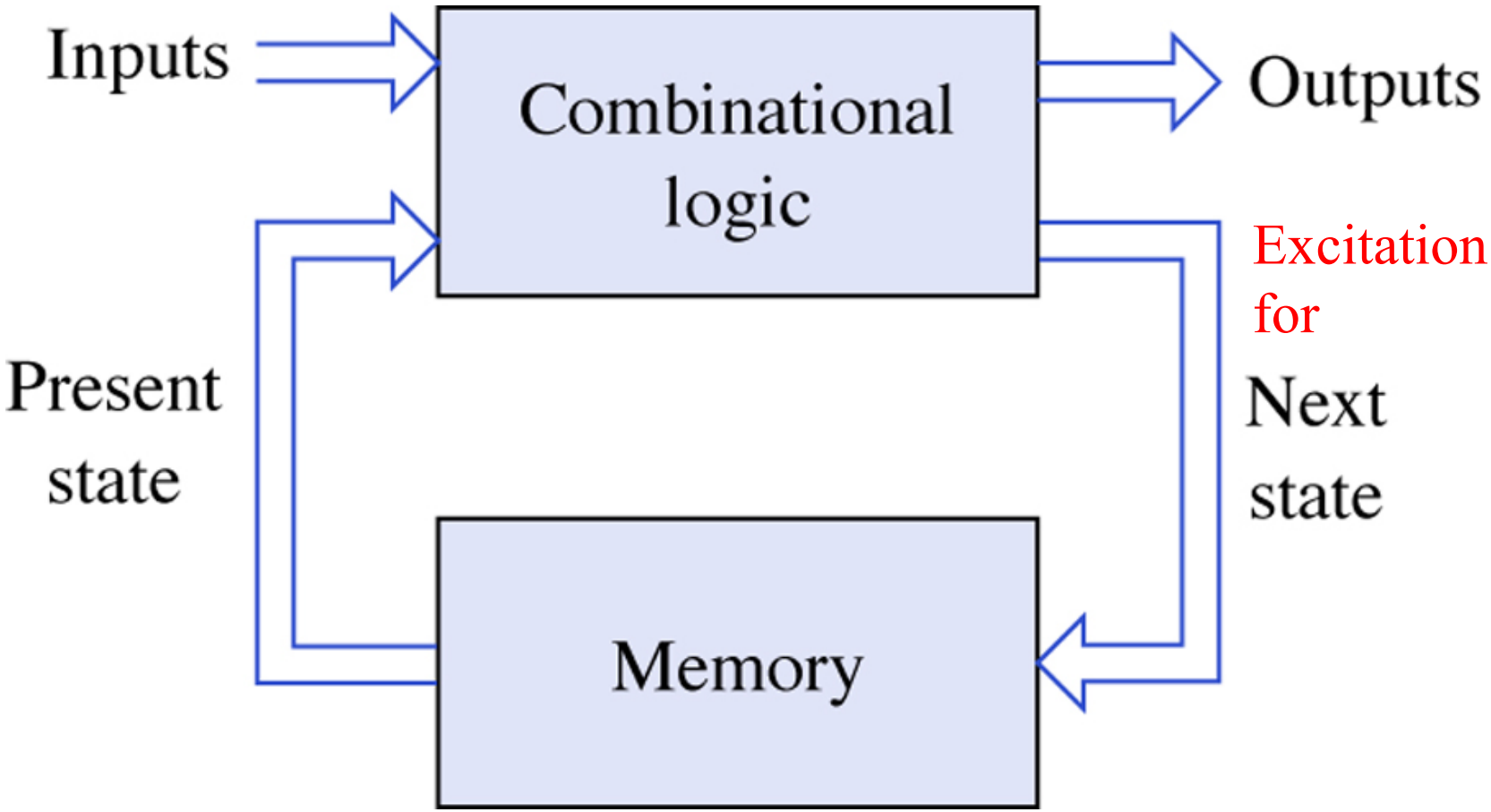
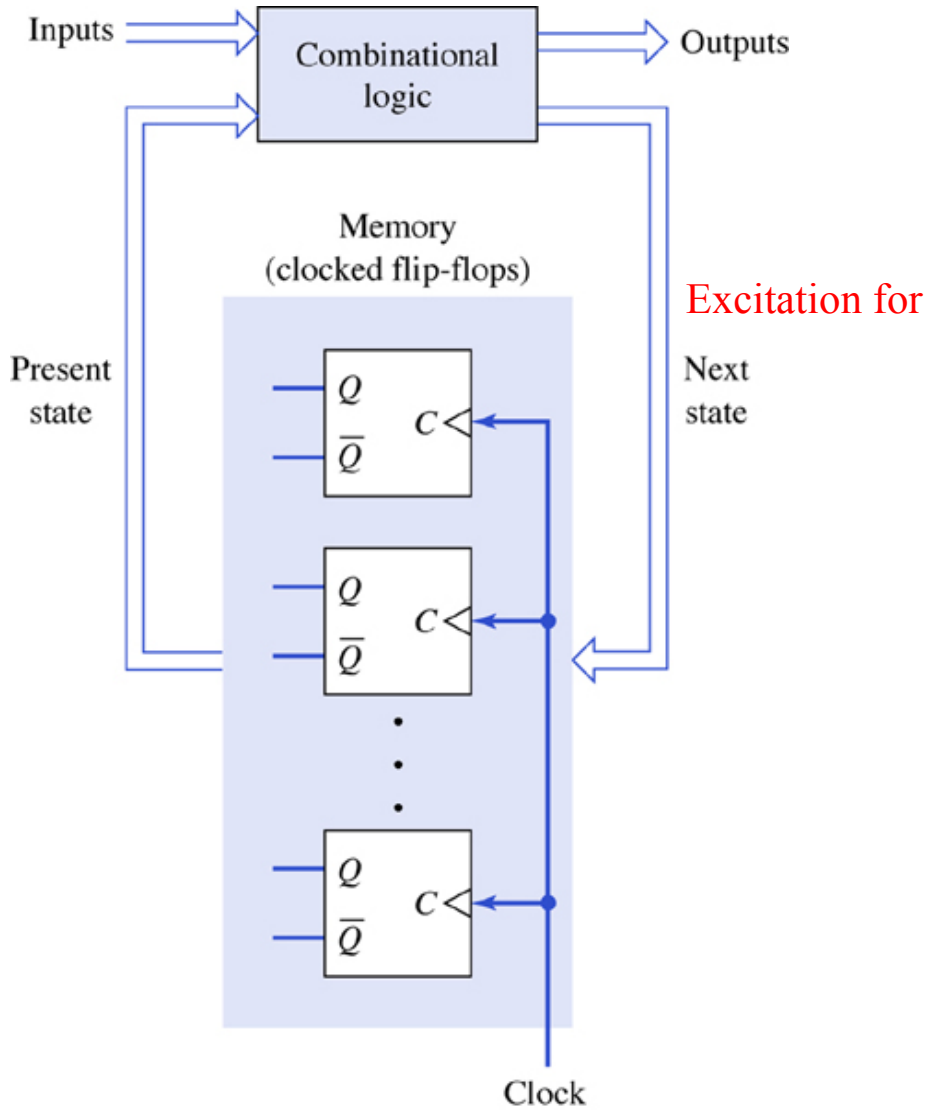
Chapter 7

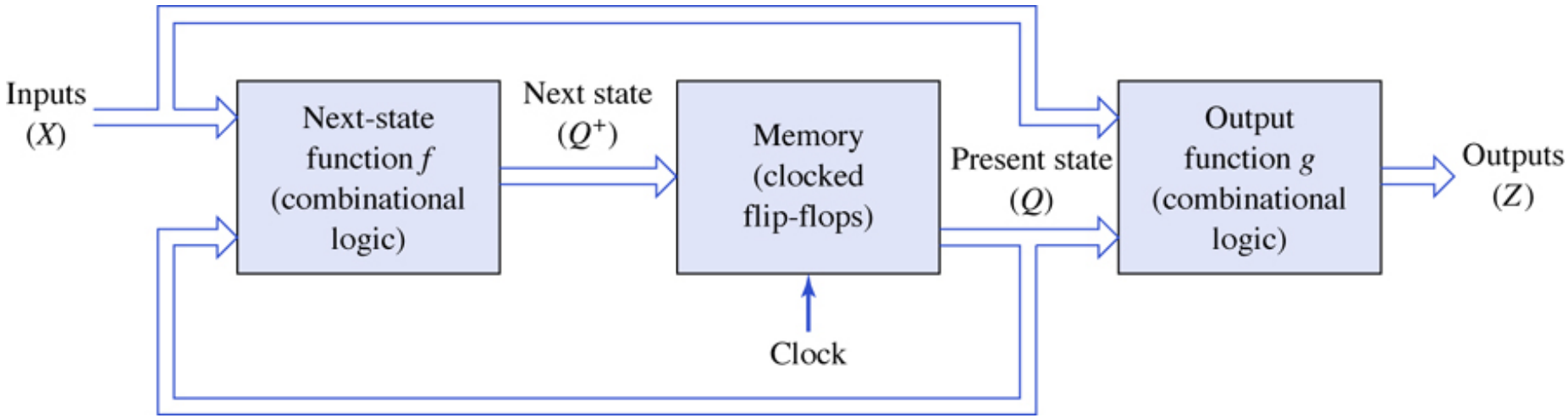
Synchronous Sequential Networks


Inputs
Combinational logic
Outputs
Excitation for
Next state
Present state
Memory

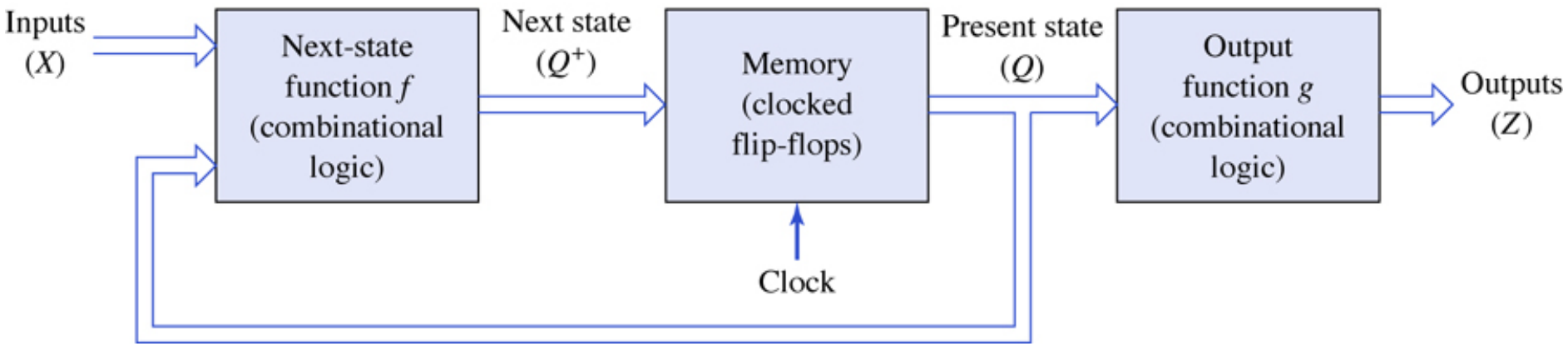
Structure of a clocked synchronous sequential network



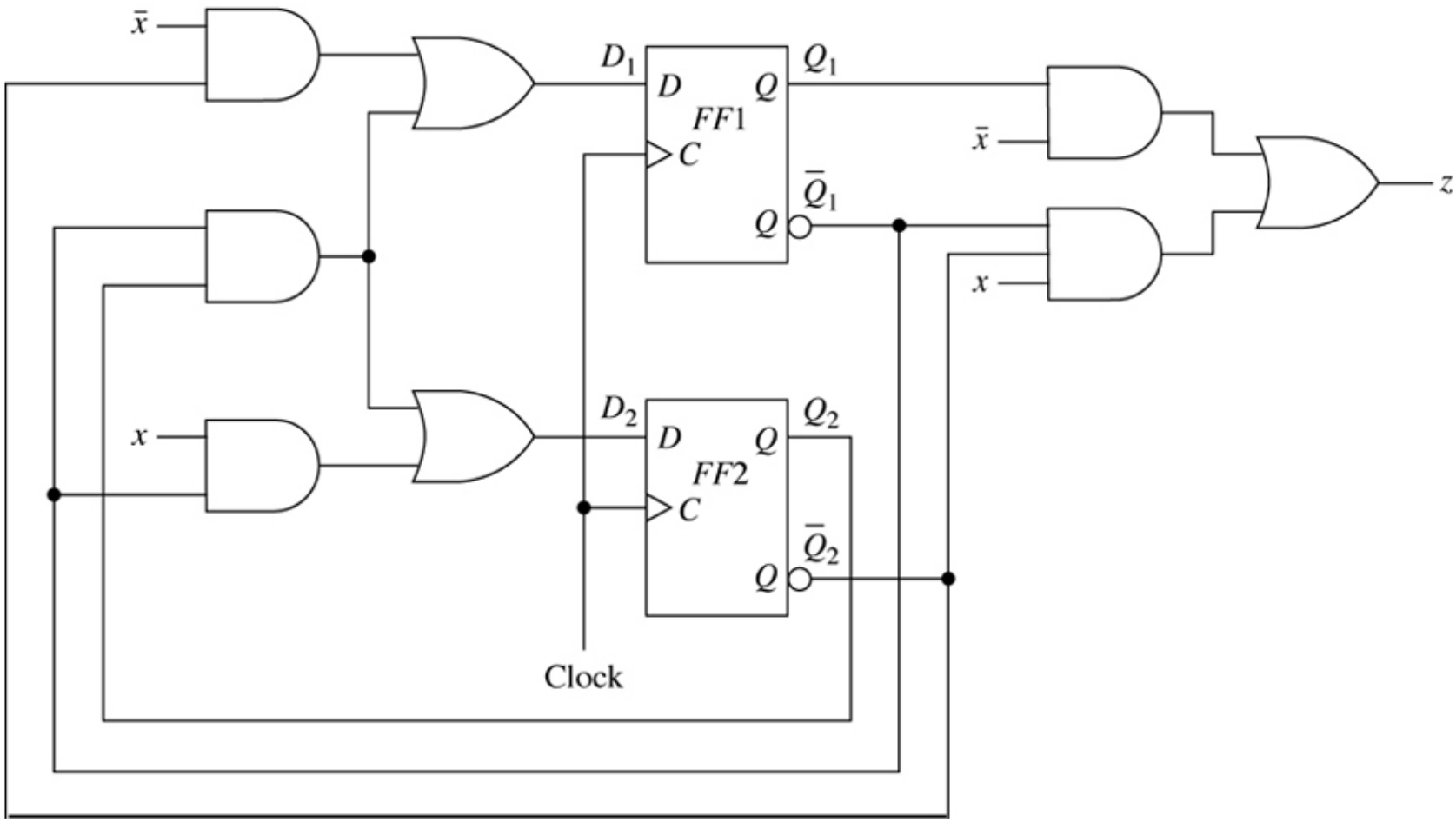
Mealy model of a clocked synchronous sequential network



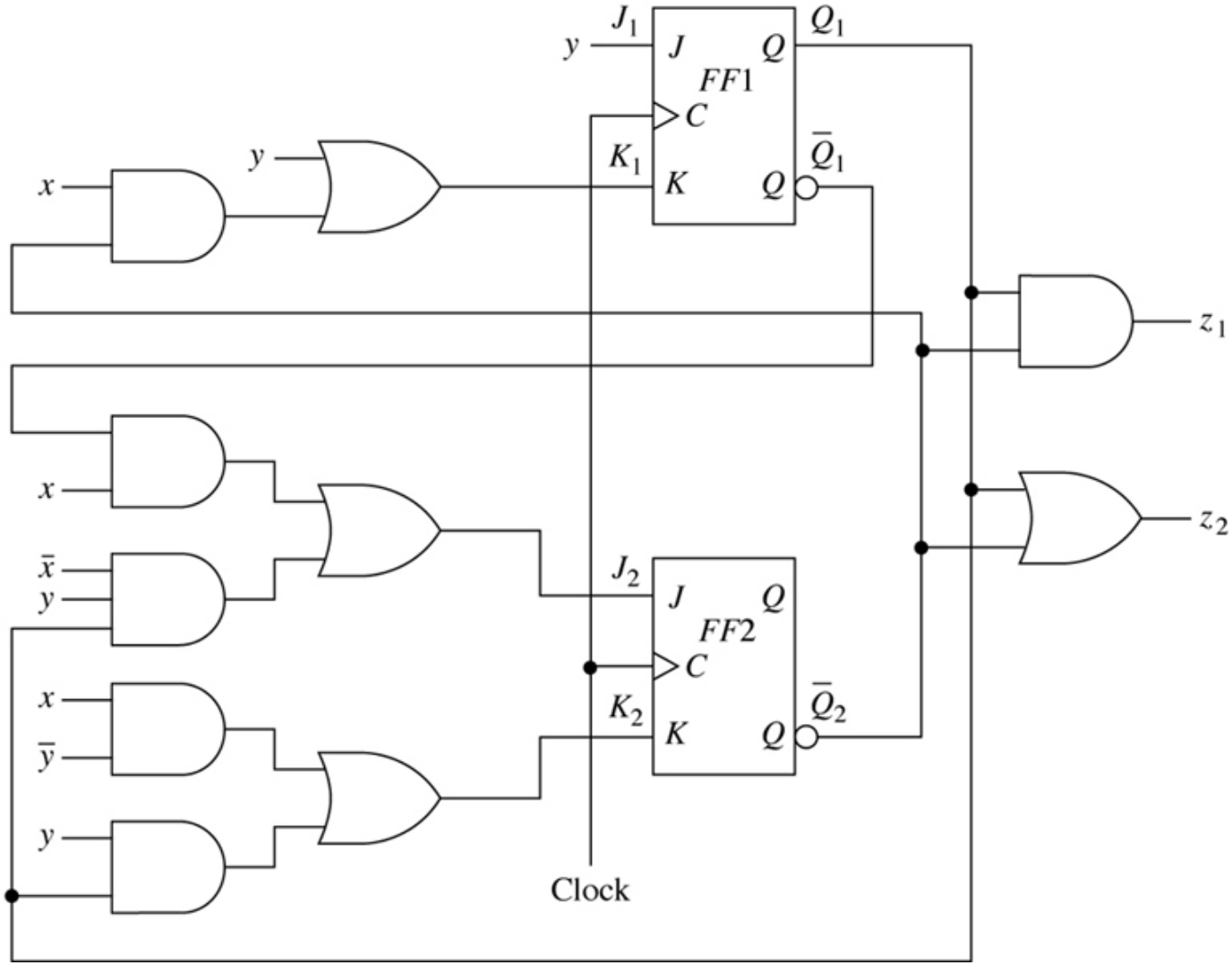
Moore model of a clocked synchronous sequential network



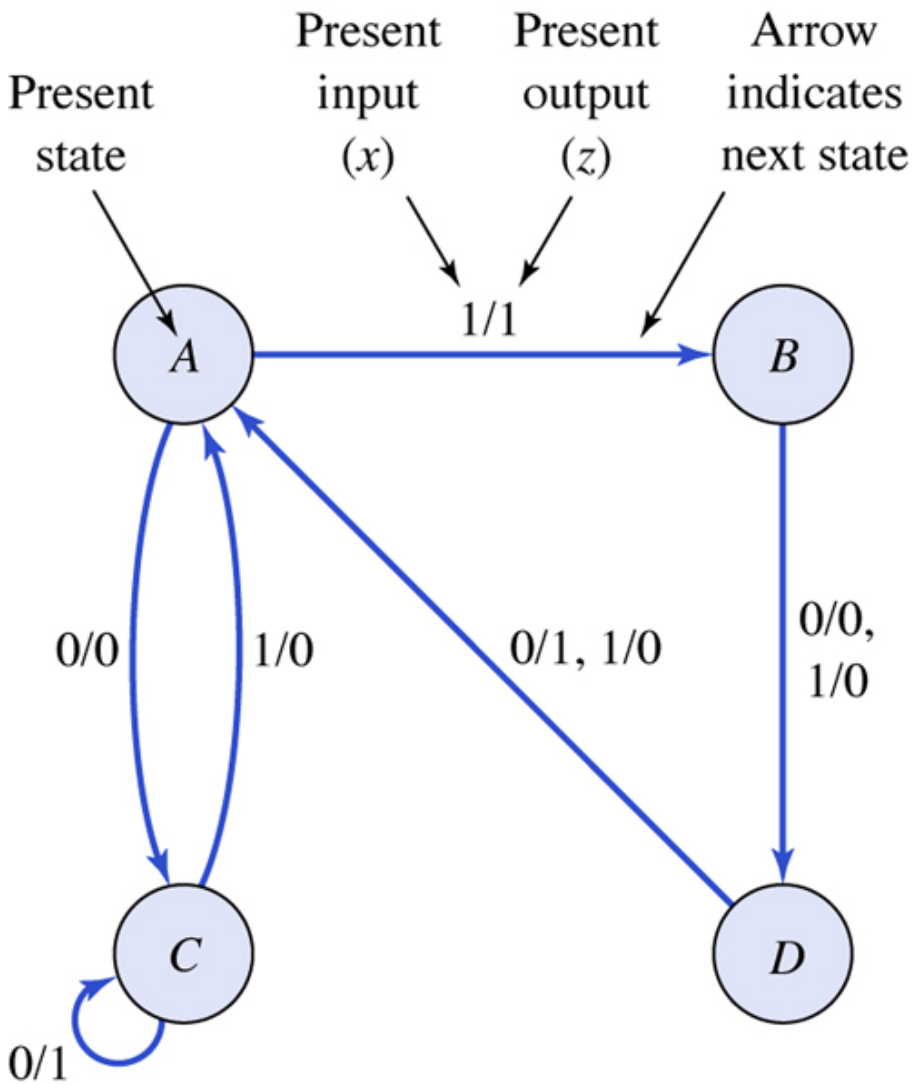
Logic diagram for Example 7.1



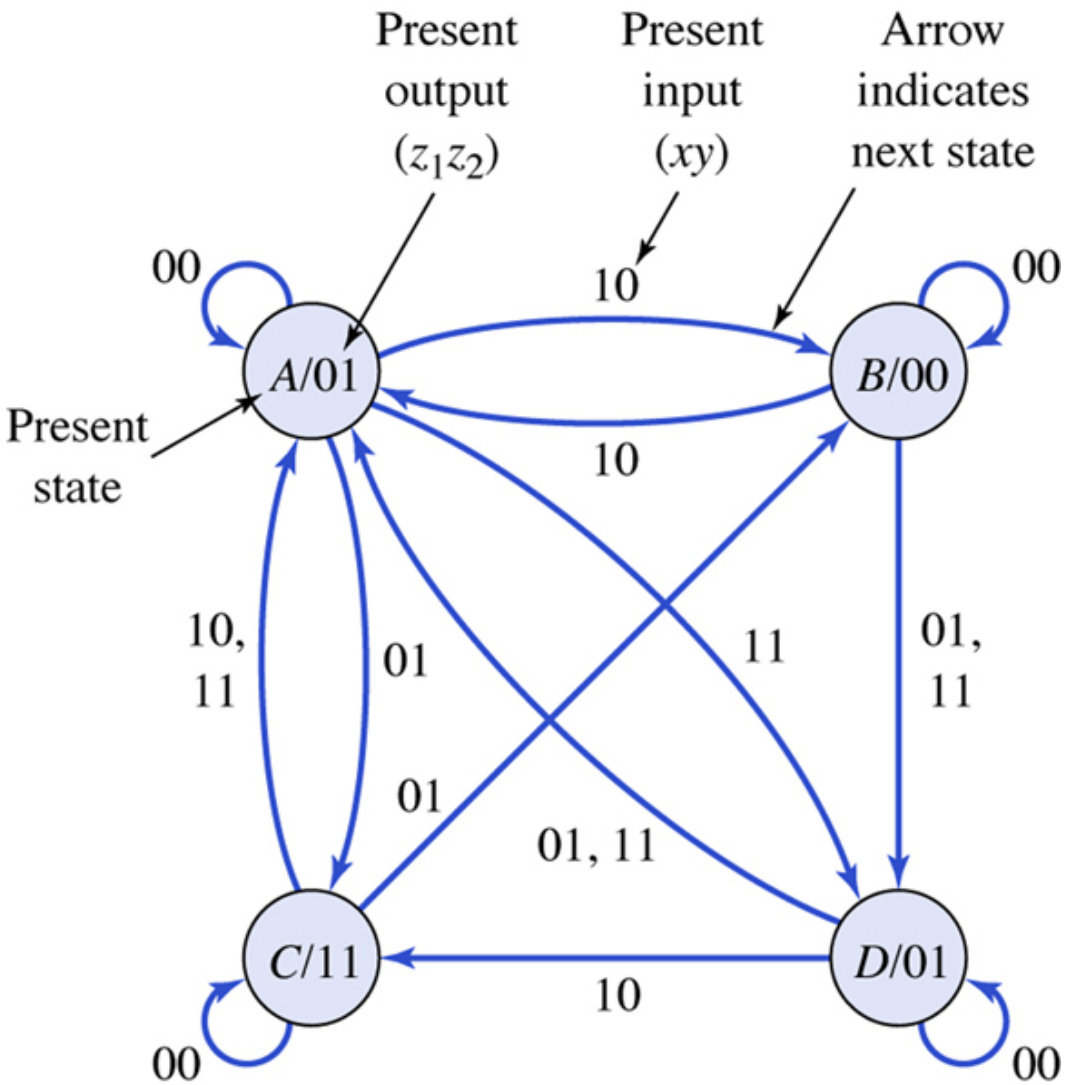
Logic diagram for Example 7.2



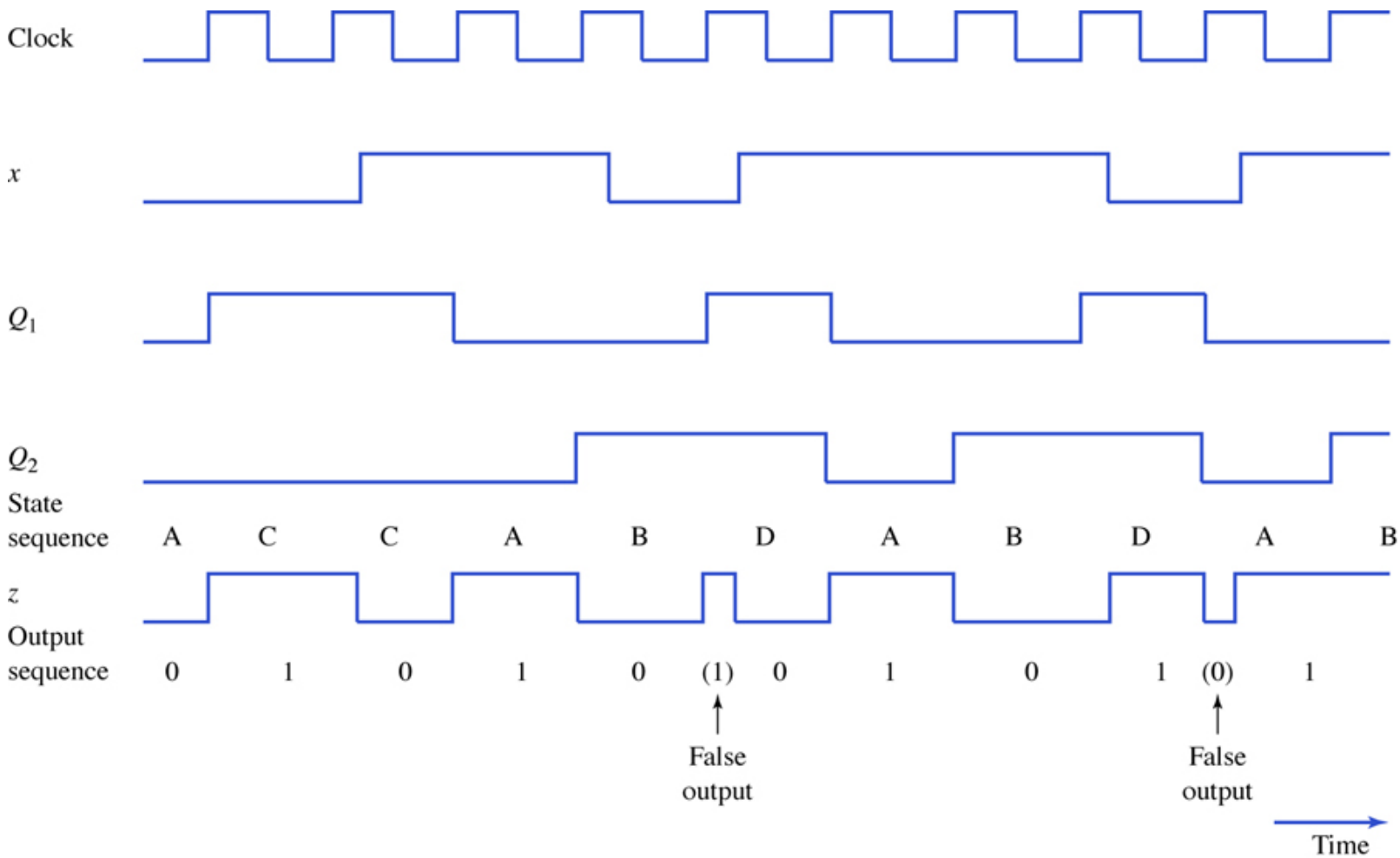
State diagram for Example 7.1



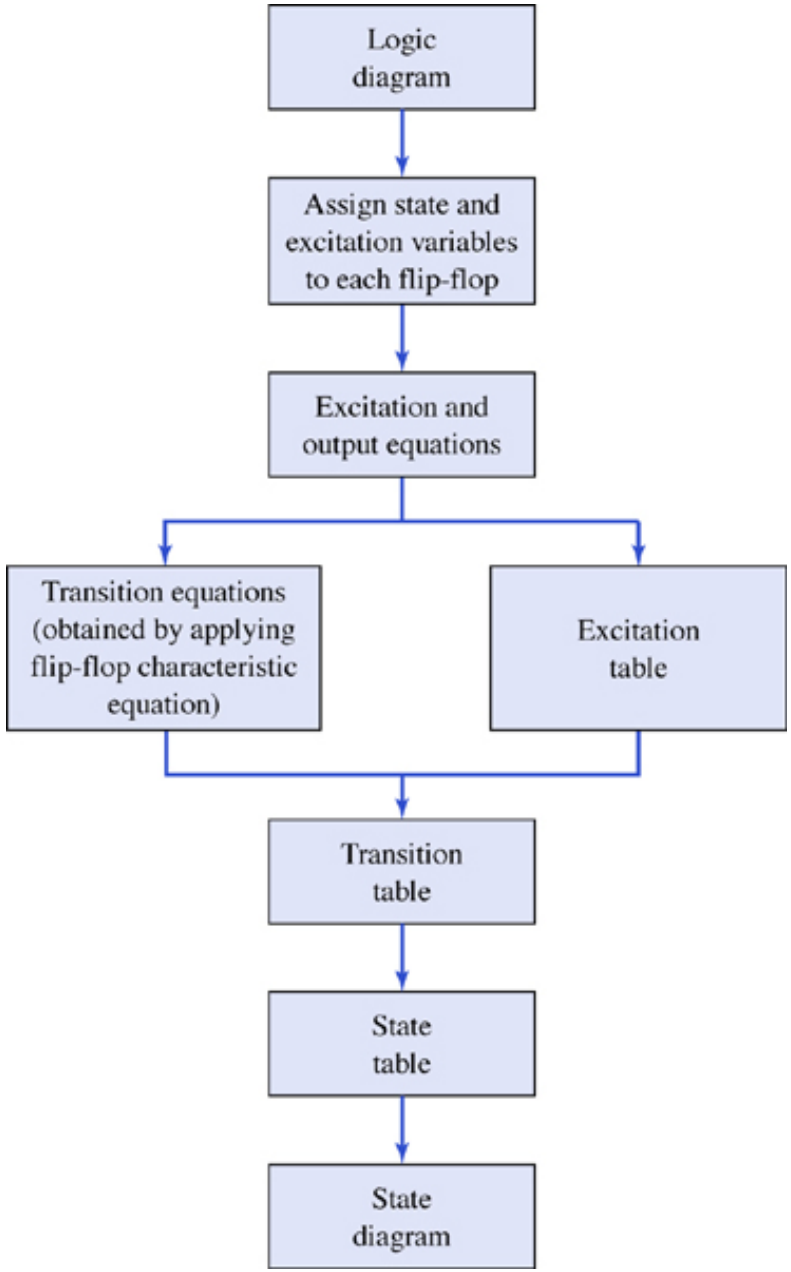
State diagram for Example 7.2



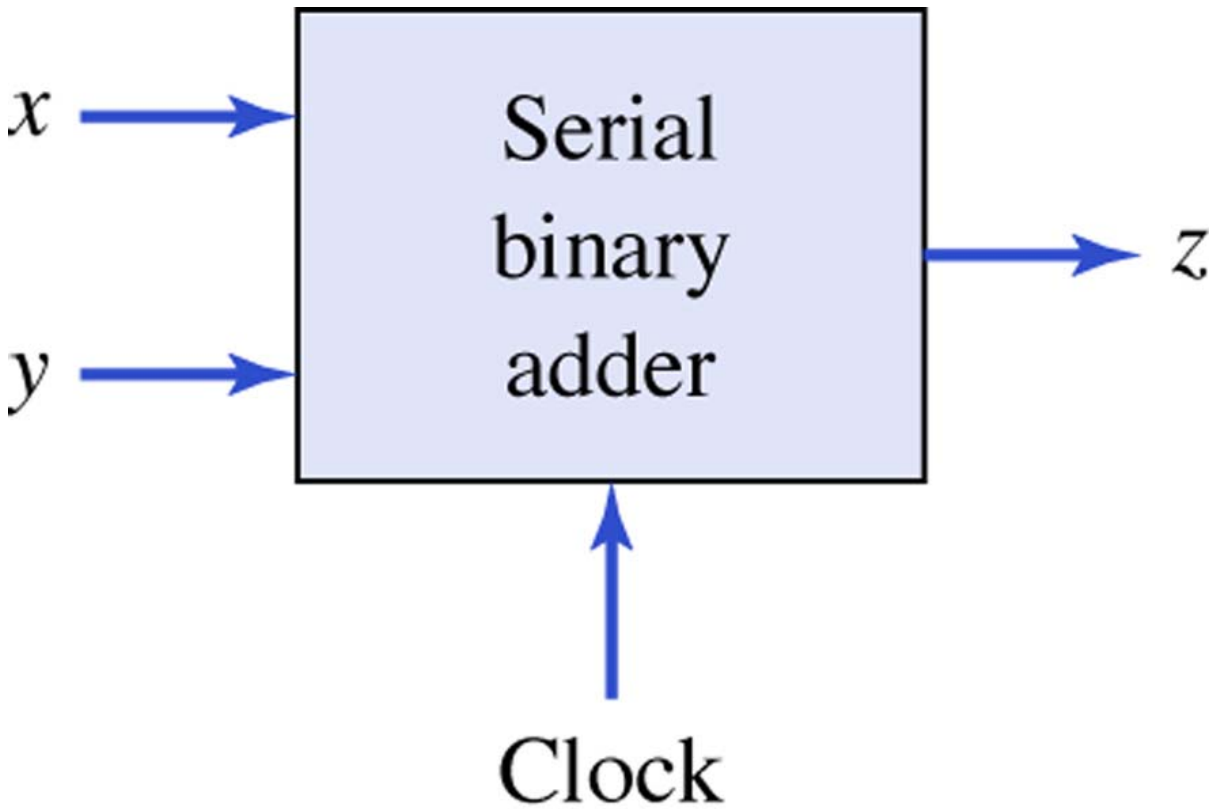
Timing diagram for Example 7.1



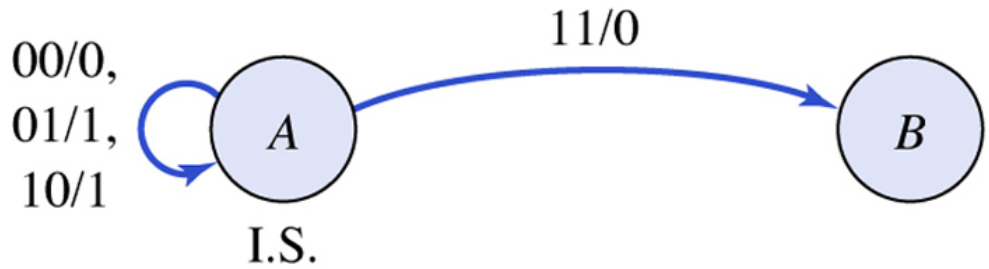
The analysis procedure



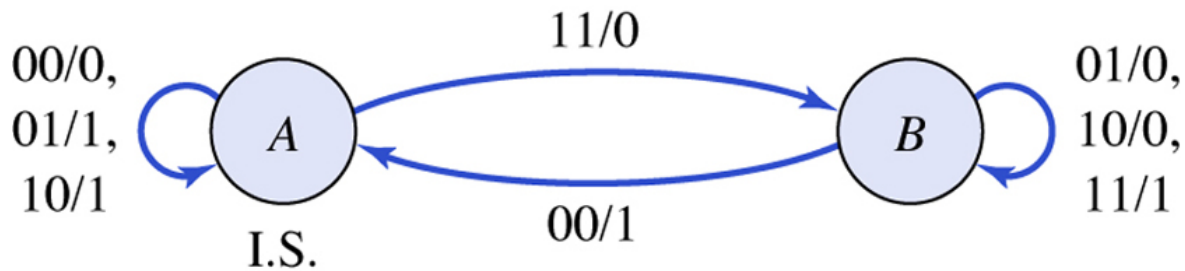
The serial binary adder



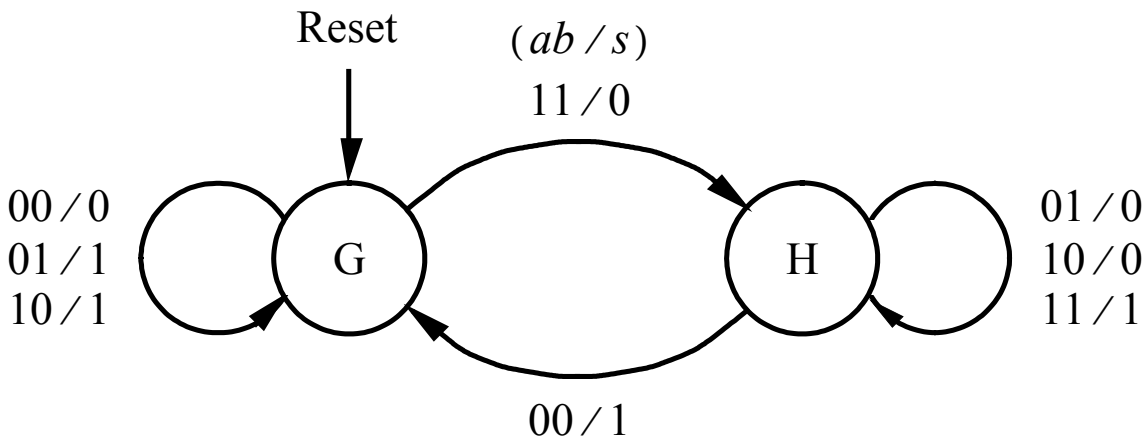
Obtaining the state diagram for a Mealy serial binary adder. (*a*) Partial state diagram. (*b*) Completed state diagram.



(*a*)



(*b*)



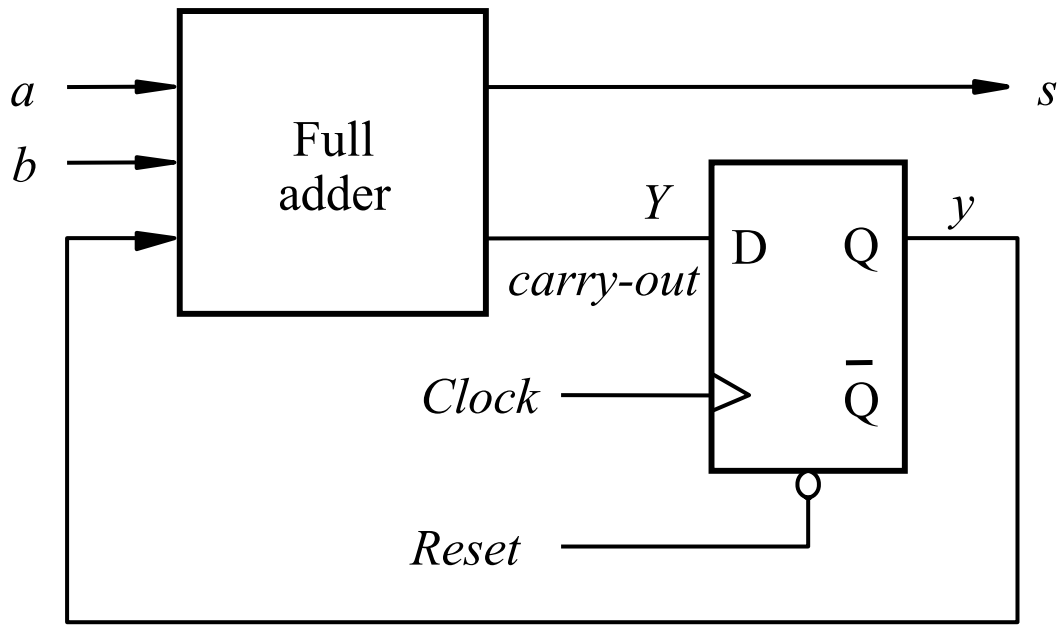
G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder

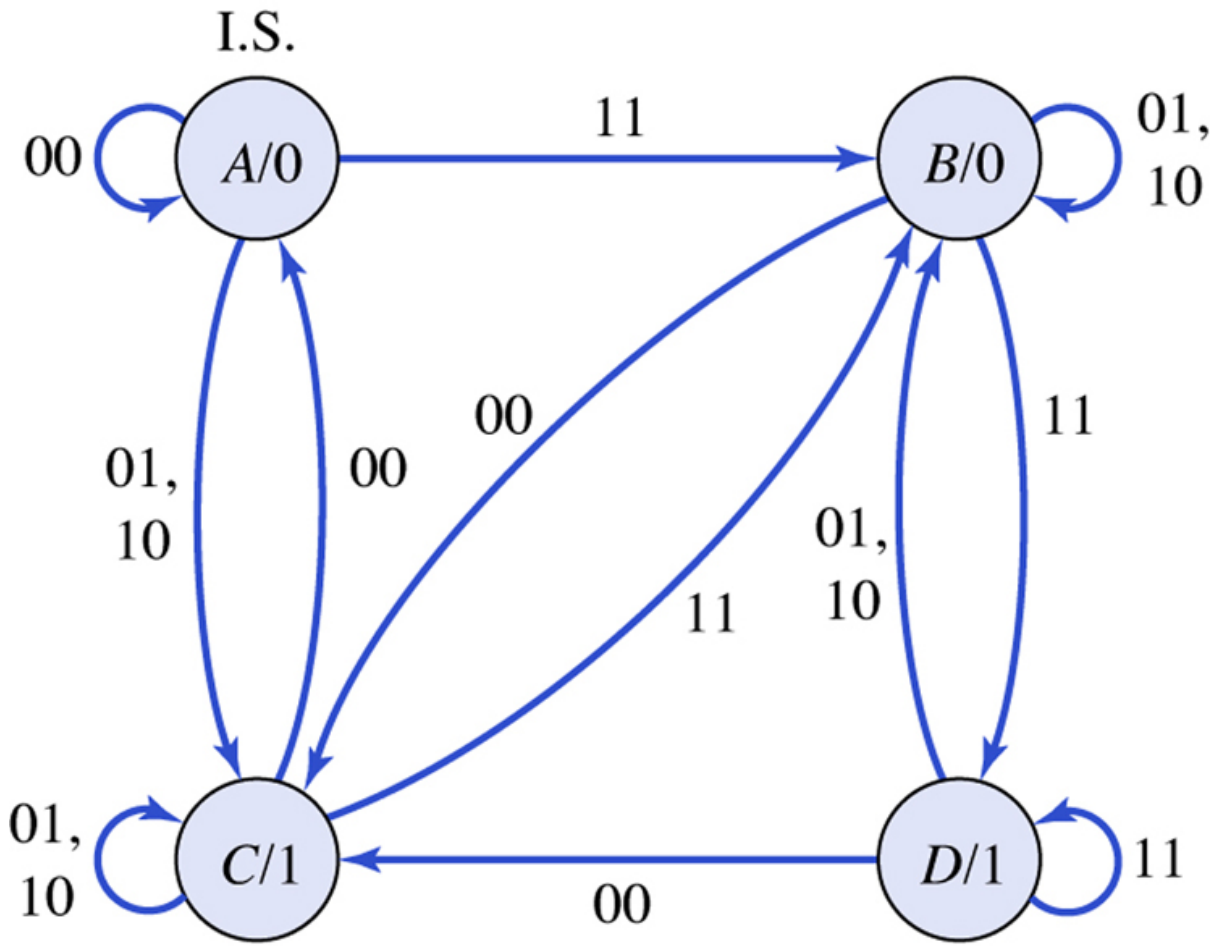
Present state	Next state				Output *s*			
	ab =00	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

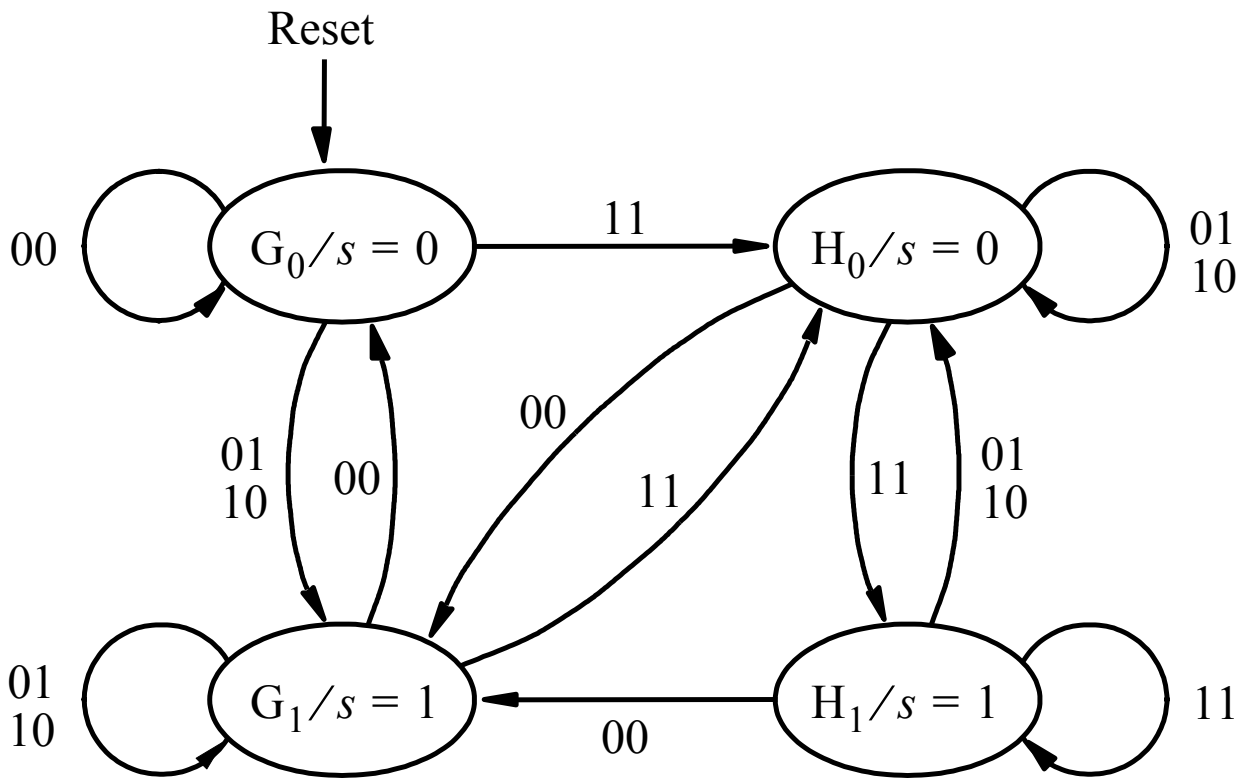
State table for the serial adder



Circuit for the adder FSM

State diagram for a Moore serial binary adder

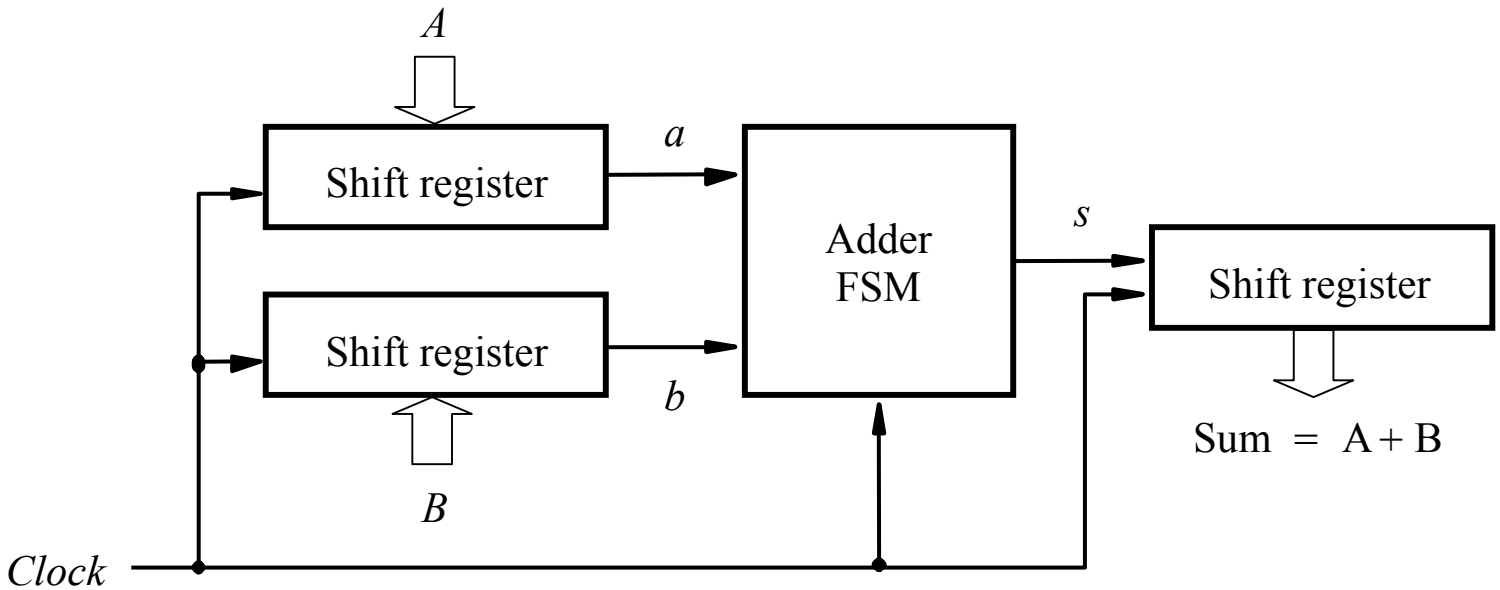




State diagram for the Moore-type serial adder FSM

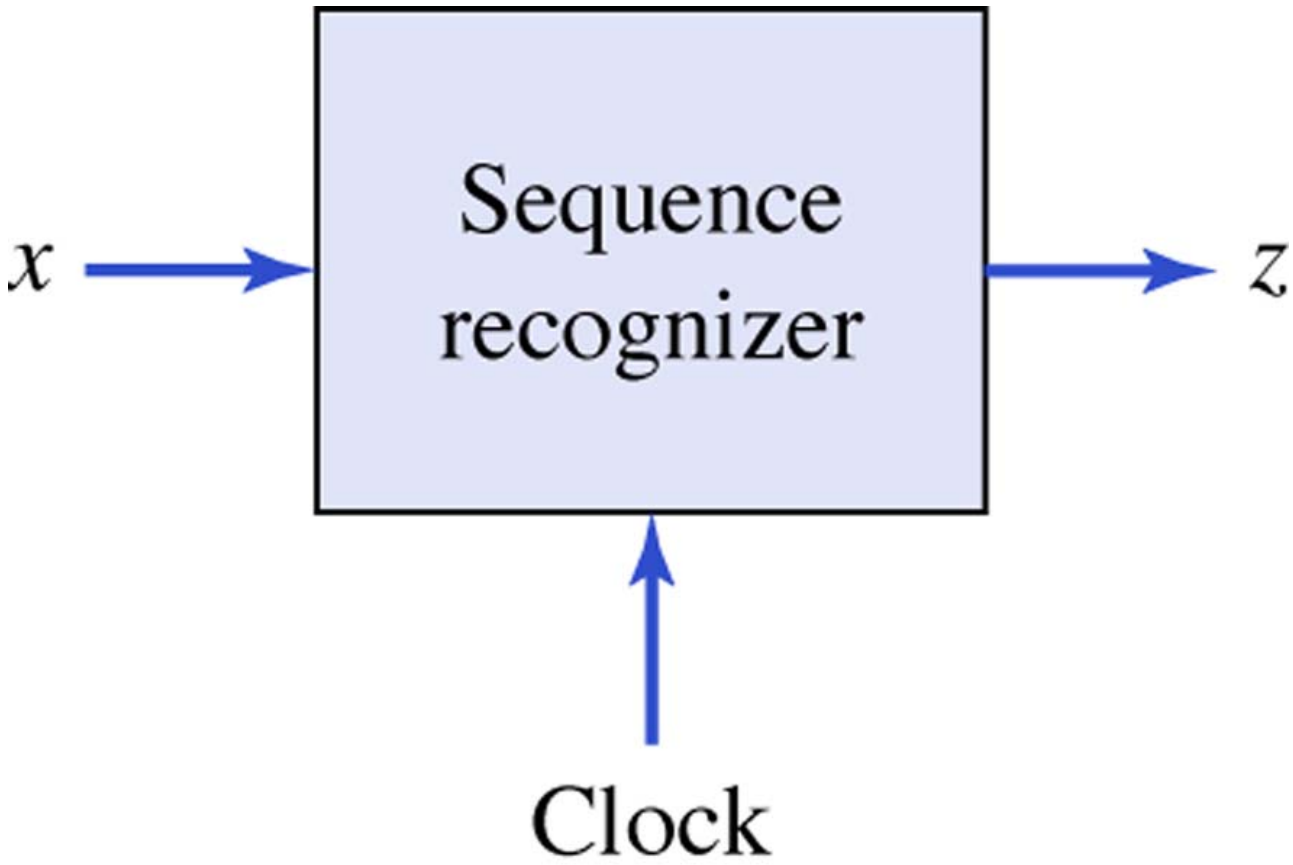
Present state	Nextstate				Output
	ab =00	01	10	11	*s*
G_0	G_0	G_1	G_1	H_0	0
G_1	G_0	G_1	G_1	H_0	1
H_0	G_1	H_0	H_0	H_1	0
H_1	G_1	H_0	H_0	H_1	1

State table for the Moore-type serial adder FSM

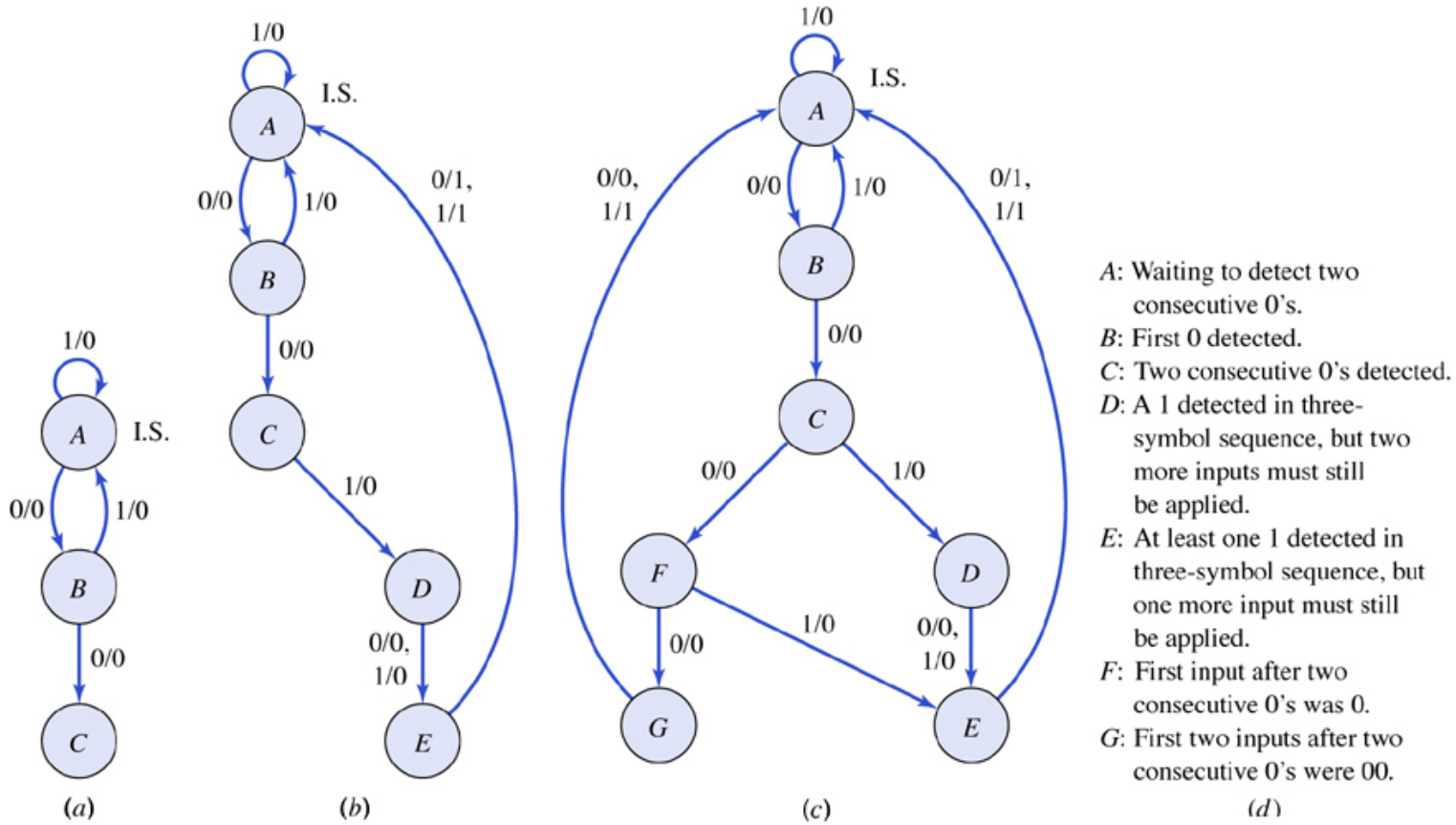


Block diagram of a serial adder

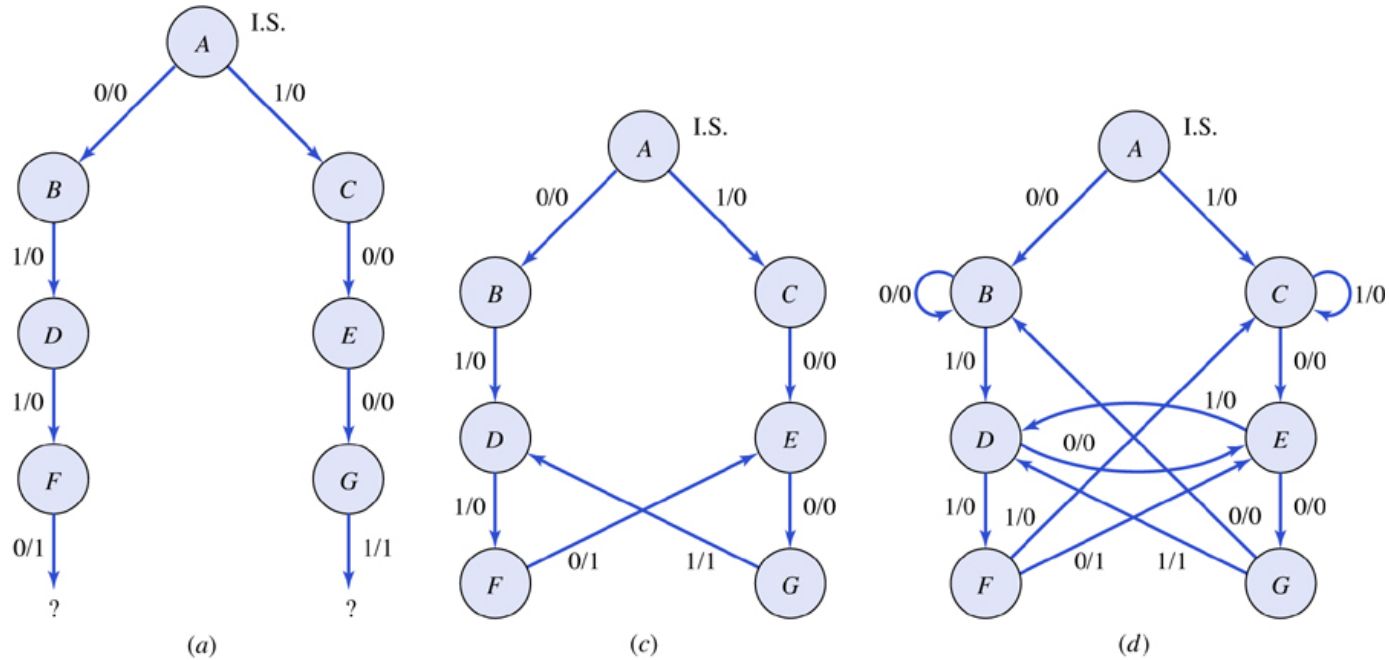
A sequence recognizer



State diagram for a sequence recognizer. (*a*) Detection of two consecutive 0's. (*b*) Partial analysis of the three-symbol sequence. (*c*) Completed state diagram. (*d*) Definition of states.



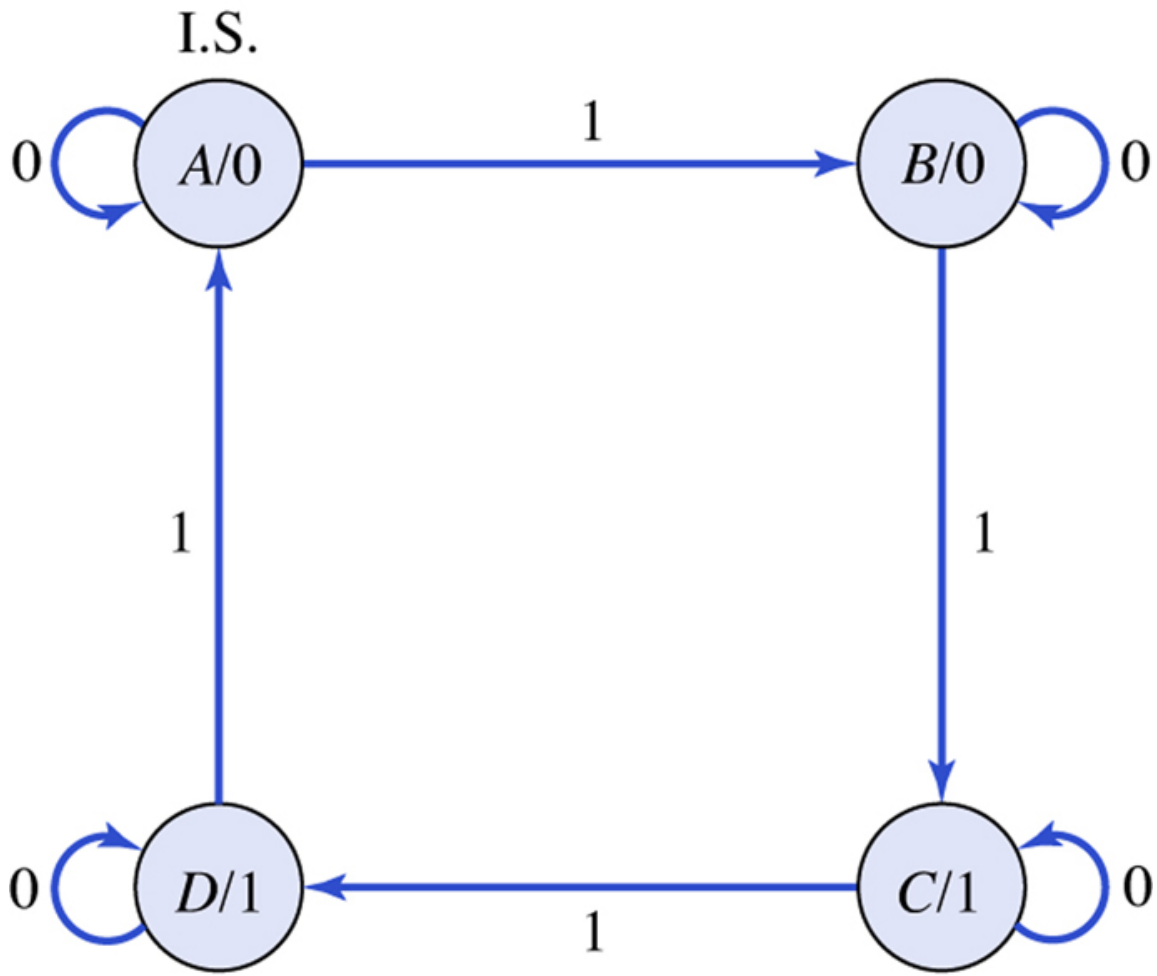
A 0110/1001 sequence recognizer. (*a*) Beginning the detection of the sequences 0110 or 1001. (*b*) Definition of states. (*c*) Completing the detection of the two sequences 0110 or 1001. (*d*) Completed state diagram.



A: No inputs received (initial state).
B: Last input received was 0.
C: Last input received was 1.
D: Last two inputs received were 01.
E: Last two inputs received were 10.
F: Last three inputs received were 011.
G: Last three inputs received were 100.

(*b*)

State diagram for the final example

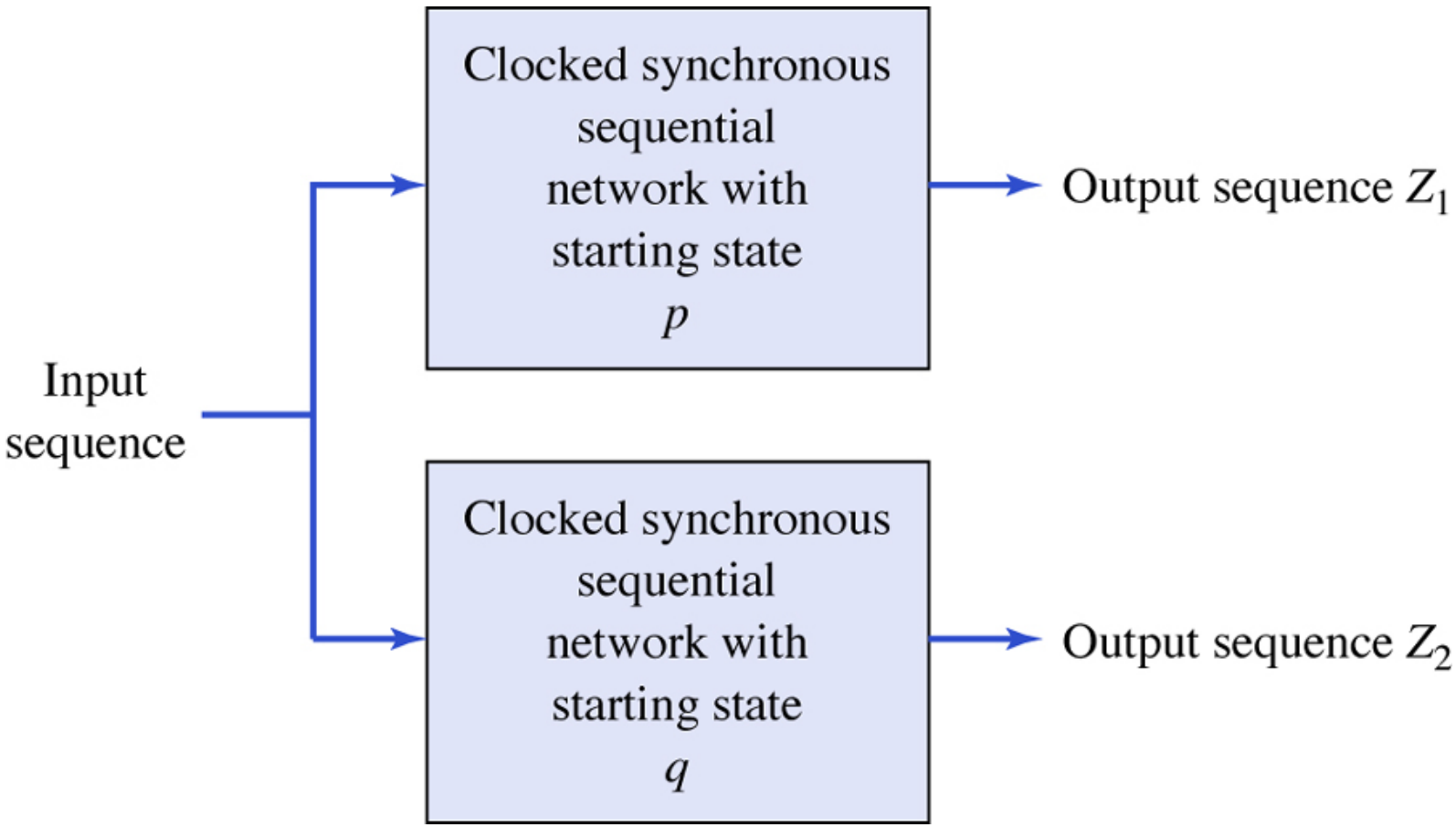


State reduction

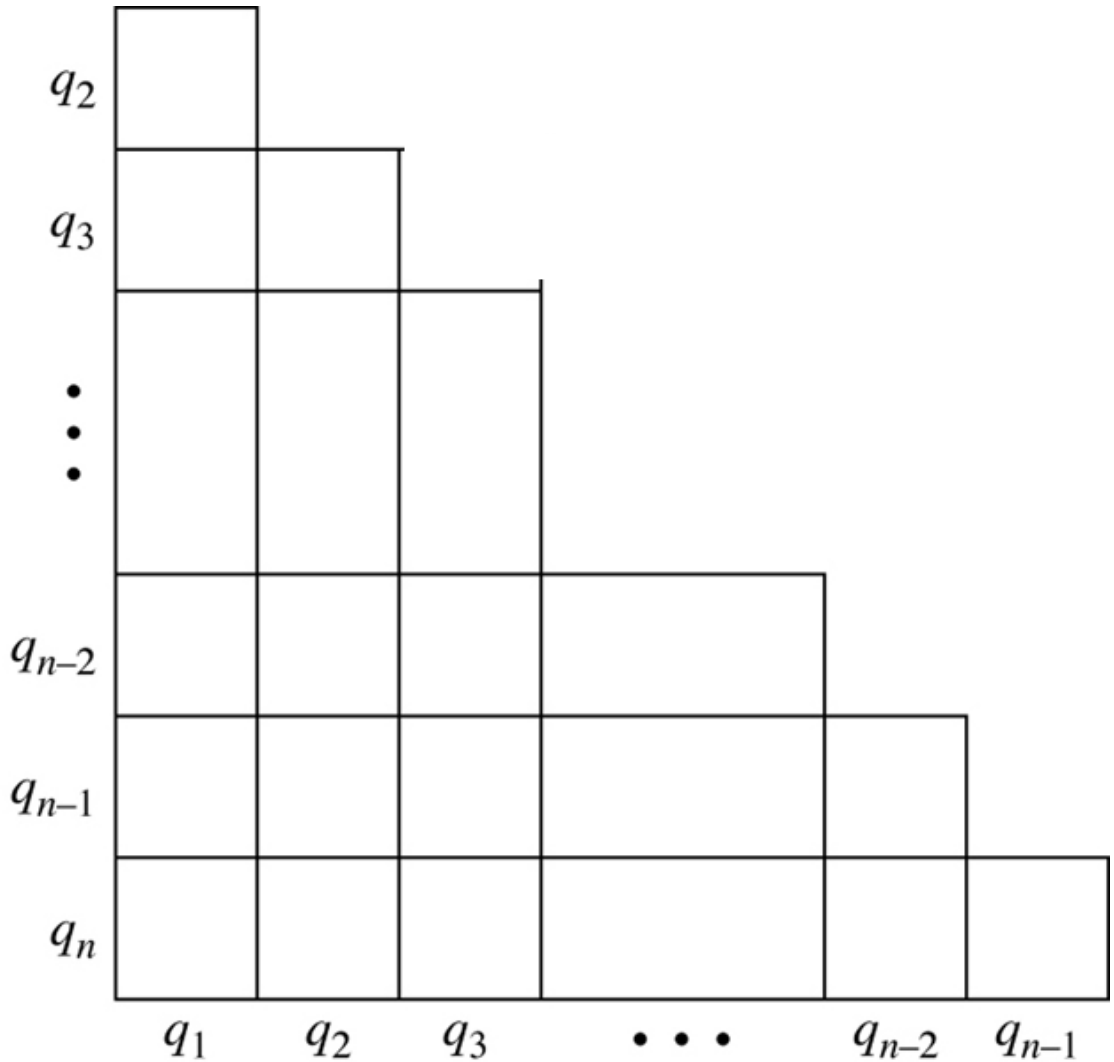
Theorem 7.1

Two states p and q of a synchronous sequential network are equivalent if and only if for each combination of values of the input variables (1) their outputs are identical and (2) their next states are equivalent.

Experiment for determining equivalent pairs of states



The structure of an implication table



Implication table inspection process

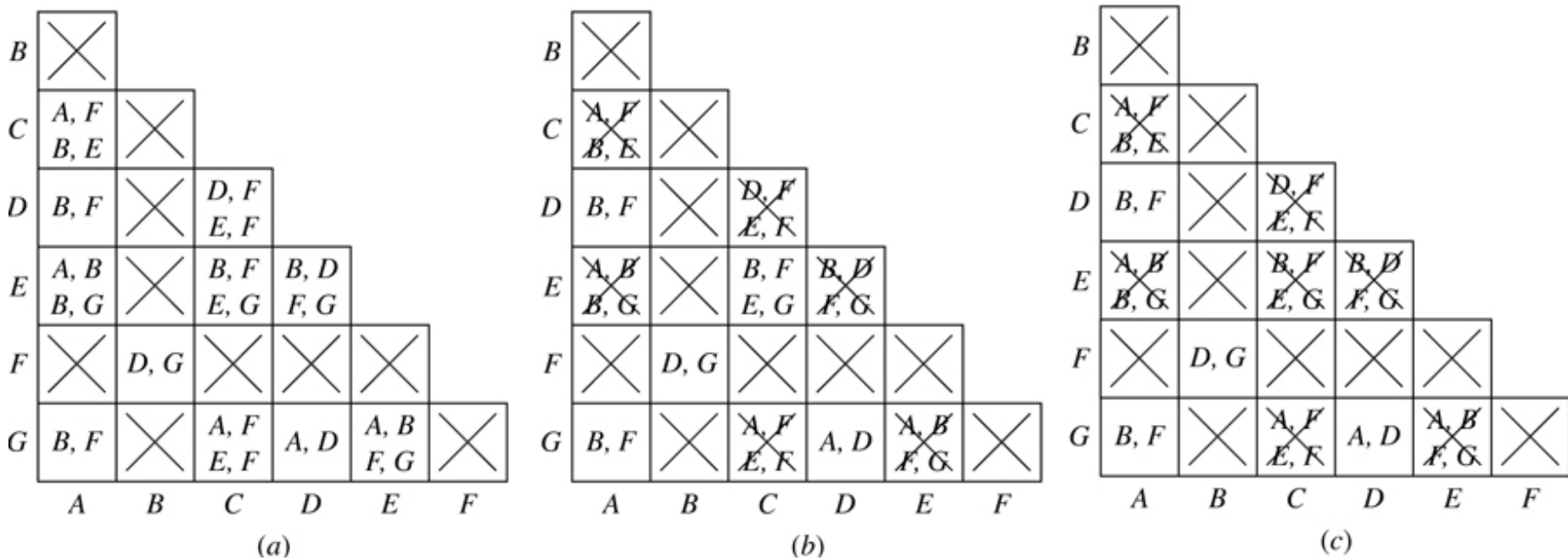
- If (q_a, q_b) is in the q_iq_j cell, and if q_aq_b cell contains a $\times$, then place a $\times$ in the q_iq_j cell and ignore all other entries in that cell.

  Otherwise inspect other pairs in q_iq_j.
- Repeat this steps until it is possible to make an entire pass without adding a new $\times$.
- Upon completion, if q_iq_j cell contains no $\times$ then state q_i is equivalent to q_j.

Example state table

present state	next state		output (z)	
	x=0	*x=1*	*x=0*	*x=1*
A	A	B	0	0
B	D	C	0	1
C	F	E	0	0
D	D	F	0	0
E	B	G	0	0
F	G	C	0	1
G	A	F	0	0

Implication table for determining the equivalent states. (*a*) The initial table. (*b*) After the first pass. (*c*) After the second pass.



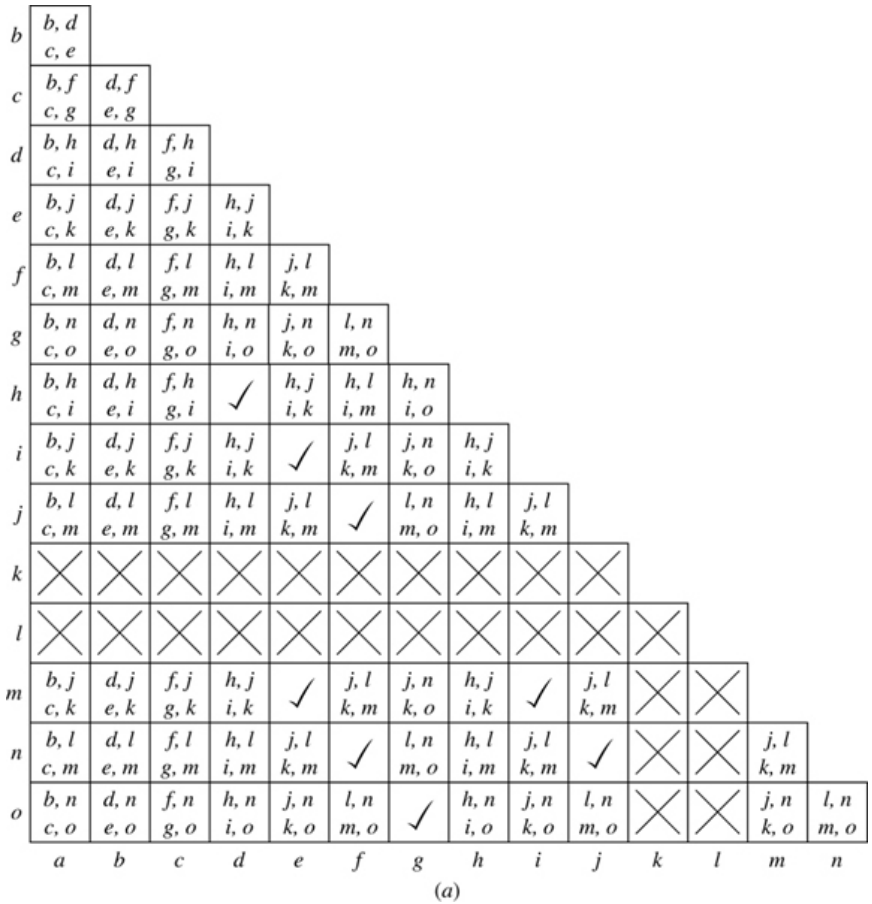
Equivalence classes of states

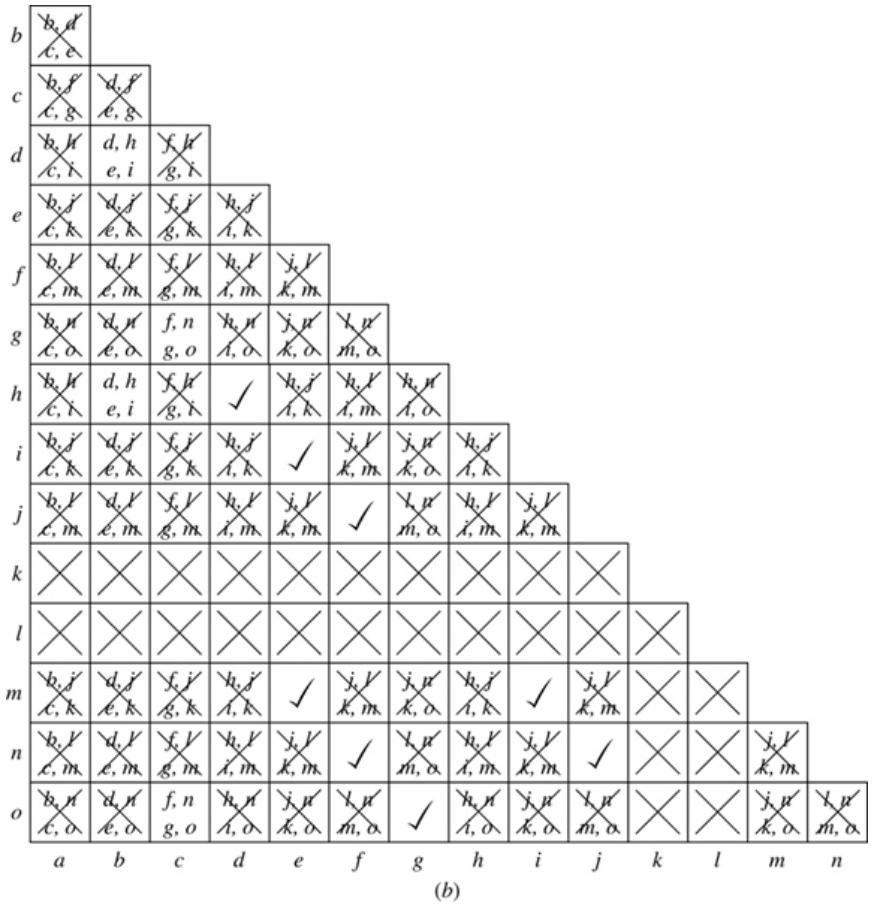
- The resulting implication table shows that A=D, A=G, B=F, and D=G.
- This leads us to the formation of the equivalence classes {(A,D,G), (B,F), (C), (E)}
- Hence the reduced state table shown below.

Minimized example state table

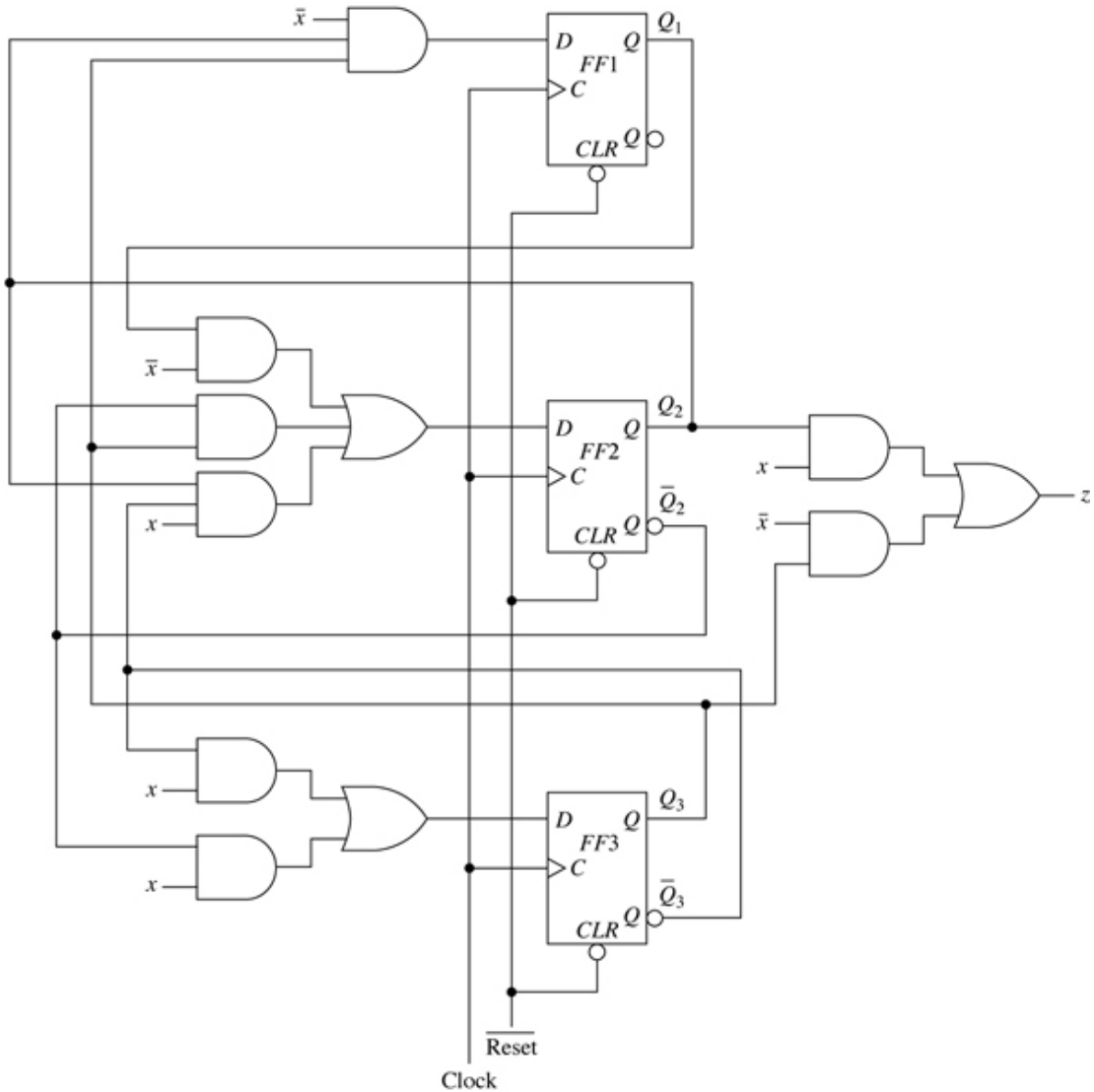
present state		next state		output (z)	
		x=0	*x=1*	*x=0*	*x=1*
(A,D,G):	α	α	β	0	0
(B,F):	β	α	γ	0	1
(C):	γ	β	δ	0	0
(E):	δ	β	α	0	0

Implication table for determining equivalent states of the 0110/1001 sequence recognizer. (*a*) Initial table. (*b*) Final table.

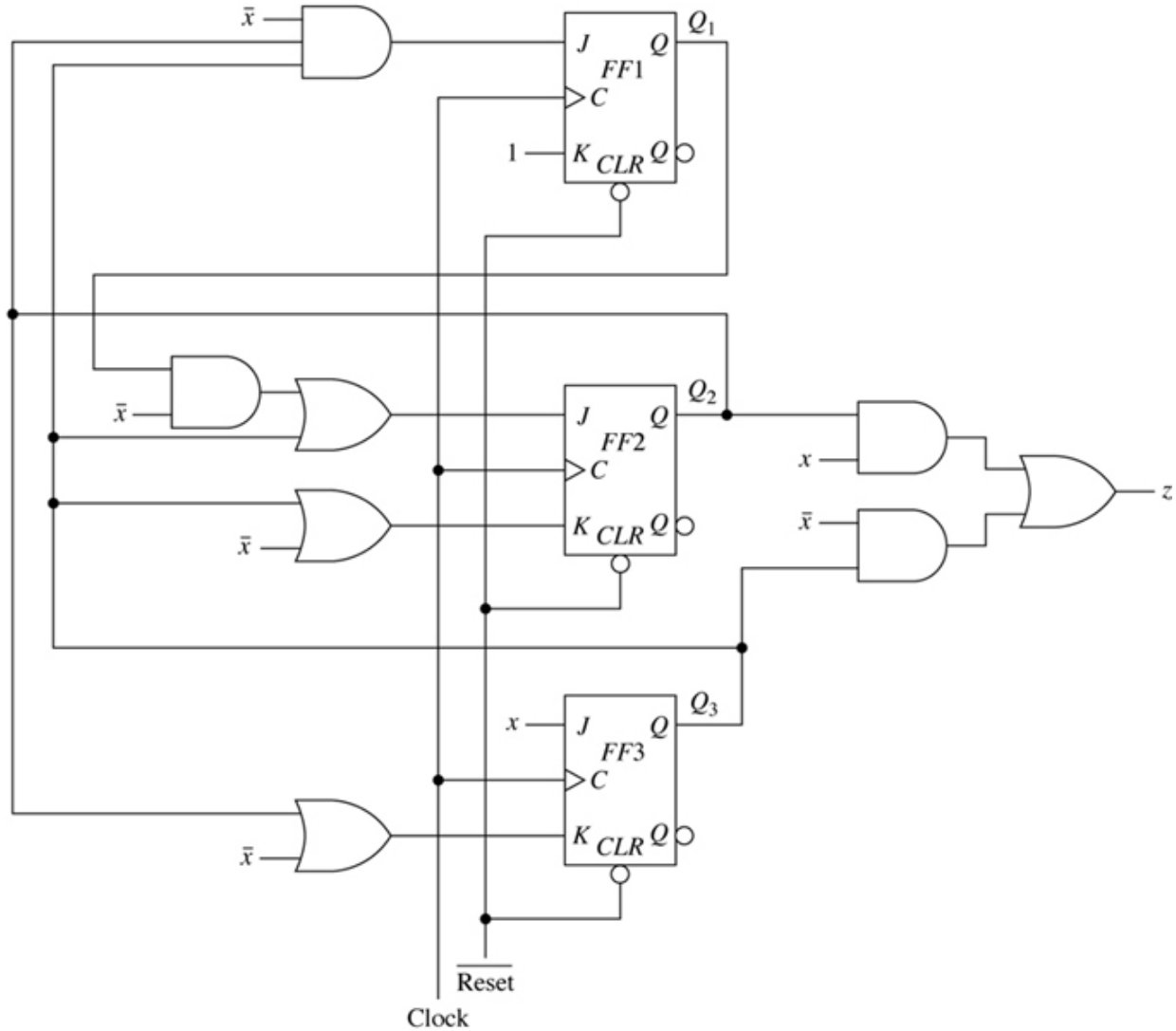




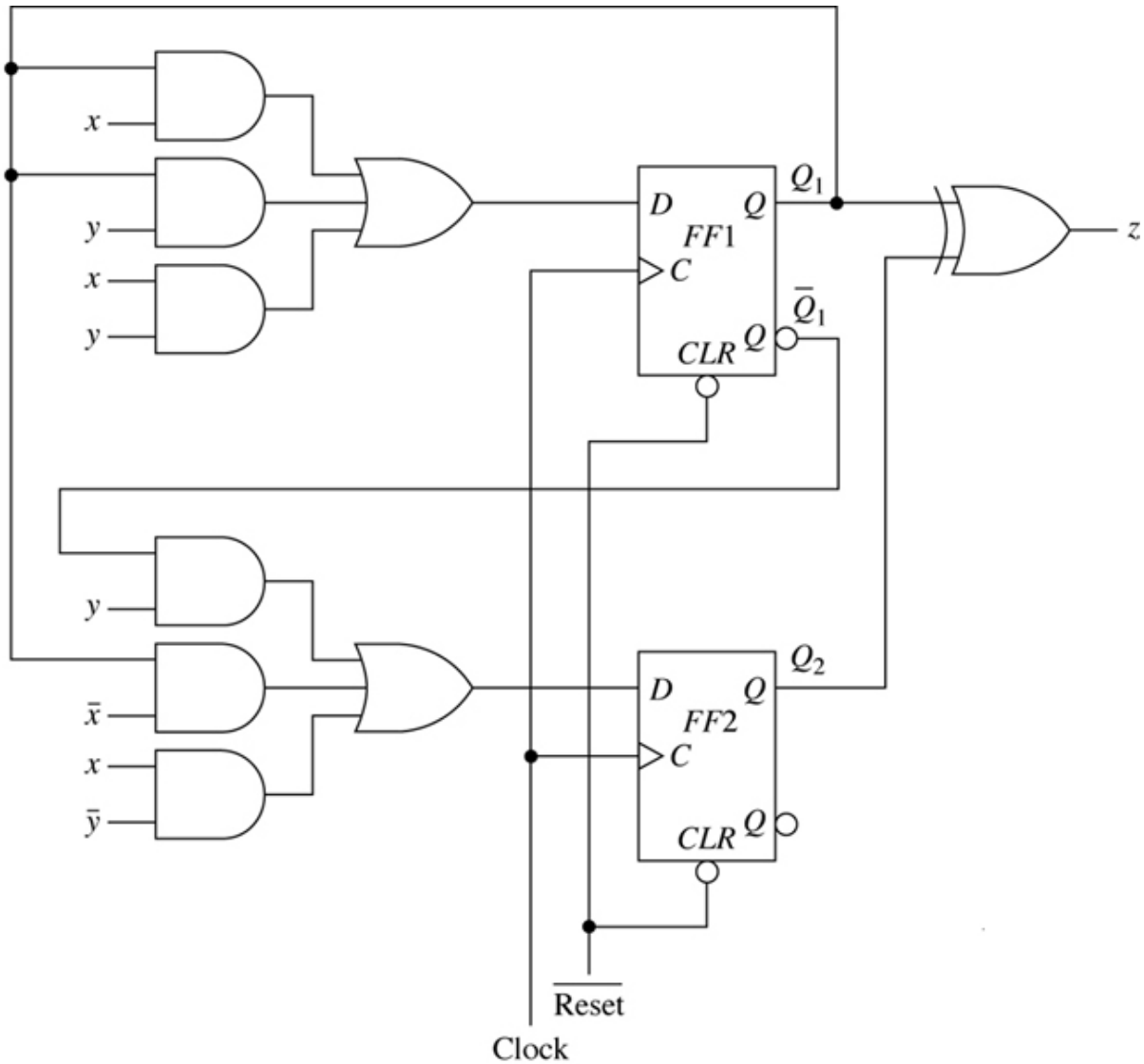
Logic diagram for the excitation table of Table 7.19

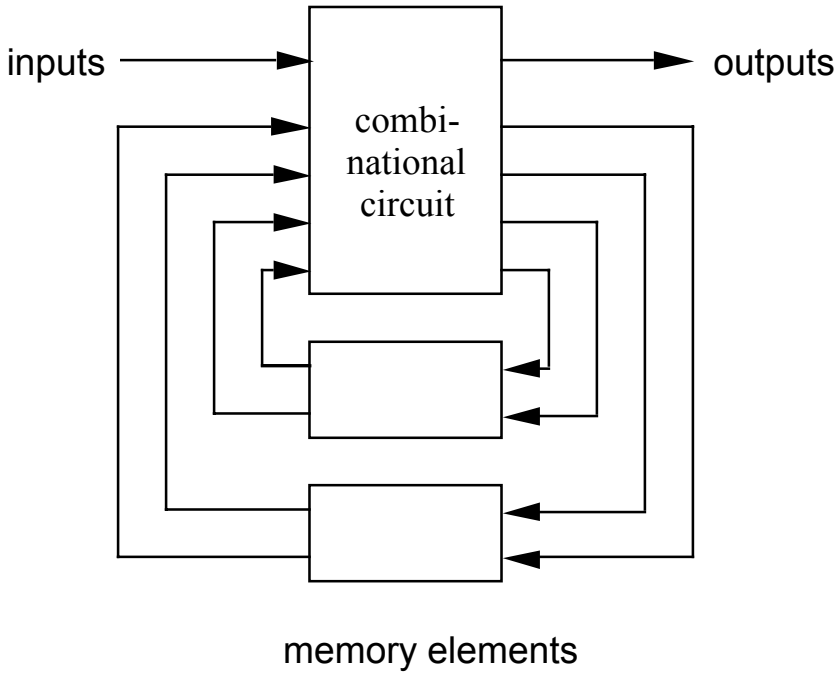


Logic diagram for the excitation table of Table 7.20

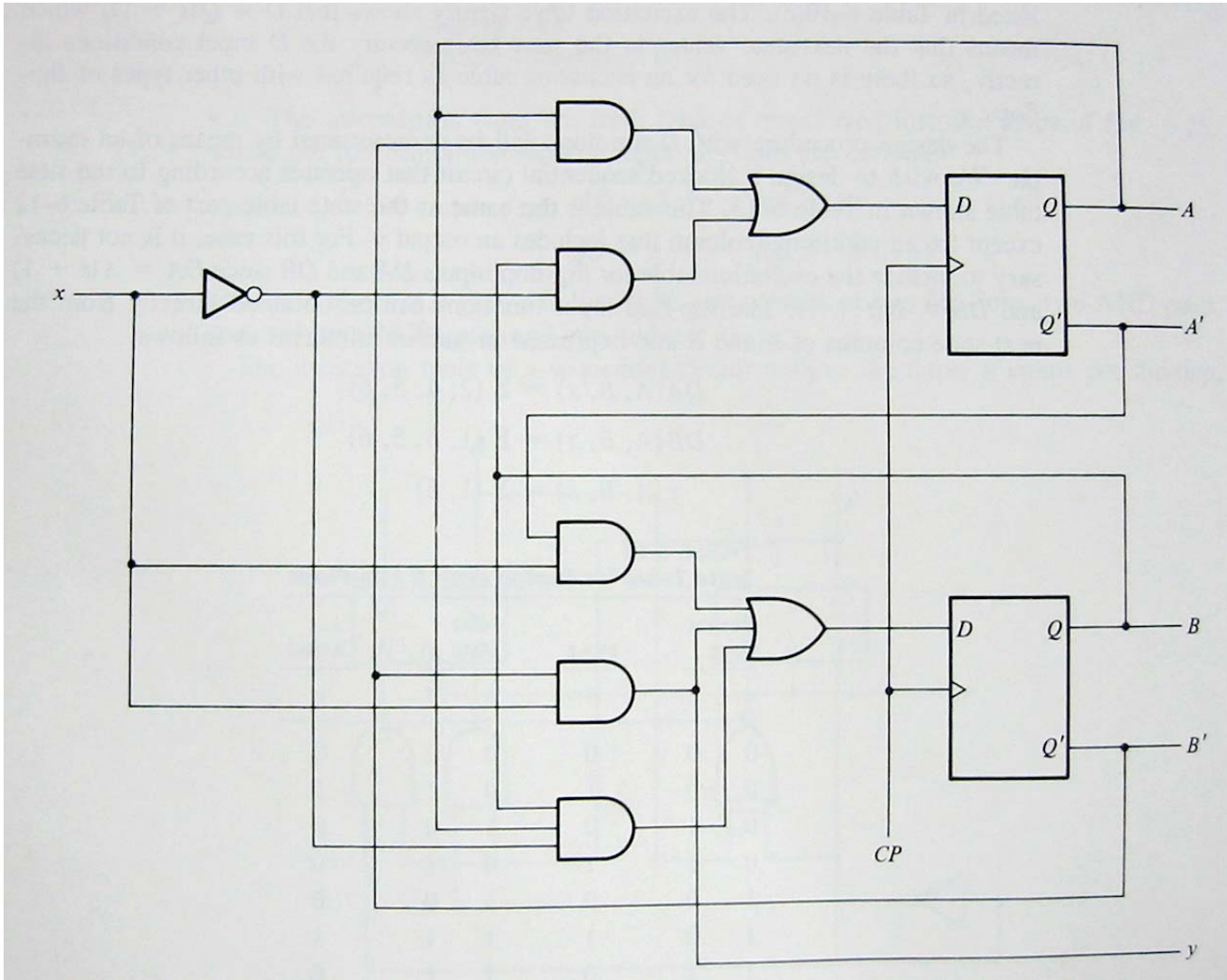


Logic diagram for the Moore serial binary adder

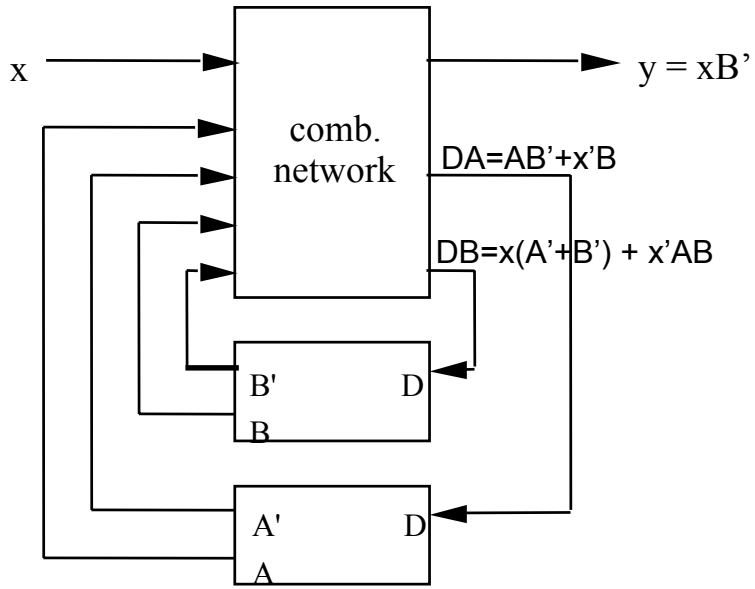




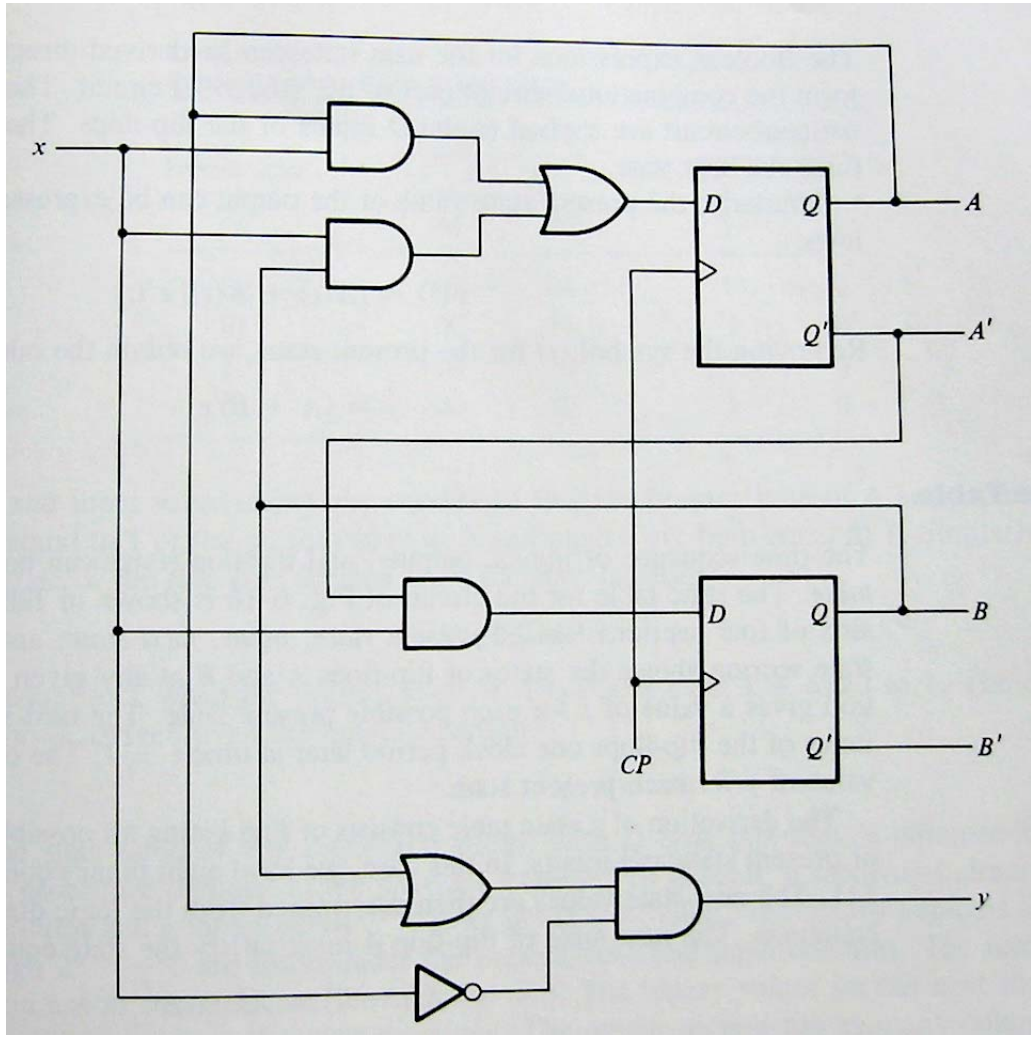
Typical block diagram of a sequential circuit



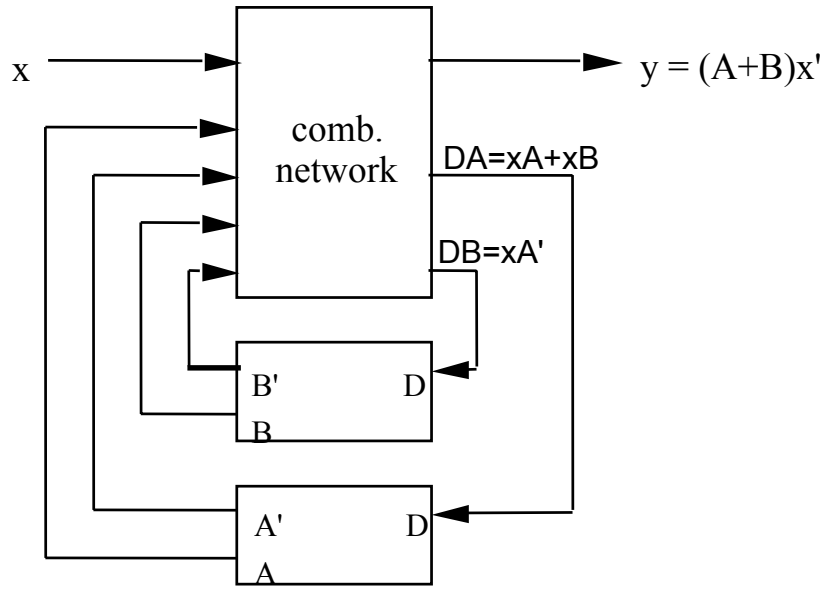
Example sequential circuit 1



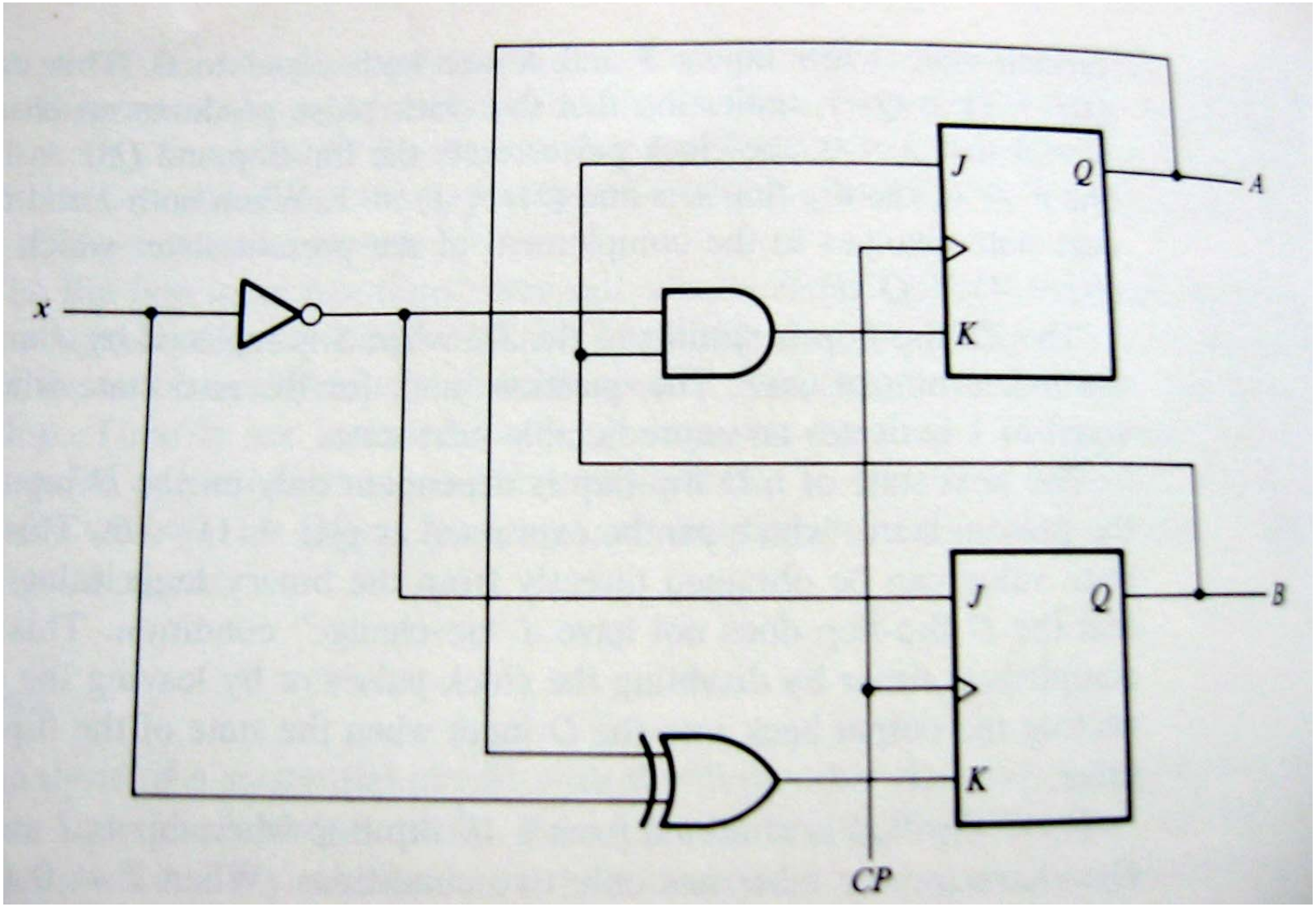
Block diagram of Example 1



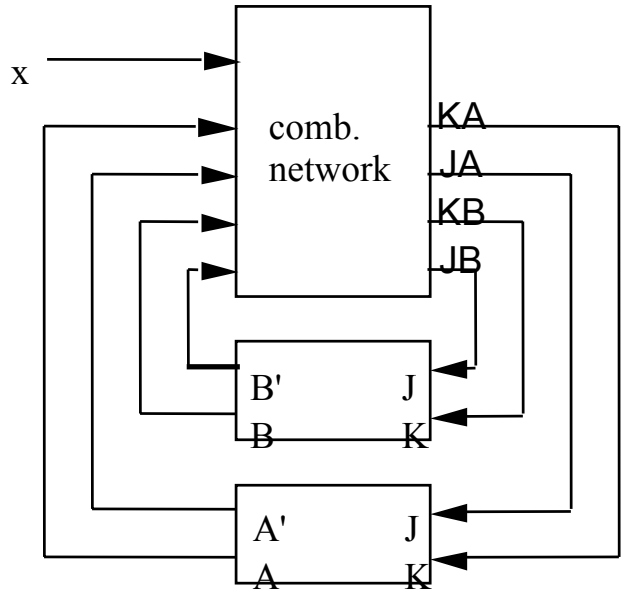
Example sequential circuit 2



Block diagram of Example 2



Example sequential circuit 3



JA = B
KA = Bx'
JB = x'
KB = A'x + Ax'

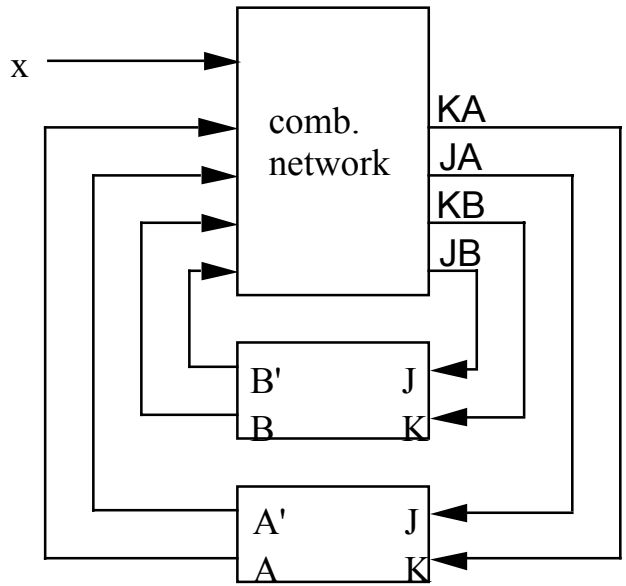
Block diagram of Example 3

present state	next state	
	x = 0	x = 1
A(t)B(t)	A(t+1)B(t+1)	A(t+1)B(t+1)
0 0	?	?
0 1	?	?
1 0	?	?
1 1	?	?

To analyze the preceding circuit is to complete this table

Steps involved:

1. Construct the truth table of the combinational network to determine the output and the input to the flip-flops.
2. Use the characteristic table of the flip-flops to determine the next states.



JA = B
KA = Bx'
JB = x'
KB = A'x + Ax'

Sequential circuit implemented with JK flip-flops (Fig. 6-19)

Step 1: construct the truth table of the combinational network.

x	A	B	JA	KA	JB	KB
0	0	0	0	0	1	0
0	0	1	1	1	1	0
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	0	0	0	1
1	0	1	1	0	0	1
1	1	0	0	0	0	0
1	1	1	1	0	0	0

Step 2: extend the truth table to inclues contents of flip-flops at time t+1.

x	A	B	JA	KA	JB	KB	A(t+1)	B(t+1)
0	0	0	0	0	1	0		
0	0	1	1	1	1	0		
0	1	0	0	0	1	1		
0	1	1	1	1	1	1		
1	0	0	0	0	0	1		
1	0	1	1	0	0	1		
1	1	0	0	0	0	0		
1	1	1	1	0	0	0		

Step 2.1: find A(t+1), with the help of a characteristic table.

x	A	B	JA	KA	JB	KB	A(t+1)	B(t+1)
0	0	0	0	0	1	0	0	
0	0	1	1	1	1	0	1	
0	1	0	0	0	1	1	1	
0	1	1	1	1	1	1	0	
1	0	0	0	0	0	1	0	
1	0	1	1	0	0	1	1	
1	1	0	0	0	0	0	1	
1	1	1	1	0	0	0	1	

The characteristic table of a 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

Step 2.2: find B(t+1), with the help of a characteristic table.

x	A	B	JA	KA	JB	KB	A(t+1)	B(t+1)
0	0	0	0	0	1	0	0	1
0	0	1	1	1	1	0	1	1
0	1	0	0	0	1	1	1	1
0	1	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0	0
1	0	1	1	0	0	1	1	0
1	1	0	0	0	0	0	1	0
1	1	1	1	0	0	0	1	1

The characteristic table of a 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

Step 2.3: final step.

x	A	B	JA	KA	JB	KB	A(t+1)	B(t+1)
0	0	0	0	0	1	0	0	1
0	0	1	1	1	1	0	1	1
0	1	0	0	0	1	1	1	1
0	1	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0	0
1	0	1	1	0	0	1	1	0
1	1	0	0	0	0	0	1	0
1	1	1	1	0	0	0	1	1

Reconstruct the state table to yield

present state	next state	
	x = 0	x = 1
A(t)B(t)	A(t+1)B(t+1)	A(t+1)B(t+1)
0 0	0 1	0 0
0 1	1 1	1 0
1 0	1 1	1 0
1 1	0 0	1 1

Mealy and Moore Models

There are two models of sequential circuit:

Mealy Model: the outputs are functions of both the present states and inputs.

Moore Model: the outputs are a function of the present state only.

Example:

State Reduction

Two states are said to be equivalent if, for each possible single input, 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 in this sense, one of them can be removed without altering the input-output relations.

Design Steps

1. Obtain the word description of the desired circuit behavior.
2. Construct the state table of the desired circuit.
3. Reduce the number of state to the extent possible (state reduction).
4. Assign a bit pattern to each state (state assignment).
5. Determine the no. of flip-flops needed and assign a letter symbol to each.
6. Choose the type of flip-flop to be used.
7. Derive the truth table of the required combinational circuit from the state table.
8. Simplify the combinational circuit.
9. Draw the logic diagram.

Design Example:

Suppose we are given a word description of the desired circuit behavior, and it is translated into the following state table (Step 2):

present state	next state		output	
	x=0	x=1	x=0	x=1
a	f	b	0	0
b	d	c	0	0
c	f	e	0	0
d	g	a	1	0
e	d	c	0	0
f	g	d	0	1
g	g	h	0	1
h	g	a	1	0

State reduction (step 3)

present state	next state		output	
	x=0	x=1	x=0	x=1
a	f	b	0	0
b	d	a ~~e~~	0	0
~~e~~	~~f~~	b ~~e~~	~~0~~	~~0~~
d	f ~~g~~	a	1	0
~~e~~	~~d~~	a ~~e~~	~~0~~	~~0~~
f	f ~~g~~	d	0	1
~~g~~	~~g~~	d ~~h~~	~~0~~	~~1~~
~~h~~	~~g~~	~~a~~	~~1~~	~~0~~

The reduced state table (Step 3):

present state	next state		output	
	x=0	x=1	x=0	x=1
a	f	b	0	0
b	d	a	0	0
d	f	a	1	0
f	f	d	0	1

Design Example:

A possible state assignment (State 4):

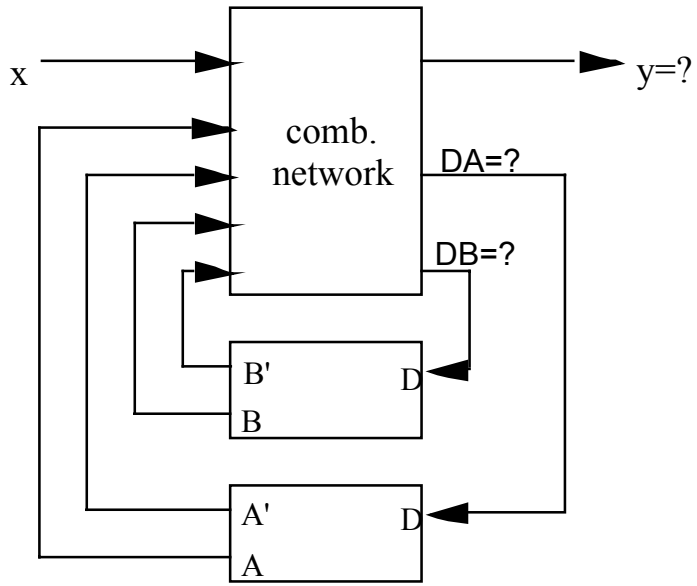
present state	next state		output	
	x=0	x=1	x=0	x=1
a	f	b	0	0
b	d	a	0	0
d	f	a	1	0
f	f	d	0	1

$00 \rightarrow a$ $01 \rightarrow b$ $10 \rightarrow d$ $11 \rightarrow f$

present state	next state		output	
	x=0	x=1	x=0	x=1
A B	A(t+1)B(t+1)	A(t+1)B(t+1)	y(t)	y(t)
0 0	1 1	0 1	0	0
0 1	1 0	0 0	0	0
1 0	1 1	0 0	1	0
1 1	1 1	1 0	0	1

Design Example

This state table can be implemented by a sequential circuit of the form depicted below using D type flip-flops (Steps 5 and 6):



Reconstruct the state table

present state	next state		output	
	x=0	x=1	x=0	x=1
A B	A(t+1)B(t+1)	A(t+1)B(t+1)	y(t)	y(t)
0 0	1 1	0 1	0	0
0 1	1 0	0 0	0	0
1 0	1 1	0 0	1	0
1 1	1 1	1 0	0	1

in preparation for expanding it into a truth table of the combinational network required (Step 7):

x	A(t)	B(t)	A(t+1)	B(t+1)	y(t)
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	1	1	1
0	1	1	1	1	0
1	0	0	0	1	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	1	0	1

Expand the state table into the truth table of the combinational network (Step 7):

x	A(t)	B(t)	A(t+1)	B(t+1)	DA	DB	y(t)
0	0	0	1	1	1		0
0	0	1	1	0	1		0
0	1	0	1	1	1		1
0	1	1	1	1	1		0
1	0	0	0	1	0		0
1	0	1	0	0	0		0
1	1	0	0	0	0		0
1	1	1	1	0	1		1

Excitation table of a D type flip-flop

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

Excitation vs. application table

- An *excitation table* is a table that shows what inputs are needed to make the flip-flops to change the states as desired.
- An *application table* is an excitation table constructed for a particular type of flip-flop.

Expand the state table into the truth table of the combinational network (Step 7):

x	A(t)	B(t)	A(t+1)	B(t+1)	DA	DB	y(t)
0	0	0	1	1	1	1	0
0	0	1	1	0	1	0	0
0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	0
1	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	1	0	1	0	1

Excitation table of a D type flip-flop

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

Simplify the Boolean functions that describe the outputs of the combinational network (Step 8):

	A'B'	A'B	AB	AB'
x'	1	1	1	1
x	0	0	1	0

$DA = x' + AB$

	A'B'	A'B	AB	AB'
x'	1	0	1	1
x	1	0	0	0

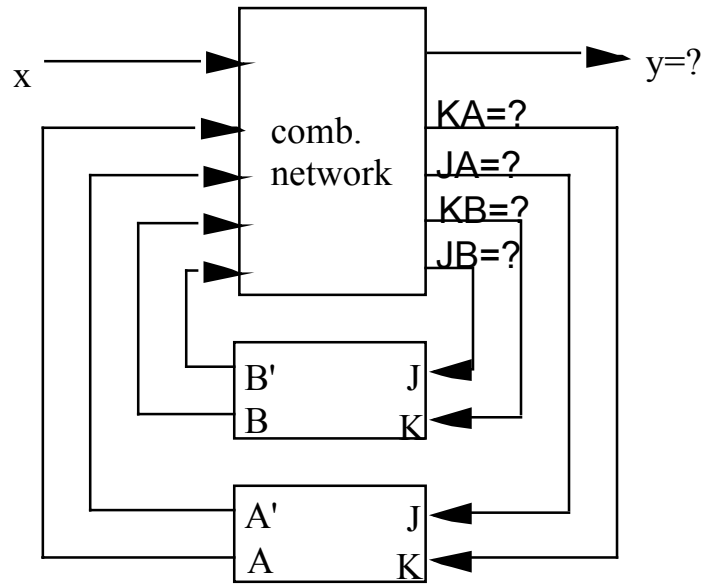
$DB = x'A + A'B'$

	A'B'	A'B	AB	AB'
x'	0	0	0	1
x	0	0	1	0

$y = x'AB' + xAB$

Design Example

This state table can be implemented by a sequential circuit of the form depicted below using JK flip-flops (Steps 5 and 6):



Again, start with the state table:

x	A(t)	B(t)	A(t+1)	B(t+1)	y(t)
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	1	1	1
0	1	1	1	1	0
1	0	0	0	1	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	1	0	1

In preparation for constructing the truth table of the required combinational circuit, expand the state table to include the columns for inputs to the flip-flops:

x	A(t)	B(t)	A(t+1)	B(t+1)	JA	KA	JB	KB	y(t)
0	0	0	1	1					0
0	0	1	1	0					0
0	1	0	1	1					1
0	1	1	1	1					0
1	0	0	0	1					0
1	0	1	0	0					0
1	1	0	0	0					0
1	1	1	1	0					1

With the help of an excitation table find inputs to flip-flop A (Step 7):

x	A(t)	B(t)	A(t+1)	B(t+1)	JA	KA	JB	KB	y(t)
0	0	0	1	1	1	X			0
0	0	1	1	0	1	X			0
0	1	0	1	1	X	0			1
0	1	1	1	1	X	0			0
1	0	0	0	1	0	X			0
1	0	1	0	0	0	X			0
1	1	0	0	0	X	1			0
1	1	1	1	0	X	0			1

Excitation table of a JK flip-flop

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

With the help of an excitation table find inputs to flip-flop B (Step 7):

x	A(t)	B(t)	A(t+1)	B(t+1)	JA	KA	JB	KB	y(t)
0	0	0	1	1	1	X	1	X	0
0	0	1	1	0	1	X	X	1	0
0	1	0	1	1	X	0	1	X	1
0	1	1	1	1	X	0	X	0	0
1	0	0	0	1	0	X	1	X	0
1	0	1	0	0	0	X	X	1	0
1	1	0	0	0	X	1	0	X	0
1	1	1	1	0	X	0	X	1	1

Excitation table of a JK flip-flop

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

Simplify the outputs of the combinational network (Step 8):

	A'B'	A'B	AB	AB'
x'	1	1	X	X
x	0	0	X	X

JA = x'

	A'B'	A'B	AB	AB'
x'	X	X	0	0
x	X	X	0	1

KA = xB'

	A'B'	A'B	AB	AB'
x'	1	X	X	1
x	1	X	X	0

$JB = x' + A'$

	A'B'	A'B	AB	AB'
x'	X	1	0	X
x	X	1	1	X

$KB = x + A'$

	A'B'	A'B	AB	AB'
x'	0	0	0	1
x	0	0	1	0

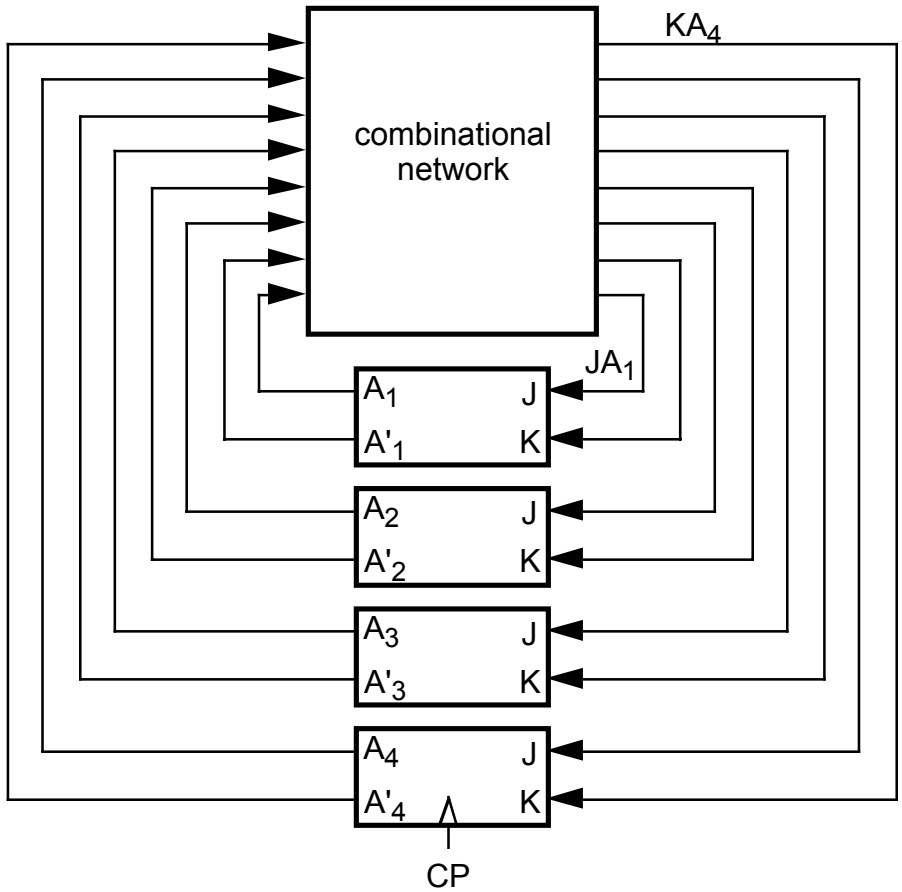
$y = x'AB' + xAB$

Design of a 4-bit synchronous up counter

The state table

present state (at time t)	next state (at t+1)
$A_4 A_3 A_2 A_1$	$A_4 A_3 A_2 A_1$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 0
0 0 1 0	0 0 1 1
0 0 1 1	0 1 0 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

The block diagram



The excitation function for the four JK flip-flops

present state (at time t)	next state (at t+1)	JA_1	KA_1	JA_2	KA_2	JA_3	KA_3	JA_4	KA_4
$A_4 A_3 A_2 A_1$	$A_4 A_3 A_2 A_1$								
0 0 0 0	0 0 0 1	1	X	0	X	0	X	0	X
0 0 0 1	0 0 1 0	X	1	1	X	0	X	0	X
0 0 1 0	0 0 1 1	1	X	X	0	0	X	0	X
0 0 1 1	0 1 0 0	X	1	X	1	1	X	0	X
0 1 0 0	0 1 0 1	1	X	0	X	X	0	0	X
0 1 0 1	0 1 1 0	X	1	1	X	X	0	0	X
0 1 1 0	0 1 1 1	1	X	X	0	X	0	0	X
0 1 1 1	1 0 0 0	X	1	X	1	X	1	1	X
1 0 0 0	1 0 0 1	1	X	0	X	0	X	X	0
1 0 0 1	1 0 1 0	X	1	1	X	0	X	X	0
1 0 1 0	1 0 1 1	1	X	X	0	0	X	X	0
1 0 1 1	1 1 0 0	X	1	X	1	1	X	X	0
1 1 0 0	1 1 0 1	1	X	0	X	X	0	X	0
1 1 0 1	1 1 1 0	X	1	1	X	X	0	X	0
1 1 1 0	1 1 1 1	1	X	X	0	X	0	X	0
1 1 1 1	0 0 0 0	X	1	X	1	X	1	X	1

From the truth table we see that the desired inputs to the flip-flops can be simplified to

$JA_1 = 1$

$KA_1 = 1$

$JA_2 = A_1$

$KA_2 = A_1$

$JA_3 = A_2A_1$

$KA_3 = A_2A_1$

$JA_4 = A_3A_2A_1$

$KA_4 = A_3A_2A_1$

and hence the logic diagram shown in Fig. 7-17.

Note that if we let $JA_1 = KA_1 = 0$ then none of the flip-flop will change its state, and therefore we can use it to stop (i.e., to disable) the counter.

Design of a 4-bit synchronous down counter

The state table

present state (at time t)	next state (at t+1)
$A_4 A_3 A_2 A_1$	$A_4 A_3 A_2 A_1$
0 0 0 0	1 1 1 1
0 0 0 1	0 0 0 0
0 0 1 0	0 0 0 1
0 0 1 1	0 0 1 0
0 1 0 0	0 0 1 1
0 1 0 1	0 1 0 0
0 1 1 0	0 1 0 1
0 1 1 1	0 1 1 0
1 0 0 0	0 1 1 1
1 0 0 1	1 0 0 0
1 0 1 0	1 0 0 1
1 0 1 1	1 0 1 0
1 1 0 0	1 0 1 1
1 1 0 1	1 1 0 0
1 1 1 0	1 1 0 1
1 1 1 1	1 1 1 0

The excitation function for the four JK flip-flops

present state (at time t)	next state (at t+1)	JA_1	KA_1	JA_2	KA_2	JA_3	KA_3	JA_4	KA_4
$A_4\,A_3\,A_2\,A_1$	$A_4\,A_3\,A_2\,A_1$								
0 0 0 0	1 1 1 1	1	X	1	X	1	X	1	X
0 0 0 1	0 0 0 0	X	1	0	X	0	X	0	X
0 0 1 0	0 0 0 1	1	X	X	1	0	X	0	X
0 0 1 1	0 0 1 0	X	1	X	0	0	X	0	X
0 1 0 0	0 0 1 1	1	X	1	X	X	1	0	X
0 1 0 1	0 1 0 0	X	1	0	X	X	0	0	X
0 1 1 0	0 1 0 1	1	X	X	1	X	0	0	X
0 1 1 1	0 1 1 0	X	1	X	0	X	0	0	X
1 0 0 0	0 1 1 1	1	X	1	X	1	X	X	1
1 0 0 1	1 0 0 0	X	1	0	X	0	X	X	0
1 0 1 0	1 0 0 1	1	X	X	1	0	X	X	0
1 0 1 1	1 0 1 0	X	1	X	0	0	X	X	0
1 1 0 0	1 0 1 1	1	X	1	X	X	1	X	0
1 1 0 1	1 1 0 0	X	1	0	X	X	0	X	0
1 1 1 0	1 1 0 1	1	X	X	1	X	0	X	0
1 1 1 1	1 1 1 0	X	1	X	0	X	0	X	0

From the truth table we see that the desired inputs to the flip-flops can be simplified to

$JA_1 = 1$

$KA_1 = 1$

$JA_2 = A'_1$

$KA_2 = A'_1$

$JA_3 = A'_2A'_1$

$KA_3 = A'_2A'_1$

$JA_4 = A'_3A'_2A'_1$

$KA_4 = A'_3A'_2A'_1$

This is reflected in the logic diagram shown in Fig. 7-18. Note that this design can be directly translated into a T flip-flop implementation because the J input to every flip-flop is identical to its K input.

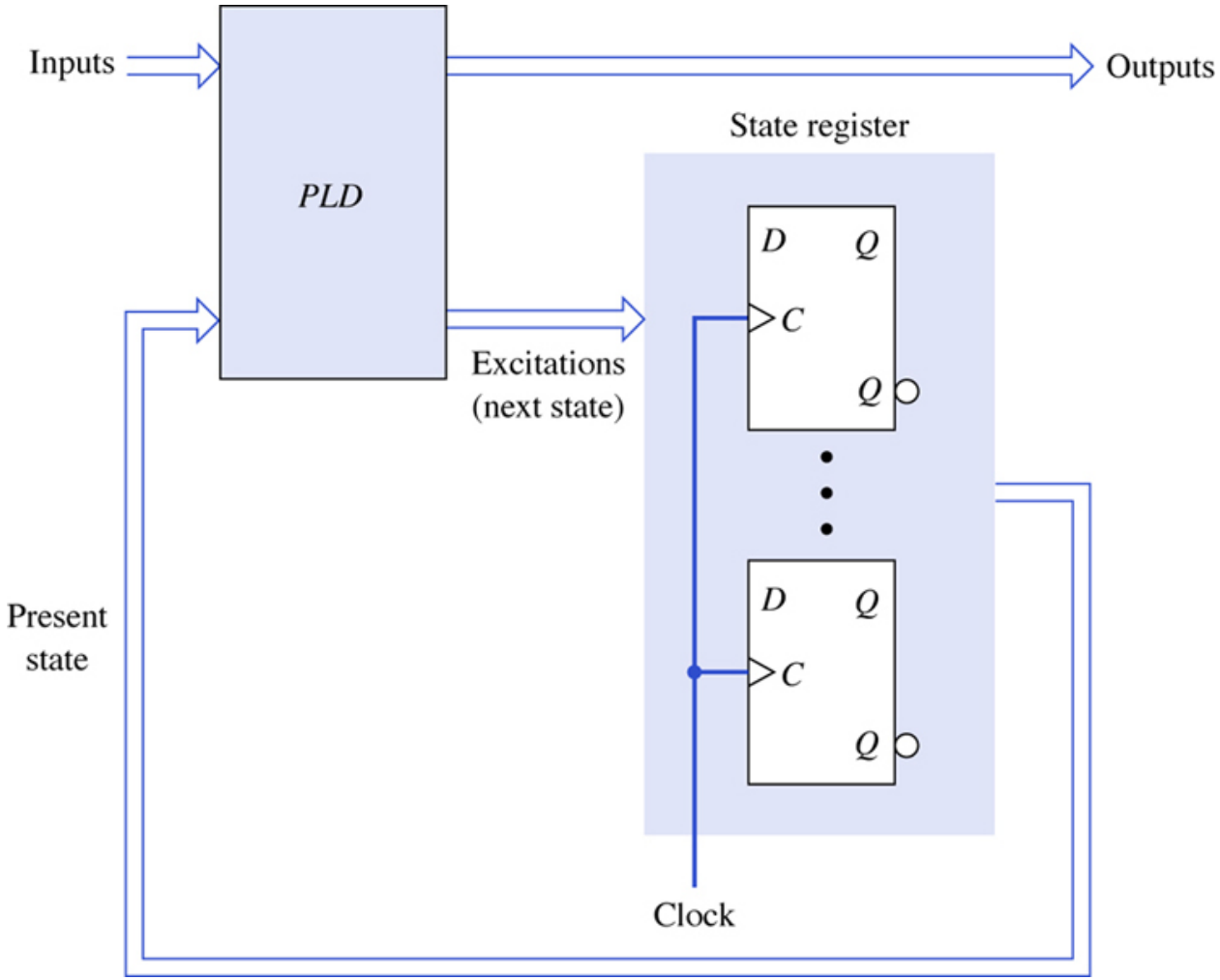
Design verification

- As the last step in design, analyze the resulting circuit to make sure that it realizes the given state table.
- If there are unused states, make sure that none of them have undesirable properties. For example, at a very minimum, none of them should be an isolated state.

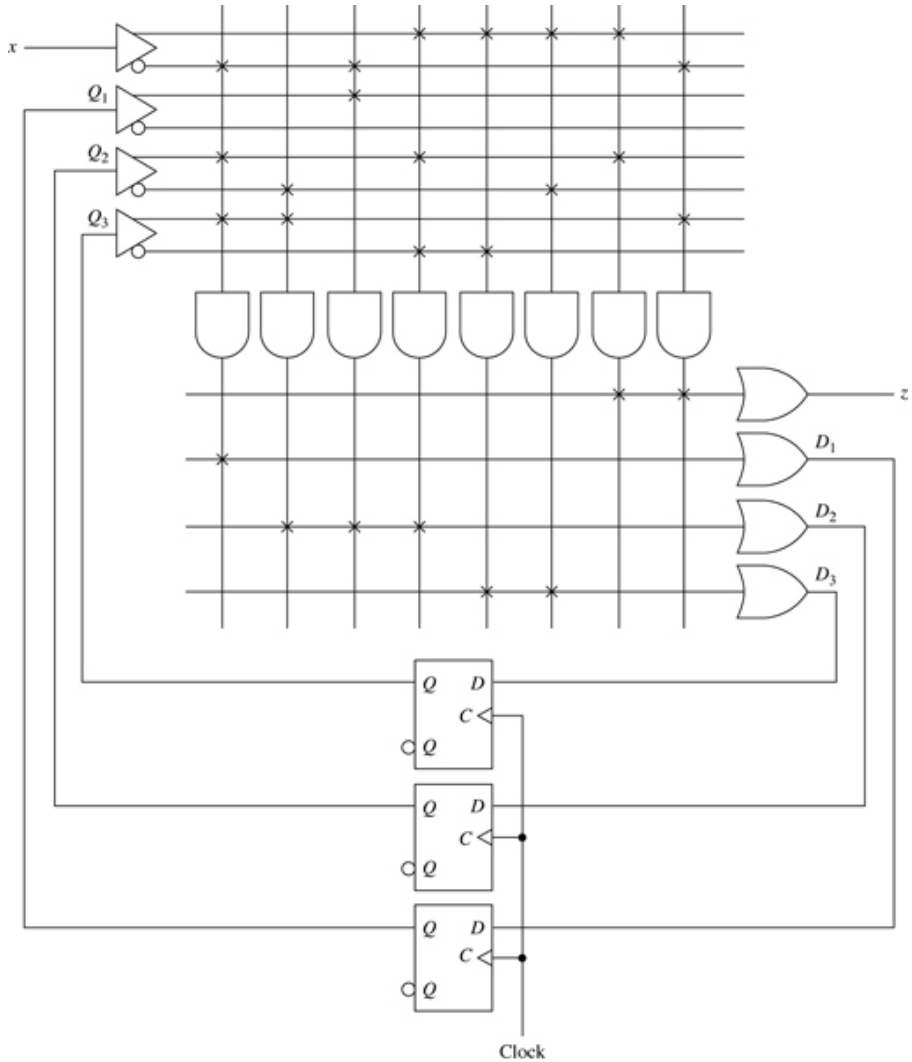
Isolated unused states

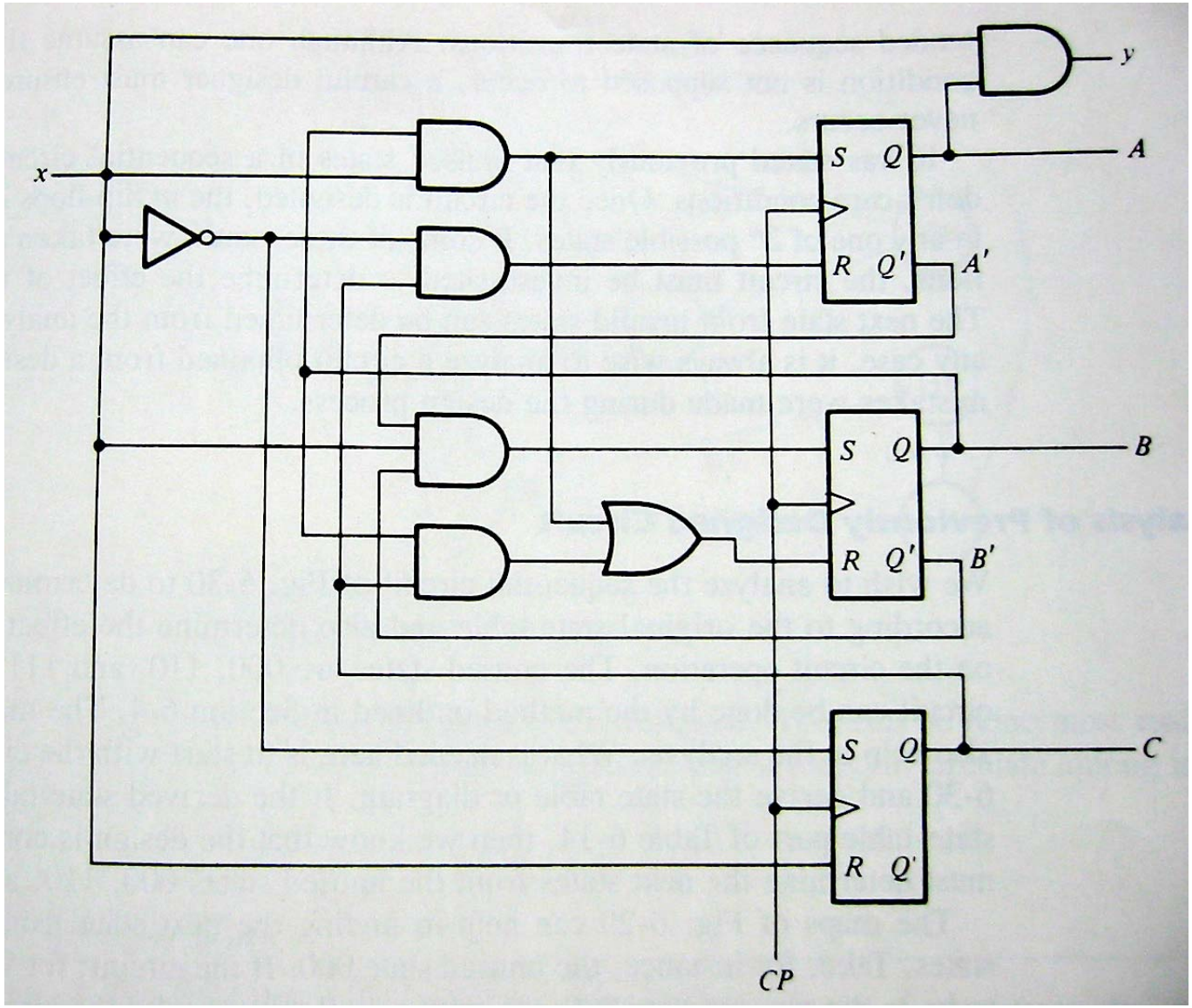
- We need to pay attention to the unused state because when the power is turned on, or noise interfere, a sequential circuit might enter an unused state.
- If the unused states are isolated, we will not be able to make the circuit to perform useful function.
- Requirements on unused states differ.

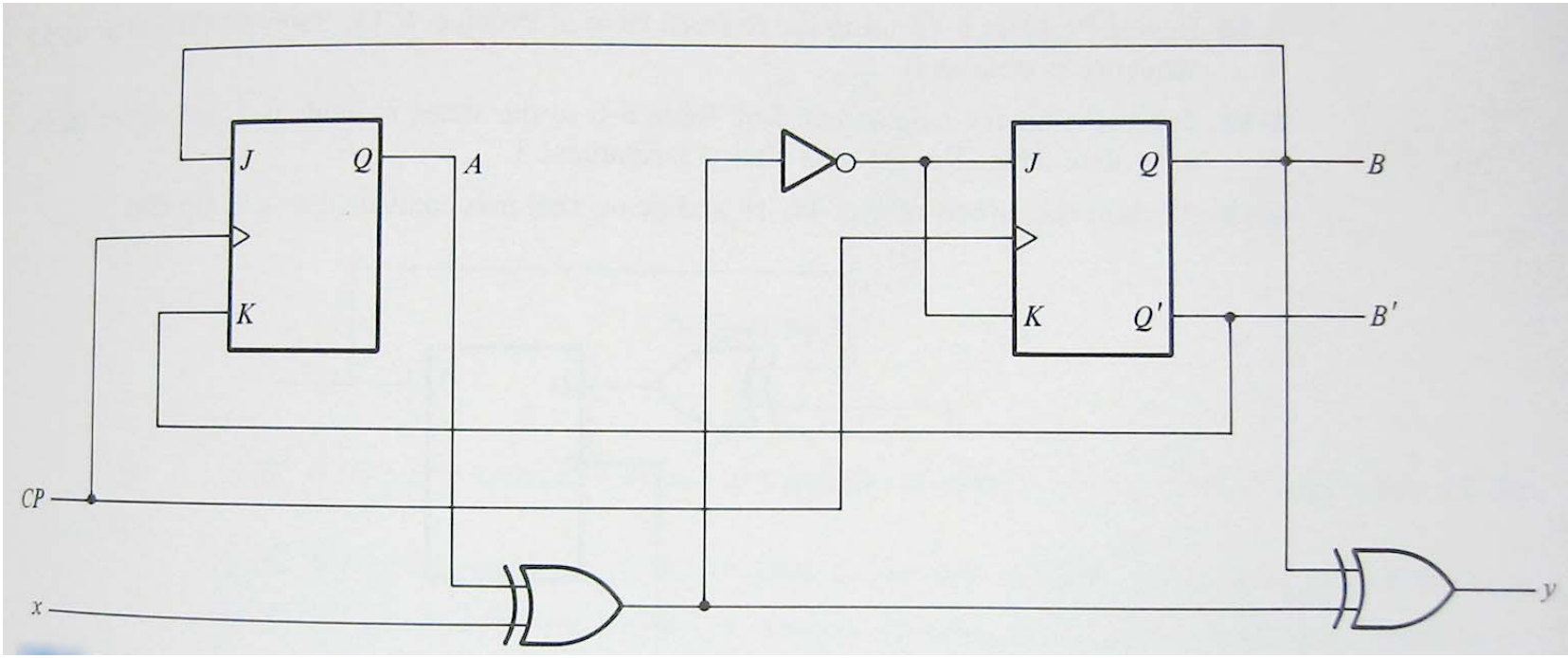
General structure of a clocked sequential network realization using a PLD and clocked *D* flip-flops



A clocked synchronous sequential network realization using a PLA and clocked *D* flip-flops




x
y
A
A'
B
B'
C
S
Q
R
Q'
CP


J
Q
A
K
J
Q
B
K
Q'
B'
CP
x
y