

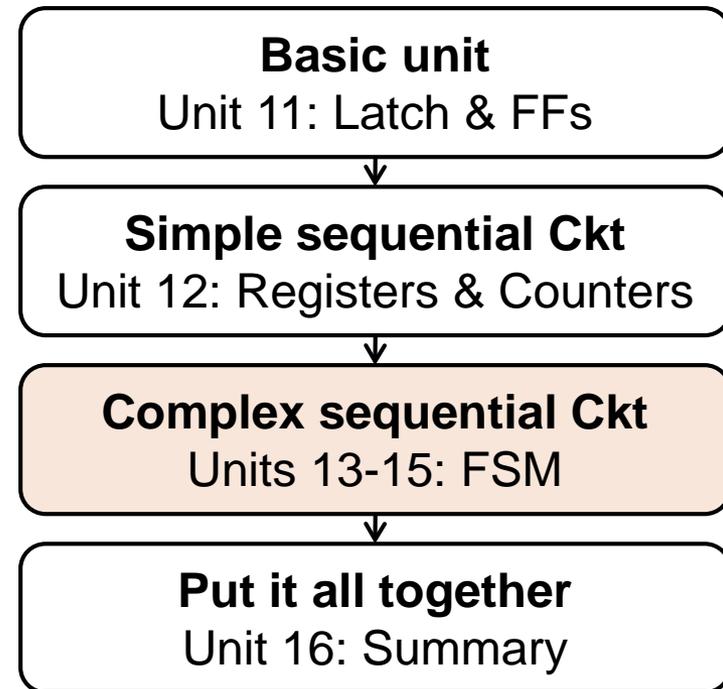
UNIT 15
REDUCTION OF STATE TABLES
STATE ASSIGNMENT



Spring 2011

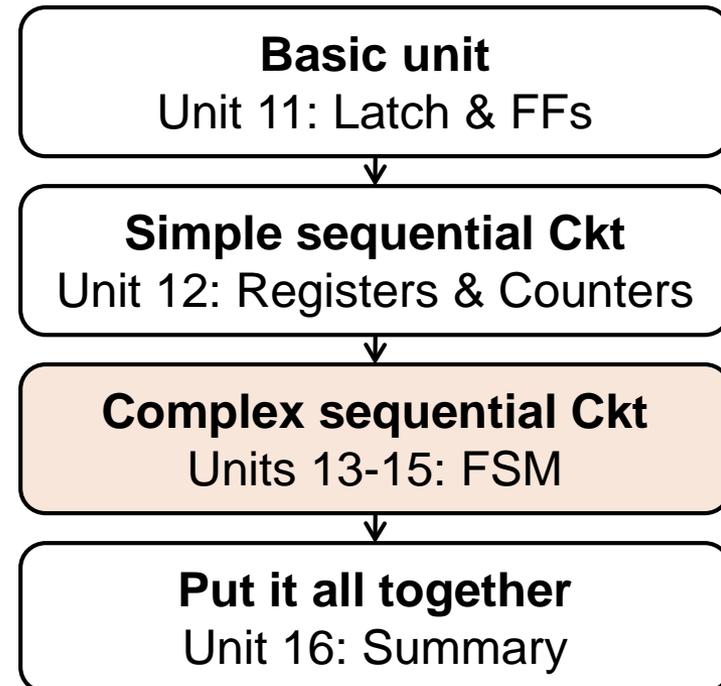
Reduction of State Tables

- **Contents**
 - ▣ Elimination of redundant states
 - ▣ Equivalent states
 - ▣ Implication table
 - ▣ Equivalent sequential circuits
 - ▣ Incompletely specified state tables
 - ▣ Derivation of FF input equations
 - ▣ Equivalent state assignment
 - ▣ Guidelines for state assignment
 - ▣ One-hot state assignment
- **Reading**
 - ▣ Unit 15



Designing a Sequential Circuit (1/2)

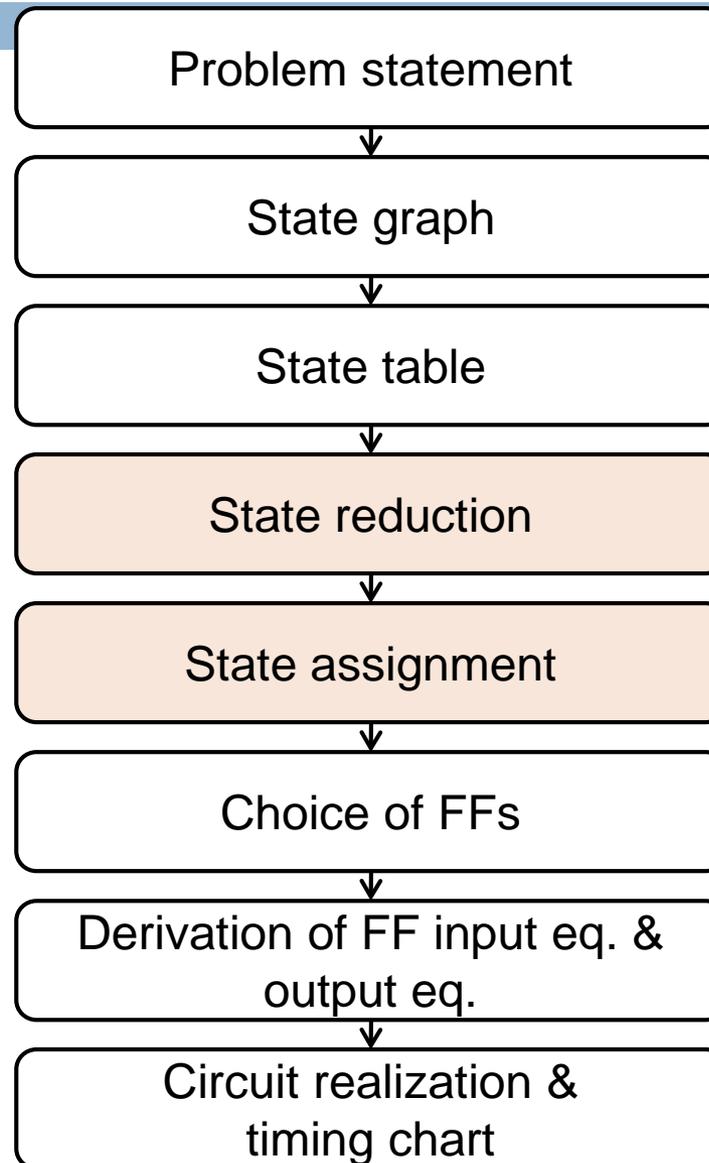
- **Given the specification of a sequential circuit**
- **Design procedure:**
 1. Construct a state table or state graph (Unit 14)
 2. Simplify (Unit 15)
 - Redundancy (equivalence)
 3. Derive FF input equations & output equations (Unit 12)



Designing a Sequential Circuit (2/2)

4

© Iris H.-R. Jiang



Reduction of state tables

5

Elimination of Redundant States

Row matching

Recap: 14.3 Case Study I (1/2)

□ Examine groups of **4 consecutive inputs** & produce an output

□ **Reset after every 4 inputs**

□ e.g., $X = \underline{0101} \underline{0010} \underline{1001} \underline{0100}$
 $Z = 000\underline{1} \ 0000 \ 000\underline{1} \ 0000$

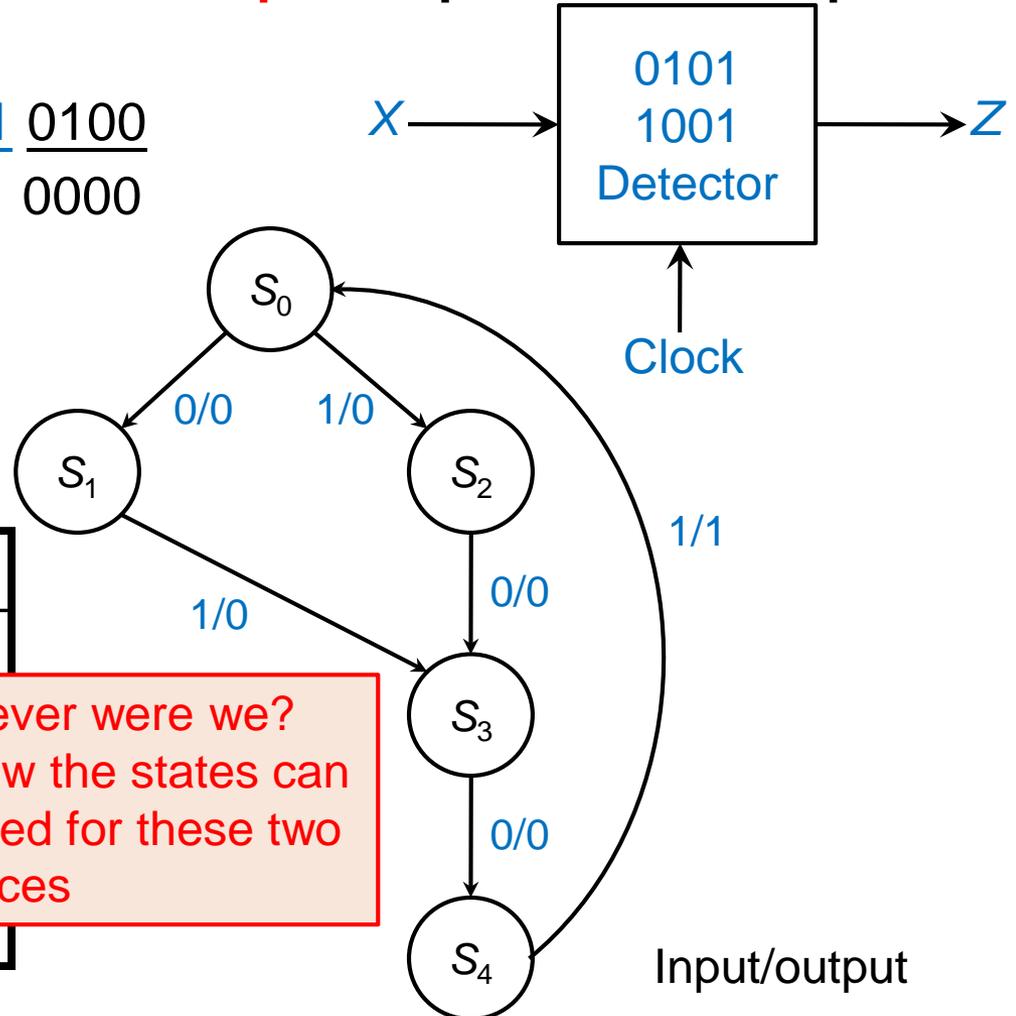
□ **Observation:** $X = \underline{0101}$
 $X = \underline{1001}$

□ **Typical sequence**

□ Partial state graph

State	Sequence received
S_0	Reset
S_1	0
S_2	1
S_3	01 or 10
S_4	<u>010</u> or <u>100</u>

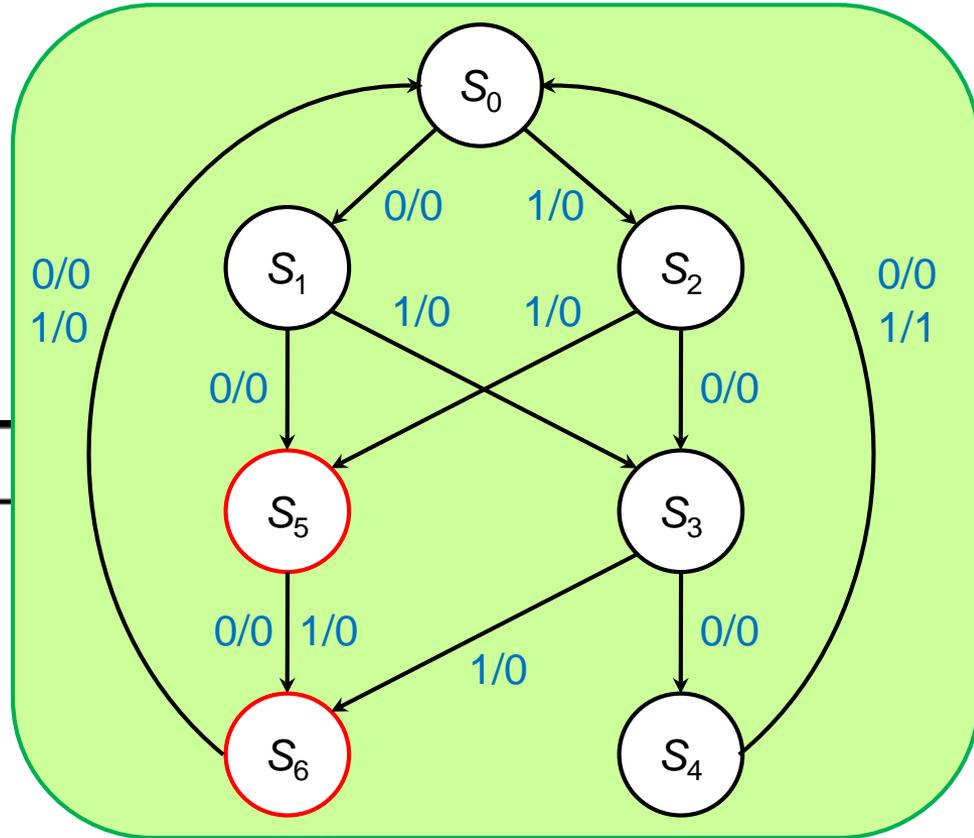
How clever were we?
 We knew the states can be shared for these two sequences



Input/output

Recap: 14.3 Case Study I (2/2)

- Complete state graph



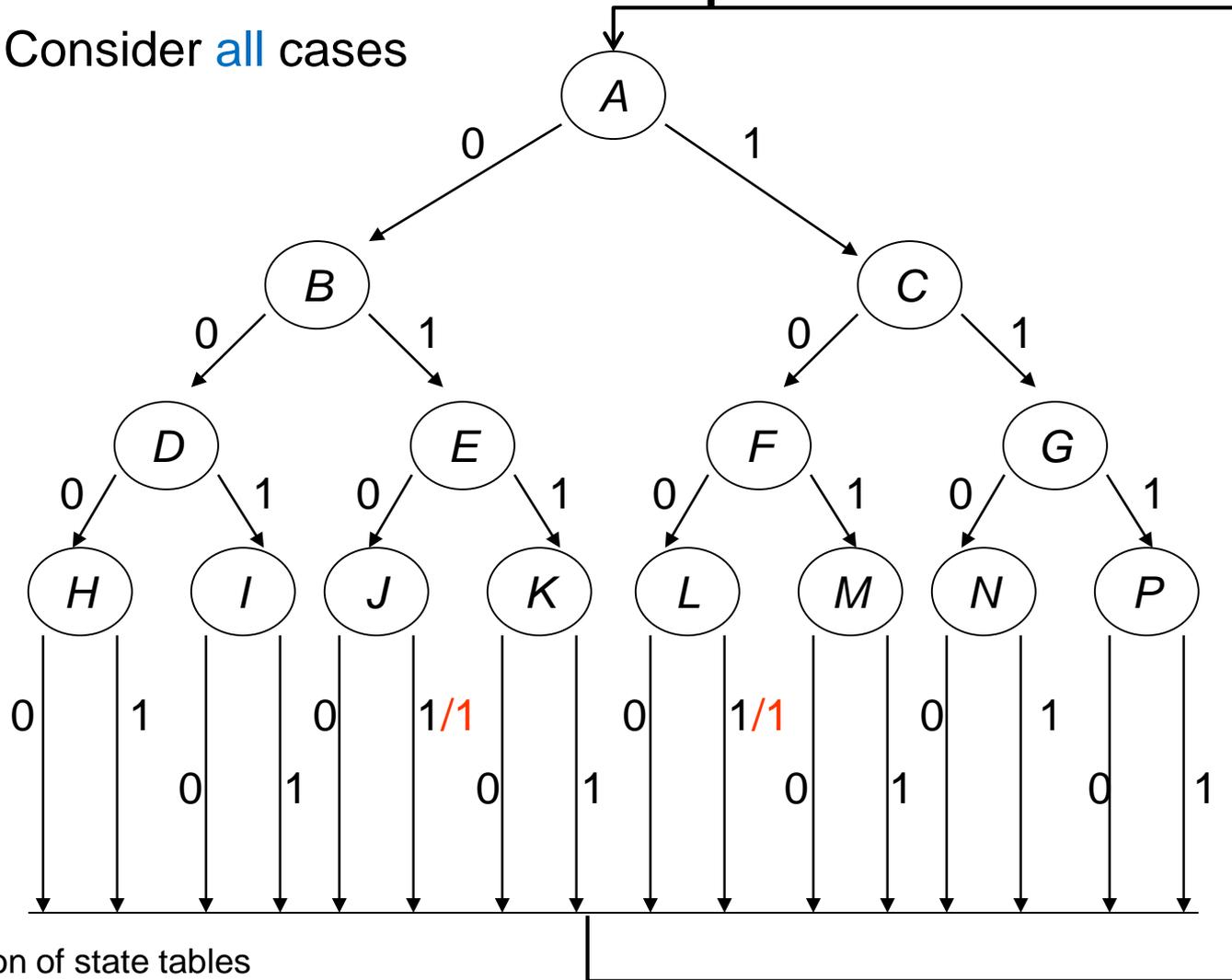
State	Sequence received
S_0	Reset
S_1	0
S_2	1
S_3	01 or 10
S_4	010 or 100
S_5	Two input received, no 1 output is possible
S_6	Three input received, no 1 output is possible

Let's see how to do it systematically
Following the method, everyone can do it

Elimination of Redundant States (1/4)

□ State table for **0101** and **1001** sequence detector

- Consider **all** cases



Reduction of state tables

Elimination of Redundant States (2/4)

□ Many similar cases \Rightarrow Reduce!

□ Equivalent state:

▣ Same NS

▣ Same Z

□ Check **H**

▣ $H \equiv I \equiv K \equiv M \equiv N \equiv P$

□ Check **J**

▣ $J \equiv L$

Input sequence	Present state	Next state		Present output	
		X = 0	X = 1	X = 0	X = 1
reset	A	B	C	0	0
0	B	D	E	0	0
1	C	F	G	0	0
00	D	H	H V	0	0
01	E	J	H K	0	0
10	F	J L	H M	0	0
11	G	H N	H 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

Elimination of Redundant States (3/4)

- Check **E**
 - $E \equiv F$
- Check **D**
 - $D \equiv G$
- **Row matching**

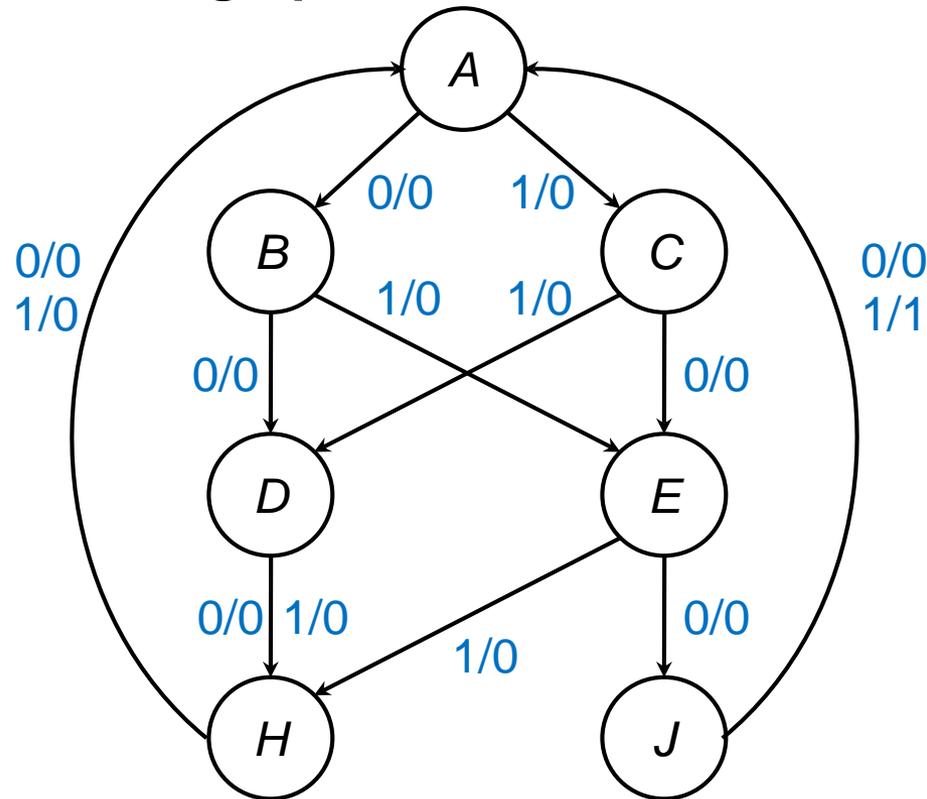
Input sequence	Present state	Next state		Present output	
		X = 0	X = 1	X = 0	X = 1
reset	A	B	C	0	0
0	B	D	E	0	0
1	C	E F	D G	0	0
00	D	H	H V	0	0
01	E	J	H K	0	0
10	F	J L	H M	0	0
11	G	H N	H P	0	0
000	H	A	A	0	0
001	I	A	A	0	0
010	J	A	A	0	1
011	K	A	A	0	0
100	L	A	A	0	1
101	M	A	A	0	0
110	N	A	A	0	0
111	P	A	A	0	0

Elimination of Redundant States (4/4)

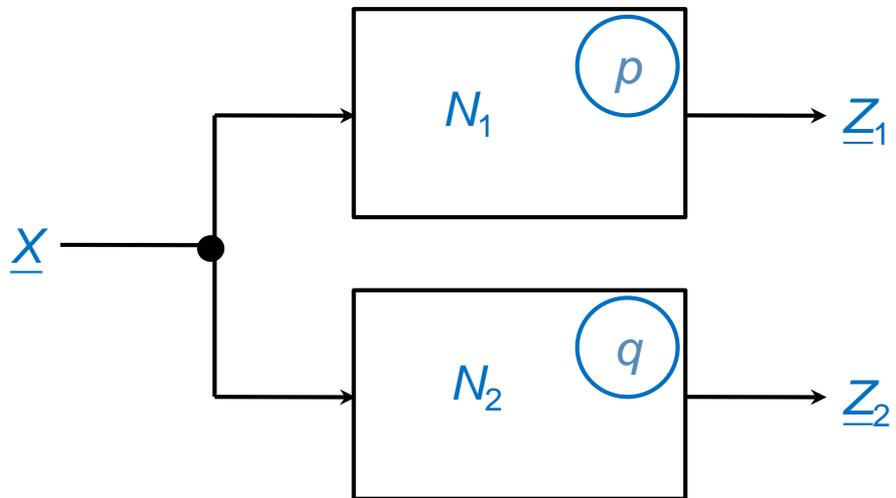
Reduced state table

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	E	D	0	0
D	H	H	0	0
E	J	H	0	0
H	A	A	0	0
J	A	A	0	1

State graph



Same as the previous one!



State Equivalence

□ Def: State equivalence

- N_1, N_2 : sequential circuits (**not necessarily different**)
- \underline{X} : a sequence of inputs of arbitrary length
- Then state p in $N_1 \equiv$ state q in N_2 iff $\lambda_1(p, \underline{X}) = \lambda_2(q, \underline{X})$ for every possible input sequence \underline{X}

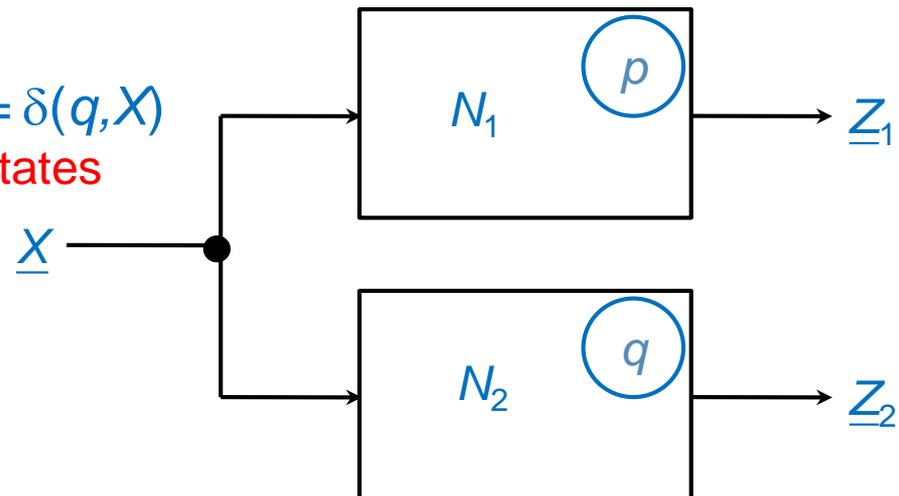
□ Difficult to check the equivalence using this definition!

□ Thm: State equivalence

- 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, i.e.,

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

outputs **next states**



Example: State Equivalence

- The following state table has no equivalent states
 - ▣ The only possible pair of equivalent states is S_0 and S_2
 - $S_0 \equiv S_2$ iff $S_3 \equiv S_3$, $S_2 \equiv S_0$, $S_1 \equiv S_1$, and $S_0 \equiv S_1$
 - ▣ But $S_0 \neq S_1$ (outputs differ), so the last condition is not satisfied and $S_0 \neq S_2$

Present state	Next state				Present output (Z_1Z_2)			
	$X_1X_2 = 00$	01	10	11	$X_1X_2 = 00$	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

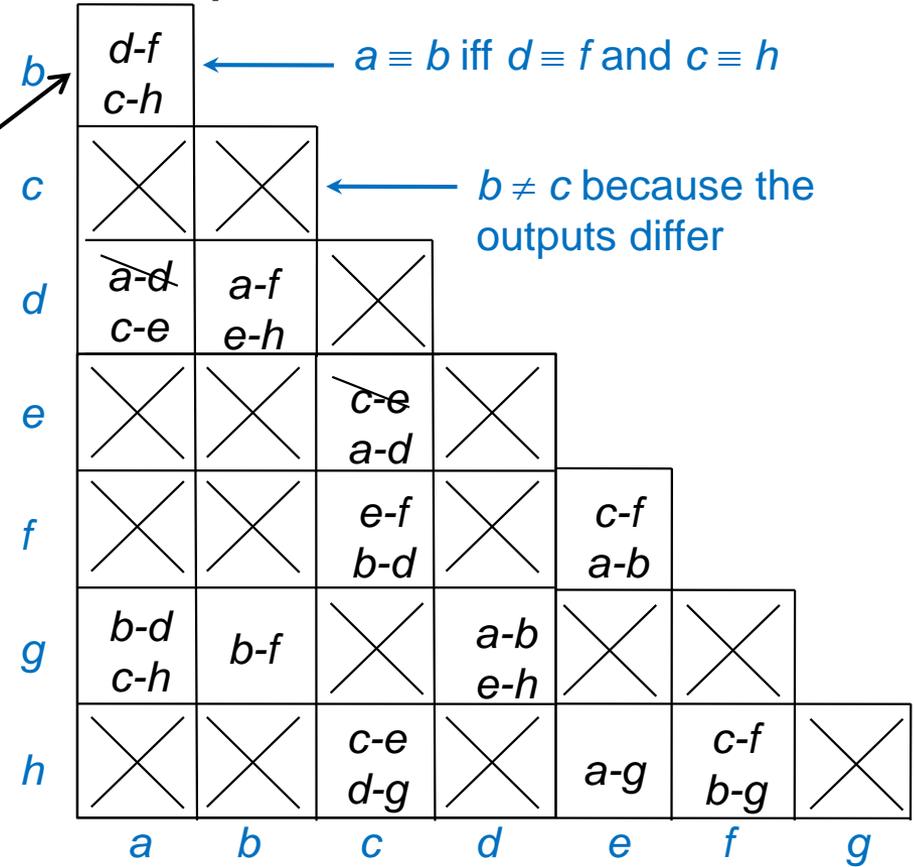
15

Implication Table

Implication Table Construction

1. Draw an empty table, where each square corresponds to a pair
2. If outputs are different, give it an X (impossible!)
3. Write down the implied pair in the square
4. Delete self-implied pairs (redundant)

Present state	Next state		Present output
	X = 0	X = 1	
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

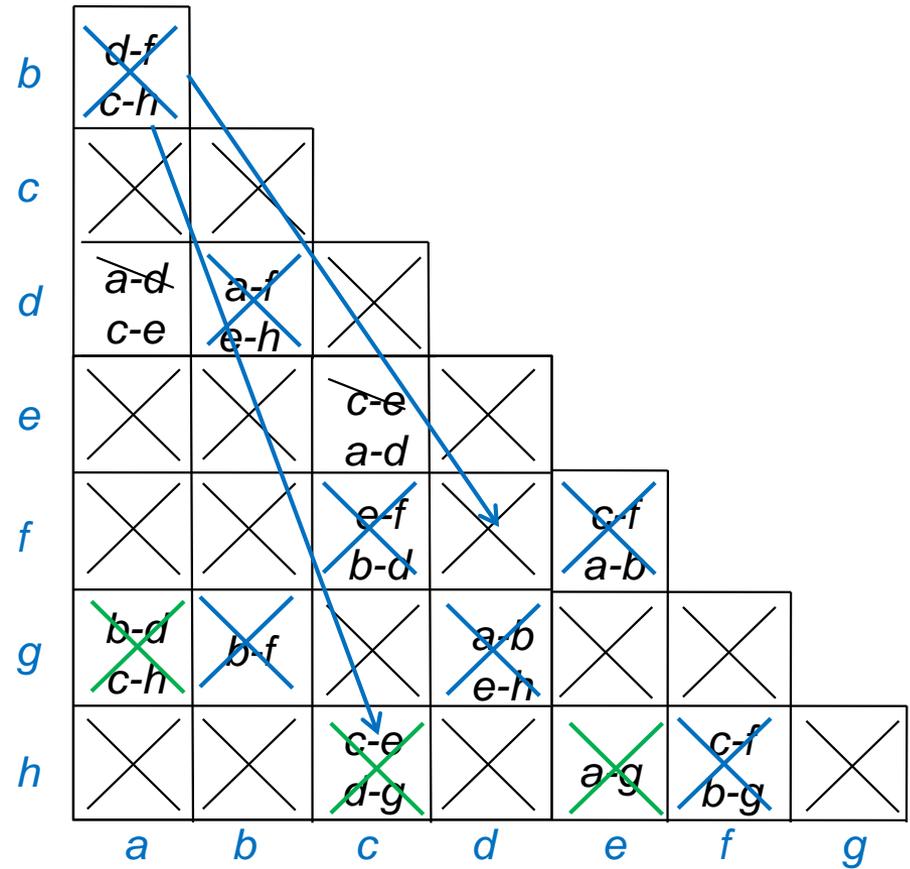


Reduction: State Comparison

5. Iteratively compare states by the implied pairs

- ▣ Only for the same output)
- ▣ First pass ~~X~~
- ▣ Second pass ~~X~~

Present state	Next state		Present output
	X = 0	X = 1	
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



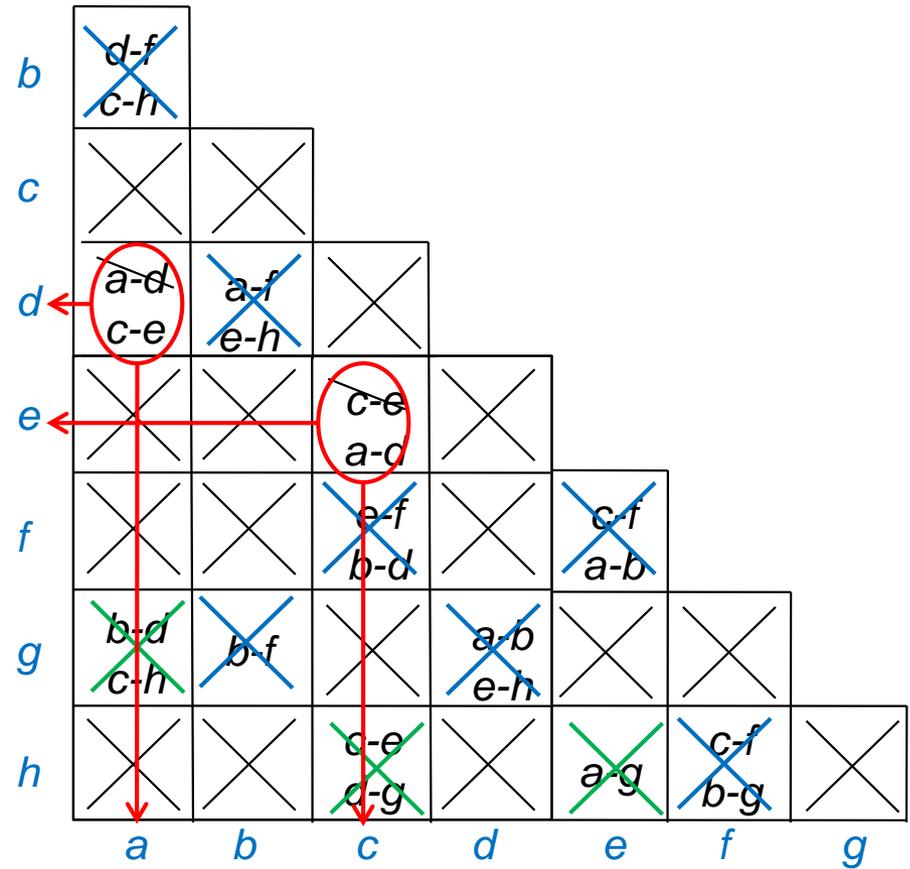
The Simplified State Table

6. Find equivalent states

- ▣ For each square $i-j$ which does not contain an X, $i \equiv j$
- ▣ The same procedure for Moore and Mealy machines

Present state	Next state		Present output
	X = 0	X = 1	
a	d a	c	0
b	f	h	0
c	e c	d a	1
d	a	e	0
e	c	a	1
f	f	b	1
g	b	h	0
h	c	g	1

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



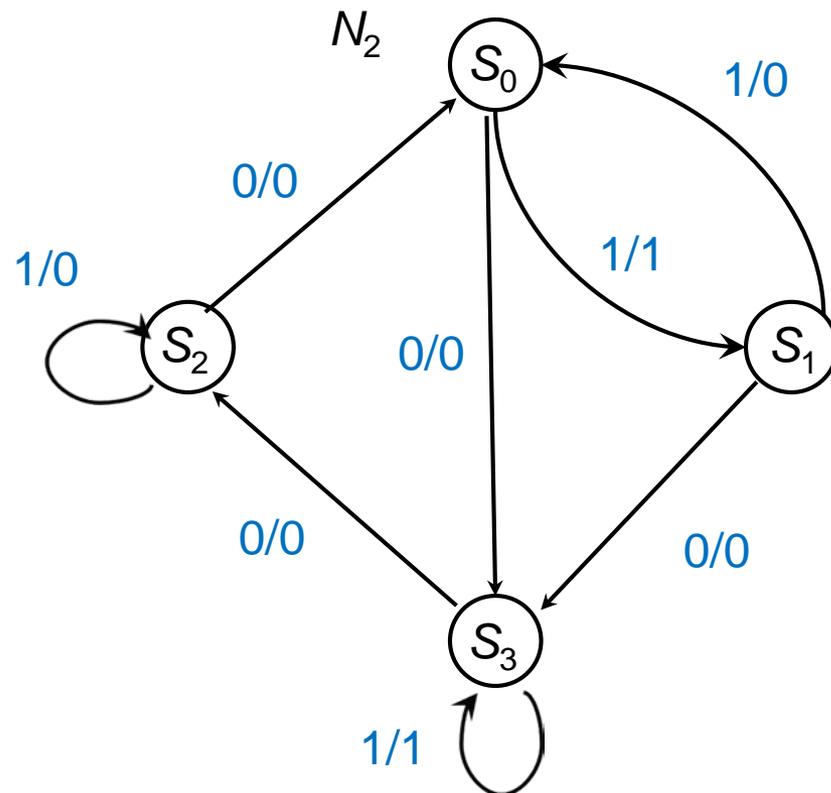
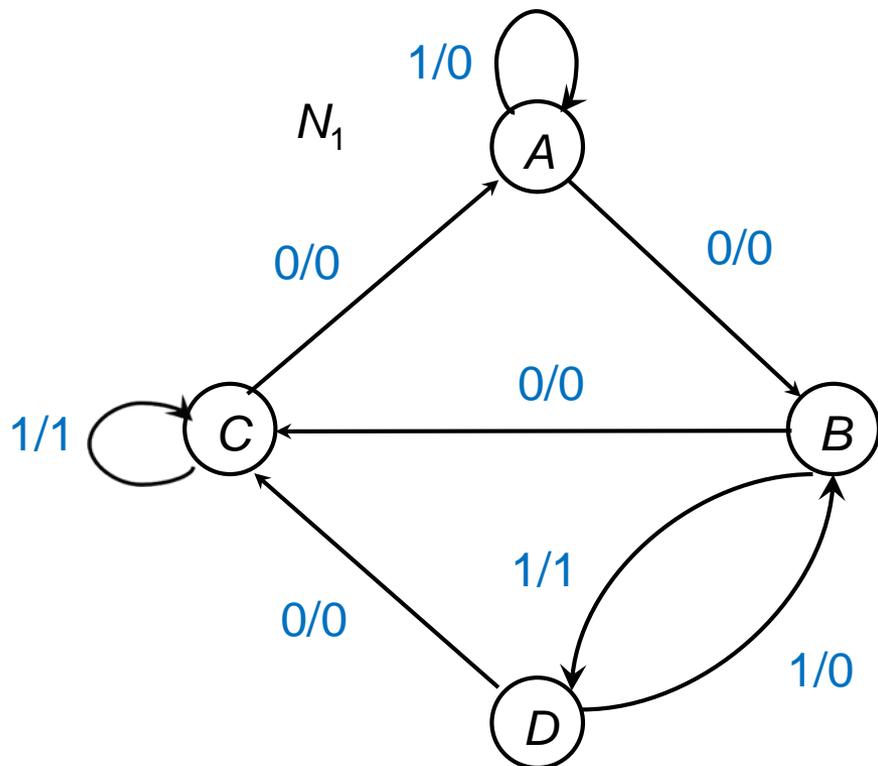
Reduction of state tables

19

Equivalent Sequential Circuits

Equivalent Sequential Circuits

- **Def: Equivalent sequential circuits**
- **Two sequential circuits are equivalent: $N_1 \equiv N_2$ if**
 - ▣ For each state p in N_1 , there is a state q in N_2 such that $p \equiv q$ and
 - ▣ For each state s in N_2 , there is a state t in N_1 such that $s \equiv t$.



Reduction of state tables

Implication Table

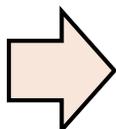
□ $N_1 \equiv N_2?$

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

Similar method 

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	B	C	D

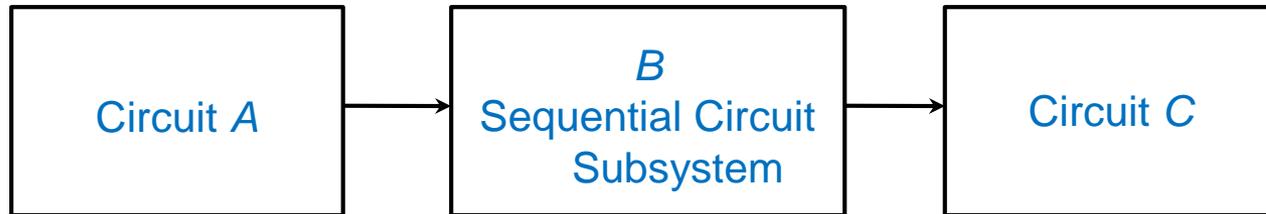
$A \equiv S_2$
 $B \equiv S_0$
 $C \equiv S_3$
 $D \equiv S_1$

 $N_1 \equiv N_2$

22

Incompletely Specified State Tables

How about Don't Cares?

- Certain sequences will never occur as inputs to the sequential circuit **B**
 - These sequences are don't cares



- A state table is **incompletely specified** if don't cares are present



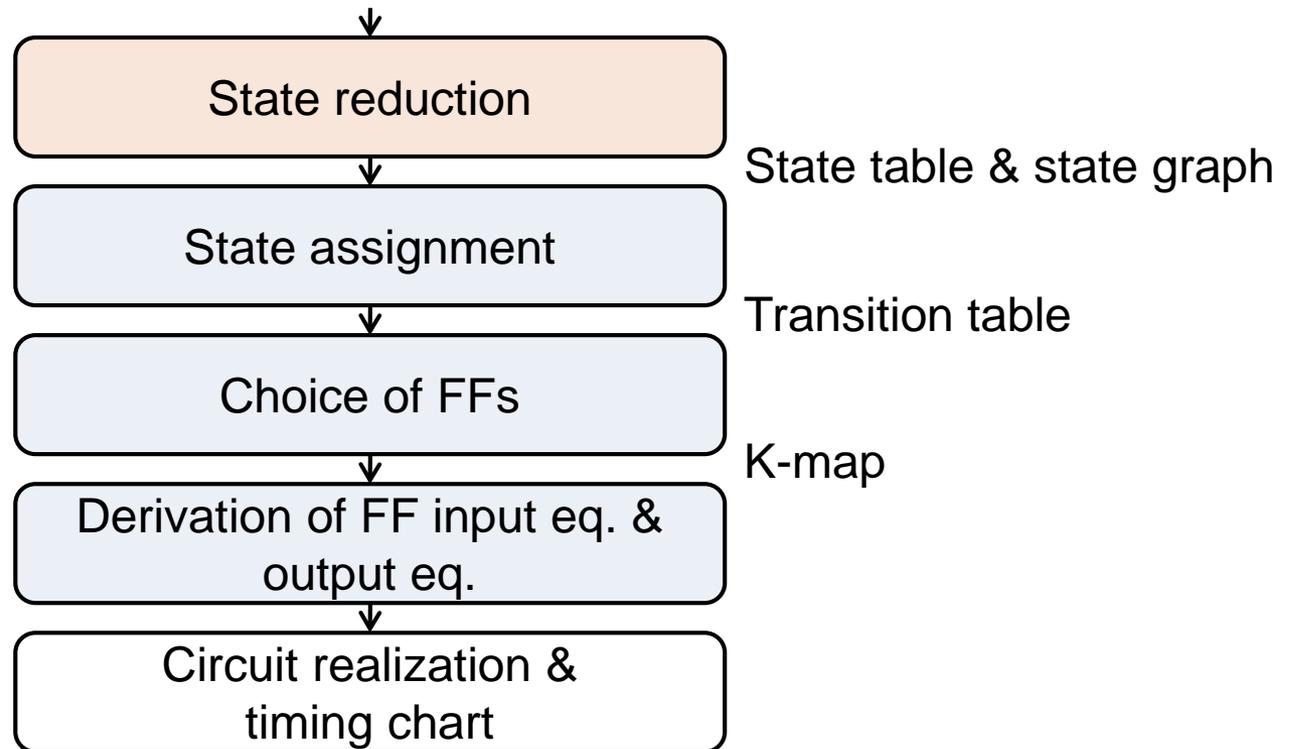
	X=0		1			X=0		1			X=0		1	
S_0	-	S_1	-	-	S_0	(S_0)	S_1	(0)	-	S_0	S_0	S_1	0	-
S_1	S_2	S_3	-	-	S_1	S_0	S_3	(1)	-	S_1	S_0	S_1	1	-
S_2	S_0	-	0	-	S_2	S_0	(S_1)	0	-					
S_3	S_0	-	1	-	S_3	S_0	(S_3)	1	-					

Reduction of state tables

Derivation of FF input equations

Equivalent state assignments

Recap: Designing a Sequential Circuit



State Assignment

□ State table:

	X=0	1	X=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

□ \Rightarrow 7 states \Rightarrow 3 FFs: A B C

□ State assignment:

□ S₀=000, S₁=110, S₂=001, S₃=111, S₄=011, S₅=101, S₆=010

Transition Table

□ State table + State assignment = Transition table

	X = 0	1	X = 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



S₀=000, S₁=110, S₂=001,
 S₃=111, S₄=011, S₅=101,
 S₆=010

ABC	A+B+C+		Z	
	X = 0	1	X = 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
100	---	---	-	-

Using D FFs

28

□ **D FF: $Q^+ = D$**

ABC	$A^+B^+C^+$		Z	
	X=0	1	X=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
100	---	---	-	-

© Iris H.-R. Jang

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

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

Reduction of state tables

Recap: Derivation of FF Input Equations

- Determine the FF input equations from the **next-state equations** using **K-maps (Unit 12)**
 - Always **copy X's** from next state maps onto input maps first

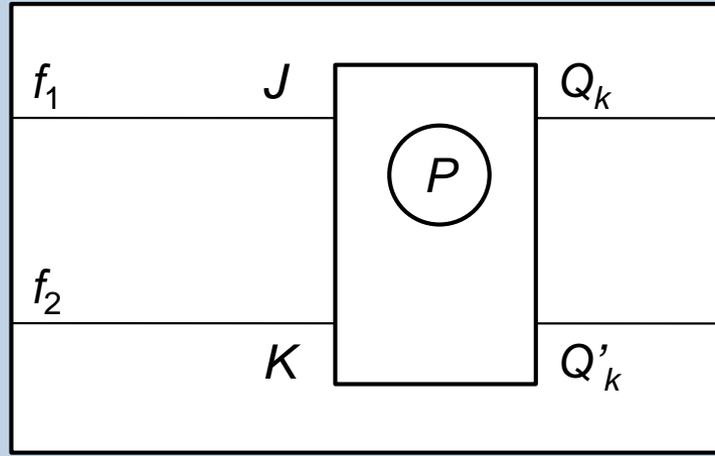
Type of F/F	Input	Q=0		Q=1		Rules for forming input map from next state map	
		Q ⁺ =0	Q ⁺ =1	Q ⁺ =0	Q ⁺ =1	Q=0 Half of Map	Q=1 Half of Map
D	D	0	1	0	1	No change	No change
T	T	0	1	1	0	No change	Complement
S-R	S	0	1	0	x	No change	Replace 1's with x's
	R	x	0	1	0	Replace 0's with x's Replace 1's with 0's	Complement
J-K	J	0	1	x	x	No change	Fill in with x's
	K	x	x	1	0	Fill in with x's	Complement

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

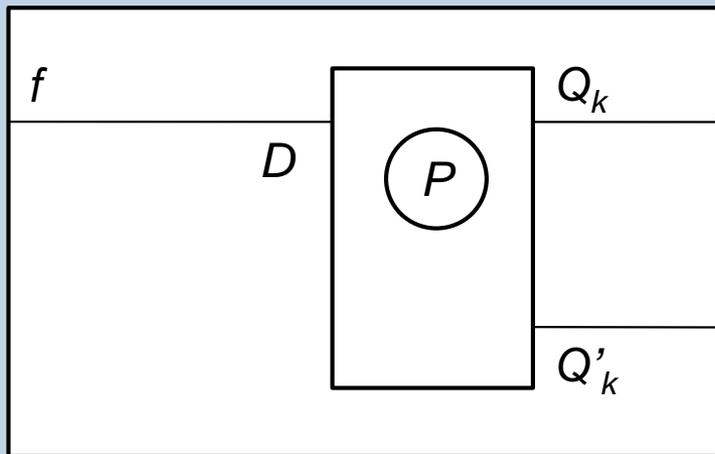
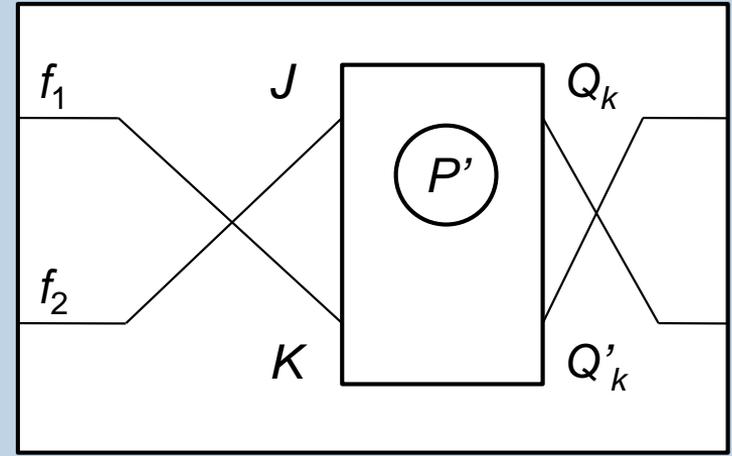
Table 15-8. State Assignments for 3-Row Tables

	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

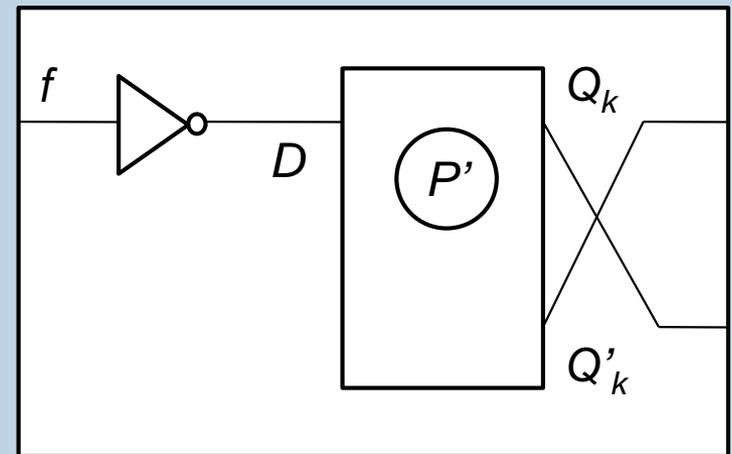
Equivalent State Assignments



\equiv



\equiv



Equivalent State Assignment

Assignments			Present State	Next State		Output	
A ₃	B ₃	C ₃		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

$$J_1 = XQ_2'$$

$$K_1 = X'$$

$$J_2 = X'Q_1$$

$$K_2 = X$$

$$Z = X'Q_1 + XQ_2$$

$$D_1 = XQ_2'$$

$$D_2 = X'(Q_1 + Q_2)$$

Assignment B

$$J_1 = XQ_1'$$

$$K_1 = X'$$

$$J_2 = X'Q_2$$

$$K_2 = X$$

$$Z = X'Q_2 + XQ_1$$

$$D_1 = XQ_1'$$

$$D_2 = X'(Q_2 + Q_1)$$

Assignment C

$$K_1 = XQ_2$$

$$J_1 = X'$$

$$K_2 = X'Q_1'$$

$$J_2 = X$$

$$Z = X'Q_1' + XQ_2'$$

$$D_1 = X' + Q_2'$$

$$D_2 = X + Q_1Q_2$$

Distinct State Assignments for 3 or 4 States

States	3-State Assignments			4-State Assignments		
	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	$\approx 5.5 \times 10^{10}$

35

Guidelines for State Assignments

State Assignment

- **State assignment determines the cost of the logic required to realize a sequential circuit**
 - ▣ A good state assignment leads to a K-map that can easily be simplified and results in few terms
- **Idea: Place 1's together (or 0's together) on the next-state maps**

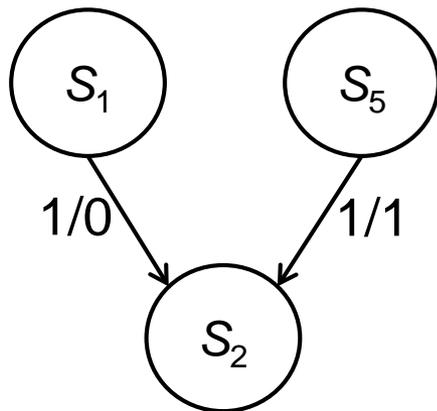
Guidelines (1/2)

37

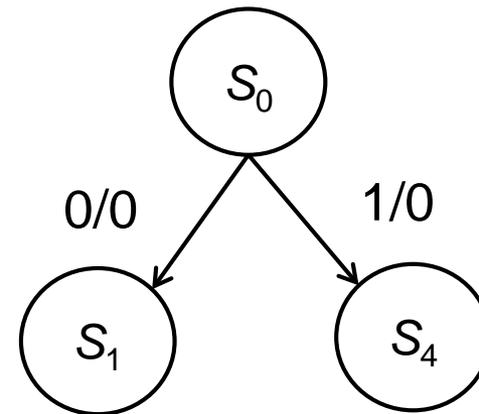
© Iris H.-R. Jiang

□ Guidelines

- ▣ No guarantee a minimum solution
 - ▣ Work for D & J-K, not for T and S-R
1. **For a given input, states with the same next state should be given adjacent assignments**
 2. **States which are next states of the same state should be given adjacent assignment**



S_1 & S_5 should be adjacent

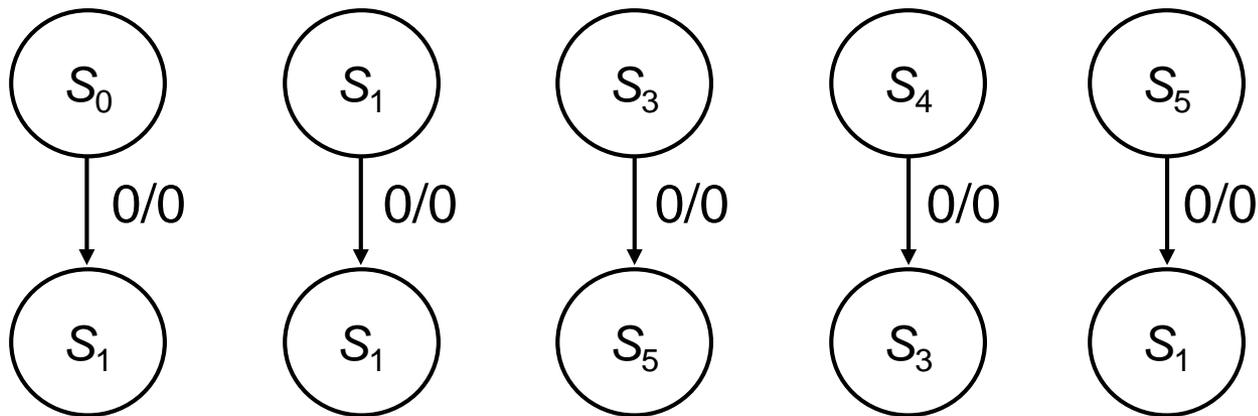


S_1 & S_4 should be adjacent

Guidelines (2/2)

3. States which have the same output \Rightarrow adjacent states

- Place 1's together on the output maps



S_0, S_1, S_3, S_4, S_5 should be adjacent

Using An Assignment Map

- List relationship according to Guidelines 1 & 2,
 1. Same NS: (S_0, S_1, S_3, S_5) (S_3, S_5) (S_4, S_6) (S_0, S_2, S_4, S_6)
 2. Same PS: (S_1, S_2) (S_2, S_3) (S_1, S_4) $(S_2, S_5) \times 2$ $(S_1, S_6) \times 2$

ABC		X=0	1	X=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

Assignment I
with guidelines

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

6 gates 13 literals

Assignment II
straight forward

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

10 gates 39 literals

Next-State Maps (1/2)

ABC		X = 0	1	X = 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

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

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

Next-State Maps (2/2)

		XA			
		00	01	11	10
BC	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

		A	
		0	1
BC	00	S ₀	
	01	S ₂	S ₅
	11	S ₄	S ₃
	10	S ₆	S ₁

These pairs are adjacent S₂ and S₅ have adjacent assignments

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

Reduction of Don't-Cares

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

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

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

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

42

One-Hot State Assignment

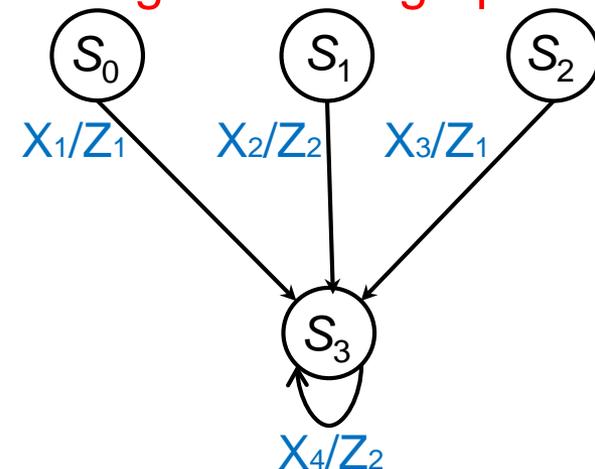
One FF for each state

One-Hot State Assignment

43

© Iris H.-R. Jiang

- **Different strategy!**
 - Do not care about how many FFs used
 - e.g., CPLD and FPGAs
 - Only care about the speed
- **One-hot state assignment: One FF for each state**
- **e.g.,**
 - 4 states use 4 FFs (Q_0, Q_1, Q_2, Q_3)
 - $S_0: Q_0 Q_1 Q_2 Q_3 = 1000, S_1: 0100, S_2: 0010, S_3: 0001$
 - **Write next-state & output eq. directly by inspecting the state graph**
 - $Q_3^+ = X_1 Q_0 + X_2 Q_1 + X_3 Q_2 + X_4 Q_3$



One-Hot + Moore Machine

44

© Iris H.-R. Jiang

- **3 states use 3 FFs (Q_0, Q_1, Q_2)**
 - $S_0: Q_0Q_1Q_2=100, S_1: 010, S_2: 001$
- **Write next-state & output eq. directly by inspecting the state graph**

- $Q_0^+ = F'R'Q_0 + FQ_2 + F'RQ_1$
- $Q_1^+ = F'R'Q_1 + FQ_0 + F'RQ_2$
- $Q_2^+ = F'R'Q_2 + FQ_1 + F'RQ_0$
- $Z_1 = Q_0$
- $Z_2 = Q_1$
- $Z_3 = Q_2$

