Synchronous Sequential Logic

2019. 4. 29.

K-S. Sohn

Contents

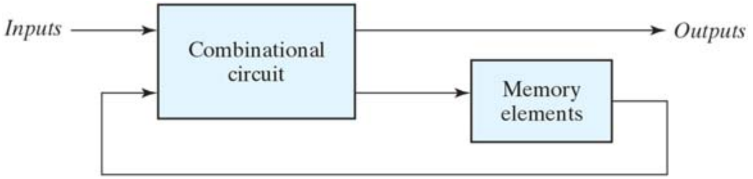
- Sequential circuits
- Latches
- Flip-flops
- Analysis of clocked secuential circuits
- State reduction and assignment
- Design of secuential circuits

Introduction

- Combinational circuits
  - neither memory element nor feedback
  - <output> = F(<input>)

Sequential circuits

- Block diagram of secuential circuit



  - feedback path
  - memory element

Sequential Circuits

- State
  - the binary information stored in the memory element at any given time defines the state of the sequential circuit at that time.
  - <output> = F(<state>, <input>: <control signal>)
- Syncrounous
  - the transition happens at discrete instants of time
- Asnychronous
  - the transition happens at any instant of time

Sequential Circuits

- Asynchronous sequential circuits
  - having only feedback path (or with the "pseudo-memory")
  - <output> = F(<state>, <input>)
  - difficult to avoid instability in design

Synchronous Sequential Circuits

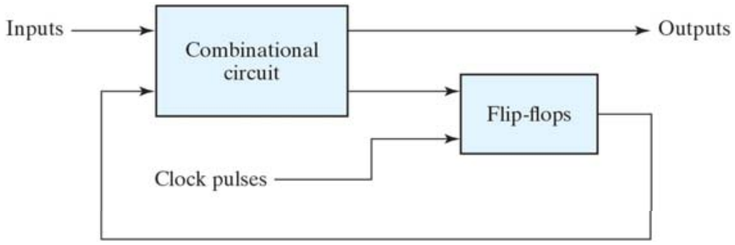
- Clock generator
  - a master-clock generator to generate a periodic train of clock pulses
  - the clock pulses are distributed throughout the system
- clocked sequential circuits
  - most commonly used
  - no instability problems
- clock speed
  - limited by the operational speed of the combinational circuit

Synchronous Sequential Circuits

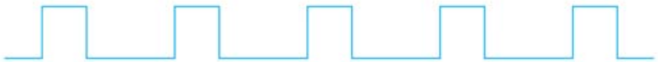
- the memory elements: flip-flops
  - binary cells capable of storing one bit of information
  - two outputs: one for the normal value and one for the
  - complement value
  - maintain a binary state indefinitely until directed by an input signal to switch states

Synchronous Sequential Circuits

- Synchronous clocked sequential circuit



(a) Block diagram



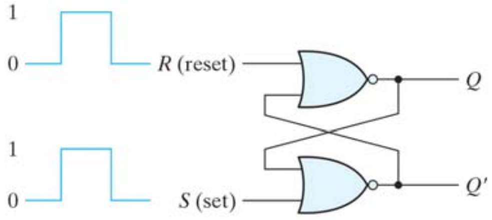
(b) Timing diagram of clock pulses

Latches

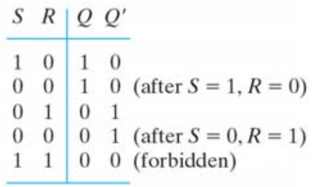
- asynchronous sequential circuits
  - binary state (0 or 1)
  - Q(t+1) = F(Q(t), <input>)
- Level sensitive circuits
  - the next state is determined by the level of inputs and the current state
  - can not be used for synchronous sequential circuits,
  - but, building blocks of flip-flop

SR Latch

- Tow NOR gates with cross-coupled connection



(a) Logic diagram



S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	(forbidden)

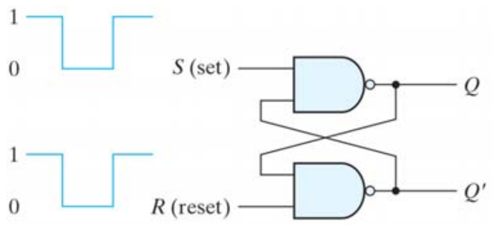
(b) Function table

SR Latch

- Most fundamental building block
  - more complicated types can be built upon this SR latch
- Functional characteristics
  - (S,R)= (0,0): no operation (usual position of the level of inputs)
  - (S,R)=(0,1): reset (Q=0, the clear state)
  - (S,R)=(1,0): set (Q=1, the set state)
  - (S,R)=(1,1): indeterminate state (Q=Q'=0, out of SR rules)

SR Latch

- NAND implementation
  - complement of the NOR version
  - S'R' latch



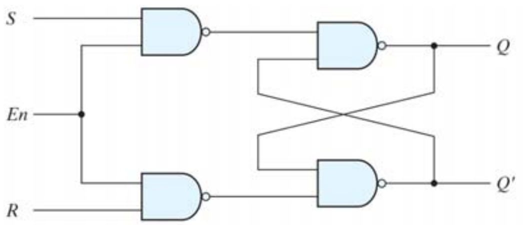
(a) Logic diagram

S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

(b) Function table

SR Latch

- SR latch with control input
  - En = 0, no change
  - En = 1, enable input



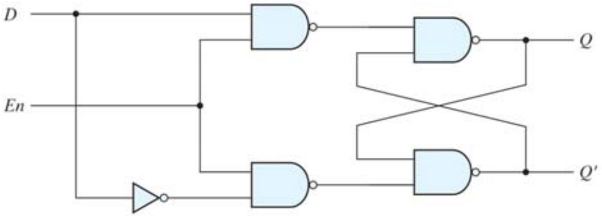
(a) Logic diagram

En	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

(b) Function table

D Latch

- Design objectives
  - eliminate the undesirable conditions of the indeterminate state in the SR latch by using an inverter and tied inputs
- 'D' comes from:
  - Q=<input **d**ata>, transparent latch



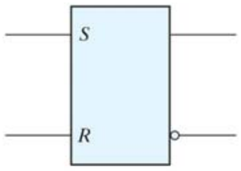
(a) Logic diagram

En	D	Next state of Q
0	X	No change
1	0	$Q = 0$; reset state
1	1	$Q = 1$; set state

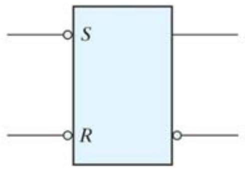
(b) Function table

Graphic Symbols

- SR latches

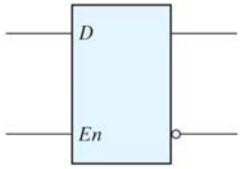


SR



$\bar{S}\bar{R}$

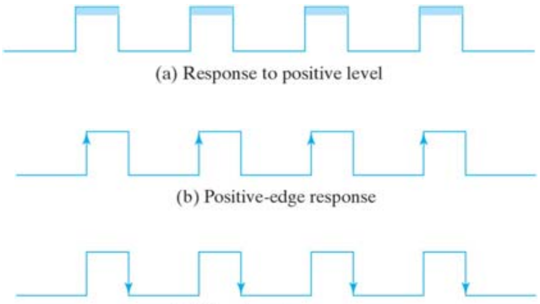
- D latch



D

Flip-Flops

- Trigger
  - the momentary change of control inputs
  - the state of a latch or flip-flop is switched by a change of the control input
- Level triggerd: latch
- Edge triggered: F-F



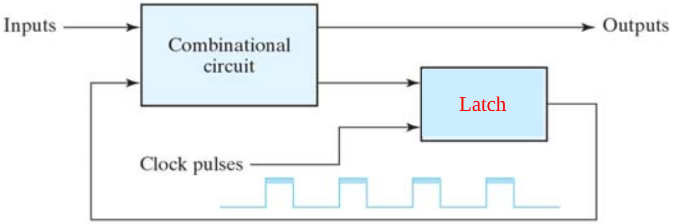
(a) Response to positive level

(b) Positive-edge response

(c) Negative-edge response

Flip-Flop

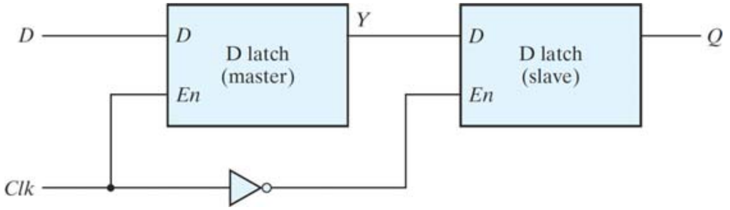
- Why F-F as the memory element in the S.S.L.?
  - If a latch used:



  - the feedback path may cause instability problem
- Edge-triggered F-Fs
  - the state transition happens only at the edge
  - eliminate the multiple-transition problem

Edge-triggered D Flip-Flop

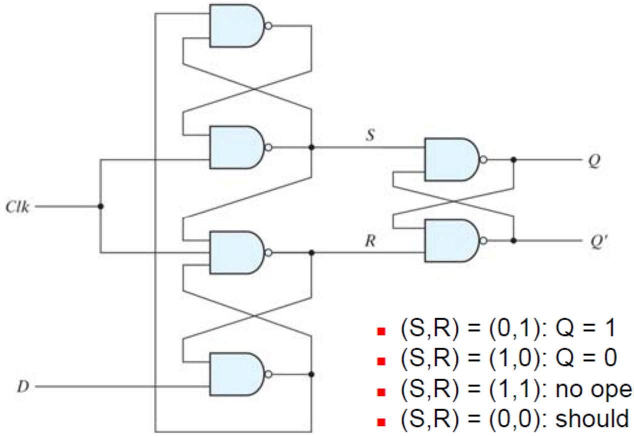
- Master-slave D flip-flop
  - two separate D latches and an inverter
  - master latch (triggered during clk's positive level)
  - slave latch(triggered during clk's negative level)



  - triggered at negative or positive edge?
  - how to reverse the direction of triggering edge?

Edge-triggered D Flip-Flop

- D type positive-edge triggered F-F



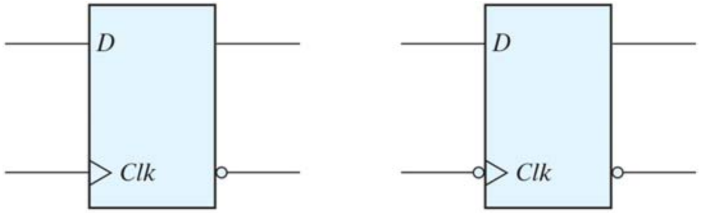
- (S,R) = (0,1): Q = 1
- (S,R) = (1,0): Q = 0
- (S,R) = (1,1): no operation
- (S,R) = (0,0): should be avoided

Edge-triggered D Flip-Flop

- In sum.
  - CP=0: (S,R) = (1,1), no state change
  - CP=↑: state change once
  - CP=1: state holds
  - eliminate the feedback problems in sequential circuits

Graphic Sumbols of D F-F

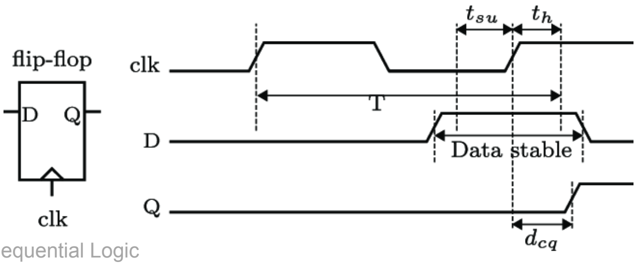
- The edge-triggered D F-F
  - the most economical and efficient
  - positive-edge and negative-edge



(a) Positive-edge (b) Negative-edge

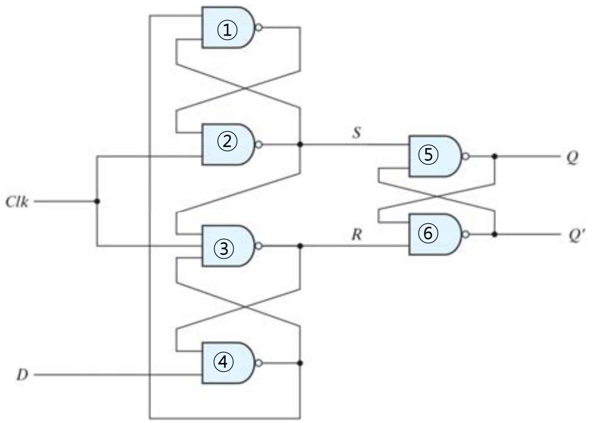
Specifications of Flip-Flop

- Setup time
  - A minimum time for which the D input must be maintained at a constant value prior to the occurrence of the clock transition
- Hold time
  - A minimum time for which the D input must not change after the application of the positive transition of the clock



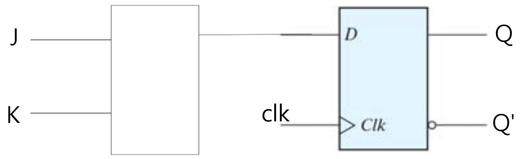
Specification of Flip-Flop

- Time contribution analysis
  - setup time > td1 + td4
  - hold time > td3



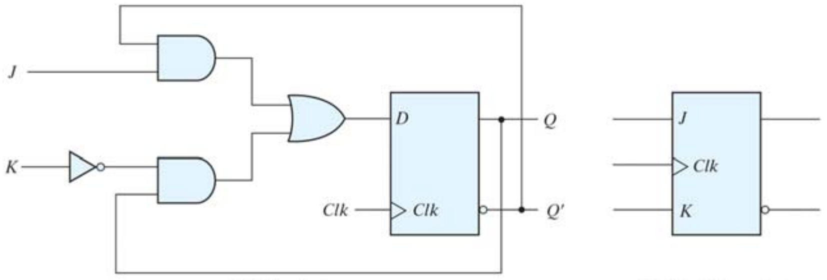
JK Flip-Flop

- Operational requirements
  - Set: J=1, K=0, Q(t+1)=1
  - Reset: J=0, K=1, Q(t+1)=0
  - Complement: J=1, K=1, Q(t+1)=Q'(t)
  - No change: J = K = 0, Q(t+1)=Q(t)
- Constraints
  - Use D F-F and some logic gates including inverters



JK Flip-Flop

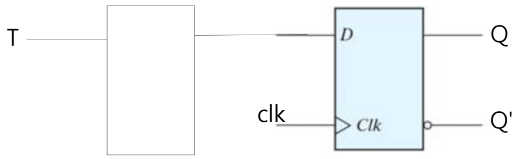
- Front logic
  - D = JQ' + K'Q
  - How?
- Logic diagram and graphical symbol



(a) Circuit diagram (b) Graphic symbol

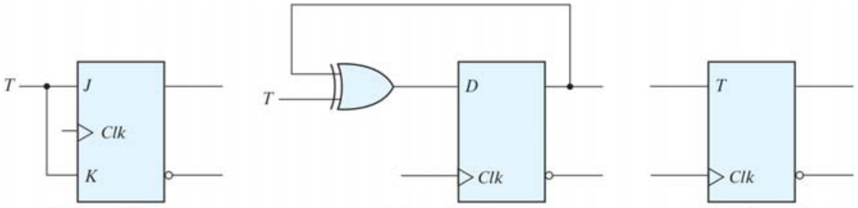
T Flip-Flop

- Functional requirements
  - toggle output with a T input
  - T=0, Q(t+1)=Q(t)
  - T=1, Q(t+1)=Q'(t)
- Constraints
  - Use D F-F and some logic gates



T Flip-Flop

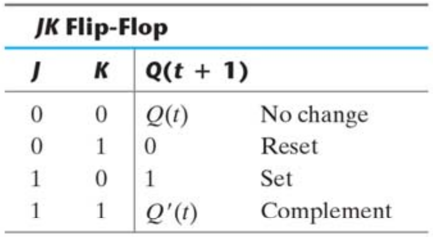
- Front logic
  - D=TQ'+T'Q
  - Why?
- Logic diagram and graphic symbol



(a) From JK flip-flop (b) From D flip-flop (c) Graphic symbol

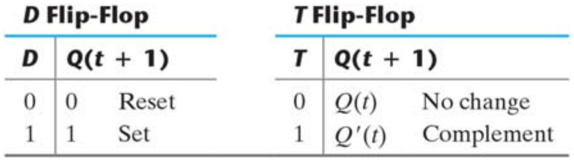
Characteristics Tables

- t
  - a period of the clock pulse
  - (t+1) means the next clock pulse period
- Q(t)
  - present state
- Q(t+1)
  - next state



***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**

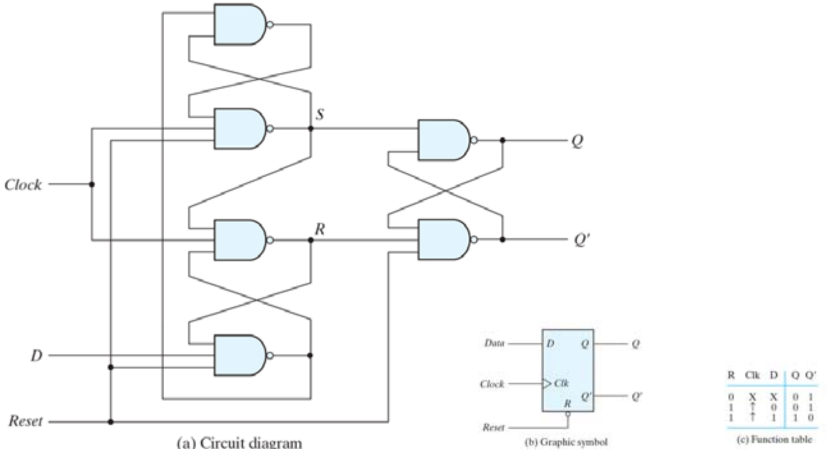
T	Q(*t* + 1)	
0	*Q*(*t*)	No change
1	*Q*′(*t*)	Complement

Characteristic Equations

- D flip-flop
  - Q(t+1)=D
- JK flip-flop
  - Q(t+1)=JQ'+K'Q
- T flip-flop
  - Q(t+1)=TQ'+T'Q

Direct Inputs

- Ansynchronous set and/or reset



(a) Circuit diagram

(b) Graphic symbol

R	Clk	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

(c) Function table

Analysis of Clocked sequential Circuits(SSC)

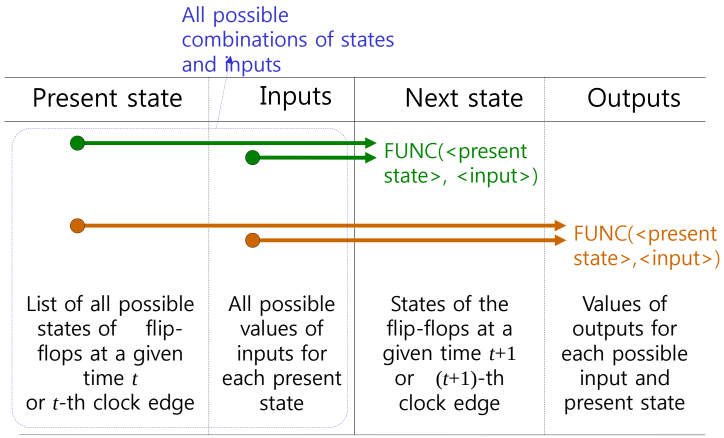
- Analysis of clocked sequential circuit(SSC)
  - the behavior of a clocked sequential circuit is determined from the inputs, the outputs, and the state of its flip-flops
  - SSC(synchronized sequential circuits)
- Clocked sequential circuit
  - <output> = FUNC(<inputs>, <curr state>)
  - <next state> = FUNC(<inputs>, <curr state>)
- Results of analysis
  - State equations
  - State table
  - State diagram

State Equations(Transition Equation)

- Definition
  - Algebraic expression of the condition of state transition ⇒ Transition equation
- Objectives -- Specifying the next state of the SSC by:
  - Specifying the output of flip-flop
  - Deriving the output of SSC ⇒ Output equation
- Specifying the output of flip-flop
  - <F-F output>(t+1) = FUNC(<F-F output>(t), <input>(t))
  - (t+1) : one clock edge later than time (t)
- Output equation
  - <SSC output>(t) = FUNC(<F-F output>(t), <input>(t))

State Table(Transition Table)

- Structure of the state table

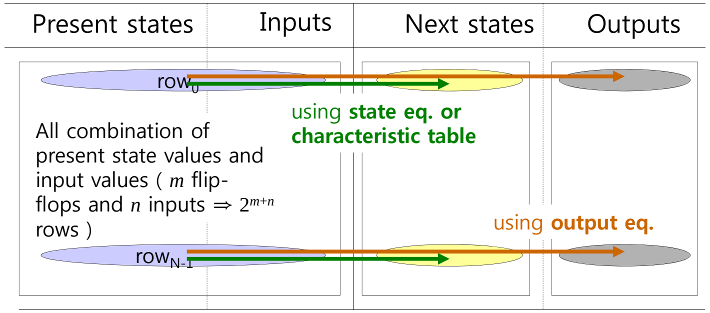


All possible combinations of states and inputs

Present state	Inputs	Next state	Outputs
		FUNC(<present state>, <input>)	
			FUNC(<present state>,<input>)
List of all possible states of flip-flops at a given time t or t-th clock edge	All possible values of inputs for each present state	States of the flip-flops at a given time t+1 or (t+1)-th clock edge	Values of outputs for each possible input and present state

State Table

- Derivation of a state table



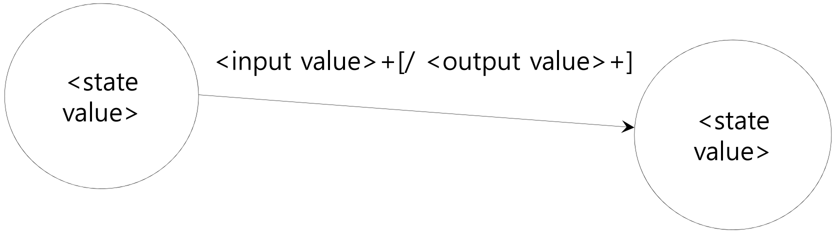
Present states	Inputs	Next states	Outputs
row$_0$			
All combination of present state values and input values (m flip-flops and n inputs ⇒ 2^{m+n} rows)			
row$_{N-1}$			

using **state eq. or characteristic table**

using **output eq.**

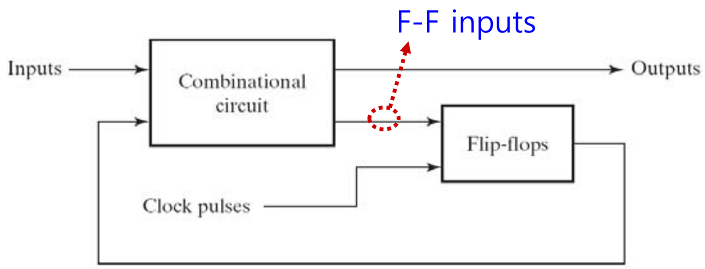
State Diagram

- Objectives
  - A graphical representation of a state table
  - Easier human interpretation
- Components
  - Bubbles(states) and directed lines(transitions)



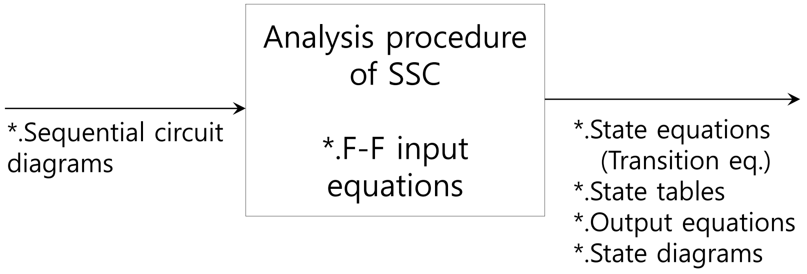
F-F Input Equations

- Definition
  - Output functions of the combinational circuit generating the F-F input.
  - Only an internal expression used for deriving the state equations



Overview of Analysis of SSC

- Flow diagram of analysis of SSC



*.Sequential circuit diagrams

Analysis procedure of SSC

*.F-F input equations

*.State equations (Transition eq.)

*.State tables

*.Output equations

*.State diagrams

Analysis Procedure

- Step-1. Check if clocked seq. circuit
- Step-2. F-F input equations
  - Use the given circuit diagram
- Step-3. Output equations
  - Use the given circuit diagram
- Step-4. State tables
  - Generate full combination of <present states> and <inputs>
  - Refer F-F's characteristic tables to get <next state> columns
- Step-5. State equations
  - Use the state table and K-map or given circuit diagram
- Step-6. State diagrams
  - User the state table or given circuit diagram

Analysis Example

- Circuit diagram
- Input equations

$$D_A = Ax + Bx$$

$$D_B = A'x$$

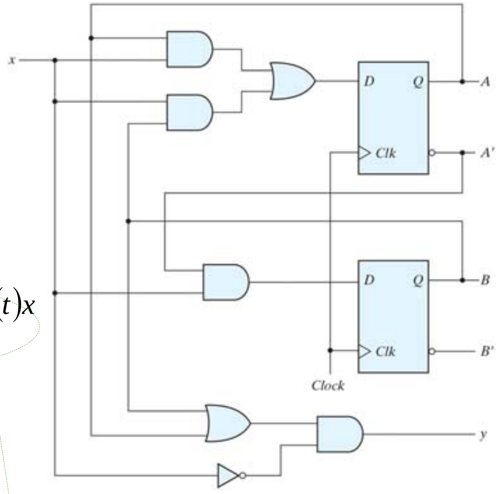
- State equations

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

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

- Output equations

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



☞ Since the next state of D F-F. is determined only the F-F. input *D*.

Analysis Example

- State table
  - state equations

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

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

all binary combination of present states and inputs

$y = (A+B)x'$

Output equations

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

Analysis Example

- State table
  - State equation

Bx

A

		1	
	1	1	

$A_{t+1} = A_t x + B_t x$

Bx

A

	1	1	

$B_{t+1} = A'_t x$

  - Output equation

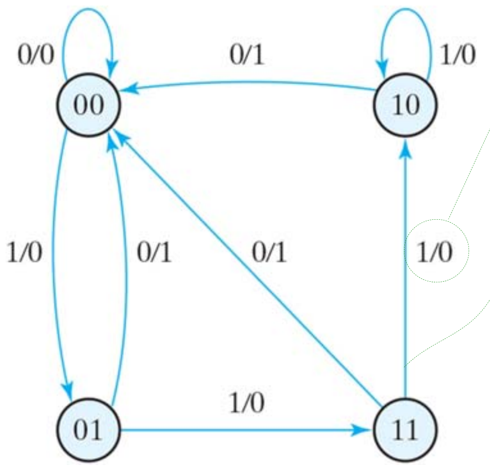
Bx

A

			1
1			1

$y_t = (A_t + B_t)x'$

Analysis Example



- State diagram (transition diagram)
  - present input and output
  - after the next clock cycle, the circuit goes to the next state 01.

Analysis Example

- F-F input equations (excitation equations)

$D_A = Ax + Bx$

$D_B = A'x$

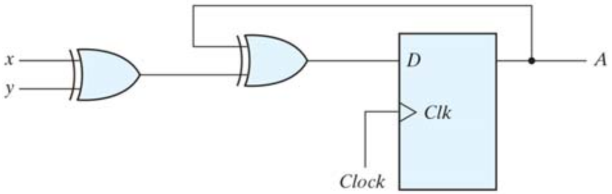
  - fully specify the combinational logic that drives the F-F's
  - necessary for drawing full logic diagram
  - imply the type of F-F from the letter symbol (D_A, J_B, K_B, T_C,...)

Analysis with D F-F

- Input equation

$$D_A = A \oplus x \oplus y$$

  - the D_A symbol implies a D F-F with output A
  - x, y are the inputs to the circuit.
  - no output equation means that <output> = <F-F output>
- Logic diagram



Analysis with D F-F's

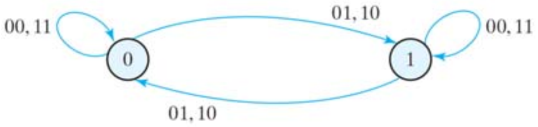
- State table
  - 1 col. for curr. state, 2 col's for inputs, 1 col. for next state.

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

- State equation

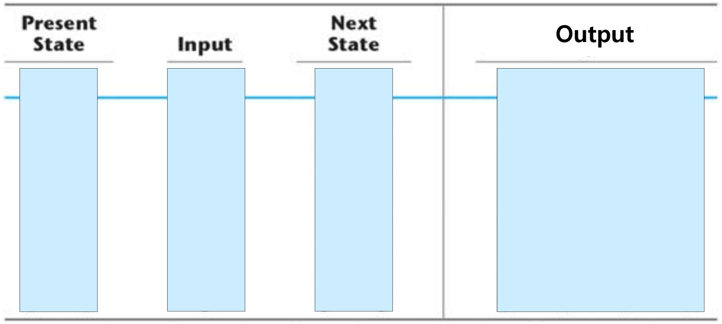
$$A(t+1) = A(t) \oplus x(t) \oplus y(t)$$

- State diagram



Analysis with JK F-F's

- Structure of the state table with JK F-F
  - 4 sections: <present state>, <inputs>, <next state>, <outputs>
  - all combination of {<present state>, <inputs>}



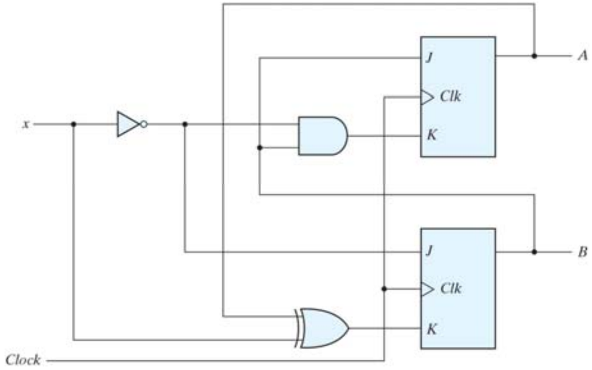
Present State	Input	Next State	Output

Analysis with JK F-F's

- Derivation steps of the next-state values in the state table
  - Determine the F-F input equation
    - in terms of the present state and input variables
  - List the binary values of each input equation
  - Use the F-F characteristic table
    - to determine the next-state values in the state table

Analysis with JK F-F's (Ex.)

- Given circuit diagram
  - 1 input, 2 JK F-F's (2-bit state value), no output (F-F's outputs may be considered as the circuit outputs)



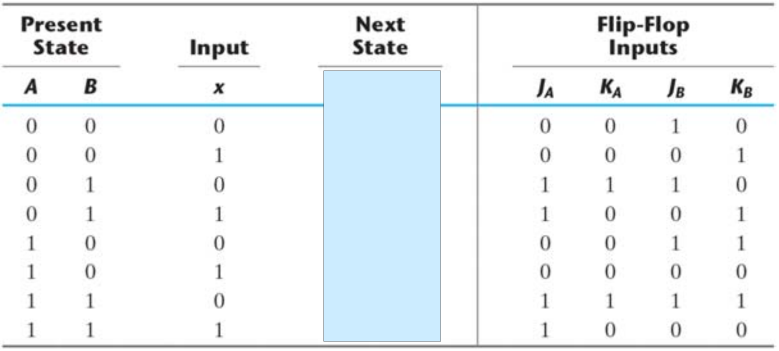
Analysis with JK F-F's (Ex.)

- F-F input equations

$$J_A = B, \qquad K_A = Bx'$$

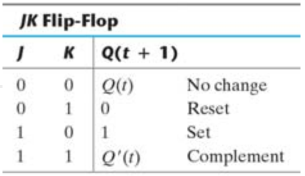
$$J_B = x', \qquad K_B = A \oplus x = A'x + Ax'$$

- F-F input binary values



Present State		Input	Next State	Flip-Flop Inputs			
A	B	x		J_A	K_A	J_B	K_B
0	0	0		0	0	1	0
0	0	1		0	0	0	1
0	1	0		1	1	1	0
0	1	1		1	0	0	1
1	0	0		0	0	1	1
1	0	1		0	0	0	0
1	1	0		1	1	1	1
1	1	1		1	0	0	0

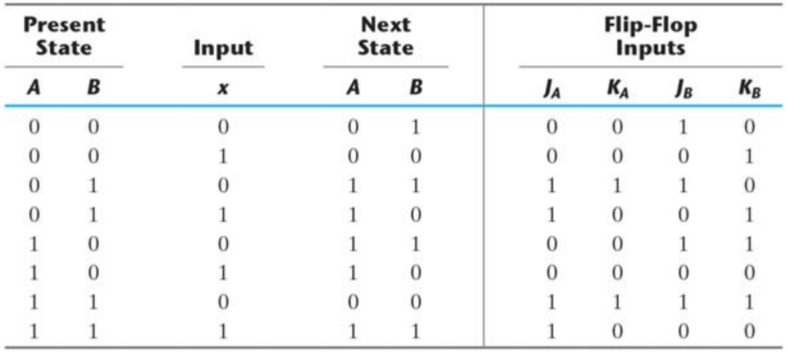
Analysis with JK F-F's (Ex.)



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

- Determine the next state(1)
  - current input JK value and the characteristic 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

Analysis with JK F-F's (Ex.)

- Derive state equations
  - JK F-F characteristic equations

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

$$A(t+1) = JA' + K'A \qquad B(t+1) = JB' + K'B$$

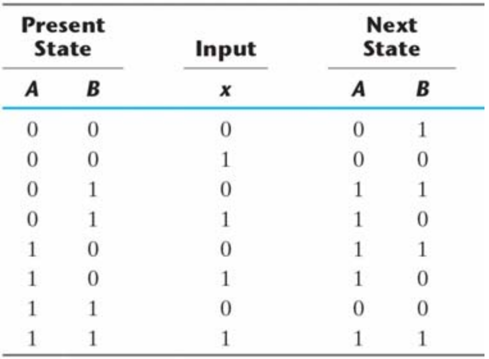
  - substitute input eq. into char. eq.

$$A(t+1) = BA' + (Bx')'A = A'B + AB' + Ax$$

$$B(t+1) = x'B' + (A \oplus x)'B = B'x' + ABx + A'Bx'$$

Analysis with JK F-F's (Ex.)

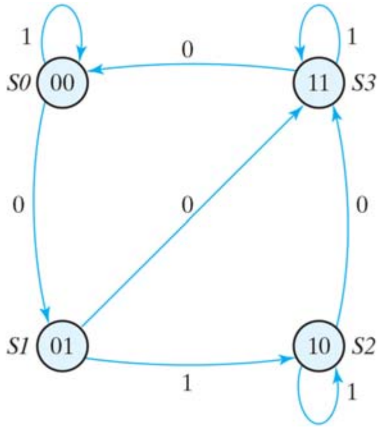
- Determine the next state value(2)
  - obtain the next state value by evaluating the state equations
  - the state equations can be obtained by substituting the F-F input equations into the F-F characteristic equation



Present State		Input	Next State	
A	B	x	A	B
0	0	0	0	1
0	0	1	0	0
0	1	0	1	1
0	1	1	1	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	1	1

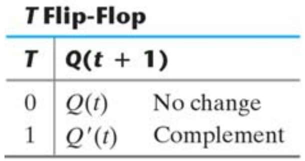
Analysis with JK F-F's (Ex.)

- State diagram



Analysis with T F-F's

- Review
  - T F-F characteristic table



T Flip-Flop

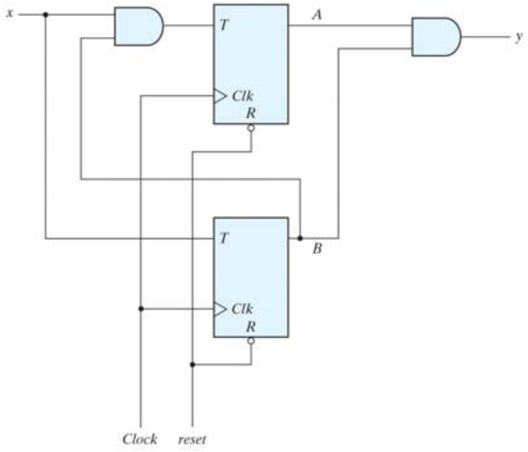
T	Q(t + 1)	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

  - characteristic equation

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

Analysis with T F-F's

- Given circuit diagram



Analysis with T F-F's

- Input equations

$$T_A = Bx$$
$$T_B = x$$

- Output equation

$$y = AB$$

- State equations

$$A(t+1) = (Bx)'A + (Bx)A' = AB' + Ax' + A'Bx$$

$$B(t+1) = x \oplus B$$

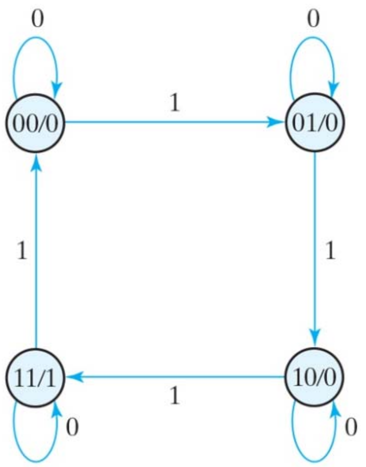
Analysis with T F-F's

- 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

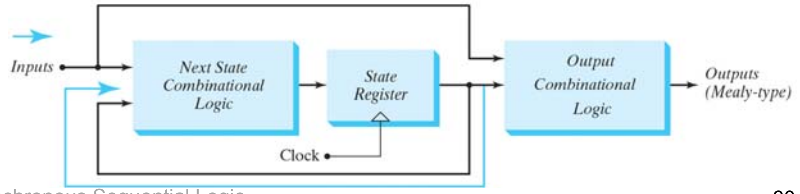
Analysis with T F-F's

- State diagram
  - Binary counter as long as x=1



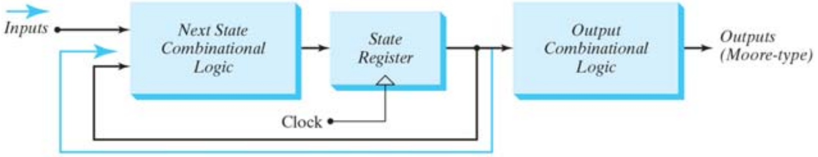
Mealy and Moore Models of FSM

- General model of a sequential circuit
  - inputs, outputs, and internal states
- Mealy model
  - Output is a function of both the present state and input
  - <output> = FUNC(<input>,<present state>)
  - the output should be sampled immediately before the active edge of the clock with concerning the output stabilization.



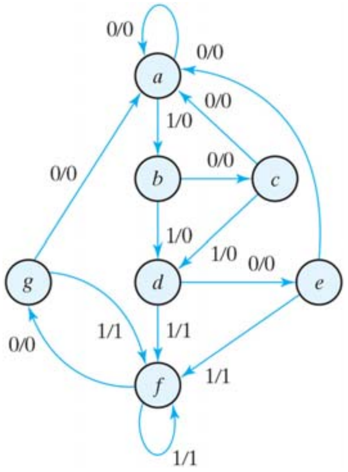
Mealy and Moore Models of FSM

- Moor model
  - output is a function of the present state only
  - <output> = FUNC(<present state>)
  - the output is synchronized with the clock
  - Examples:
    - circuit output is the flip-flop states



State Reduction and Assignment

- Number of F-F's in a SC
  - determined as $\log_2 N$, where N is the number of required states
  - cost ∝ N
  - in general, state reduction is needed.
- As an example, a state diagram is given
  - 7 states: a ~ g



State Reduction

- Input-output relationship (a is the initial state)
  - input: 01010110100
  - output: 00000110100

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	

- State reduction problem
  - to find ways of reducing N without altering the input-output relationships

State Reduction

- State table

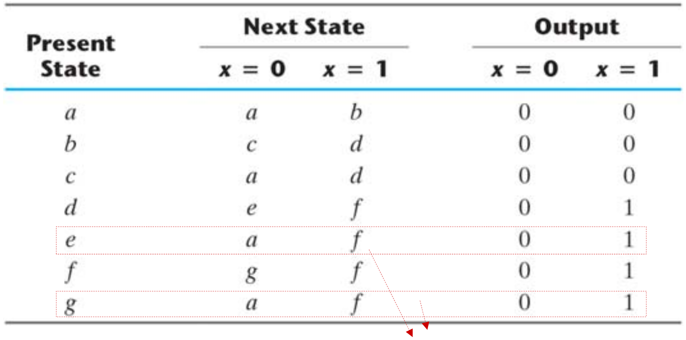
Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*g*	*f*	0	1
g	*a*	*f*	0	1

State Reduction

- two circuits are equivalent:
  - have identical outputs for all input sequences
  - the number of states is not important
- Two states are said to be equivalent:
  - for each member of the set of inputs, they give exactly the same output and send the circuit to the same state or to an equivalent state
  - one of them can be removed

State Reduction

- Find the equivalent states in the state table

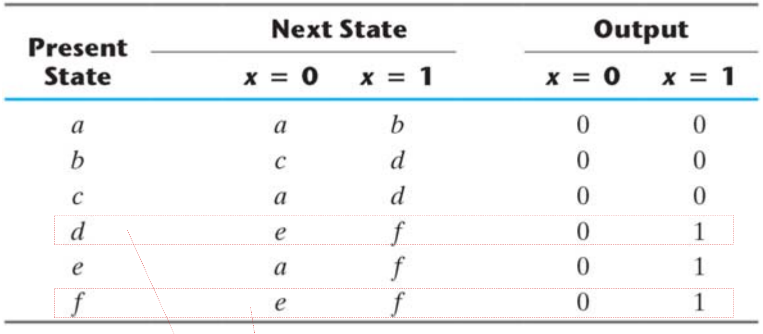


Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

☞ e and g are equivalent. 'e' or 'g' can be eliminated if 'g' is eliminated, all 'g' is replaced by 'e' and the row that 'g' is the present state is eliminated from both the table and the diagram.

State Reduction

- Reduced state table(1)

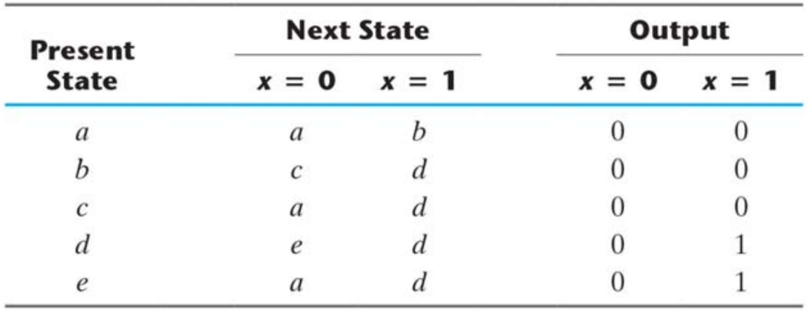


Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

☞ d and f are equivalent again. 'd' or 'f' can be eliminated. If 'f' is eliminated, all 'f's are replaced by 'd' and the row that 'f' is the present state is erased.

State Reduction

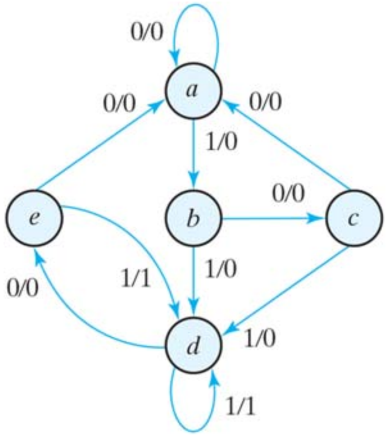
- Reduced state table(2)



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

State Reduction

- Reduced state diagram



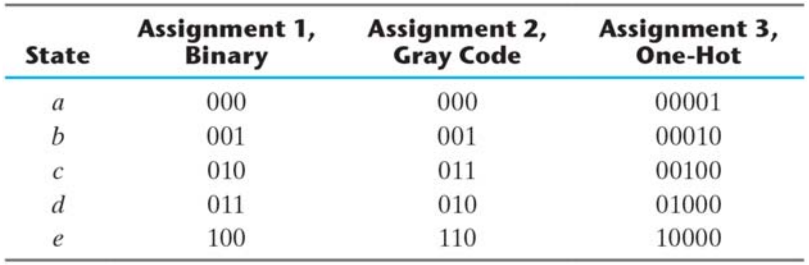
State Reduction

- Test of reduction (a is the initial state)
  - input: 01010110100
  - output: 00000110100

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	

State Assignment

- Why assign values to states
  - to design a SC with physical components.
- Possible state binary assignment

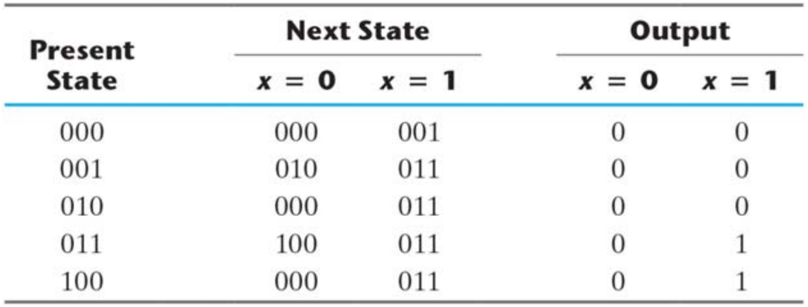


State	Assignment 1, Binary	Assignment 2, Gray Code	Assignment 3, One-Hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

  - gray code: simplification of the Boolean function
  - one-hot : use easy decoder logic

State Assignment

- Reduced state table with binary assignment

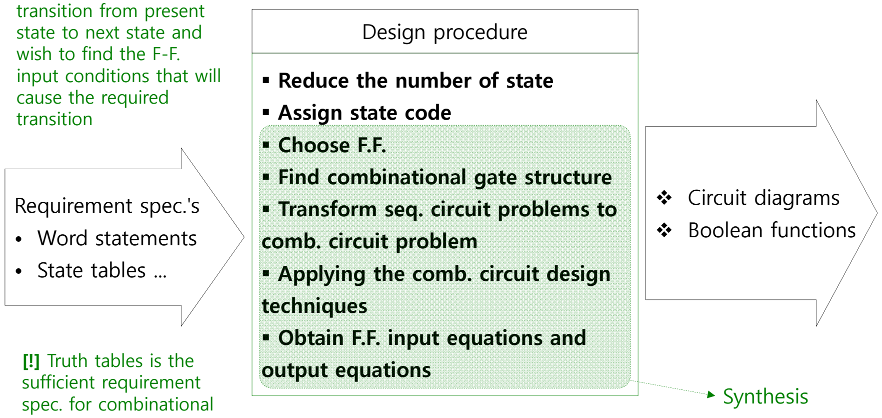


Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

  - Unused states can be considered as don't care

Design Procedure Overview

[!] We usually know the transition from present state to next state and wish to find the F-F. input conditions that will cause the required transition



Requirement spec.'s

- Word statements
- State tables ...

Design procedure

- Reduce the number of state
- Assign state code
- Choose F.F.
- Find combinational gate structure
- Transform seq. circuit problems to comb. circuit problem
- Applying the comb. circuit design techniques
- Obtain F.F. input equations and output equations

❖ Circuit diagrams
❖ Boolean functions

Synthesis

[!] Truth tables is the sufficient requirement spec. for combinational circuit

Requirements Specification

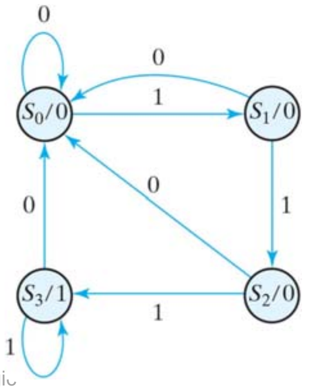
- Word description
  - the word description of the circuit behavior
- State representation
  - state diagram
  - state table
  - state equation
- Etc.
  - block diagram
  - truth table

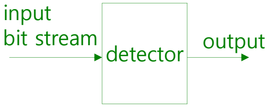
Design Procedure

- Core of design procedure
  - state reduction if necessary
  - assign binary values to the states
  - obtain the binary-coded state table
  - choose the type of flip-flops
  - derive the simplified flip-flop input equations and output equations
  - draw the logic diagram

Example: Bit Sequence Detector

- Word description
  - We wish to design a circuit that detects three or more consecutive 1's in a string of bits coming through an input line
- State diagram

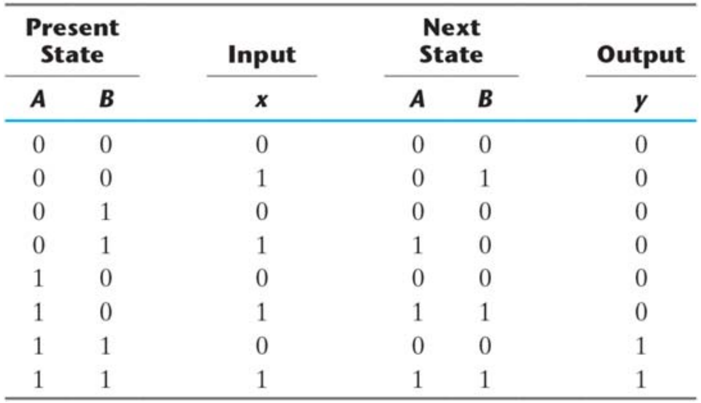




Moore model
∵output is determined by the state

Ex.) Bit Sequence Detector

- Construct a state table
  - by assigning binary value to each state



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	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

Synthesis with D F-F's

- Number of F-F's
  - 2 D F-F's to represent 4 states
- Characteristic function

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

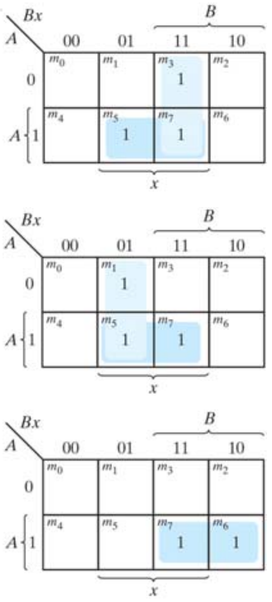
- State equation

$$A(t+1) = \sum(3,5,7) = Ax + Bx$$

$$B(t+1) = \sum(1,5,7) = Ax + B'x$$

- Output equation

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



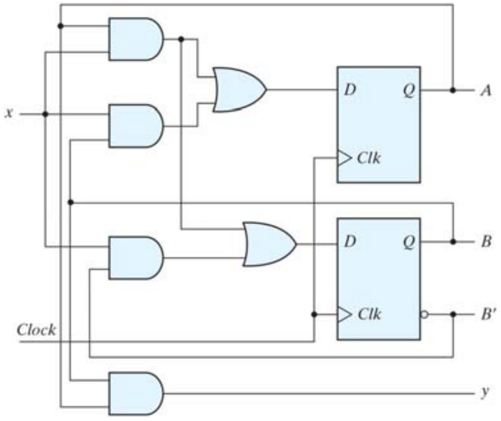
Synthesis with D F-F's

- Input equation
  - Input function is the same form as state equation

$$D_A = Ax + Bx$$

$$D_B = Ax + B'x$$

- Circuit diagram



Excitation Table

- Definition of excitation table
  - Explains the requirement for state transition of the sequential circuits
  - Represents the F-F. input conditions required for the given change of state
- Why excitation table?
  - F-F. input equations can not be derived directly from the state table in case of JK and T F-F.s.
- [C.F.] Characteristic tables
  - Provide the value of the next state of a F-F. when the F-F. inputs and present state are known
  - Provides the information for constructing the excitation table

Excitation Table

- Typical form of excitation table

<Present state≭Next state>	<F-F. inputs>
List of given changes of states	List of F-F. inputs required for each state transition

- Excitation tables of JK F-F. and T F-F.

$Q(t)$	$Q(t=1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

$Q(t)$	$Q(t=1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

Synthesis with JK F-F's

- State table and JK F-F inputs
  - JK F-F input equations are derived from the excitation 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	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

Excitation table

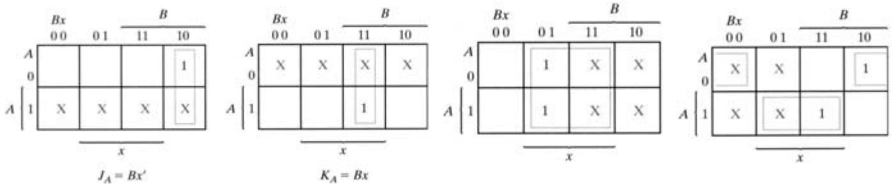
Synthesis with JK F-F's

- Derive JK F-F. input equations
  - Using the extended state table

$$J_A = Bx'$$
$$K_A = Bx$$
$$J_B = x$$
$$K_B = (A \oplus x)'$$

<present state>	<inputs>	<F-F. inputs>

[NOTE] <next state> is not used for deriving the F-F. input equations

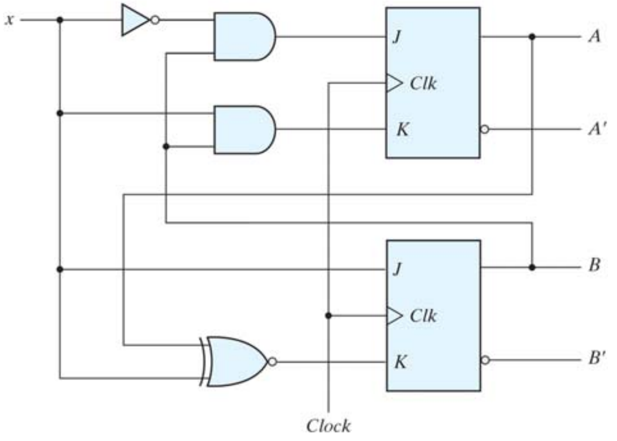


$J_A = Bx'$

$K_A = Bx$

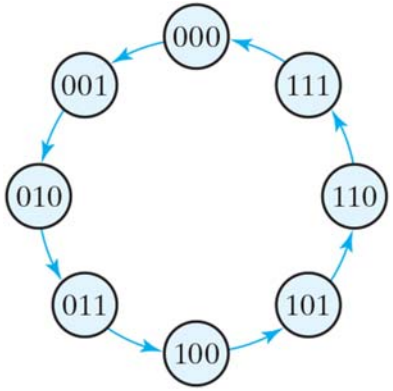
Synthesis with JK F-F's

- Circuit diagram



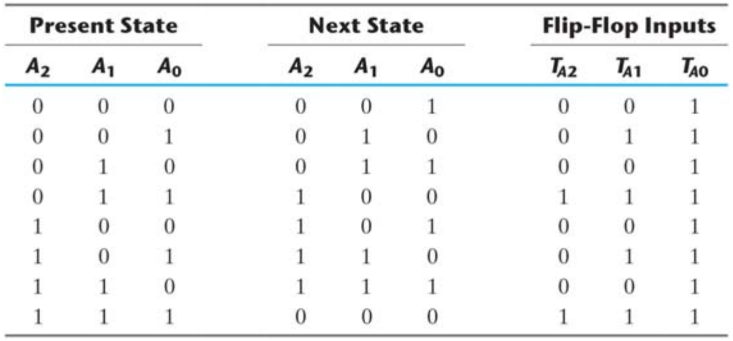
3-bit Binary Counter

- Requirement spec.
  - The counter is driven by the clock signal
  - There's no input and no output notation



Synthesis with T F-F's

- Obtain T F-F inputs
  - from the state diagram and T F-F excitation table

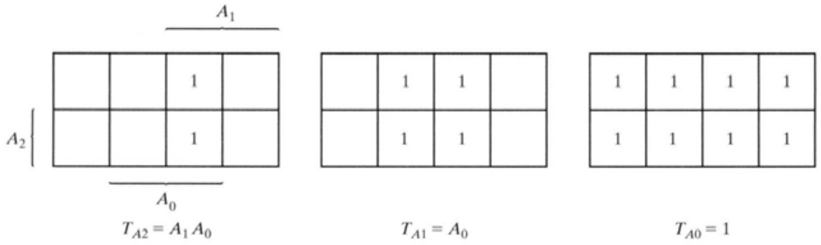


Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

Synthesis with T F-F's

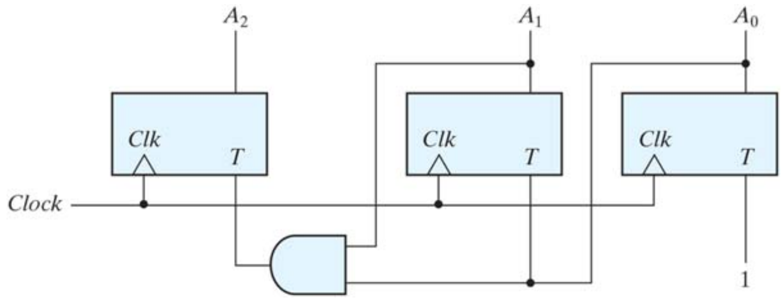
- Derive F-F. input equations
  - From the extended state table and by simpliflng with K-maps

$$T_{A2} = A_1A_0 \qquad T_{A1} = A_0 \qquad T_{A0} = 1$$



Synthesis with T F-F's

- Logic diagram of 3-Bit Binary Counter



Discussion~~~