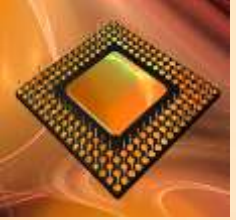
# Modern Digital System Design

## ECE 2372 / Fall 2018 / Lecture 15

Texas Tech University
Dr. Tooraj Nikoubin

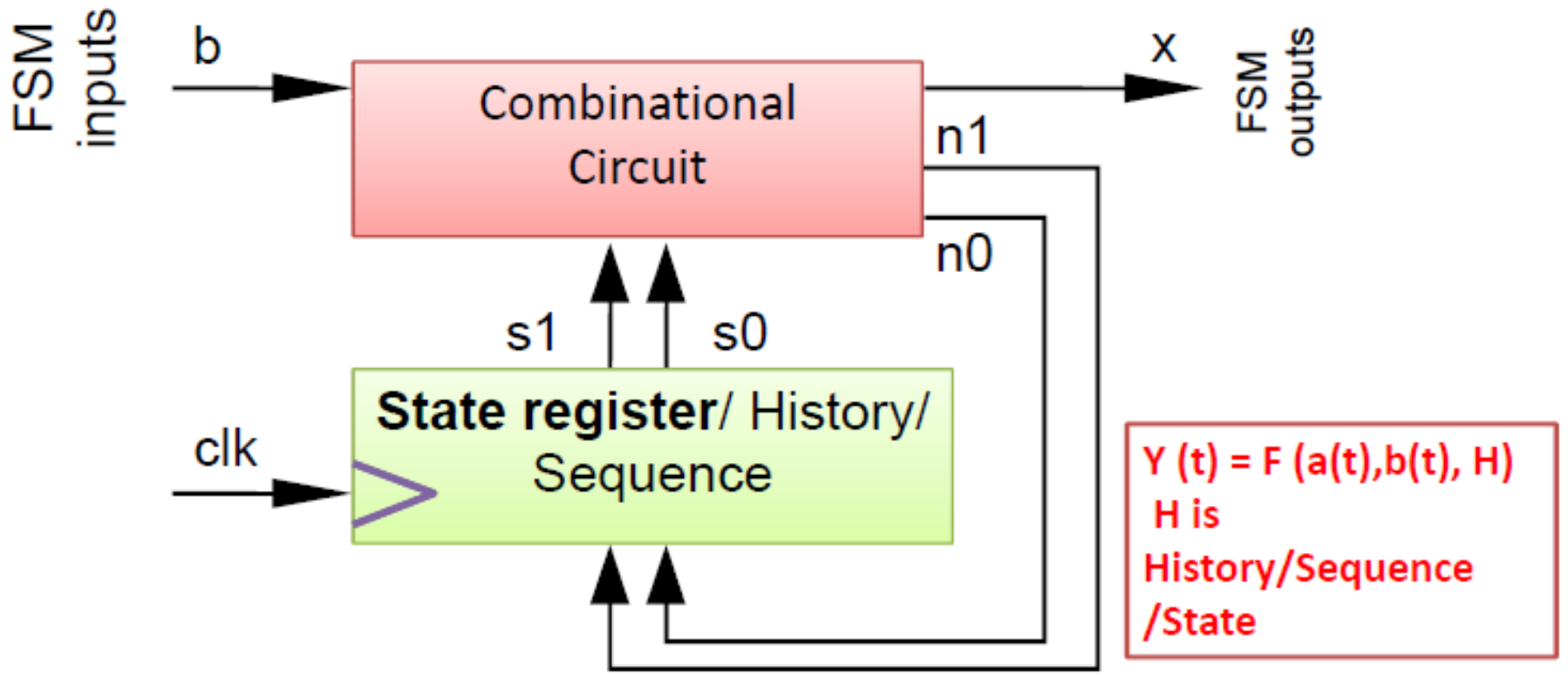
## Finite State Machines



# Outline

- **Finite State Machine**
  - Describe the sequential behavior using a **FSM**
  - **Example of FSM**
  - Convert a finite state machine to a **Controller** – a sequential circuit having
    - **A register and Combinational logic**

# Sequential Circuit: FSM

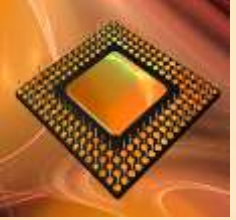



Y (t) = F (a(t),b(t), H)
H is History/Sequence /State

Formally Describe/mathematically Describe

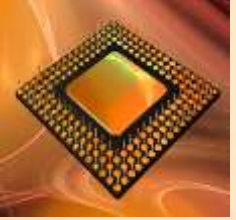
Boolean Algebra: Working Combinational Circuit

Finite State Machine: Working of Sequential Circuit



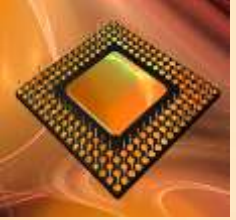
# Need Better Way to Design Sequential Circuits

- Trial and error : not a good design method
  - Will we be able to "guess" a circuit that works for other desired behavior?
    - How about counting up from 1 to 9?
    - Pulsing an output for 1 cycle every 10 cycles?
    - Detecting the sequence 1 3 5 in binary on a 3-bit input?



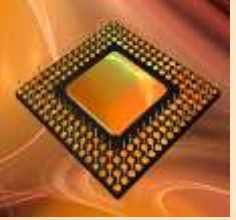
# Need a Better Way to Design Sequential Circuits

- Combinational circuit design process had two important things
  1. A formal way to describe desired circuit behavior
     - Boolean equation, or truth table
  2. A well-defined process to convert that behavior to a circuit
- We need those things for sequence circuit design



# Finite State Machine

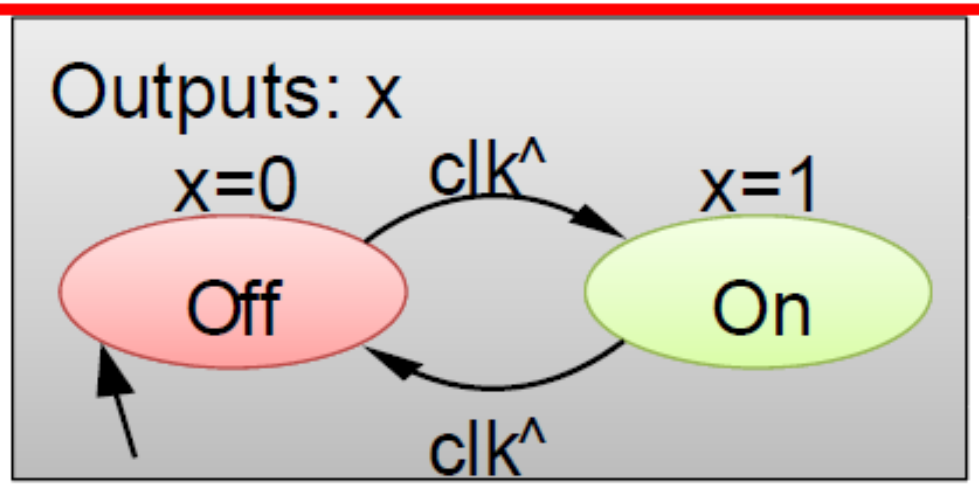
- Finite-State Machine (FSM)
  - A way to describe desired behavior of sequential circuit
  - Akin to Boolean equations for combinational behavior
  - List states, and transitions among states

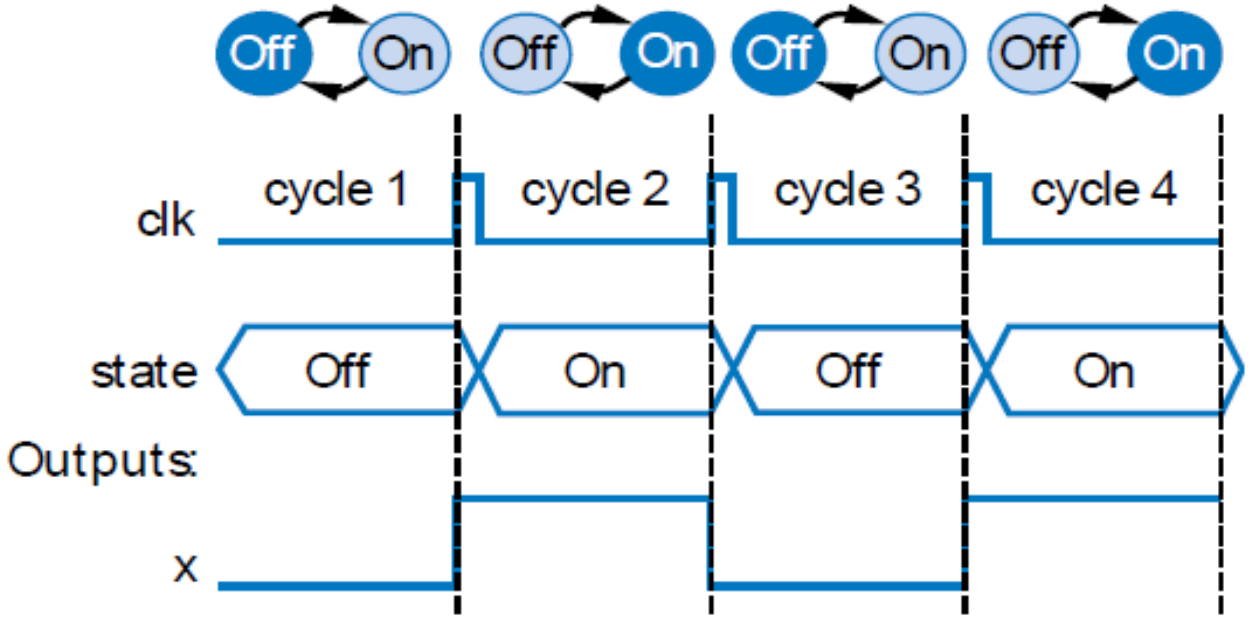


# Finite State Machine: Example

- Example: Make x change toggle (0 to 1, or 1 to 0) every clock cycle
- Two states: “Off” (x=0), and “On” (x=1)
- Transition from Off to On, or On to Off, on rising clock edge
- Arrow with no starting state points to initial state (when circuit first starts)

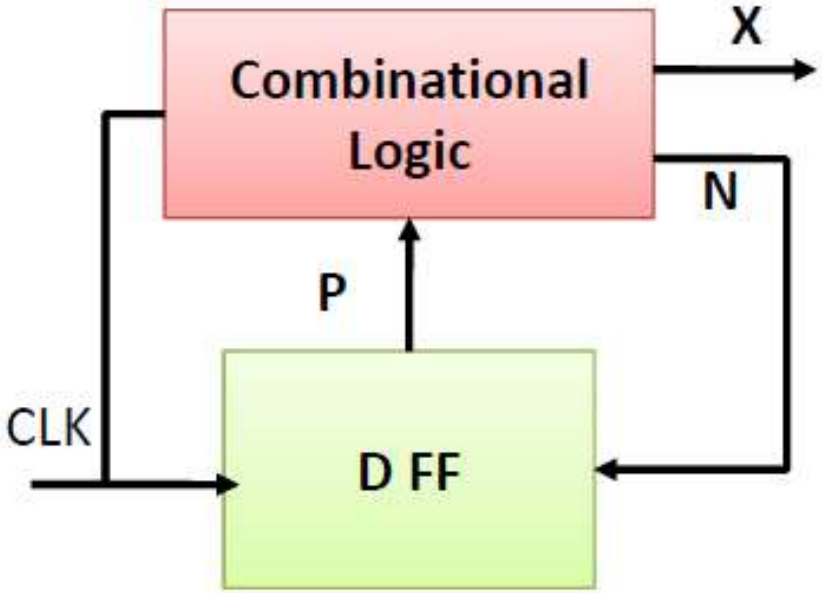
# Finite State Machine: Example

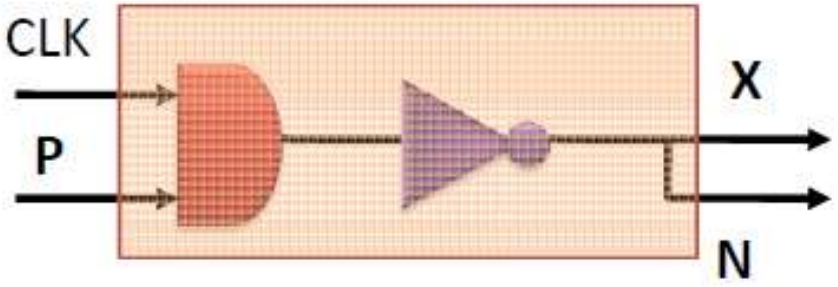


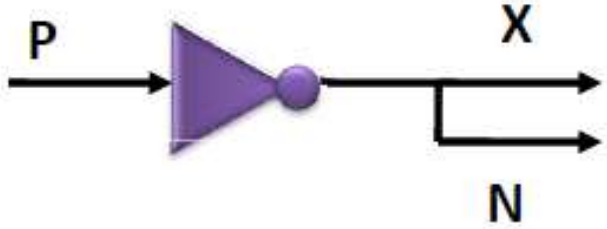
# Controller for On-Off

| Input | | Output | |
|---|---|---|---|
| CLK | P | X | N |
| RE 1 | 0 | 1 | 1 |
| RE 1 | 1 | 0 | 0 |



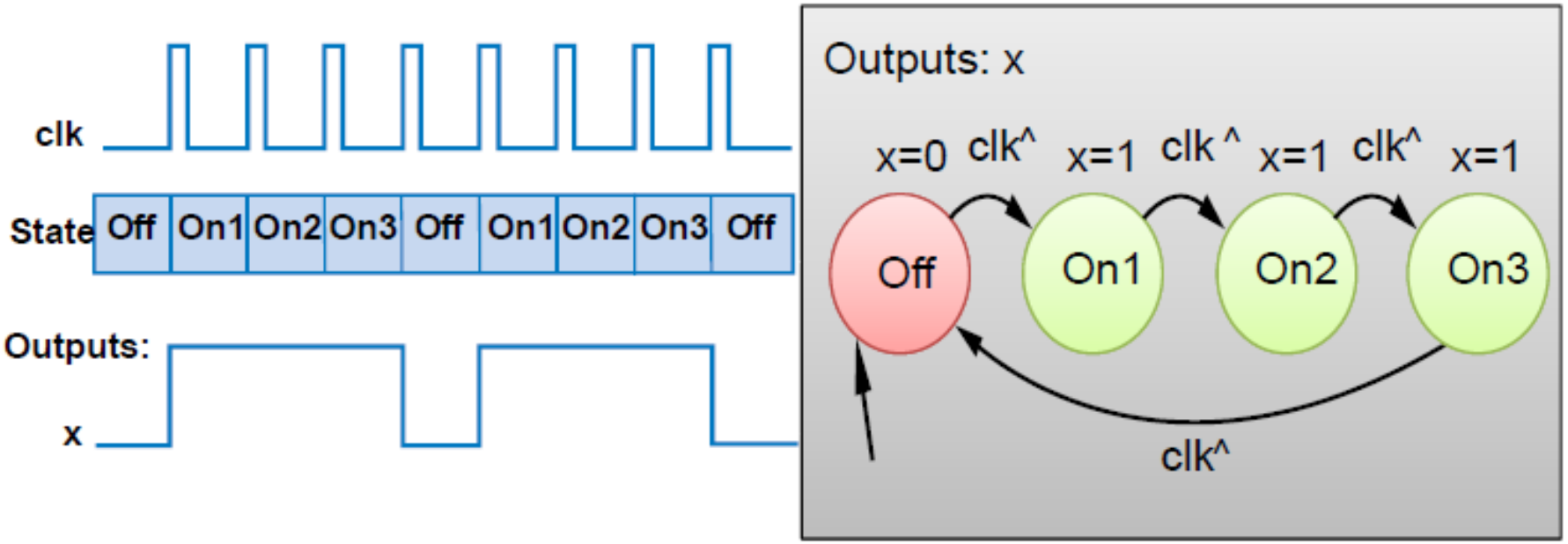

Think of Clock Enable: only **Rising Edge (RE)**
Above one may not work: Level Sensitive

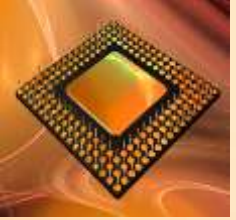



Rising Edge: Clock implicit

# FSM Example: 0,1,1,1,repeat

- Want 0, 1, 1, 1, 0, 1, 1, 1, …
  - Each value for one clock cycle
- Can describe as FSM: Four states, Transition on rising clock edge to next state

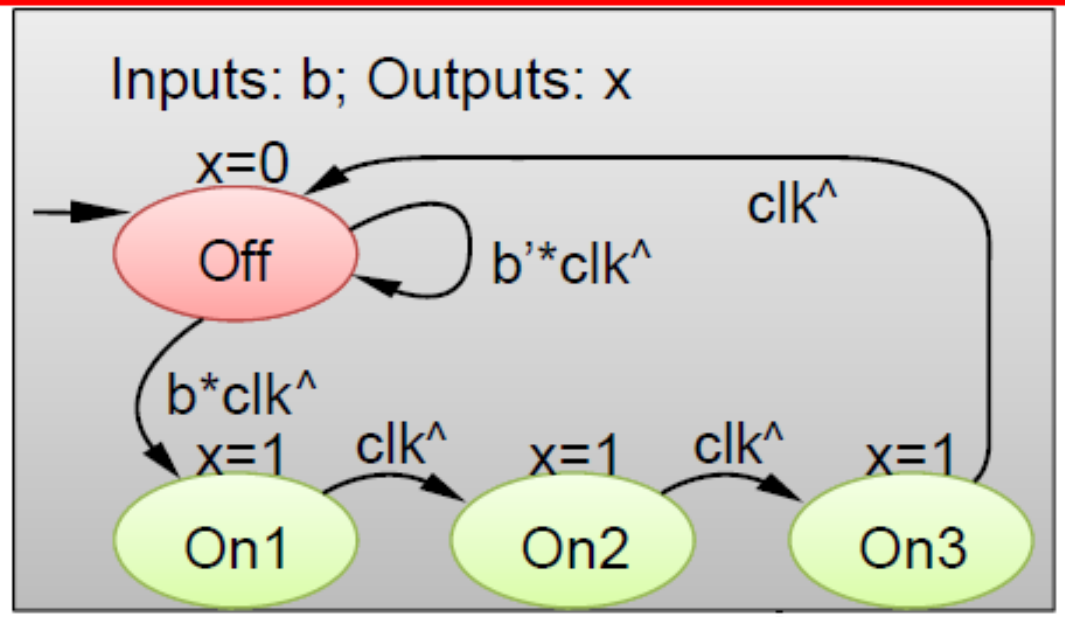


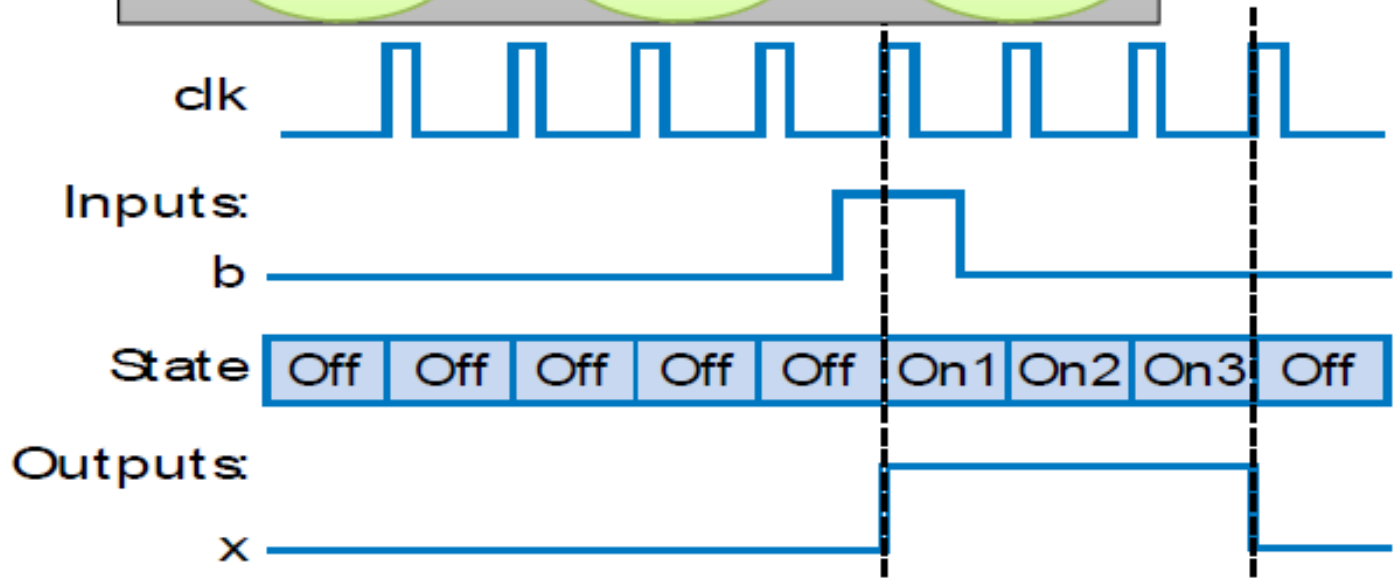


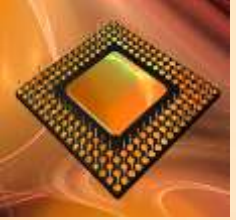
# Extend FSM to Three-Cycles High Laser Timer

- Four states: Wait in "Off" state while b is 0 (b')
- When b=1 (& rising clock edge), transition to On1
  - Sets X=1
  - On next two clock edges, transition to On2, then On3, which also set x=1
- So x=1 for three cycles after button pressed

# Extend FSM to Three-Cycles High Laser Timer



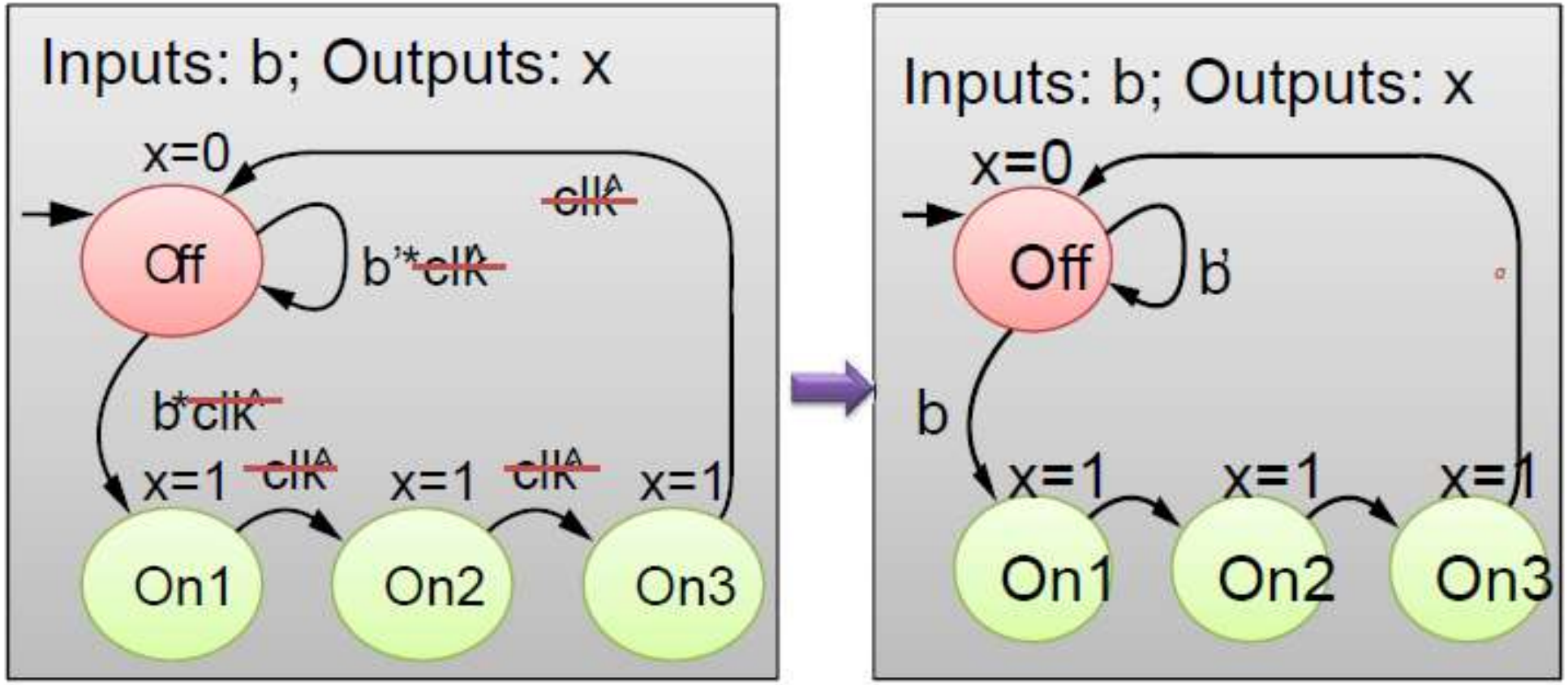





# FSM Simplification: Rising Clock Edges Implicit

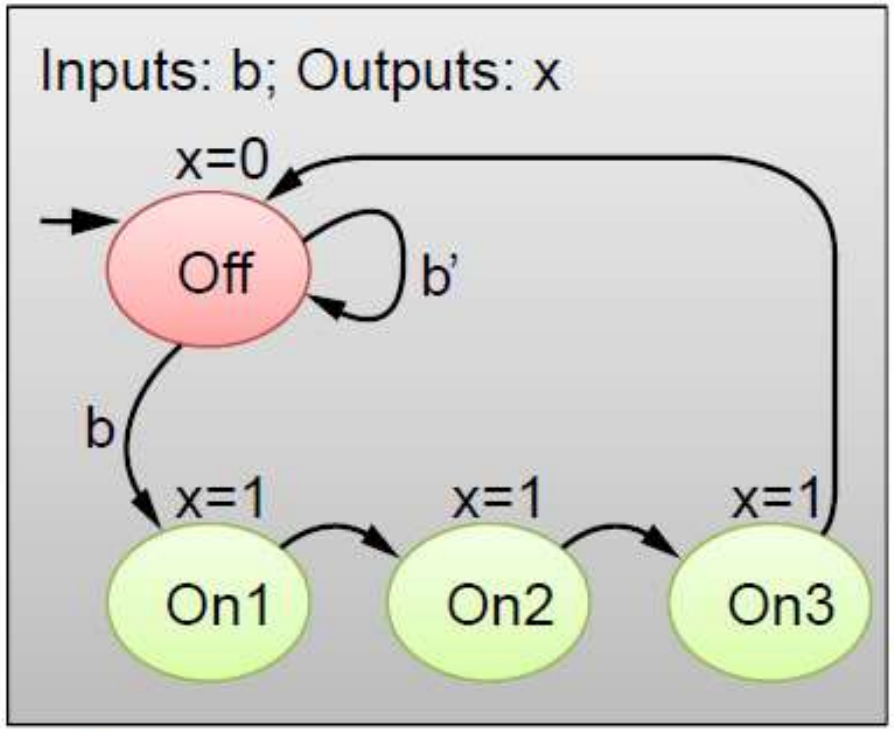
- Showing rising clock on every transition: cluttered
- Make implicit -- assume every edge has rising clock
- What if we wanted a transition *without* a rising edge
  - Asynchronous FSMs -- less common, and advanced topic
  - We consider ***synchronous*** FSMs
  - ***All*** **transition on rising edge**

# FSM Simplification: Rising Clock Edges Implicit



# FSM Definition

- **Set of states**
  - Ex: {Off, On1, On2, On3}
- **Set of inputs & set of outputs**
  - Ex: Inputs: {x}, Outputs: {b}
- **Initial state**
  - Ex: "Off"
- **Set of transitions**
  - Describes next states, Ex: Has 5 transitions
- **Set of actions**
  - Sets outputs while in states
  - Ex: x=0, x=1, x=1, and x=1

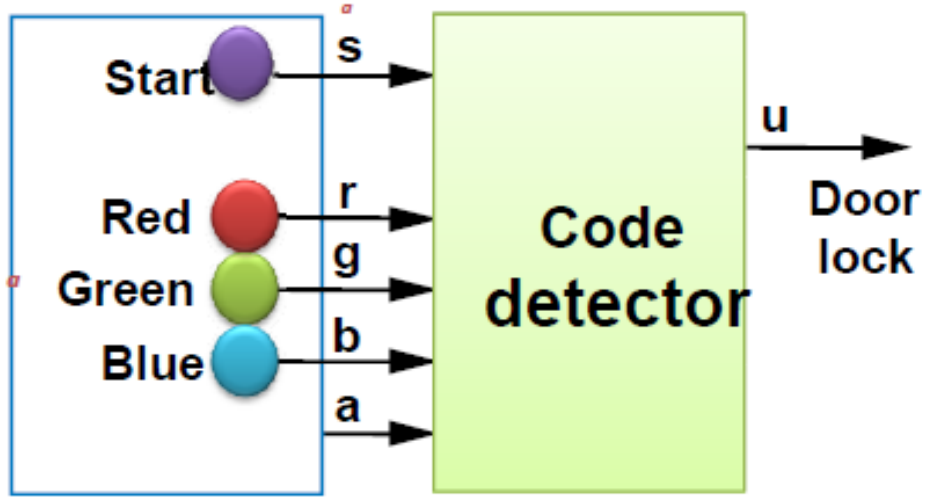


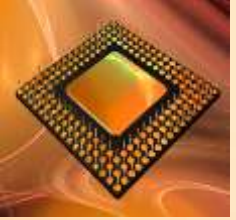

We often draw FSM graphically, known as ***state diagram***

Can also use table (state table), or textual languages

# FSM Example: Code Detector

- Unlock door (u=1) only when buttons pressed in sequence:
  - **start, then red, blue, green, red**
- Input from each button: *s, r, g, b*
  - Also, output *a* indicates that some colored button pressed

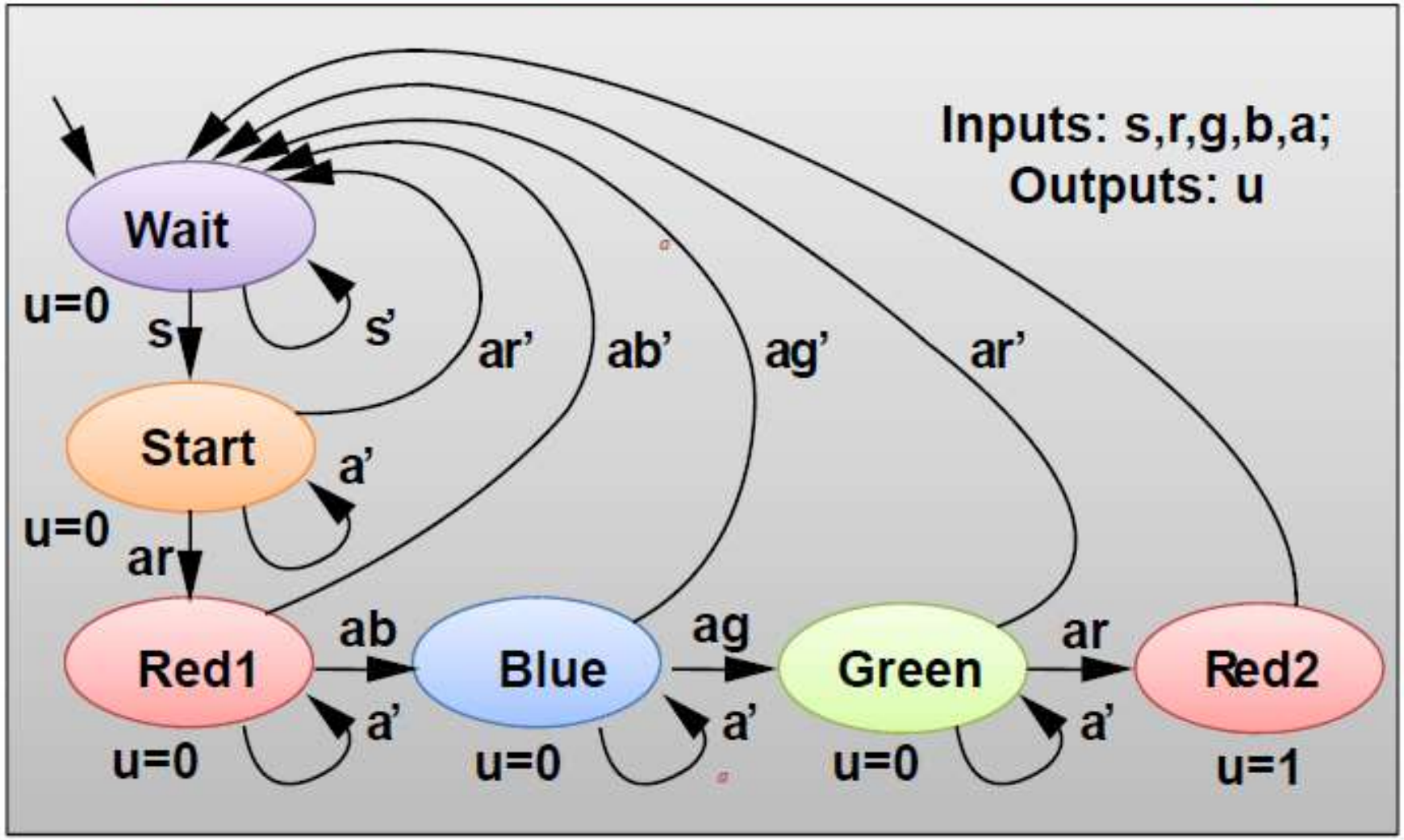




# FSM Example: Code Detector

- Wait for start (s=1) in “Wait”,
- **Once started (“Start”)**
  - If see red, go to “Red1”
  - Then, if see blue, go to “Blue”, Then, if see green, go to “Green”, Then, if see red, go to “Red2”
  - In that state, open the door (u=1)
  - Wrong button at any step, return to “Wait”

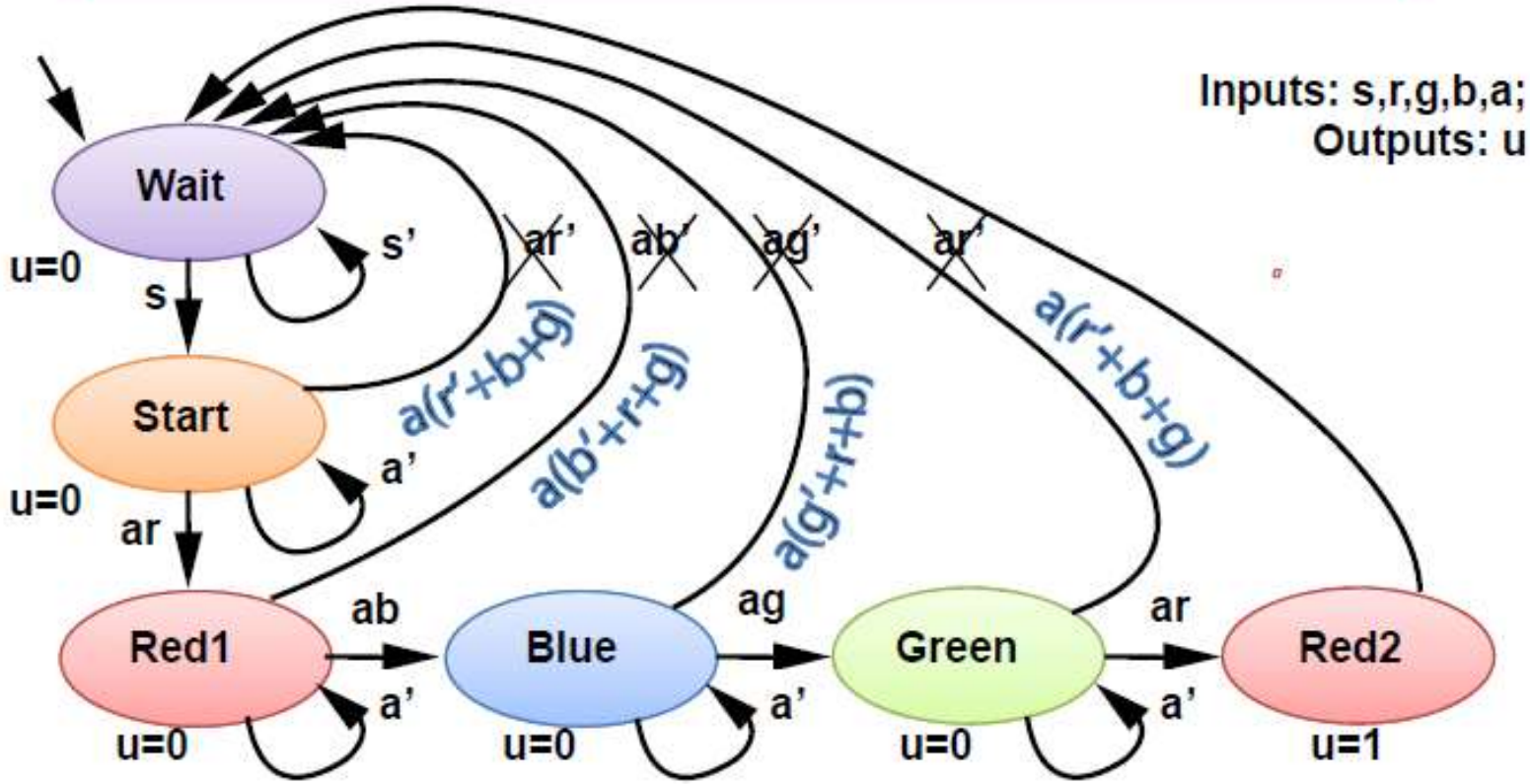
# FSM Example: Code Detector




Q: Can you trick this FSM to open the door, without knowing the code?

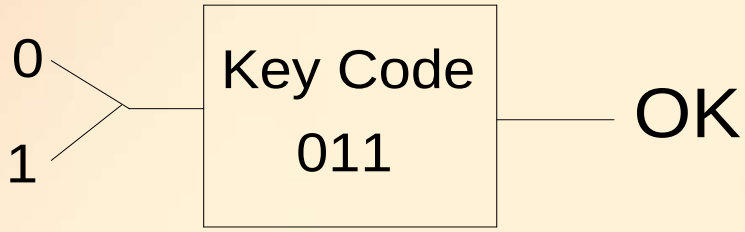
A: Yes, hold all buttons simultaneously

# Improve FSM for Code Detector



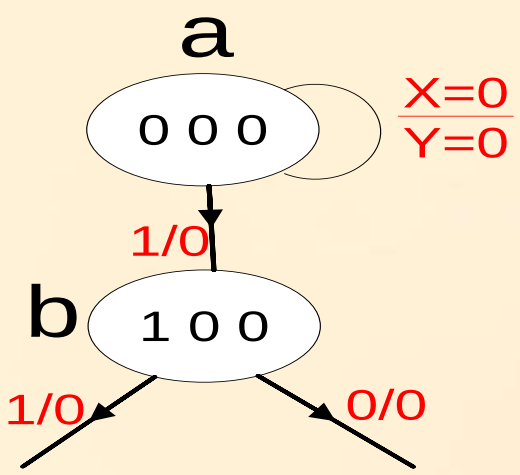

- New transition conditions detect if wrong button pressed, returns to "Wait"
- FSM provides formal, concrete means to accurately define desired behavior

# Finite State Machines : Design of digital lock ( 011)

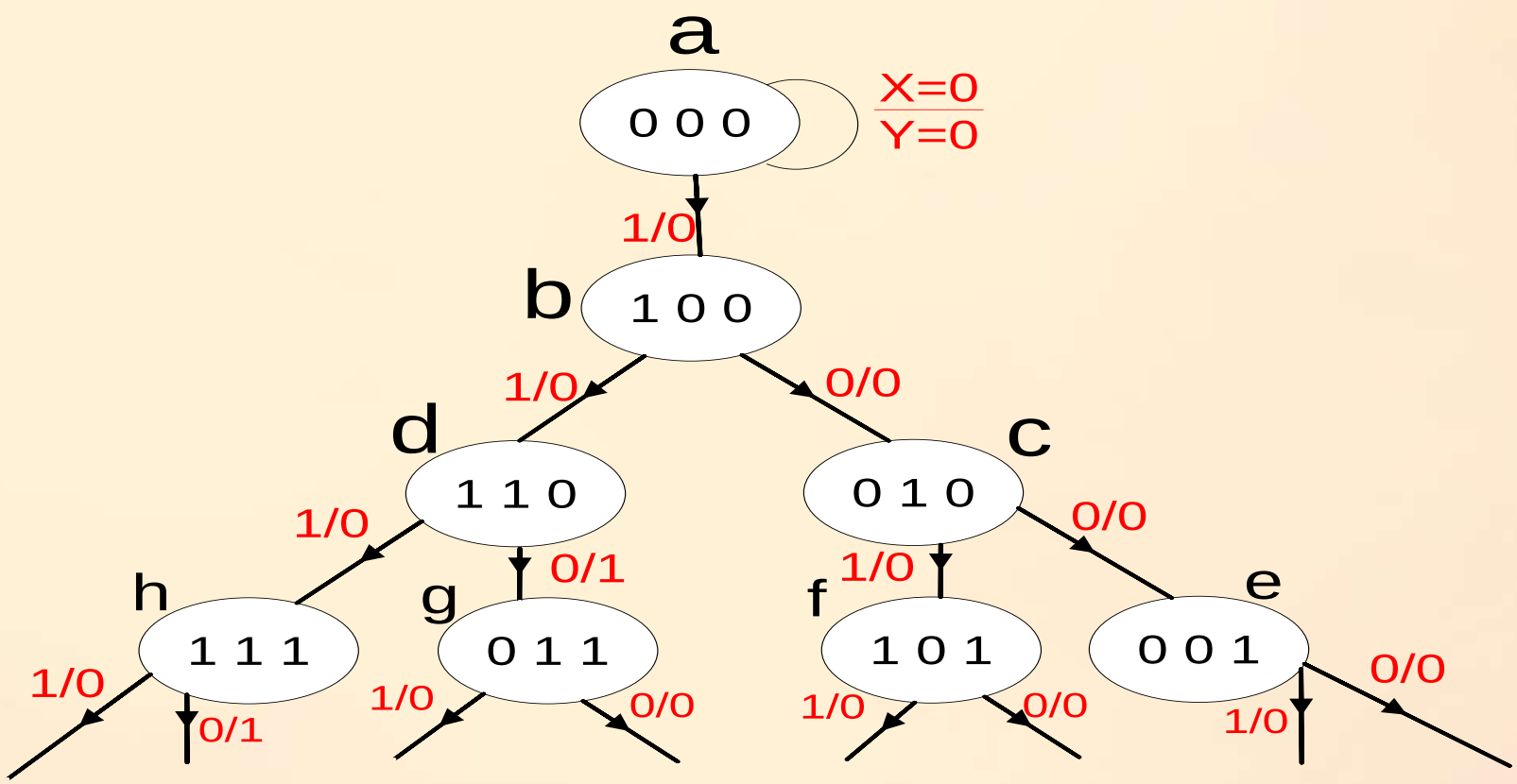



$$2^3 = 8$$

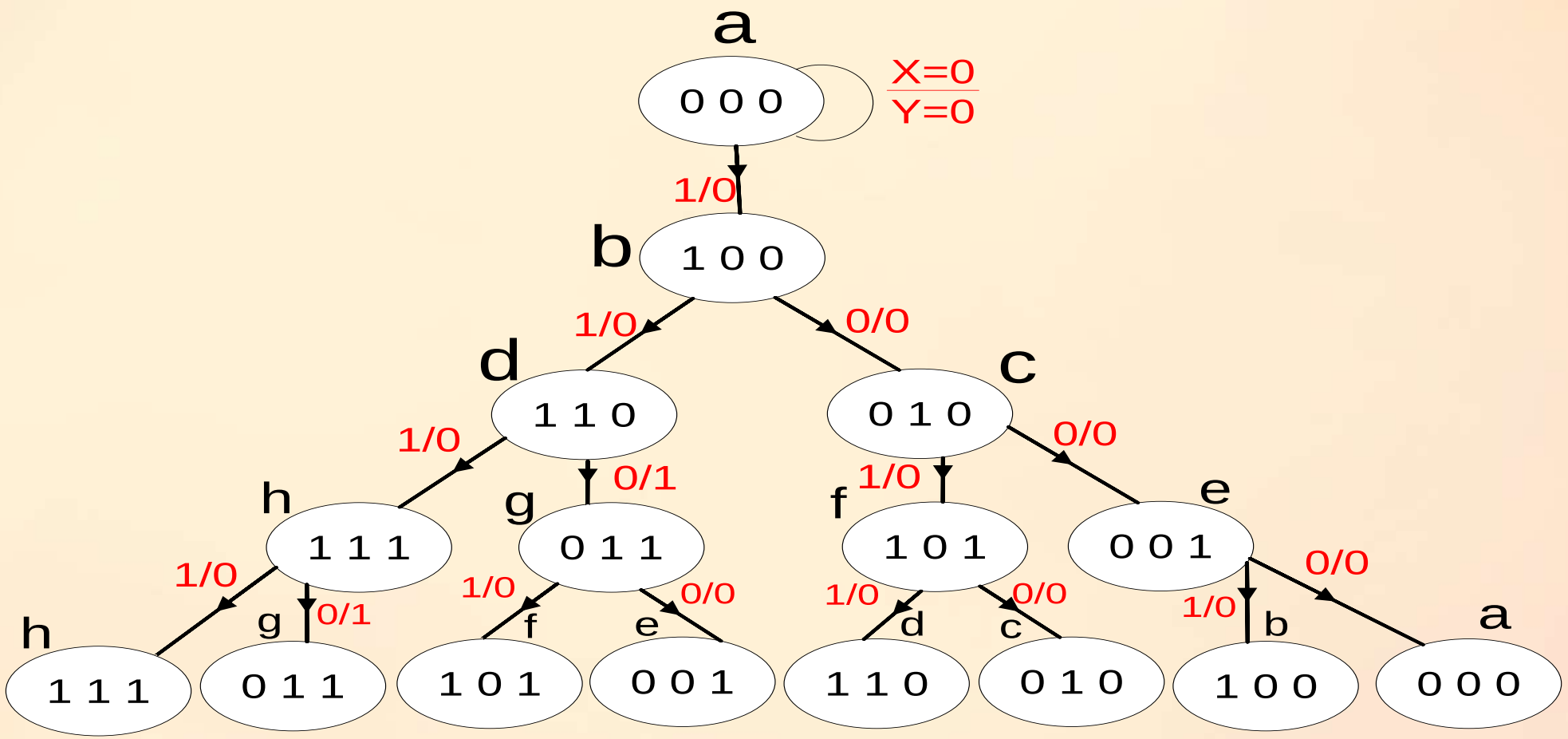
# State Machines : **State Diagram**



# State Machines : **State Diagram**

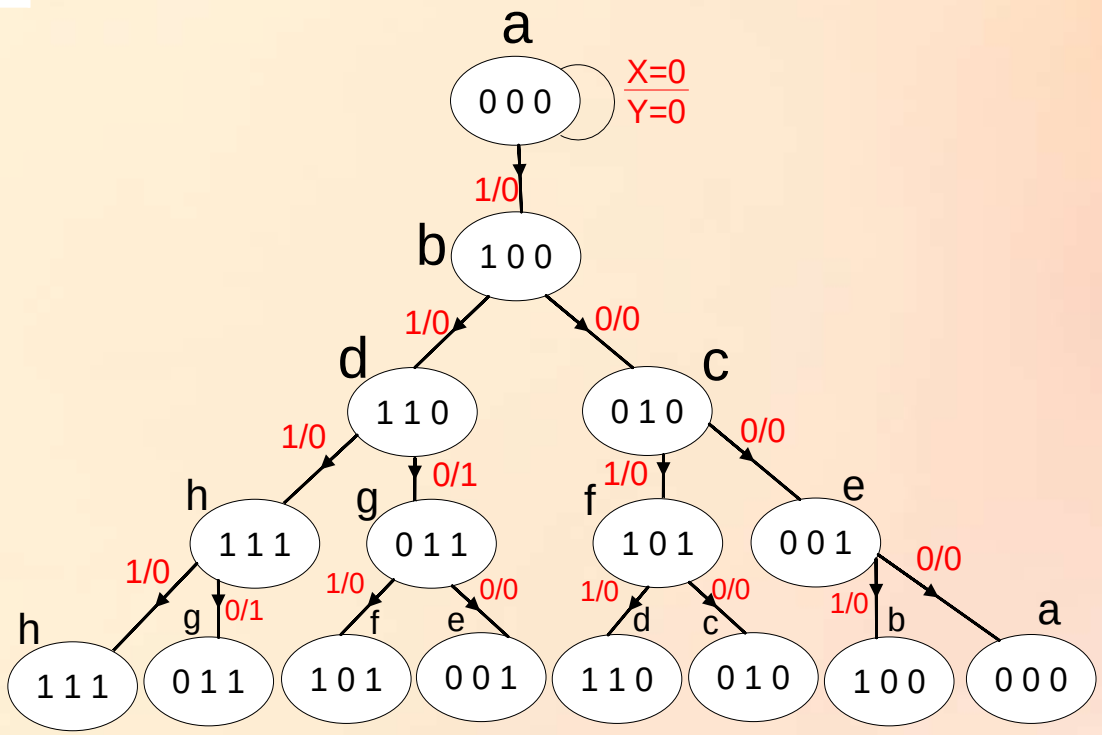


# State Machines : **State Diagram**



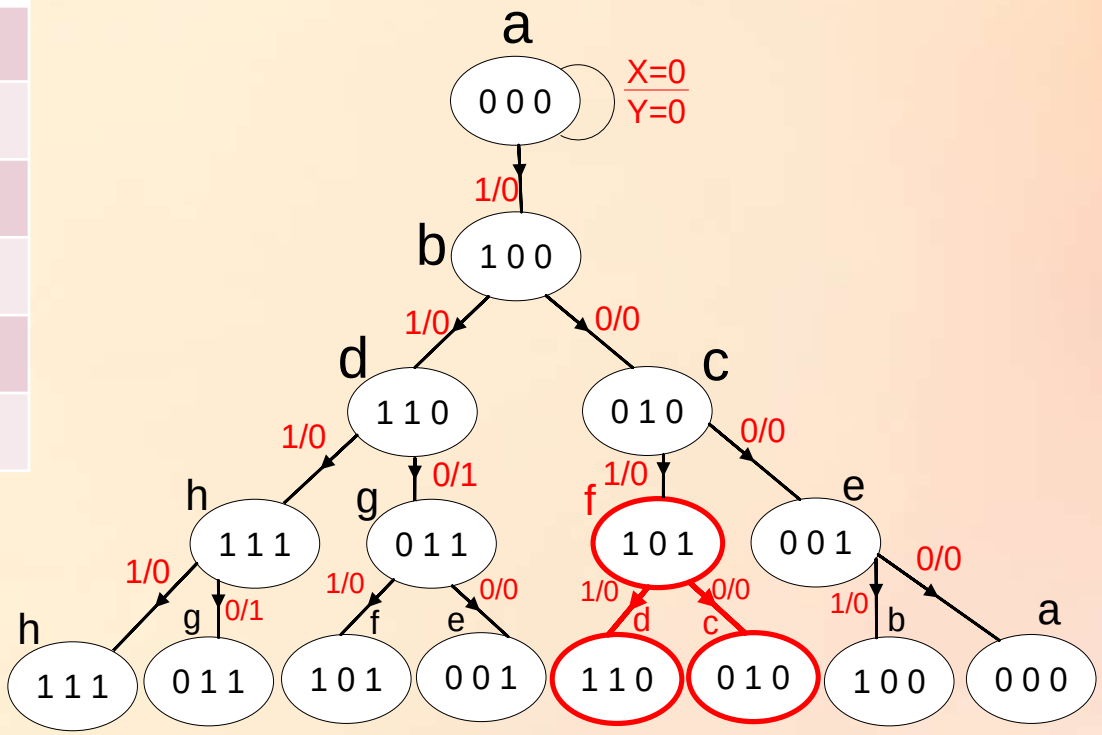
# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |



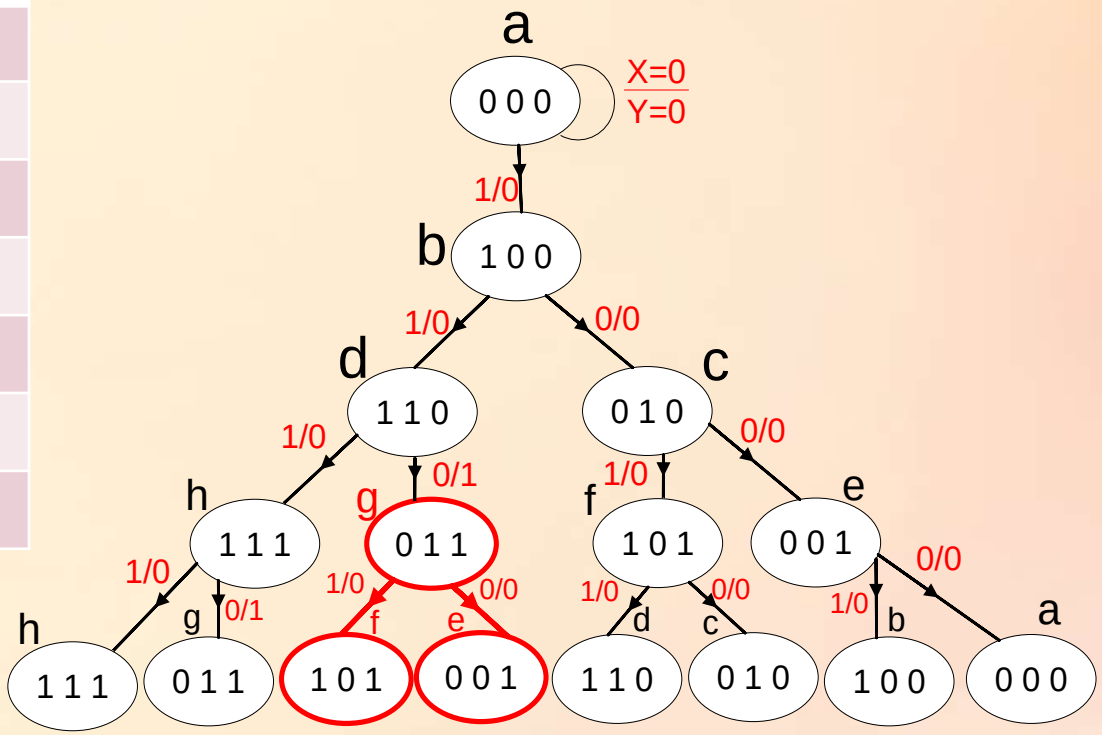
# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| f | c | d | 0 | 0 |



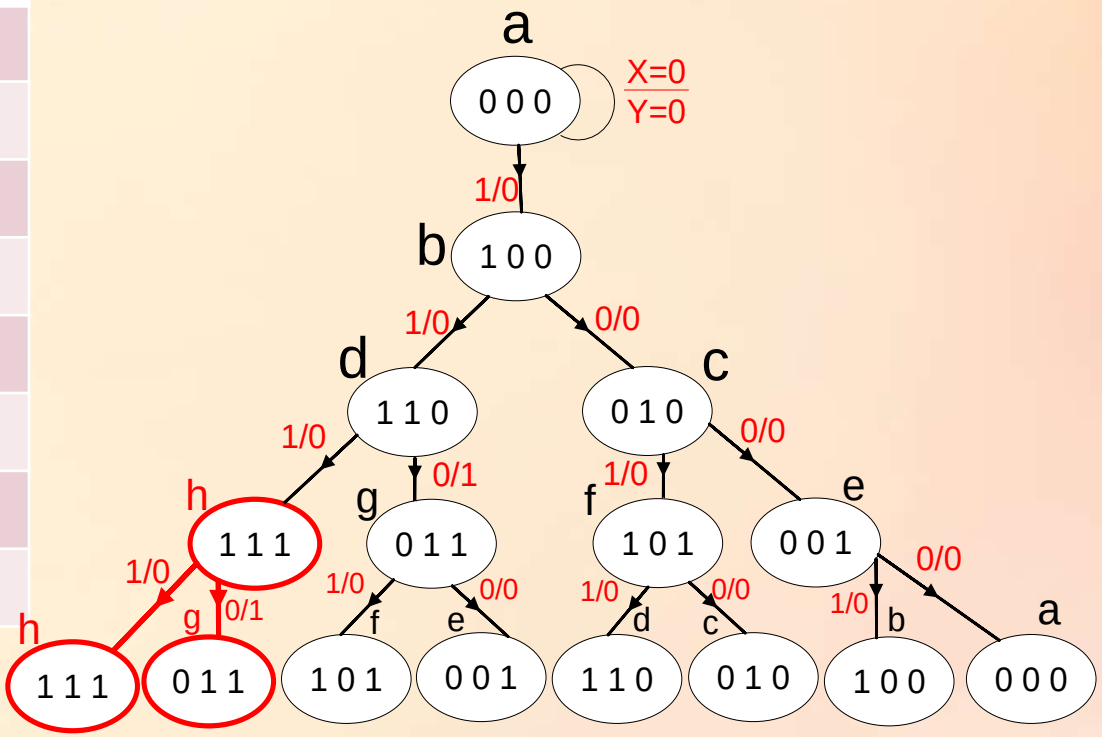
# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| f | c | d | 0 | 0 |
| g | e | f | 0 | 0 |



# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| f | c | d | 0 | 0 |
| g | e | f | 0 | 0 |
| h | g | h | 1 | 0 |



# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| f | c | d | 0 | 0 |
| g | e | f | 0 | 0 |
| h | g | h | 1 | 0 |

b←

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| g | e | f | 0 | 0 |
| h | g | h | 1 | 0 |

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| g | e | f | 0 | 0 |
| h | g | h | 1 | 0 |

c ←

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| h | g | h | 1 | 0 |

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |
| h | g | h | 1 | 0 |

d ←

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |
| e | a | b | 0 | 0 |

a ←

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | f | 0 | 0 |
| d | g | h | 1 | 0 |

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | **f** | 0 | 0 |
| d | g | h | 1 | 0 |

$$f \rightarrow b$$

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | b | 0 | 0 |
| d | **g** | h | 1 | 0 |

$$g \rightarrow c$$

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | b | 0 | 0 |
| d | c | **h** | 1 | 0 |

$$h \rightarrow d$$

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | e | b | 0 | 0 |
| d | c | d | 1 | 0 |

$e \rightarrow a$

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | a | b | 0 | 0 |
| d | c | d | 1 | 0 |

# State Machines : State Diagram

| n | n+1 x=0 | n+1 x=1 | (y)Out put x=0 | (y)Out put x=1 |
|---|---|---|---|---|
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | a | b | 0 | 0 |
| d | c | d | 1 | 0 |

a ←

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| d | c | d | 1 | 0 |

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | **c** | d | 0 | 0 |
| d | **c** | d | 1 | 0 |

$$c \longrightarrow a$$

# State Machines : **State Diagram**

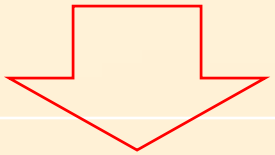
| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | a | d | 0 | 0 |
| d | a | d | 1 | 0 |

# State Machines : **State Diagram**

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | a | d | 0 | 0 |
| d | a | d | 1 | 0 |

# State Machines : State Diagram

| n | n+1 | | (y)Out put | |
|---|---|---|---|---|
| | x=0 | x=1 | x=0 | x=1 |
| a | a | b | 0 | 0 |
| b | a | d | 0 | 0 |
| d | a | d | 1 | 0 |



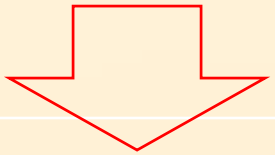
| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

# State Machines : **State Diagram**

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

# State Machines : State Diagram

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |



| | | x=0 | x=1 | x=0 | x=1 |
|---|---|---|---|---|---|
| A | B | AB | AB | y | |
| 0 | 0 | 00 | 01 | 0 | 0 |
| 0 | 1 | 00 | 10 | 0 | 0 |
| 1 | 0 | 00 | 10 | 1 | 0 |

# State Machines : State Diagram

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

| | | x=0 | x=1 | x=0 | x=1 |
|---|---|---|---|---|---|
| A | B | AB | AB | y | |
| 0 | 0 | 00 | 01 | 0 | 0 |
| 0 | 1 | 00 | 10 | 0 | 0 |
| 1 | 0 | 00 | 10 | 1 | 0 |

$$J_B = \overset{X\ A\ B}{100}$$

# State Machines : State Diagram

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

| | | x=0 | x=1 | x=0 | x=1 |
|---|---|---|---|---|---|
| A | B | AB | AB | y | |
| 0 | 0 | 00 | 01 | 0 | 0 |
| 0 | 1 | 00 | 10 | 0 | 0 |
| 1 | 0 | 00 | 10 | 1 | 0 |

$$K_B = \overset{X\ A\ B}{001+101}$$

# State Machines : State Diagram

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

| | | x=0 | x=1 | x=0 | x=1 |
|---|---|---|---|---|---|
| A | B | AB | AB | y | |
| 0 | 0 | 00 | 01 | 0 | 0 |
| 0 | 1 | 00 | 10 | 0 | 0 |
| 1 | 0 | 00 | 10 | 1 | 0 |

$$J_A = \begin{matrix} X & A & B \\ 1 & 0 & 1 \end{matrix}$$

# State Machines : State Diagram

| | A | B |
|---|---|---|
| a | 0 | 0 |
| b | 0 | 1 |
| d | 1 | 0 |
| X | 1 | 1 |

| | | x=0 | x=1 | x=0 | x=1 |
|---|---|---|---|---|---|
| A | B | AB | AB | y | |
| 0 | 0 | 00 | 01 | 0 | 0 |
| 0 | 1 | 00 | 10 | 0 | 0 |
| 1 | 0 | 00 | 10 | 1 | 0 |

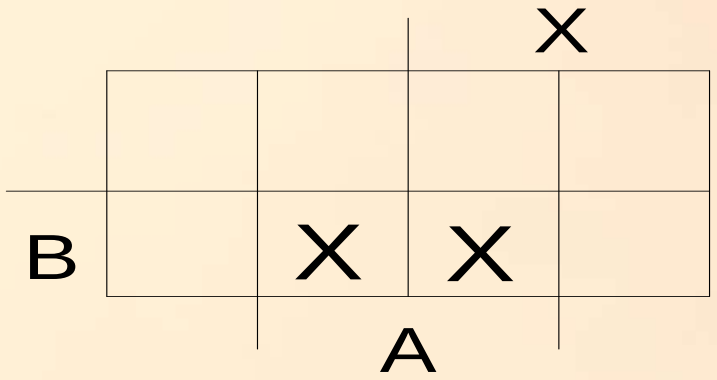
$$K_A = \overset{X\ A\ B}{010}$$

# State Machines : **State Diagram**

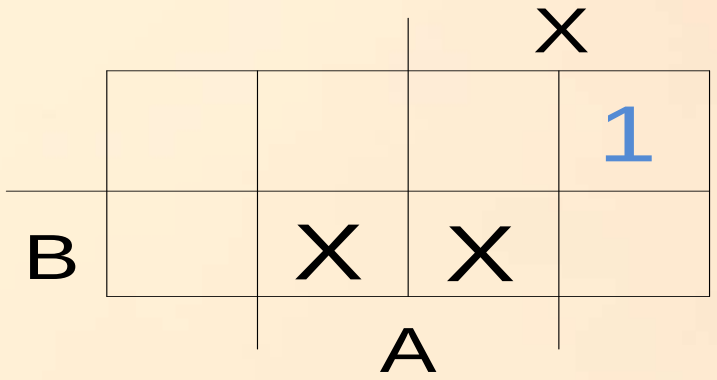
| | | X | | |
|---|---|---|---|---|
| | 0 | 2 | 6 | 4 |
| B | 1 | 3 | 7 | 5 |
| | | A | | |

| | | X | | |
|---|---|---|---|---|
| | | | | |
| B | | X | X | |
| | | A | | |

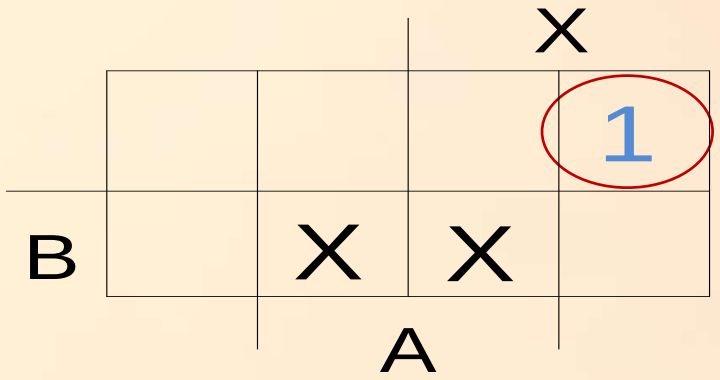
$$J_B = 100$$



$$J_B = 100$$

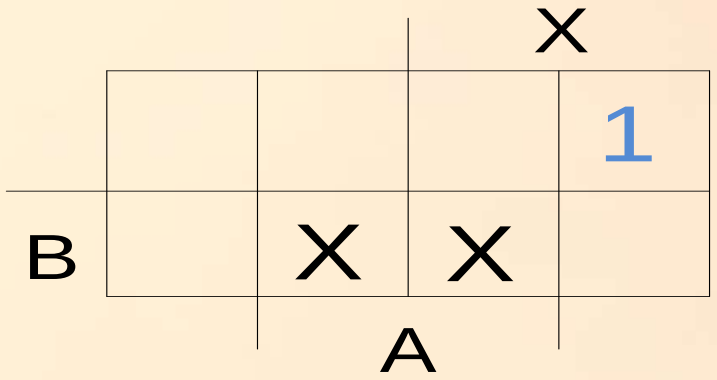


$$J_B = 100 = \overline{A}\,\overline{B}X$$



$$J_B = 100 = \overline{A}\,\overline{B}X$$

$$K_B = 001 + 101$$



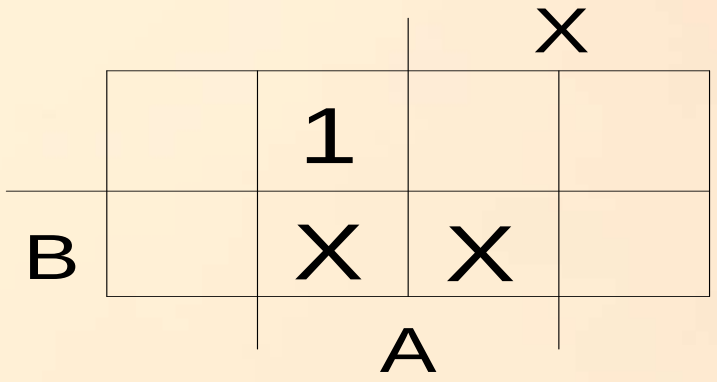
$$J_B = 100 = \overline{A}\,\overline{B}X$$

$$K_B = 001 + 101 = B$$

$$J_A = 101 = BX$$

$$K_A = 010 = A\overline{X}$$

$$Y = 010$$



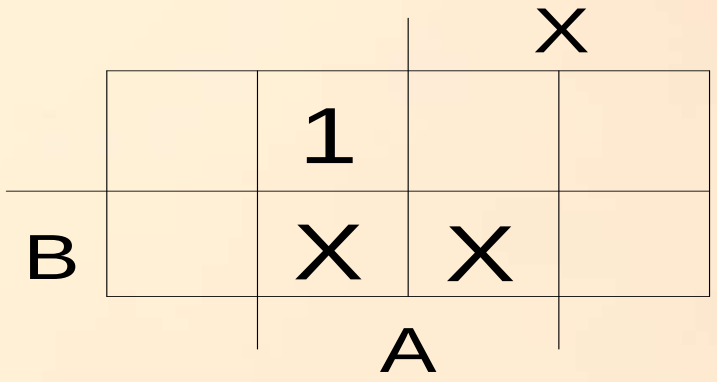
$$J_B = 100 = \overline{A}\,\overline{B}X$$

$$K_B = 001 + 101 = B$$

$$J_A = 101 = BX$$

$$K_A = 010 = A\overline{X}$$

$$Y = 010$$



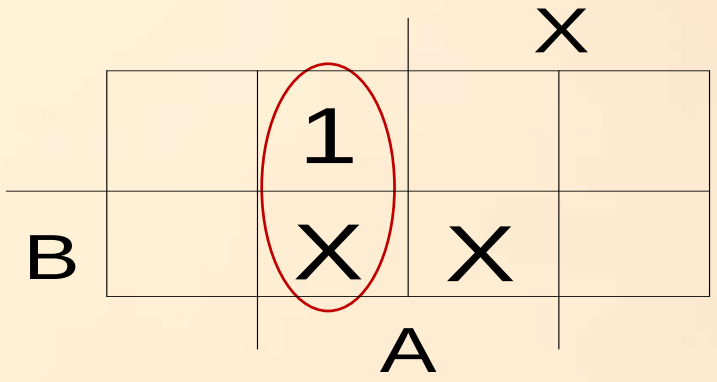
$$J_B = 100 = \overline{A}\,\overline{B}X$$

$$K_B = 001 + 101 = B$$

$$J_A = 101 = BX$$

$$K_A = 010 = A\overline{X}$$

$$Y = 010 = A\overline{X}$$



$$J_B = 100 = \overline{A}\,\overline{B}X$$

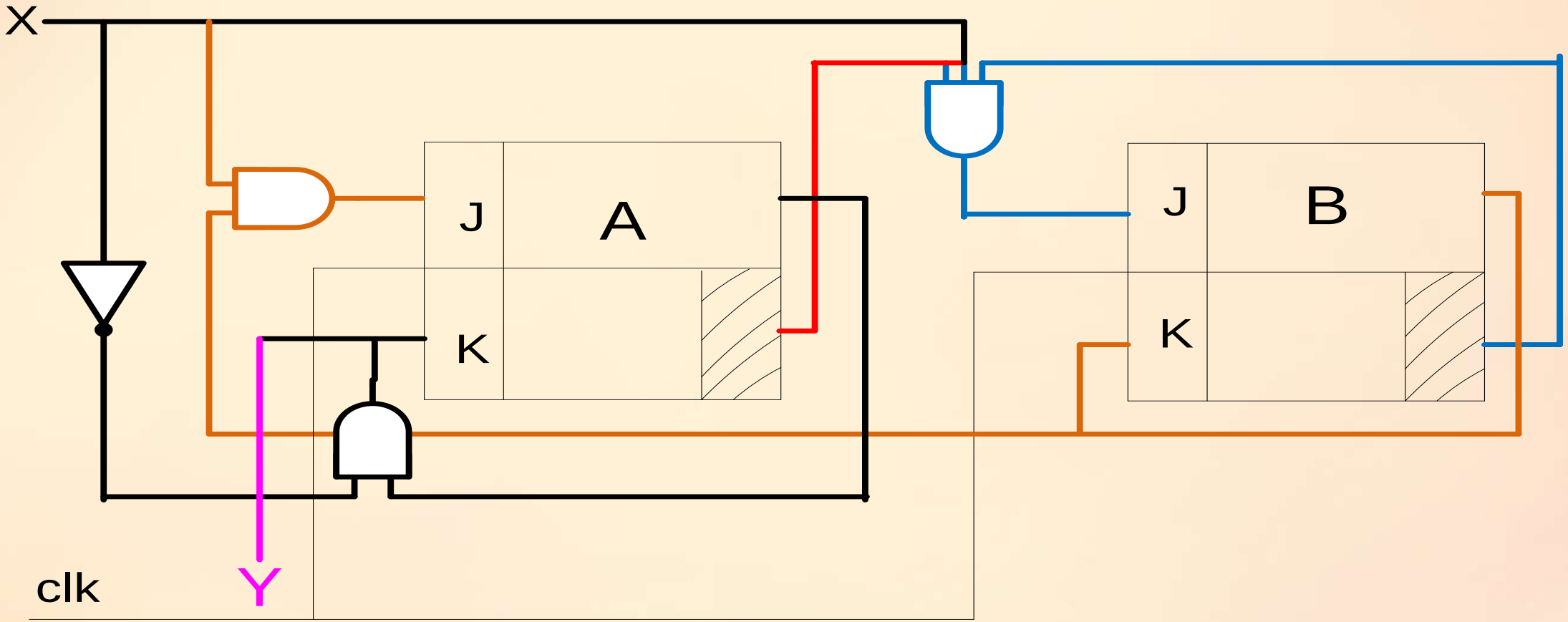
$$K_B = 001 + 101 = B$$

$$J_A = 101 = BX$$

$$K_A = 010 = A\overline{X}$$

$$Y = 010 = A\overline{X}$$

# State Machines : **State Diagram**



$$J_B = 100 = \overline{A}\,\overline{B}X = \overline{A}X$$

$$K_B = 001 + 101 = B = 1$$

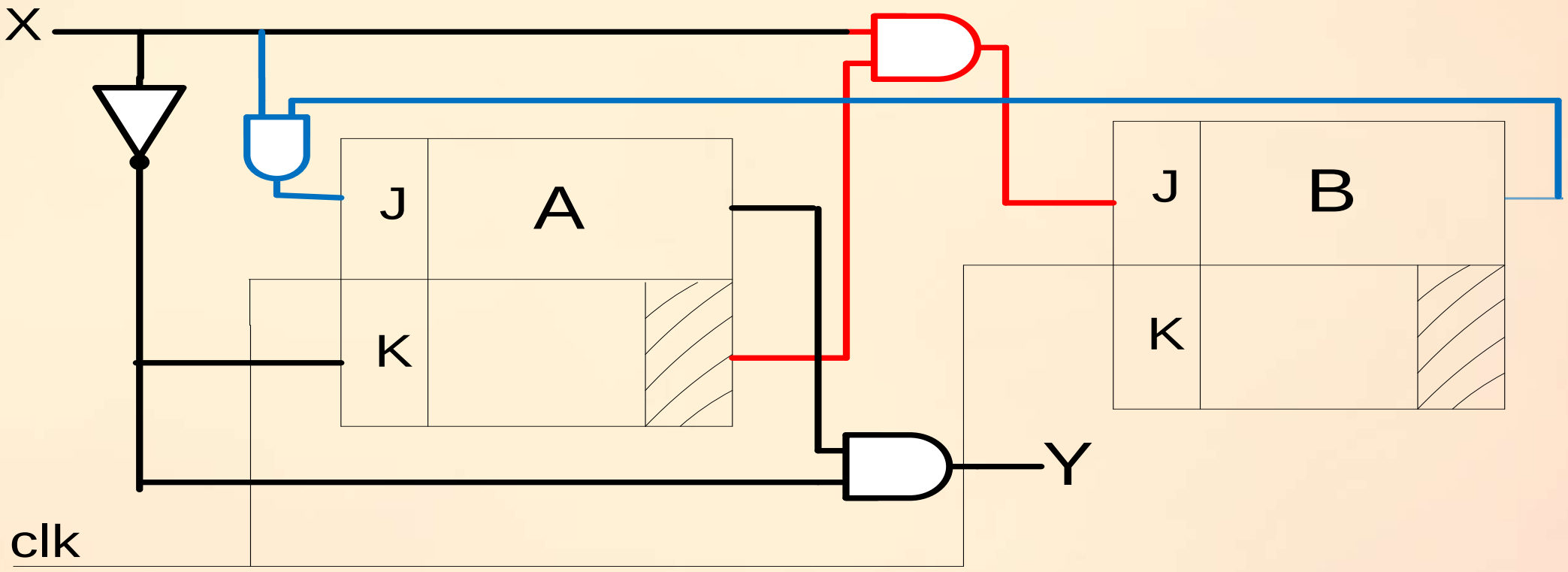
$$J_A = 101 = BX$$

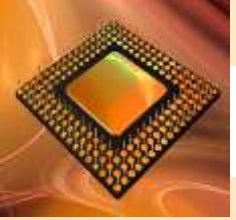
$$K_A = 010 = A\overline{X} = \overline{X}$$

$$Y = 010 = A\overline{X}$$

With simplification

# State Machines : **State Diagram**

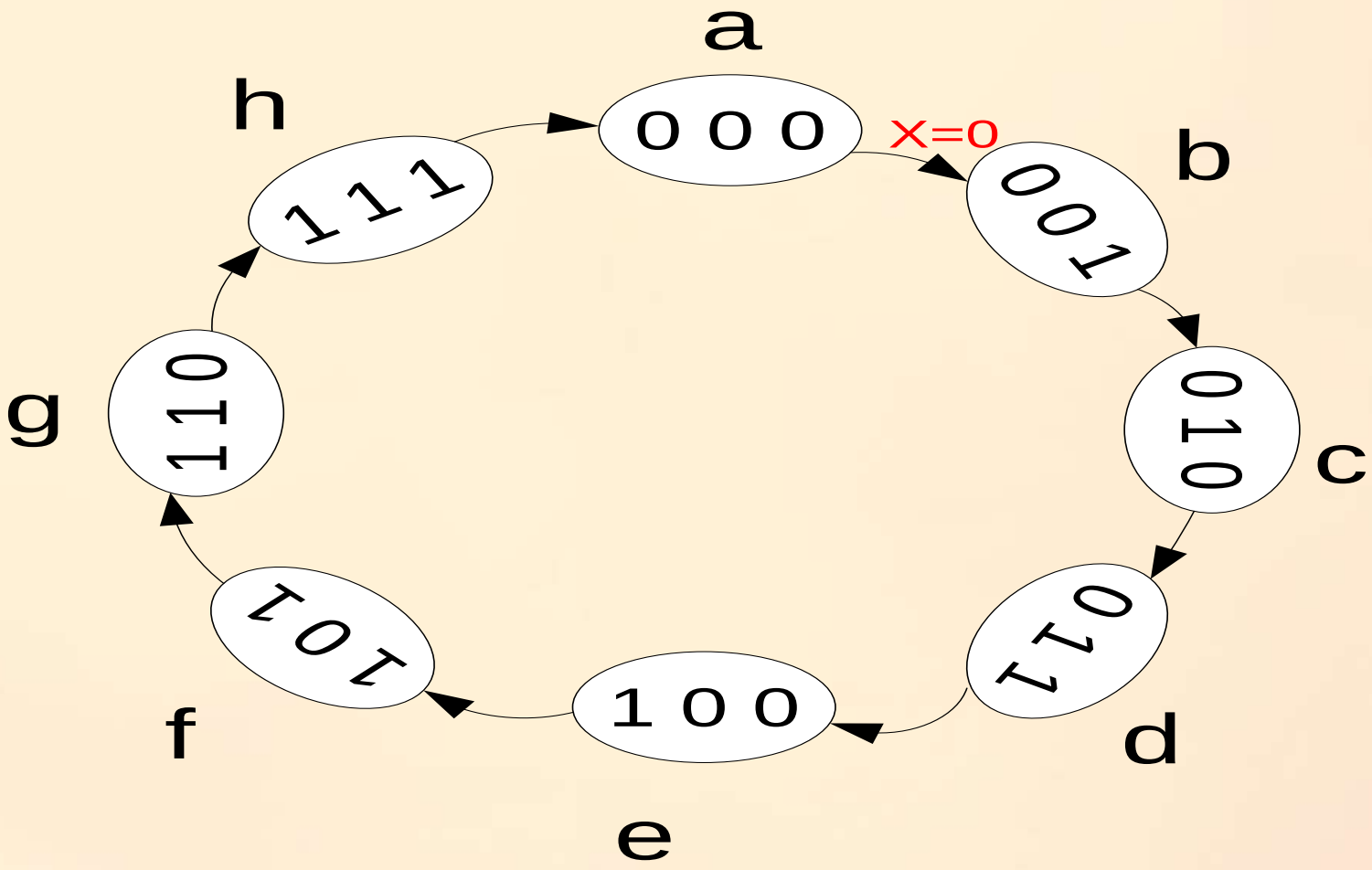




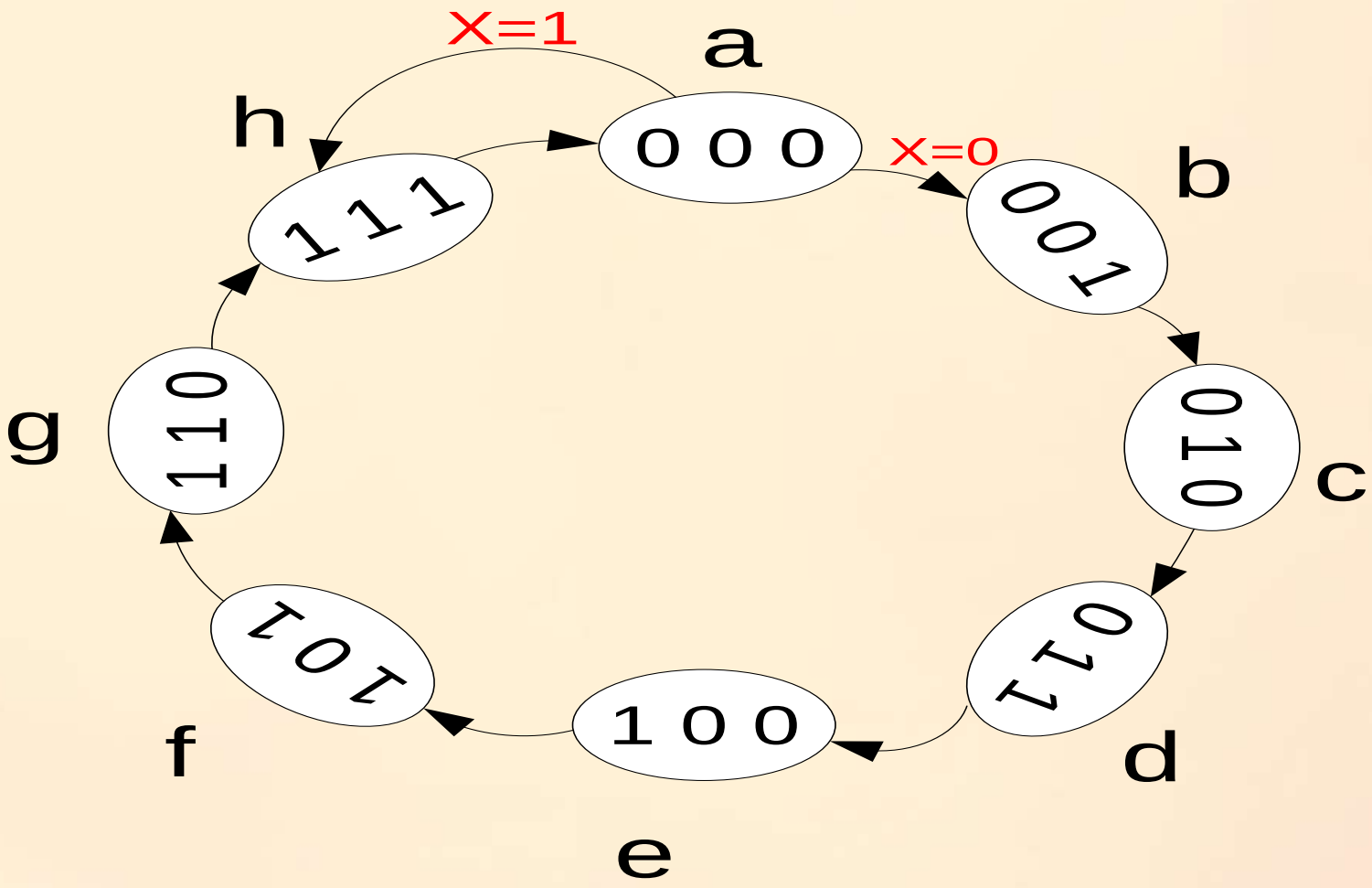
# State Machines : **State Diagram**

**Up-Down counter with input controller :**
**If x="0" then count up counter.**
**If x="1" then count down counter.**

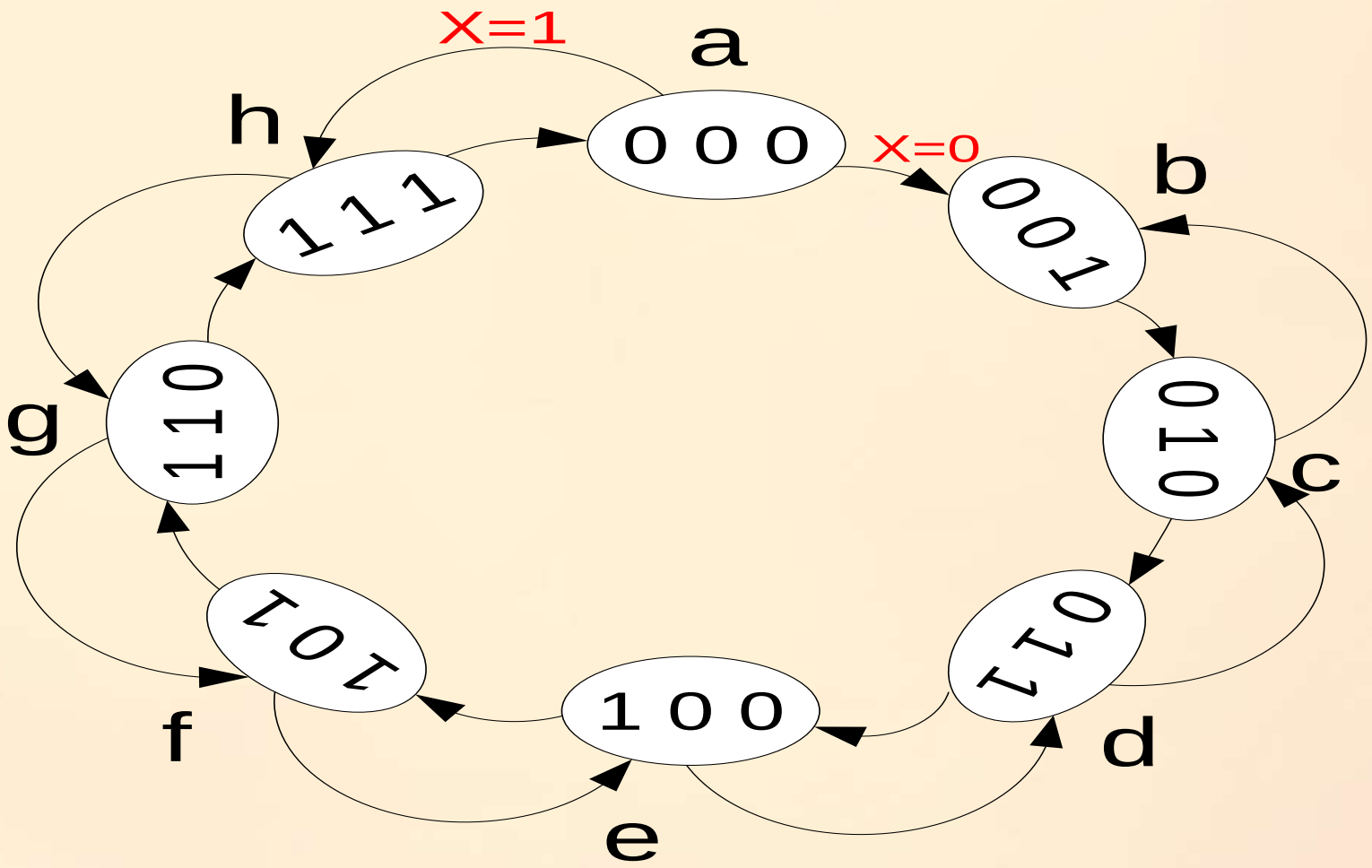
# State Machines : **State Diagram**



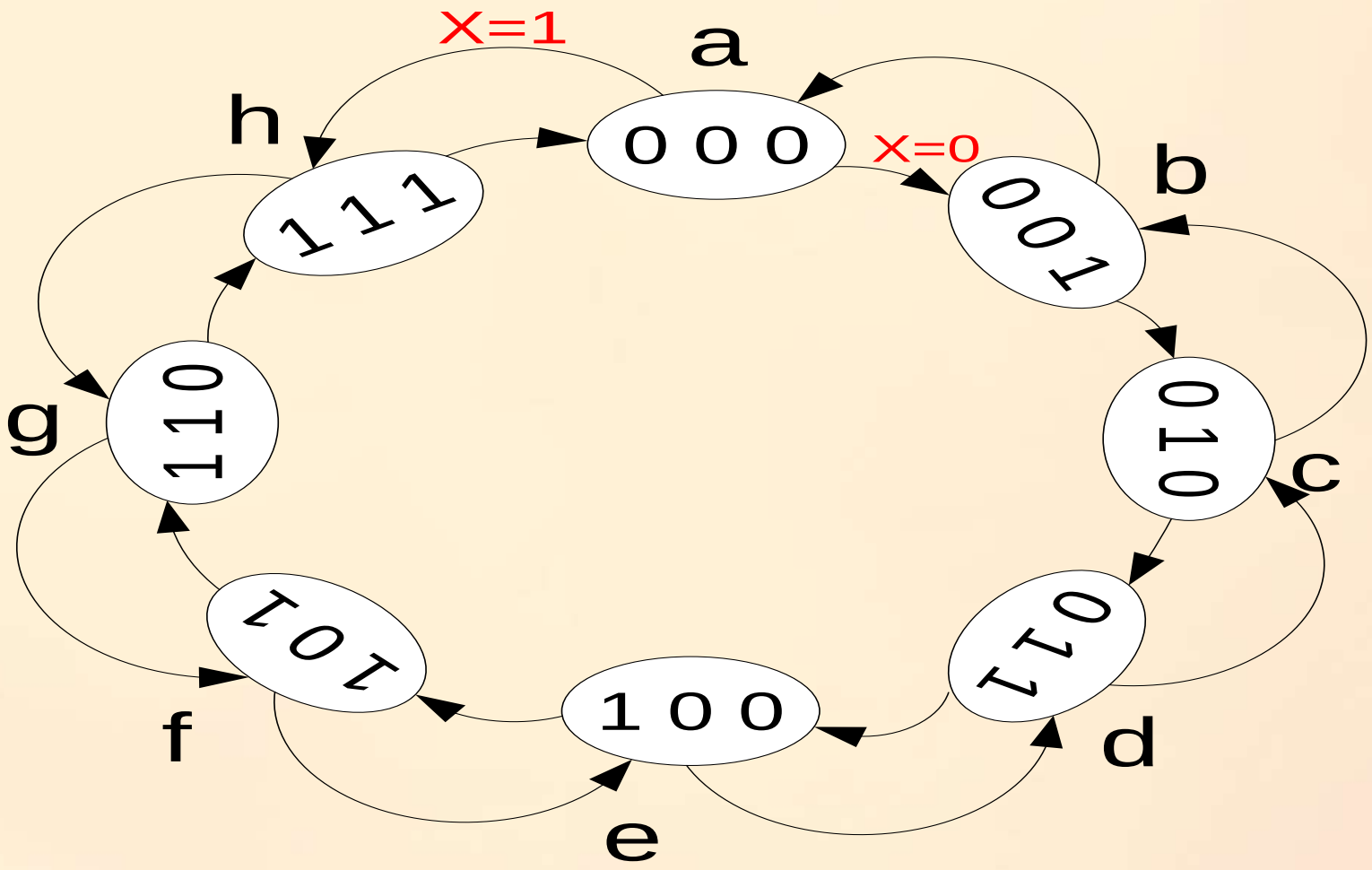
# State Machines : **State Diagram**



# State Machines : **State Diagram**



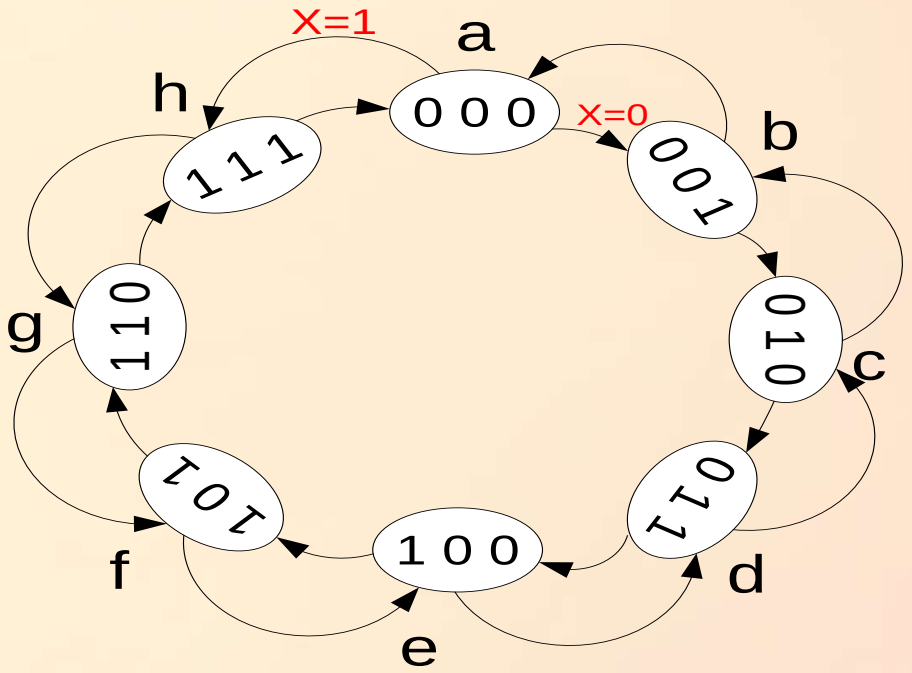
# State Machines : **State Diagram**



# State Machines : **State Diagram**

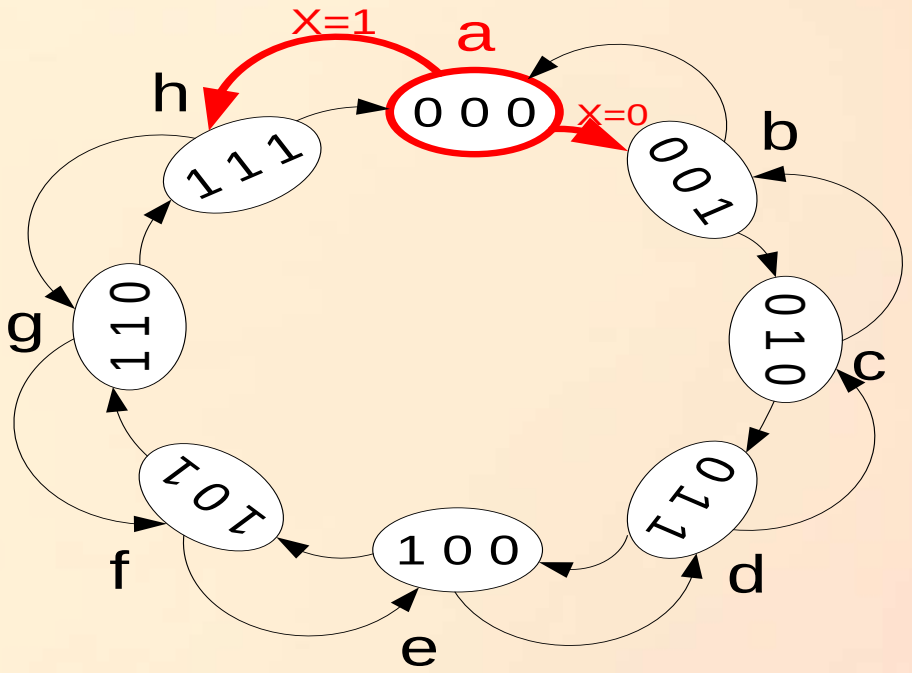
## State Table

| | x=0 | x=1 |
|---|---|---|



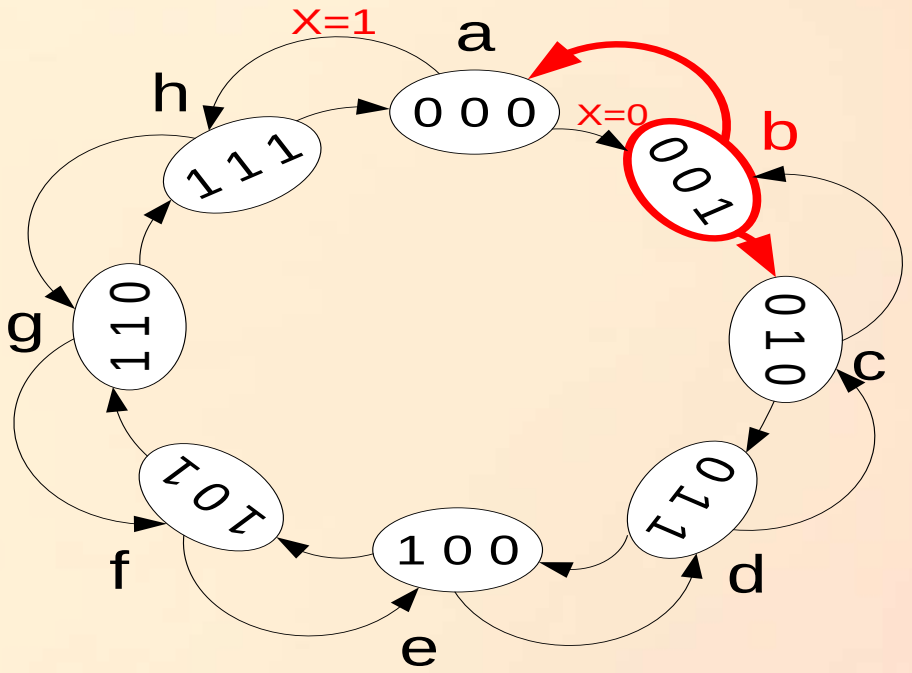
## State Table

| | x=0 | x=1 |
|---|---|---|
| a | b | h |



## State Table

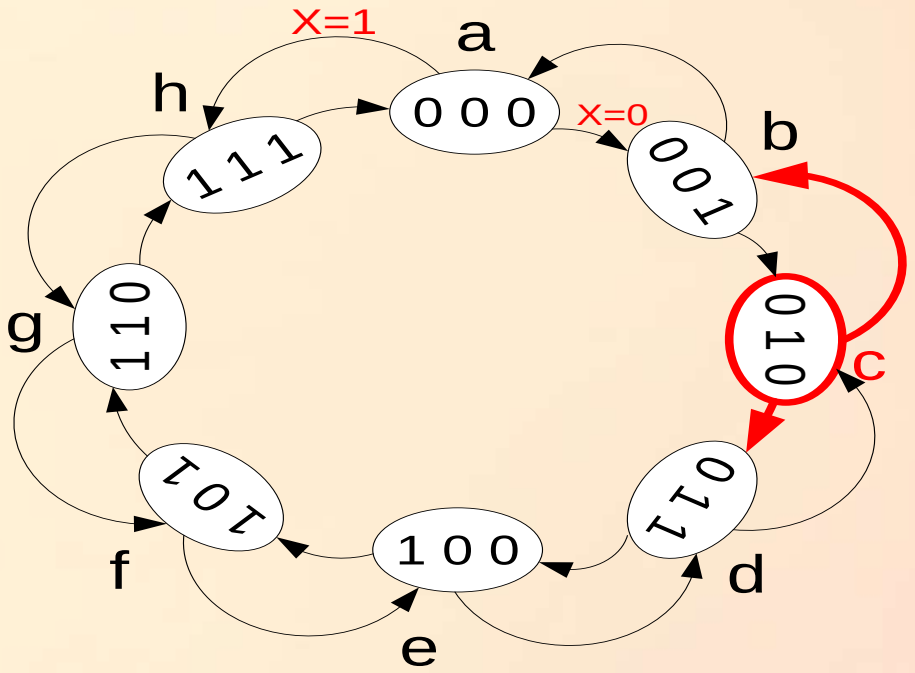
| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |



# State Machines : **State Diagram**

## State Table

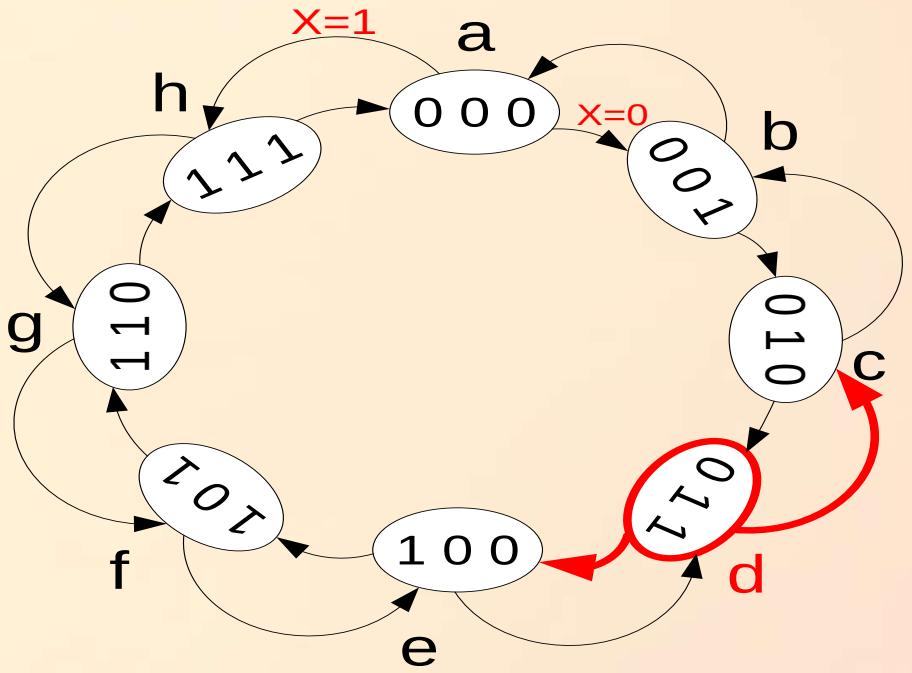
| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |



# State Machines : **State Diagram**

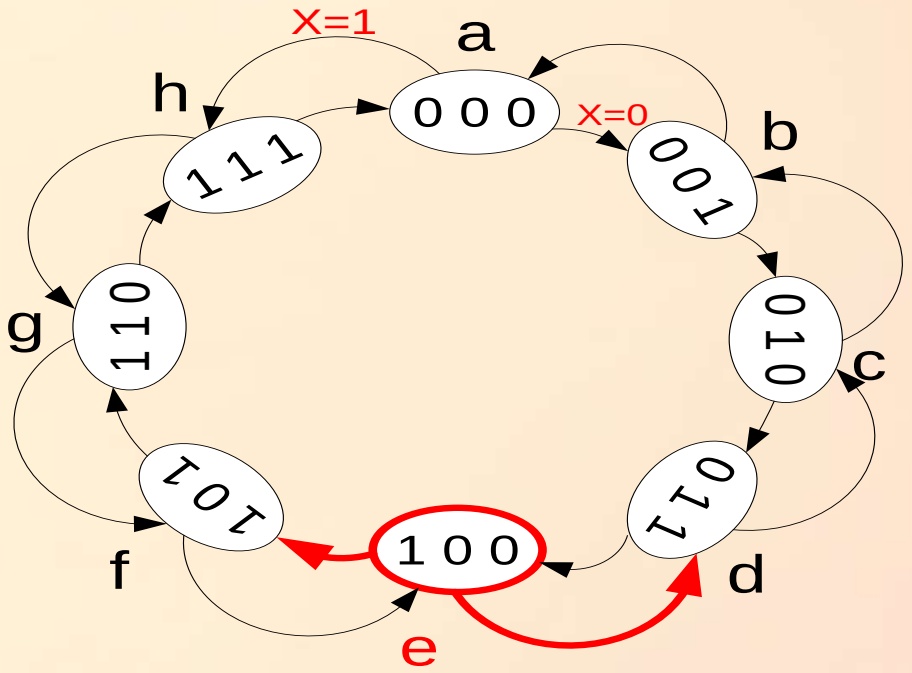
## State Table

| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |
| d | e | c |



## State Table

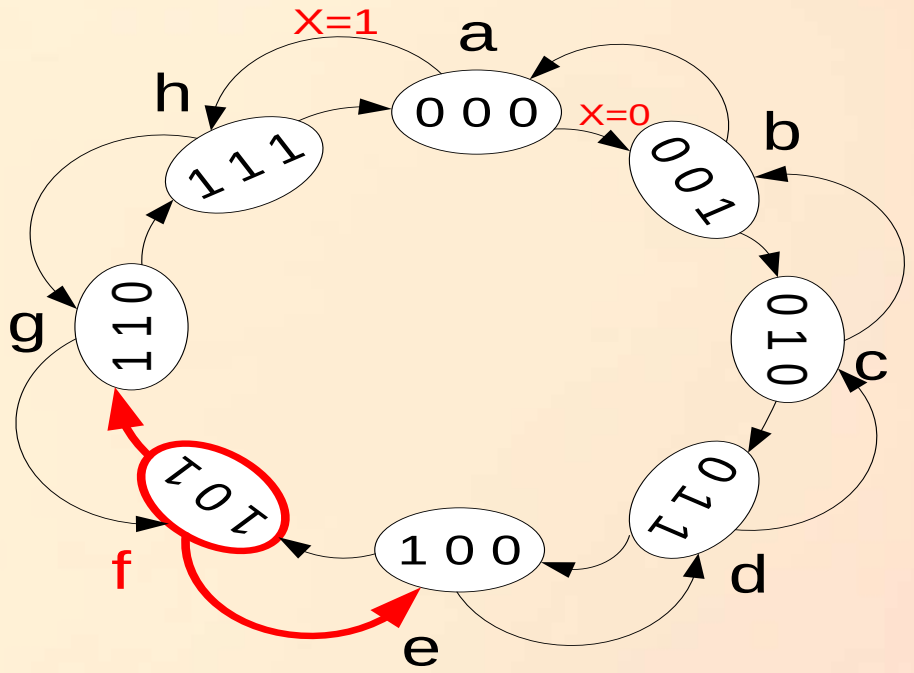
| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |
| d | e | c |
| e | f | d |



# State Machines : **State Diagram**

## State Table

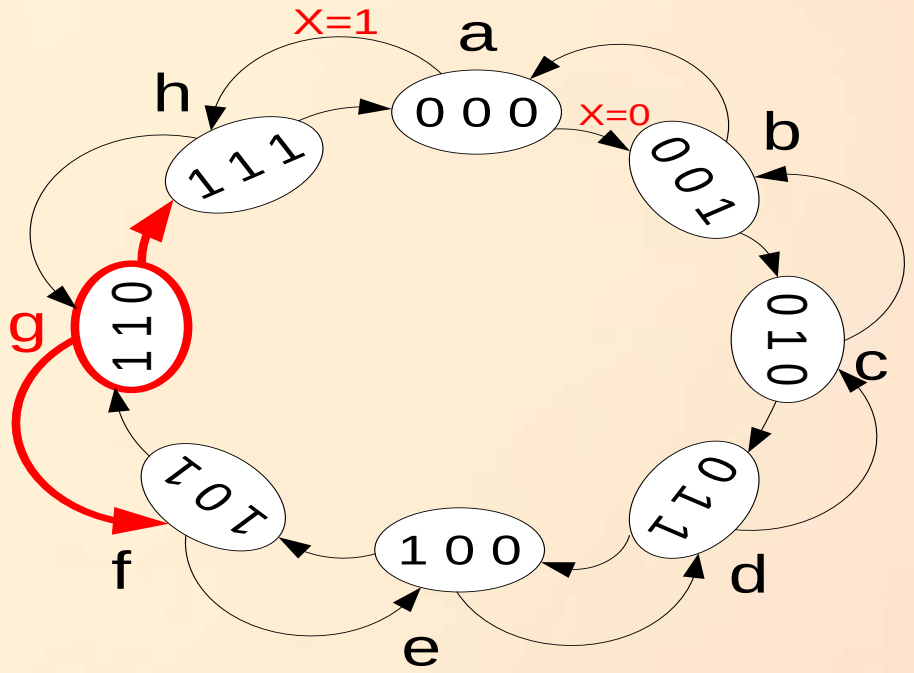
| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |
| d | e | c |
| e | f | d |
| f | g | e |



# State Machines : **State Diagram**

## State Table

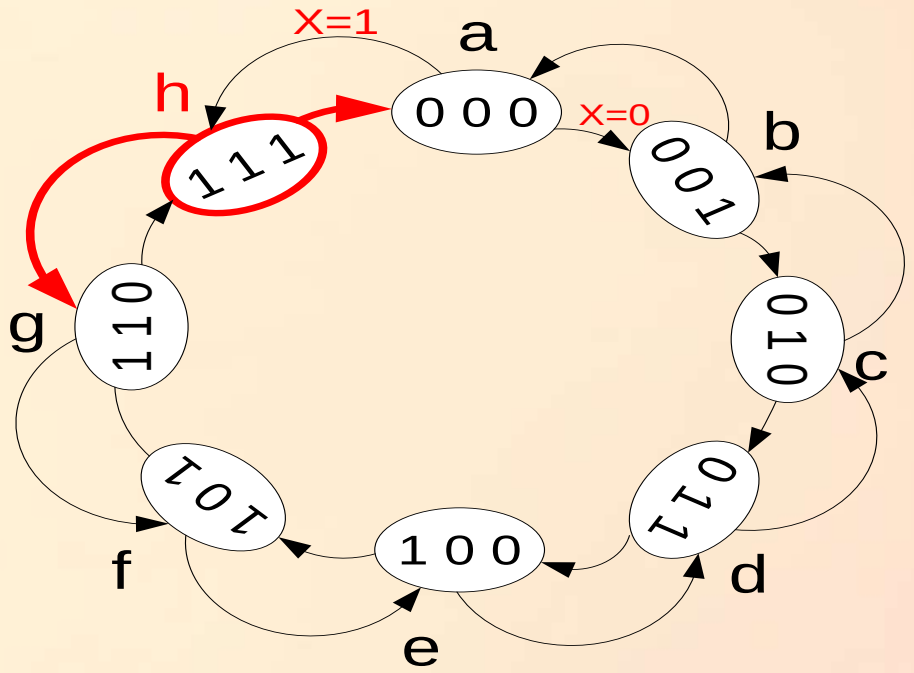
| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |
| d | e | c |
| e | f | d |
| f | g | e |
| g | h | f |

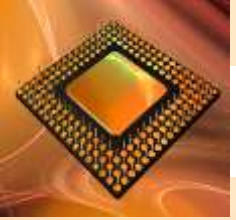


# State Machines : **State Diagram**

## State Table

| | x=0 | x=1 |
|---|---|---|
| a | b | h |
| b | c | a |
| c | d | b |
| d | e | c |
| e | f | d |
| f | g | e |
| g | h | f |
| h | a | g |





# State Machines : **State Diagram**

| | A | B | C |
|---|---|---|---|
| a | 0 | 0 | 0 |
| b | 0 | 0 | 1 |
| c | 0 | 1 | 0 |
| d | 0 | 1 | 1 |
| e | 1 | 0 | 0 |
| f | 1 | 0 | 1 |
| g | 1 | 1 | 0 |
| h | 1 | 1 | 1 |

# State Machines : State Diagram

| | | | X=0 | | | X=1 | | |
|---|---|---|---|---|---|---|---|---|
| A | B | C | A | B | C | A | B | C |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

$$J_A = 1000 + 0011$$

# State Machines : **State Diagram**

| | | | X=0 | | | X=1 | | |
|---|---|---|---|---|---|---|---|---|
| A | B | C | A | B | C | A | B | C |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

$$K_A = 1100 + 0111$$

# State Machines : **State Diagram**

| | | | X=0 | | | X=1 | | |
|---|---|---|---|---|---|---|---|---|
| A | B | C | A | B | C | A | B | C |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

$$J_B = 1000 + 0001 + 1100 + 0101$$

# State Machines : **State Diagram**

| | | | X=0 | | | X=1 | | |
|---|---|---|---|---|---|---|---|---|
| A | B | C | A | B | C | A | B | C |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

$$K_B = 1010 + 0011 + 1110 + 0111$$

$$J_C = 0000 + 1000 + 0010 + 1010 + 0100 + 1100 + 0110 + 1110$$

$$K_C = 0001 + 1001 + 0011 + 1011 + 0101 + 1101 + 0111 + 1111$$

$$J_A = 1000 + 0011$$
$$K_A = 1100 + 0111$$

$$J_B = 1000 + 0001 + 1100 + 0101$$
$$K_B = 1010 + 0011 + 1110 + 0111$$

$$J_C = 0000 + 1000 + 0010 + 1010 + 0100 + 1100 + 0110 + 1110$$
$$K_C = 0001 + 1001 + 0011 + 1011 + 0101 + 1101 + 0111 + 1111$$

$$J_A = 1000 + 0011 = X\,\overline{A}\,\overline{B}\,\overline{C} + \overline{X}\,\overline{A}BC$$

$$K_A = 1100 + 0111 = XA\,\overline{B}\,\overline{C} + \overline{X}\,ABC$$

$$J_B = 1000 + 0001 + 1100 + 0101 = X\,\overline{B}\,\overline{C} + \overline{X}\,\overline{B}C$$

$$K_B = 1010 + 0011 + 1110 + 0111 = XB$$

$$J_C = 0000 + 1000 + 0010 + 1010 + 0100 + 1100 + 0110 + 1110 = \overline{C}$$

$$K_C = 0001 + 1001 + 0011 + 1011 + 0101 + 1101 + 0111 + 1111 = C$$

$$J_A = 1000 + 0011 = X\,\overline{A}\,\overline{B}\,\overline{C} + \overline{X}\,\overline{A}BC = X\,\overline{B}\,\overline{C} + \overline{X}\,BC$$

$$K_A = 1100 + 0111 = XA\,\overline{B}\,\overline{C} + \overline{X}\,ABC = X\,\overline{B}\,\overline{C} + \overline{X}\,BC$$

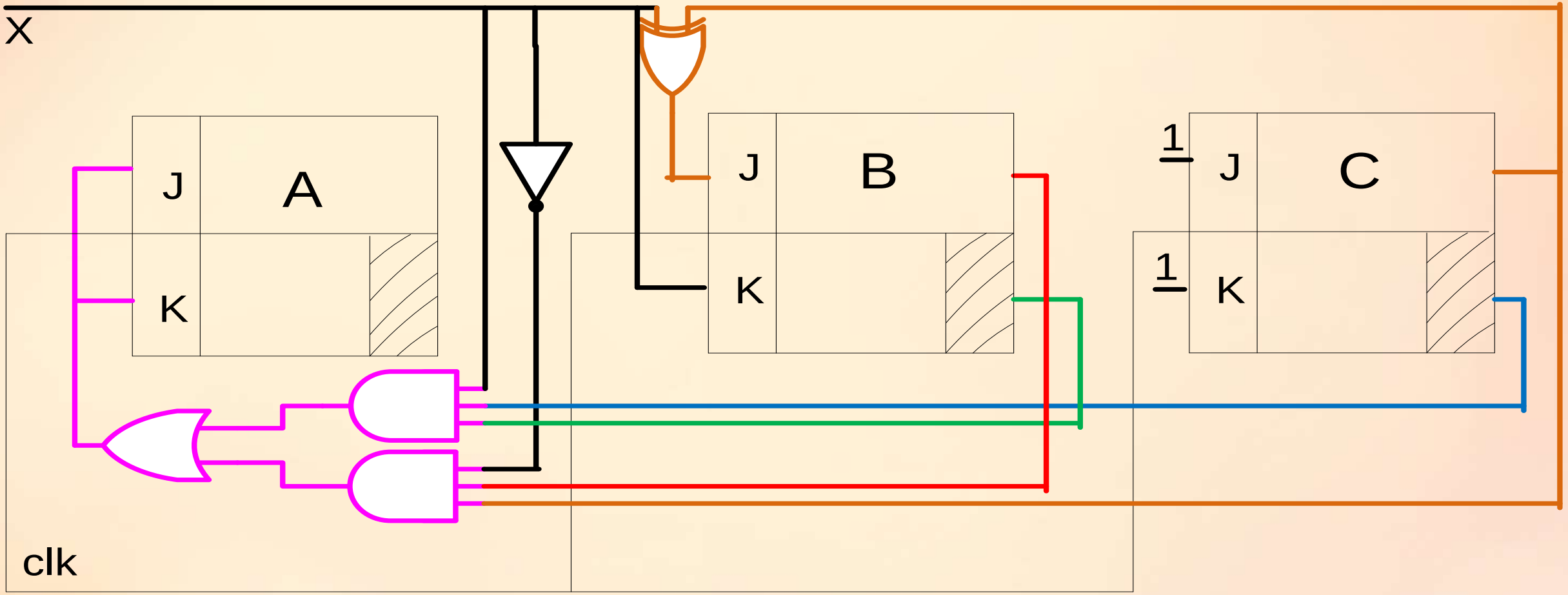
$$J_B = 1000 + 0001 + 1100 + 0101 = X\,\overline{B}\,\overline{C} + \overline{X}\,\overline{B}C = X\,\overline{C} + \overline{X}C = X \oplus C$$

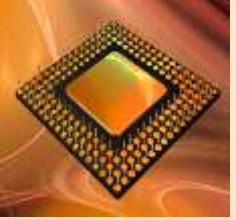
$$K_B = 1010 + 0011 + 1110 + 0111 = XB = X$$

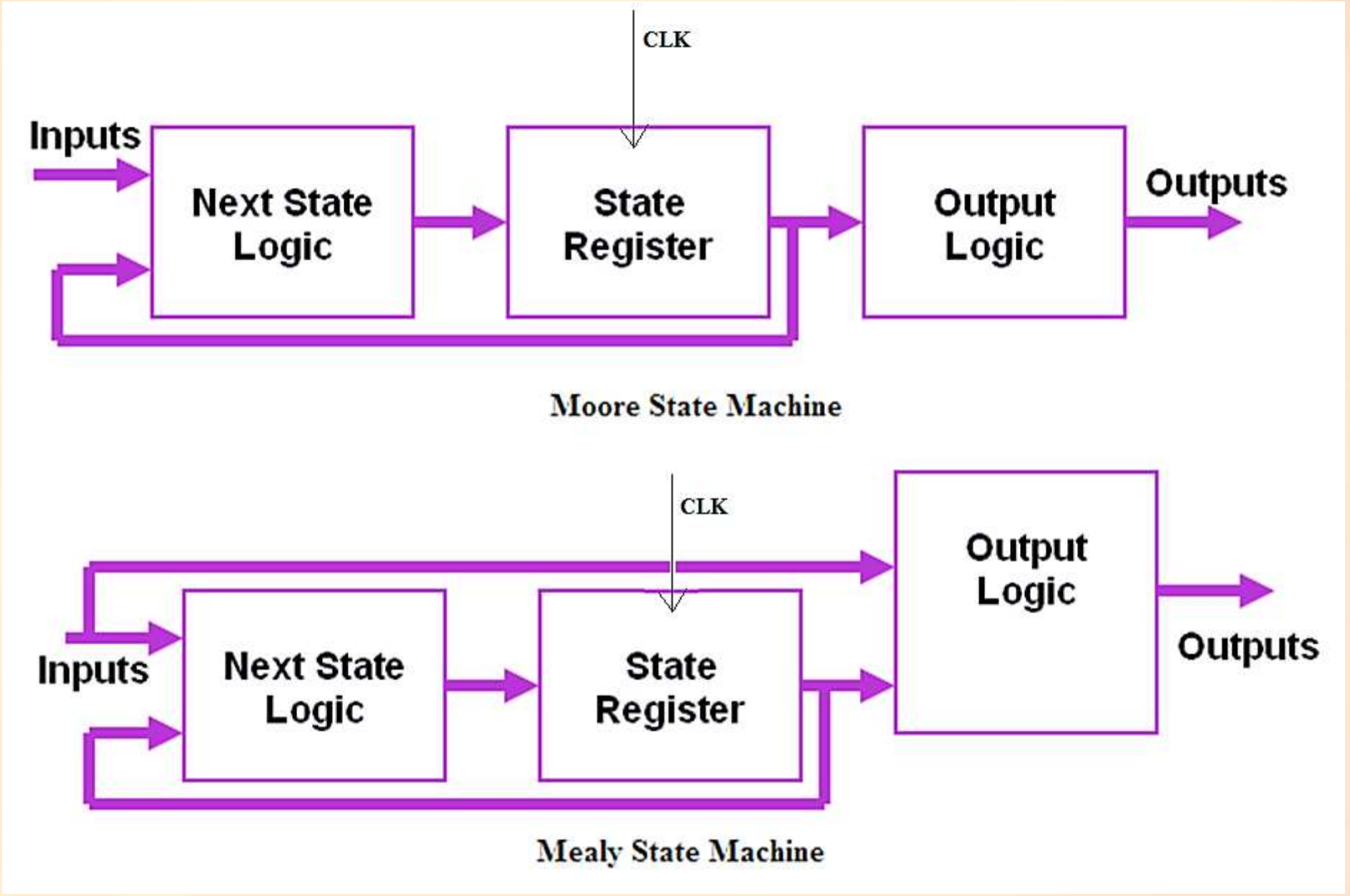
$$J_C = 0000 + 1000 + 0010 + 1010 + 0100 + 1100 + 0110 + 1110 = \overline{C} = 1$$

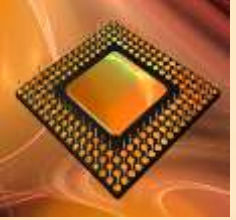
$$K_C = 0001 + 1001 + 0011 + 1011 + 0101 + 1101 + 0111 + 1111 = C = 1$$

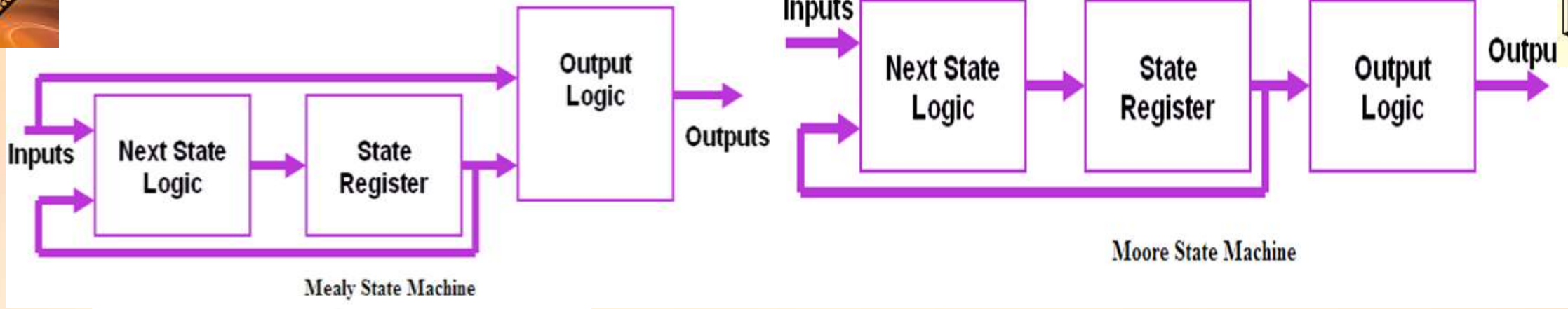
# State Machines : **State Diagram**

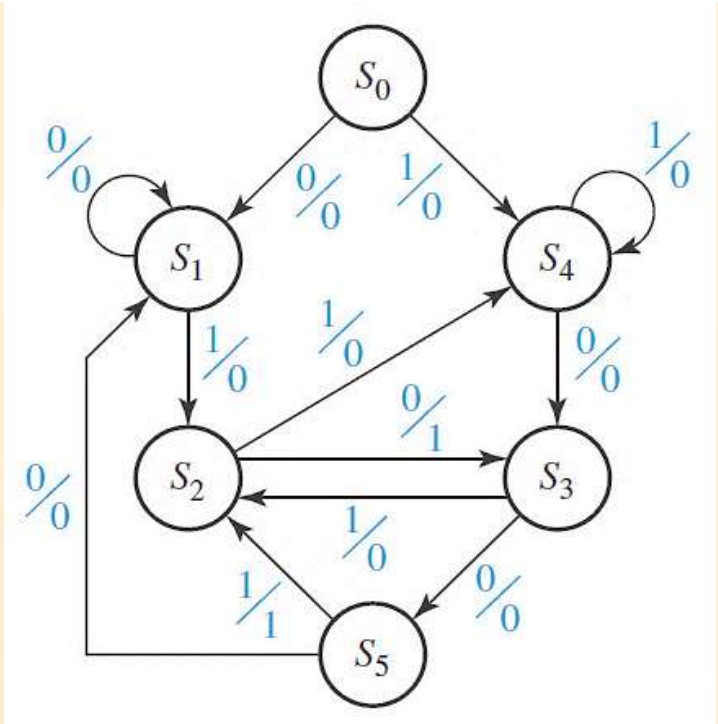


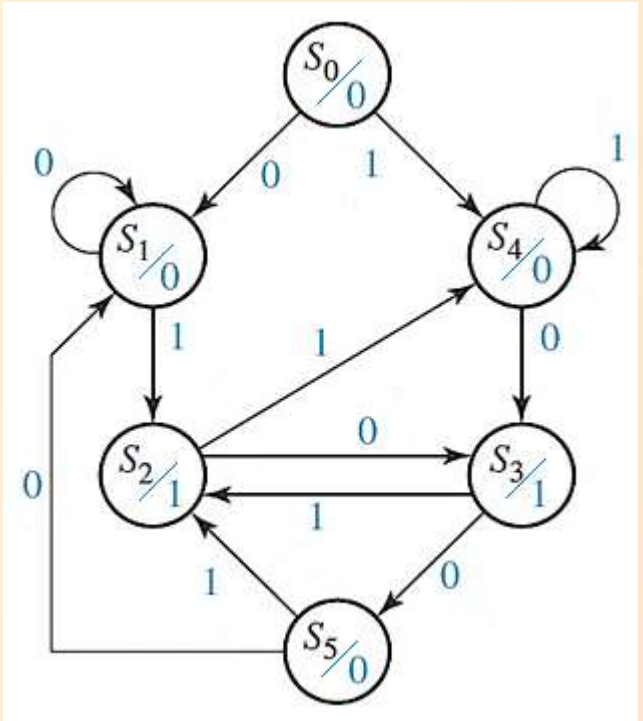


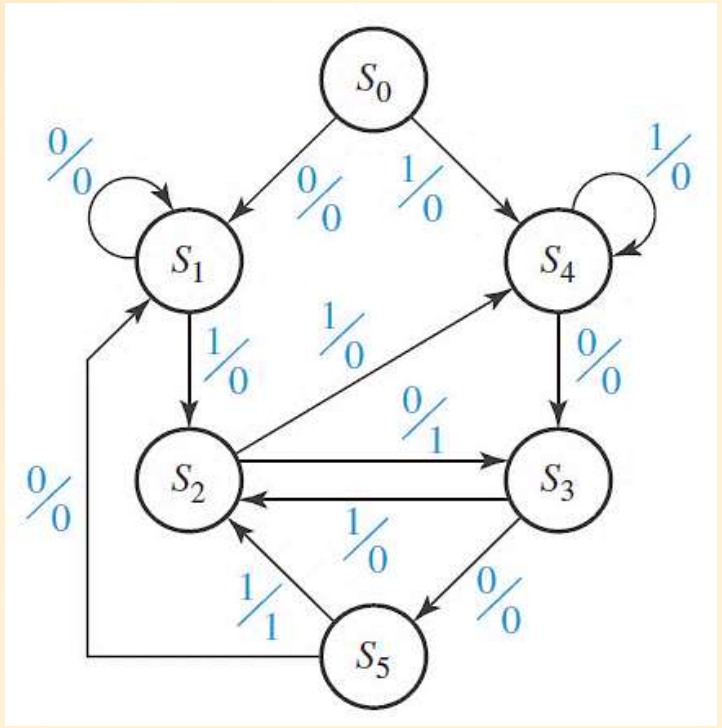

CLK
Inputs
Next State Logic
State Register
Output Logic
Outputs
Moore State Machine
CLK
Output Logic
Inputs
Next State Logic
State Register
Outputs
Mealy State Machine

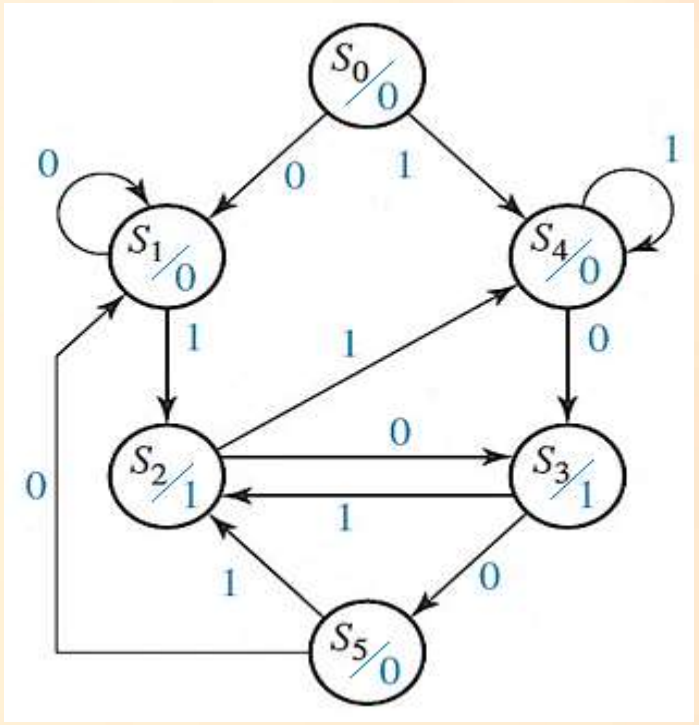



Inputs
Next State Logic
State Register
Output Logic
Outputs
Mealy State Machine
Inputs
Next State Logic
State Register
Output Logic
Outpu
Moore State Machine


$S_0$
$S_1$
$S_2$
$S_3$
$S_4$
$S_5$

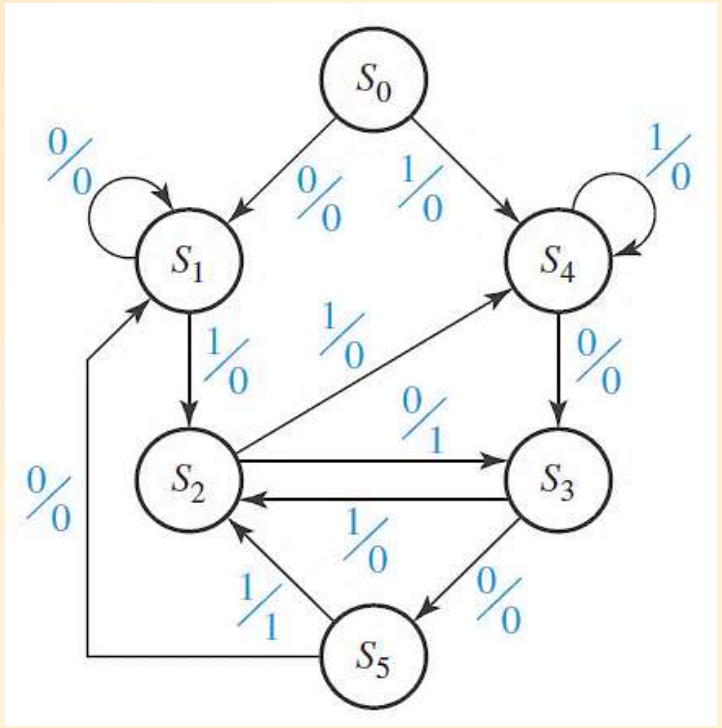

$S_0/0$
$S_1/0$
$S_2/1$
$S_3/1$
$S_4/0$
$S_5/0$

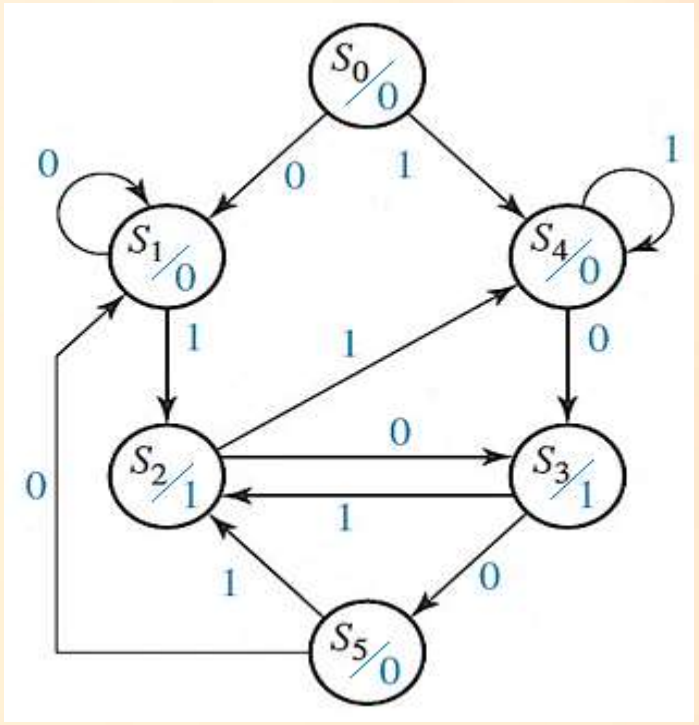



| Present State | Next State | | Output (Z) | |
|---|---|---|---|---|
| | X=0 | X=1 | X=0 | X=1 |
| S0 | | | | |
| S1 | | | | |
| S2 | | | | |
| S3 | | | | |
| S4 | | | | |
| S5 | | | | |
| --- | | | | |

| Present State | Next State | | Output (Z) | |
|---|---|---|---|---|
| | X=0 | X=1 | X=0 | X=1 |
| S0 | | | | |
| S1 | | | | |
| S2 | | | | |
| S3 | | | | |
| S4 | | | | |
| S5 | | | | |
| --- | | | | |



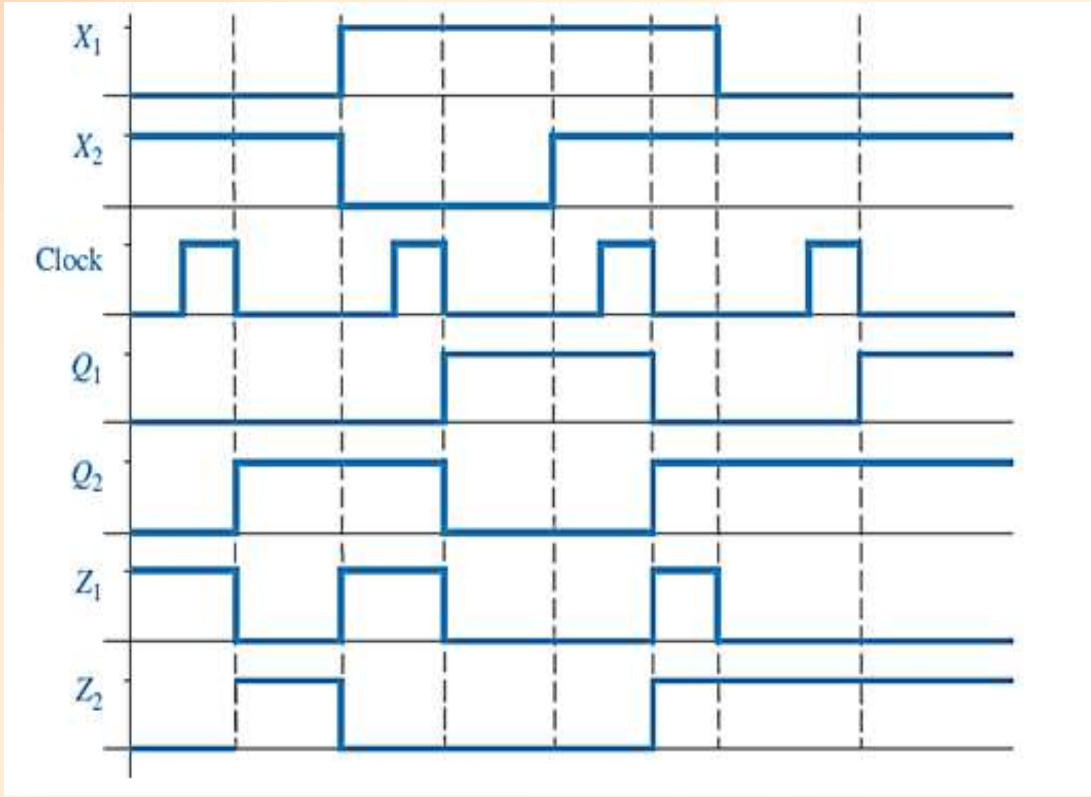




| Present State | Next State | | Output (Z) | |
|---|---|---|---|---|
| | X=0 | X=1 | X=0 | X=1 |
| S0 | s1 | s4 | 0 | 0 |
| S1 | s1 | s2 | 0 | 0 |
| S2 | s3 | s4 | 1 | 0 |
| S3 | s5 | s2 | 0 | 0 |
| S4 | s3 | s4 | 0 | 0 |
| S5 | s1 | s2 | 0 | 1 |
| --- | | | | |

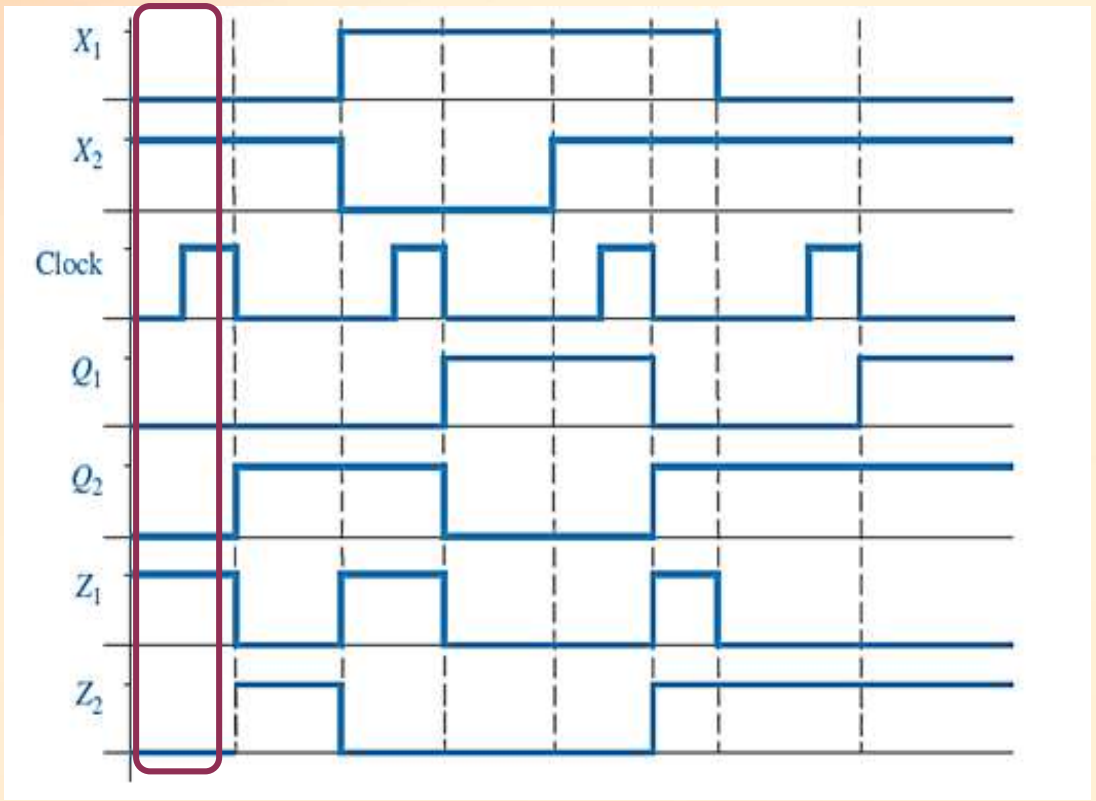
| Present State | Next State | | Output (Z) | |
|---|---|---|---|---|
| | X=0 | X=1 | X=0 | X=1 |
| S0 | s1 | s4 | 0 | 0 |
| S1 | s1 | s2 | 0 | 0 |
| S2 | s3 | s4 | 1 | 1 |
| S3 | s5 | s2 | 1 | 1 |
| S4 | s3 | s4 | 0 | 0 |
| S5 | s1 | s2 | 0 | 0 |
| --- | | | | |

**Example:**

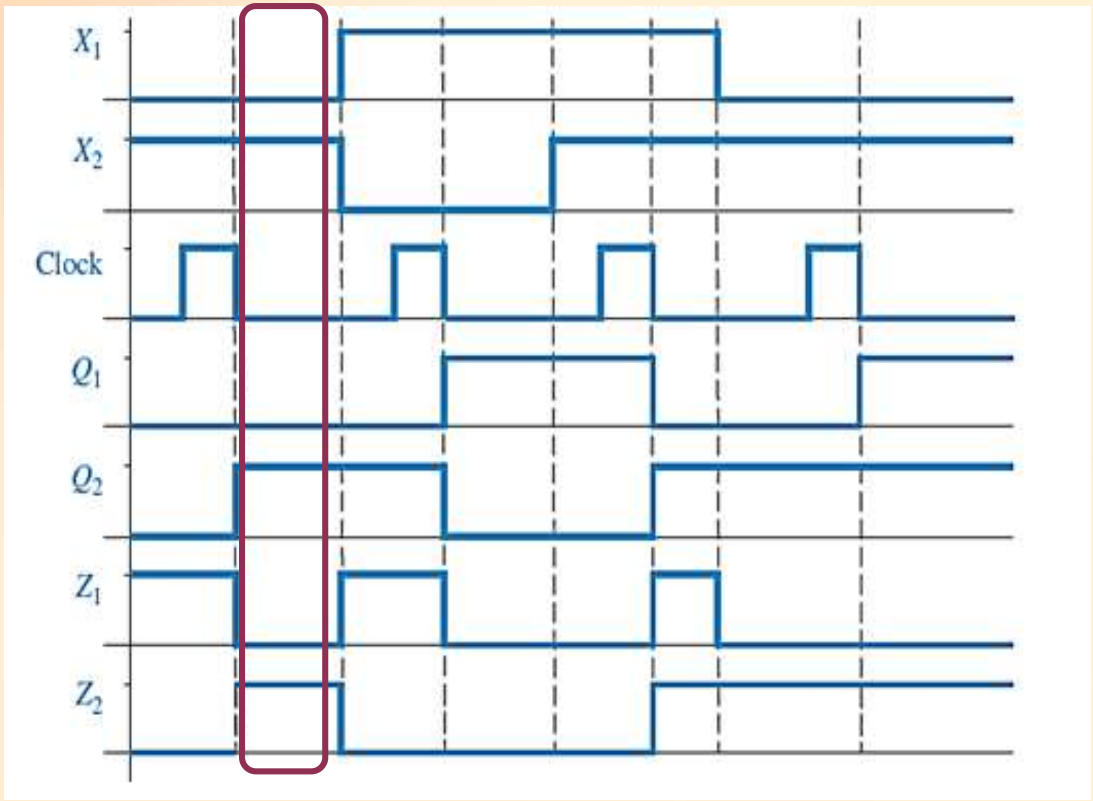
**Given the following timing chart for sequential circuit, construct as much of the state table as possible. Is this a Mealy or Moore machine? In the chart X1 and X2 are independent input signals and Z1 and Z2 are output signals. (X2X1/Z2Z1)**



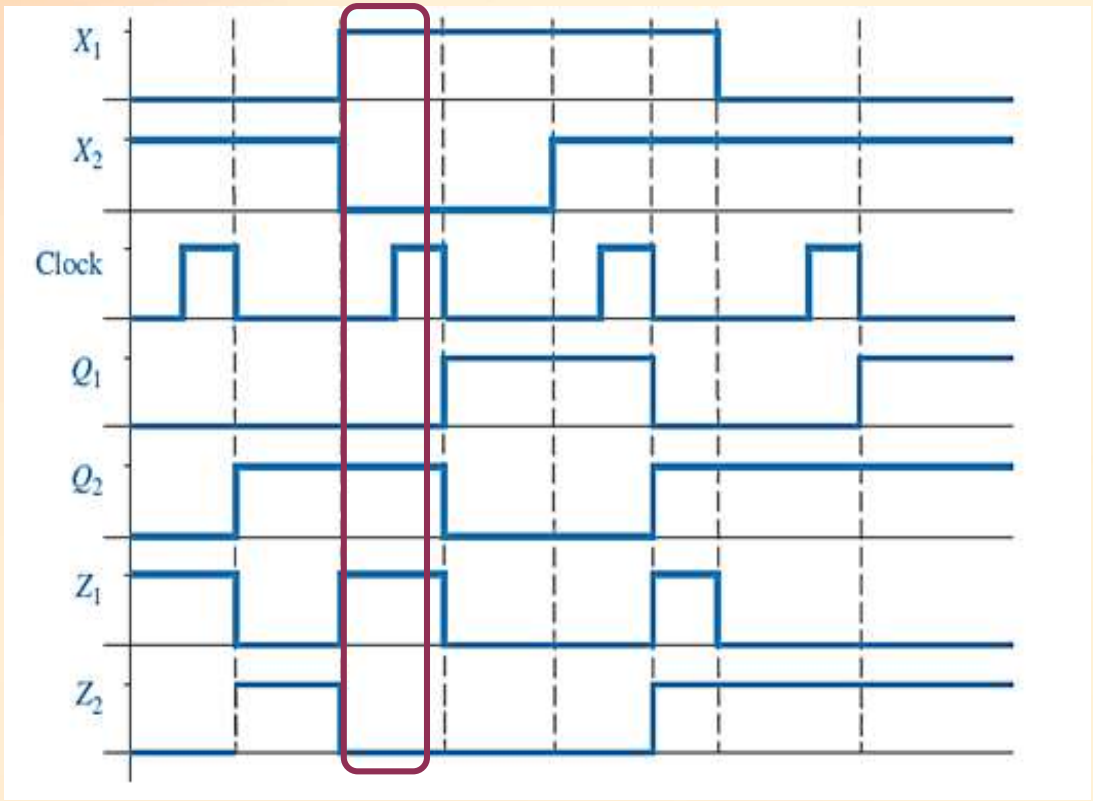

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | | |
| 01 | | | | | | | | |
| 10 | | | | | | | | |
| 11 | | | | | | | | |



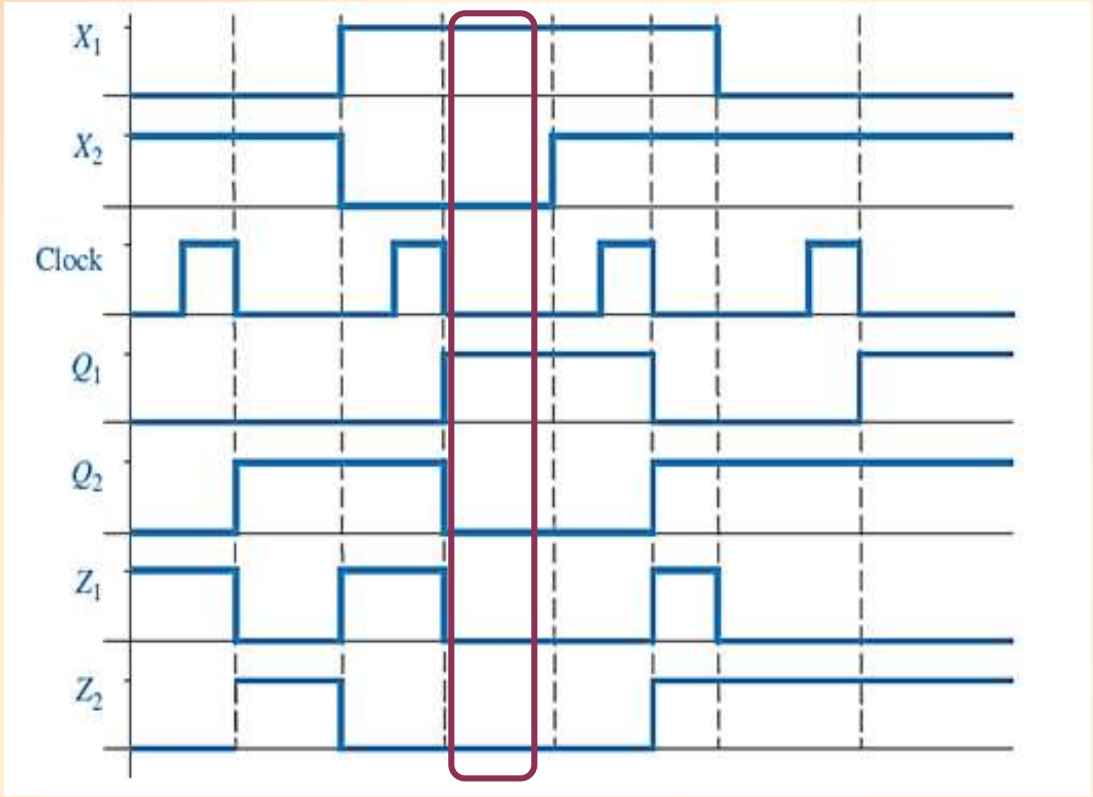

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | | | |
| 10 | | | | | | | | |
| 11 | | | | | | | | |



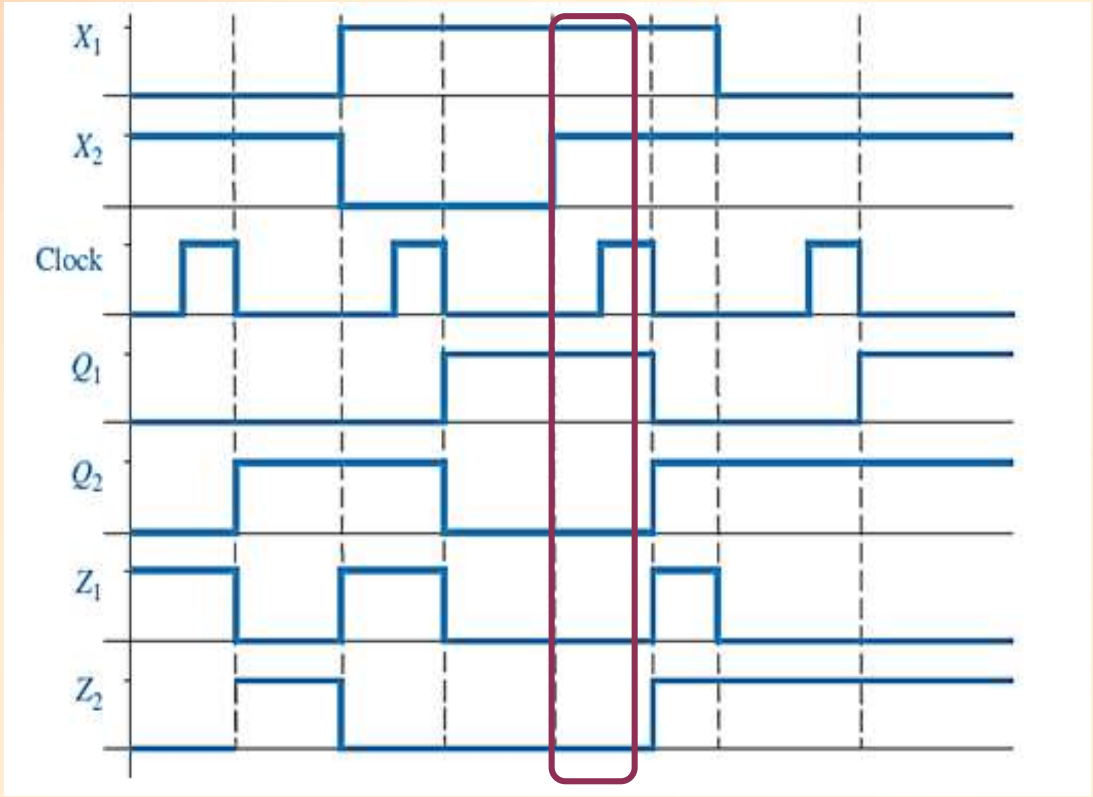

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | | | |
| 10 | | | | | | | 10 | |
| 11 | | | | | | | | |



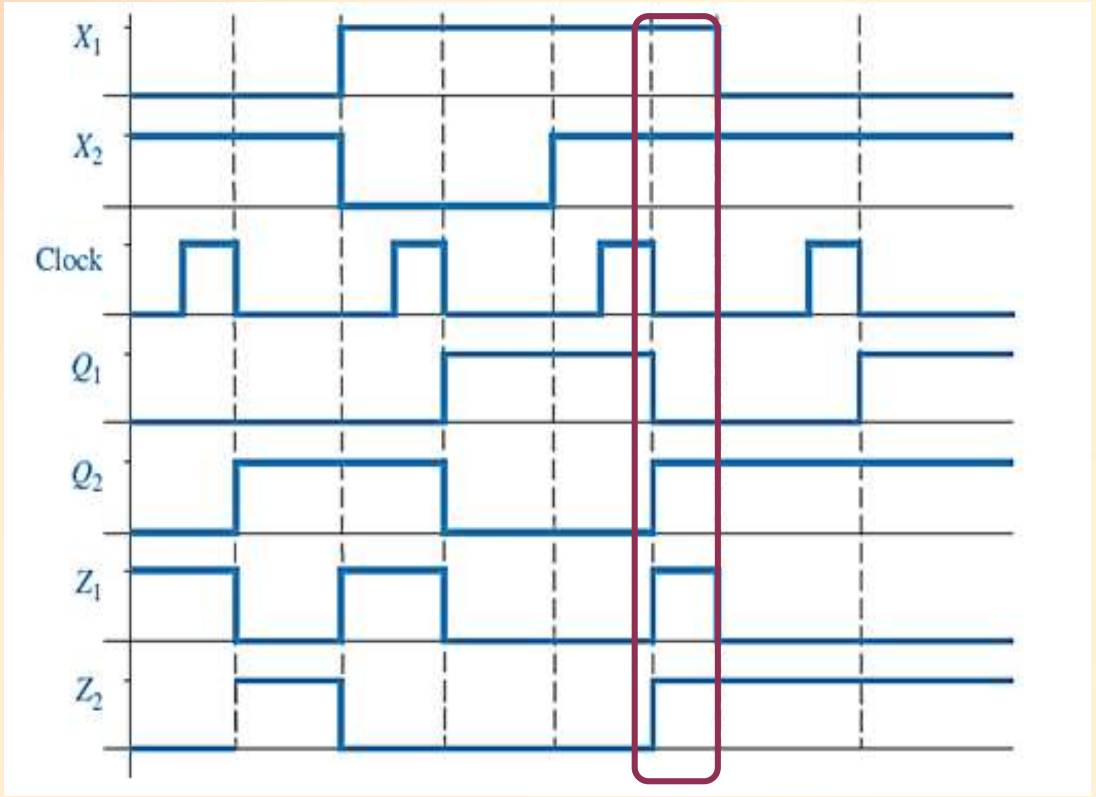

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | | | |
| 10 | | | | | | 01 | 10 | |
| 11 | | | | | | | | |



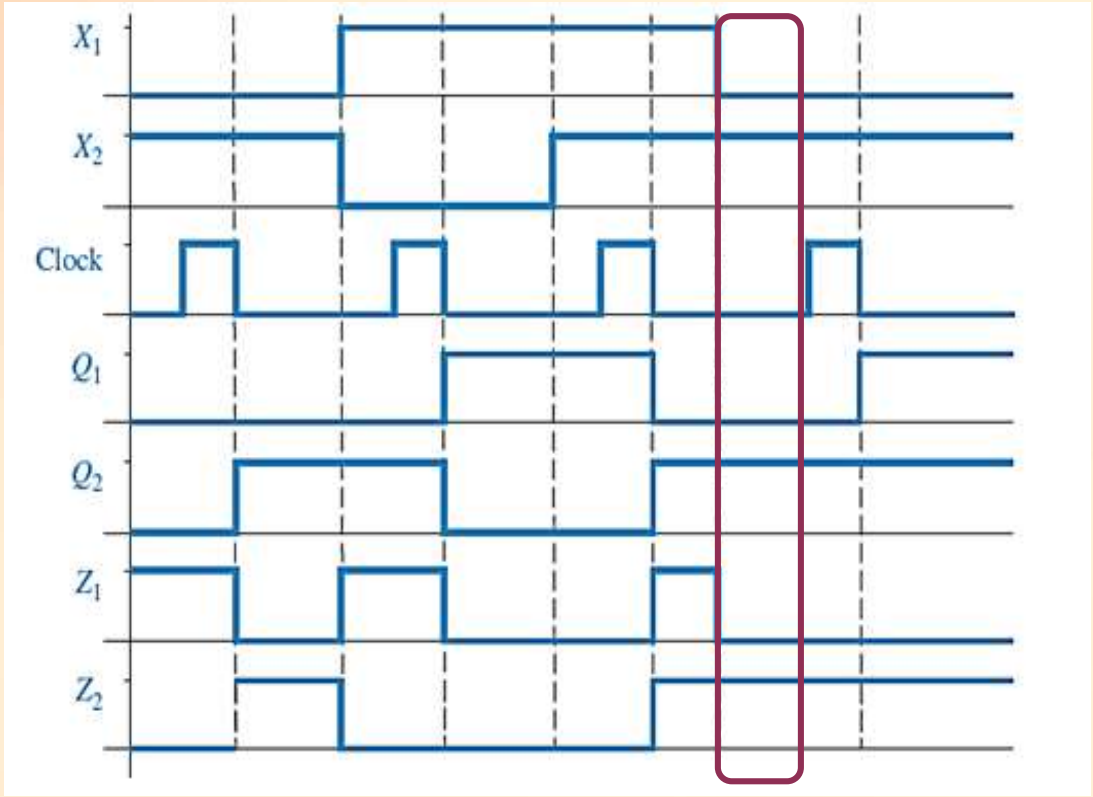

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | |
| 10 | | | | | | 01 | 10 | |
| 11 | | | | | | | | |



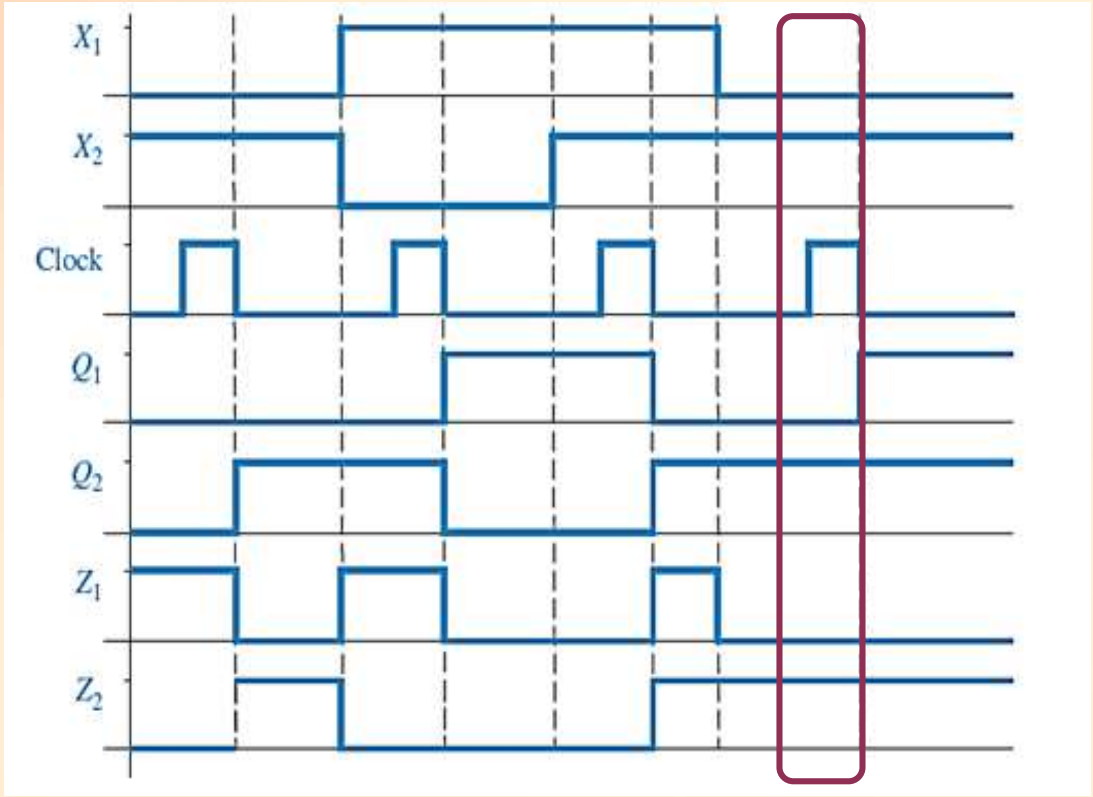

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | |
| 11 | | | | | | | | |



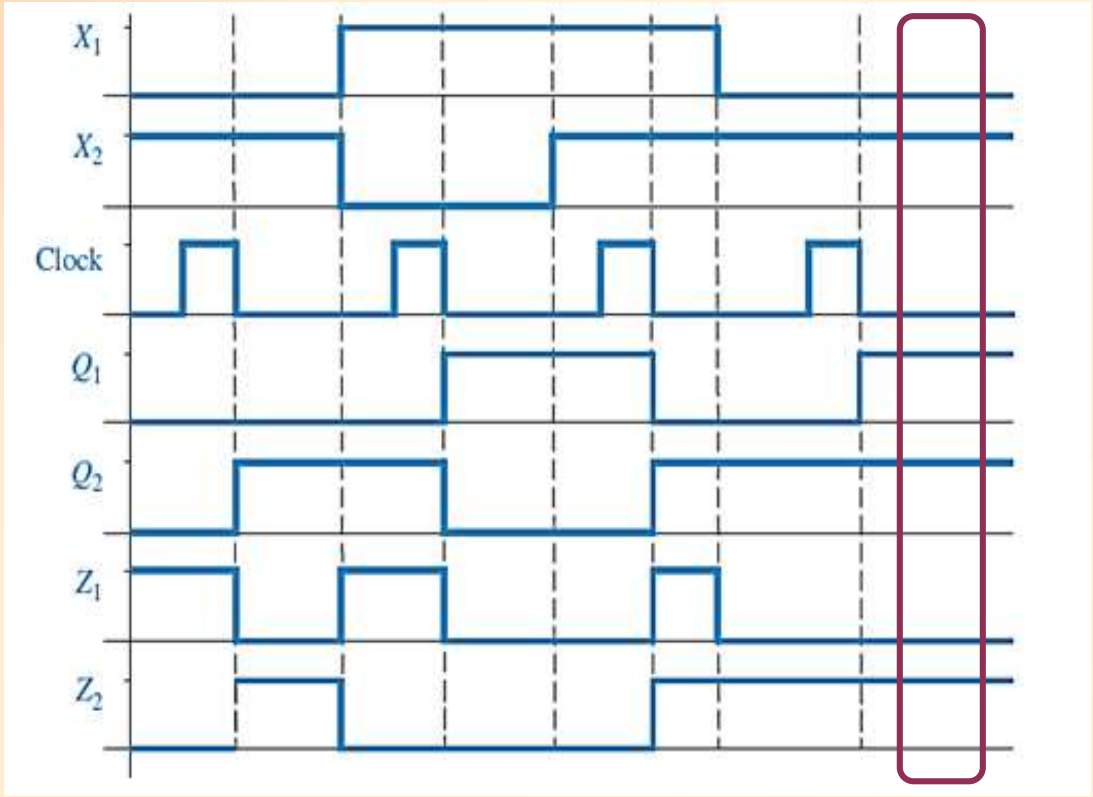

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | 11 |
| 11 | | | | | | | | |



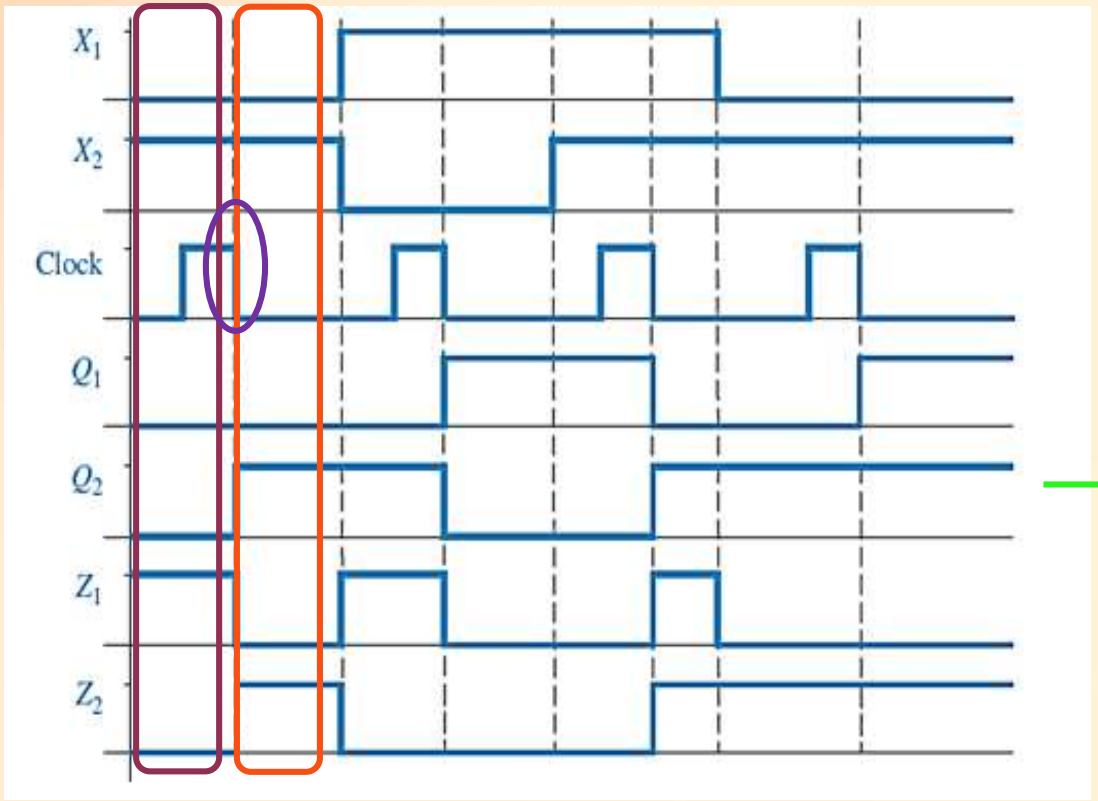

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | 11 |
| 11 | | | | | | | | |



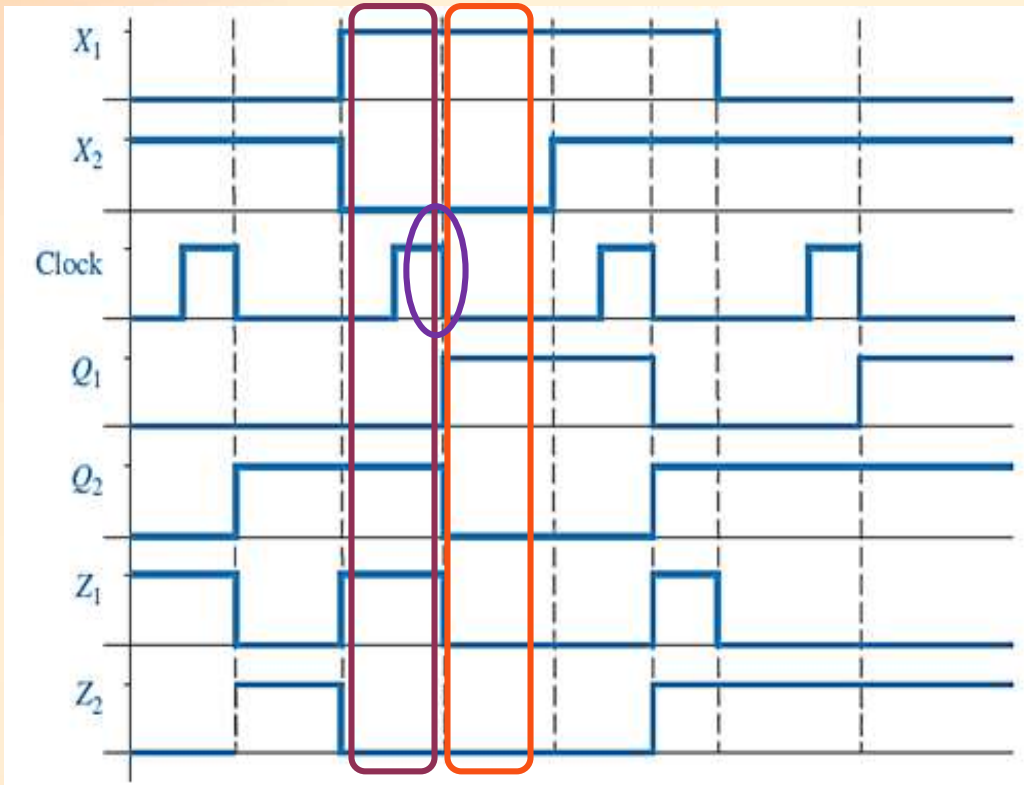

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | 11 |
| 11 | | | | | | | | |



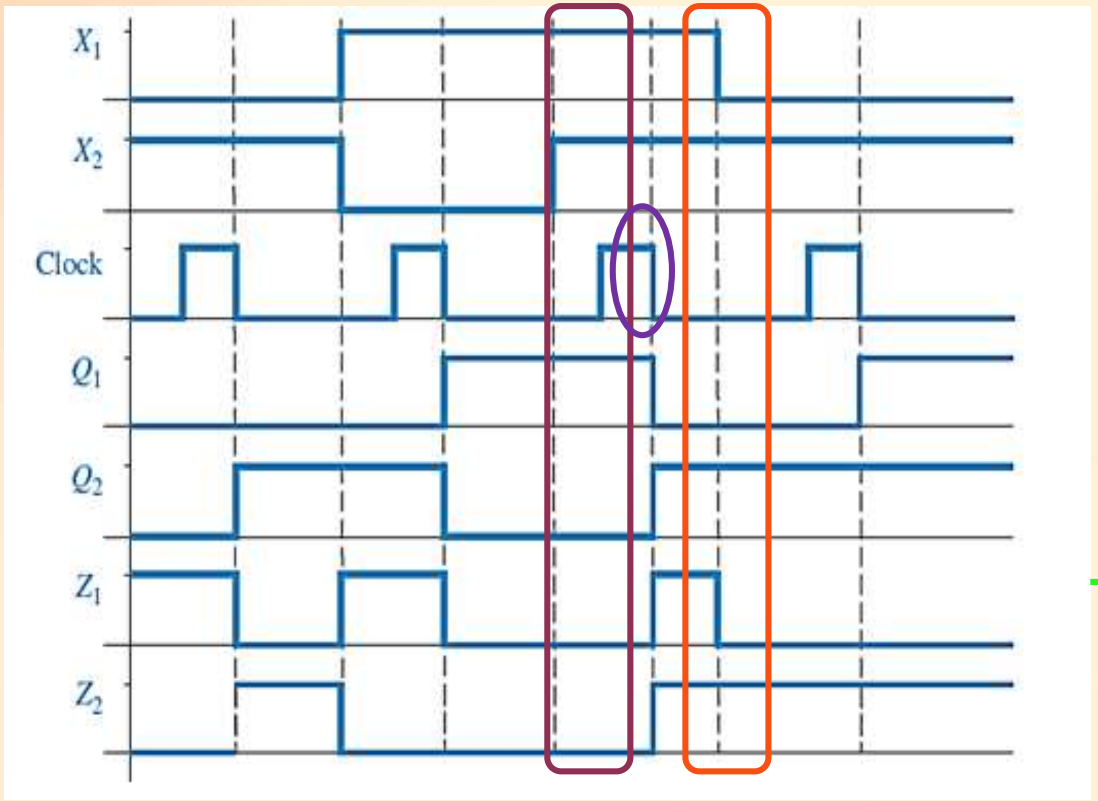

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | 11 |
| 11 | | | | | | | 10 | |



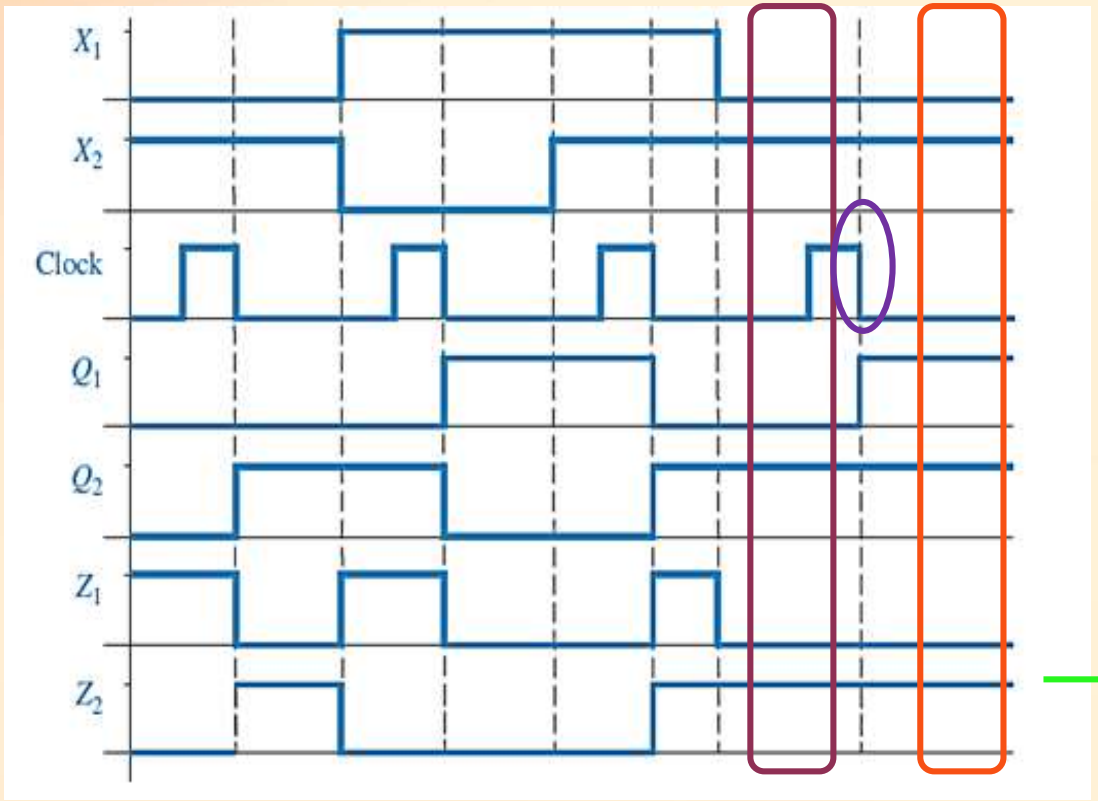

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | 10 | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | | | | | 01 | 10 | 11 |
| 11 | | | | | | | 10 | |



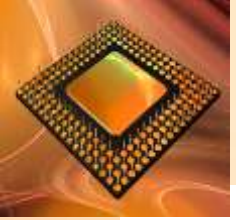

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | 10 | | | | 01 | |
| 01 | | | | | | 00 | | 00 |
| 10 | | 01 | | | | 01 | 10 | 11 |
| 11 | | | | | | | 10 | |




| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | 10 | | | | 01 | |
| 01 | | | | 10 | | 00 | | 00 |
| 10 | | 01 | | | | 01 | 10 | 11 |
| 11 | | | | | | | 10 | |



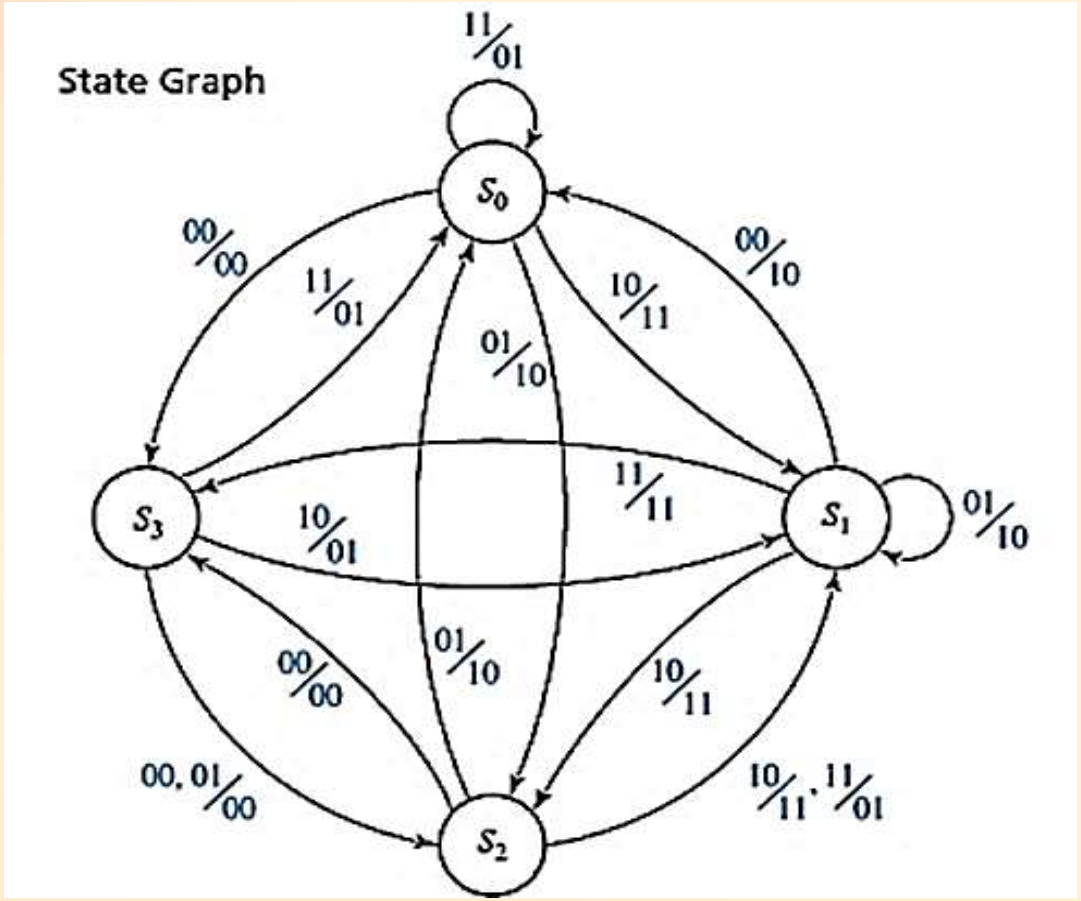

| Present State (Q2Q1) | Next State (Q2Q1) | | | | Output (Z2,Z1) | | | |
|---|---|---|---|---|---|---|---|---|
| | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 | X2X1 00 | X2X1 01 | X2X1 10 | X2X1 11 |
| | Q2Q1 | Q2Q1 | Q2Q1 | Q2Q1 | Z2Z1 | Z2Z1 | Z2Z1 | Z2Z1 |
| 00 | | | 10 | | | | 01 | |
| 01 | | | | 10 | | 00 | | 00 |
| 10 | | 01 | 11 | | | 01 | 10 | 11 |
| 11 | | | | | | | 10 | |

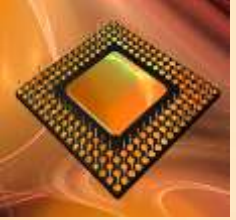


## Determine the state graph of following state table:

| Present State | Next State $X=0$ | Next State $X=1$ | Output ($Z$) $X=0$ | Output ($Z$) $X=1$ |
|---|---|---|---|---|
| 000 | 100 | 101 | 1 | 0 |
| 001 | 100 | 101 | 0 | 1 |
| 010 | 000 | 000 | 1 | 0 |
| 011 | 000 | 000 | 0 | 1 |
| 100 | 111 | 110 | 1 | 0 |
| 101 | 110 | 110 | 0 | 1 |
| 110 | 011 | 010 | 1 | 0 |
| 111 | 011 | 011 | 0 | 1 |

## Determine the state table of following state graph:





14.34 A sequential circuit has an input ($X$) and an output ($Z$). The output is the same as the input was two clock periods previously. For example,

$X$ = 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 1

$Z$ = 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0

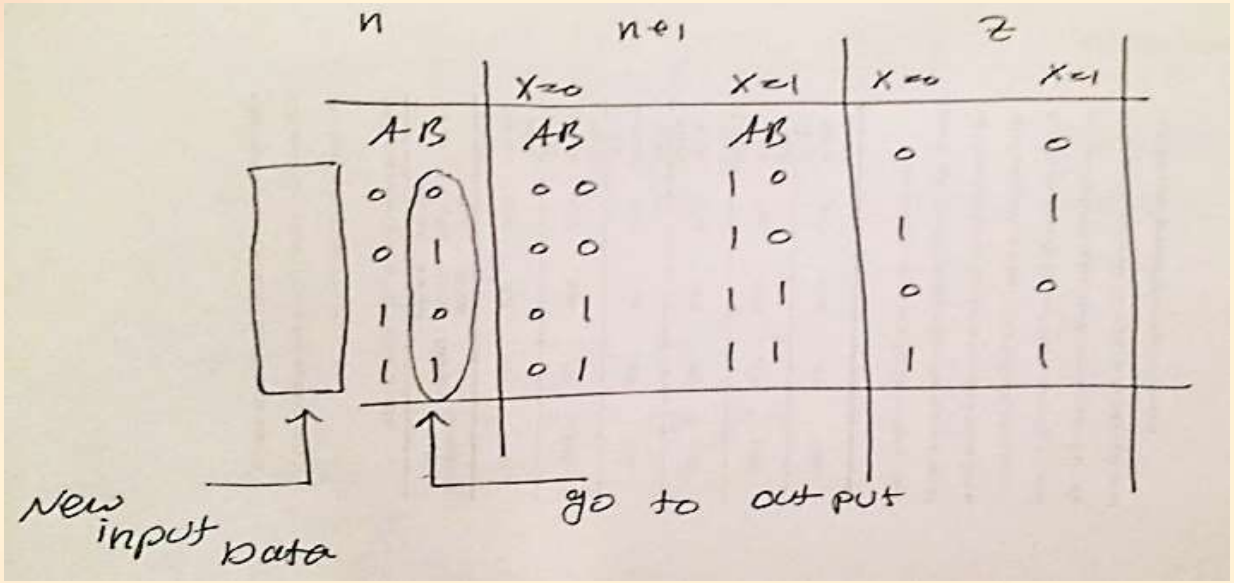
The first two values of $Z$ are 0. Find a Mealy state graph and table for the circuit.

**14.34 A sequential circuit has an input ($X$) and an output ($Z$). The output is the same as the input was two clock periods previously. For example,**

$X$ = 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 1

$Z$ = 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0

**The first two values of $Z$ are 0. Find a Mealy state graph and table for the circuit.**

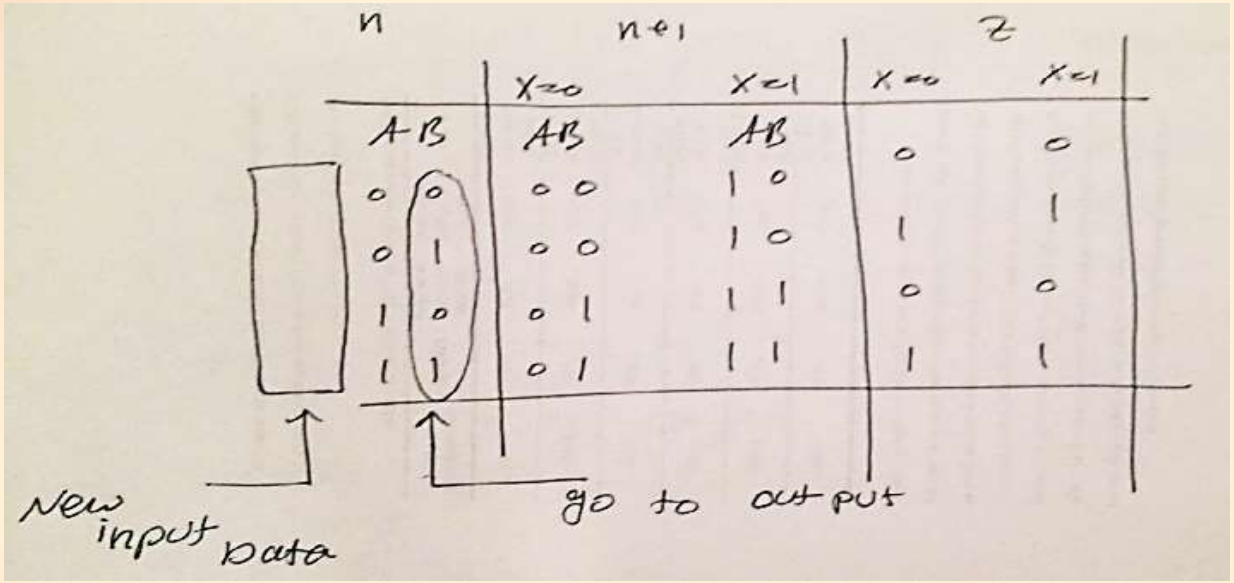


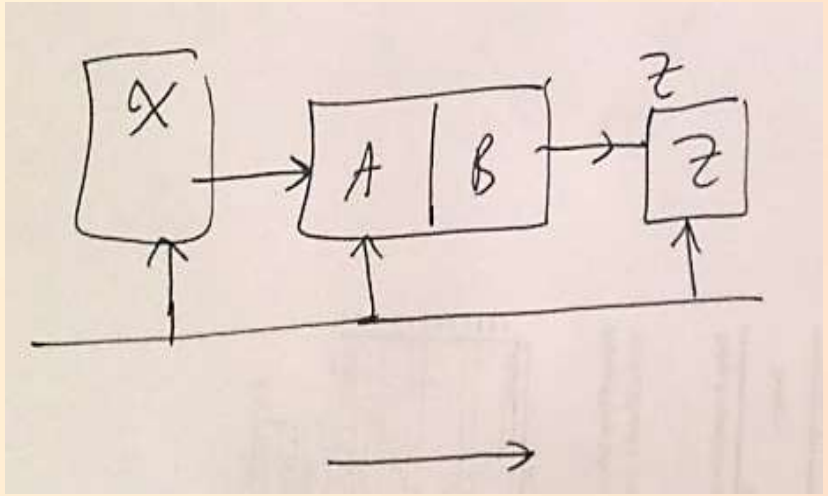
**14.34 A sequential circuit has an input ($X$) and an output ($Z$). The output is the same as the input was two clock periods previously. For example,**

$X$ = 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 1

$Z$ = 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0

**The first two values of $Z$ are 0. Find a Mealy state graph and table for the circuit.**



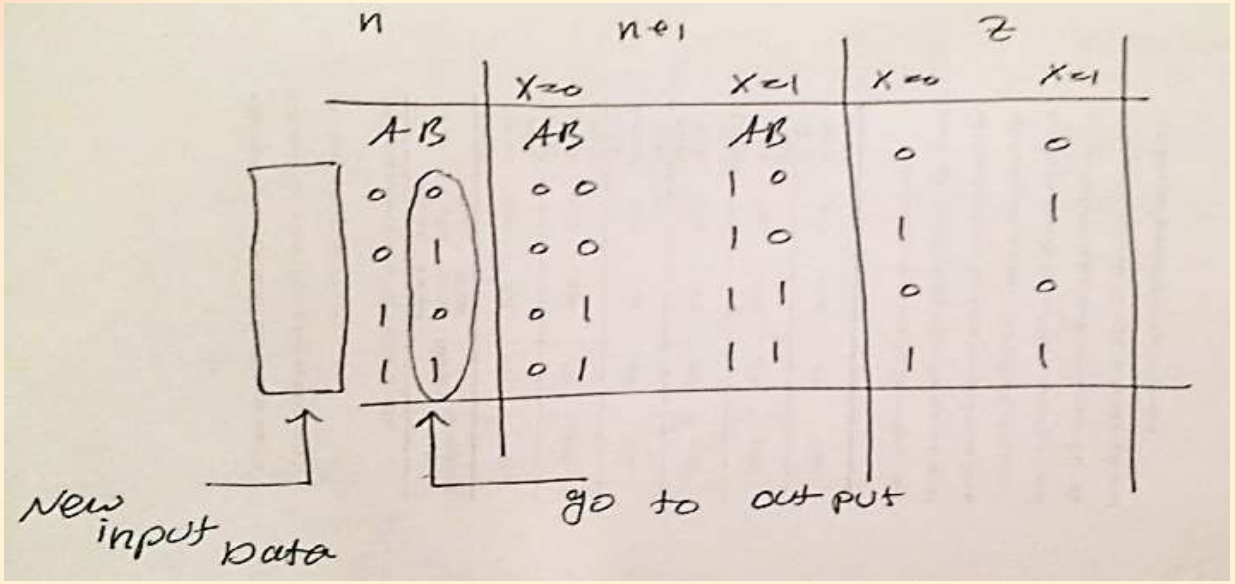

The direction of data is left to right
It could be right to left too.

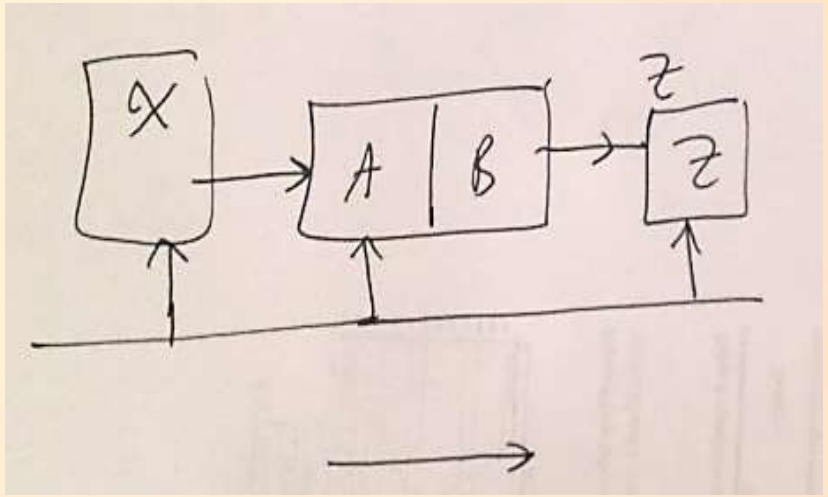
14.34 A sequential circuit has an input ($X$) and an output ($Z$). The output is the same as the input was two clock periods previously. For example,

$X$ = 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 1

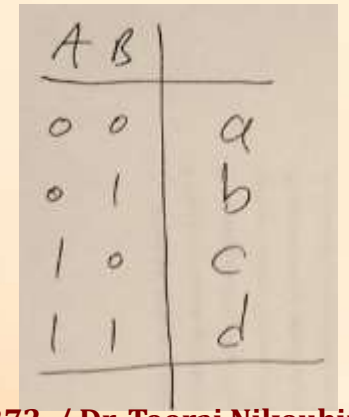
$Z$ = 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0

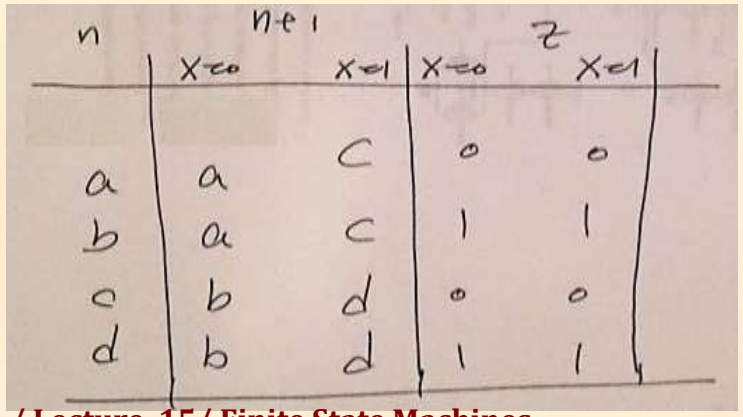
The first two values of $Z$ are 0. Find a Mealy state graph and table for the circuit.




The direction of data is left to right
It could be right to left too.



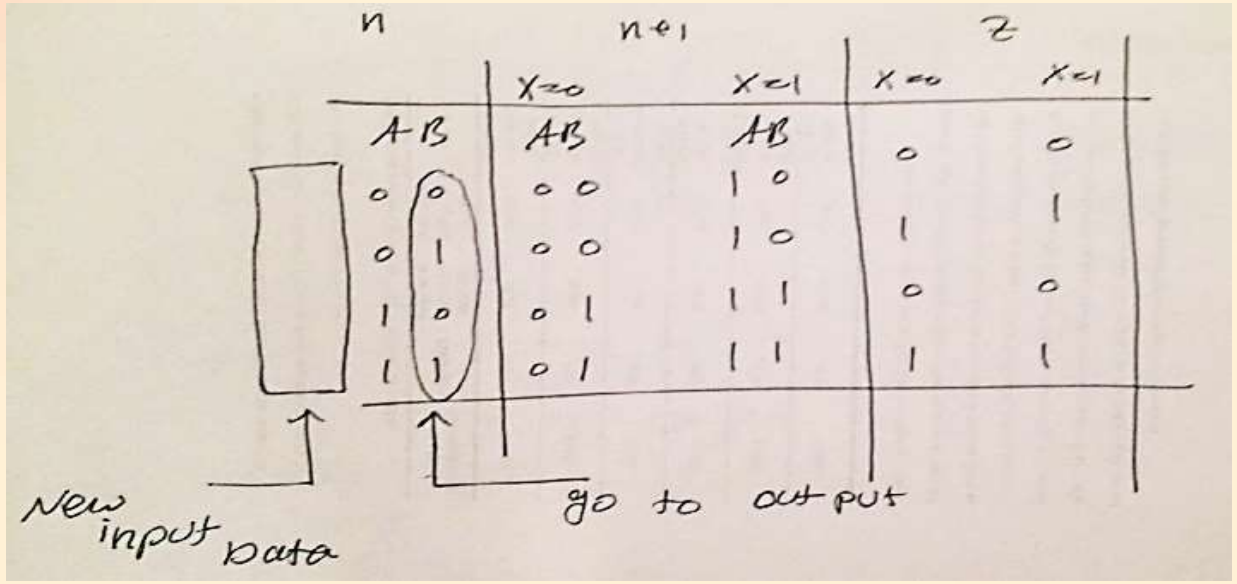



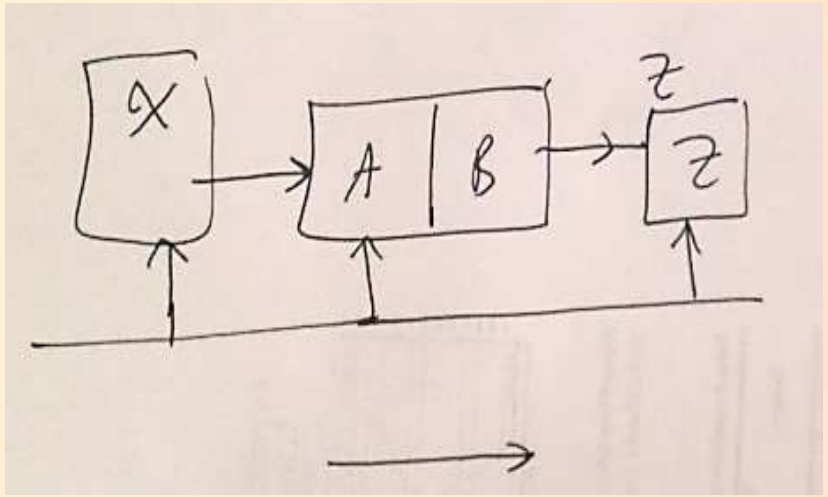
**14.34 A sequential circuit has an input ($X$) and an output ($Z$). The output is the same as the input was two clock periods previously. For example,**

$X$ = 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 1

$Z$ = 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0

**The first two values of $Z$ are 0. Find a Mealy state graph and table for the circuit.**

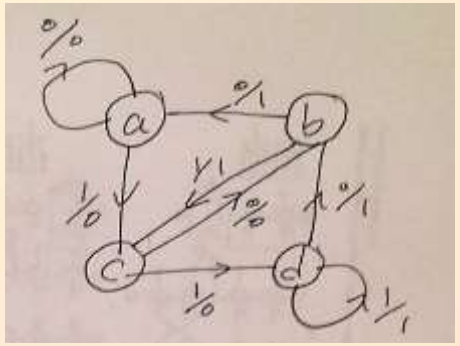



The direction of data is left to right
It could be right to left too.

| A | B | |
|---|---|---|
| 0 | 0 | a |
| 0 | 1 | b |
| 1 | 0 | c |
| 1 | 1 | d |

| n | n+1 X=0 | n+1 X=1 | Z X=0 | Z X=1 |
|---|---|---|---|---|
| a | a | c | 0 | 0 |
| b | a | c | 1 | 1 |
| c | b | d | 0 | 0 |
| d | b | d | 1 | 1 |



# THANK YOU