

$0 \rightarrow 3 \rightarrow 5 \rightarrow 7 \rightarrow 6 \rightarrow 0$



- Same design process as before
- One significant change
  - Missing states: 1, 2, and 4
  - Use don't cares for these states

Synchronous Counters with JK Flip-Flop (cont'd)

Present state			Next state			JK 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	1	1	0	X	1	X	1	X
0	0	1	X	X	X	X	X	X	X	X	X
0	1	0	X	X	X	X	X	X	X	X	X
0	1	1	1	0	1	1	X	X	1	X	0
1	0	0	X	X	X	X	X	X	X	X	X
1	0	1	1	1	1	X	0	1	X	X	0
1	1	0	0	0	0	X	1	X	1	0	X
1	1	1	1	1	0	X	0	X	0	X	1

State table and JK flip-flop inputs

JK flip-flop Excitation Table or Transition Table

Q	Q_{next}	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Synchronous Counters with JK Flip-Flop (cont'd)

K-maps to simplify JK input expressions

X=d

A \ BC	00	01	11	10
0	0	d	1	d
1	d	d	d	d

$J_A = B$

A \ BC	00	01	11	10
0	d	d	d	d
1	d	0	0	1

$K_A = \overline{C}$

A \ BC	00	01	11	10
0	1	d	d	d
1	d	1	d	d

$J_B = 1$

A \ BC	00	01	11	10
0	d	d	1	d
1	d	d	0	1

$K_B = \overline{A} + \overline{C}$

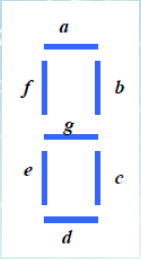
A \ BC	00	01	11	10
0	1	d	d	d
1	d	d	d	0

$J_C = \overline{A}$

A \ BC	00	01	11	10
0	d	d	0	d
1	d	0	1	d

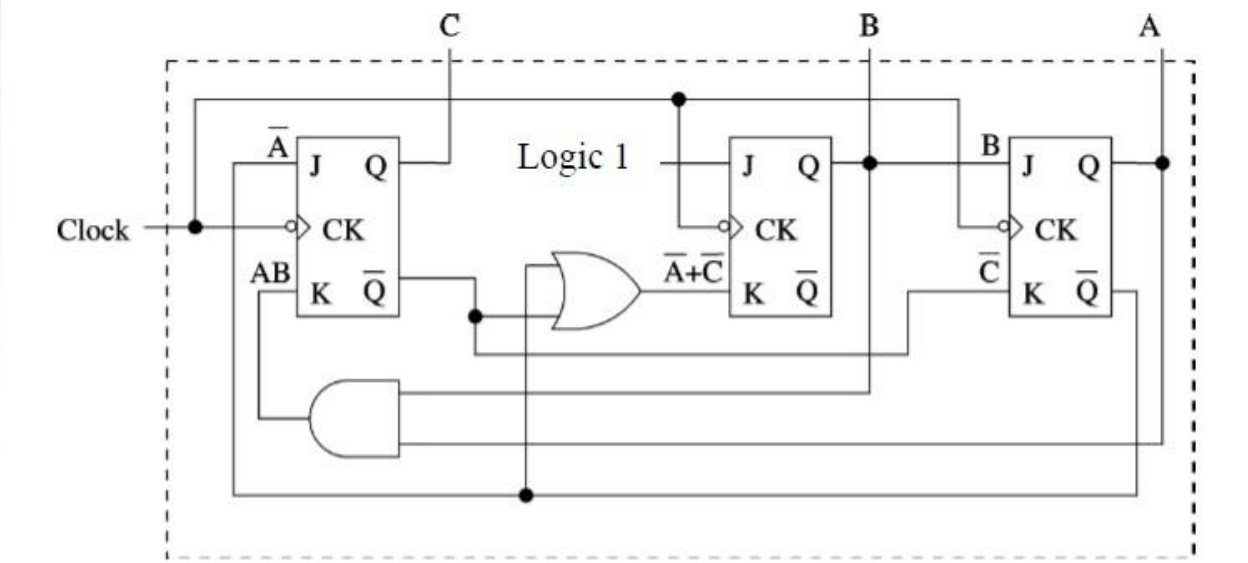
$K_C = A\,B$

Synchronous Counters with JK Flip-Flop (cont'd)



BCD to 7 Segment Display Decoder

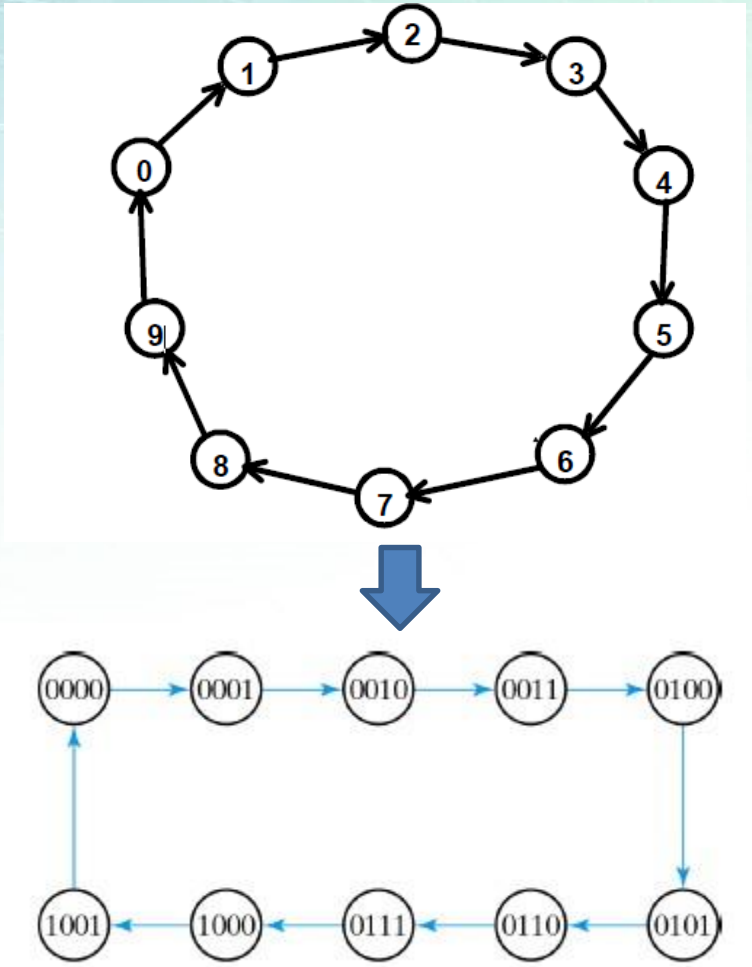
LSB MSB



Logic diagram of Mod-5 synchronous, truncated-sequence, 3-bit counter

Synchronous BCD Counter

- Design a BCD counter (or decade counter) using T flip-flop
- 4-bit binary up counter
- Counter with missing states (truncated sequence count)
- 4 T flip-flops are needed
- Current state and next state outputs are 4 bits each
- 4 T inputs



Synchronous BCD Counter(cont'd)

Current State $Q_8\ Q_4\ Q_2\ Q_1$	Next State $Q_8\ Q_4\ Q_2\ Q_1$	T-FF inputs $T_8\ T_4\ T_2\ T_1$
0 0 0 0	0 0 0 1	0 0 0 1
0 0 0 1	0 0 1 0	0 0 1 1
0 0 1 0	0 0 1 1	0 0 0 1
0 0 1 1	0 1 0 0	0 1 1 1
0 1 0 0	0 1 0 1	0 0 0 1
0 1 0 1	0 1 1 0	0 0 1 1
0 1 1 0	0 1 1 1	0 0 0 1
0 1 1 1	1 0 0 0	1 1 1 1
1 0 0 0	1 0 0 1	0 0 0 1
1 0 0 1	0 0 0 0	1 0 0 1

State table and T flip-flop inputs

T flip-flop Excitation Table or Transition Table

Q	Q_{next}	T
0	0	0
0	1	1
1	0	1
1	1	0

Note: The unused states for minterms 10 to 15 are taken as Don't care terms and have been left out here

Synchronous BCD Counter (cont'd)

- **Use K-Maps to minimize the next state function:**

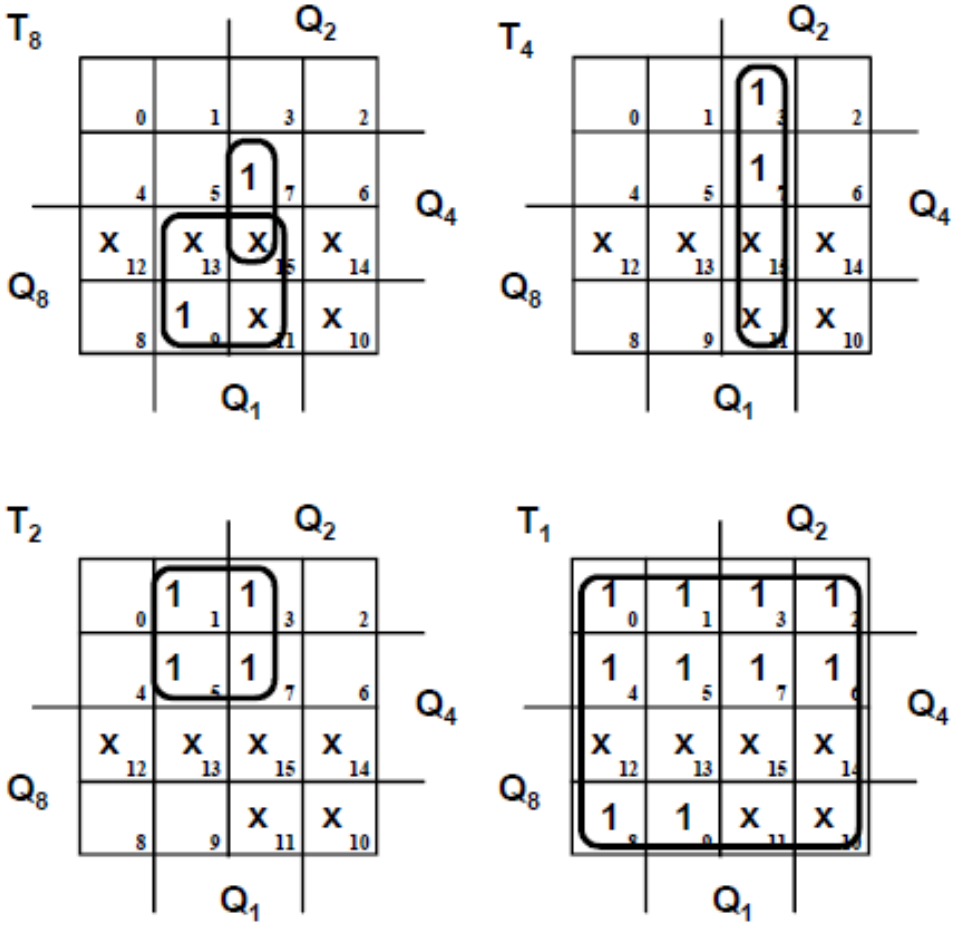
$T_8 = Q_8 \bullet Q_1 + Q_4 \bullet Q_2 \bullet Q_1$

$T_4 = Q_2 \bullet Q_1$

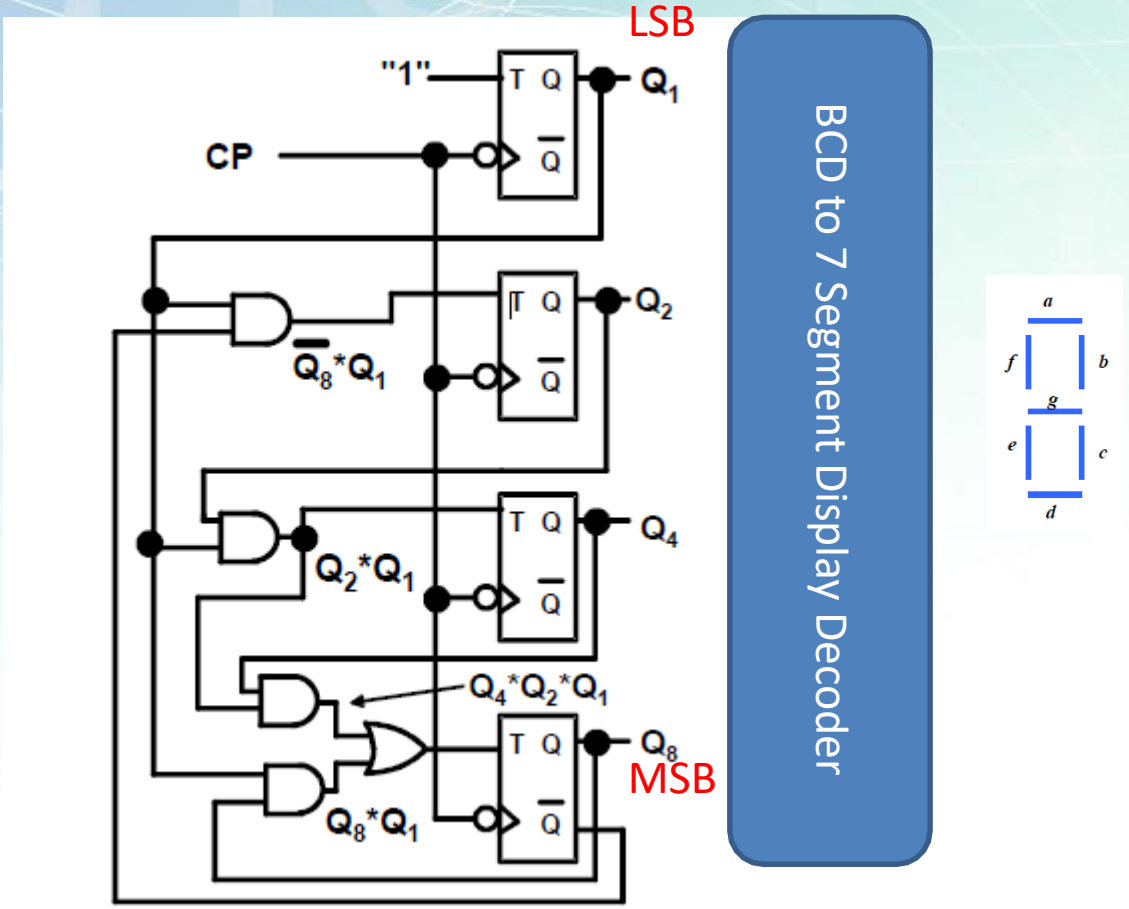
$T_2 = Q_8' \bullet Q_1$

$T_1 = \text{"1"}$

Note: <u>Don't Cares</u> are included here.



Synchronous BCD Counter (cont'd)



Logic diagram of Synchronous BCD, truncated-sequence, mod-10, 4-bit up counter

Synchronous BCD Counter (cont'd)

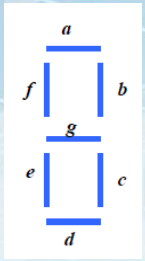
- **Synchronous BCD counters** can be **cascaded** to form a **counter** for **decimal numbers** of **any length (>9)**
  - By enabling the count of the next-higher significant decade (**y=1**)

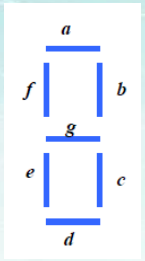
Present State Q_8	Q_4	Q_2	Q_1	Next State Q_8	Q_4	Q_2	Q_1	Output y	Next State TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1.	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

$TQ_1 = 1, \quad TQ_2 = Q'_8Q_1, \quad TQ_4 = Q_2Q_1, \quad TQ_8 = Q_8Q_1 + Q_4Q_2Q_1 \quad y = Q_8Q_1$

Synchronous BCD Counter (cont'd)

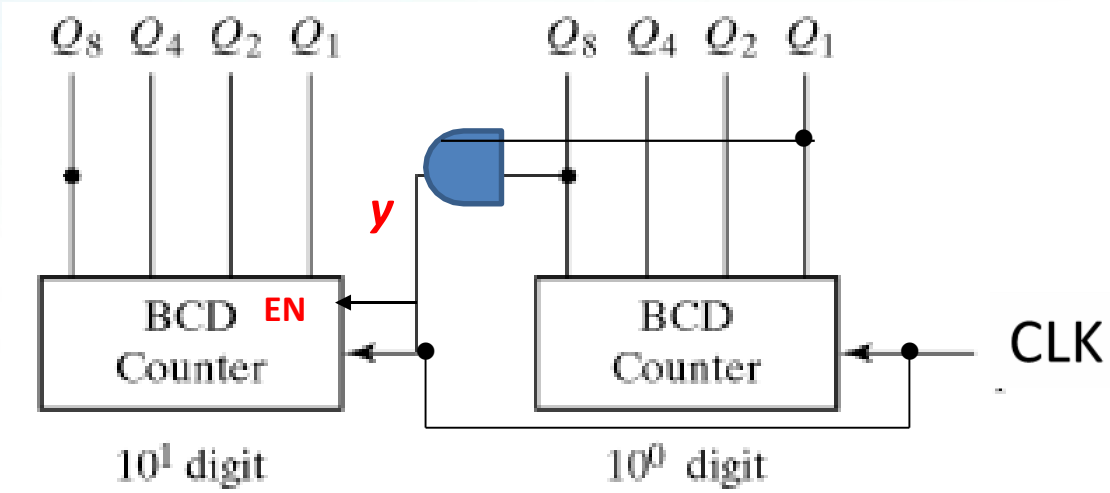
- **2-decade synchronous BCD counter** which is **mod-100 up counter**





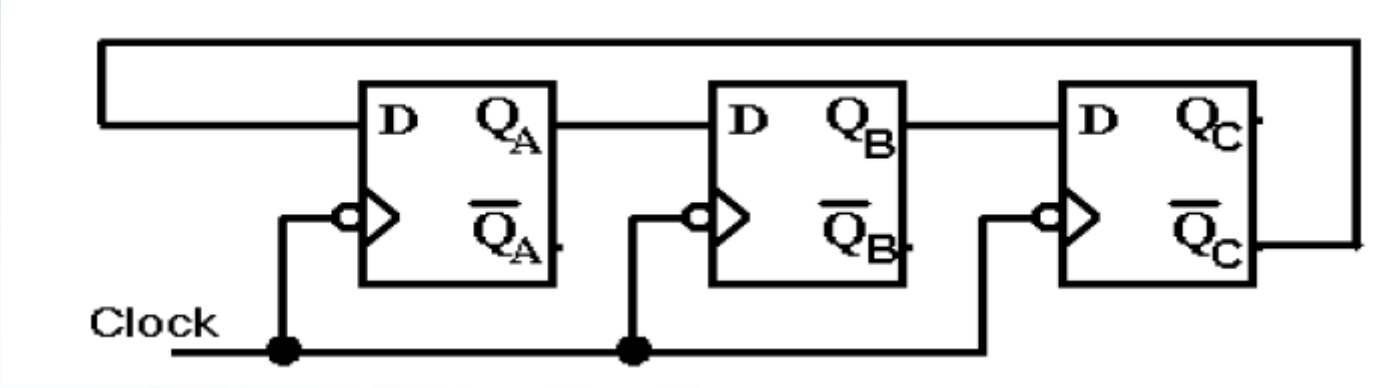
BCD to 7 Segment Display Decoder

BCD to 7 Segment Display Decoder

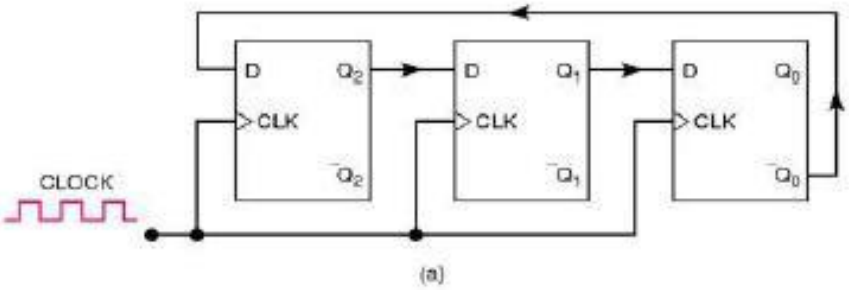


Johnson Counter

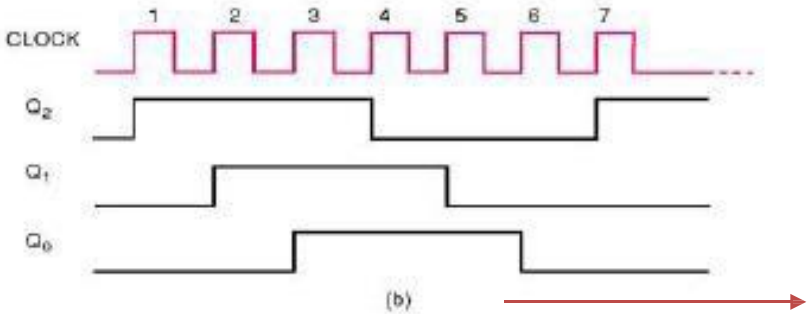
- **A Johnson Counter re-circulates** the **last flip-flop Q'** (inverted) **output back** to the **input** of the **first flip-flop**
  - Count sequence: 000, 100, 110, 111, 011, 001



Johnson Counter (cont'd)



(a)

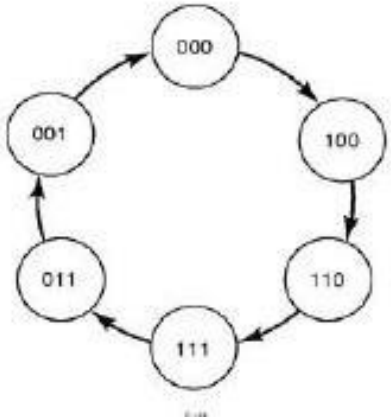


(b)

Time

Q_2	Q_1	Q_0	CLOCK pulse
0	0	0	0
1	0	0	1
1	1	0	2
1	1	1	3
0	1	1	4
0	0	1	5
0	0	0	6
1	0	0	7
1	1	0	8
.	.	.	.
.	.	.	.
.	.	.	.

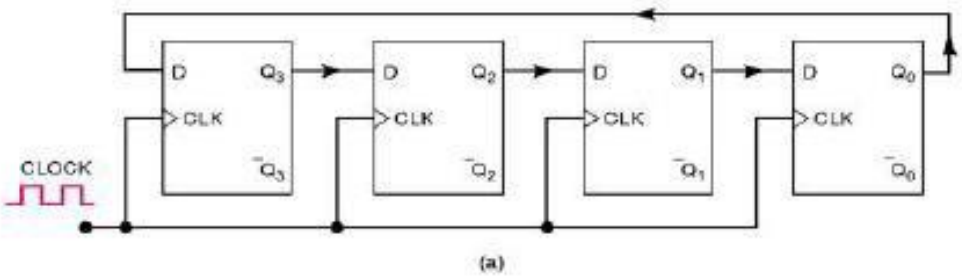
(c)



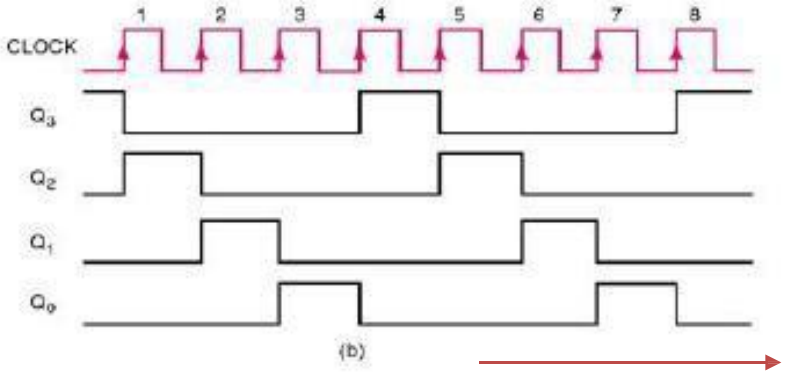
Ring Counter

- A ring counter takes the serial output of the last flip-flop of a shift register and provides it to the serial input of the first flip-flop
- This is also known as a re-circulating shift register

Ring Counter (cont'd)



(a)

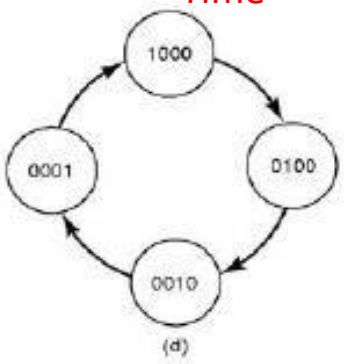


(b)

Time

Q_3	Q_2	Q_1	Q_0	CLOCK pulse
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7
.	.	.	.	.
.	.	.	.	.

(c)



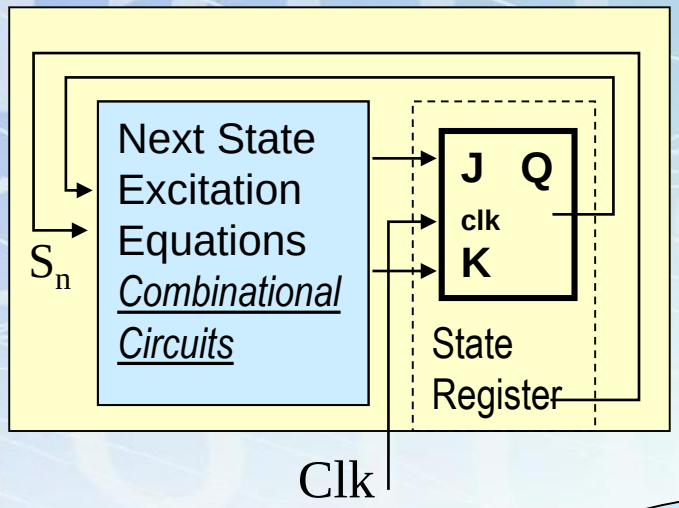
(d)

Summary

Synchronous 4-bit Counter Design

Step 1

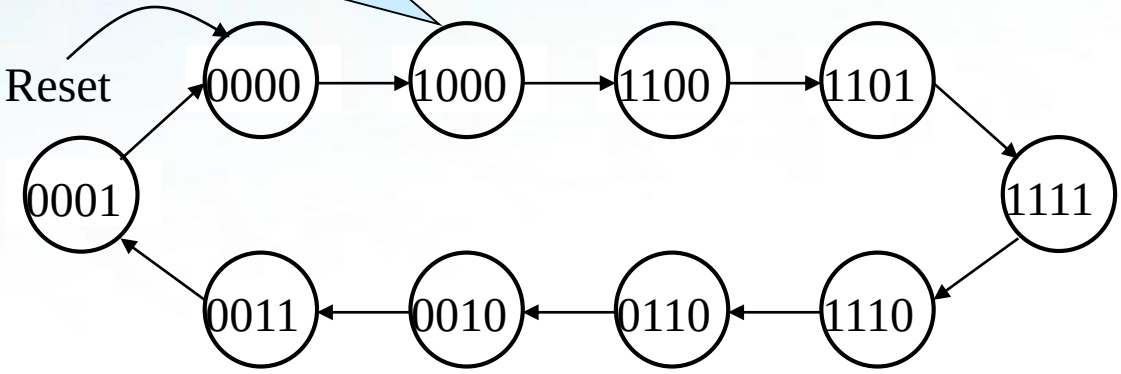
Sequential Circuit Block Diagram



JK-FF Excitation Table

Q^n	Q^{n+1}	J^n	K^n
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

State Diagram



Transition Table

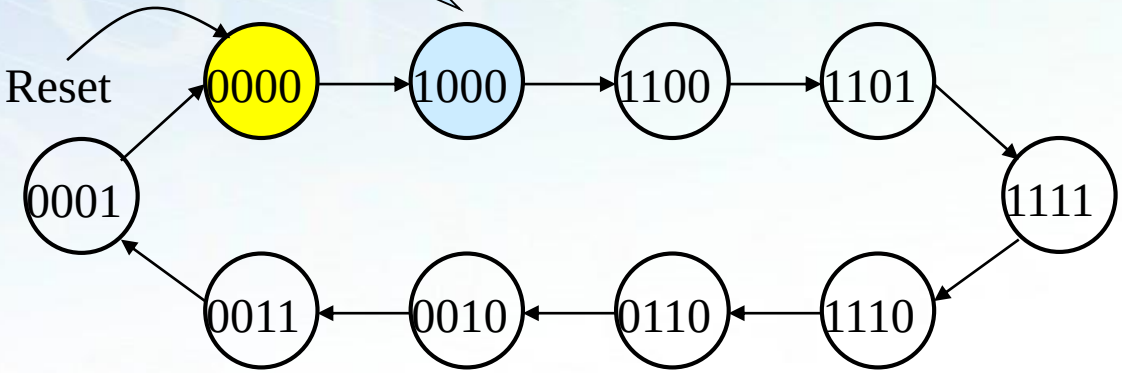
Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	1
0	0	1	1	0	0	0	1
0	1	0	0	x	x	x	x
0	1	0	1	x	x	x	x
0	1	1	0	0	0	1	0
0	1	1	1	x	x	x	x
1	0	0	0	1	1	0	0
1	0	0	1	x	x	x	x
1	0	1	0	x	x	x	x
1	0	1	1	x	x	x	x
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	1	1	0

Synchronous 4-bit Counter Design

Step 1

To derive the State Table, go through the state diagram, starting from initial state (0000).

State Diagram



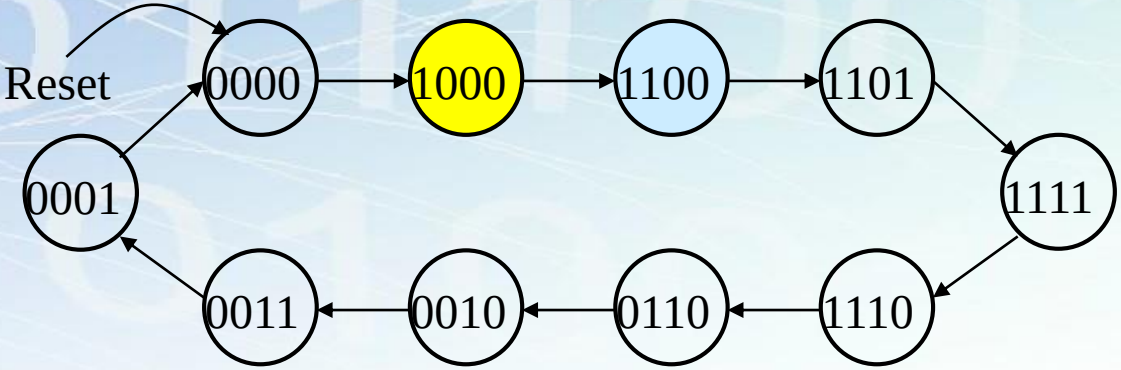
Transition Table

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	1	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				

Synchronous 4-bit Counter Design

State Diagram

Step 1



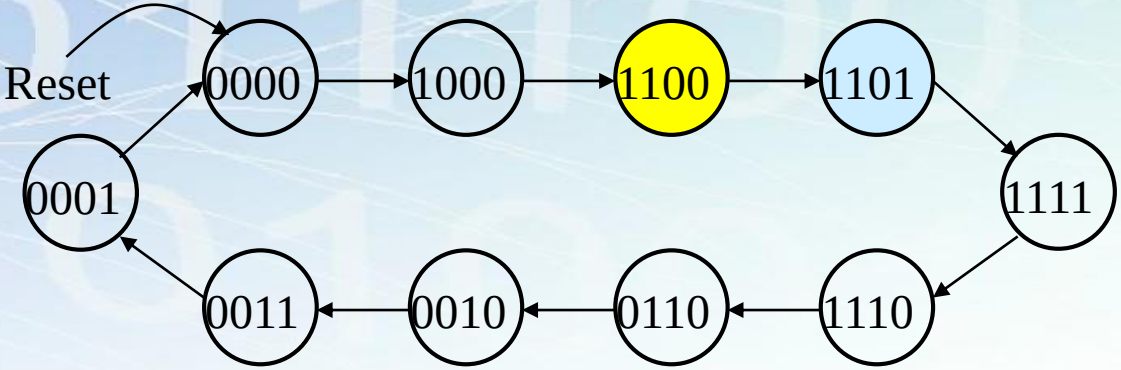
Transition Table

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	**0**	**0**	**0**	**1**	**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**	**1**	**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				

Synchronous 4-bit Counter Design

State Diagram

Step 1



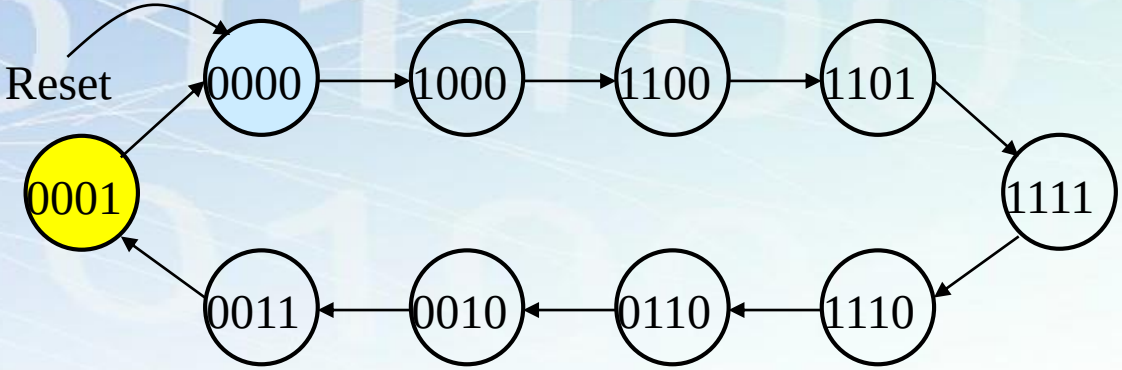
Transition Table

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	**0**	**0**	**0**	**1**	**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**	**1**	**0**	**0**
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	**1**	**0**	**0**	**1**	**1**	**0**	**1**
1	1	0	1				
1	1	1	0				
1	1	1	1				

Synchronous 4-bit Counter Design

State Diagram

Step 1



Transition Table

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	1
0	0	1	1	0	0	0	1
0	1	0	0				
0	1	0	1				
0	1	1	0	0	0	1	0
0	1	1	1				
1	0	0	0	1	1	0	0
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	1	1	0

Synchronous 4-bit Counter Design

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	1
0	0	1	1	0	0	0	1
0	1	0	0				
0	1	0	1				
0	1	1	0	0	0	1	0
0	1	1	1				
1	0	0	0	1	1	0	0
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	1	1	0

Step 1

Empty cells represent "**Can't happen conditions**" and they can be used as ***don't care*** terms for minimization

Present state S^n				Next state S^{n+1}			
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	1
0	0	1	1	0	0	0	1
0	1	0	0	x	x	x	x
0	1	0	1	x	x	x	x
0	1	1	0	0	0	1	0
0	1	1	1	x	x	x	x
1	0	0	0	1	1	0	0
1	0	0	1	x	x	x	x
1	0	1	0	x	x	x	x
1	0	1	1	x	x	x	x
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	1	1	0

Counter Design

Excitation Table

Step 3

Present state S^n				Next state S^{n+1}				Q_3 input		Q_2 input		Q_1 input		Q_0 input	
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	1	0	0	0	1	x	0	x	0	x	0	x
0	0	0	1	0	0	0	0	0	x	0	x	0	x	x	1
0	0	1	0	0	0	1	1	0	x	0	x	x	0	1	x
0	0	1	1	0	0	0	1	0	x	0	x	x	1	x	0
0	1	0	0	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
0	1	0	1	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
0	1	1	0	0	0	1	0	0	x	x	1	x	0	0	x
0	1	1	1	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
1	0	0	0	1	1	0	0	x	0	1	x	0	x	0	x
1	0	0	1	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
1	0	1	0	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
1	0	1	1	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
1	1	0	0	1	1	0	1	x	0	x	0	0	x	1	x
1	1	0	1	1	1	1	1	x	0	x	0	1	x	x	0
1	1	1	0	0	1	1	0	x	1	x	0	x	0	0	x
1	1	1	1	1	1	1	0	x	0	x	0	x	0	x	1

Synchronous 4-bit Counter Design

Excitation Equations

Step 4

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	1	0	0	0
01	**x**	**x**	**x**	
11	x	x	x	x
10	x	**x**	**x**	**x**

$$J_3 = \overline{Q_1} \cdot \overline{Q_0}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	x	x	x
01	**x**	**x**	**x**	x
11	0	0	0	1
10	0	**x**	**x**	**x**

$$K_3 = Q_1 \cdot \overline{Q_0}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	**x**	**x**	**x**	x
11	x	x	x	x
10	1	**x**	**x**	**x**

$$J_2 = Q_3$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	x	x	x
01	**x**	**x**	**x**	1
11	0	0	0	0
10	x	**x**	**x**	**x**

$$K_2 = \overline{Q_3}$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	x	x
01	**x**	**x**	**x**	x
11	0	1	x	x
10	0	**x**	**x**	**x**

$$J_1 = Q_3 \cdot Q_0$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	x	1	0
01	**x**	**x**	**x**	0
11	x	x	0	0
10	x	**x**	**x**	**x**

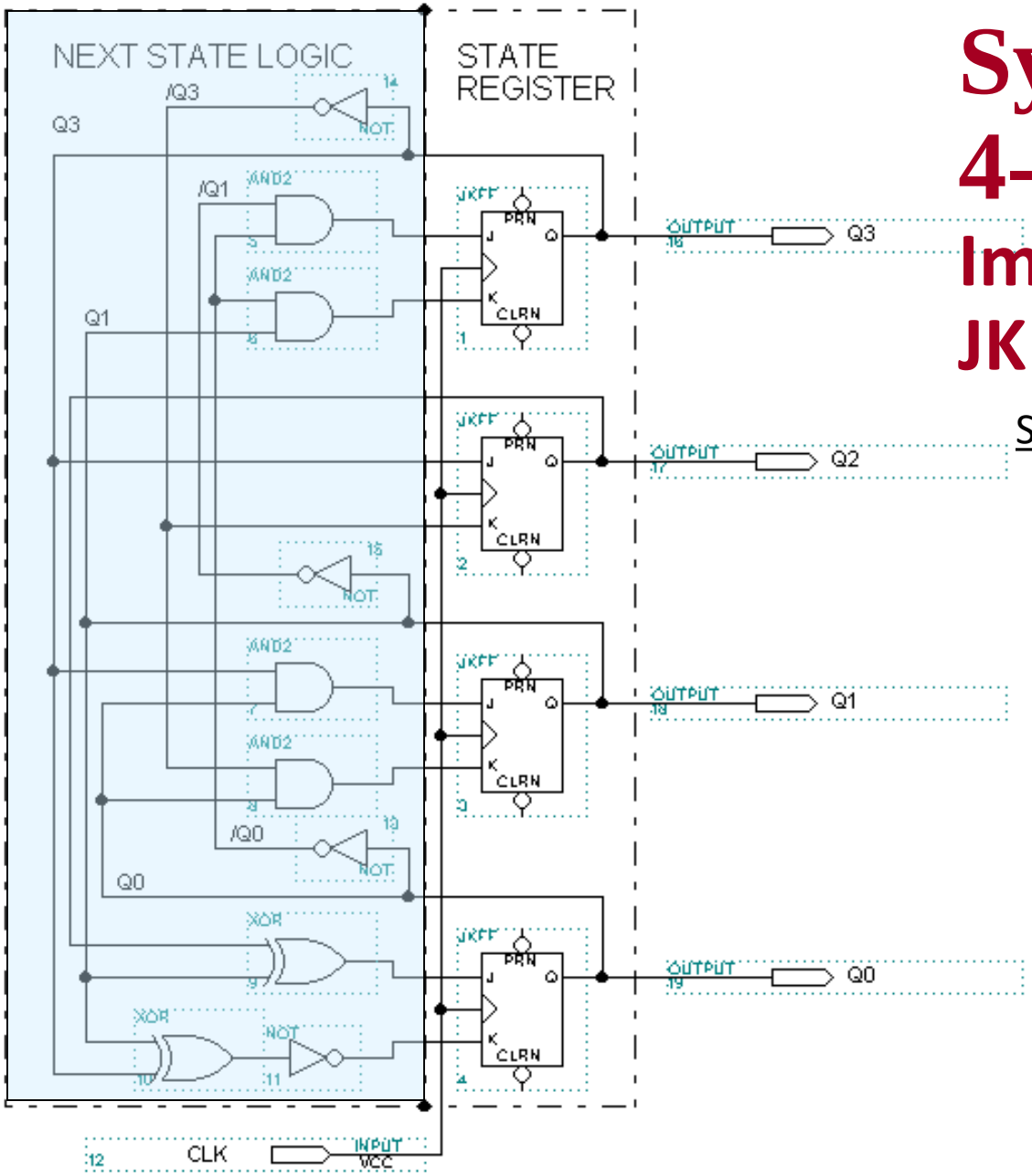
$$K_1 = \overline{Q_3} \cdot Q_0$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	x	x	1
01	**x**	**x**	**x**	0
11	1	x	x	0
10	0	**x**	**x**	**x**

$$J_0 = Q_2 \oplus Q_1$$

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	x	1	0	x
01	**x**	**x**	**x**	x
11	x	0	1	x
10	x	**x**	**x**	**x**

$$K_0 = \overline{Q_3 \oplus Q_1}$$



Synchronous 4-bit Counter

Implementation with JK flip-flops

Step 5

$$J_3 = \overline{Q_1} \cdot \overline{Q_0}$$
$$K_3 = Q_1 \cdot \overline{Q_0}$$

$$J_2 = Q_3$$
$$K_2 = \overline{Q_3}$$

$$J_1 = Q_3 \cdot Q_0$$
$$K_1 = \overline{Q_3} \cdot Q_0$$

$$J_0 = \overline{Q_2 \oplus Q_1}$$
$$K_0 = Q_3 \oplus Q_1$$