ECE/CS 3700: Fundamentals of Digital System Design

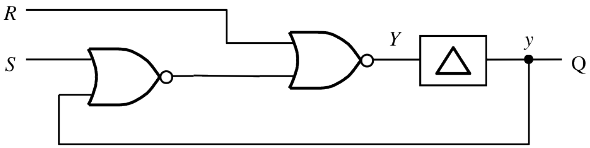
Chris J. Myers

Lecture 9: Asynchronous Sequential Circuits

Asynchronous Sequential Circuits

- *Asynchronous sequential circuits* do not use a clock or flip-flops for state variables.
- Changes in state occur in response to changes on the inputs.
- This chapter describes *single input change* (SIC) *fundamental-mode* asynchronous circuits.
- Only one input allowed to change at a time.
- Time between input changes must be sufficient for the circuit to stabilize.

Analysis of the SR Latch



(b) Circuit with modeled gate delay

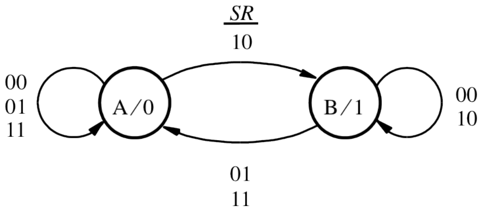
Present state y	Next state $SR = 00$ Y	01 Y	10 Y	11 Y
0	0	0	1	0
1	1	0	1	0

(b) State-assigned table

FSM Model for the SR Latch

Present state	Next state				Output Q
	$SR =$ 00	01	10	11	
A	A	A	B	A	0
B	B	A	B	A	1

(a) State table

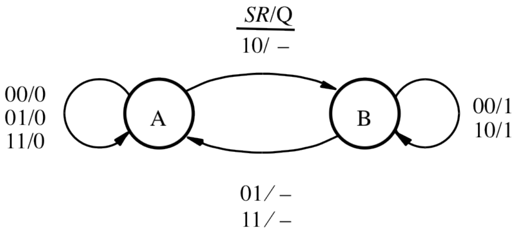


(b) State diagram

Mealy Representation of the SR Latch

Present state	Next state				Output, Q			
	SR = 00	01	10	11	00	01	10	11
A	A	A	B	A	0	0	–	0
B	B	A	B	A	1	–	1	–

(a) State table

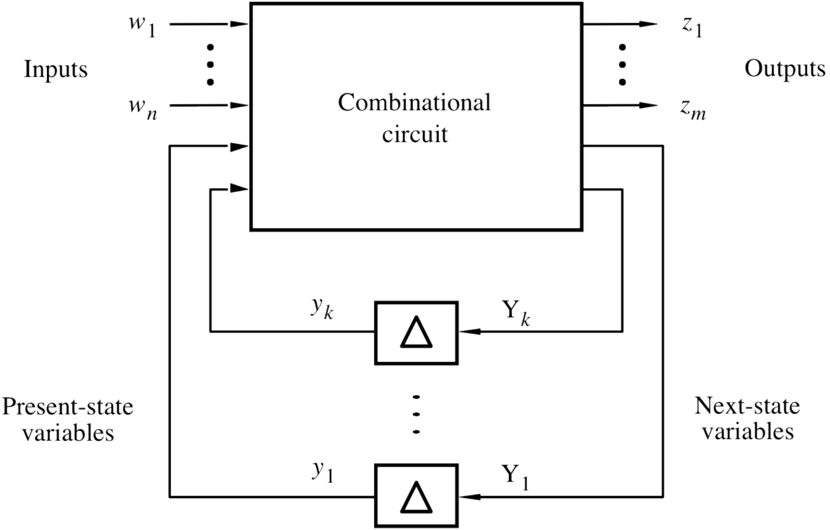


(b) State diagram

Terminology

- state table = *flow table*.
- state-assigned table = *transition table* or *excitation table*.

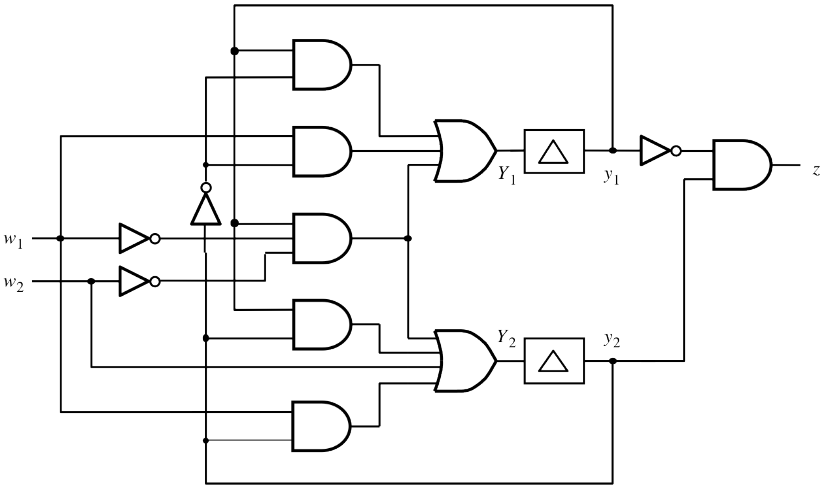
General Model of a Sequential Circuit



Analysis of Asynchronous Circuits

- Cut feedback paths and insert delay element.
  - Input to delay element is next-state, and output is the present-state.
  - *Cut set* may not be unique.
- Derive next-state and output expressions from the circuit.
- Derive the excitation table.
- Derive a flow table.
- Derive a state diagram, if desired.

An Asynchronous Circuit



Excitation and Flow Tables

Present state $y_2 y_1$	Next state $w_2 w_1 = 00$ $Y_2 Y_1$	01 $Y_2 Y_1$	10 $Y_2 Y_1$	11 $Y_2 Y_1$	Output z
00	00	01	10	11	0
01	11	01	11	11	0
10	00	10	10	10	1
11	11	10	10	10	0

(a) Excitation table

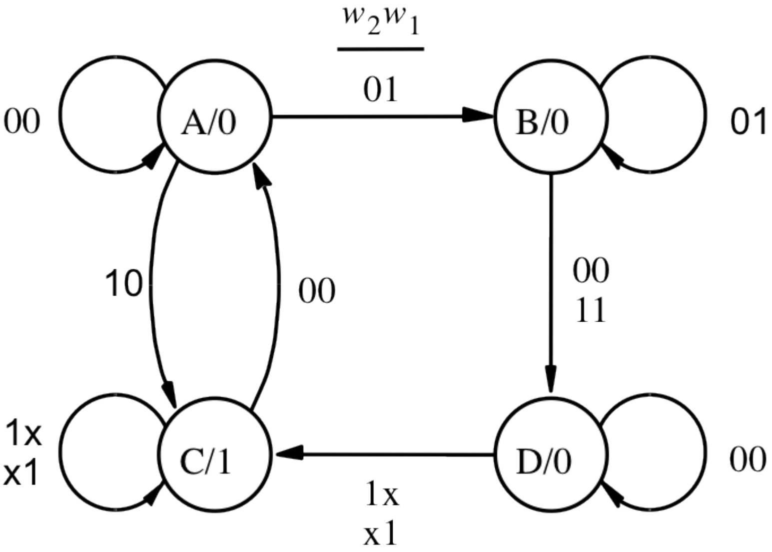
Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	B	C	D	0
B	D	B	D	D	0
C	A	C	C	C	1
D	D	C	C	C	0

(b) Flow table

Modified Flow Table

Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	B	C	—	0
B	D	B	—	D	0
C	A	C	C	C	1
D	D	C	C	C	0

State Diagram



Flow Table for a Simple Vending Machine

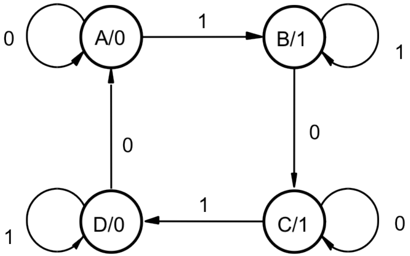
Present state	Next state				Output
	$w_2 w_1$ = 00	01	10	11	z
A	A	B	C	—	0
B	D	B	—	—	0
C	A	C	C	—	1
D	D	C	C	—	0

$w_2 \equiv$ dime $\quad w_1 \equiv$ nickel

Synthesis of Asynchronous Circuits

- Devise a state diagram.
- Derive the flow table.
- Minimize number of states.
- Perform *race-free* state assignment.
- Derive excitation table.
- Obtain next-state and output expressions.
- Construct a *hazard-free* circuit.

Parity-generating Asynchronous FSM



(a) Statediagram

Present State	Nextstate $w = 0$	$w = 1$	Output z
A	A	B	0
B	C	B	1
C	C	D	1
D	A	D	0

(b) Flow table

State Assignment

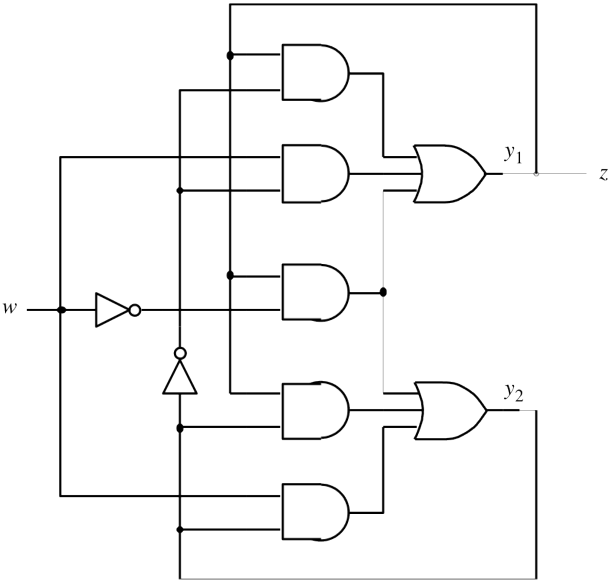
Present state $y_2 y_1$	Nextstate $w = 0$		Output z
	$Y_2 Y_1$ ($w = 0$)	$Y_2 Y_1$ ($w = 1$)	
00	00	01	0
01	10	01	1
10	10	11	1
11	00	11	0

(a) Poor state assignment

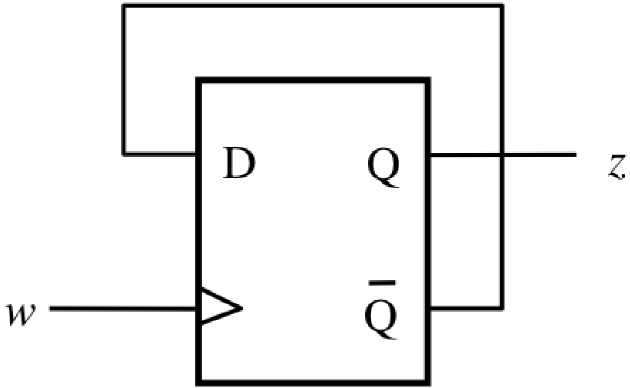
Present state $y_2 y_1$	Nextstate $w = 0$		Output z
	$Y_2 Y_1$ ($w = 0$)	$Y_2 Y_1$ ($w = 1$)	
00	00	01	0
01	11	01	1
11	11	10	1
10	00	10	0

(b) Good state assignment

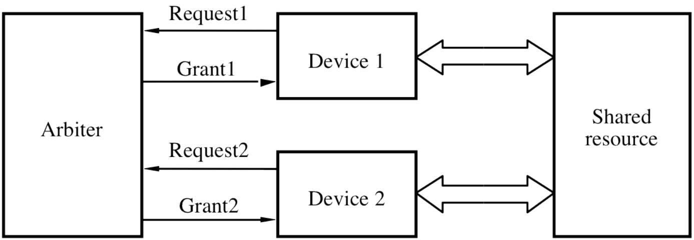
Circuit that Implements the FSM



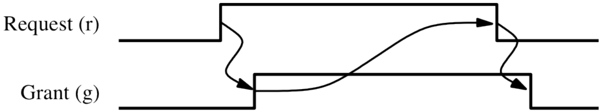
Synchronous Solution



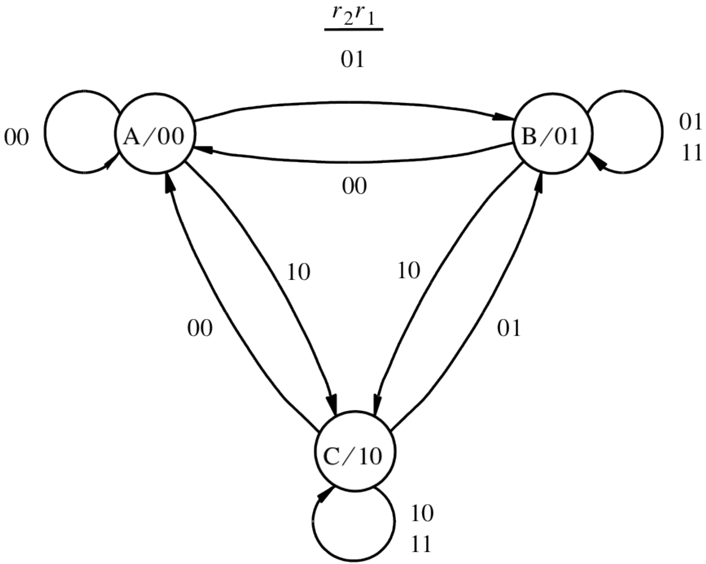
Arbitration Example



(a) Arbitration structure



(b) Handshake signaling


$\underline{r_2 r_1}$
01
00
A/00
B/01
01
11
00
10
10
00
01
C/10
10
11

Implementation of the Arbiter

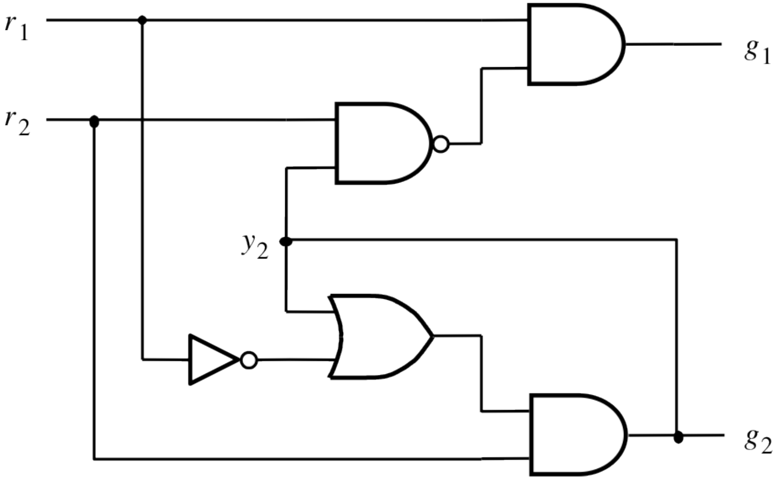
Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	A	B	C	—	00
B	A	B	C	B	01
C	A	B	C	C	10

(a) Flow table

	Present state y_2y_1	Next state r_2r_1 = 00 (Y_2Y_1)	01	10	11	Output g_2g_1
A	00	00	01	10	—	00
B	01	00	01	10	01	01
C	10	00	01	10	10	10
D	11	—	01	10	—	dd

(b) Excitation table

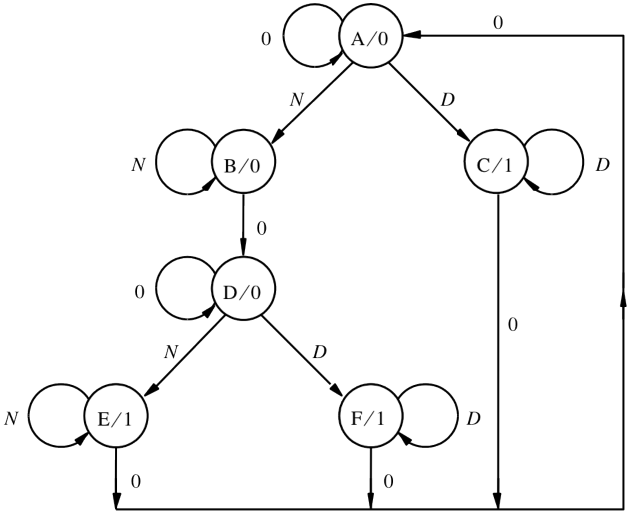
THIS CIRCUIT IS WRONG!



State Reduction

- Usually start with *primitive flow table* (i.e., a single stable state per row).
- Asynchronous FSMs likely have many unspecified (don't care) entries.
- Two-step state reduction process:
  - Apply partitioning procedure in which don't care entries must match.
  - Merge rows exploiting don't cares.

Derivation of an FSM for the Simple Vending Machine



(a) Initial state diagram

Derivation of an FSM for the Simple Vending Machine

Present State	Next state				Output
	DN = 00	01	10	11	z
A	A	B	C	–	0
B	D	B	–	–	0
C	A	–	C	–	1
D	D	E	F	–	0
E	A	E	–	–	1
F	A	–	F	–	1

(b) Initial flow table

Derivation of an FSM for the Simple Vending Machine

Present State	Next state				Output
	DN = 00	01	10	11	z
A	A	B	C	–	0
B	D	B	–	–	0
C	A	–	C	–	1
D	D	E	F	–	0
E	A	E	–	–	1
F	A	–	F	–	1

(b) Initial flow table

$$P_1 = (AD)(B)(CF)(E)$$

Derivation of an FSM for the Simple Vending Machine

Present State	Next state DN = 00	01	10	11	Output z
A	A	B	C	–	0
B	D	B	–	–	0
C	A	–	C	–	1
D	D	E	F	–	0
E	A	E	–	–	1
F	A	–	F	–	1

(b) Initial flow table

$$
\begin{aligned}
P_1 &= (AD)(B)(CF)(E) \\
P_2 &= (A)(D)(B)(CF)(E)
\end{aligned}
$$

Derivation of an FSM for the Simple Vending Machine

Present State	Next state DN = 00	01	10	11	Output z
A	A	B	C	–	0
B	D	B	–	–	0
C	A	–	C	–	1
D	D	E	F	–	0
E	A	E	–	–	1
F	A	–	F	–	1

(b) Initial flow table

$$
\begin{aligned}
P_1 &= (AD)(B)(CF)(E) \\
P_2 &= (A)(D)(B)(CF)(E) \\
P_3 &= P_2
\end{aligned}
$$

First-step Reduction of the Vending Machine FSM

Present state	Next state				Output
	DN = 00	01	10	11	z
A	A	B	C	—	0
B	D	B	—	—	0
C	A	—	C	—	1
D	D	E	C	—	0
E	A	E	—	—	1

Flow Table for a Simple Vending Machine

Present state	Next state				Output
	$w_2 w_1$ = 00	01	10	11	z
A	A	B	C	—	0
B	D	B	—	—	0
C	A	C	C	—	1
D	D	C	C	—	0

$w_2 \equiv$ dime $\quad w_1 \equiv$ nickel

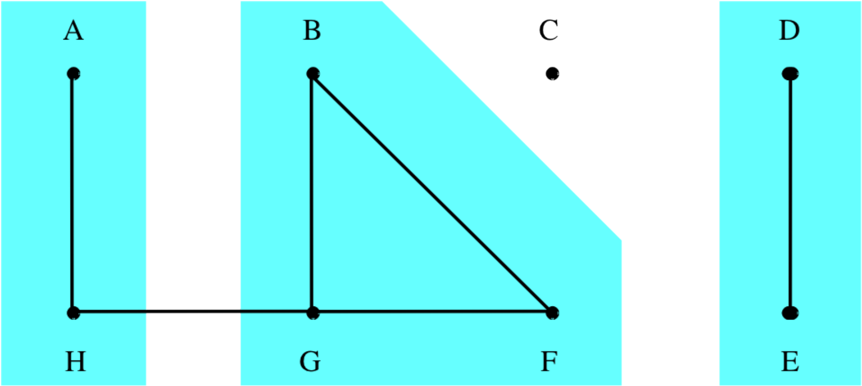
Merging Procedure

- May be many possibilities for row mergers.
- Two states S_i and S_j are compatible if there are no state conflicts for any input valuation. For each input valuation, one of the following is true:
  - Both S_i and S_j have same successor, or
  - Both S_i and S_j are stable, or
  - The successor of S_i or S_j or both is unspecified.
- S_i and S_j must also have same output when specified.

A Primitive Flow Table

Present state	Next state w_2w_1 = 00	01	10	11	Output z
A	A	H	B	—	0
B	F	—	B	C	0
C	—	H	—	C	1
D	A	D	—	E	1
E	—	D	G	E	1
F	F	D	—	—	0
G	F	—	G	—	0
H	—	H	—	E	0

Merger Diagram which Preserves the Moore Model



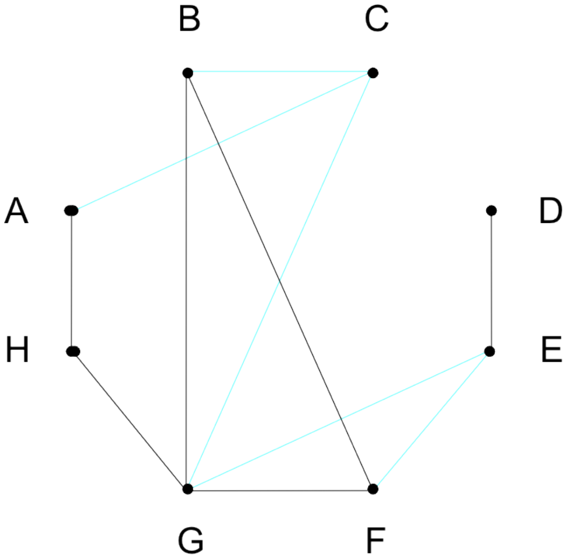
Reduce Moore-type Flow Table

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z
A	A	A	B	D	0
B	B	D	B	C	0
C	—	A	—	C	1
D	A	D	B	D	1

A Primitive Flow Table

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z
A	A	H	B	—	0
B	F	—	B	C	0
C	—	H	—	C	1
D	A	D	—	E	1
E	—	D	G	E	1
F	F	D	—	—	0
G	F	—	G	—	0
H	—	H	—	E	0

Complete Merger Diagram



State Reduction Procedure

1. Use partitioning to eliminate equivalent states in primitive flow table.
2. Construct merger diagram.
3. Choose subsets of equivalent states with each state in only one subset.
4. Derive reduced flow table.
5. Repeat 2 to 4 until no reduction.

Flow Table for Example 9.8

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	A	F	C	—	0
B	A	B	—	H	1
C	G	—	C	D	0
D	—	F	—	D	1
E	G	—	E	D	1
F	—	F	—	K	0
G	G	B	J	—	0
H	—	L	E	H	1
J	G	—	J	—	0
K	—	B	E	K	1
L	A	L	—	K	1

Flow Table for Example 9.8

Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	F	C	—	0
B	A	B	—	H	1
C	G	—	C	D	0
D	—	F	—	D	1
E	G	—	E	D	1
F	—	F	—	K	0
G	G	B	J	—	0
H	—	L	E	H	1
J	G	—	J	—	0
K	—	B	E	K	1
L	A	L	—	K	1

$$P_1 = (AG)(BL)(C)(D)(E)(F)(HK)(J)$$

Flow Table for Example 9.8

Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	F	C	—	0
B	A	B	—	H	1
C	G	—	C	D	0
D	—	F	—	D	1
E	G	—	E	D	1
F	—	F	—	K	0
G	G	B	J	—	0
H	—	L	E	H	1
J	G	—	J	—	0
K	—	B	E	K	1
L	A	L	—	K	1

$$P_1 = (AG)(BL)(C)(D)(E)(F)(HK)(J)$$
$$P_2 = (A)(G)(BL)(C)(D)(E)(F)(HK)(J)$$

Flow Table for Example 9.8

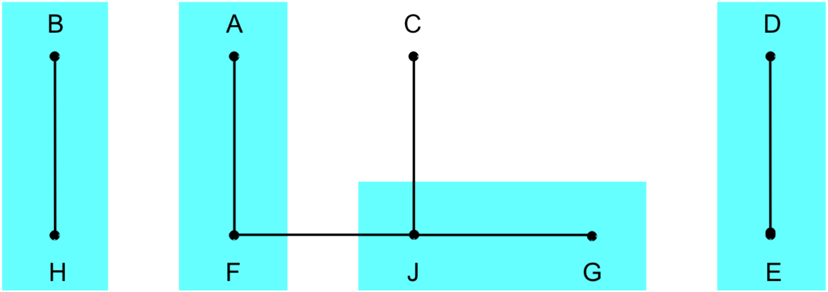
Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	F	C	—	0
B	A	B	—	H	1
C	G	—	C	D	0
D	—	F	—	D	1
E	G	—	E	D	1
F	—	F	—	K	0
G	G	B	J	—	0
H	—	L	E	H	1
J	G	—	J	—	0
K	—	B	E	K	1
L	A	L	—	K	1

$$P_1 = (AG)(BL)(C)(D)(E)(F)(HK)(J)$$
$$P_2 = (A)(G)(BL)(C)(D)(E)(F)(HK)(J)$$
$$P_3 = P_2$$

Reduction Obtained by Using the Partitioning Procedure

Present state	Next state $w_2 w_1 = 00$	01	10	11	Output z
A	A	F	C	—	0
B	A	B	—	H	1
C	G	—	C	D	0
D	—	F	—	D	1
E	G	—	E	D	1
F	—	F	—	H	0
G	G	B	J	—	0
H	—	B	E	H	1
J	G	—	J	—	0

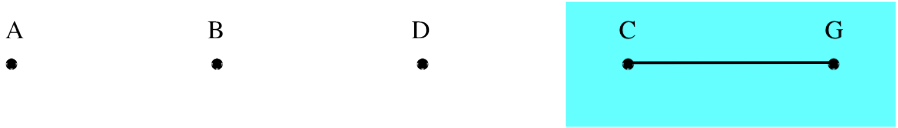
Merger Diagram



Reduction Obtained from the Merger Diagram

Present state	Next state				Output
	$w_2 w_1 = 00$	01	10	11	z
A	A	A	C	B	0
B	A	B	D	B	1
C	G	—	C	D	0
D	G	A	D	D	1
G	G	B	G	—	0

Merger Diagram



Reduced Flow Table for Example 9.8

Present state	Next state				Output
	$w_2w_1 = 00$	01	10	11	z
A	A	A	C	B	0
B	A	B	D	B	1
C	C	B	C	D	0
D	C	A	D	D	1

Flow Table for Example 9.9

Present state	Next state w_2w_1 = 00	01	10	11	Output z
A	A	G	E	—	0
B	K	—	B	D	0
C	F	C	—	H	1
D	—	C	E	D	0
E	A	—	E	D	1
F	F	C	J	—	0
G	K	G	—	D	1
H	—	—	E	H	1
J	F	—	J	D	0
K	K	C	B	—	0

Flow Table for Example 9.9

Present state	Next state $w_2 w_1$ = 00	01	10	11	Output z
A	A	G	E	—	0
B	K	—	B	D	0
C	F	C	—	H	1
D	—	C	E	D	0
E	A	—	E	D	1
F	F	C	J	—	0
G	K	G	—	D	1
H	—	—	E	H	1
J	F	—	J	D	0
K	K	C	B	—	0

$$P_1 = (AFK)(BJ)(CG)(D)(E)(H)$$

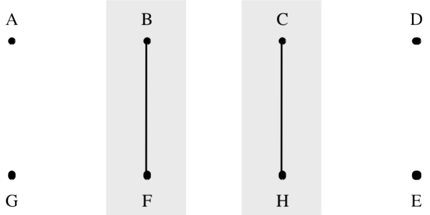
$$P_2 = (A)(FK)(BJ)(C)(G)(D)(E)(H)$$

$$P_3 = P_2$$

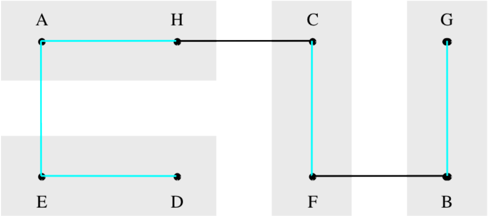
Reduction Resulting from the Partitioning Procedure

Present state	Next state				Output
	w_2w_1 = 00	01	10	11	z
A	A	G	E	—	0
B	F	—	B	D	0
C	F	C	—	H	1
D	—	C	E	D	0
E	A	—	E	D	1
F	F	C	B	—	0
G	F	G	—	D	1
H	—	—	E	H	1

Merger Diagrams



(a) Preserving the Moore model



(b) Complete merger diagram

An FSM Specified in the Form of a Mealy Model

Present state	Next state				Output z			
	$w_2 w_1$ = 00	01	10	11	00	01	10	11
A	A	G	E	–	0	–	–	–
B	F	–	B	D	0	–	0	0
C	F	C	–	H	–	1	–	1
D	–	C	E	D	–	–	–	0
E	A	–	E	D	–	–	1	–
F	F	C	B	–	0	–	0	–
G	F	G	–	D	–	1	–	–
H	–	–	E	H	–	–	1	1

Reduced Flow Table for Example 9.9

Present state	Next state				Output z			
	$w_2 w_1$ = 00	01	10	11	00	01	10	11
A	A	B	D	A	0	—	1	1
B	C	B	B	D	0	1	0	0
C	C	C	B	A	0	1	0	1
D	A	C	D	D	—	—	1	0

Flow Table for Example 9.10

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z
A	A	B	C	—	0
B	F	B	—	H	0
C	F	—	C	H	0
D	D	G	C	—	1
E	A	E	—	H	0
F	F	E	C	—	0
G	D	G	—	H	0
H	—	G	C	H	1

Flow Table for Example 9.10

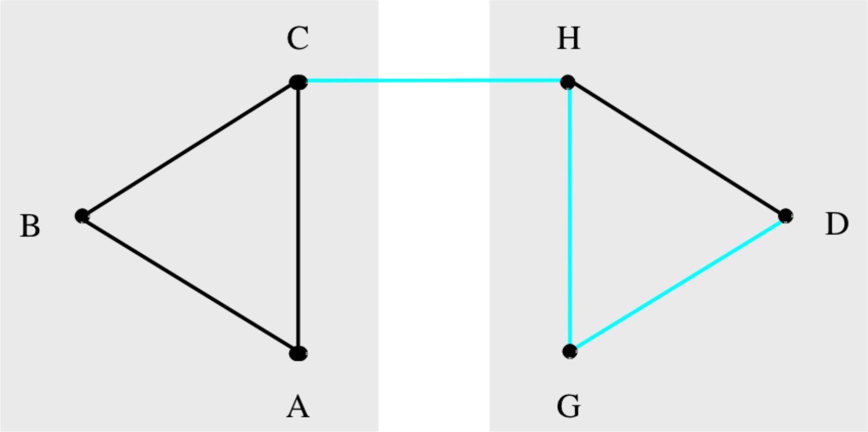
Present state	Next state $w_2w_1 = 00$	01	10	11	Output z
A	A	B	C	—	0
B	F	B	—	H	0
C	F	—	C	H	0
D	D	G	C	—	1
E	A	E	—	H	0
F	F	E	C	—	0
G	D	G	—	H	0
H	—	G	C	H	1

$$
\begin{aligned}
P_1 &= (AF)(BEG)(C)(D)(H) \\
P_2 &= (AF)(BE)(G)(C)(D)(H) \\
P_3 &= P_2
\end{aligned}
$$

Reduction after the Partitioning Procedure

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z
A	A	B	C	—	0
B	A	B	—	H	0
C	A	—	C	H	0
D	D	G	C	—	1
G	D	G	—	H	0
H	—	G	C	H	1

Merger Diagram



Reduced Flow Table for Example 9.10

Present state	Next state				Output z			
	$w_2 w_1 =$ 00	01	10	11	00	01	10	11
A	A	A	A	D	0	0	0	—
D	D	D	A	D	1	0	—	1

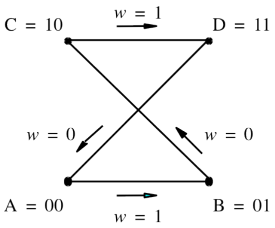
State Assignment

- Impossible to ensure that a change of two or more state variables occur at the same time.
- To achieve reliable operation, should make state variables change one at a time.
- *Hamming distance* is number of bits different in two bit strings.
- Ideal state assignment has Hamming distance of 1 for all state transitions.

Transitions

Present state y_2y_1	Next state $w = 0$ (Y_2Y_1)	Next state $w = 1$ (Y_2Y_1)	Output z
00	00	01	0
01	10	01	1
10	10	11	1
11	00	11	0

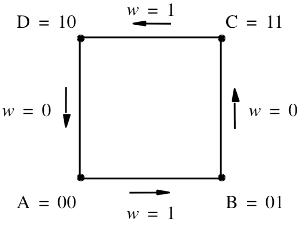
(a) Poor state assignment



(a) Corresponding to Figure 9.14 *a*

Present state y_2y_1	Next state $w = 0$ (Y_2Y_1)	Next state $w = 1$ (Y_2Y_1)	Output z
00	00	01	0
01	11	01	1
11	11	10	1
10	00	10	0

(b) Good state assignment



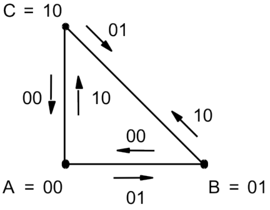
(b) Corresponding to Figure 9.14 *b*

Transitions for the Arbiter

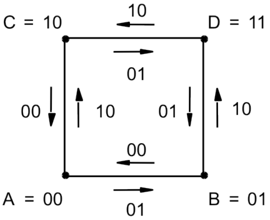
Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	A	B	C	—	00
B	A	B	C	B	01
C	A	B	C	C	10

(a) Flow table

Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	A	B	C	—	00
B	A	B	D	B	01
C	A	D	C	C	10
D	—	B	C	—	10



(a) Transitions in Figure 9.21*a*



(b) Using the extra state D

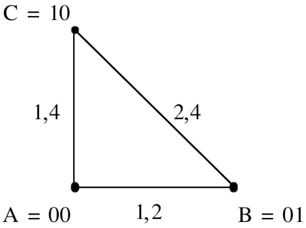
Transition Diagram

- *Transition diagram* illustrates all transitions in a flow table.
- Good state assignment means no diagonals in the transition diagram.
- Must *embed* the transition diagram onto a k-dimensional cube.
- n state variables can be embedded onto an n-dimensional cube.

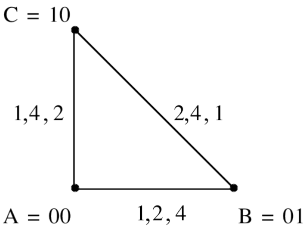
Relabeled Flow Table

Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	1	2	4	—	00
B	1	2	4	3	01
C	1	2	4	5	10

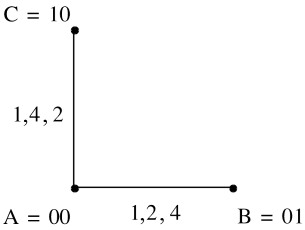
Transition Diagrams



(a) Transitions in Figure 9.50



(b) Complete transition diagram



(c) Selected transition diagram

An Alternative for Avoiding a Critical Race

Present state	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
A	A	B	C	—	00
B	A	B	A	B	01
C	A	A	C	C	10

(a) Modified flow table

Present state y_2y_1	Next state r_2r_1 = 00	01	10	11	Output g_2g_1
	Y_2Y_1				
00	00	01	10	—	00
01	00	01	00	01	01
10	00	00	10	10	10

(b) Modified excitation table

Deriving Transition Diagrams

- Derive relabeled flow table.
  - Transitions through unstable states that lead to stable state are given the same number.
- Represent each row of flow table by vertex.
- Join V_i and V_j by edge if they have same number in any column.
- For any column in which V_i and V_j have same number, label edge with that number.

Flow Tables

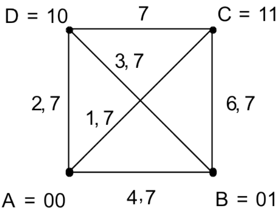
Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	A	B	C	A	00
B	B	B	D	C	01
C	A	C	D	C	10
D	B	—	D	A	11

(a) Flow table

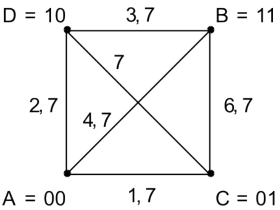
Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	1	4	7	2	00
B	3	4	7	6	01
C	1	5	7	6	10
D	3	—	7	2	11

(b) Relabeled flow table

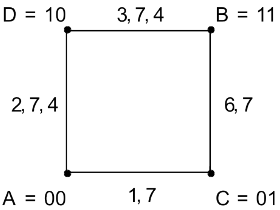
Transition Diagrams



(a) First transition diagram



(b) Second transition diagram



(c) Augmented transition diagram

Realization of an FSM

Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1 00	01	10	11
A	A	D	D	A	00	00	11	00
B	B	B	D	C	01	01	11	01
C	A	C	B	C	–0	10	1–	10
D	B	B	D	A	–1	0–	11	00

(a) Modified flow table

	Present state y_2y_1	Next state w_2w_1 = 00	01	10	11	Output 00	01	10	11
		Y_2Y_1				z_2z_1			
A	00	00	10	10	00	00	00	11	00
B	11	11	11	10	01	01	01	11	01
C	01	00	01	11	01	–0	10	1–	10
D	10	11	11	10	00	–1	0–	11	00

(b) Excitation table

FSM for Example 9.14

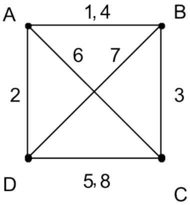
Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	A	A	C	B	00
B	A	B	D	B	01
C	C	B	C	D	10
D	C	A	D	D	11

(a) Flow table

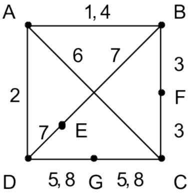
Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	1	2	6	4	00
B	1	3	7	4	01
C	5	3	6	8	10
D	5	2	7	8	11

(b) Relabeled flow table

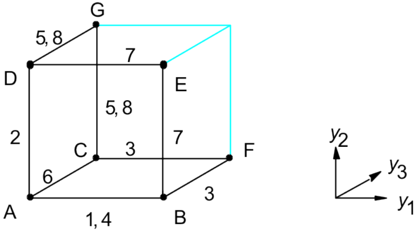
Transition Diagrams



(a) Transition diagram



(b) Augmented transition diagram



(c) Embedded transition diagram

Modified Flow Table for Example 9.14

Present state	Nextstate $w_2 w_1$ = 00	01	10	11	Output $z_2 z_1$
A	A	A	C	B	00
B	A	B	E	B	01
C	C	F	C	G	10
D	G	A	D	D	11
E	–	–	D	–	–1
F	–	B	–	–	01
G	C	–	–	D	1–

(a) Modified flow table

Modified Excitation Table for Example 9.14

	Present state $y_3 y_2 y_1$	Next state $w_2 w_1 = 00$	01	10	11	Output $z_2 z_1$
		$Y_3 Y_2 Y_1$				
A	000	000	000	100	001	00
B	001	000	001	011	001	01
C	100	100	101	100	110	10
D	010	110	000	010	010	11
E	011	–	–	010	–	–1
F	101	–	001	–	–	01
G	110	100	–	–	010	1–

(b) Excitation table

FSM for Example 9.14

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z_2z_1
A	A	A	C	B	00
B	A	B	D	B	01
C	C	B	C	D	10
D	C	A	D	D	11

(a) Flow table

Present state	Next state $w_2w_1 = 00$	01	10	11	Output z_2z_1
A	1	2	6	4	00
B	1	3	7	4	01
C	5	3	6	8	10
D	5	2	7	8	11

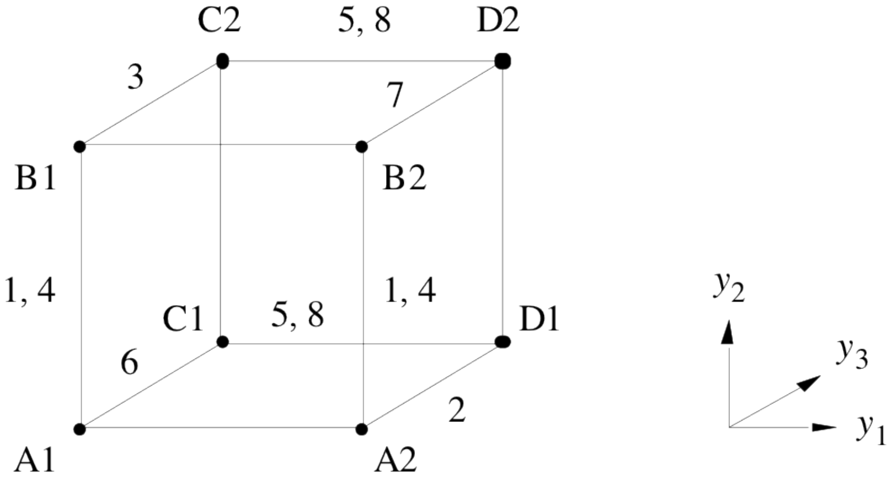
(b) Relabeled flow table

Modified Flow Table

Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A1	A1	A1	C1	B1	00
A2	A2	A2	A1	B2	00
B1	A1	B1	B2	B1	01
B2	A2	B2	D2	B2	01
C1	C1	C2	C1	D1	10
C2	C2	B1	C2	D2	11
D1	C1	A2	D1	D1	11
D2	C2	D1	D2	D2	11

(a) Modified flow table

Embedded Transition Diagram



Modified Excitation Table

	Present state $y_3y_2y_1$	Next state $Y_3Y_2Y_1$ w_2w_1 = 00	01	10	11	Output z_2z_1
A1	000	000	000	100	010	00
A2	001	001	001	000	011	00
B1	010	000	010	011	010	01
B2	011	001	011	111	011	01
C1	100	100	110	100	101	10
C2	110	110	010	110	111	10
D1	101	100	001	101	101	11
D2	111	110	101	111	111	11

(b) Excitation table

FSM for Example 9.14

Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	A	A	C	B	00
B	A	B	D	B	01
C	C	B	C	D	10
D	C	A	D	D	11

(a) Flow table

Present state	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
A	1	2	6	4	00
B	1	3	7	4	01
C	5	3	6	8	10
D	5	2	7	8	11

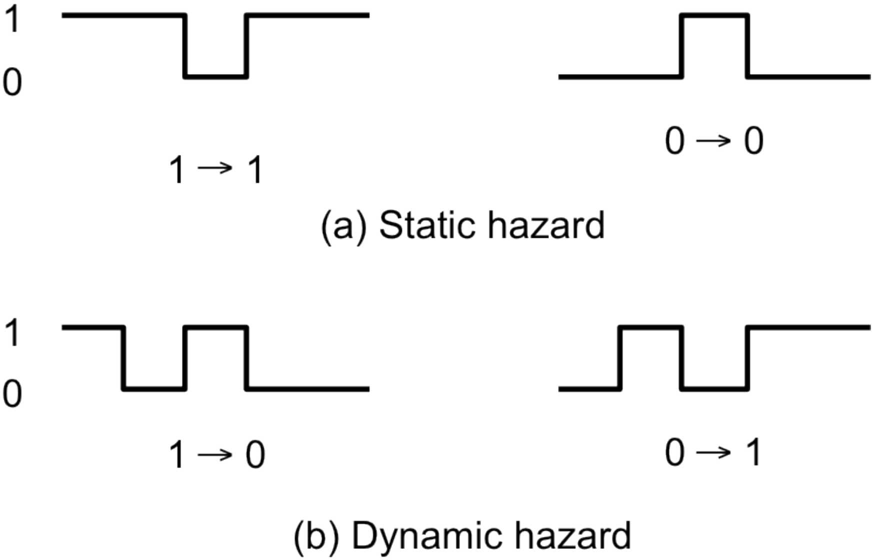
(b) Relabeled flow table

State Assignment with One-hot Encoding

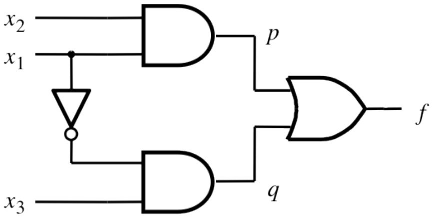
State assignment	Present State	Next state w_2w_1 = 00	01	10	11	Output z_2z_1
0001	A	A	A	E	F	00
0010	B	F	B	G	B	01
0100	C	C	H	C	I	10
1000	D	I	J	D	D	11
0101	E	—	—	C	—	–0
0011	F	A	—	—	B	0–
1010	G	—	—	D	—	–1
0110	H	—	B	—	—	01
1100	I	C	—	—	D	1–
1001	J	—	A	—	—	00

Hazards

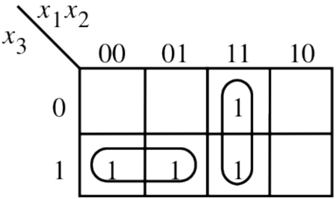
- In asynchronous circuits, undesirable *glitches* must not occur.
- Glitches caused by structure of circuit and propagation delays are called *hazards*.
- Designer must eliminate all hazards from an asynchronous circuit.


1
0
1 → 1
0 → 0
(a) Static hazard
1
0
1 → 0
0 → 1
(b) Dynamic hazard

Circuit with a Static Hazard

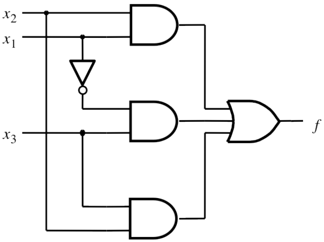


(a) Circuit with a hazard

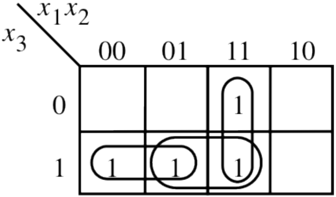


(b) Karnaugh map

Hazard-free Circuit



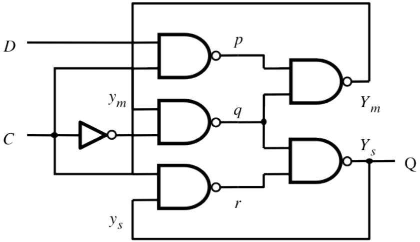
(c) Hazard-free circuit



(b) Karnaugh map

Removal of Static 1-Hazards

- Hazard exists whenever 2 adjacent 1s in a K-map are not covered by a single product.
- To remove all static hazards, find a cover that includes each pair of adjacent 1s.

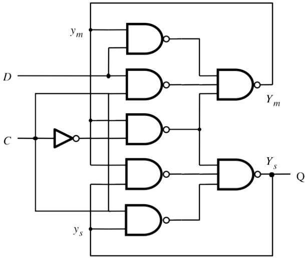


(a) Minimum-cost circuit

$y_m y_s$ \ CD	00	01	11	10
00			1	
01			1	
11	1	1	1	
10	1	1	1	

$y_m y_s$ \ CD	00	01	11	10
00				
01			1	1
11	1	1	1	1
10	1	1		

Hazard-free Master-slave Flip-flop



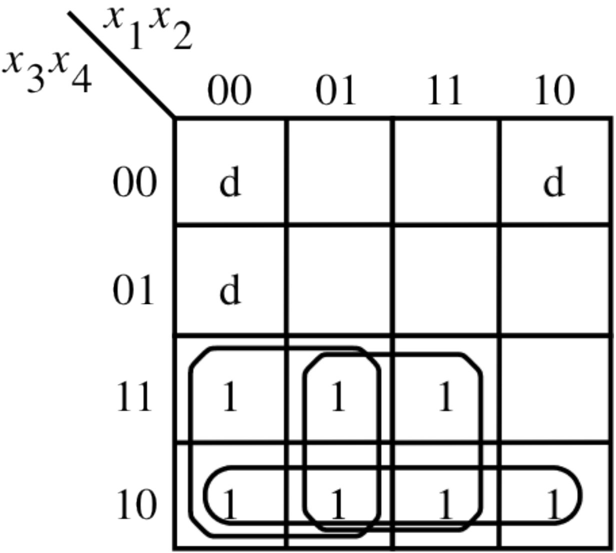
(c) Hazard-free circuit

$y_m y_s$ \ CD	00	01	11	10
00			1	
01			1	
11	1	1	1	
10	1	1	1	

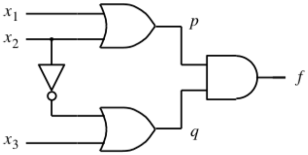
$y_m y_s$ \ CD	00	01	11	10
00				
01			1	1
11	1	1	1	1
10	1	1		

(b) Karnaugh maps for Y_m and Y_s in Figure 9.6*a*

Do Not Need to Include All Prime Implicants



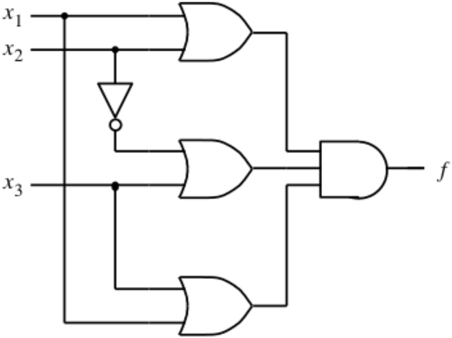
Static Hazard in POS Circuit



(a) Circuit with a hazard

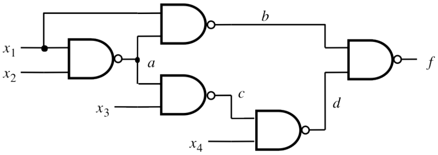
$x_3 \backslash x_1x_2$	00	01	11	10
0	0	0	0	1
1	0	1	1	1

(b) Karnaugh map

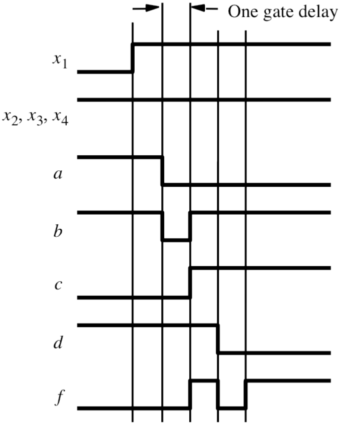


(c) Hazard-free circuit

Circuit with a Dynamic Hazard



(a) Circuit



(b) Timing diagram

Significance of Hazards

- A glitch in an asynchronous circuit can cause the circuit to enter an incorrect state and possibly become stable in that state.
- Next-state logic must be hazard-free.
- Synchronous circuits can have hazards as long as they are stable by the setup time of the flip-flops.

Concluding Remarks

- Analysis of asynchronous circuits.
- Synthesis of asynchronous circuits.
    - State reduction
    - State assignment
    - Hazard-free logic design
- More details on asynchronous design in ECE/CS 5750.