



## Chapter 5

---

# Synchronous Sequential Logic

5-1



## Outline

---

- Sequential Circuits
- Latches
- Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-2

## Sequential Circuits

- Consist of a combinational circuit to which storage elements are connected to form a feedback path
- State – the state of the memory devices now, also called **current state**
- Next states and outputs are functions of inputs and present states of storage elements

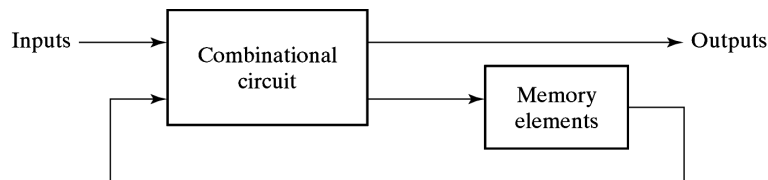
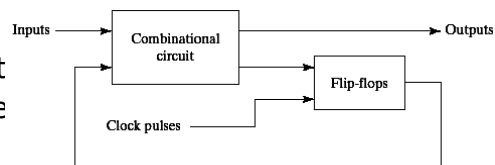


Fig. 5-1 Block Diagram of Sequential Circuit

5-3

## Two Types of Sequential Circuits

- Asynchronous sequential circuit
  - Depends upon the input signals at **any instant** of time and their change order
  - May have better performance but hard to design
- Synchronous sequential circuit
  - Defined from the knowledge of its signals at **discrete instants** of time
  - Much easier to design (preferred design style)
  - Synchronized by a periodic train of clock pulses



(a) Block diagram



(b) Timing diagram of clock pulses

5-4

## Memory Elements

- Allow sequential logic design
- Latch — a level-sensitive memory element
  - SR latches
  - D latches
- Flip-Flop — an edge-triggered memory element
  - Master-slave flip-flop
  - Edge-triggered flip-flop
- RAM and ROM — a mass memory element
  - Discussed in Chapter 7

5-5

## Outline

- Sequential Circuits
- Latches
- Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-6

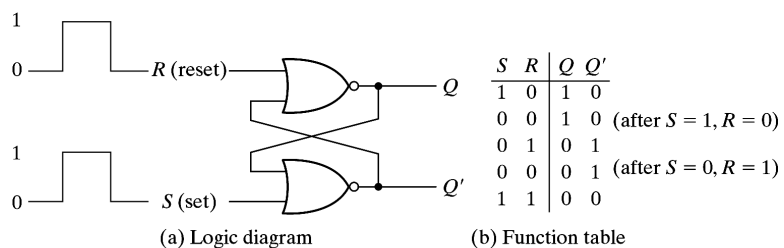
## Latches

- The most basic types of flip-flops operate with signal levels
- The basic circuits from which all flip-flops are constructed
- Useful for storing binary information and for the design of asynchronous sequential circuits
  - Not practical for use in synchronous sequential circuits
  - Avoid to use latches as possible in synchronous sequential circuits to avoid design problems

5-7

## SR Latch

- A circuit with two cross-coupled NOR gates or two cross-coupled NAND gates
- Two useful states:
  - $S=1, R=0 \rightarrow$  set state ( $Q$  will become to 1)
  - $S=0, R=1 \rightarrow$  reset state ( $Q$  will become to 0)
- When  $S=0$  and  $R=0 \rightarrow$  keep the current value



(a) Logic diagram

(b) Function table

5-8

# Undefined State in SR Latch

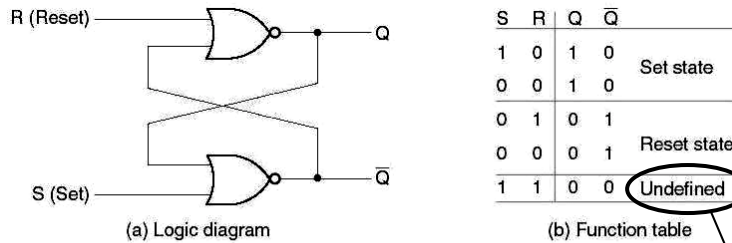


Fig. 4-4 SR Latch with NOR Gates

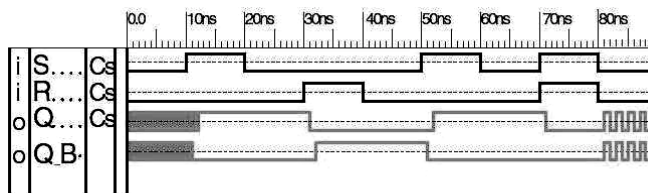


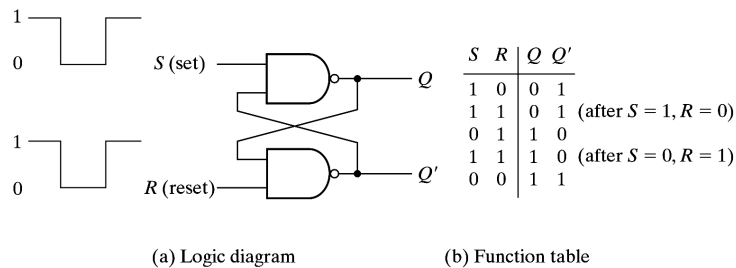
Fig. 4-5 Logic Simulation of SR Latch Behavior

Should be very careful for this case

5-9

# SR Latch with NAND Gates

- The SR latches constructed with two cross-coupled NAND gates are **active-low**
  - $S=1, R=0 \rightarrow$  reset state (Q will become to 0)
  - $S=0, R=1 \rightarrow$  set state (Q will become to 1)
  - $S=1, R=1 \rightarrow$  unchanged



(a) Logic diagram

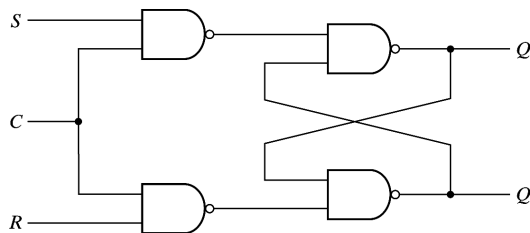
(b) Function table

Fig. 5-4 SR Latch with NAND Gates

5-10

## SR Latch with Control Input

- Add an additional control input to determine when the state of the latch can be changed
- $C=0$ : S and R are disabled (no change at outputs)
- $C=1$ : S and R are active-high



$C$	$S$	$R$	Next state of $Q$
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$ ; Reset state
1	1	0	$Q = 1$ ; set state
1	1	1	Indeterminate

(a) Logic diagram

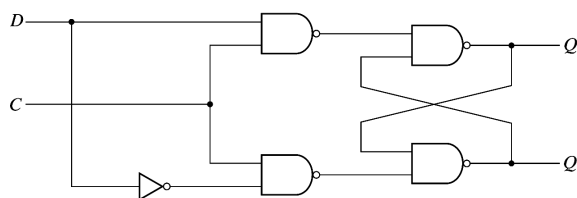
(b) Function table

Fig. 5-5 SR Latch with Control Input

5-11

## D Latch

- D latch has only two inputs: D(data) and C(control)
  - Use the value of D to set the output value
  - Eliminate the indeterminate state in the SR latches
- The D input goes directly to the S input and its complement is applied to the R input
  - $D=1 \rightarrow Q=1 \rightarrow S=1, R=0$



$C$	$D$	Next state of $Q$
0	X	No change
1	0	$Q = 0$ ; Reset state
1	1	$Q = 1$ ; Set state

(a) Logic diagram

(b) Function table

Fig. 5-6 D Latch

5-12

## Graphic Symbols for Latches

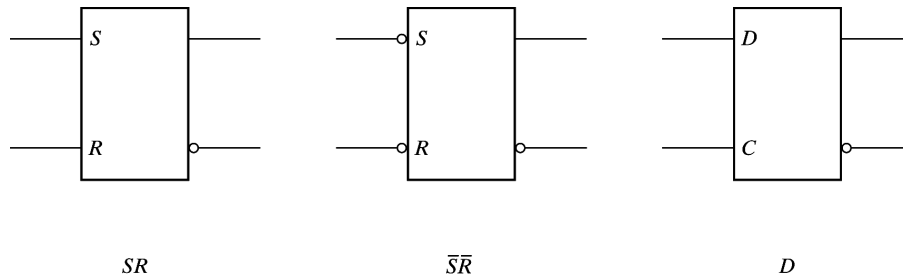


Fig. 5-7 Graphic Symbols for Latches

5-13

## Outline

- Sequential Circuits
- Latches
- **Flip-Flops**
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-14

# Flip-Flops

- The state of a latch or flip-flop is switched by a change in the control input
- This momentary change is called a **trigger**
- Latch: level-sensitive
- Flip-Flop: edge-triggered

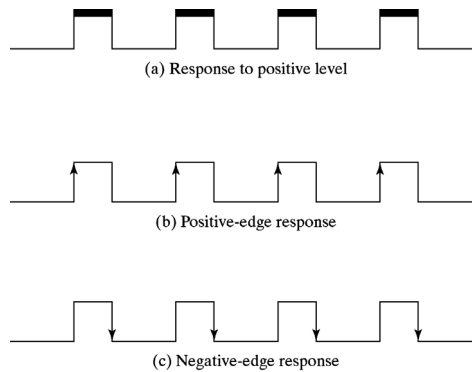
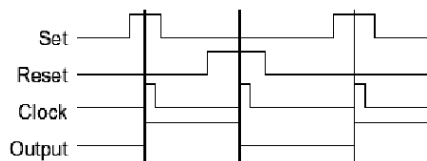


Fig. 5-8 Clock Response in Latch and Flip-Flop

5-15

# Latch vs. Flip-Flop

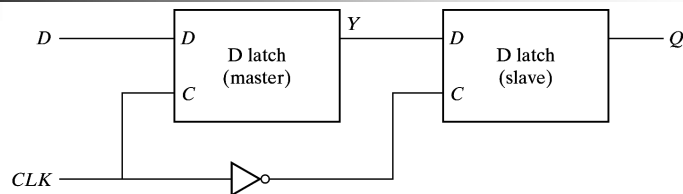
- Latch:
  - Change stored value under specific status of the control signals
  - Transparent for input signals when control signal is "on"
  - May cause combinational feedback loop and extra changes at the output
- Flip-Flop:
  - Can only change stored value by a momentary switch in value of the control signals
  - Cannot "see" the change of its output in the same clock pulse
  - Encounter fewer problems than using latches



5-16



## Master-Slave D Flip-Flop



- Constructed with two D latches and an inverter
- The first latch (master) is enabled when  $CLK=1$ 
  - It reads the input changes but stops before the second one
- The second latch (slave) is enabled when  $CLK=0$ 
  - Close the first latch to isolate the input changes
  - Deliver the final value at the moment just before  $CLK$  changes
- The circuit samples the D input and changes its output Q only at the **negative-edge** of the controlling clock

5-17

## Edge-Triggered D Flip-Flop

- If only SR latches are available, three latches are required
- Two latches are used for locking the two inputs (CLK & D)
- The final latch provides the output of the flip-flop

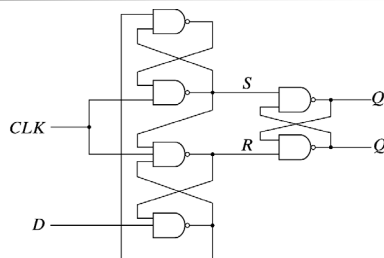
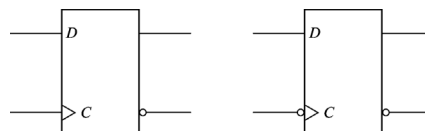


Fig. 5-10 D-Type Positive-Edge-Triggered Flip-Flop



(a) Positive-edge

(a) Negative-edge

Fig. 5-11 Graphic Symbol for Edge-Triggered D Flip-Flop 5-18

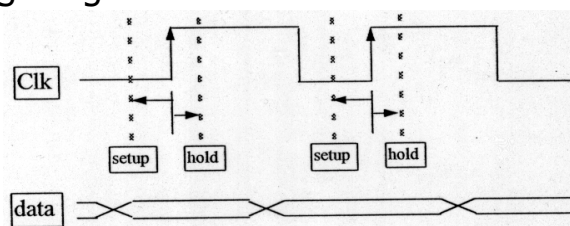
## Setup & Hold Times

- The response time of a flip-flop to input changes must be taken into consideration
- Setup Time: The length of time that data must stabilize before the clock transition
  - The maximum data path is used to determine if the setup time is met
- Hold Time: The length of time that data must remain stable at the input pin after the active clock transition
  - The minimum data path is used to determine if hold time is met

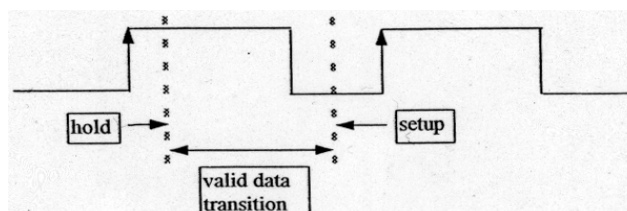
5-19

## Setup & Hold Times

- Timing Diagram



- Valid Data Transition



5-20

## Other Flip-Flops

- The most economical and efficient flip-flop is the edge-triggered D flip-flop
  - It requires the smallest number of gates
- Other types of flip-flops can be constructed by using the D flip-flop and external logic
  - JK flip-flop
  - T flip-flops
- Three major operations that can be performed with a flip-flop:
  - Set it to 1
  - Reset it to 0
  - Complement its output

5-21

## Edge-Triggered JK Flip-Flop

$$D = JQ' + K'Q$$

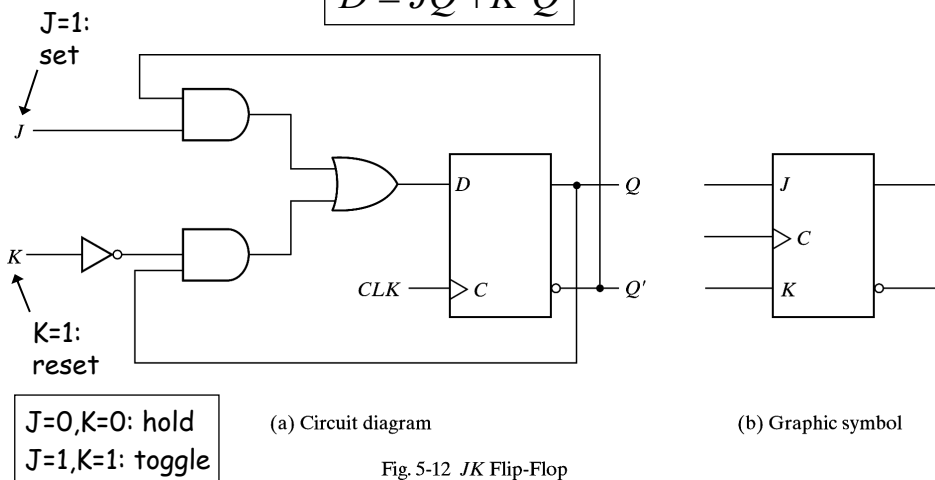
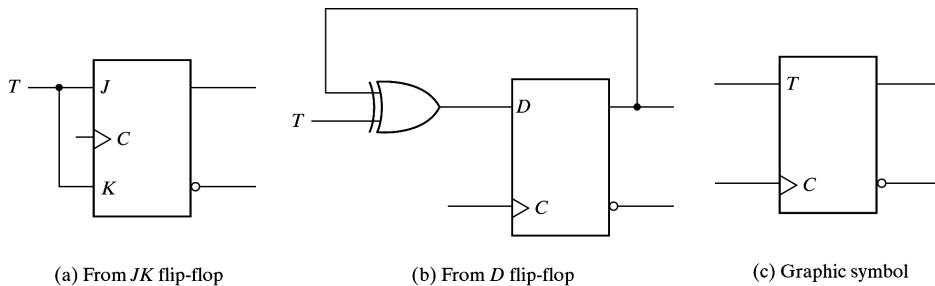


Fig. 5-12 JK Flip-Flop

5-22

## Edge-Triggered T Flip-Flop

$$D = T \oplus Q = TQ' + T'Q$$



(a) From JK flip-flop

(b) From D flip-flop

(c) Graphic symbol

T=0: hold  
T=1: toggle

Fig. 5-13 T Flip-Flop

5-23

## Characteristic Tables

- Define the logical properties in tabular form

JK flip-flop

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

D Flip-Flop

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

T Flip-Flop

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

5-24

## Characteristic Equations

- Algebraically describe the next state
- Can be derived from characteristic tables
- D flip-flop:

$$Q(t+1) = D$$

- JK flip-flop:

$$Q(t+1) = JQ' + K'Q$$

- T flip-flop:

$$Q(t+1) = T \oplus Q = TQ' + T'Q$$

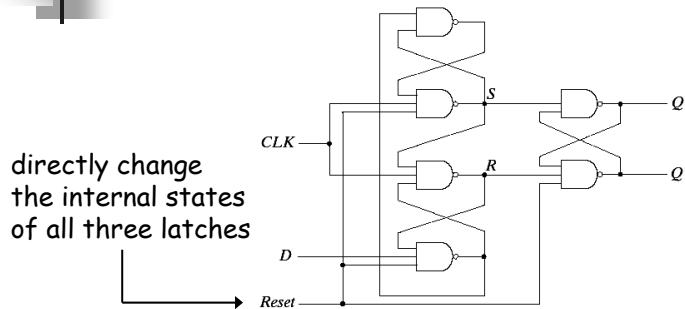
5-25

## Direct Inputs

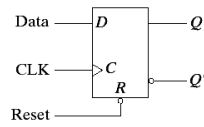
- Force the flip-flop to a particular state immediately
  - Independent of clock signal
  - Have higher priority than any other inputs
  - Useful to bring all flip-flops from unknown into known state while power up
- The input that sets the flip-flop to 1 is called **preset** or **direct set**
- The input that clears the flip-flop to 0 is called **clear** or **direct reset**
- Also called **asynchronous** set/reset

5-26

## D F/F with Asynchronous Reset



(a) Circuit diagram



(b) Graphic symbol

$R$	$C$	$D$	$Q$	$Q'$
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

(b) Function table

→ other inputs have no effects

5-27

## Outline

- Sequential Circuits
- Latches
- Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-28

## Sequential Circuit Analysis

- The behavior of a clocked sequential circuit is determined from
  - The inputs
  - The outputs
  - The state of its flip-flops
- The outputs and the next state are both a function of the inputs and the present state
- To analyze a sequential circuit, we can use
  - State equations
  - State table
  - State diagram
  - Flip-Flop input equations

5-29

## State Equations

- Specify the next state as a function of the present state and inputs
  - Also called transition equation
- Analyze the combinational part directly
- EX:

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

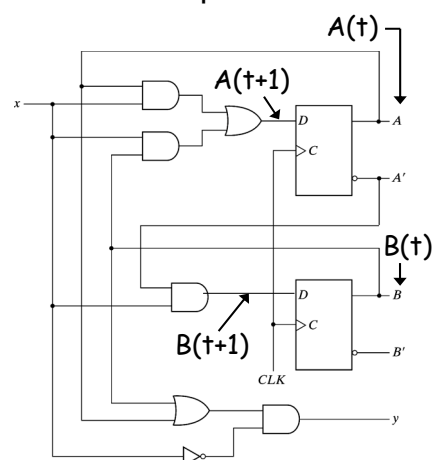
$$\Rightarrow A(t+1) = Ax + Bx$$

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

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

$$y(t) = [A(t) + B(t)] x(t)$$

$$\Rightarrow y = (A+B)x'$$



5-30

## State Table

- Enumerate the time sequence of inputs, outputs, and flip-flop states
  - Also called transition table
  - Similar to list the truth table of state equations
- Consist of four sections
  - Present state, input, next state, and output
- A sequential circuit with  $m$  flip-flops and  $n$  inputs need  $2^{m+n}$  rows in the state table

Present state		input	Next state		output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

5-31

## Second Form of State Table

- The state table has only three section: present state, next state, and output
- The input conditions are enumerated under next state and output sections

Present State		Next State				Output	
		X=0		X=1		X=0	X=1
A	B	A	B	A	B	Y	Y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

5-32



## State Diagram

- Graphically represent the information in a state table
  - Circle: a state (with its state value inside)
  - Directed lines: state transitions (with inputs/outputs above)
- Ex: starting from state 00
  - If the input is 0, it stays at state 00 with output=0
  - If the input is 1, it goes to state 01 with output=0
- The state table is easier to derive from a given logic diagram and state equations
- The state diagram is suitable for human interpretation

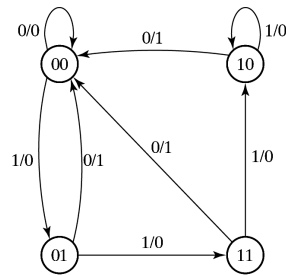


Fig. 5-16 State Diagram of the Circuit of Fig. 5-15

5-33

## Flip-Flop Input Equations

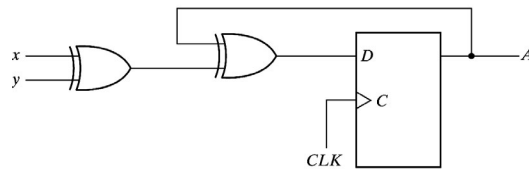
- To draw the logic diagram of a sequential circuit, we need
  - The type of flip-flops
  - A list of Boolean expressions of the combinational circuits
- The Boolean functions for the circuit that generates external outputs is called output equations
- The Boolean functions for the circuit that generates the inputs to flip-flops is flip-flop input equations
  - Sometimes called excitation equations
- The flip-flop input equations provide a convenient form for specifying the logic diagram of a sequential circuit
- Ex: (Fig. 5-15)
 

Input:	Output:
$D_A = Ax + Bx$	$y = (A + B)x'$
$D_B = A'x$	

5-34

## Analysis with D Flip-Flop

- Input equation:  $D_A = A \oplus x \oplus y$



(a) Circuit diagram

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

(b) State table



(c) State diagram

Fig. 5-17 Sequential Circuit with D Flip-Flop

5-35

## Analysis with Other Flip-Flops

- The sequential circuit using other flip-flops such as JK or T type can be analyzed as follows
  - Determine the flip-flop input equations in terms of the present state and input variables
  - List the binary values of each input equation
  - Use the corresponding flip-flop characteristic table to determine the next state values in the state table

5-36

## Analysis with JK Flip-Flops (1/2)

### Step 1: input equations

$$J_A = B \quad K_A = Bx' \quad J_B = x' \quad K_B = A \oplus x'$$

### Step 2: state equations

$$\begin{aligned} A(t+1) &= JA' + K'A \\ &= BA' + (Bx')'A \\ &= A'B + AB' + Ax \end{aligned}$$

$$\begin{aligned} B(t+1) &= JB' + K'B \\ &= x'B' + (A \oplus x)'B \\ &= B'x' + ABx + A'Bx' \end{aligned}$$

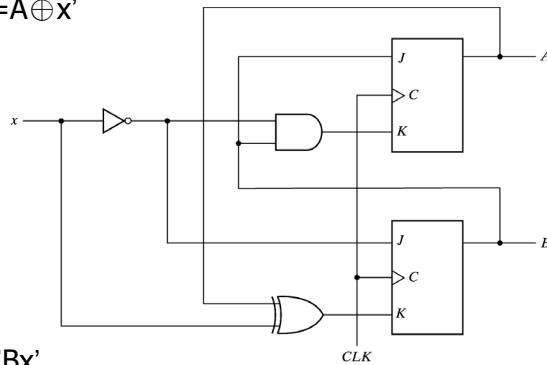


Fig. 5-18 Sequential Circuit with JK Flip-Flop

5-37

## Analysis with JK Flip-Flops (2/2)

### Step 3: state table

Present state		Input	Next state		Flip-Flop Inputs			
A	B	X	A	B	$J_A$	$K_A$	$J_B$	$K_B$
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

### Step 4: state diagram

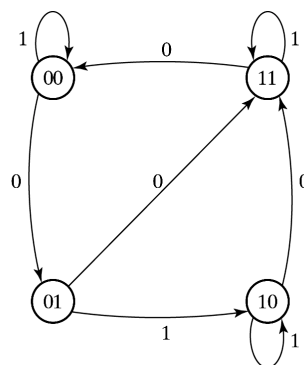
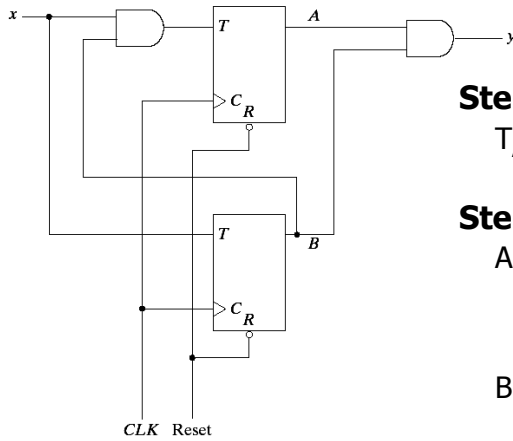


Fig. 5-19 State Diagram of the Circuit of Fig. 5-18

5-38

## Analysis with T Flip-Flops (1/2)



(a) Circuit diagram

**Step 1:** input equations

$$T_A = Bx \quad T_B = x \quad y = AB$$

**Step 2:** state equations

$$\begin{aligned} A(t+1) &= T'A + TA' \\ &= (Bx)'A + (Bx)A' \\ &= AB' + Ax' + A'Bx \end{aligned}$$

$$\begin{aligned} B(t+1) &= T'B + TB' \\ &= x'B + xB' \\ &= x \oplus B \end{aligned}$$

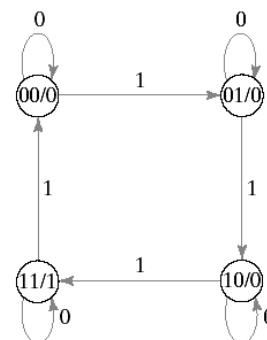
5-39

## Analysis with T Flip-Flops (2/2)

**Step 3:** state table

Present state		Input	Next state		Output
A	B	X	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

**Step 4:** state diagram

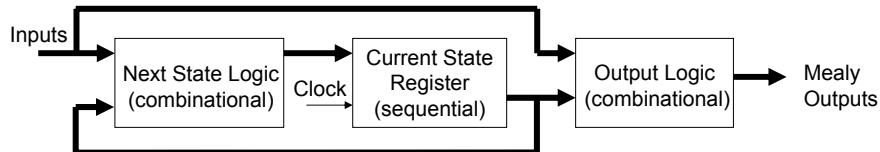


(b) State diagram

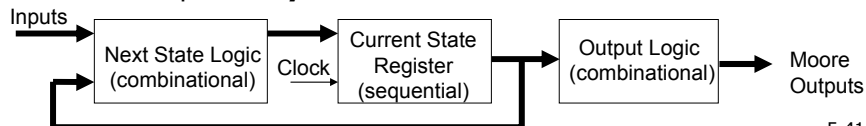
5-40

## Mealy and Moore Model

- Mealy model :
  - The output is a function of **both** the present state and input
  - The output may change if the inputs change during a clock cycle



- Moore model :
  - The output is a function of the **present state only**
  - The output are **synchronized** with the clock



5-41

## Outline

- Sequential Circuits
- Latches
- Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-42

## State Reduction

- Reducing the number of states in a state table, while keeping the external input-output requirements unchanged
- Example:
  - Total 7 states
  - A sequence as follows

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

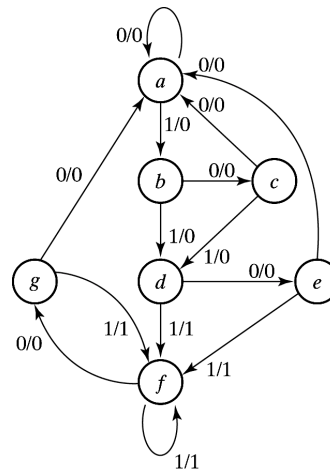


Fig. 5-22 State Diagram

5-43

## State Reduction Rules

- Two states are said to be equivalent if, for every possible inputs, they give exactly the same output and have equivalent next state

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

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

delete state g and replaced with state e

5-44

## Further State Reduction

- After the first reduction, we can see that state d and state f will have the same output and next state for both  $x=0$  and  $x=1$ 
  - Further reduce one state

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

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

delete state f and replaced with state d

5-45

## Reduced State Diagram

- After reduction, the circuit has only 5 states with same input/output requirements
- Original output sequence:

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

- Reduced output sequence:

state	a	a	b	c	d	e	d	d	e	d	e	a
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

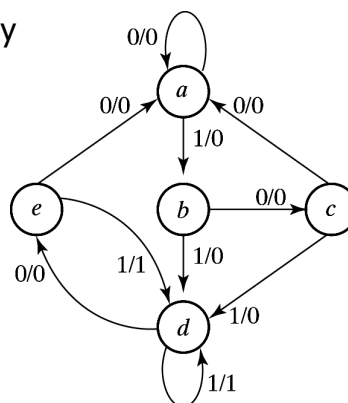


Fig. 5-23 Reduced State Diagram

5-46

## Implication Chart Method (1/3)

- Step 1: build the implication chart

Present State	Next State		Present Output
	X = 0	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

\*For details, see "Fundamentals of Logic Design", 4th Ed., by C. H. Roth, Jr.

5-47

## Implication Chart Method (2/3)

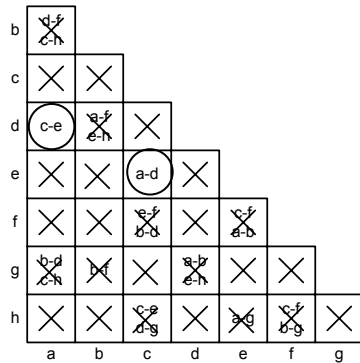
- Step 2: delete the node with unsatisfied conditions

5-48



## Implication Chart Method (3/3)

- Step 3: repeat Step 2 until equivalent states found



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

5-49

## State Assignment

- Assign **coded binary values** to the states for physical implementation
- For a circuit with  $m$  states, the codes must contain  $n$  bits where  $2^n \geq m$
- Unused states are treated as don't care conditions during the design
  - Don't cares can help to obtain a simpler circuit
- There are many possible state assignments
  - Have large impacts on the final circuit size

Assignment:	
a = 000	d = 011
b = 001	e = 100
c = 010	

Present State	Next State		Output	
	X=0	X=1	X=0	X=1
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

5-50

## Popular State Assignments

- Binary: assign the states in binary order
  - Typical method without other considerations
- Gray code: assign the states by gray code
  - Lower power consumption during state transitions (if in order)
- One-hot: assign a specific flip-flop for each state
  - Simplify the circuit design but may have larger hardware cost

State	Assignment 1 Binary	Assignment 2 Gray code	Assignment 3 One-hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

5-51

## Outline

- Sequential Circuits
- Latches
- Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure

5-52

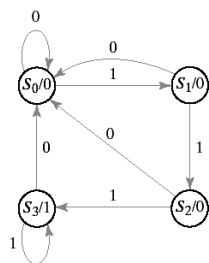
## Design Procedure

- Design procedure of synchronous sequential circuits:
  - Derive a state diagram for the circuit from specifications
  - Reduce the number of states if necessary
  - Assign binary values to the states
  - Obtain the binary-coded state table
  - Choose the type of flip-flop to be used
  - Derive the simplified flip-flop input equations and output equations
  - Draw the logic diagram
- Step 4 to 7 can be automated
  - Use HDL synthesis tools

5-53

## Synthesis Using D Flip-Flops

- Ex: design a circuit that detects 3 or more consecutive 1's at inputs



$$A(t+1) = D_A(A, B, x) = \sum(3, 5, 7)$$

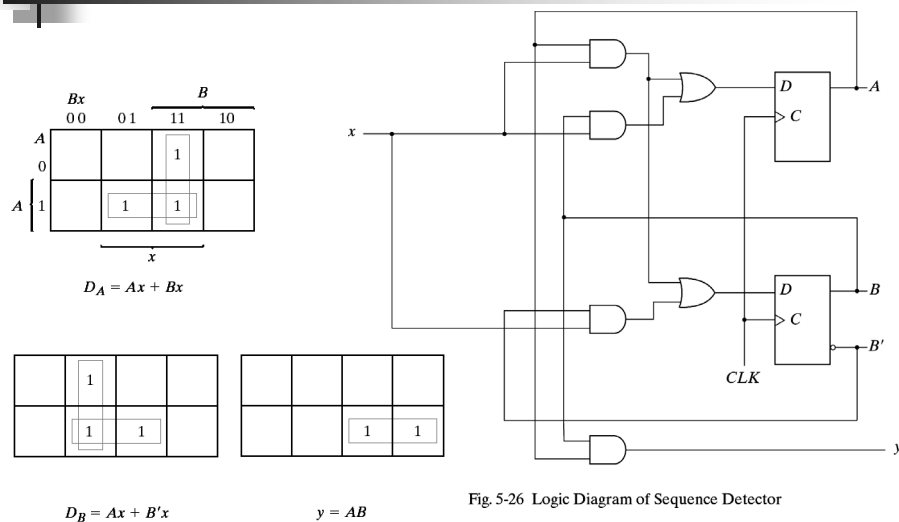
$$B(t+1) = D_B(A, B, x) = \sum(1, 5, 7)$$

$$y(A, B, x) = \sum(6, 7)$$

Present state		Input		Next state		Output
A	B	X	A	B	y	
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	0	0	0
1	0	1	1	1	0	0
1	1	0	0	0	1	1
1	1	1	1	1	1	1

5-54

## Synthesis Using D Flip-Flops



5-55

## Excitation Tables

- Record the flip-flop input conditions that will cause the required transition in STG
  - Equal to next state equations for D flip-flop
- For JK flip-flop:
  - J=0, K=X: no change (JK=00) or set to zero (JK=01)
  - J=1, K=X: toggle (JK=11) or set to one (JK=10)
  - J=X, K=1: toggle (JK=11) or set to zero (JK=01)
  - J=X, K=0: no change (JK=00) or set to one (JK=10)

JK F/F	JK Flip-Flop				T Flip-Flop			T F/F
	Q(t)	Q(t+1)	J	K	Q(t)	Q(t+1)	T	
	0	0	0	X	0	0	0	
	0	1	1	X	0	1	1	
	1	0	X	1	1	0	1	
	1	1	X	0	1	1	0	

5-56

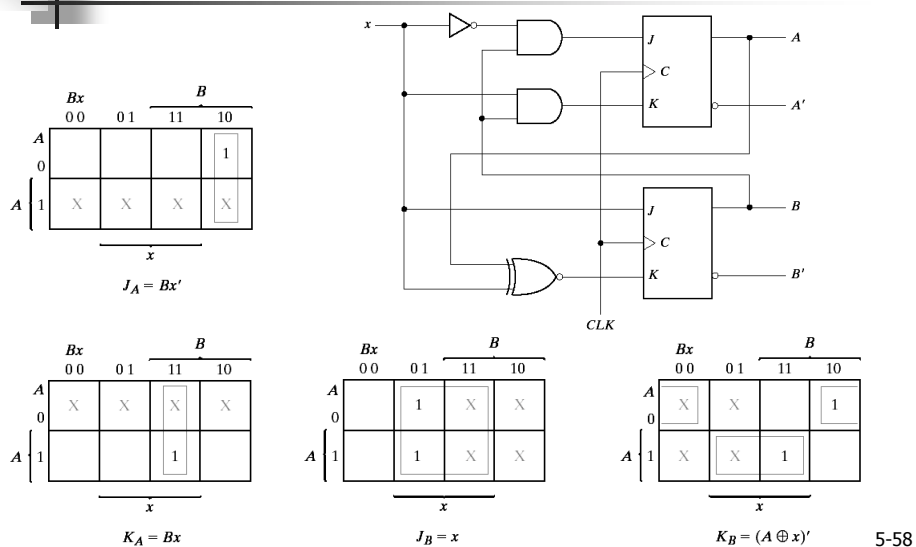
## Synthesis Using JK Flip-Flops

- Derive the state table with the excitation inputs
- Other design procedures are the same

Present State		Input X	Next State		Flip-Flop Inputs			
A	B		A	B	J <sub>A</sub>	K <sub>A</sub>	J <sub>B</sub>	K <sub>B</sub>
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

5-57

## Synthesis Using JK Flip-Flops

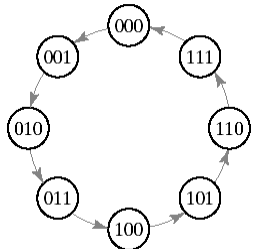


5-58

## Synthesis Using T Flip-Flops

- Derive the state table with the excitation inputs
- Other design procedures are the same

3-bit binary counter



Present State			Next State			Flip-Flop Inputs		
A2	A1	A0	A2	A1	A0	T <sub>A2</sub>	T <sub>A1</sub>	T <sub>A0</sub>
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

5-59

## Synthesis Using T Flip-Flops

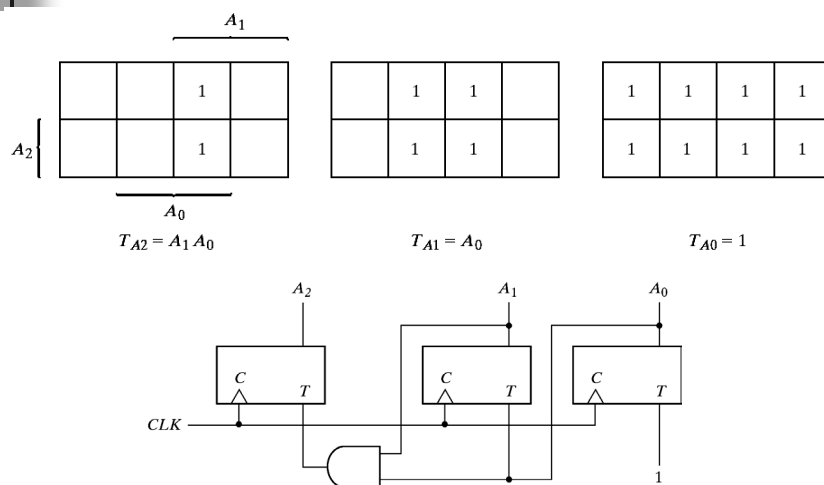


Fig. 5-31 Logic Diagram of 3-Bit Binary Counter

5-60