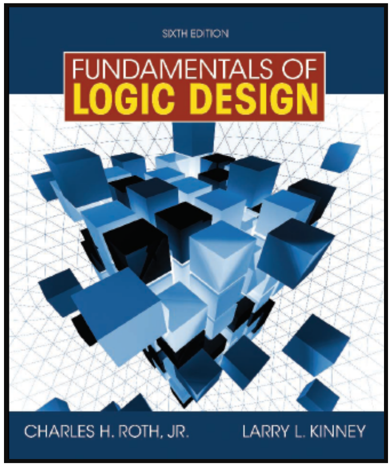
SLIDES FOR

CHAPTER 15

REDUCTION OF STATE TABLES
STATE ASSIGNMENT



This chapter in the book includes:

Click the mouse to move to the next page.
Use the ESC key to exit this chapter.

Elimination of Redundant States

In Unit 14, we were careful to avoid introducing unnecessary states when setting up a state graph or table. We will now approach the problem of deriving the state graph somewhat differently. When first setting up the state table, we will not be overly concerned with inclusion of extra states, and when the table is complete, we will eliminate any redundant states.

We will rework Example 1 in Section 14.3 using excess states, and then eliminate the excess states.

Section 15.1 (p. 474)

Table 15-1: State Table for Sequence Detector

Input Sequence	Present State	Next State $X = 0$	Next State $X = 1$	Present Output $X = 0$	Present Output $X = 1$
reset	*A*	*B*	*C*	0	0
0	*B*	*D*	*E*	0	0
1	*C*	*F*	*G*	0	0
00	*D*	*H*	*I*	0	0
01	*E*	*J*	*K*	0	0
10	*F*	*L*	*M*	0	0
11	*G*	*N*	*P*	0	0
000	*H*	*A*	*A*	0	0
001	*I*	*A*	*A*	0	0
010	*J*	*A*	*A*	0	1
011	*K*	*A*	*A*	0	0
100	*L*	*A*	*A*	0	1
101	*M*	*A*	*A*	0	0
110	*N*	*A*	*A*	0	0
111	*P*	*A*	*A*	0	0

Table 15-2. State Table for Sequence Detector

Since states *H* and *I* have the same next states and the same outputs, there is no way of telling states *H* and *I* apart, and we can replace *I* with *H*.

Present State	Next State		Present Output	
	X = 0	X = 1	X = 0	X = 1
A	B	C	0	0
B	D	E	0	0
C	~~F~~ E	~~G~~ D	0	0
D	H	~~I~~ H	0	0
E	J	~~K~~ H	0	0
~~F~~	~~L J~~	~~M H~~	~~0~~	~~0~~
~~G~~	~~N H~~	~~P H~~	~~0~~	~~0~~
H	A	A	0	0
~~I~~	~~A~~	~~A~~	~~0~~	~~0~~
J	A	A	0	1
~~K~~	~~A~~	~~A~~	~~0~~	~~0~~
~~L~~	~~A~~	~~A~~	~~0~~	~~1~~
~~M~~	~~A~~	~~A~~	~~0~~	~~0~~
~~N~~	~~A~~	~~A~~	~~0~~	~~0~~
~~P~~	~~A~~	~~A~~	~~0~~	~~0~~

Present State	Next State		Output	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
A	B	C	0	0
B	D	E	0	0
C	E	D	0	0
D	H	H	0	0
E	J	H	0	0
H	A	A	0	0
J	A	A	0	1

(a)

Figure 15-1a: **Reduced State Table for Sequence Detector**

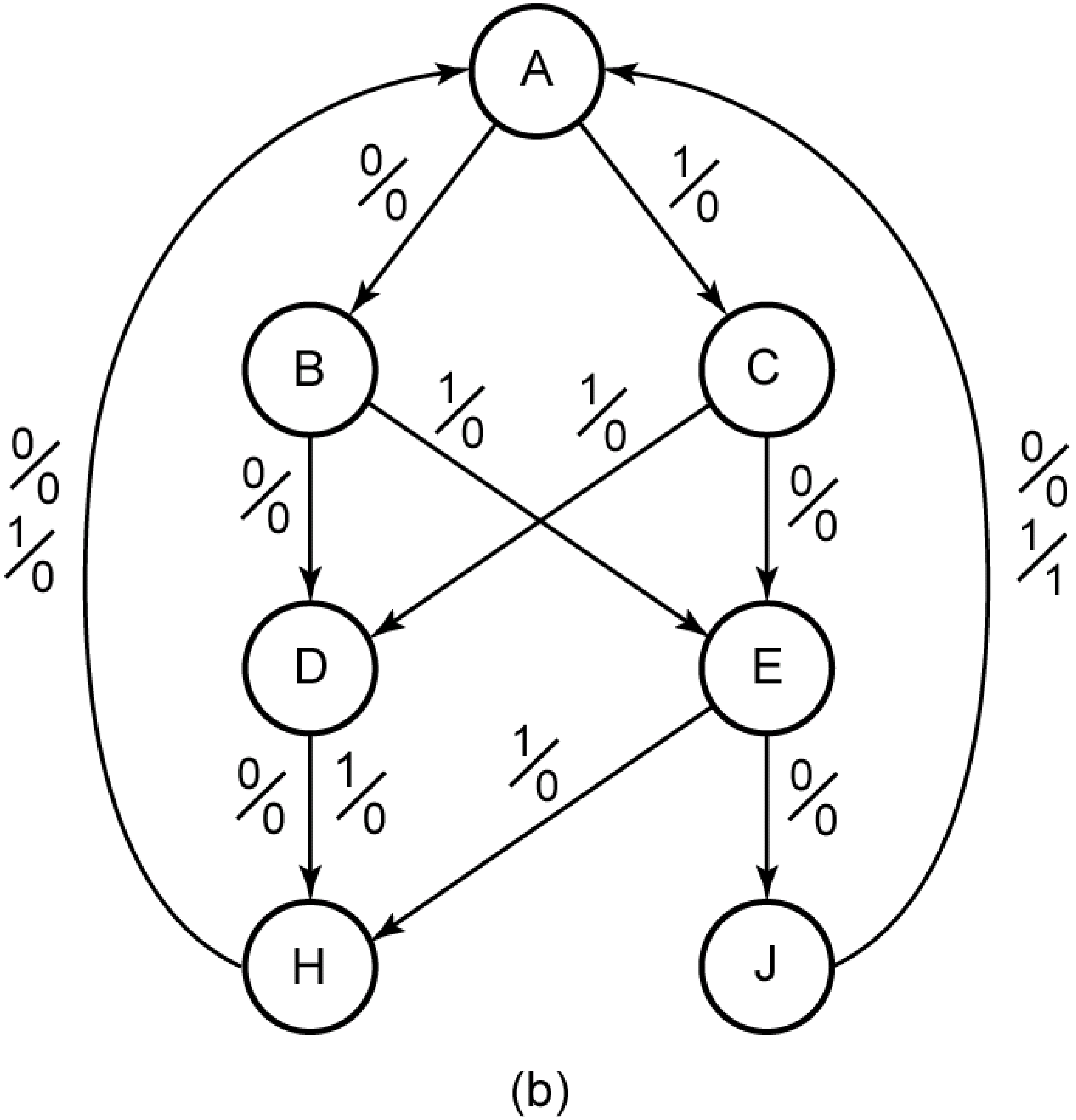


Figure 15-1b: **Reduced State Graph for Sequence Detector**

Equivalent States

Definition 15.1:

Let N_1 and N_2 be sequential circuits (not necessarily different). Let $\underline{X}$ represent a sequence of inputs of arbitrary length. Then state p in N_1 is equivalent to state q in N_2 iff $\lambda_1(p, \underline{X}) = \lambda_2(q, \underline{X})$ for every possible input sequence $\underline{X}$.

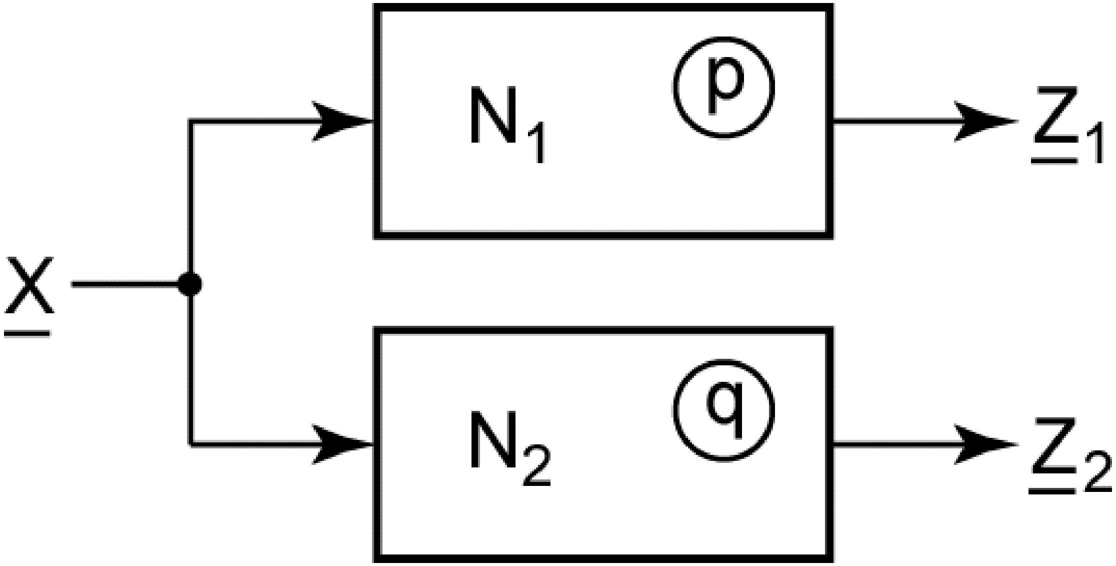


Figure 15-2

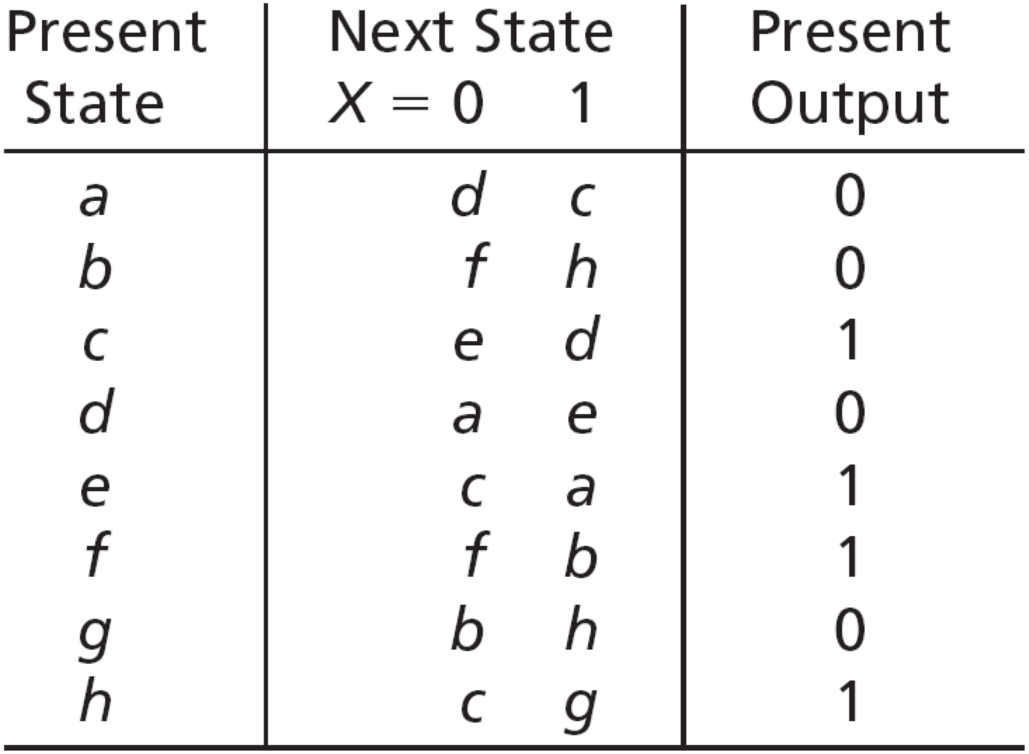
Theorem 15.1: Two states *p* and *q* of a sequential circuit are equivalent iff for every single input *X*, the outputs are the same and the next states are equivalent, that is,

$$\lambda(p, X) = \lambda(q, X) \qquad \text{and} \qquad \delta(p, X) \equiv \delta(q, X)$$

where $\lambda(p, X)$ is the output given the present state *p* and input *X*, and $\delta(p, X)$ is the next state given the present state *p* and input *X*. Note that the next states do not have to be equal, just equivalent.

Determination of State Equivalence

Table 15-3.



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

$a \equiv b$ iff

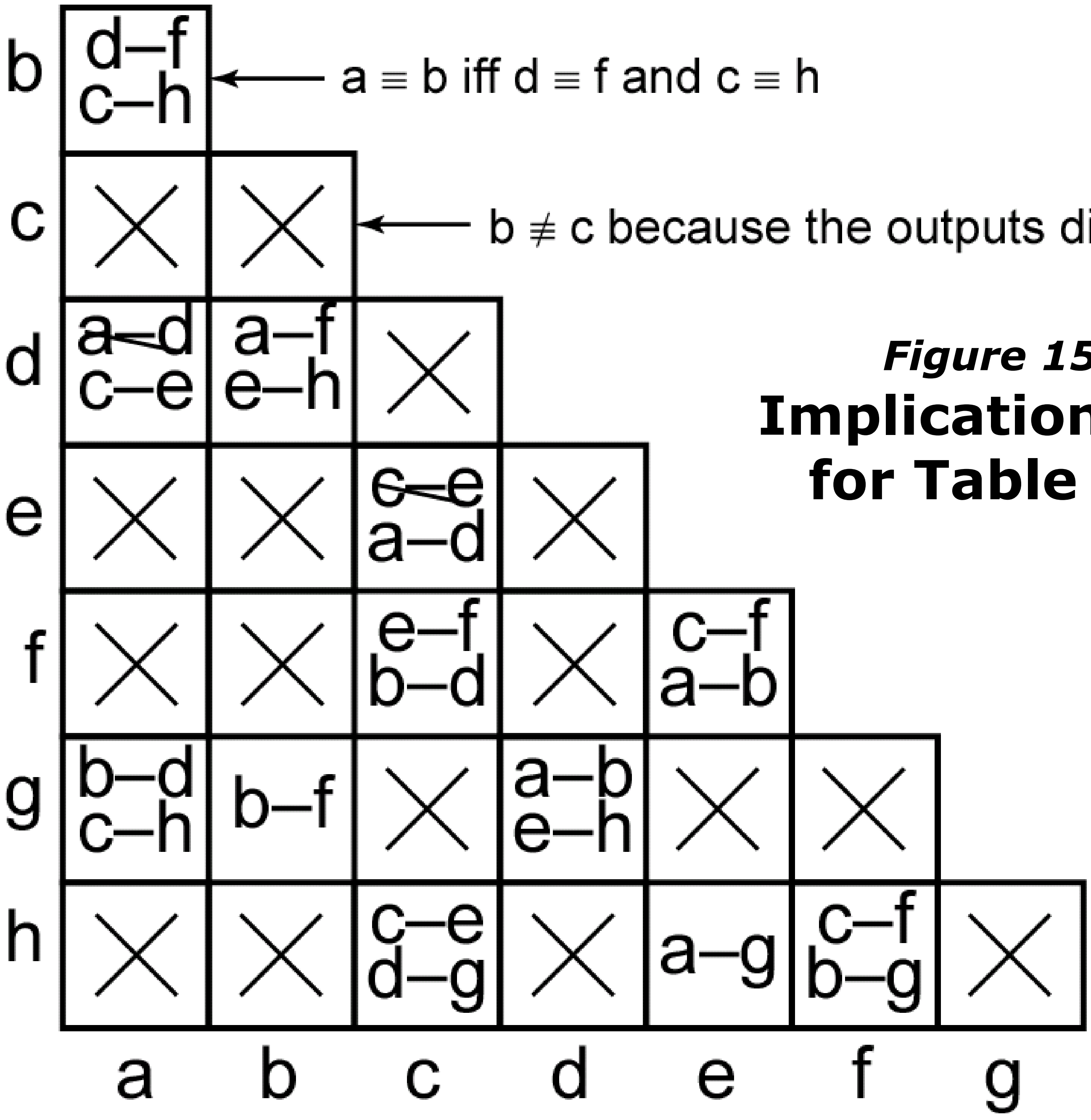
$d \equiv f$ and $c \equiv h$

$a \equiv d$ iff

$a \equiv d$ and $c \equiv e$

Figure 15-3:

Implication Chart for Table 15-3



	a	b	c	d	e	f	g
b	d–f c–h						
c	✗	✗					
d	~~a–d~~ c–e	a–f e–h	✗				
e	✗	✗	~~c–e~~ a–d	✗			
f	✗	✗	e–f b–d	✗	c–f a–b		
g	b–d c–h	b–f	✗	a–b e–h	✗	✗	
h	✗	✗	c–e d–g	✗	a–g	c–f b–g	✗

a ≡ b iff d ≡ f and c ≡ h

b ≢ c because the outputs differ

Figure 15-4:

Implication Chart After First Pass

	a	b	c	d	e	f	g
b	~~d–f~~ ~~c–h~~						
c	×	×					
d	c–e	~~a–f~~ ~~e–h~~	×				
e	×	×	a–d	×			
f	×	×	~~e–f~~ ~~b–d~~	×	~~c–f~~ ~~a–b~~		
g	b–d c–h	~~b–f~~	×	~~a–b~~ ~~e–h~~	×	×	
h	×	×	c–e d–g	×	a–g	~~c–f~~ ~~b–g~~	×

Figure 15-5:
Implication Chart After Second Pass

	a	b	c	d	e	f	g
b	~~d–f~~ ~~c–h~~						
c	✗	✗					
d	c–e	~~a–f~~ ~~e–h~~	✗				
e	✗	✗	a–d	✗			
f	✗	✗	~~e–f~~ ~~b–d~~	✗	~~c–f~~ ~~a–b~~		
g	~~b–d~~ ~~c–h~~	~~b–f~~	✗	~~a–b~~ ~~e–h~~	✗	✗	
h	✗	✗	~~c–e~~ ~~d–g~~	✗	~~a–g~~	~~c–f~~ ~~b–g~~	✗

If we replace *d* with *a* and *e* with *c* in Table 15-3, we can eliminate rows *d* and *e*, and the table reduces to six rows:

Table 15-4.

Present State	Next State $X = 0$	1	Output
a	*a*	*c*	0
b	*f*	*h*	0
c	*c*	*a*	1
f	*f*	*b*	1
g	*b*	*h*	0
h	*c*	*g*	1

Implication Table Method

The implication table method of determining state equivalence can be summarized as follows:

1. Construct a chart which contains a square for each pair of states.
2. Compare each pair of rows in the state table. If the outputs associated with states *i* and *j* are different, place an X in square *i-j* to indicate that $i \neq j$. If the outputs are the same, place the implied pairs in square *i-j*. (If the next states of *i* and *j* are *m* and *n* for some input *x*, then *m-n* is an implied pair.) If the outputs and next states are the same (or if *i-j* only implies itself), place a check (√) in square *i-j* to indicate that $i \equiv j$.

Section 15.3 (p. 481)

3. Go through the table square-by-square. If square *i-j* contains the implied pair *m-n*, and square *m-n* contains an X, then $i \not\equiv j$, and an X should be placed in square *i-j*.
4. If any X's were added in step 3, repeat step 3 until no more X's are added.
5. For each square *i-j* which does not contain an X, $i \equiv j$.

Equivalent Sequential Circuits

Definition 15.2:

Sequential circuit N_1 is equivalent to sequential circuit N_2 if for each state p in N_1, there is a state q in N_2 such that $p \equiv q$, and conversely, for each state s in N_2, there is a state t in N_1 such that $s \equiv t$.

Section 15.4 (p. 481)

	N_1			
	$X = 0$	1	$X = 0$	1
A	B	A	0	0
B	C	D	0	1
C	A	C	0	1
D	C	B	0	0

	N_2			
	$X = 0$	1	$X = 0$	1
S_0	S_3	S_1	0	1
S_1	S_3	S_0	0	0
S_2	S_0	S_2	0	0
S_3	S_2	S_3	0	1

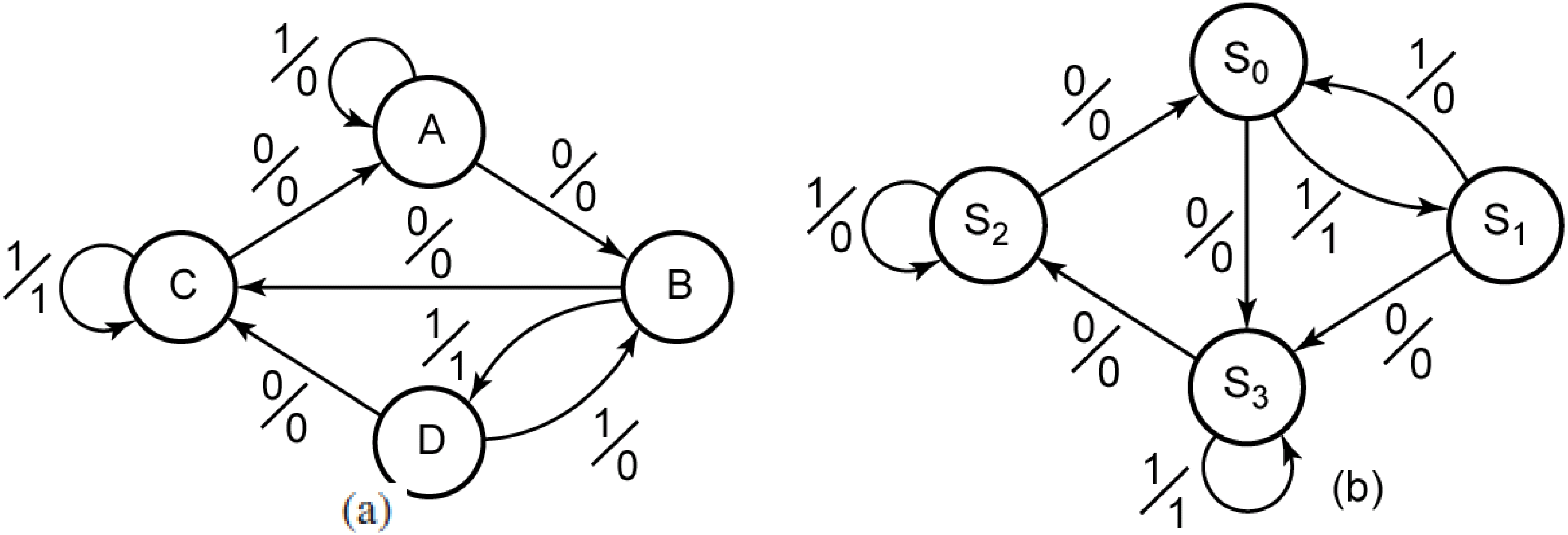


Figure 15-6: **Graphs for Equivalent Circuits**

	A	B	C	D
S_0	X	C–S_3 D–S_1	A–S_3 C–S_1	X
S_1	B–S_3 A–S_0	X	X	C–S_3 B–S_0
S_2	B–S_0 A–S_2	X	X	C–S_0 B–S_2
S_3	X	C–S_2 D–S_3	A–S_2 C–S_3	X

(a)

	A	B	C	D
S_0	X	C–S_3 D–S_1	~~A–S_3~~ ~~C–S_1~~	X
S_1	~~B–S_3~~ ~~A–S_0~~	X	X	C–S_3 B–S_0
S_2	B–S_0 A–S_2	X	X	~~C–S_0~~ ~~B–S_2~~
S_3	X	~~C–S_2~~ ~~D–S_3~~	A–S_2 C–S_3	X

(b)

Figure 15-7: **Implication Tables for Determining Circuit Equivalence**

Incompletely Specified State Tables

Assume that circuit *A* can only generate two possible output sequences, *X* = 100 and *X* = 110. Thus, the sequential circuit subsystem (*B*) has only two possible input sequences. Also, *C* only reads *Z* during everything third input.

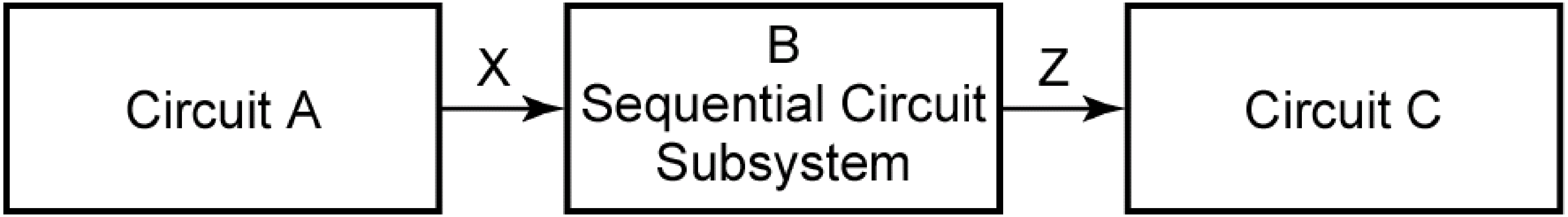


Figure 15-8

	t_0	t_1	t_2		t_0	t_1	t_2
$X =$	1	0	0	$Z =$	–	–	0
	1	1	0		–	–	1

(– is a don’t-care output)

Table 15-5. Incompletely Specified State Table

	$X = 0$	1	0	1
S_0	-	S_1	-	-
S_1	S_2	S_3	-	-
S_2	S_0	-	0	-
S_3	S_0	-	1	-

(a)

	$X = 0$	1	0	1
S_0	(S_0)	S_1	(0)	-
S_1	~~S_2~~ S_0	S_3	(1)	-
S_2	S_0	(S_1)	0	-
S_3	S_0	(S_3)	1	-

(b)

$S_0 \equiv S_2,\ S_1 \equiv S_3$

(c)

	$X = 0$	1	0	1
S_0	S_0	S_1	0	-
S_1	S_0	S_1	1	-

Derivation of Flip-Flop Input Equations

After the number of states in a state table has been reduced, the following procedure can be used to derive the flip-flop input equations:

1. Assign flip-flop state values to correspond to the states in the reduced table.
2. Construct a transition table which gives the next states of the flip-flops as a function of the present states and inputs.
3. Derive the next-state maps from the transition table.
4. Find flip-flop input maps from the next-state maps using the techniques developed in Unit 12 and find the flip-flop input equations from the maps.

Section 15.6 (p. 484)

1. Assign flip-flop state values to correspond to the states in the reduced table.
2. Construct a transition table which gives the next states of the flip-flops as a function of the present states and inputs.

Table 15-6.

(a) State table

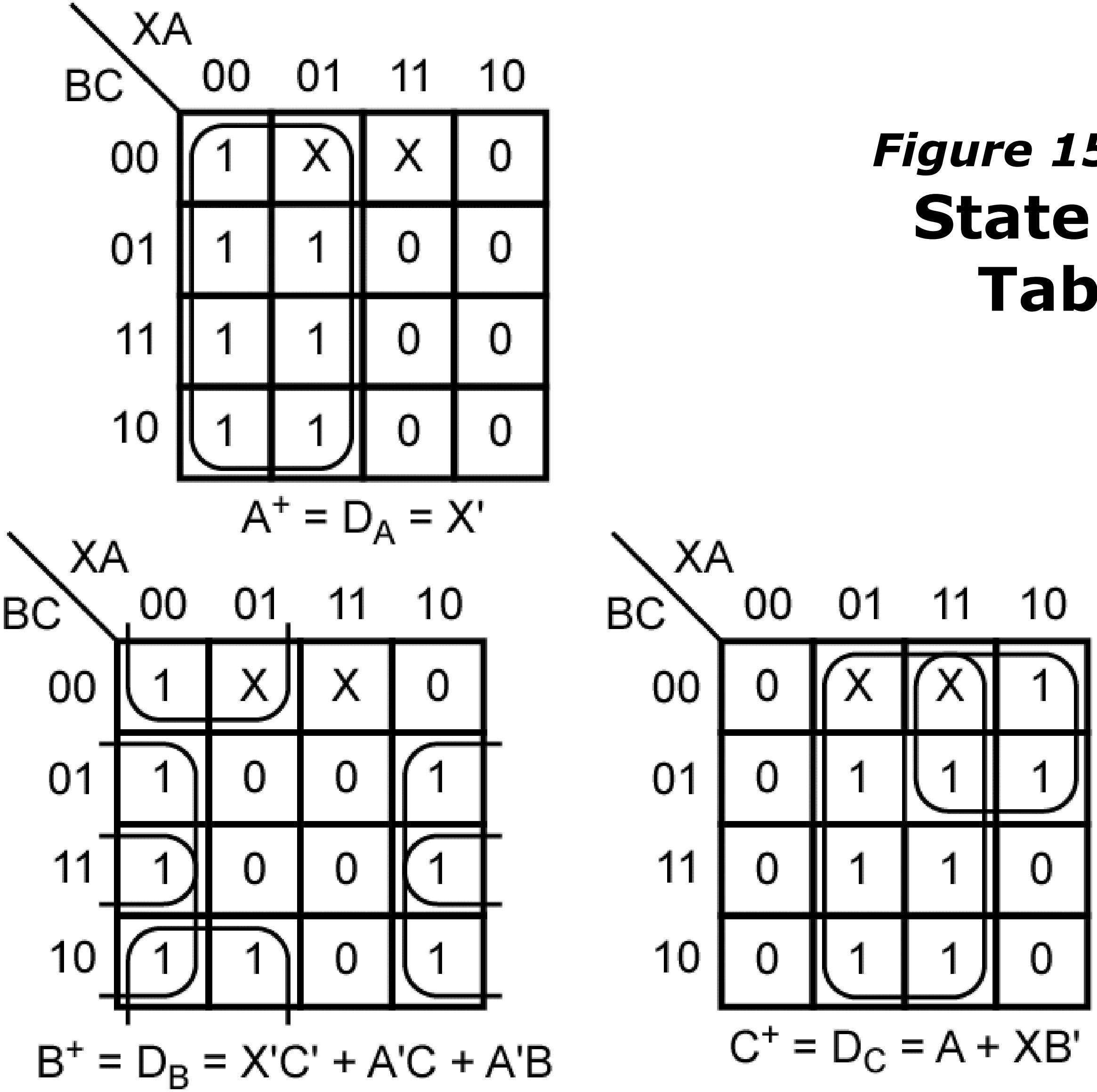
	$X = 0$	1	0	1
S_0	S_1	S_2	0	0
S_1	S_3	S_2	0	0
S_2	S_1	S_4	0	0
S_3	S_5	S_2	0	0
S_4	S_1	S_6	0	0
S_5	S_5	S_2	1	0
S_6	S_1	S_6	0	1

(b) Transition table

	$A^+B^+C^+$		Z	
ABC	$X = 0$	1	0	1
000	110	001	0	0
110	111	001	0	0
001	110	011	0	0
111	101	001	0	0
011	110	010	0	0
101	101	001	1	0
010	110	010	0	1

3. Derive the next-state maps from the transition table.

Figure 15-9a: **Next-State Maps for Table 15-6**



BC \ XA	00	01	11	10
00	1	X	X	0
01	1	1	0	0
11	1	1	0	0
10	1	1	0	0

$A^+ = D_A = X'$

BC \ XA	00	01	11	10
00	1	X	X	0
01	1	0	0	1
11	1	0	0	1
10	1	1	0	1

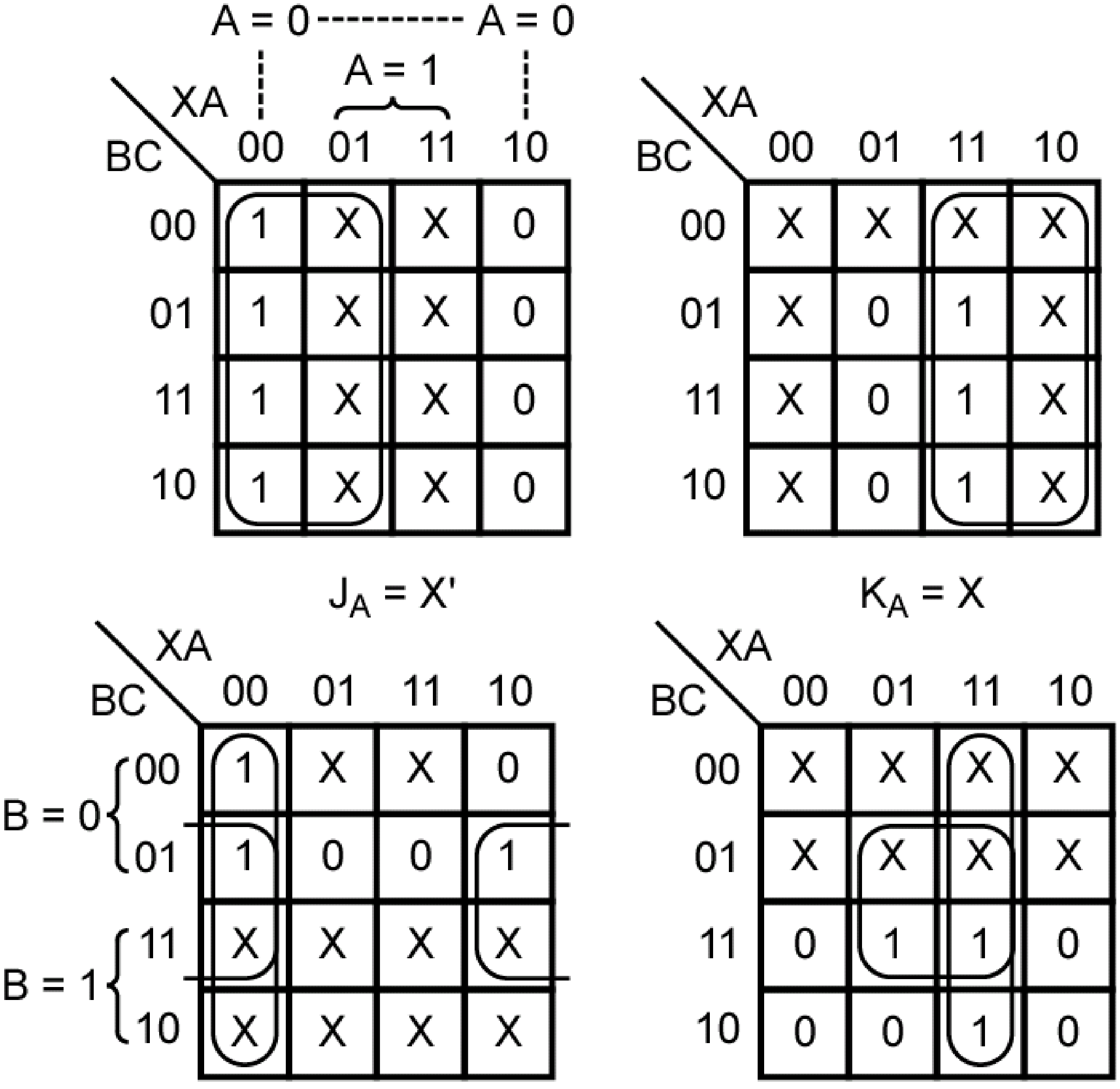
$B^+ = D_B = X'C' + A'C + A'B$

BC \ XA	00	01	11	10
00	0	X	X	1
01	0	1	1	1
11	0	1	1	0
10	0	1	1	0

$C^+ = D_C = A + XB'$

(a) Derivation of D flip-flop input equations

4. Find flip-flop input maps from the next-state maps using the techniques developed in Unit 12 and find the flip-flop input equations from the maps.



BC \ XA	00	01	11	10
00	1	X	X	0
01	1	X	X	0
11	1	X	X	0
10	1	X	X	0

$J_A = X'$

BC \ XA	00	01	11	10
00	X	X	X	X
01	X	0	1	X
11	X	0	1	X
10	X	0	1	X

$K_A = X$

BC \ XA	00	01	11	10
00	1	X	X	0
01	1	0	0	1
11	X	X	X	X
10	X	X	X	X

$J_B = X'A' + A'C$

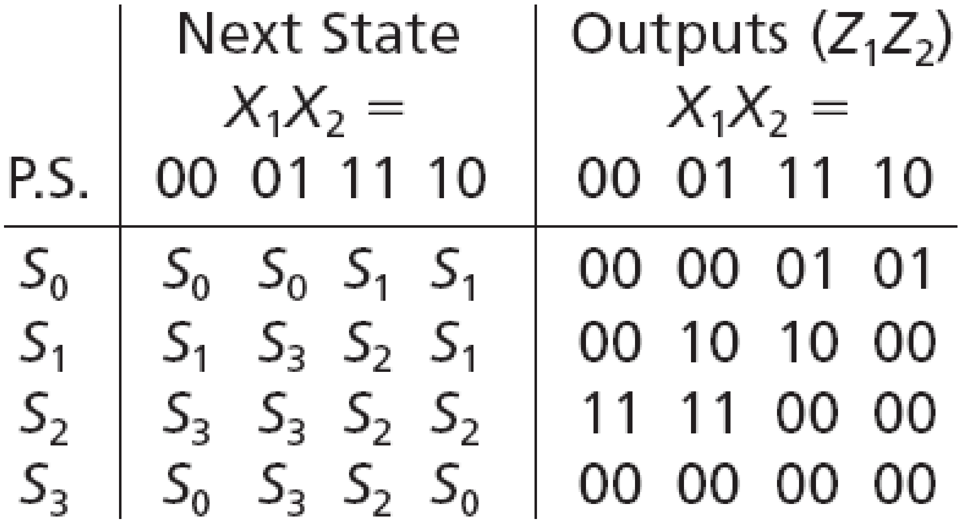
BC \ XA	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	0	1	1	0
10	0	0	1	0

$K_B = AC + XA$

(b) Derivation of J-K flip-flop input equations

Figure 15-9b: **Next-State Maps for Table 15-6**

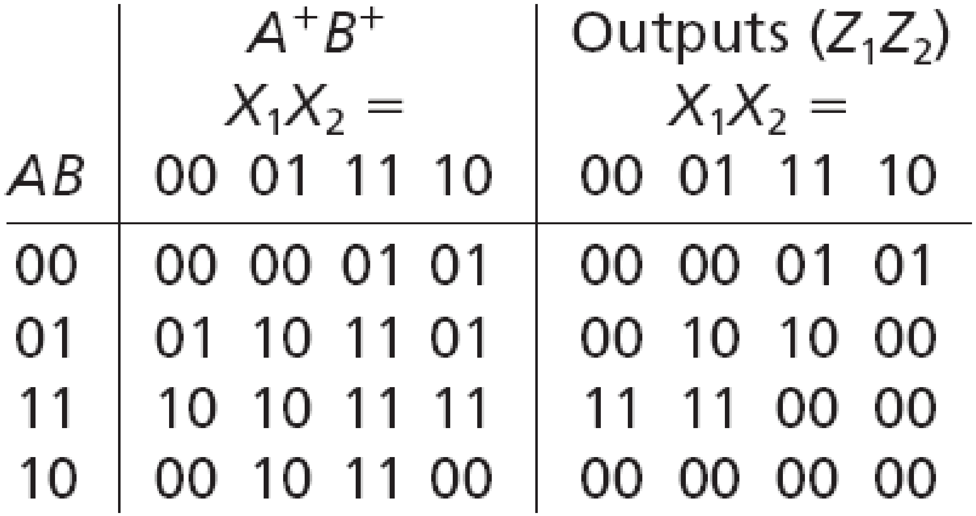
(a) State table



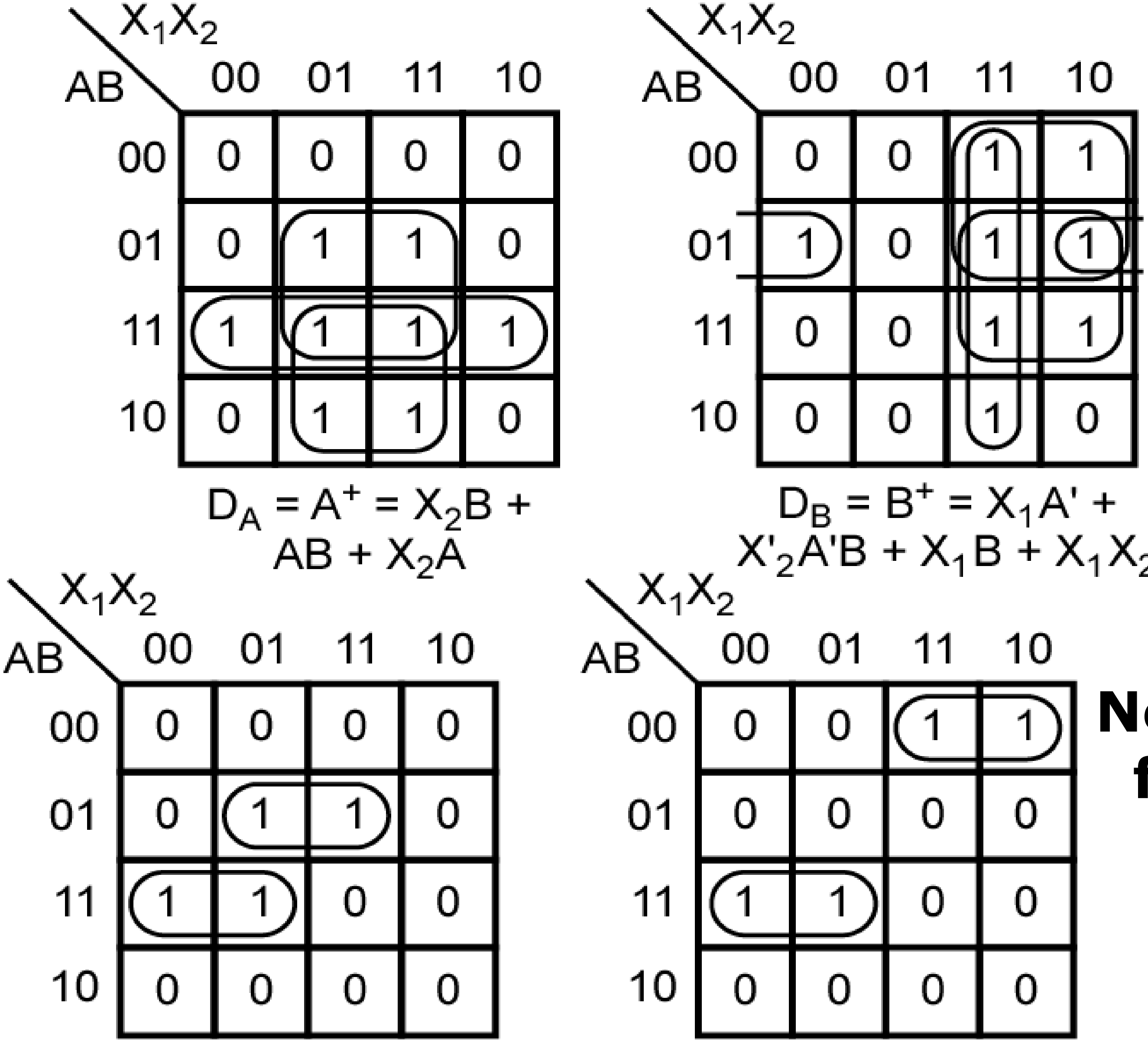
	Next State $X_1X_2 =$				Outputs (Z_1Z_2) $X_1X_2 =$			
P.S.	00	01	11	10	00	01	11	10
S_0	S_0	S_0	S_1	S_1	00	00	01	01
S_1	S_1	S_3	S_2	S_1	00	10	10	00
S_2	S_3	S_3	S_2	S_2	11	11	00	00
S_3	S_0	S_3	S_2	S_0	00	00	00	00

Table 15-7: Represents a sequential circuit with two inputs (X_1 and X_2) and two outputs (Z_1 and Z_2).

(b) Transition table



	A^+B^+ $X_1X_2 =$				Outputs (Z_1Z_2) $X_1X_2 =$			
AB	00	01	11	10	00	01	11	10
00	00	00	01	01	00	00	01	01
01	01	10	11	01	00	10	10	00
11	10	10	11	11	11	11	00	00
10	00	10	11	00	00	00	00	00



$AB \backslash X_1X_2$	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	1	1	1	1
10	0	1	1	0

$D_A = A^+ = X_2B + AB + X_2A$

$AB \backslash X_1X_2$	00	01	11	10
00	0	0	1	1
01	1	0	1	1
11	0	0	1	1
10	0	0	1	0

$D_B = B^+ = X_1A' + X'_2A'B + X_1B + X_1X_2$

$AB \backslash X_1X_2$	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	1	1	0	0
10	0	0	0	0

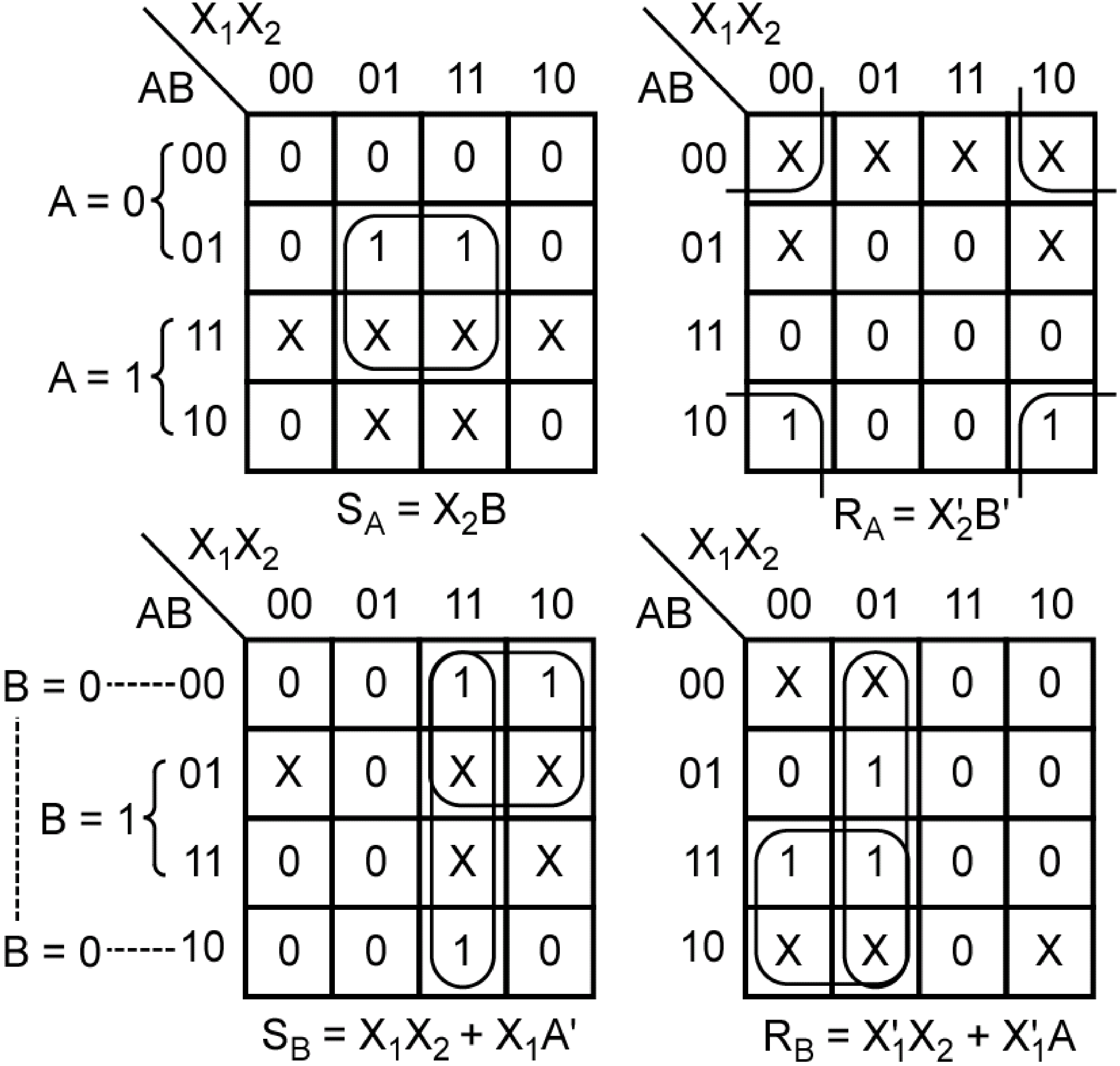
$Z_1 = X_2A'B + X'_1AB$

$AB \backslash X_1X_2$	00	01	11	10
00	0	0	1	1
01	0	0	0	0
11	1	1	0	0
10	0	0	0	0

$Z_2 = X_1A'B' + X'_1AB$

Figure 15-10:

Next-State Maps for Table 15-7



AB \ X_1X_2	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	X	X	X	X
10	0	X	X	0

$S_A = X_2B$

AB \ X_1X_2	00	01	11	10
00	X	X	X	X
01	X	0	0	X
11	0	0	0	0
10	1	0	0	1

$R_A = X_2'B'$

AB \ X_1X_2	00	01	11	10
00	0	0	1	1
01	X	0	X	X
11	0	0	X	X
10	0	0	1	0

$S_B = X_1X_2 + X_1A'$

AB \ X_1X_2	00	01	11	10
00	X	X	0	0
01	0	1	0	0
11	1	1	0	0
10	X	X	0	X

$R_B = X_1'X_2 + X_1'A$

Figure 15-11: **Derivation of S-R Equations for Table 15-7**

Equivalent State Assignments

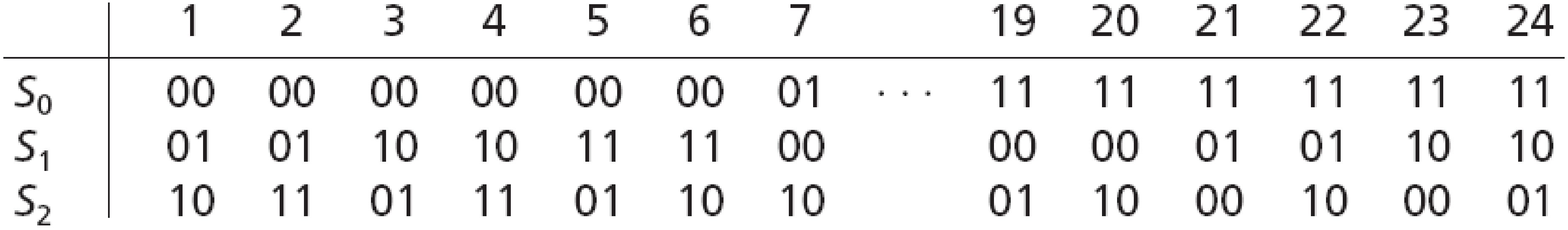
After the number of states in a state table has been reduced, the next step in realizing the table is to assign flip-flop states to correspond to the states in the table.

The cost of the logic required to realize a sequential circuit is strongly dependent on the way this state assignment is made.

Section 15.7 (p. 487)

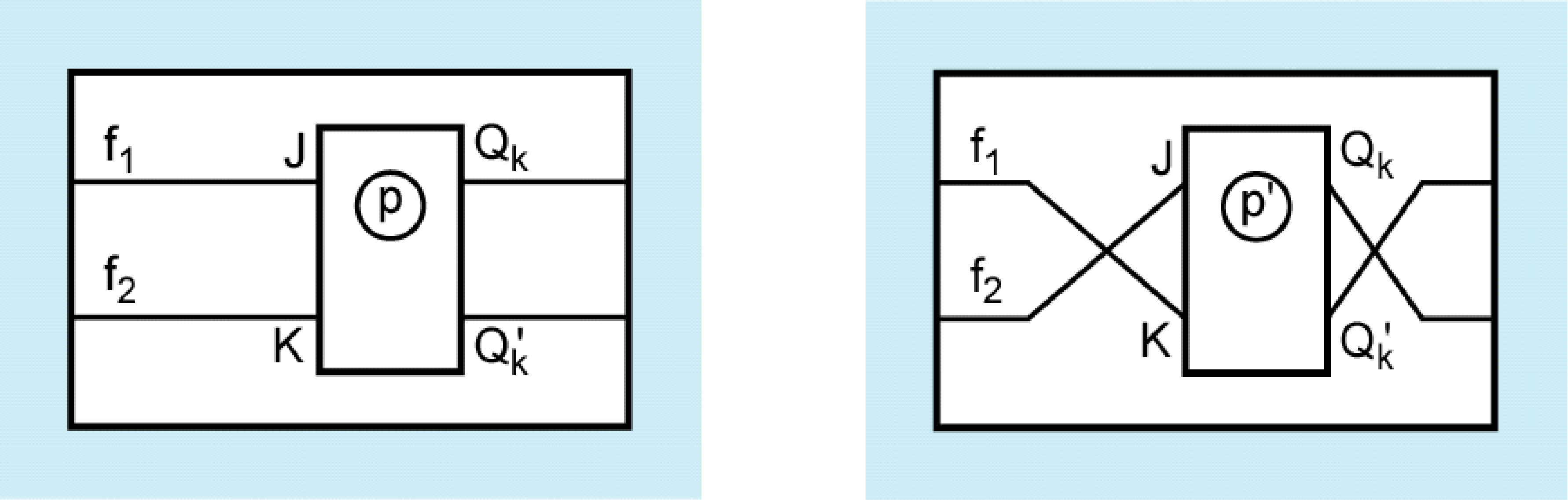
Given a sequential circuit with three states and two flip-flops (*A* and *B*), there are 4 × 3 × 2 = 24 possible state assignments for the three states.

Table 15-8. State Assignments for 3-Row Tables



	1	2	3	4	5	6	7		19	20	21	22	23	24
S_0	00	00	00	00	00	00	01	· · ·	11	11	11	11	11	11
S_1	01	01	10	10	11	11	00		00	00	01	01	10	10
S_2	10	11	01	11	01	10	10		01	10	00	10	00	01

Complementing one or more columns of a state assignment will have no effect on the cost of realization.

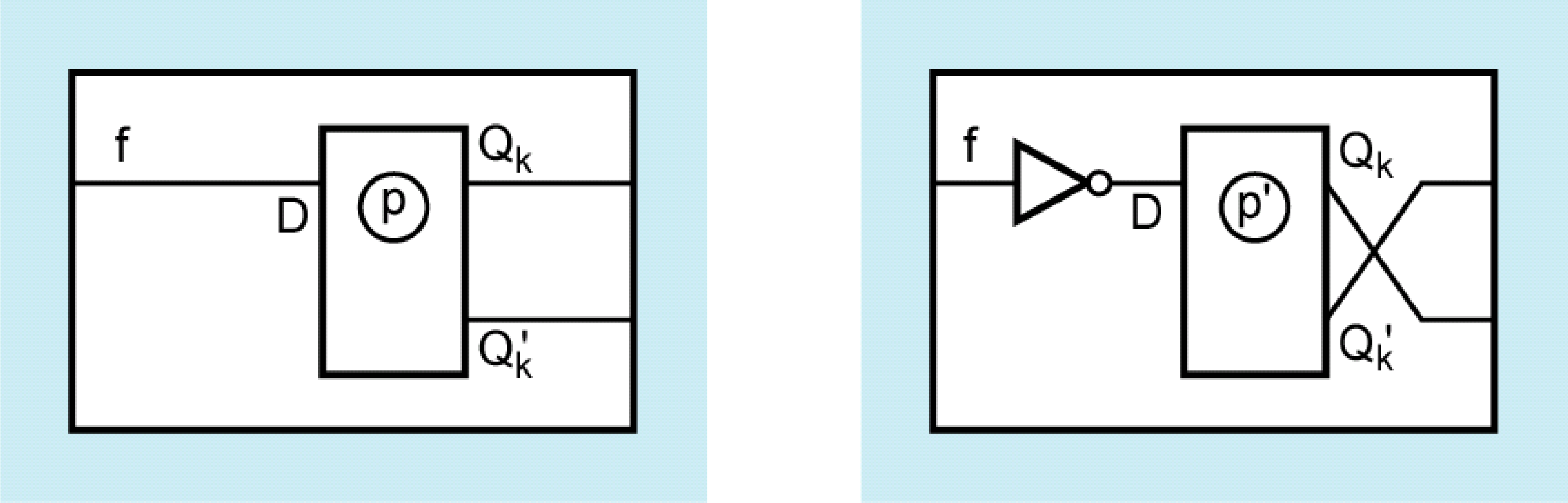


(a) Circuit A

(b) Circuit B (identical to A except leads to flip-flop Q_k are crossed)

Figure 15-12: **Equivalent Circuits Obtained by Complementing *Q_k***

If unsymmetrical flip-flops are used such as a D flip-flop, complementing a column may require adding an inverter to the circuit.



(a) Circuit A

(b) Circuit B
(identical to A except for connections to flip-flop Q_k)

***Figure 15-13:* Equivalent Circuits Obtained by Complementing Q_k**

Table 15-9.

Assignments			Present	Next State		Output	
A_3	B_3	C_3	State	$X = 0$	1	0	1
00	00	11	S_1	S_1	S_3	0	0
01	10	10	S_2	S_2	S_1	0	1
10	01	01	S_3	S_2	S_3	1	0

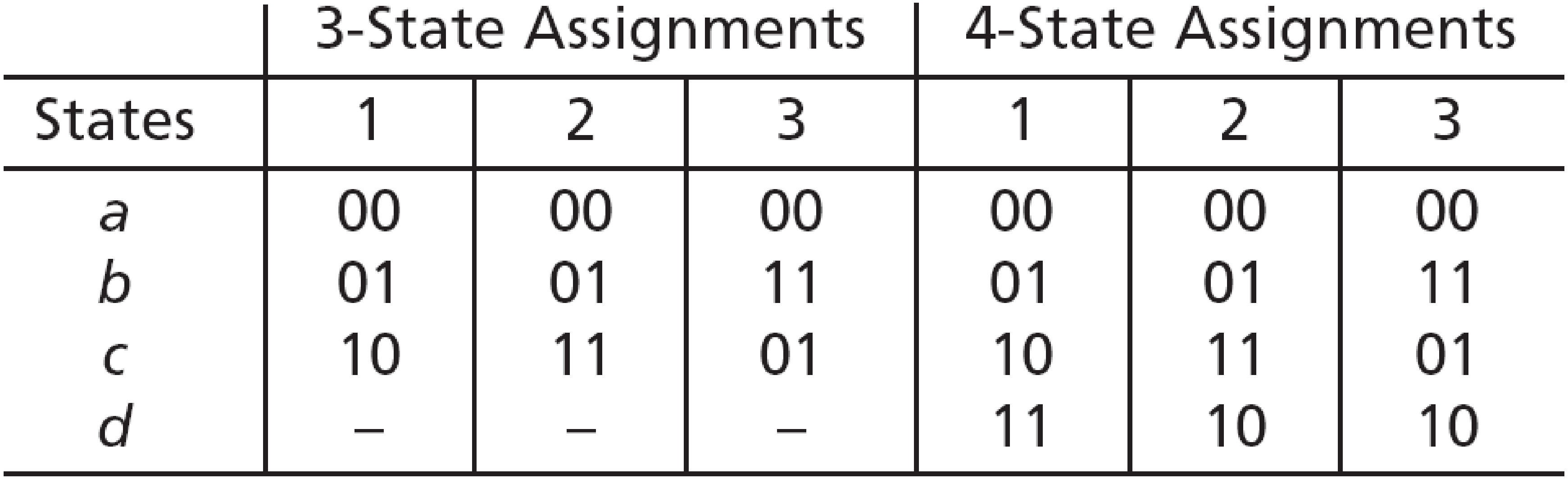
Assignment "A"	Assignment "B"	Assignment "C"
$J_1 = XQ'_2$	$J_2 = XQ_1'$	$K_1 = XQ_2$
$K_1 = X'$	$K_2 = X'$	$J_1 = X'$
$J_2 = X'Q_1$	$J_1 = X'Q_2$	$K_2 = X'Q_1'$
$K_2 = X$	$K_1 = X$	$J_2 = X$
$Z = X'Q_1 + XQ_2$	$Z = X'Q_2 + XQ_1$	$Z = X'Q_1' + XQ_2'$
$D_1 = XQ_2'$	$D_2 = XQ_1'$	$D_1 = X' + Q_2'$
$D_2 = X'(Q_1+Q_2)$	$D_1 = X'(Q_2 + Q_1)$	$D_2 = X + Q_1Q_2$

Section 15.7 (p. 489)

When realizing a three-state sequential circuit with symmetrical flip-flops, it is only necessary to try three different states to be assured of a minimum cost realization.

Similarly, only three different assignments must be tried for four states.

Table 15-10. Nonequivalent Assignments for 3 and 4 States



	3-State Assignments			4-State Assignments		
States	1	2	3	1	2	3
a	00	00	00	00	00	00
b	01	01	11	01	01	11
c	10	11	01	10	11	01
d	–	–	–	11	10	10

Table 15-11. Number of Distinct (Nonequivalent) State Assignments

Number of States	Minimum Number of State Variables	Number of Distinct Assignments
2	1	1
3	2	3
4	2	3
5	3	140
6	3	420
7	3	840
8	3	840
9	4	10,810,800
⋮	⋮	⋮
16	4	$\approx 5.5 \times 10^{10}$

Guidelines for State Assignment

Trying all nonequivalent state assignments is not practical in most cases. The following *guidelines* are useful in making assignments which will place 1’s together (or 0’s together) on the next-state maps:

1. States which have the same next state for a given input should be given adjacent assignments.
2. States which are the next states of the same state should be given adjacent assignments.
3. States which have the same output for a given input should be given adjacent assignments.

Section 15.8 (p. 490)

According to Guideline 1, S_0, S_2, S_4, and S_6 should be given adjacent assignments because they all have S_1 as a next state (with input 0).

ABC		X = 0	1	0	1
000	S_0	S_1	S_2	0	0
110	S_1	S_3	S_2	0	0
001	S_2	S_1	S_4	0	0
111	S_3	S_5	S_2	0	0
011	S_4	S_1	S_6	0	0
101	S_5	S_5	S_2	1	0
010	S_6	S_1	S_6	0	1

(a) State table

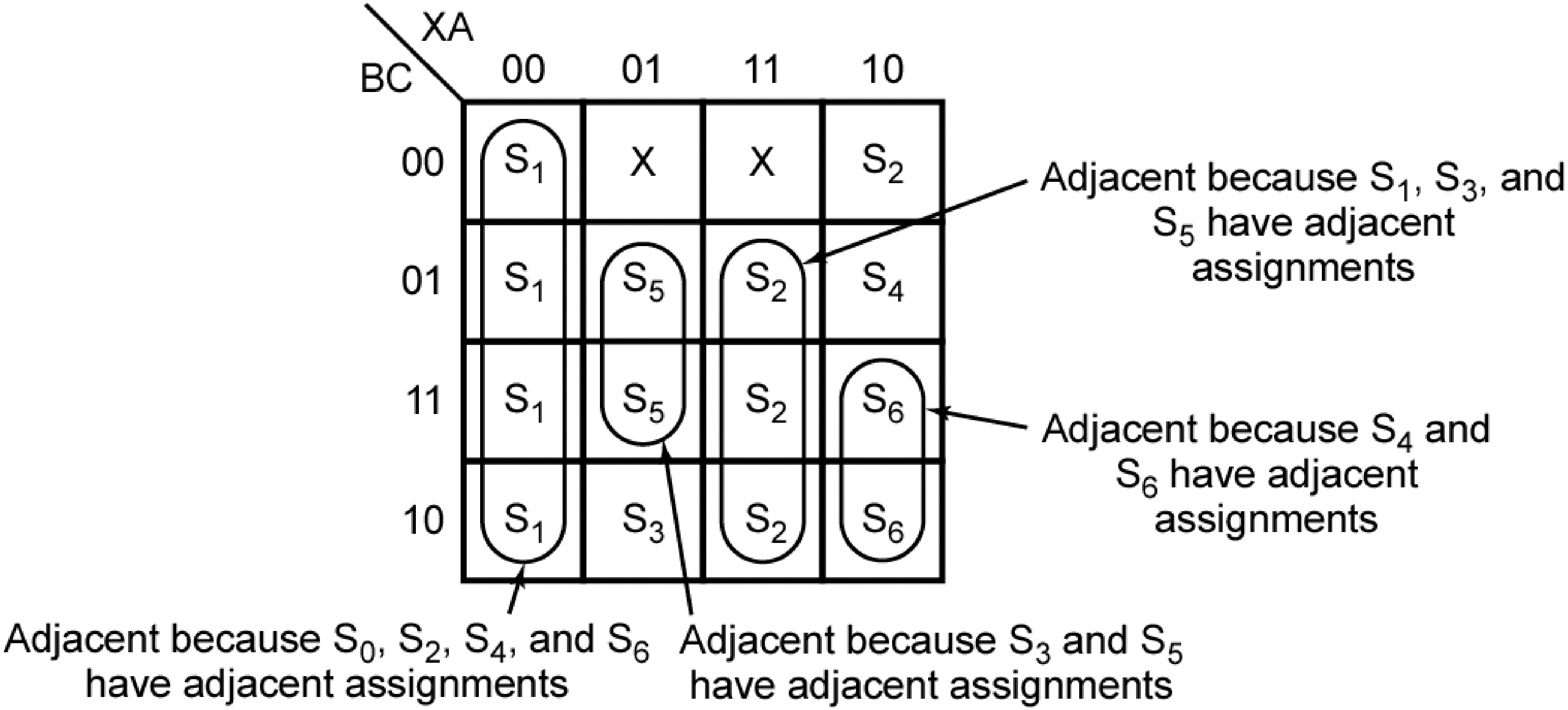
BC \ A	0	1
00	S_0	
01	S_2	S_5
11	S_4	S_3
10	S_6	S_1

BC \ A	0	1
00	S_0	
01	S_1	S_6
11	S_3	S_4
10	S_5	S_2

(b) Assignment maps

From Guideline 2: S_1 and S_2 should be given adjacent assignments (both are next states of S_0).

Figure 15-14b



BC \ XA	00	01	11	10
00	S_1	X	X	S_2
01	S_1	S_5	S_2	S_4
11	S_1	S_5	S_2	S_6
10	S_1	S_3	S_2	S_6

Adjacent because S_1, S_3, and S_5 have adjacent assignments

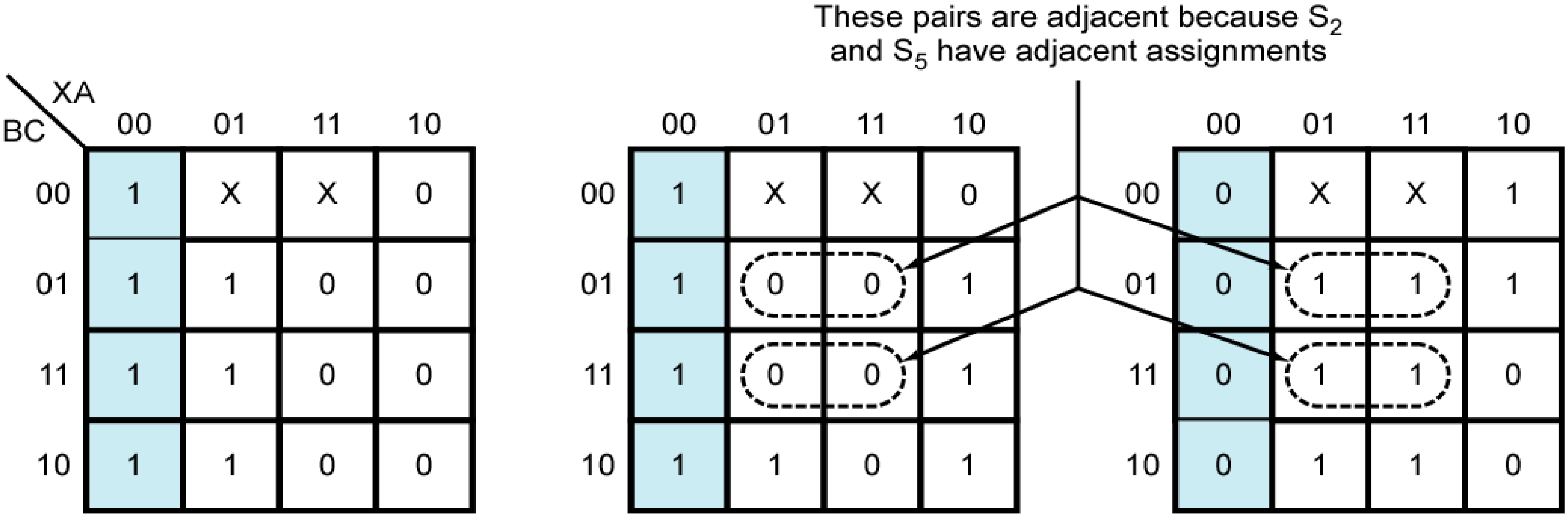
Adjacent because S_4 and S_6 have adjacent assignments

Adjacent because S_0, S_2, S_4, and S_6 have adjacent assignments

Adjacent because S_3 and S_5 have adjacent assignments

(a) Next-state maps for Figure 15-14

Figure 15-15a: **Next-State Maps for Figure 15-14**



These pairs are adjacent because S_2 and S_5 have adjacent assignments

BC \ XA	00	01	11	10
00	1	X	X	0
01	1	1	0	0
11	1	1	0	0
10	1	1	0	0

$A^+ = D_A = X'$

BC \ XA	00	01	11	10
00	1	X	X	0
01	1	0	0	1
11	1	0	0	1
10	1	1	0	1

$B^+ = D_B = X'C' + A'C + A'B$

BC \ XA	00	01	11	10
00	0	X	X	1
01	0	1	1	1
11	0	1	1	0
10	0	1	1	0

$C^+ = D_C = A + XB'$

(b) Next-state maps for Figure 15-14 (cont.)

Figure 15-15b: Next-State Maps for Figure 15-14

Q_2Q_3 \ Q_1	0	1
00	a	c
01		e
11	d	b
10		f

a = 000
b = 111
c = 100
d = 011
e = 101
f = 110

(b)

Q_2Q_3 \ Q_1	0	1
00	c	a
01		e
11	d	b
10	f	

a = 100
b = 111
c = 000
d = 011
e = 101
f = 010

(c)

	$X = 0$	1	$X = 0$	1
a	*a*	*c*	0	0
b	*d*	*f*	0	1
c	*c*	*a*	0	0
d	*d*	*b*	0	1
e	*b*	*f*	1	0
f	*c*	*e*	1	0

(a)

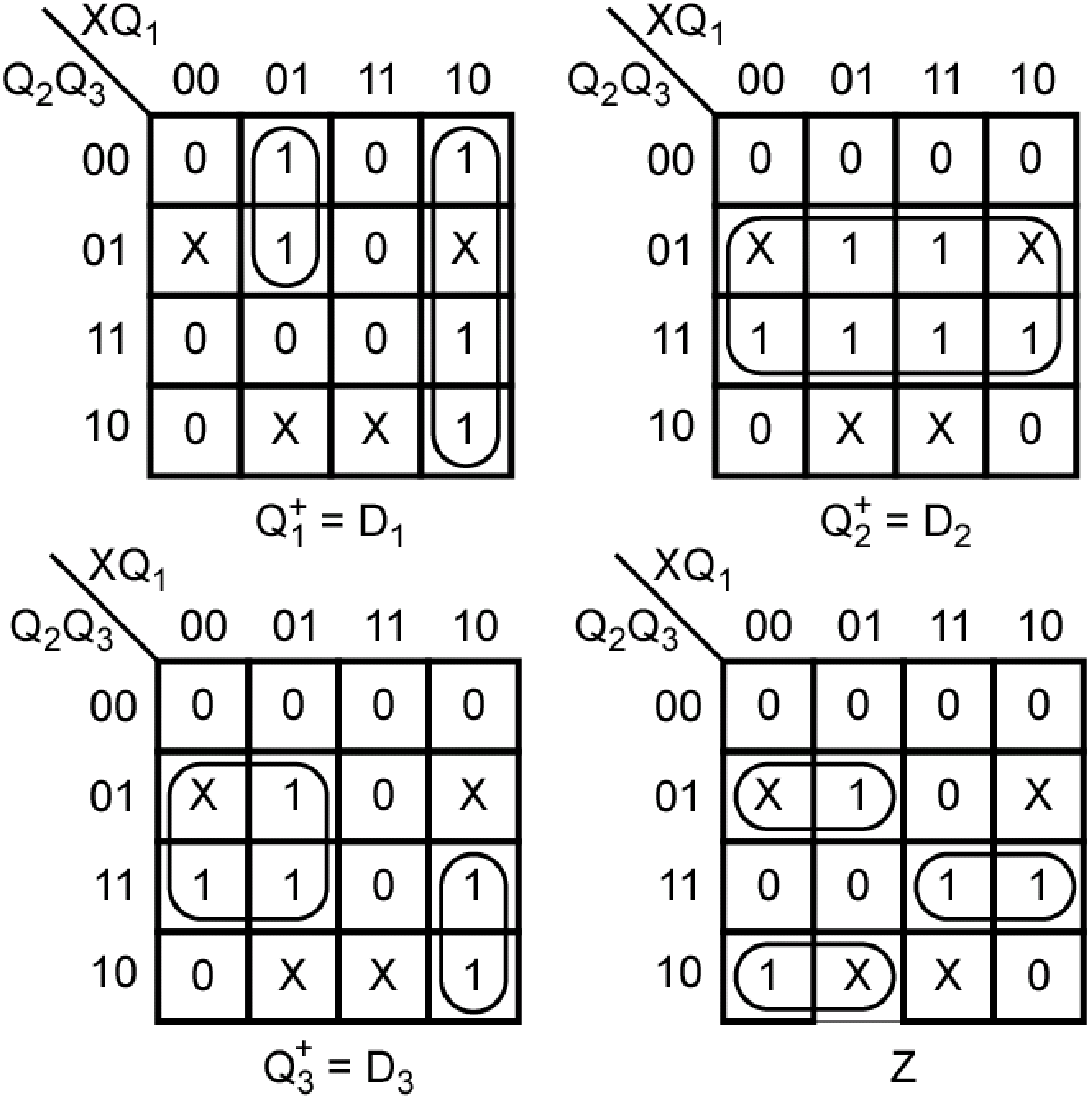
Adjacencies by Guidelines:

1. (*b, d*) (*c, f*) (*b, e*)
2. (*a, c*)2x (*d, f*) (*b, d*) (*b, f*) (*c, e*)
3. (*a, c*) (*b, d*) (*e, f*)

Figure 15-16: **State Table and Assignments**

Table 15-12. Transition table for Figure 15-16(a)

	$Q_1{}^+Q_2{}^+Q_3{}^+$			
$Q_1Q_2Q_3$	$X = 0$	1	$X = 0$	1
1 0 0	100	000	0	0
1 1 1	011	010	0	1
0 0 0	000	100	0	0
0 1 1	011	111	0	1
1 0 1	111	010	1	0
0 1 0	000	101	1	0



Q_2Q_3 \ XQ_1	00	01	11	10
00	0	1	0	1
01	X	1	0	X
11	0	0	0	1
10	0	X	X	1

$Q_1^+ = D_1$

Q_2Q_3 \ XQ_1	00	01	11	10
00	0	0	0	0
01	X	1	1	X
11	1	1	1	1
10	0	X	X	0

$Q_2^+ = D_2$

Q_2Q_3 \ XQ_1	00	01	11	10
00	0	0	0	0
01	X	1	0	X
11	1	1	0	1
10	0	X	X	1

$Q_3^+ = D_3$

Q_2Q_3 \ XQ_1	00	01	11	10
00	0	0	0	0
01	X	1	0	X
11	0	0	1	1
10	1	X	X	0

Z

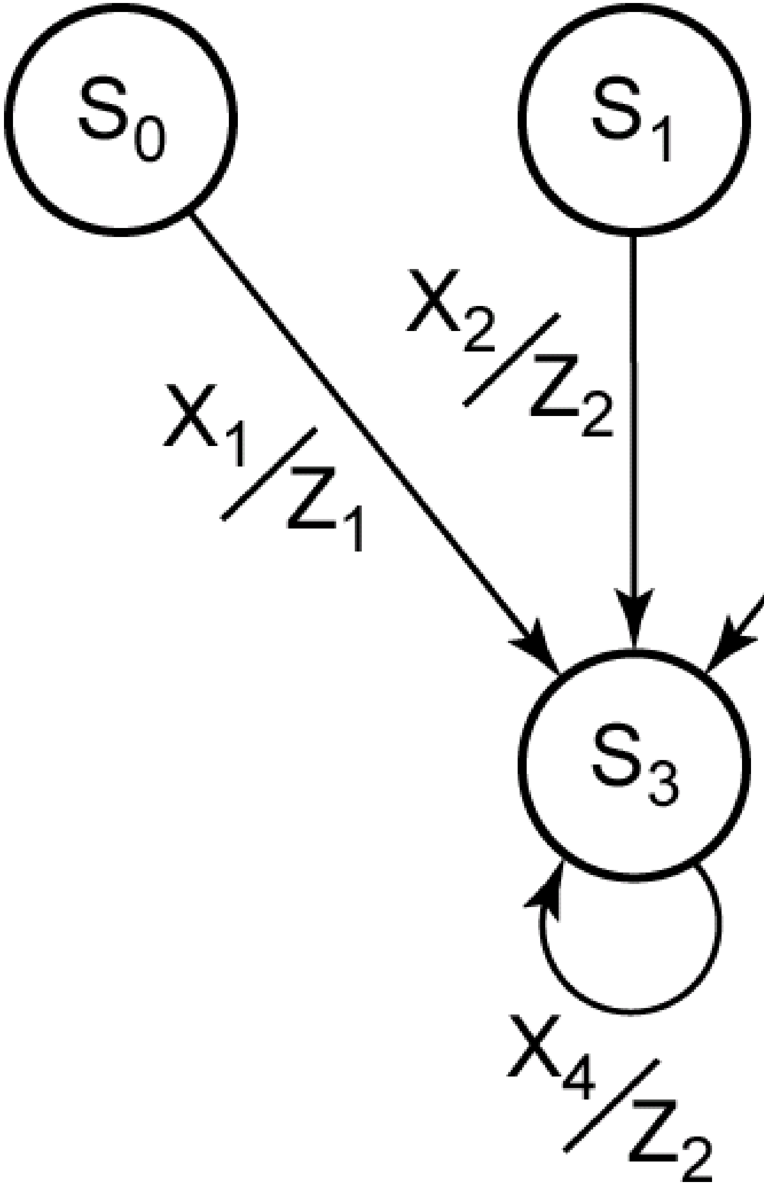
Figure 15-17: **Next-State and Output Maps for Table 15-12**

Using a One-Hot State Assignment

Sometimes reducing the number of flip-flops used is not as important as reducing the logic feeding into the flip-flops. Using a *one-hot* state assignment may help accomplish this.

The one-hot assignment uses one flip-flop for each state, so a state machine with N states requires N flip-flops. Exactly one of the flip-flops is set to one in each state.

Section 15.9 (p. 495)



By simple inspection of the graph:

$$Q_3^+ = X_1Q_0 + X_2Q_1 + X_3Q_2 + X_4Q_3$$

$$Z_2 = X_2Q_1 + X_4Q_3$$

Figure 15-18: **Partial State Graph**

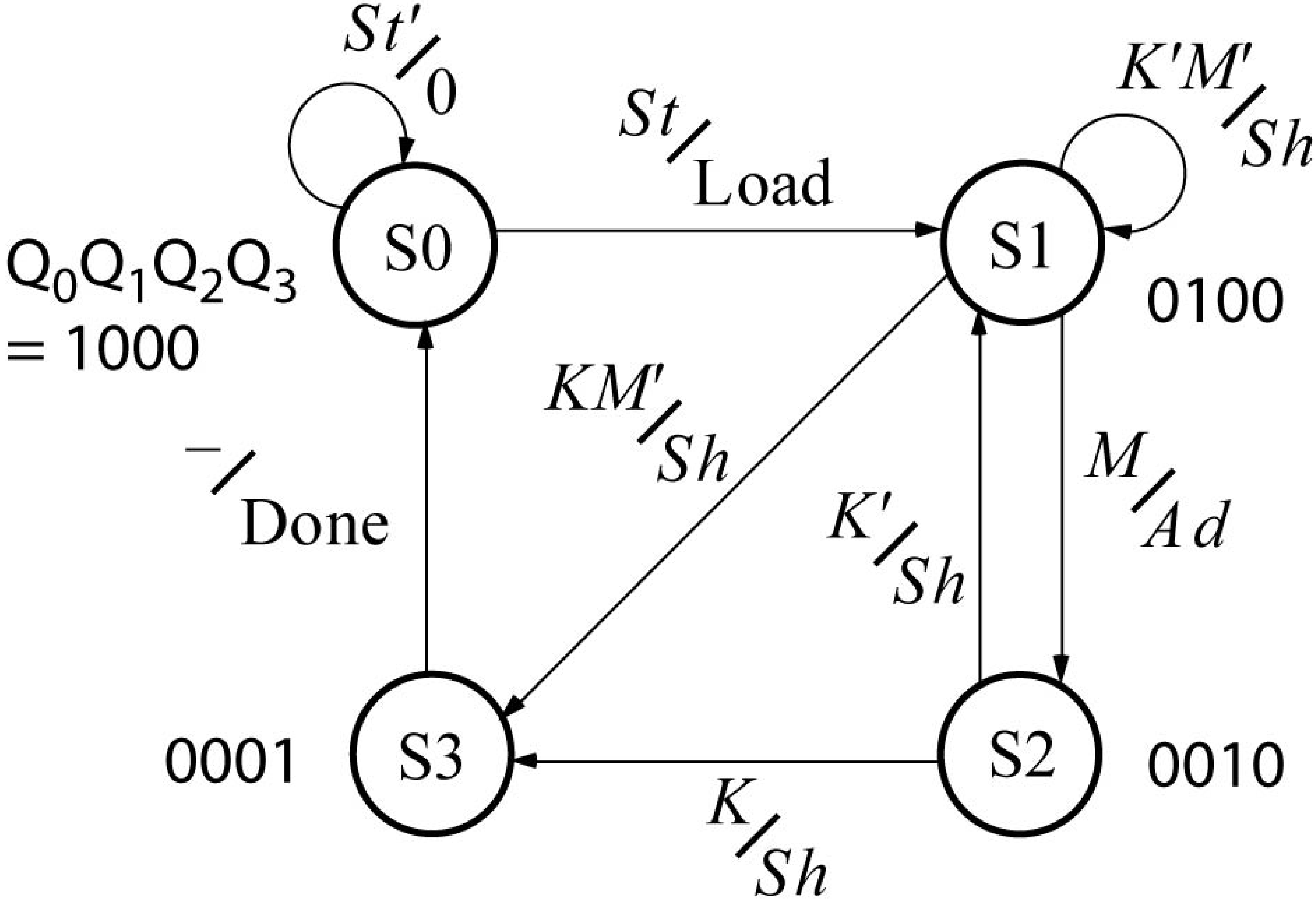


Figure 15-19: **Multiplier Control State Graph**

Equations derived by inspection of Figure 15-19:

$$Q_0^+ = Q_0 St' + Q_3$$

$$Q_1^+ = Q_0 St + Q_1 K'M' + Q_2 K'$$

$$Sh = Q_1(K'M' + KM') + Q_2(K' + K) = Q_1 M' + Q_2$$