Synthesis of clocked sequential circuits

Synthesis of clocked sequential circuit

- **Clocked sequential circuit** – a logic design circuit which includes flip-flops with clock inputs.

- May or may not include combinational logic gates.

- **State table and state diagram** – describe the behavior of a sequential circuit.

Synthesis of clocked sequential circuit

- Behavior of a clocked sequential circuit - determined from the inputs, outputs and the state of its flip-flops.

- Outputs and next state are both – function of the inputs and the present state.

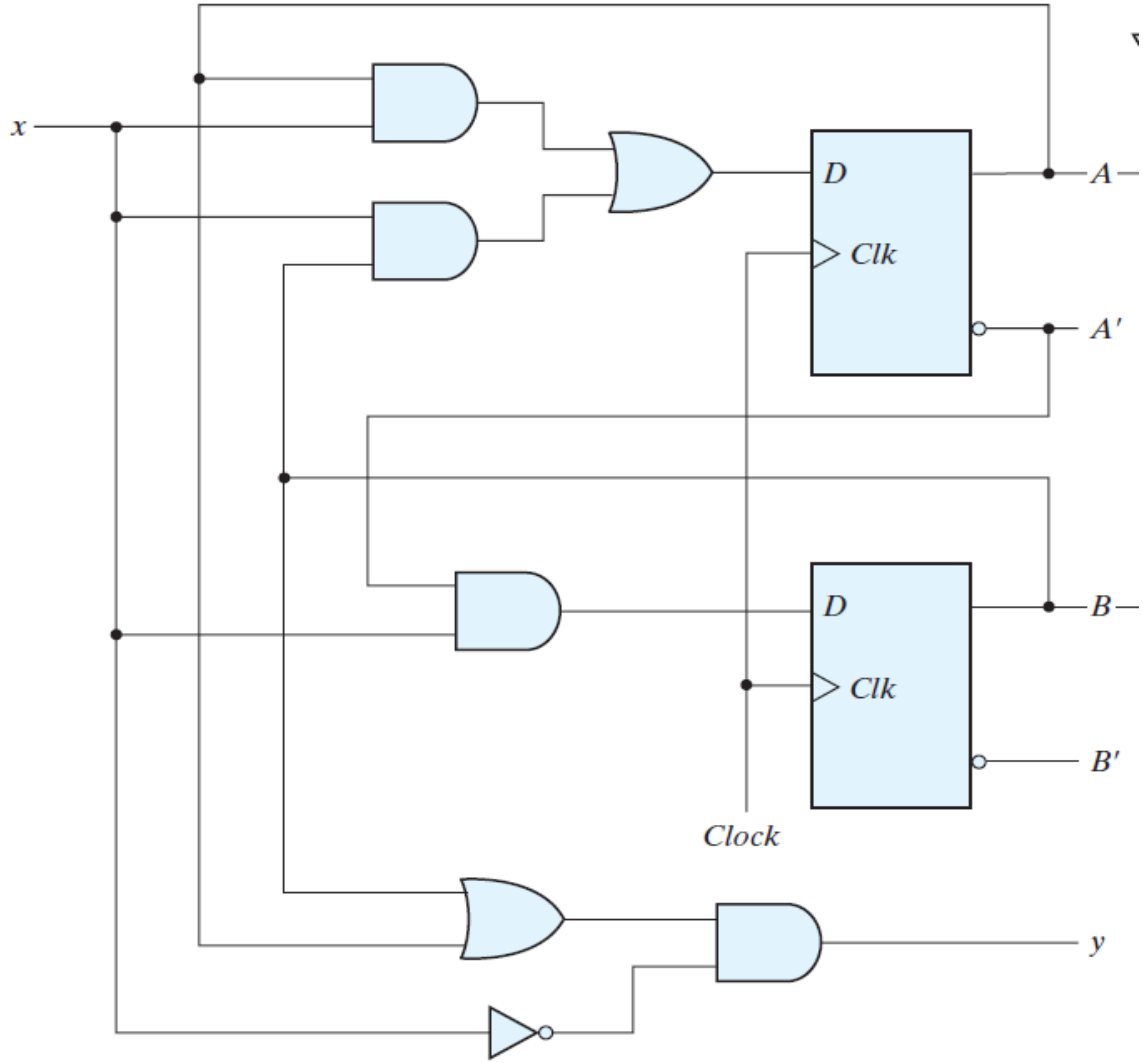
- Analysis of a sequential circuit – obtaining a table or a diagram for the time sequence of inputs, outputs and internal states.

Synthesis of clocked sequential circuit

- **State equations**: algebraic description of behavior of a clocked sequential circuit.

Transition equation – specifies the next state as a function of the present state and inputs.

Synthesis of clocked sequential circuit



$A(t + 1) - Ax + Bx$

$B(t + 1) – A'x$

$y = (A + B)x'$

Synthesis of clocked sequential circuit

$A(t + 1) - Ax + Bx$

$B(t + 1) – A'x$

$y = (A + B)x'$

$A(t + 1) = A(t)x(t) + B(t)x(t)$

$B(t + 1) = A'(t)x(t)$

$y(t) = [A(t) + B(t)]x'(t)$

Synthesis of clocked sequential circuit

- sequential circuit with *m* flip-flops and *n* inputs needs - 2^{m+n} rows in the state table.
- Binary numbers from 0 through 2^{m+n} - 1 are listed under the present-state and input columns.
- Next-state section has *m* columns, one for each flip-flop.
- Output section - No.of columns = No. of output variables.

State table - 1

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

State Table

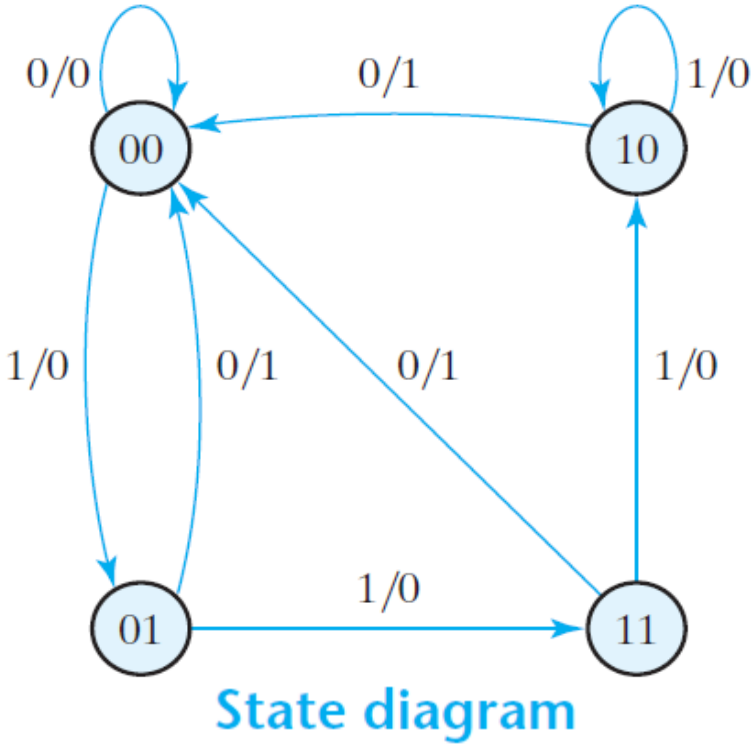
State table - 2

Second Form of the State Table

Present State		Next State				Output	
		$x = 0$		$x = 1$		$x = 0$	$x = 1$
A	B	A	B	A	B	y	y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

State Table

State diagram

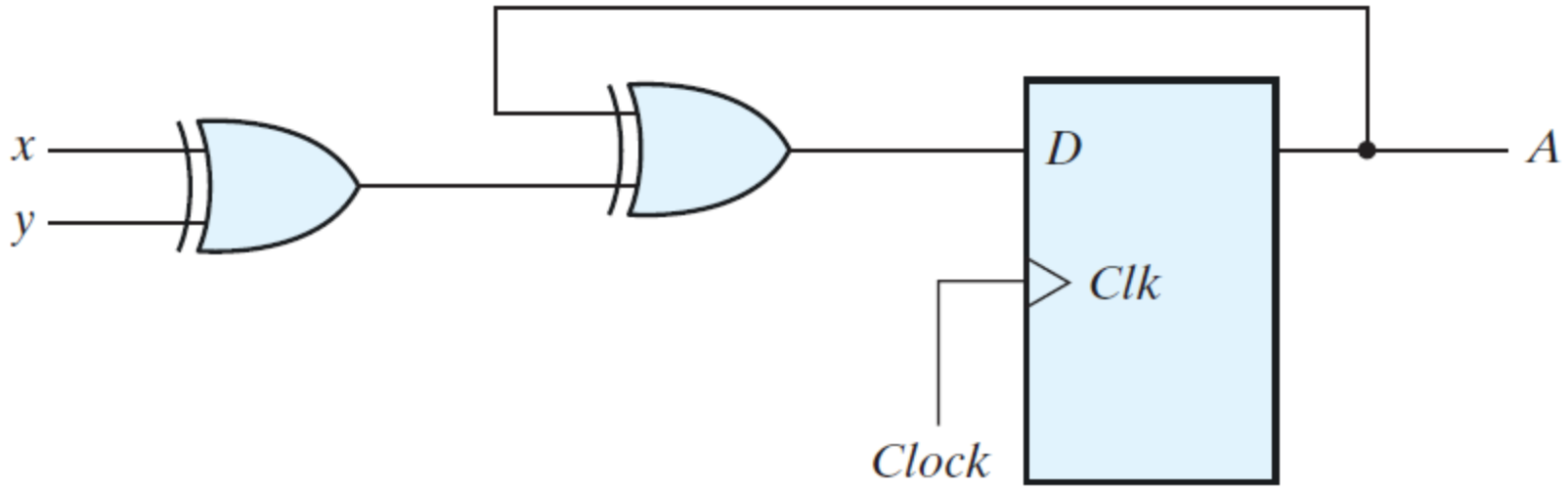


State diagram

A directed line connecting a circle with itself - no change of state occurs.

- State diagrams - Graphical representation of information available in state-table.
- Each state – represented by a circle.
- Transition between the states – directed lines connecting the states.
- Input during the present state / output during the present state for the given input.

sequential circuit – example 2

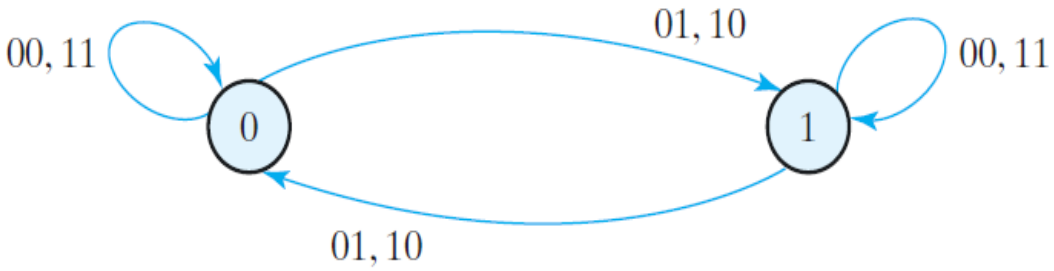


Sequential circuit with *D* flip-flop

Synthesis of clocked sequential circuit

Present state	Inputs		Next state
A	x	y	A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

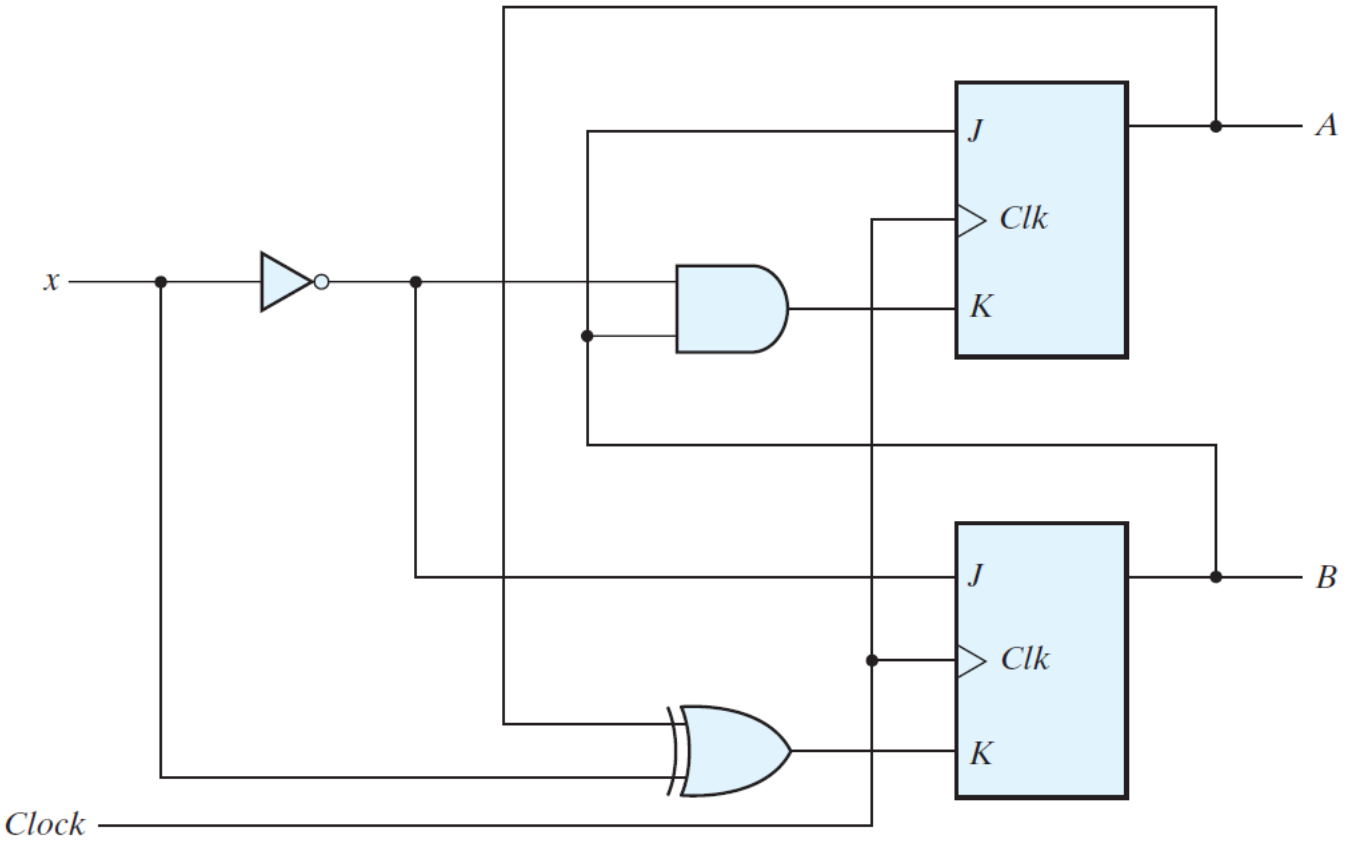
(b) State table



(c) State diagram

Sequential circuit with *D* flip-flop

Sequential circuit – example 3



Sequential circuit with *JK* flip-flop

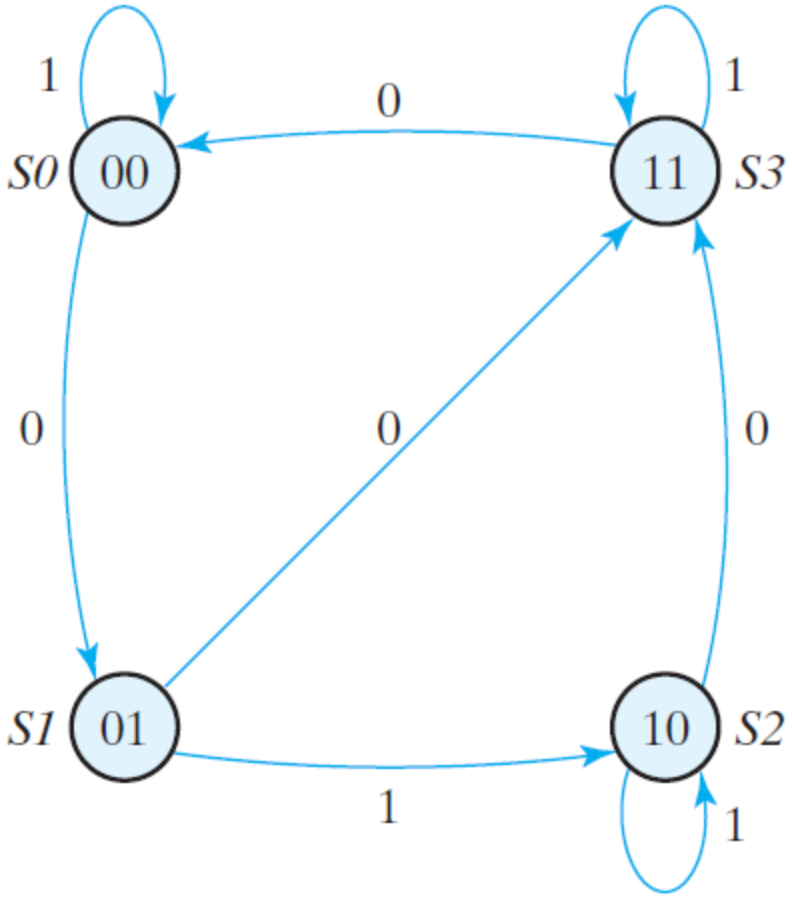
Synthesis of clocked sequential circuit

State Table for Sequential Circuit with JK Flip-Flops

Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

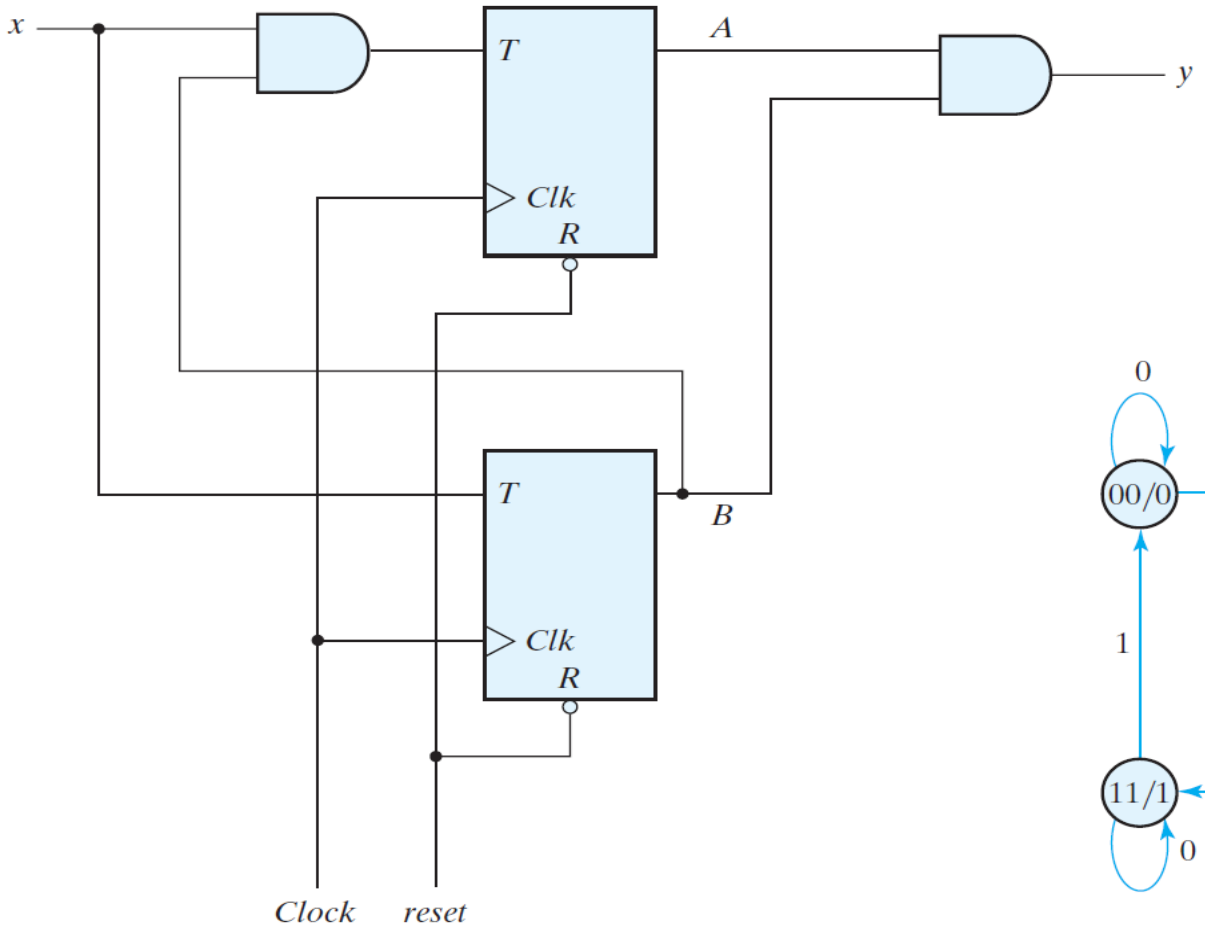
Sequential circuit with *JK* flip-flop

Synthesis of clocked sequential circuit

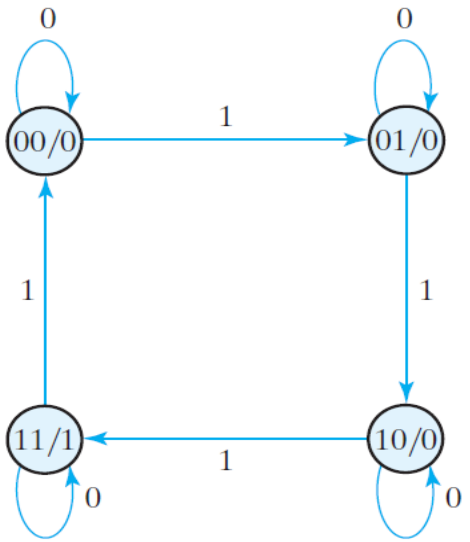


State diagram

Sequential circuit – example 4



(a) Circuit diagram



(b) State diagram

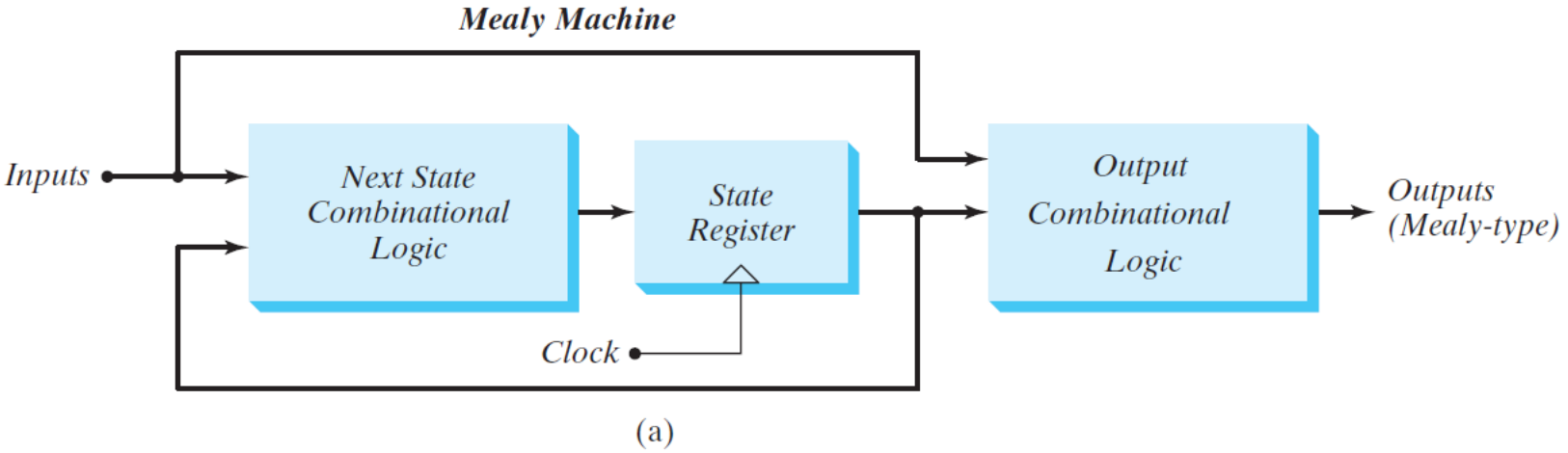
Sequential circuit with *T* flip-flops

Synthesis of clocked sequential circuit

State Table for Sequential Circuit with T Flip-Flops

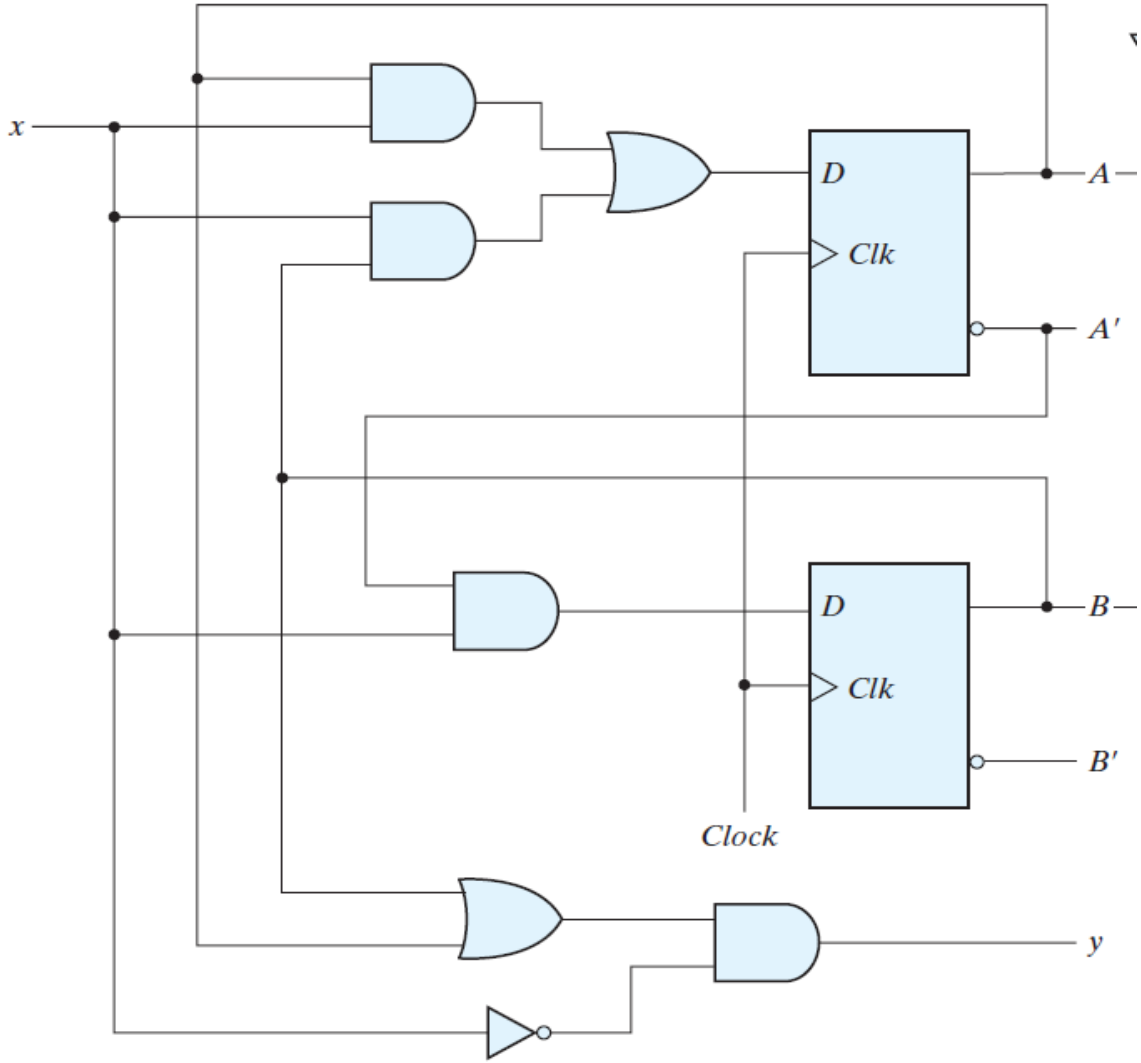
Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

Mealy and Moore state machines

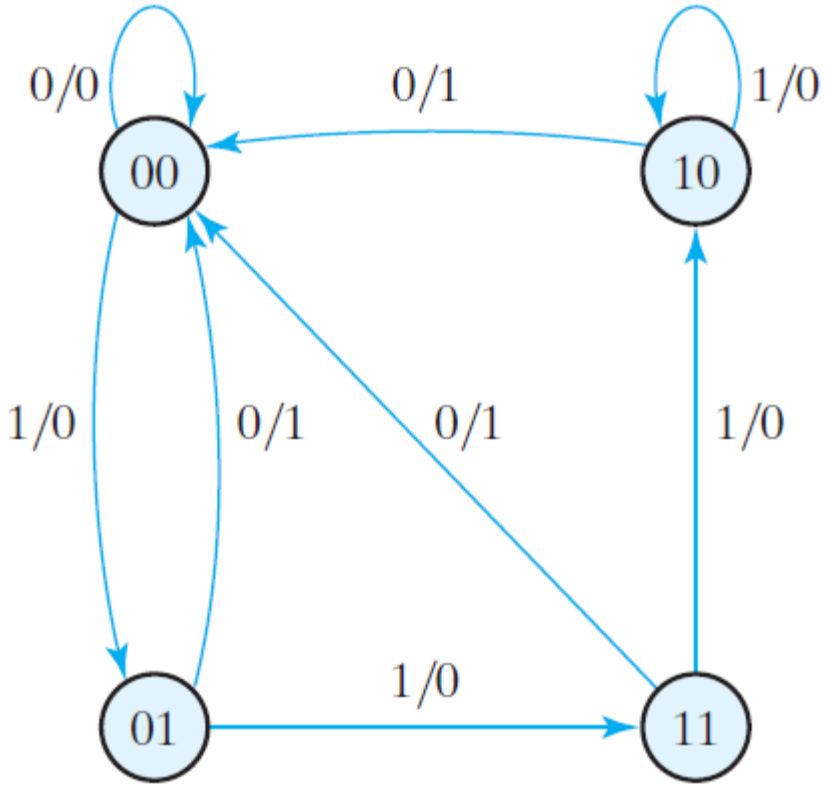


Block diagrams of Mealy and Moore state machines

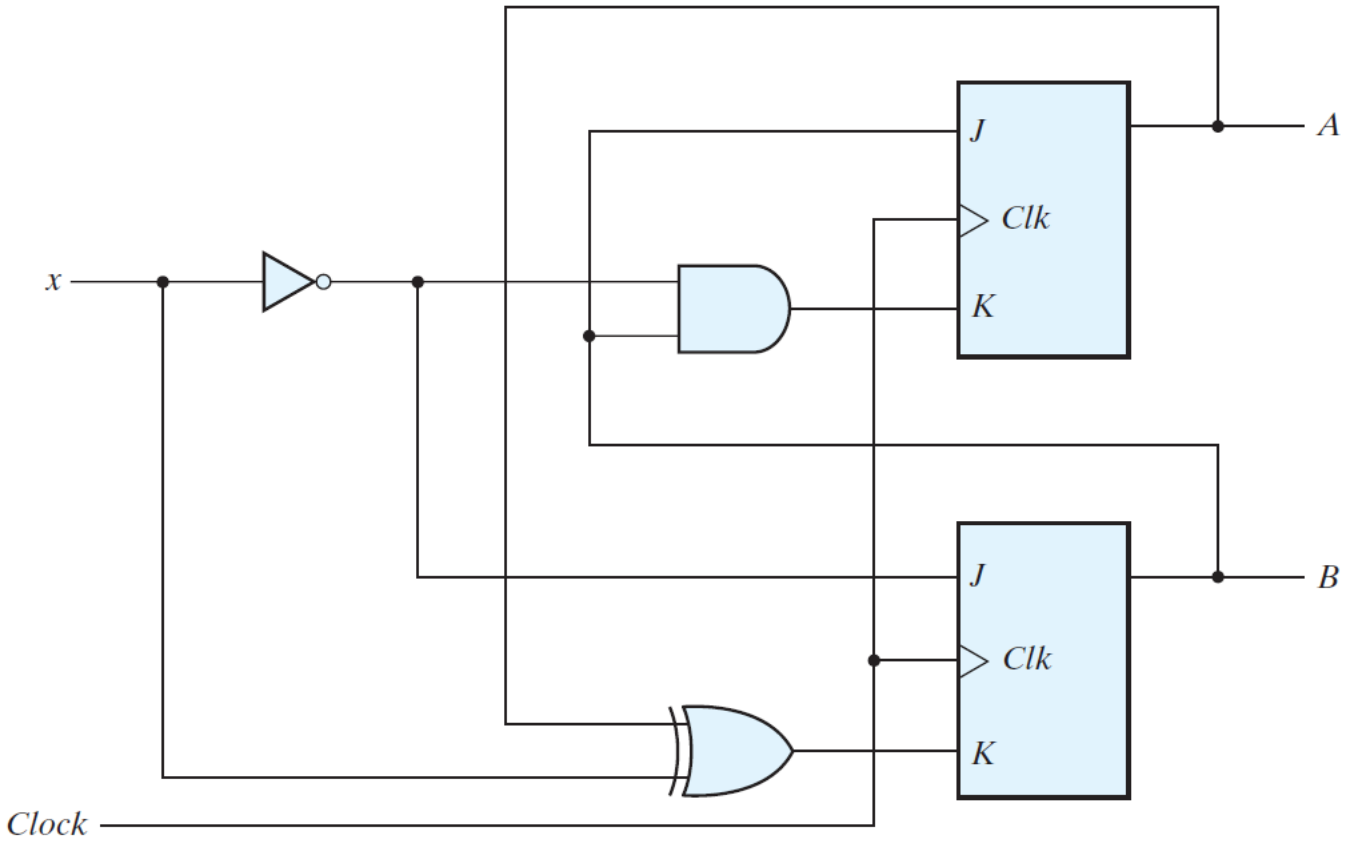
Mealy FSM



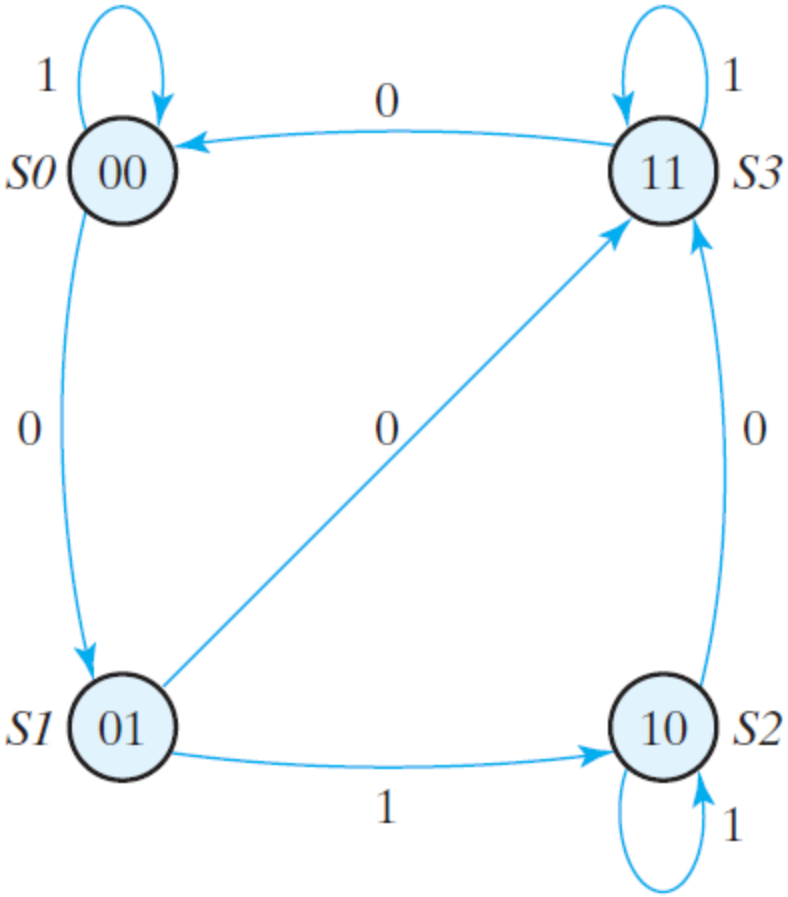
Mealy FSM



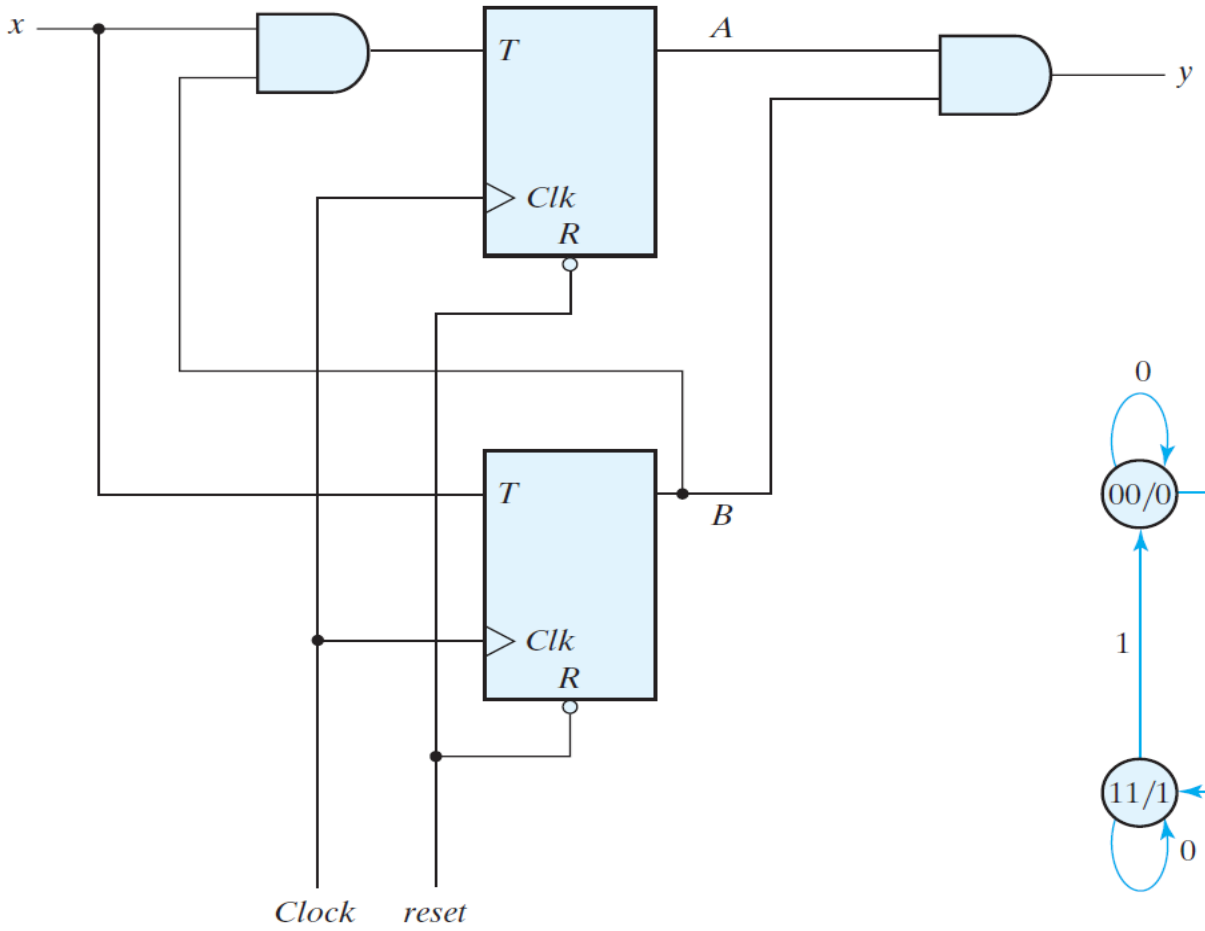
Moore FSM



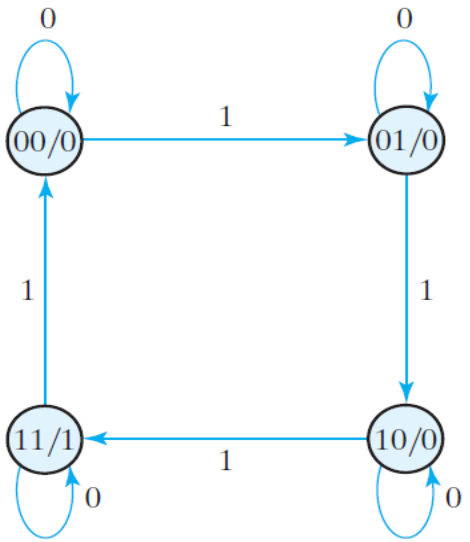
Moore FSM



Moore FSM



(a) Circuit diagram



(b) State diagram

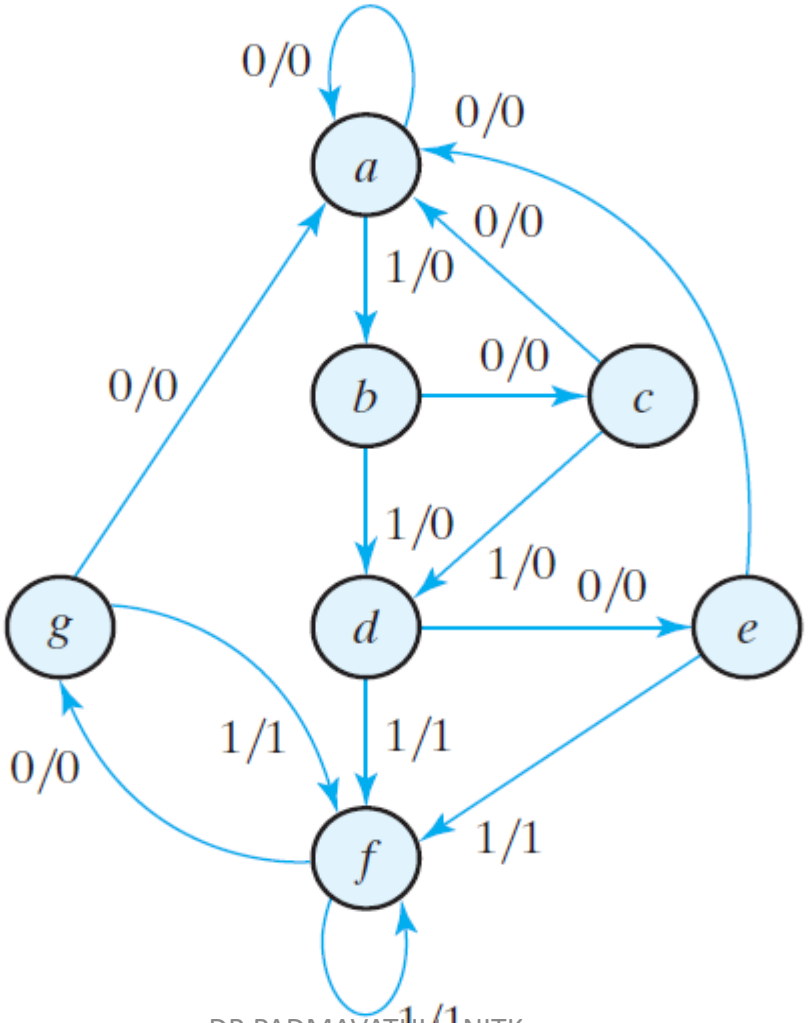
State reduction

State reduction

- Reduction in number of flip-flops in a sequential circuit – state reduction.

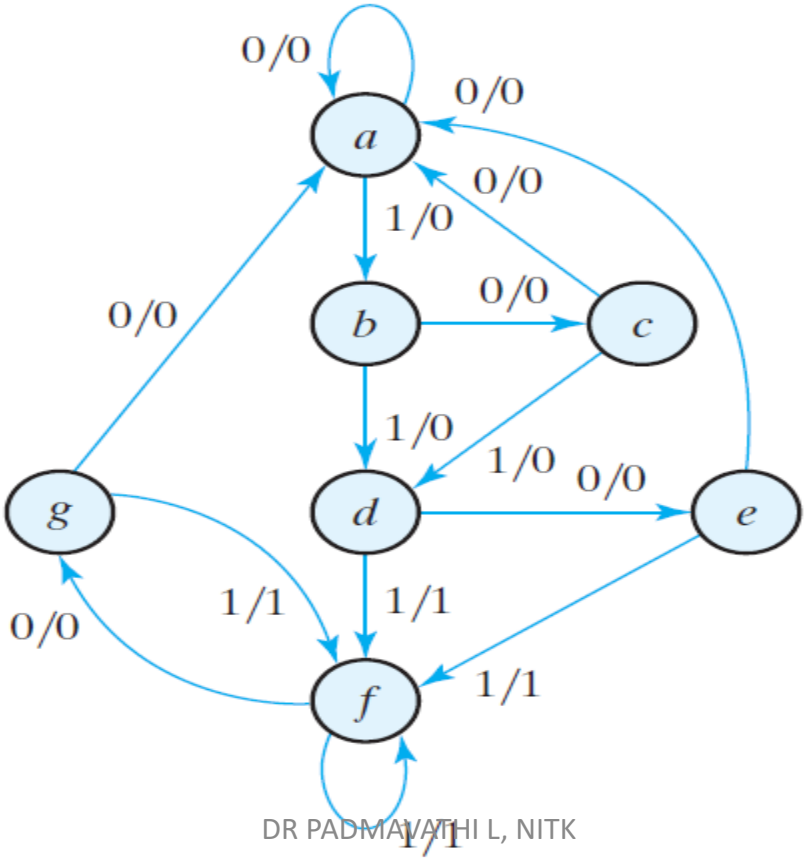
- Reduces the number of states from a state table – keeping the external input output requirements unchanged.

State reduction procedure - example



State reduction procedure - example

state	***a***	***a***	***b***	***c***	***d***	***e***	***f***	***f***	***g***	***f***	***g***	***a***
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	



State reduction procedure - example

state	***a***	***a***	***b***	***c***	***d***	***e***	***f***	***f***	***g***	***f***	***g***	***a***
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

- Output and state sequence for a given input sequence.
- Top of the next column – indicates next state.
- Problem of state reduction – finding ways to reduce the number of states in a sequential circuit without altering input/output relationships.

State reduction procedure - example

state	***a***	***a***	***b***	***c***	***d***	***e***	***f***	***f***	***g***	***f***	***g***	***a***
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

State Table

	Next State		**Output**	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*g*	*f*	0	1
g	*a*	*f*	0	1

State reduction procedure - example

- “Two states are said to be equivalent if, for each member of the set of inputs, they give exactly the same output and send the circuit either to the same state or to an equivalent state.”

- When two states are equivalent, one of them can be removed without altering the input–output relationships.

State reduction procedure - example

State Table

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

- e and g – same next state , same output for both input combinations.
- g and e – equivalent states.

State reduction procedure - example

Reducing the State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

- Row with present state g removed
- State g – replaced with state e each time it occurs in next state column.

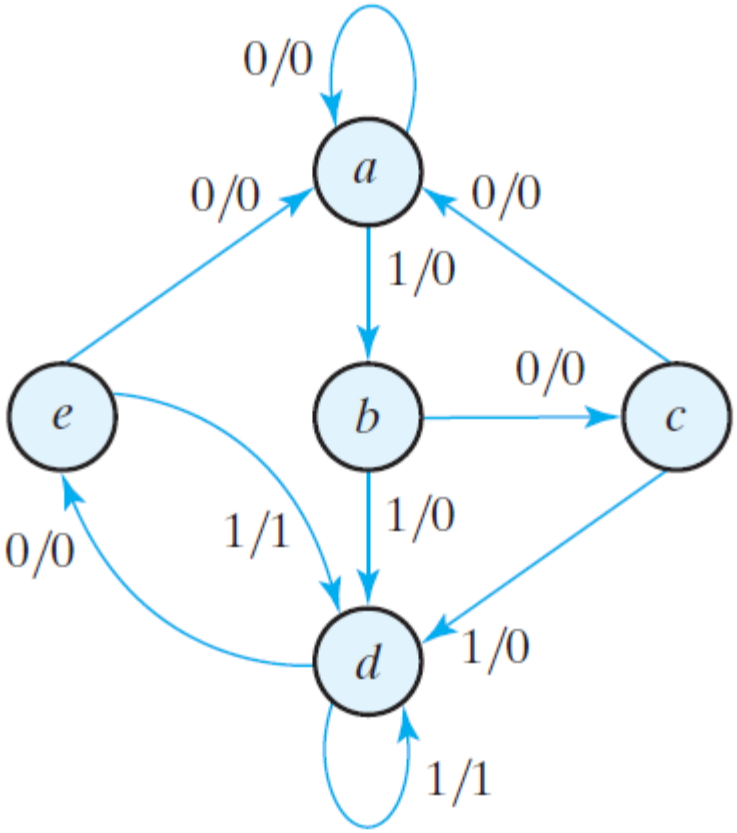
State reduction procedure - example

Reduced State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

Final reduced state table – 5 states.

State reduction procedure - example



Final reduced state diagram

Design procedure for sequential circuits

- Design starts from set of specifications – logic diagram or a list of Boolean functions.
- Truth table – combinational circuit, state table – sequential circuit.
- Design specification – state table or state diagram – choose the flip-flops + required combinational circuits.
- No.of flip-flops – no. of states needed.
- Derive Combinational circuit – by the evaluation of flip-flop input –output equations.

Design procedure for sequential circuits

1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.

2. Reduce the number of states if necessary.

3. Assign binary values to the states.

4. Obtain the binary-coded state table.

5. Choose the type of flip-flops to be used.

6. Derive the simplified flip-flop input equations and output equations.

7. Draw the logic diagram.

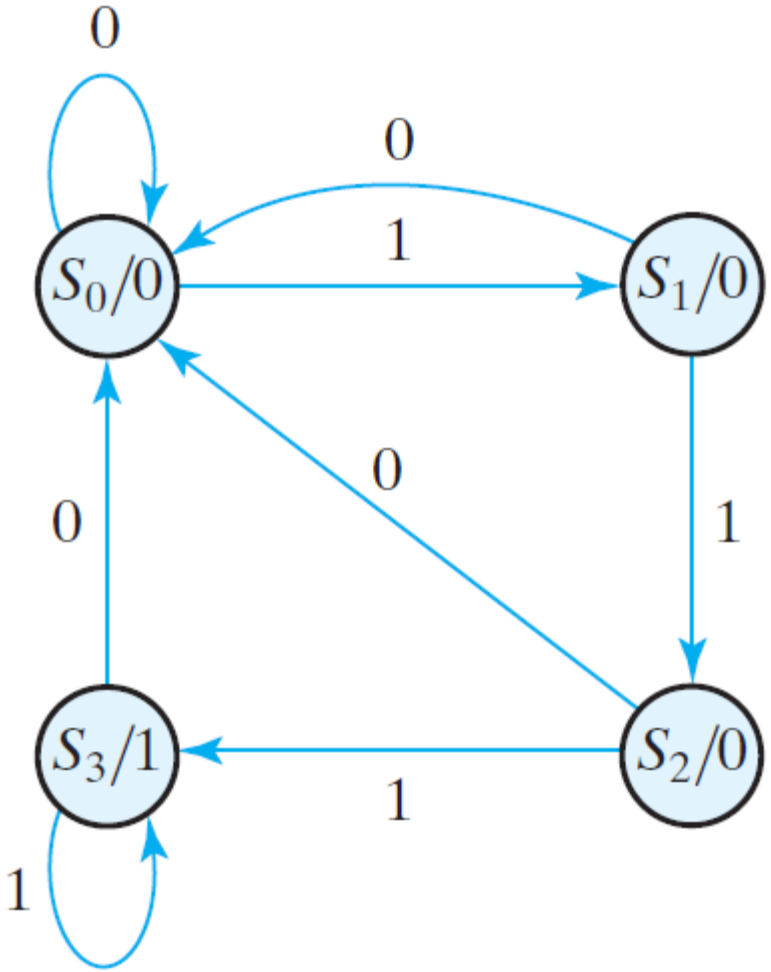
Design procedure for sequential circuits

- Logic synthesis tools (software) – develop HDL description from state diagram.
- Synthesis tools – determine the circuit elements and structure – implements the description.
- Automated synthesis tools in industries – designing massive integrated circuits.

State diagram from word specification - example

Design a circuit that detects a sequence of three or more consecutive 1's in a string of bits coming through an input line..

State diagram from word specification - example



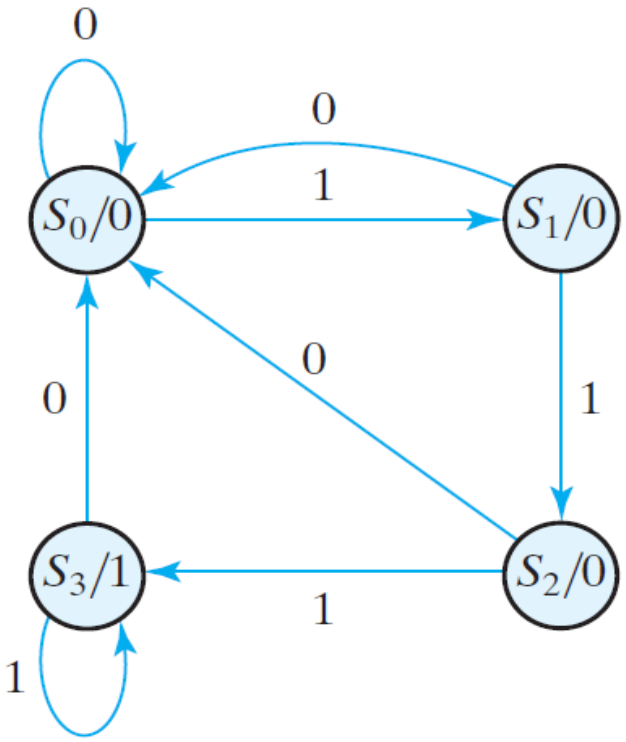
State diagram from word specification - example

State diagram – HDL tools – manual design.

Next step – state table – state assignment.

Four states – two binary variables – implement the design with D flip-flops.

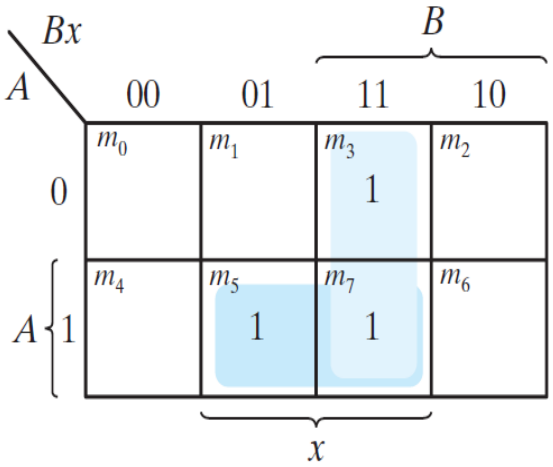
State diagram from word specification - example



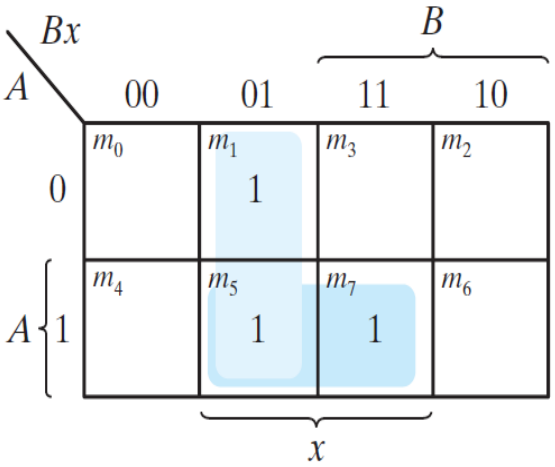
State Table for Sequence Detector

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

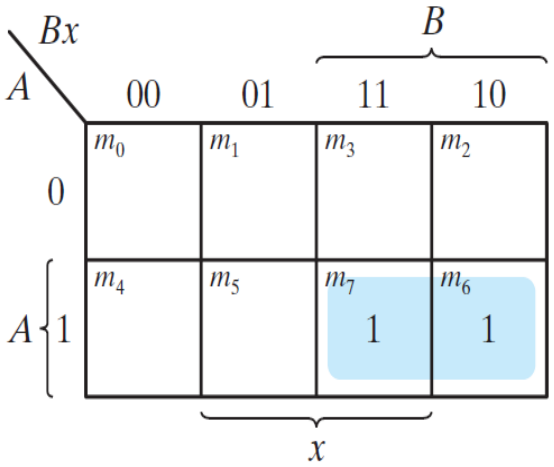
State diagram from word specification - example



$D_A = Ax + Bx$



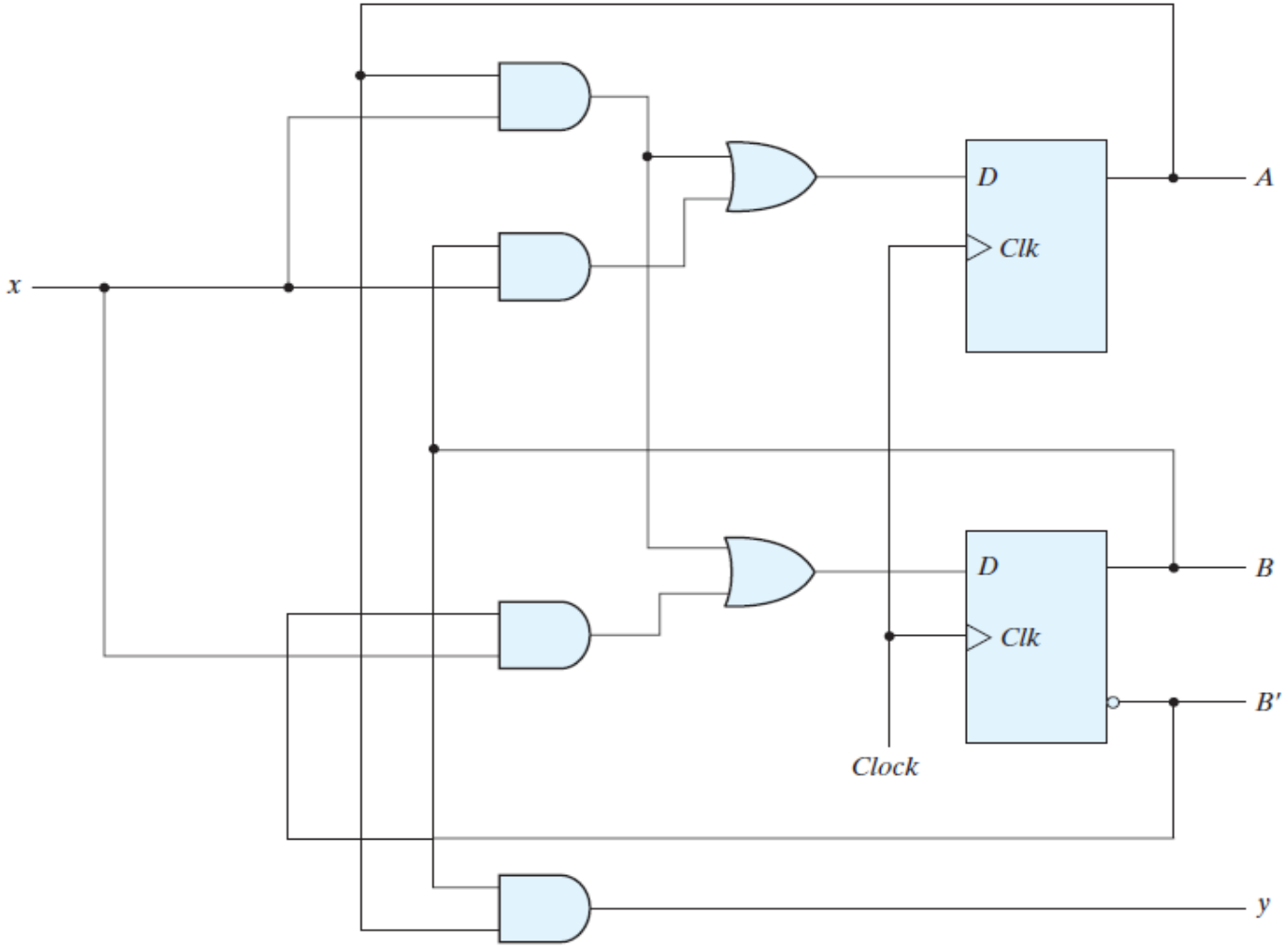
$D_B = Ax + B'x$



$y = AB$

K-Maps for sequence detector

State diagram from word specification - example



Logic diagram of a Moore-type sequence detector

Flip-flop characteristic table

Flip-Flop Characteristic Tables

JK Flip-Flop

J	K	Q(t + 1)	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

D Flip-Flop

D	Q(t + 1)	
0	0	Reset
1	1	Set

T Flip-Flop

T	Q(t + 1)	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

Flip-flop excitation table

- Characteristic table useful - analyzing the next state when the present state and input known.

- In design process – present state to next state transition is known – need to identify the flip-flop input conditions making required transition.

Flip-Flop Excitation Tables

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

(a) JK Flip-Flop

$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

(b) T Flip-Flop

Design with JK flip-flop

State Table and JK Flip-Flop Inputs

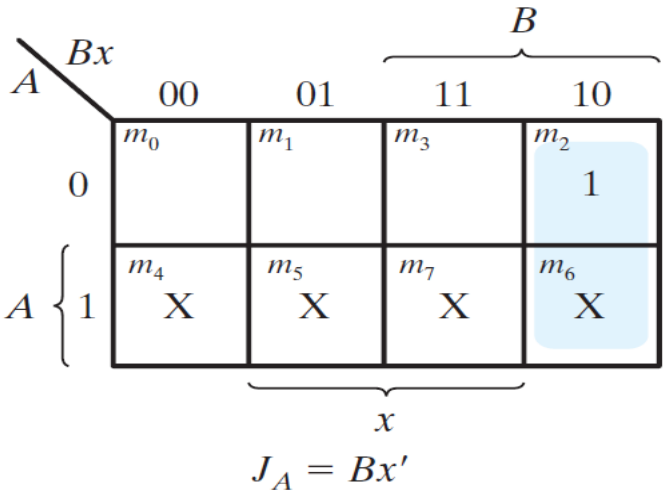
Present State		Input	Next State	
A	*B*	*x*	*A*	*B*
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

Design with JK flip-flop

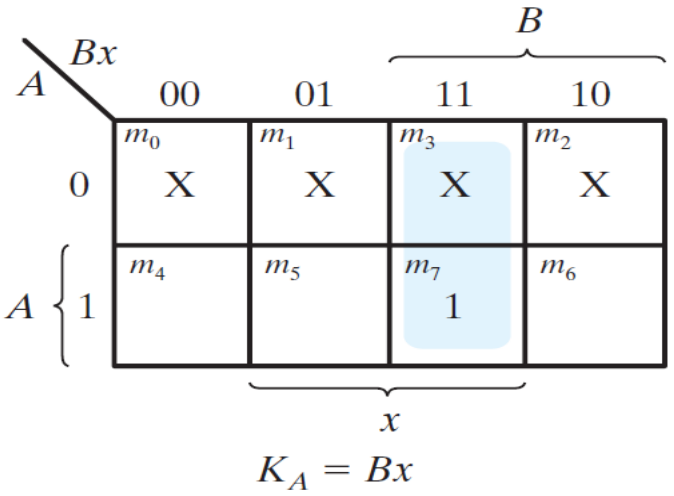
State Table and JK Flip-Flop Inputs

Present State		Input	Next State		Flip-Flop Inputs			
A	*B*	*x*	*A*	*B*	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

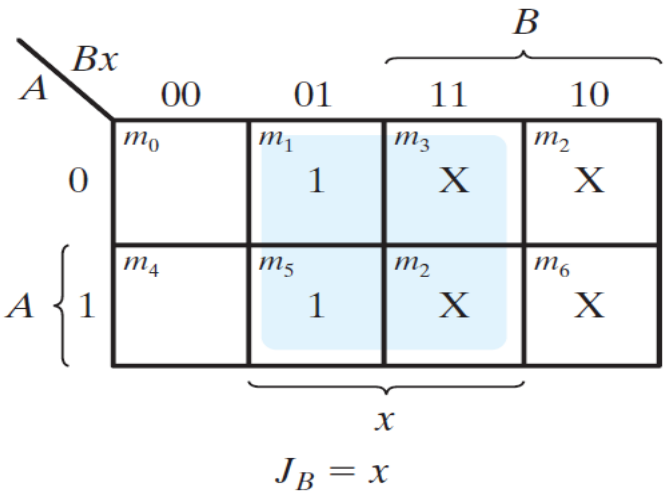
Design with JK flip-flop



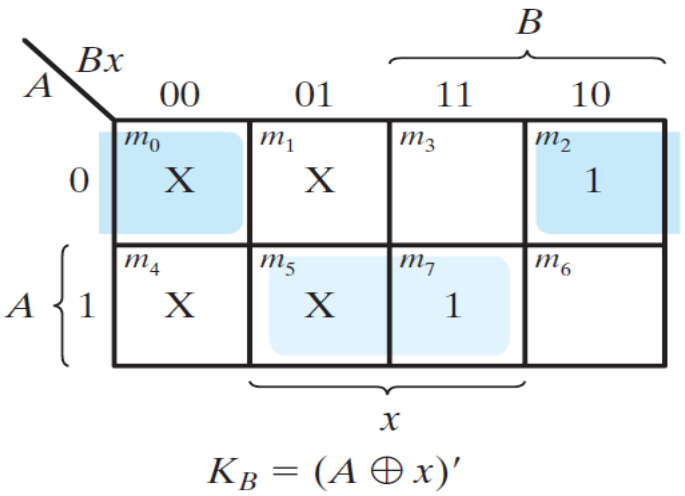
$J_A = Bx'$



$K_A = Bx$



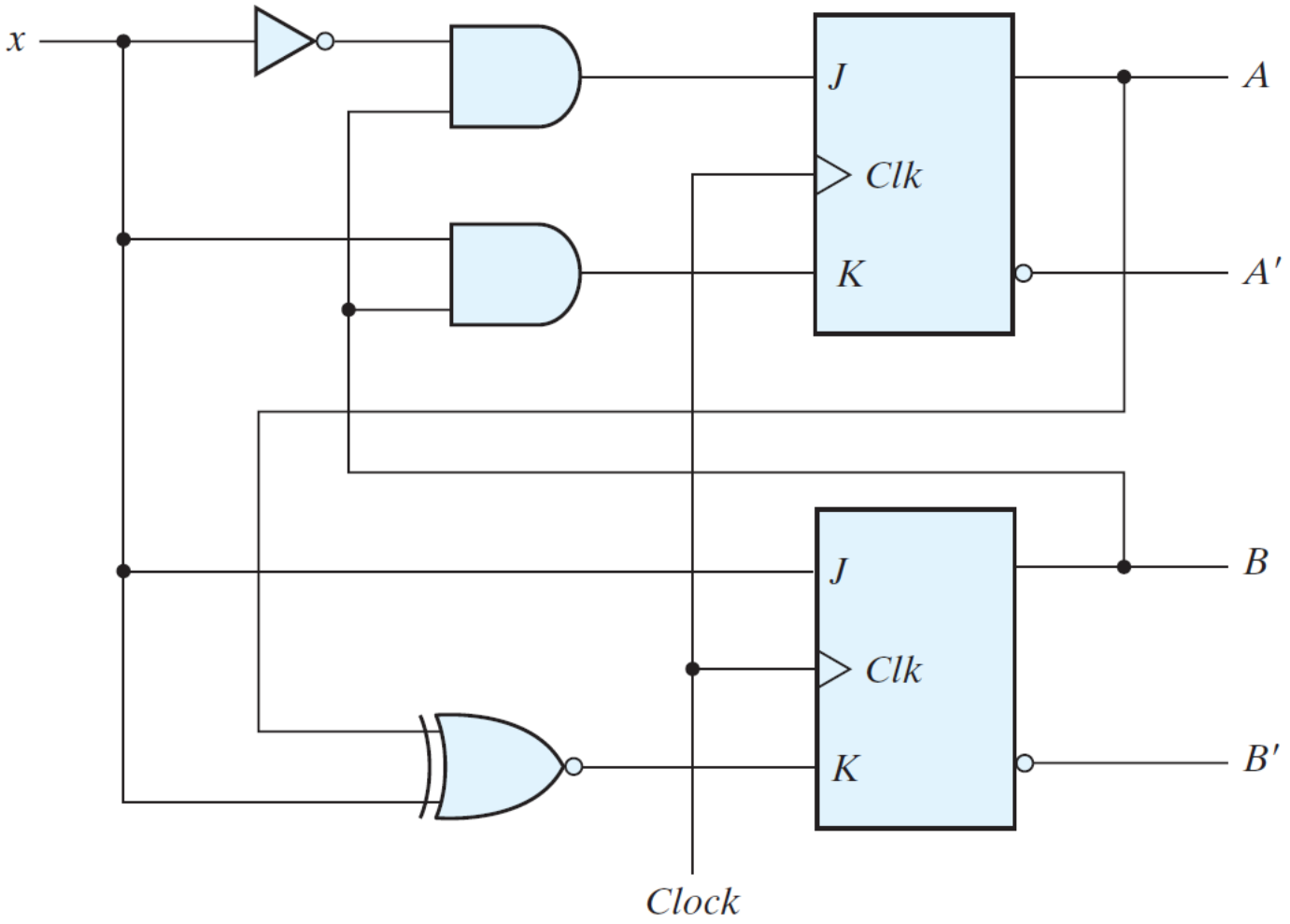
$J_B = x$



$K_B = (A \oplus x)'$

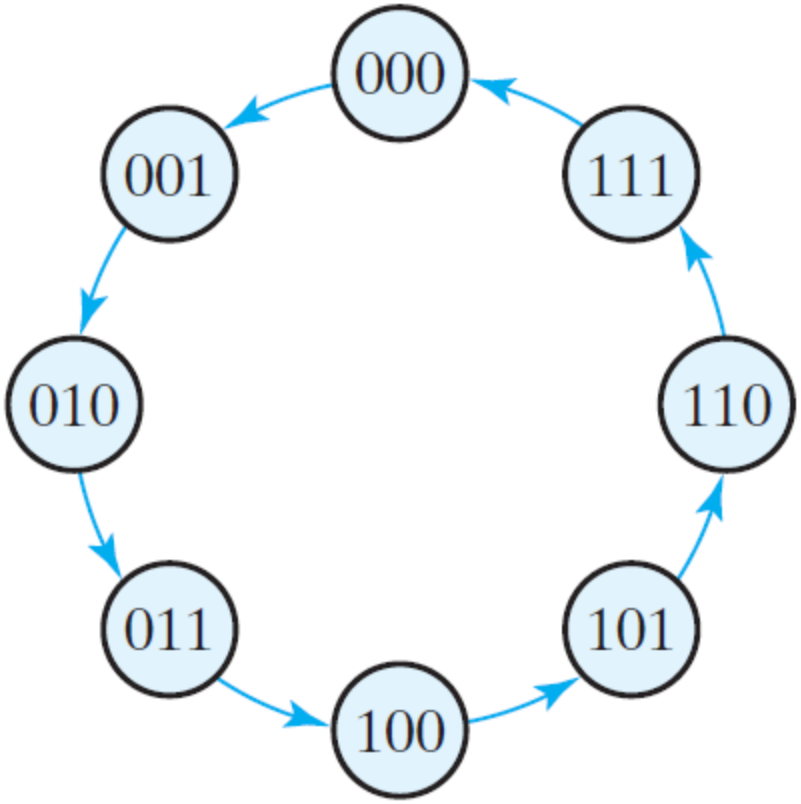
Maps for *J* and *K* input equations

Design with JK flip-flop



Logic diagram for sequential circuit with *JK* flip-flops

Design with T flip-flop



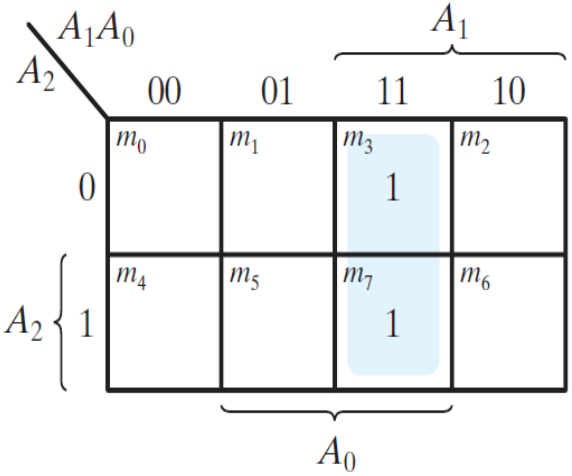
State diagram of three-bit binary counter

Design with T flip-flop

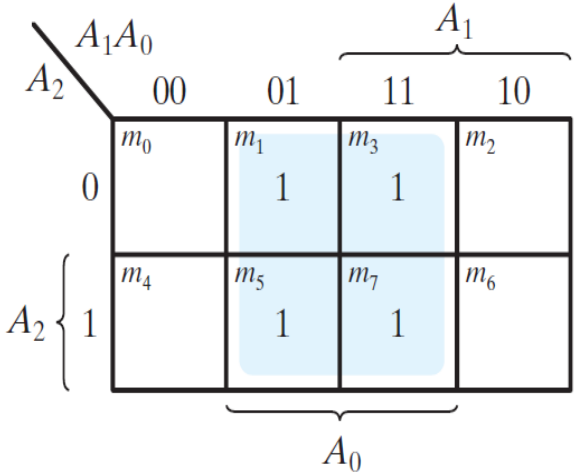
State Table for Three-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	0	1	1	1

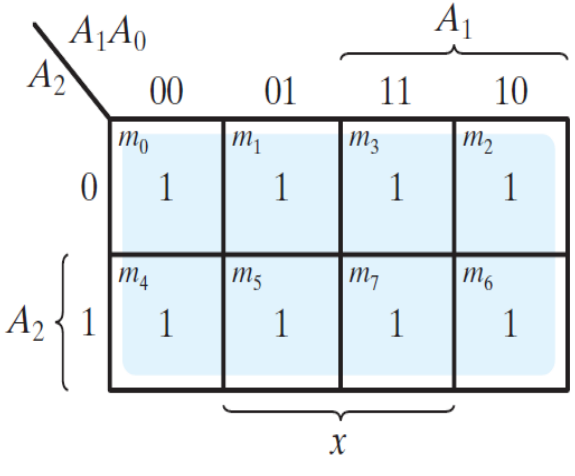
Design with T flip-flop



$T_{A2} = A_1A_0$



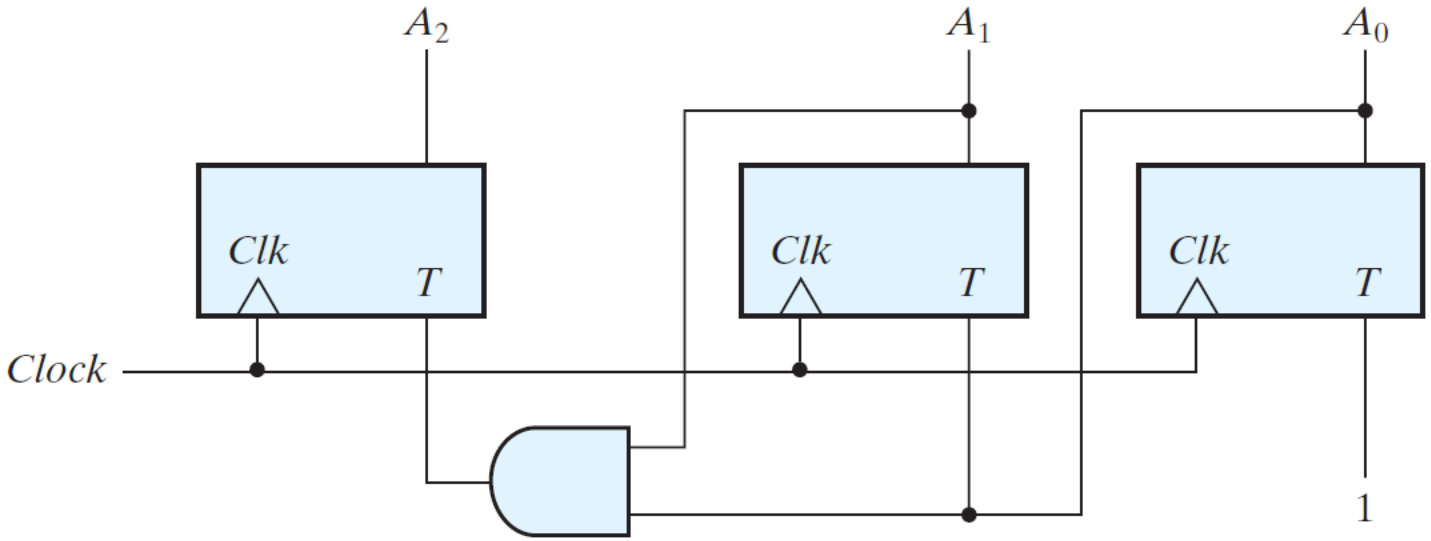
$T_{A1} = A_0$



$T_{A0} = 1$

Maps for three-bit binary counter

Design with T flip-flop

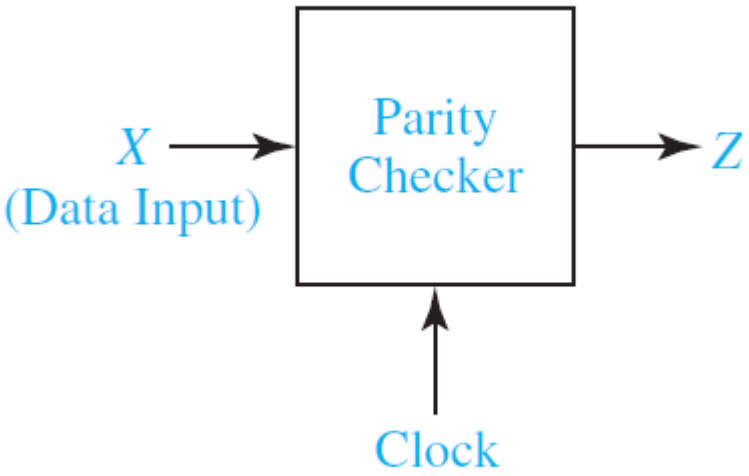


Logic diagram of three-bit binary counter

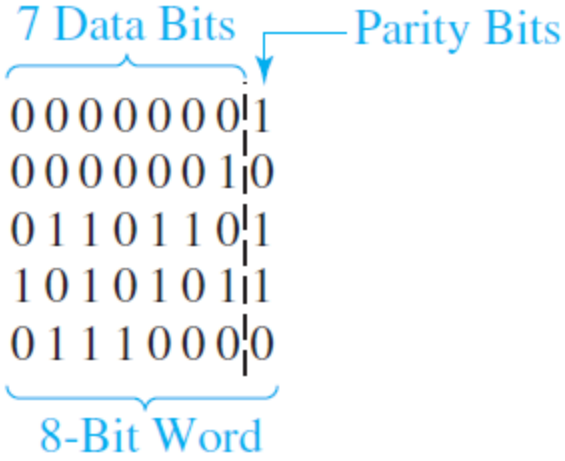
Timing diagrams

- Timing relationship between the inputs, outputs and clock for sequential circuits.
- Timing diagrams important –sequential circuits used as a part of larger digital system.
- State change always occurs – in response to active clock edge.
- Circuit output change – at the time of flip-flops state change / input change – depends the type

Parity checker



Block Diagram
for Parity Checker



Example 0f 8 bit words with odd parity

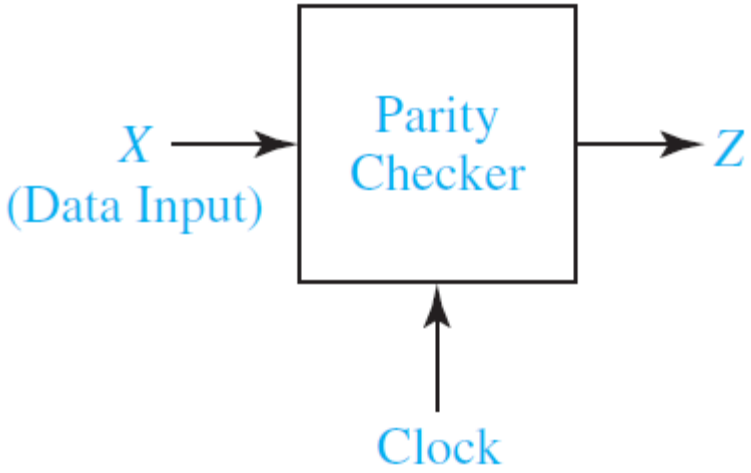
One input, clock

Serial data coming in(data enters the

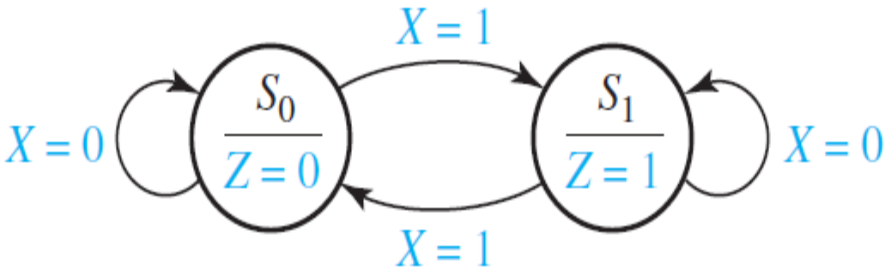
Parity checker

- One input, clock
- Serial data coming in(data enters the circuit sequentially – one bit at a time.)
- Sequence of 0's and 1's applied to the X input.
- Output Z = 1, if the total number of 1 inputs received odd – i.e – input parity odd.
- Output Z = 0, error in transmission, total no.of one received is even.

Parity checker



Block Diagram
for Parity Checker



State Graph for
Parity Checker

Parity checker

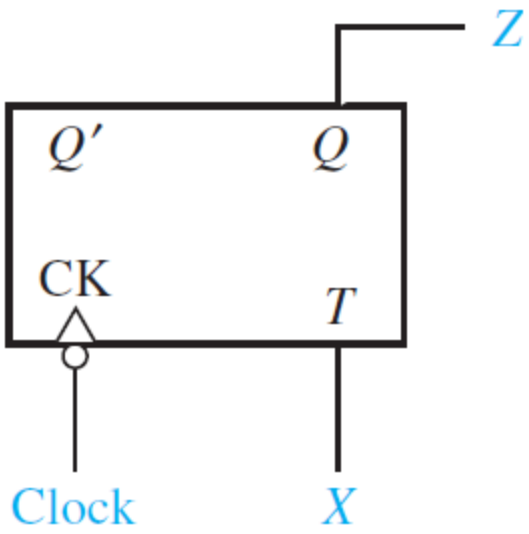
(a)

Present State	Next State $X=0$	Next State $X=1$	Present Output
S_0	S_0	S_1	0
S_1	S_1	S_0	1

(b)

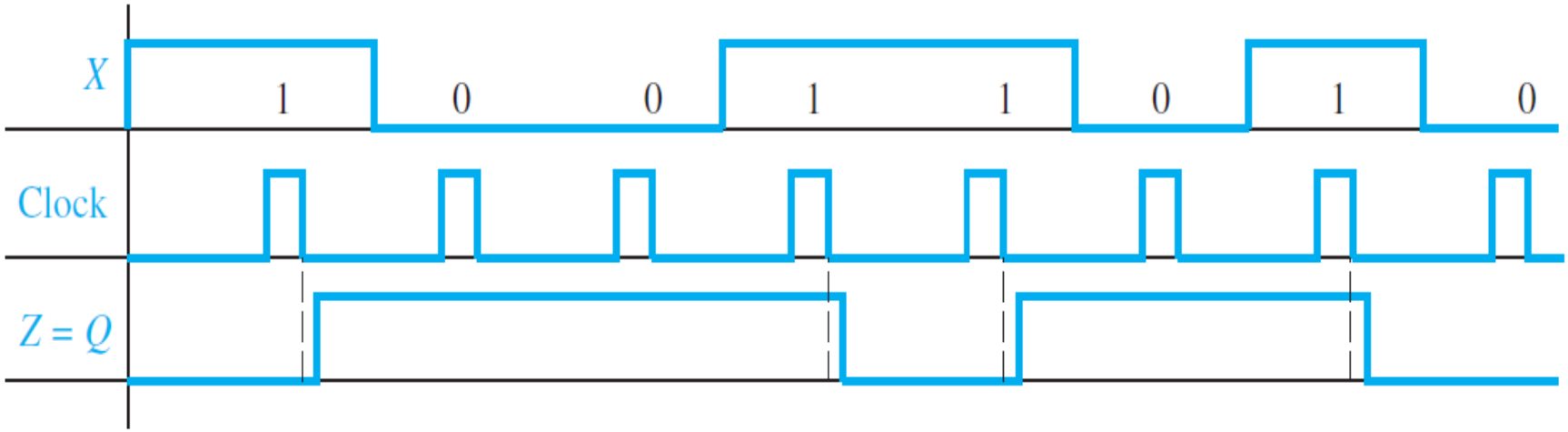
Q	Q^+ $X=0$	Q^+ $X=1$	T $X=0$	T $X=1$	Z
0	0	1	0	1	0
1	1	0	0	1	1

State Table for Parity Checker



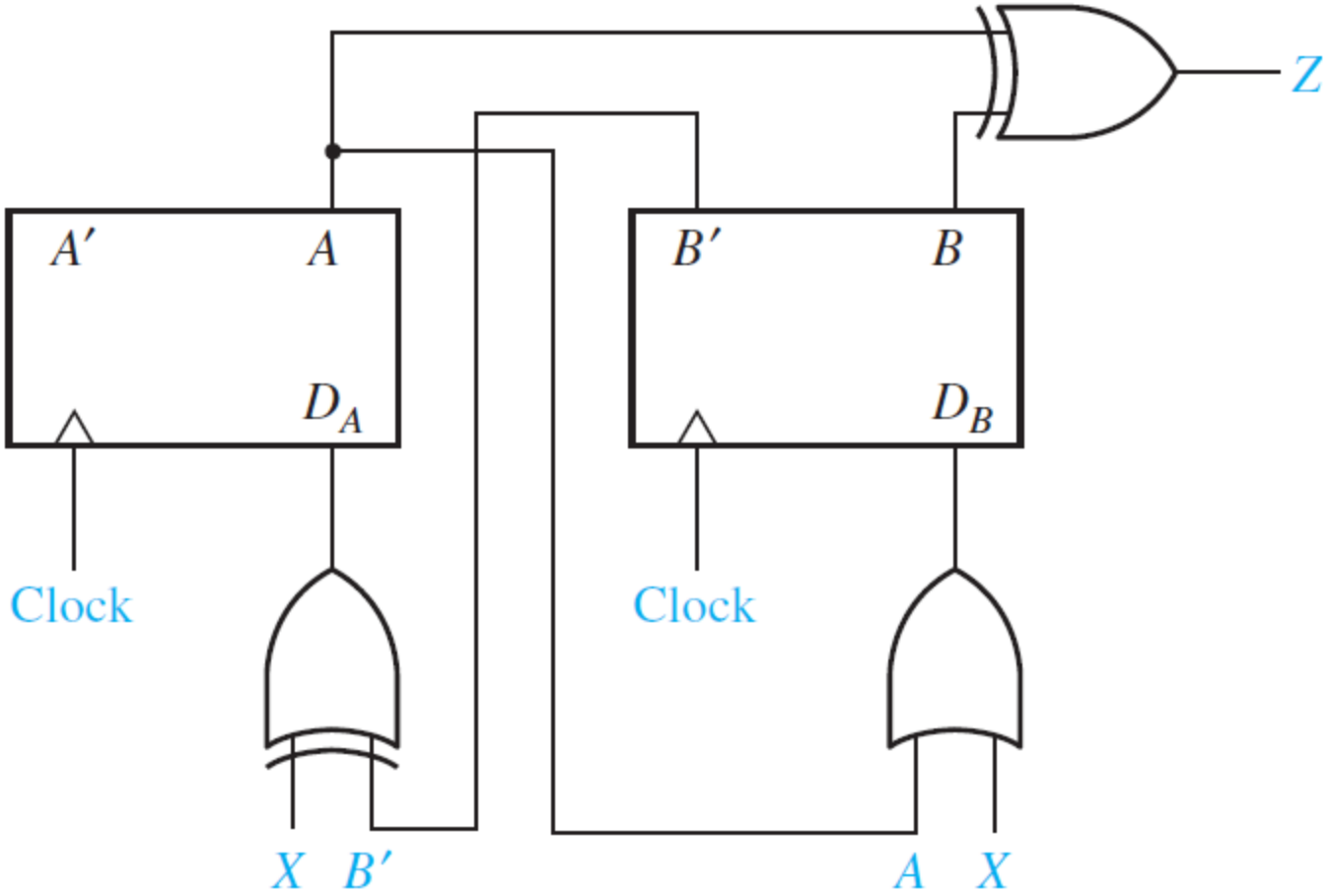
Parity Checker

Parity checker



Waveforms for
Parity Checker

Moore sequential circuit



Timing diagrams

- X input synchronized with clock. Initial state A = B = 0
- State change occurs after the rising edge of the clock.
- Z = A EX-OR B.
- X = 0 D_A = 1 and D_B = 0
- Next state A = 1, B =1 , after first rising edge of the clock.

Timing diagrams

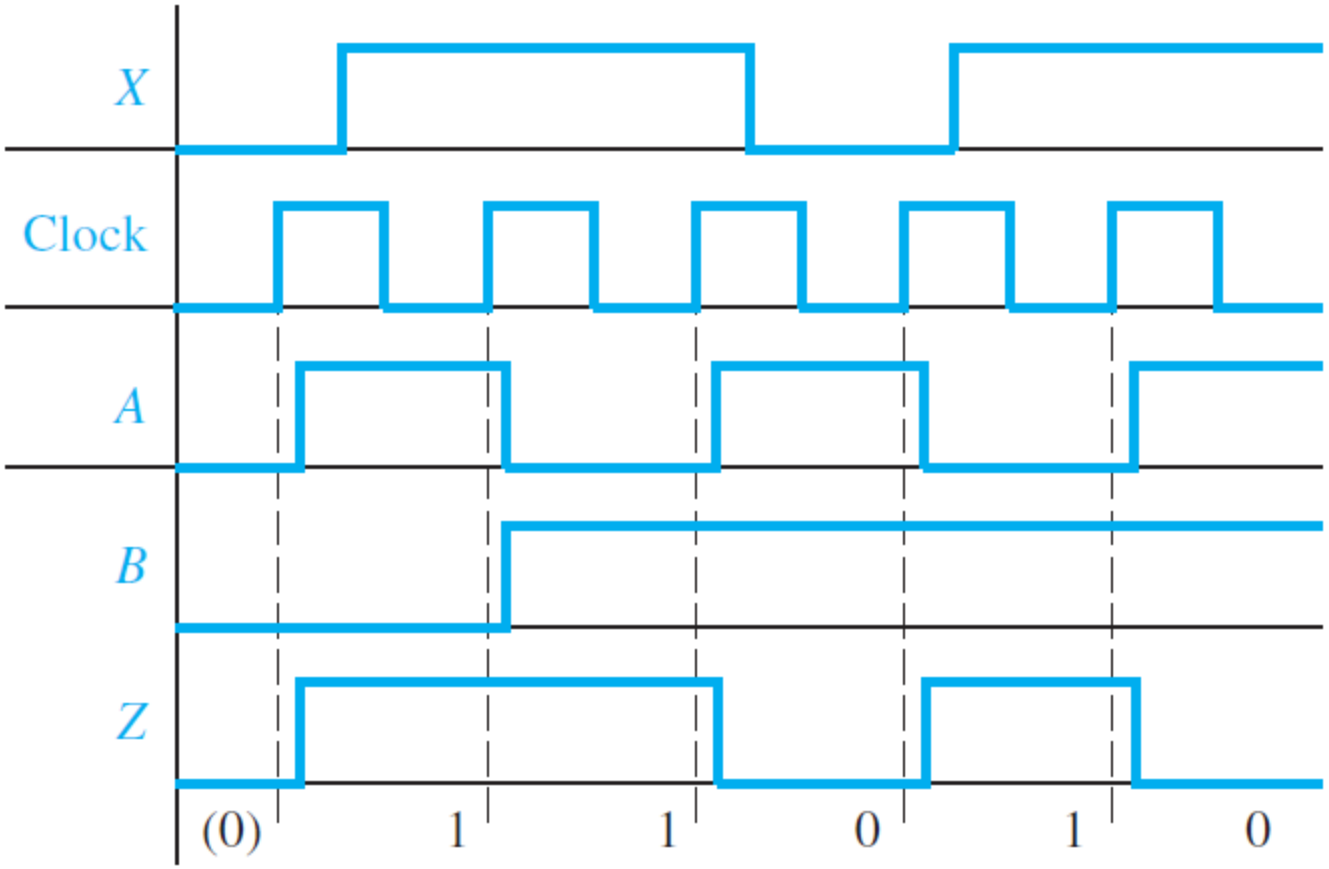
- X = 01101

- A = 0, B = 0, X = 0, Z = 0, $D_A = 1$ and $D_B = 0$
- A = 1, B = 0, X = 1, Z = 1, $D_A = 0$ and $D_B = 1$
- A = 0, B = 1, X = 1, Z = 1 $D_A = 1$ and $D_B = 1$
- A = 1, B = 1, X = 0, Z = 0 $D_A = 0$ and $D_B = 1$
- A = 0, B = 1, X = 1, Z = 1 $D_A = 1$ and $D_B = 1$
- A = 1, B = 1

Timing diagrams

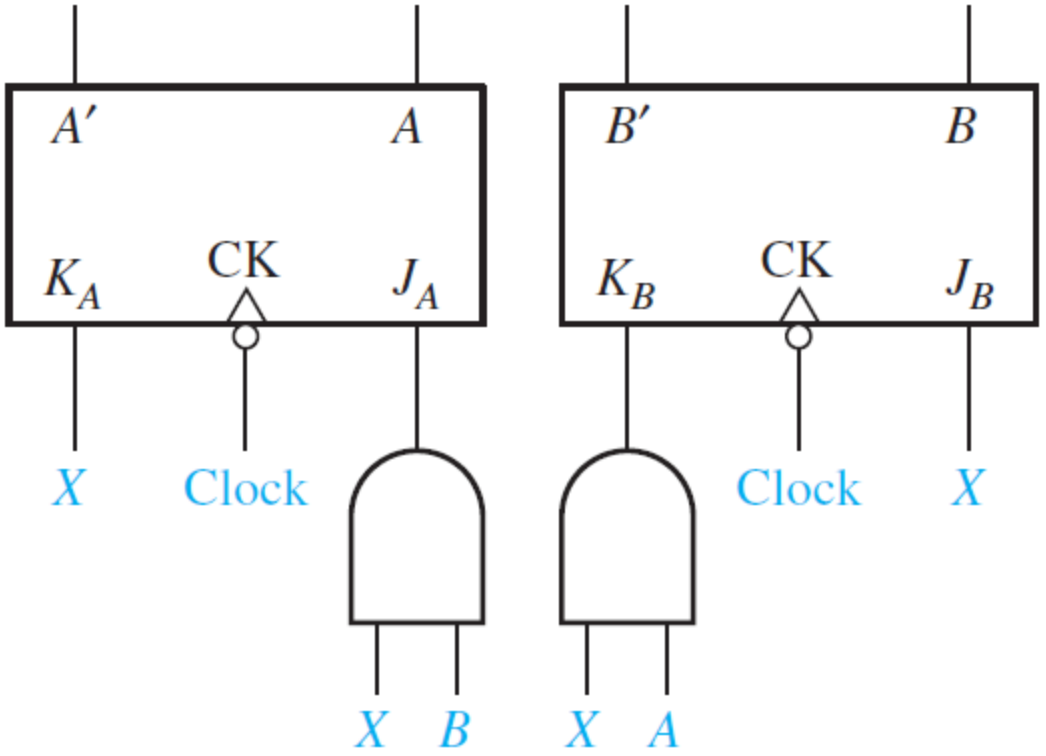
$X =$	0	1	1	0	1	
$A =$	0	1	0	1	0	1
$B =$	0	0	1	1	1	1
$Z =$	(0)	1	1	0	1	0

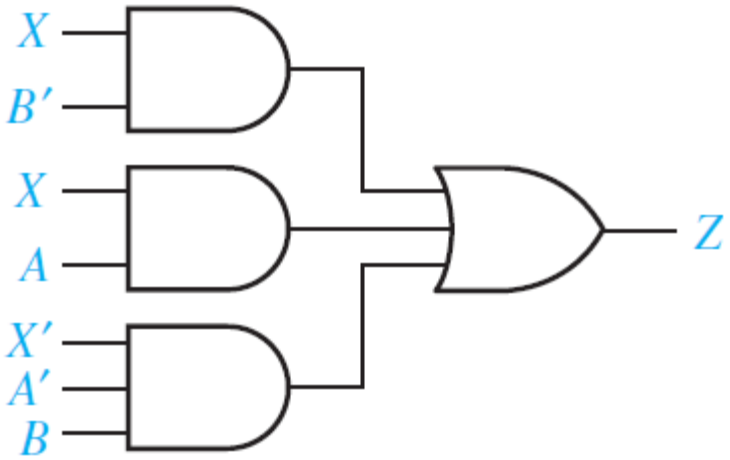
Moore sequential circuit



Timing Chart

Mealy sequential circuit



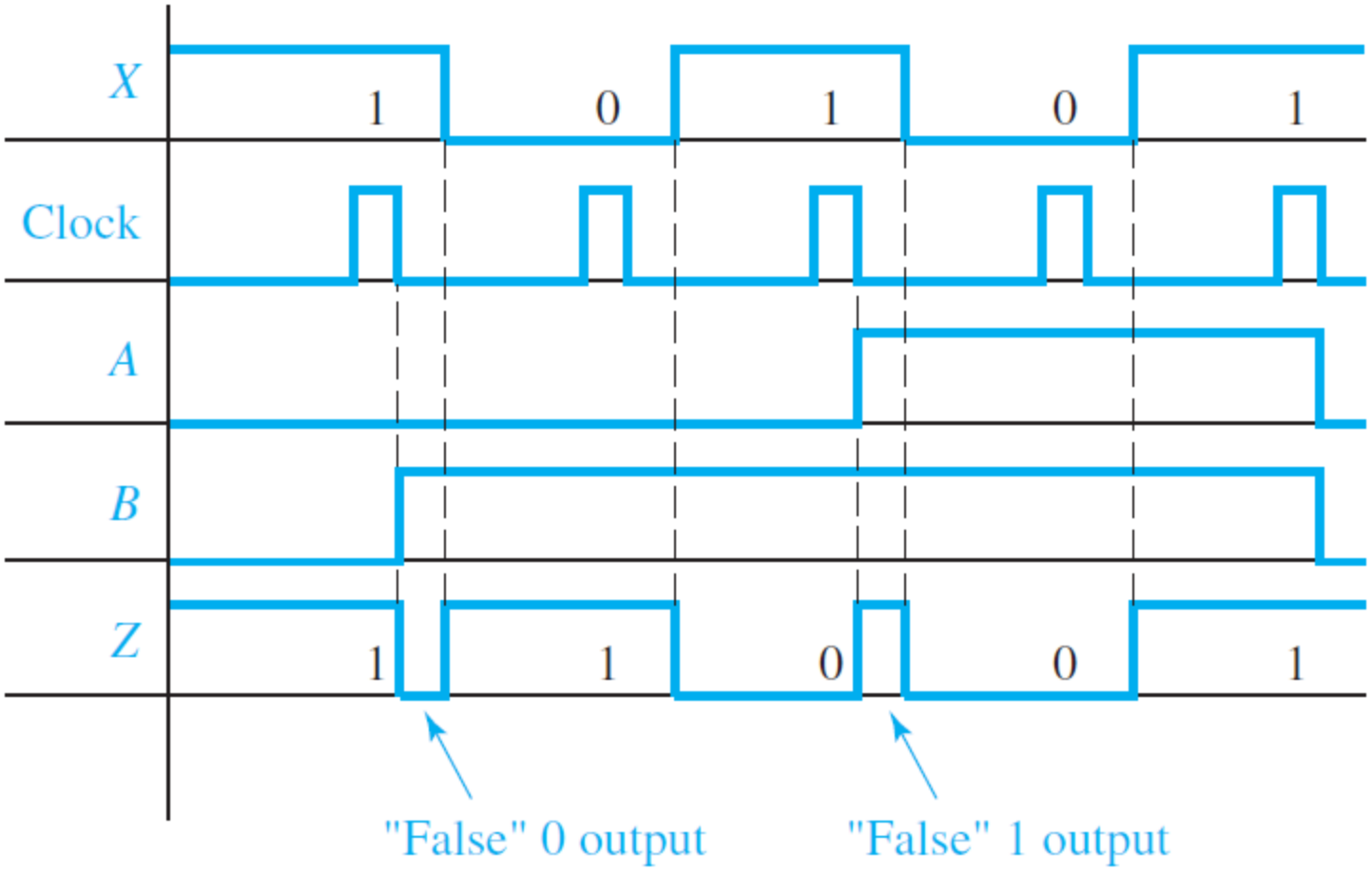


Mealy sequential circuit

$X =$	1		0		1		0	1	
$A =$	0		0		0		1	1	0
$B =$	0		1		1		1	1	0
$Z =$	1(0)		1		0(1)		0	1	

(False outputs are indicated in parentheses.)

Mealy sequential circuit



Timing Chart

Timing diagrams

- Necessary to interpret the output waveform carefully in a Mealy circuit.

- After change in circuit state – before change in input – output temporarily assume an incorrect value – false output.

- False value – circuit changed to new state, but old input associated with the previous state is still present.

Timing diagrams

- Mealy circuit output – only of interest immediately proceeding the active clock edge.
- Extra output changes might occur between active clock edges – should be ignored.
- False outputs – glitches and spikes.
- State change – output change – input has not changed to new value – output value may not be correct – results in false output or glitch.
- Ignoring the false outputs – Z = 11001

Timing diagrams

- Moore circuit – output change only when the flip flop changes state – which is synchronized with the clock – no false outputs or glitches in a Moore circuit.

- Displacement of output sequence w.r.t input sequence – Moore circuit.

Interpretation of timing charts

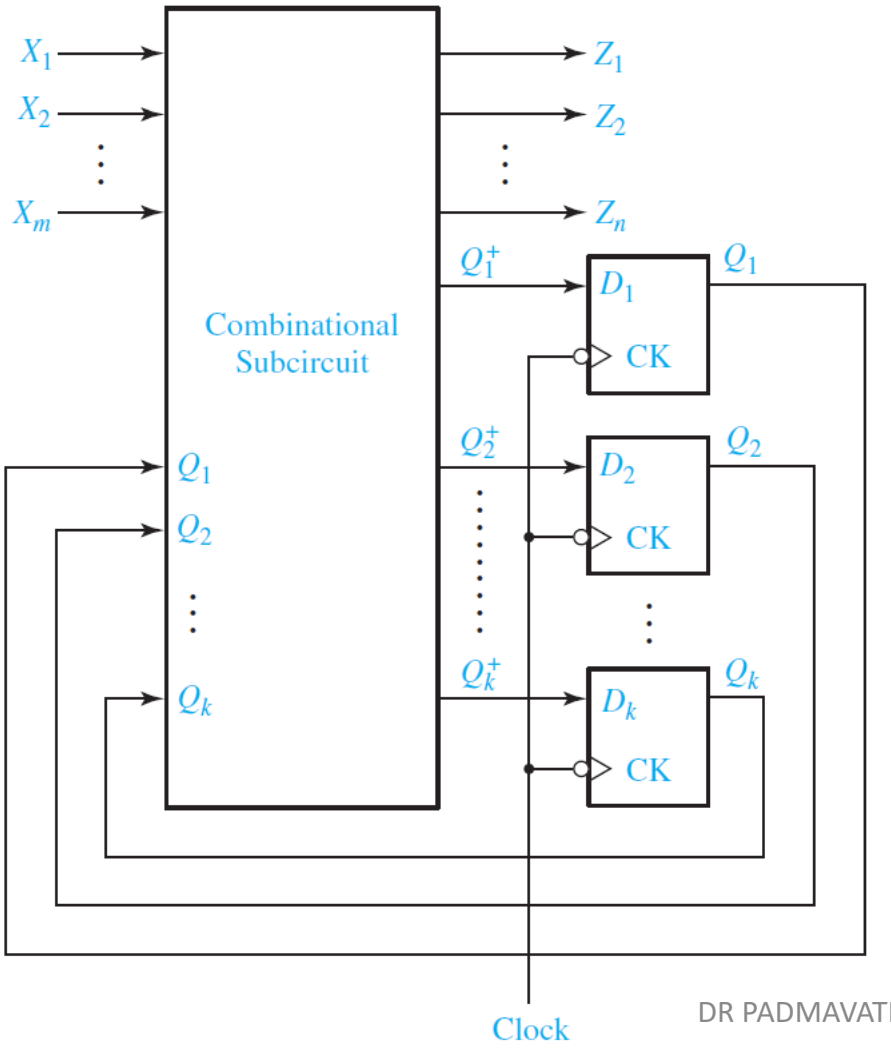
When constructing timing charts, note that a state change can only occur after the rising (or falling) edge of the clock, depending on the type of flip-flop used.

The input will normally be stable immediately before and after the active clock edge.

For a Moore circuit, the output can change only when the state changes, but for a Mealy circuit, the output can change when the input changes as well as when the state changes. A false output may occur between the time the state changes and the time the input is changed to its new value. (In other words, if the state has changed to its next value, but the old input is still present, the output may be temporarily incorrect.)

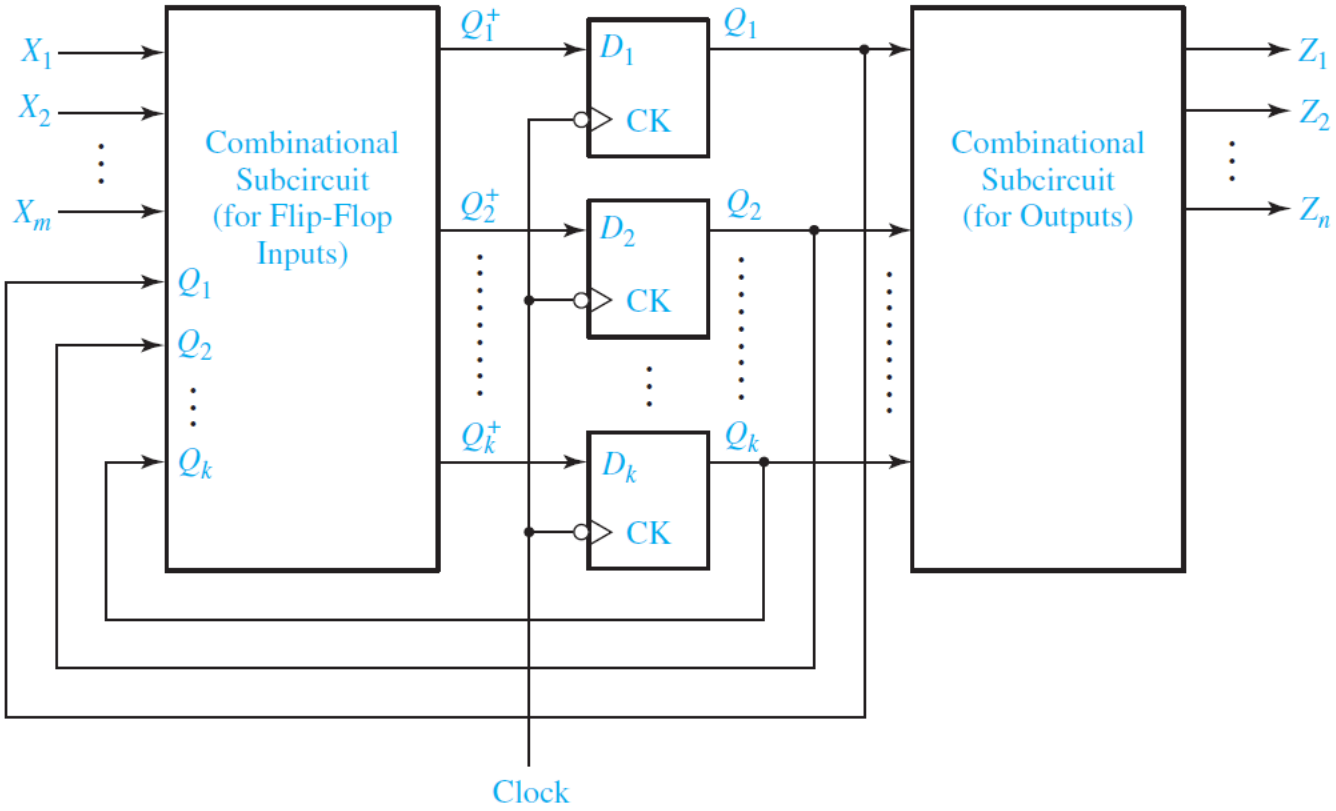
For Mealy circuits, the best time to read the output is just before the active edge of the clock, because the output should always be correct at that time.

Timing diagrams



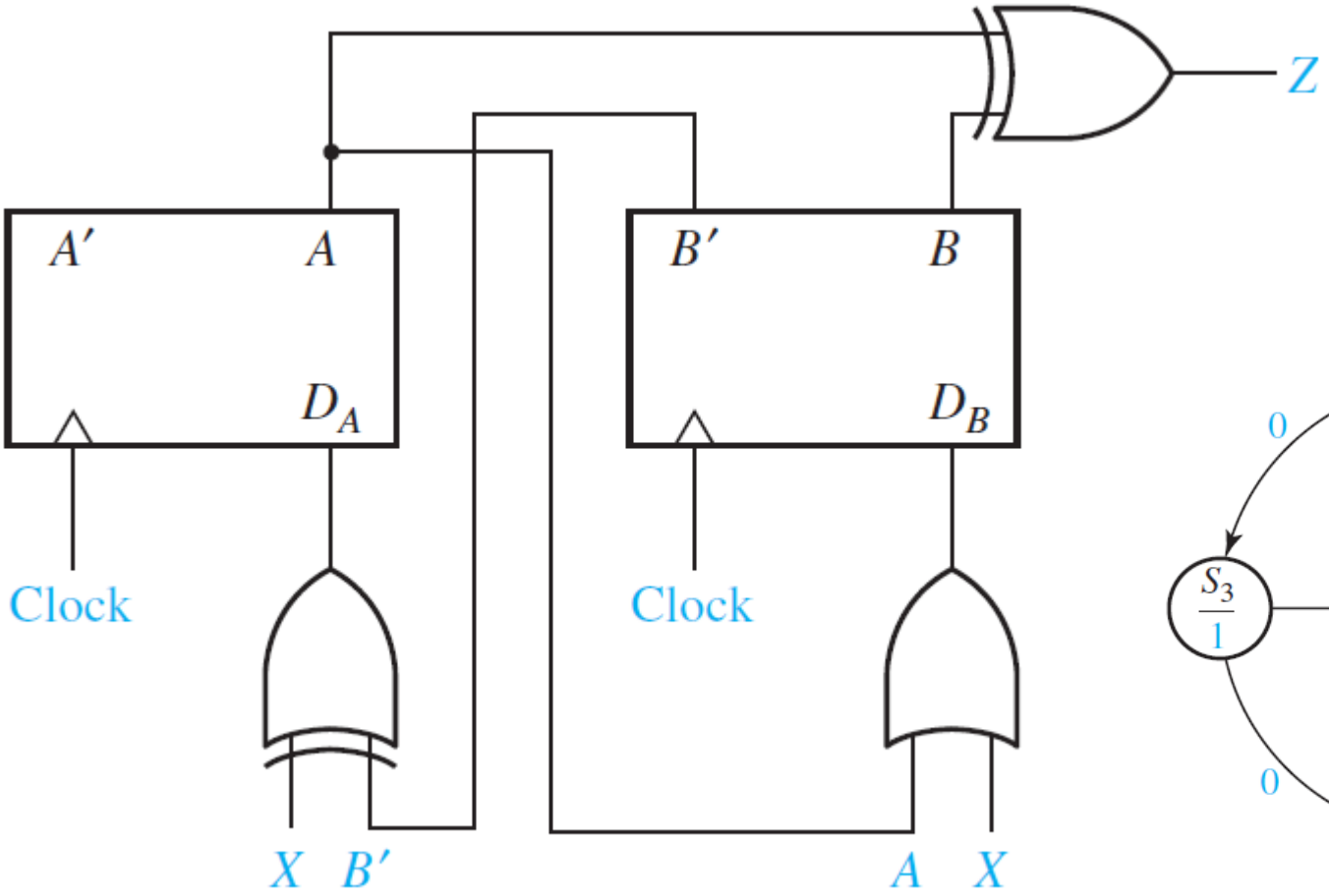
General Model for Mealy Circuit Using Clocked D Flip-Flops

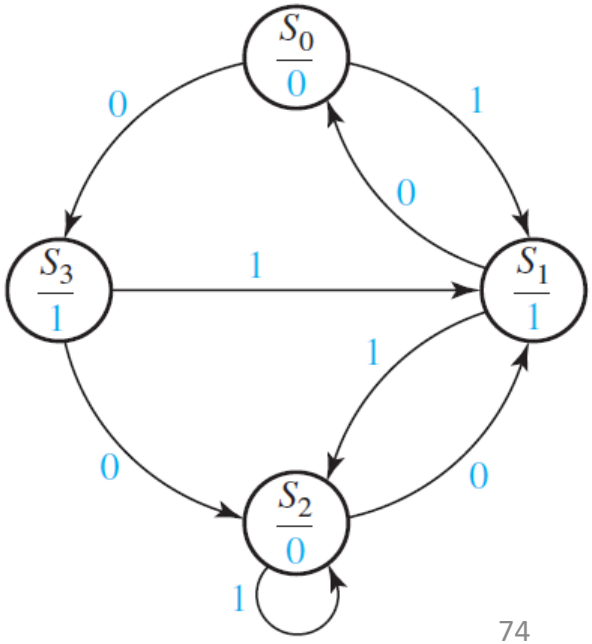
Timing diagrams



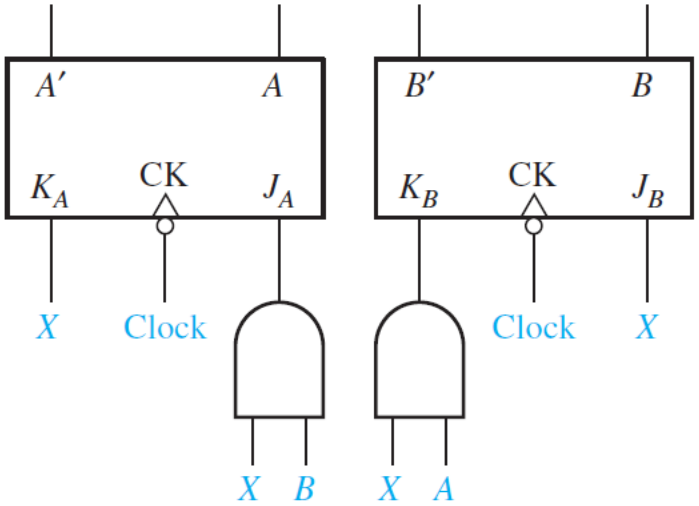
General Model for Moore Circuit Using Clocked D Flip-Flops

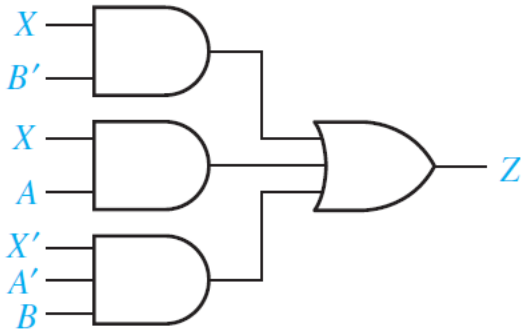
State table and state diagram – practice problem - 1

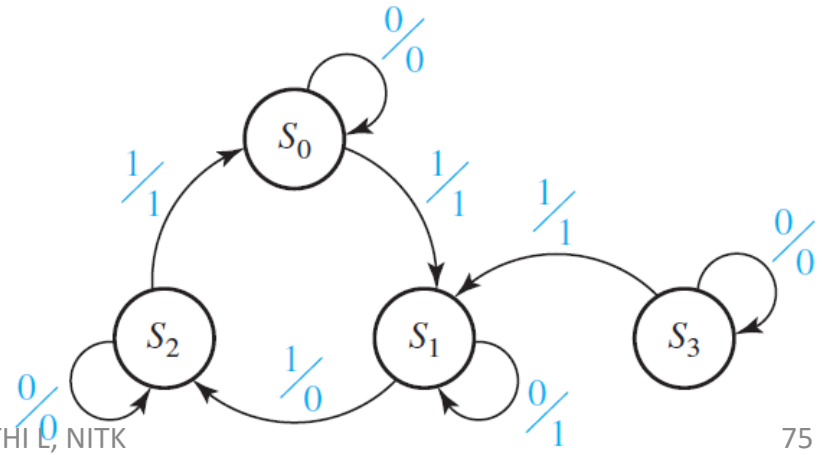




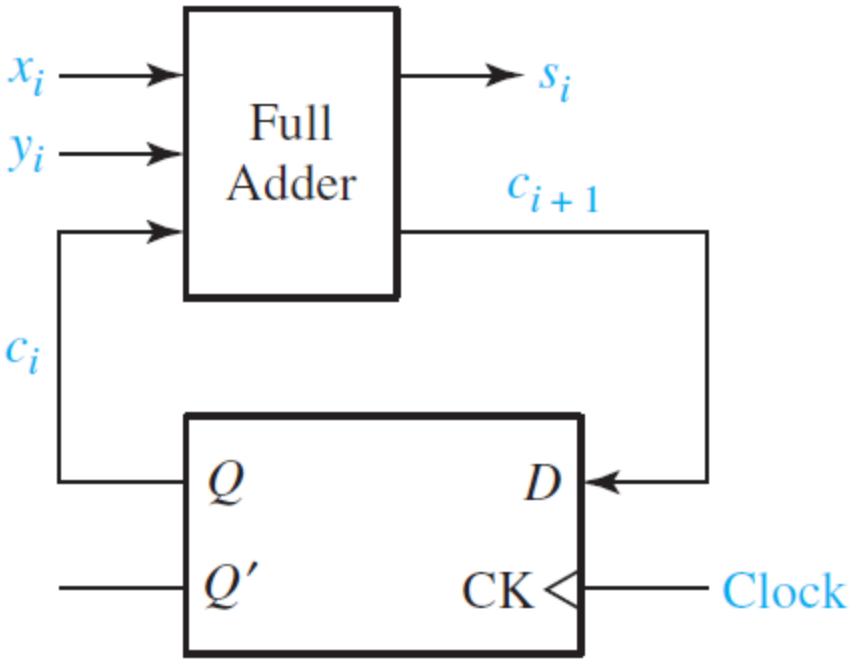
State table and state diagram – practice problem - 2



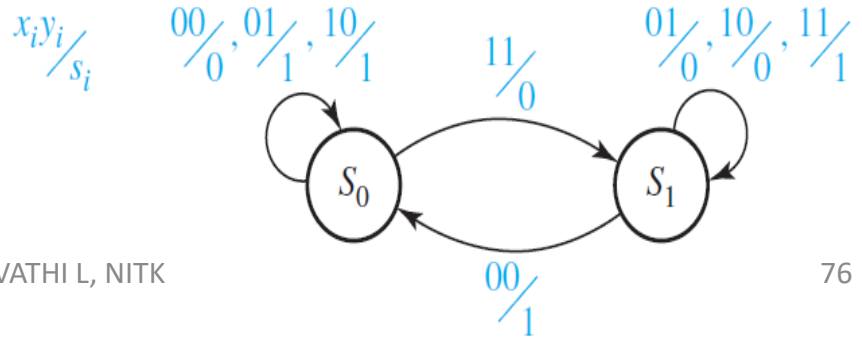




State table and state diagram – practice problem - 3



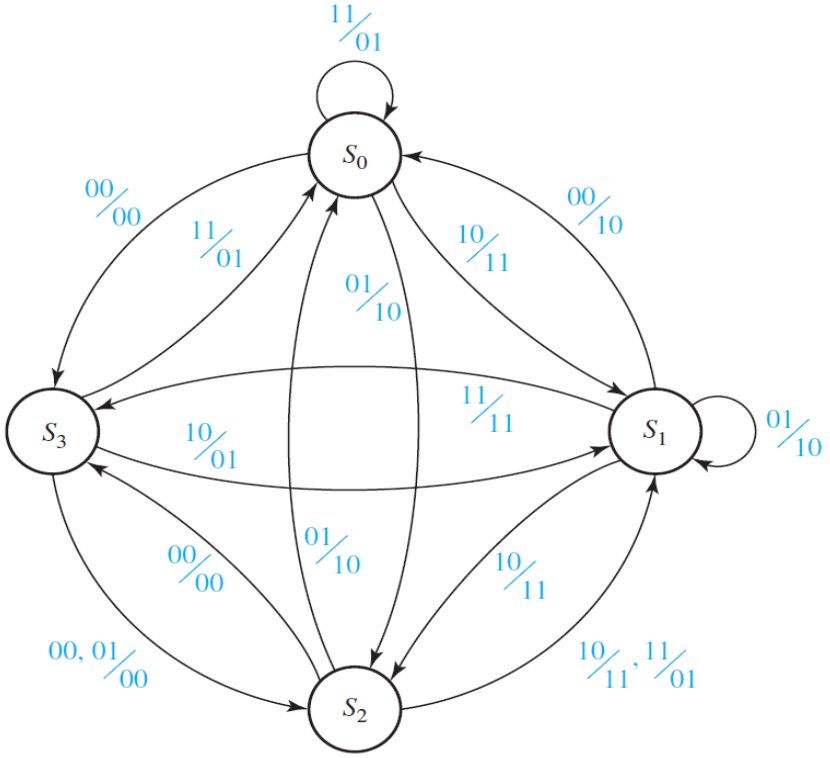
x_i	y_i	c_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



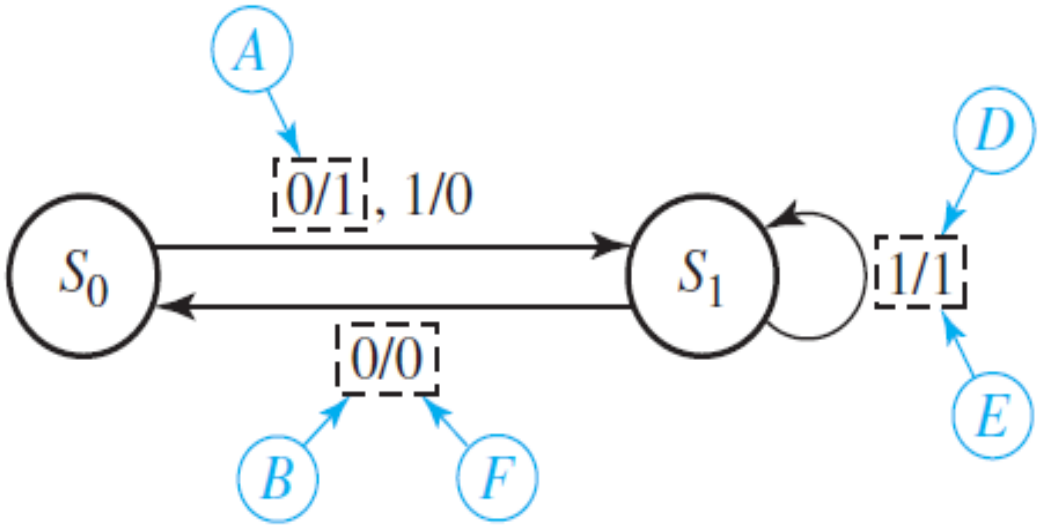
State table and state diagram – practice problem - 4

Present State	Next State $X_1X_2 = 00$	01	10	11	Present Output (Z_1Z_2) $X_1X_2 = 00$	01	10	11
S_0	S_3	S_2	S_1	S_0	00	10	11	01
S_1	S_0	S_1	S_2	S_3	10	10	11	11
S_2	S_3	S_0	S_1	S_1	00	10	11	01
S_3	S_2	S_2	S_1	S_0	00	00	01	01

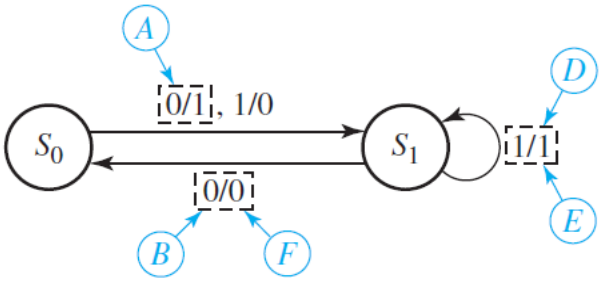
State table with multiple input and output



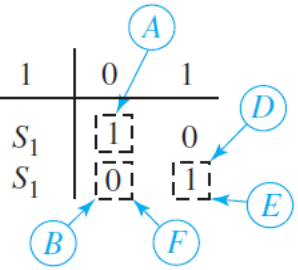
Timing chart – example

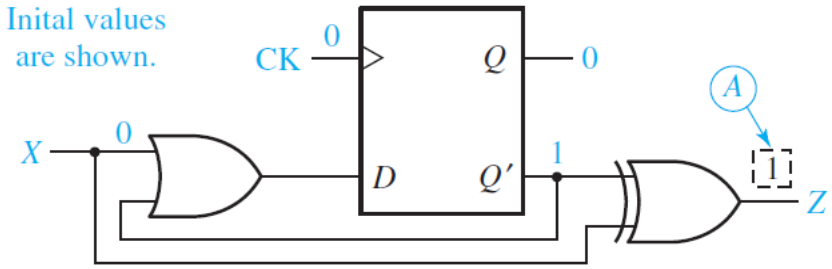


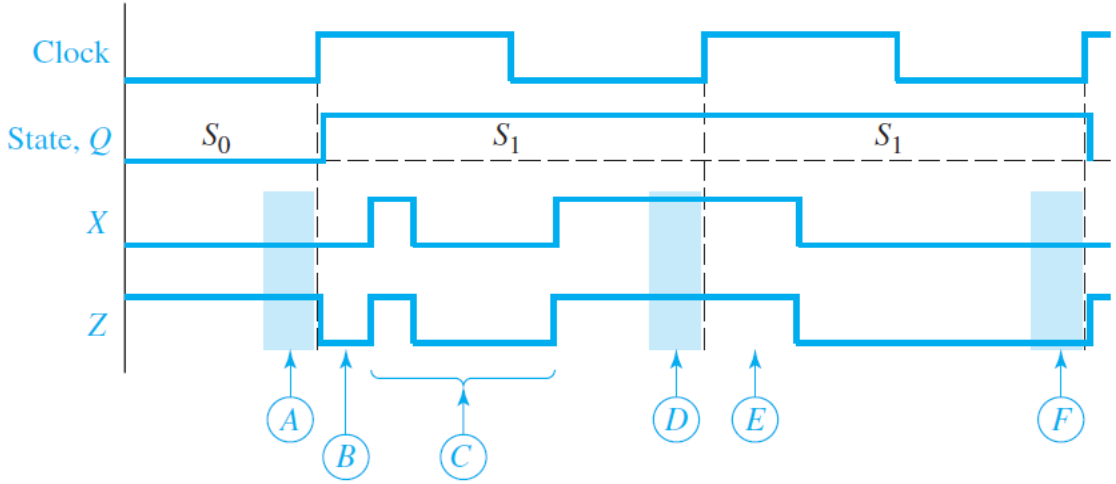
Timing chart – example



Q		$X = 0$	1
0	S_0	S_1	S_1
1	S_1	S_0	S_1







Drawn shaded area
(before rising edge of clock).

Timing diagrams

- Each time input X changes, trace the changes on the state graph, state table, the circuit and the timing chart.

Timing diagrams

- Several changes in input – affects output as well.
- Input must assume correct value before the rising edge of the clock – output should be read at this time (D).
- After rising edge of the clock – no state change – no output change.
- Input changes to new value – output change to its new value (F) – should be read before the next rising clock edge.

Timing diagrams

- Input and output sequence before each rising edge of the clock

$$X = 0\ 1\ 0$$
$$Z = 1\ 1\ 0$$

- Verify this in state table, state graph and state diagram.