Lecture Outline

Reminders to self:

- ❑ Turn on lecture recording to Cloud
- ❑ Turn on Zoom microphone

- Last Lecture
  - Finished Counters
    - Finished counter design with other types of flip-flops (SR and JK)
    - Analysis of Counter Circuits – verifying design does the expected
- Today’s Lecture
  - Start Analysis of clocked sequential circuits
    - Parity checker design example
    - Analysis by signal tracing and timing charts
    - Definitions of Moore and Mealy machines
    - Moore & Mealy machine analysis examples
    - Start Analysis by Transition Tables & State Graphs

Handouts and Announcements

- Announcements
  - Homework Problem: No new homework assignment
  - Homework Reminder
    - HW 12-4 Due: 11:59pm Tuesday 3/21
    - HW 12-5 Due: 11:25am Wednesday 3/22
  - Read for Monday 3/20: pages 432-436
  - Mini-Exam 3 regrade continuing
    - A-some C regraded
    - N-Z regraded

Clocked Sequential Circuits

- The counters we designed:
  - Went through a fixed sequence of states
  - Had only the clock as an input, which triggered change of state
  - Exception:
    - The of last lecture had additional inputs
    - Allowed a selection between of states
- In general, sequential circuits may have additional inputs
  - Sequence of will depend on those inputs
  - Sequence of will depend on those inputs
- We will start by looking at specific examples
- First example: Sequential Parity Checker

Sequential Parity Checker

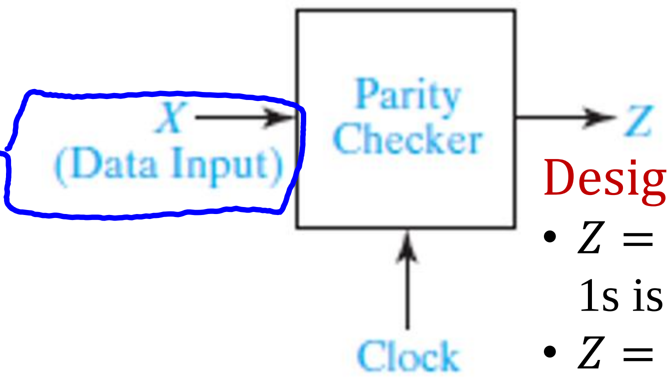
- Parity:
  - When binary data is transmitted/received (TX/RX) or stored it is susceptible to corruption from things such as noise or interference
  - An extra bit (a parity bit) is frequently added to each data "word" for error detection
  - Example: 7-bit data word + 1 parity bit = 8 bit TX word
  - Odd Parity:
    - Total number of "1" bits in the word (including the parity bit) is odd
    - Value of parity bit chosen to make that true (e.g. 0110111<u>0</u>)
  - Even Parity:
    - Total number of "1" bits in the word (including the parity bit) is even
    - Value of parity bit chosen to keep make true (e.g. 0110111<u>1</u>)

Sequential Parity Checker

- Parity:
  - Odd Parity:
    - Total number of "1" bits in the word (including the parity bit) is odd
    - Value of parity bit chosen to keep that true (e.g. 01101110)
  - Even Parity:
    - Total number of "1" bits in the word (including the parity bit) is even
    - Value of parity bit chosen to keep that true (e.g. 01101111)
  - If noise or interference causes any single bit in the word to flip, the presence of an error can be detected from the change in parity
    - As an FYI:
      - Detection of more than one error in a word requires adding additional "check bits"
      - Error correction is also possible - generally requires that even more extra bits be added
      - Not this example or this course

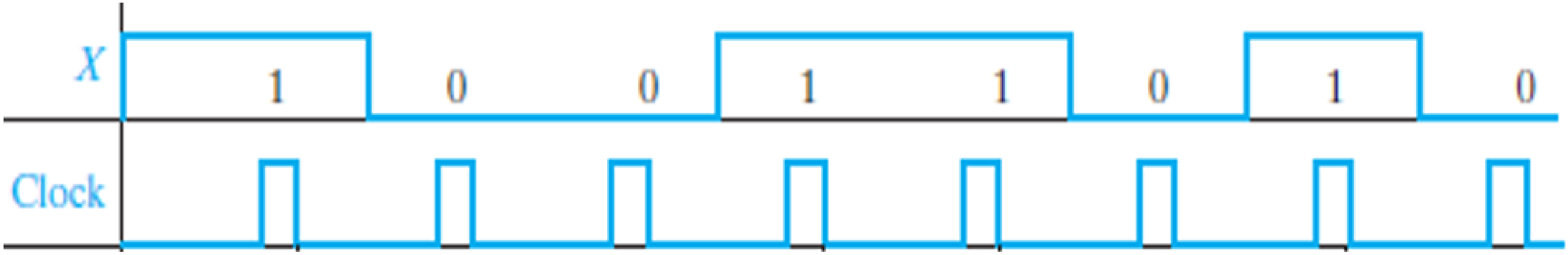
Sequential Parity Checker

- Parity Checker:
  - We will design a parity checker for serial data
  - "Serial" means that data enters circuit one bit at a time



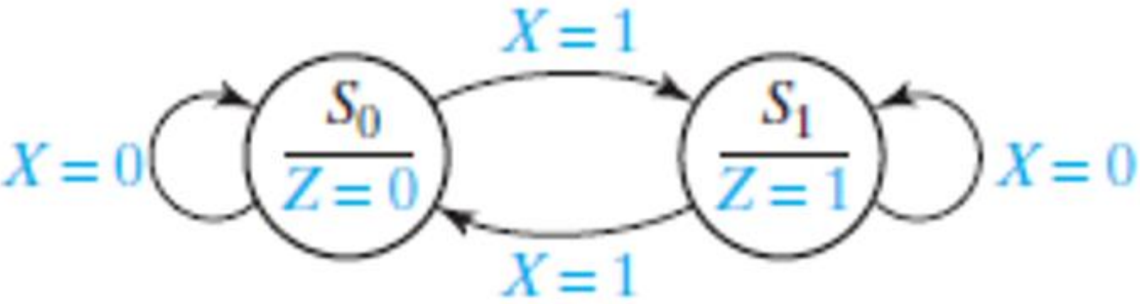
Design such that
- $Z = 1$ when # of 1s is odd
- $Z = 0$ when # of 1s is even

  - When data with odd parity is transmitted, a final output $Z = 0$ indicates that a transmission error occurred
  - Value of X read at time of active clock edge
    - The X input data stream must be synchronized with clock, so that it assumes its next value before the next active clock edge
    - The clock is necessary in order to distinguish consecutive 0s or consecutive 1s



Sequential Parity Checker

- Since only two states are needed one flip-flop is sufficient
  - Let state S_0 correspond to an even number of 1s received
  - Let state S_1 correspond to an odd number of 1s received
- State Graph:

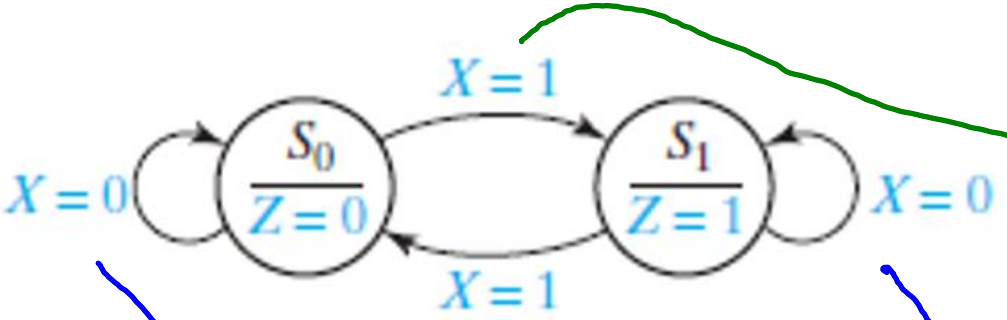


Design such that

- $Z = 1$ when # of 1s is ODD
- $Z = 0$ when # of 1s is EVEN

- Initially zero ones have been received
- Zero is an even number, so start system in state S_0 (   )
  - $X = 0$ arrives, stay in S_0 (# 1s still   )
  - $X = 1$ arrives, go to state S_1 (# 1s now   )
- When in state S_1 (   )
  - $X = 0$ arrives, stay in S_1 (# 1s still   )
  - $X = 1$ arrives, go to state S_0 (# 1s now   )

Sequential Parity Checker

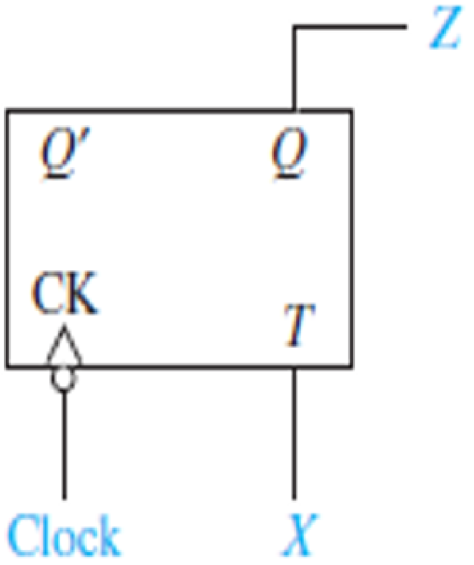


State Table corresponding to State Graph

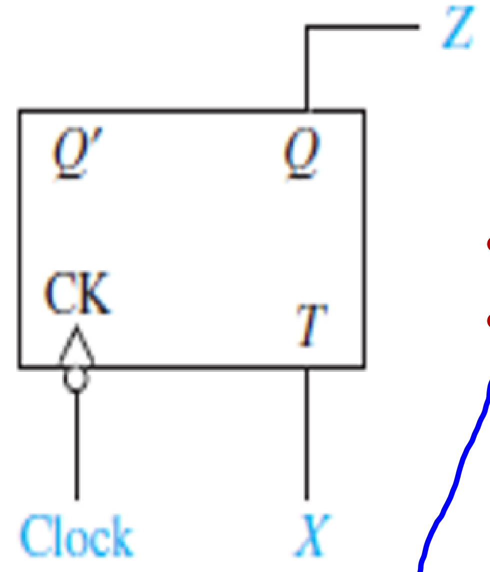
Present State	Next State X = 0	Next State X = 1	Present Output
S_0	S_0	S_1	0
S_1	S_1	S_0	1

T Flip-Flop Transition Table corresponding to State Table

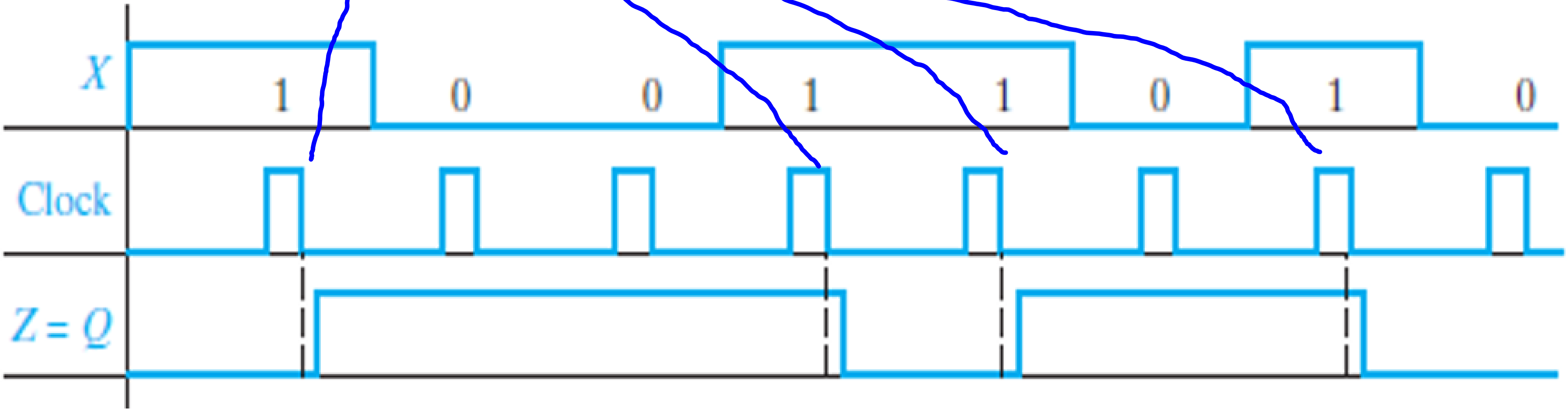
Q	Q^+ X = 0	Q^+ X = 1	T X = 0	T X = 1	Z
0	0	1	0	1	0
1	1	0	0	1	1



Sequential Parity Checker



- Falling edge clock for this example flip-flop
- State change at each falling edge when $X = T = 1$
  - For our 8-bit TX word and this example
  - Final value is 0 since even number of 1 bits received



Analysis by Signal Tracing & Timing Charts

Basic Procedure to Find Output Sequence By Tracing 0 and 1 Signals through Circuit:

1. Assume an initial state of the flip-flops (all flip-flops reset to 0 unless otherwise specified)
2. For 1st set of inputs in the given sequence, determine
   - Circuit output(s)
   - Flip-flop inputs
3. Determine the new set of flip-flop states (after next active clock edge)
4. Determine output(s) that corresponds to the new states
5. Repeat 2, 3, and 4 for each input in the given sequence

Analysis by Signal Tracing & Timing Charts

Two types of clocked sequential circuits:

1. Moore machine
    - Output of sequential circuit is a function of the present state only
    - State graph for a Moore machine has the output associated with the state
2. Mealy machine
    - Output is a function of both the present state and the input
    - State graph for a Mealy machine has the output associated with the arrow going between states

Analysis by Signal Tracing & Timing Charts

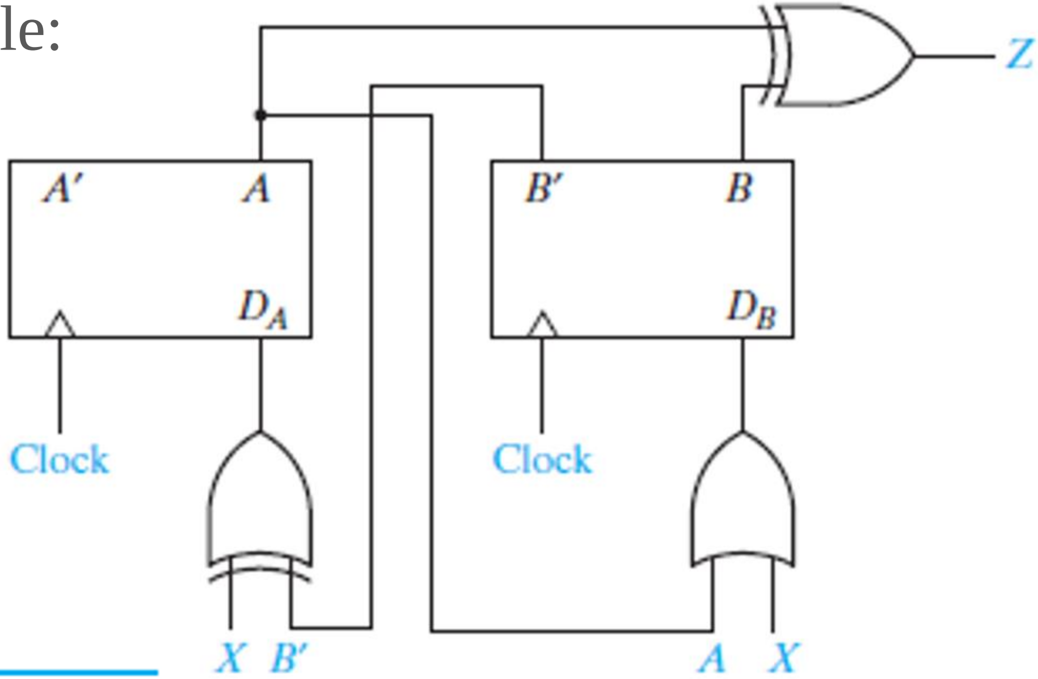
Finite State Machines:

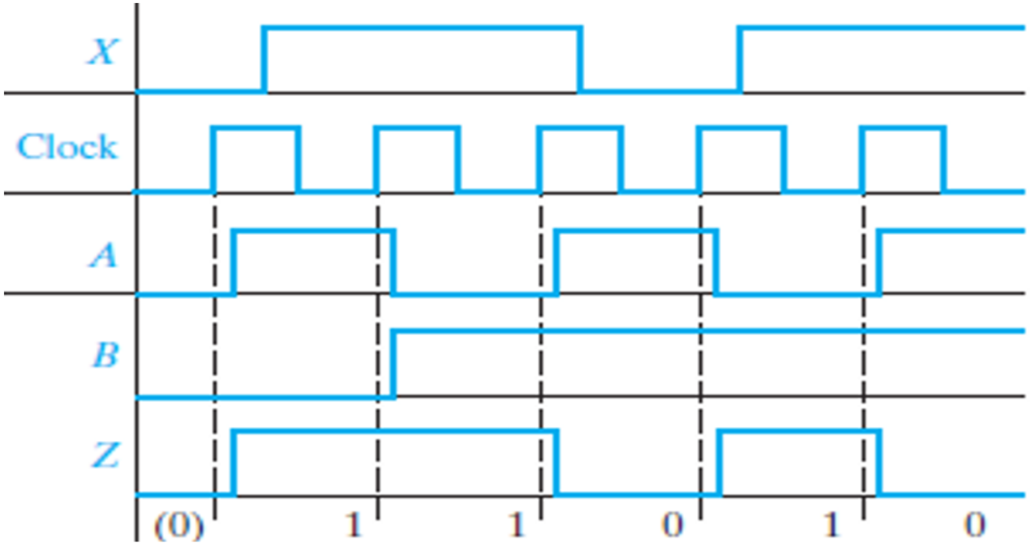
- Special purpose hardware used to implement simple algorithms
- Two most prevalent approaches: Mealy & Moore
  - Both define specific states
  - Use “inputs” to determine state changes
- Mealy: Output depends on both the present state and the inputs
- Moore:
  - Output depends only on present state
  - The parity detector we just designed is a Moore machine
  - Usually has more states
  - Stable outputs between clock transitions
- If inputs have a “glitch,” Mealy Machine output may change

Analysis by Signal Tracing & Timing Charts

Moore Machine Analysis Example:

- X is input
- Z is output
- Initial state $A = B = 0$
- Analyze for input sequence $X = 01101$





- $Z = A \otimes B$ function of present state only
- Changes only when state changes

  moore
- Initially $X = 0$

  $D_A = X \otimes B' = 0 \oplus 1 = 1$

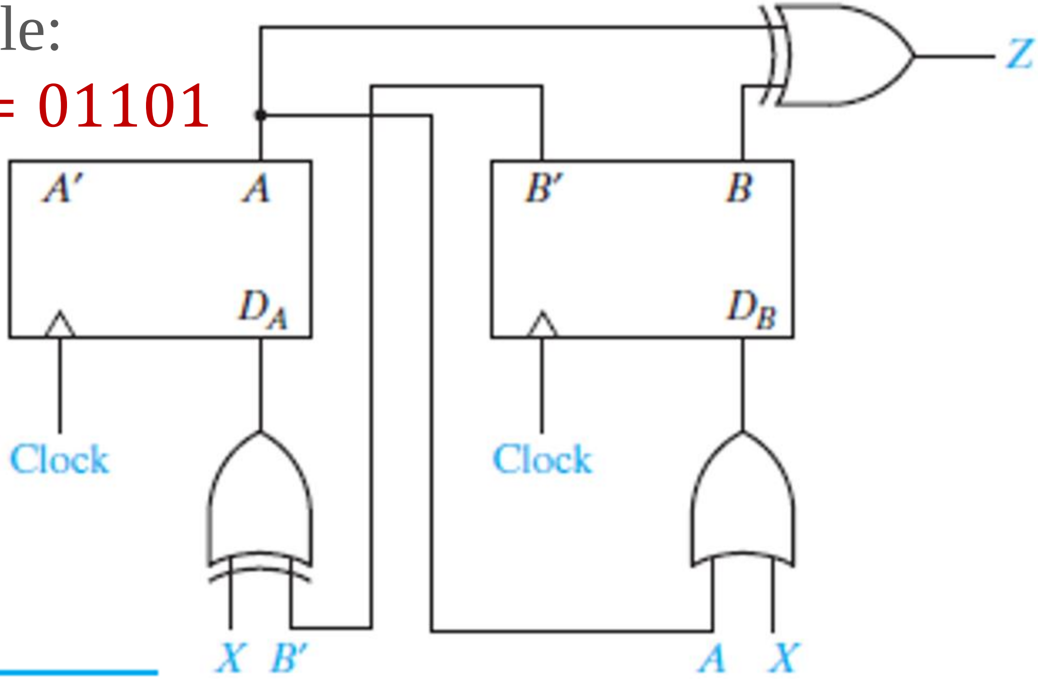
  $D_B = X + A =$
- Next state $A =$, $B =$
- New output $Z = A \otimes B = 1 \oplus 0 = 1$

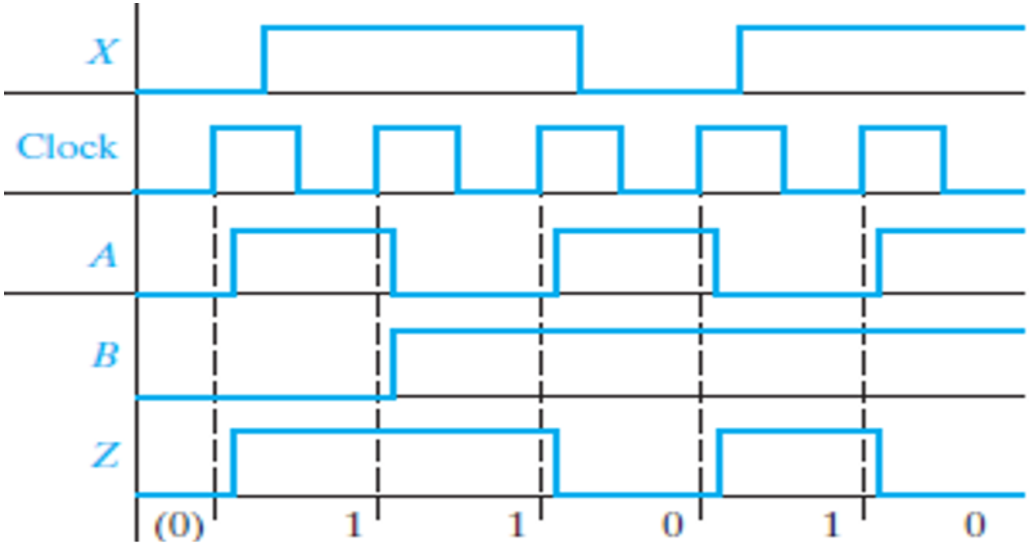
Analysis by Signal Tracing & Timing Charts

Moore Machine Analysis Example:

- Analyze for input sequence $X = 01101$
  - New present state $A = 1$, $B = 0$
  - New $X = 1$
    $D_A = X \oplus B' =$
    $D_B = X + A =$
  - Next state $A =$, $B =$
  - New output $Z = A \oplus B =$

  - Repeat for each X



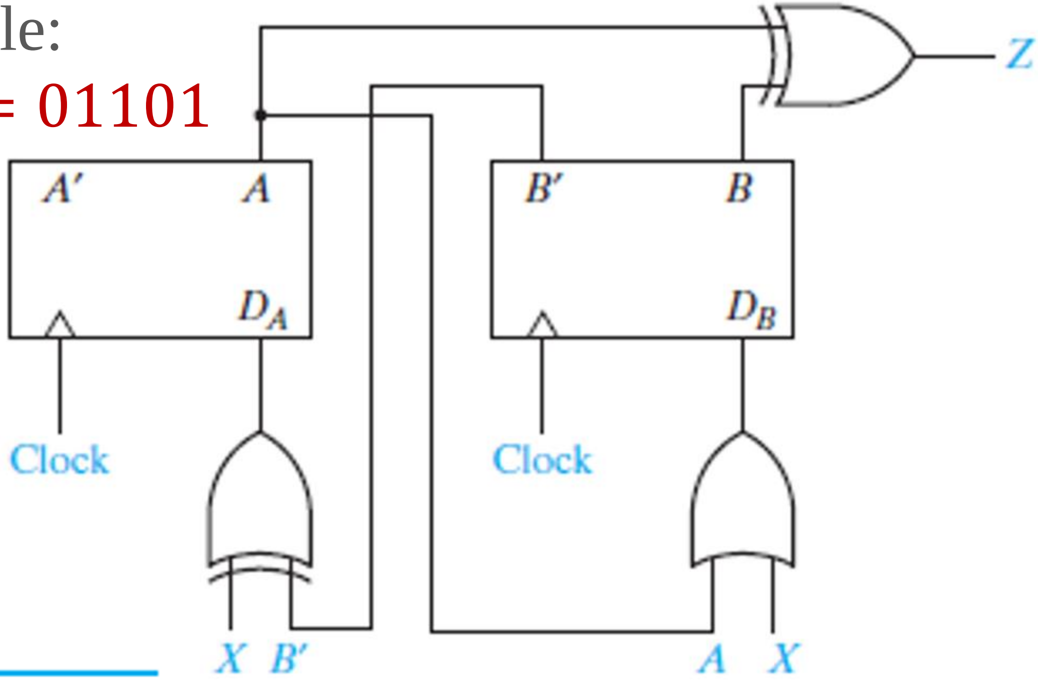


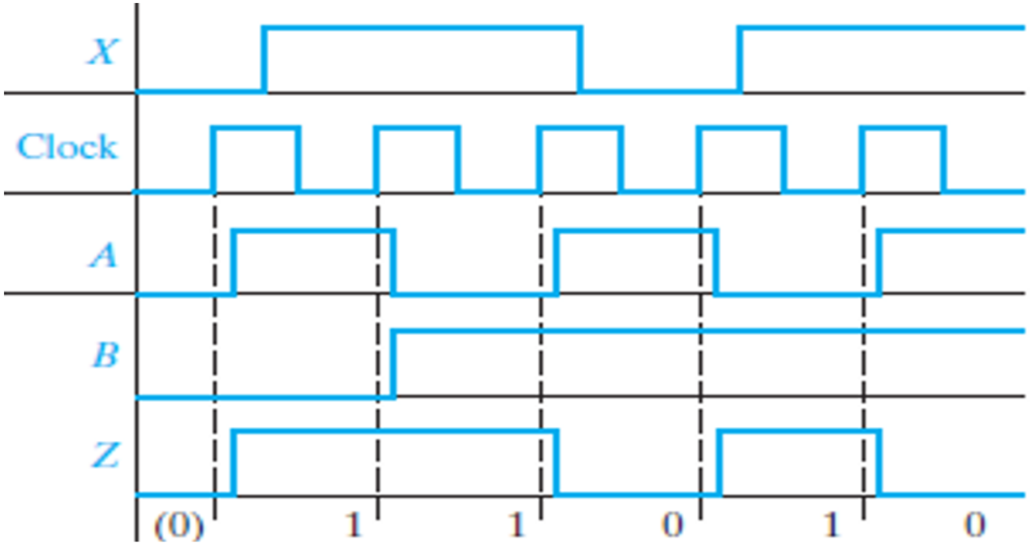
$X =$	0	1	1	0	1	
$A =$	0	1	0			
$B =$	0	0	1			
$Z =$	(0)	1	1			

Analysis by Signal Tracing & Timing Charts

Moore Machine Analysis Example:

- Analyze for input sequence $X = 01101$
- Output that results from application of given input appears after active clock edge
- Output sequence is displaced in time with respect to input sequence

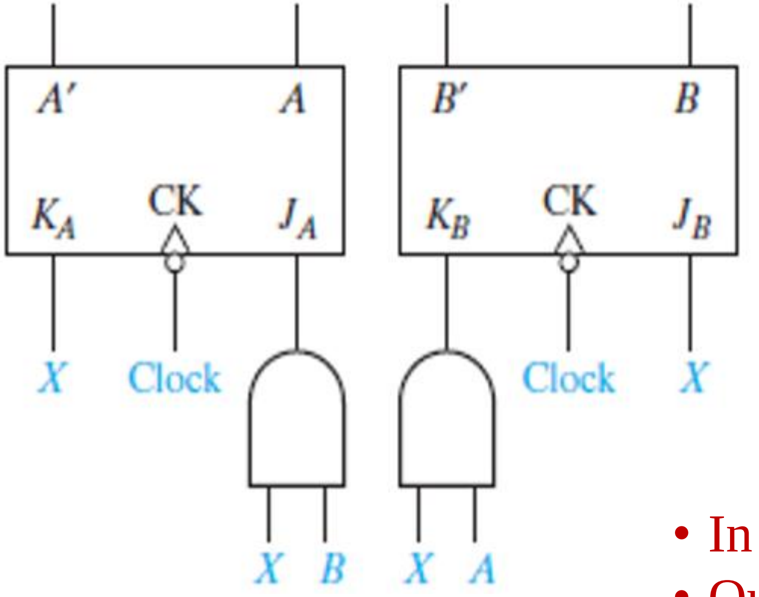


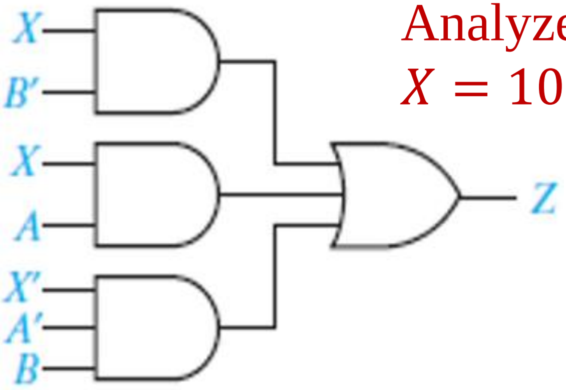


$X =$	0	1	1	0	1	
$A =$	0	1	0	1	0	1
$B =$	0	0	1	1	1	1
$Z =$	(0)	1	1	0	1	0

Analysis by Signal Tracing & Timing Charts

Mealy Machine Analysis Example:





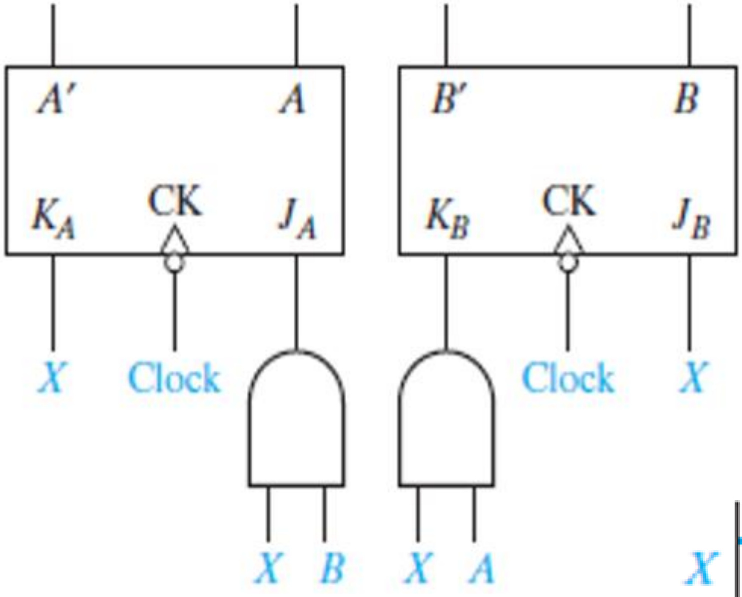
Analyze for input sequence
$X = 10101$

- In addition to depending on state of flip-flops, A and B
- Output Z also depends on input X

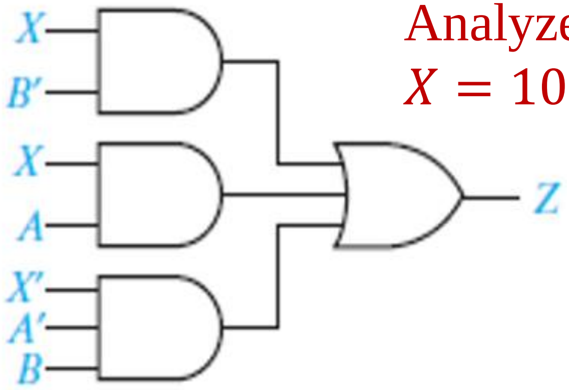
- Initially $X = 1$
- $Z =$
  $J_A = XB =$
  $K_B = XA =$
- $J_B = K_A = X =$
- Next state A : $A =$
  B , $B =$
- New output $Z =$
- Repeat for subsequent values of X

- Z can change either:
  - When the state changes, or
  - When X changes
- Falling edge clock for these J-K flip-flops
- Initial state $A = B = 0$

Analysis by Signal Tracing & Timing Charts

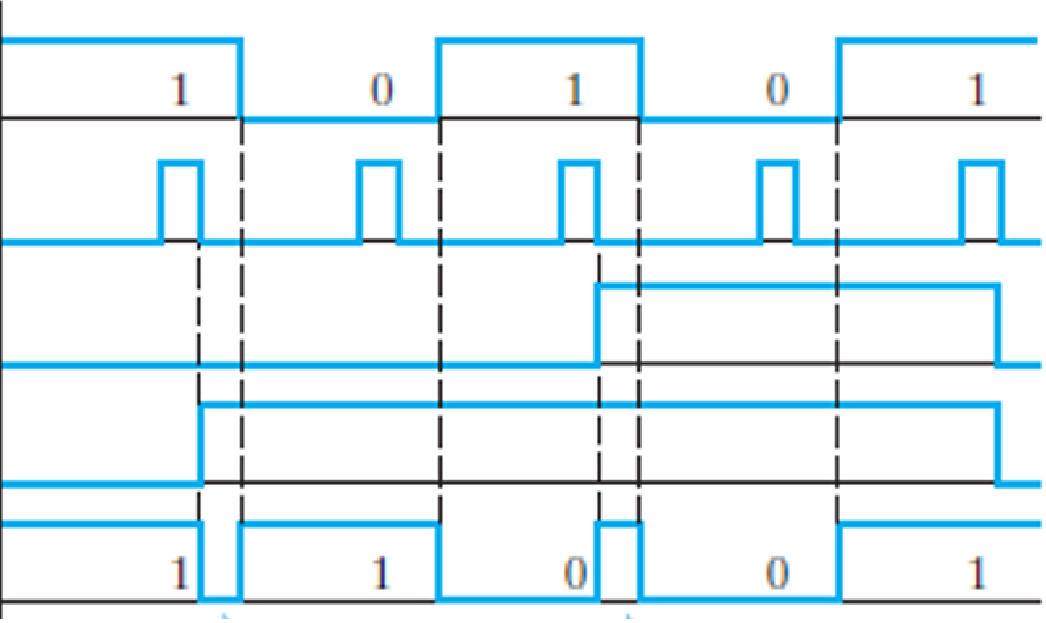


Mealy Machine Analysis Example:



Analyze for input sequence $X = 10101$

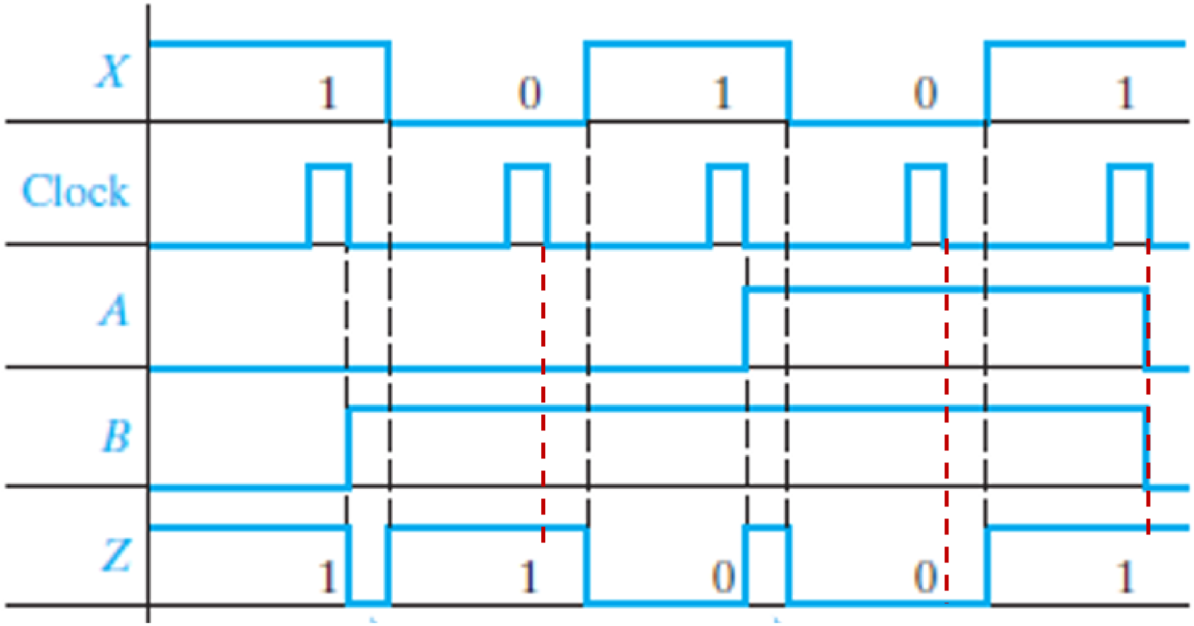
- What we just calculated
- But when X changes to next value $X = 0$
- Output back to 1 before next clock falling edge
- For clocked sequential circuits input values immediately before active clock edges determine next state



Analysis by Signal Tracing & Timing Charts

Mealy Machine Analysis Example:

- For clocked sequential circuits, input values immediately before active clock edges determine next state
- In a similar manner, output from Mealy circuit is of interest only immediately before active clock edge
- Extra output changes between active clock edges should be ignored (   )
- Ignore false outputs by reading outputs just before active clock edge
- In this case, output sequence $Z = 11001$
- If output of Mealy circuit is read by another sequential circuit using the same clock the false outputs are automatically ignored by the second circuit



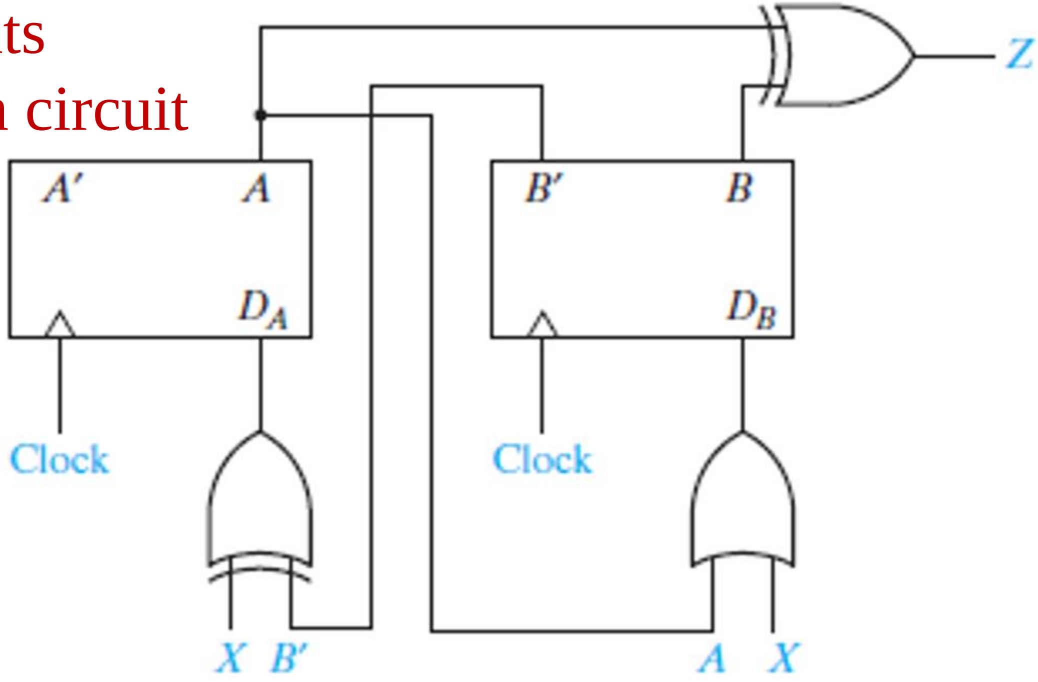
Basic Procedure to Find Output Sequence By Transition Tables and Graphs:

1. Determine equations for inputs to flip-flops and outputs from circuit
2. Derive next-state equations for each flip-flop from its input equations (using flip-flop next-state relations)
   - D: $Q^+ = D$
   - D-CE: $Q^+ = D \cdot CE + Q \cdot CE'$
   - T: $Q^+ = TQ' + T'Q = T \oplus Q$
   - S-R: $Q^+ = S + R'Q$
   - J-K: $Q^+ = JQ' + K'Q$
3. Plot a next-state map for each flip-flop
4. Combine these maps to form the transition table that gives the next state of the flip-flop as a function of current state and circuit inputs
5. Use the transition table to form the state table
6. Use the state table to draw the state graph

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:

1. Determine equations for inputs to flip-flops and outputs from circuit

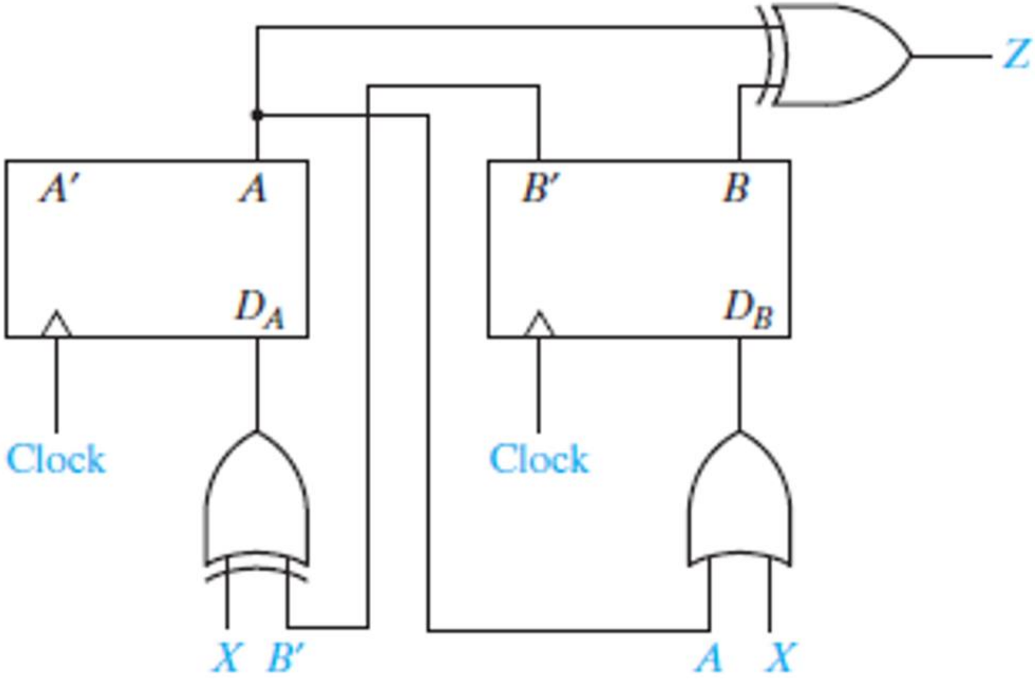


- $D_A = X \oplus B'$
- $D_B = X + A$
- $Z = A \oplus B$
- These are same as when worked before
- Except before we plugged in values of all variables for the initial state right away
- And then repeated as we stepped through each state

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:

- $D_A = X \oplus B'$
- $D_B = X + A$
- $Z = A \oplus B$



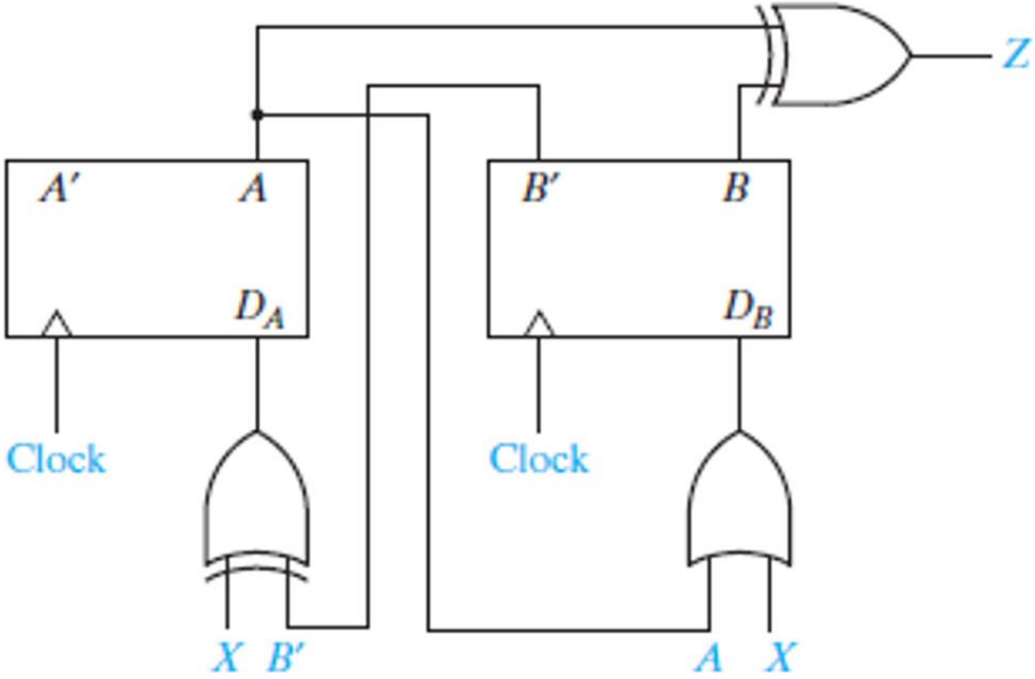
2. Derive next-state equations for each flip-flop from its input equations (using flip-flop next-state relations)
   - D: $Q^+ = D$

- $A^+ = X \oplus B'$
- $B^+ = X + A$

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:

- $A^+ = X \oplus B'$
- $B^+ = X + A$
- $Z = A \oplus B$



3. Plot a next-state map for each flip-flop

A^+

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

B^+

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

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:

- $Z = A \oplus B$

4. Combine these maps to form the transition table that gives the next state of the flip-flop as a function of current state and circuit inputs

A^+

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

B^+

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

	A^+B^+		
AB	$X = 0$	$X = 1$	$Z = A \oplus B$
00	10		0
01	00		1
11	01		0
10	11		1

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:

- $Z = A \oplus B$

State Name definitions for this example

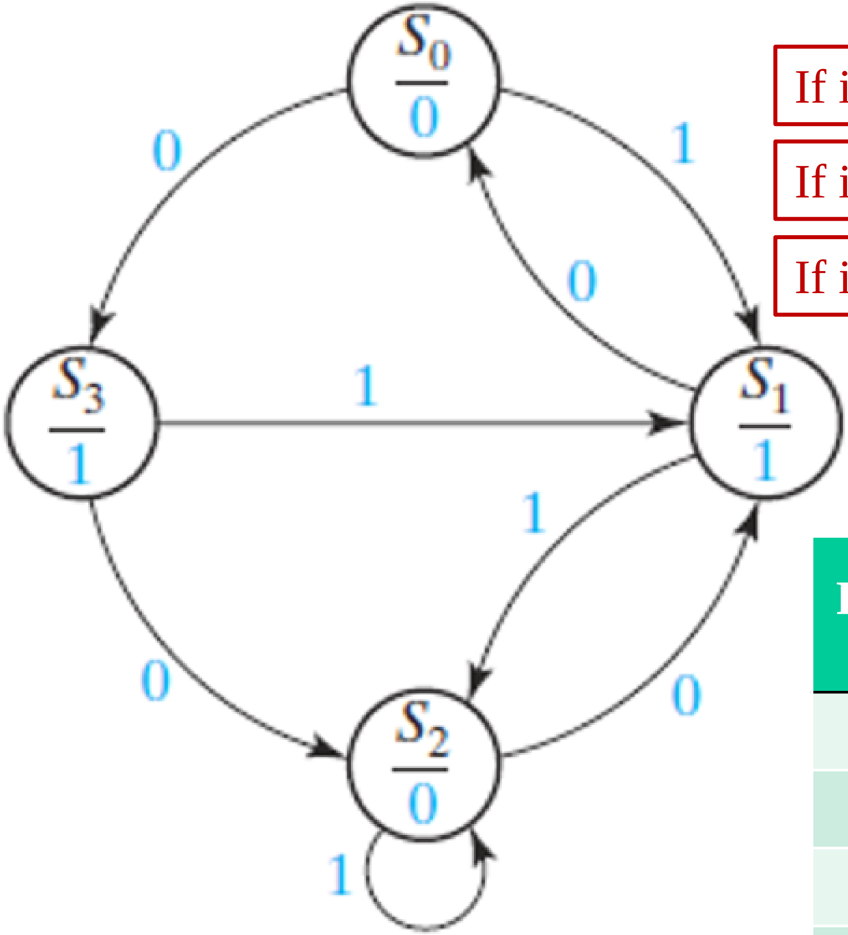
4. Combine these maps to form the transition table that gives the next state of the flip-flop as a function of current state and circuit inputs

	A^+B^+		
AB	$X = 0$	$X = 1$	$Z = A \oplus B$
00	10	01	0
01	00	11	1
11	01	11	0
10	11	01	1

Present State	Next State		Present Output
	$X = 0$	$X = 1$	$Z = A \oplus B$
S_0			0
S_1			1
S_2			0
S_3			1

Analysis by Transition Tables & Graphs

Repeat Moore example using these techniques:



If in S_0 and $X = 0$, go to S_3 at next active clock edge

If in S_0 and $X = 1$, go to S_1 at next active clock edge

If in S_1 and $X = 1$, go to S_2 at next active clock edge

If in S_2 and $X = 0$, go to S_1 at next active clock edge … etc.

Present State	Next State		Present Output
	$X = 0$	$X = 1$	$Z = A \oplus B$
S_0	S_3	S_1	0
S_1	S_0	S_2	1
S_2	S_1	S_2	0
S_3	S_2	S_1	1