江西理工大学
Jiangxi University of Science and Technology
信息工程学院
School of information engineering

Dr Ata Jahangir Moshayedi

Digital System Design

Clip Lecture series

Spring_2020

Prof Associate , **School of information engineering Jiangxi university of science and technology, China**
EMAIL: ajm@jxust.edu.cn

Jiangxi University of Science and Technology

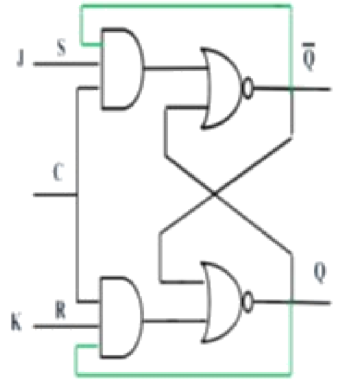
State Tables / State Diagrams and Finite-State Machines (FSMs)

All Flip-Flop Characteristic Tables

Table 5.1
Flip-Flop Characteristic Tables

***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

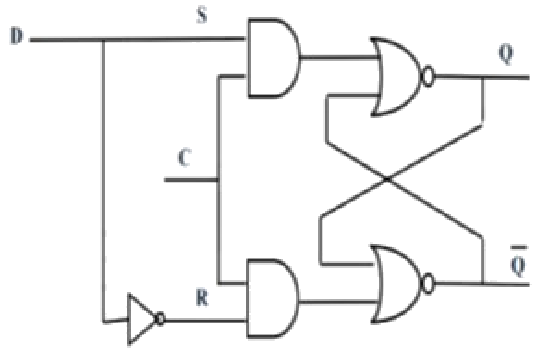


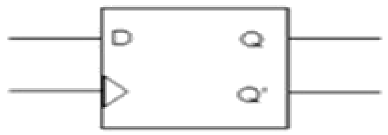
$\boxed{Q^* = J.\bar{Q} + \bar{K}.Q}$

$\boxed{Q^* = D}$

***D* Flip-Flop**

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

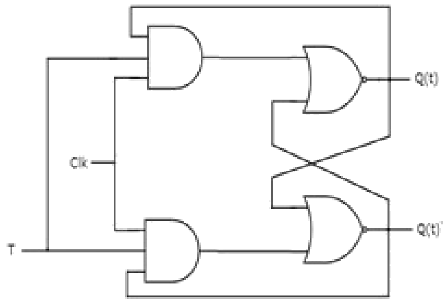




(b) Graphic Symbol

***T* Flip-Flop**

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



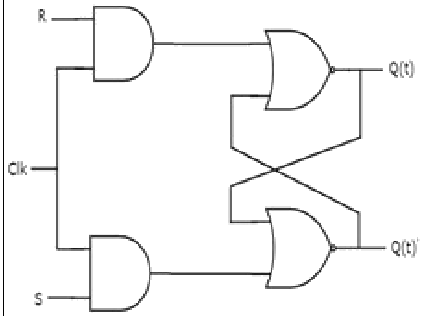
$Q(t+1)=T \oplus Q(t)$

SR FF

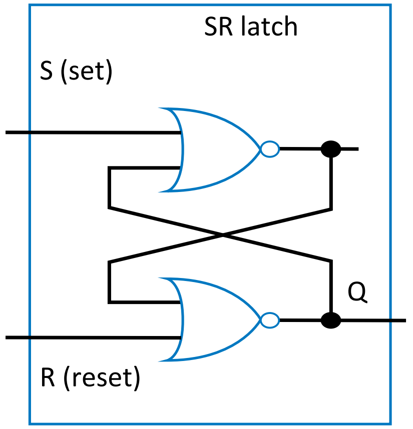
S	R	Q*
0	0	Q
0	1	0
1	0	1
1	1	X

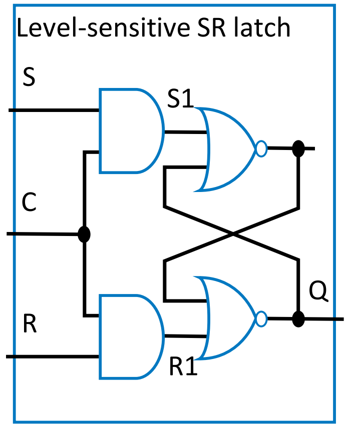
$\boxed{Q^* = S + \bar{R}.Q}$

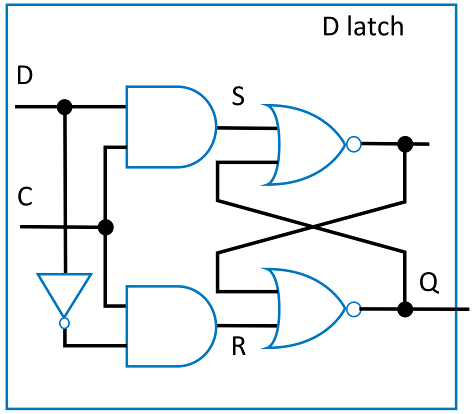
$S.R = 0$

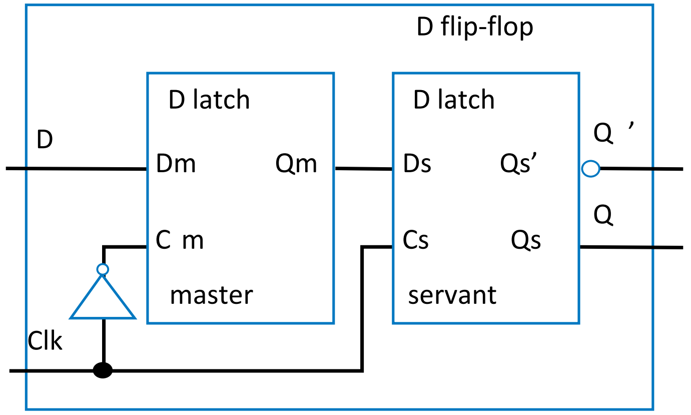


Bit Storage Summary









Feature: S=1 sets Q to 1, R=1 resets Q to 0. Problem: SR=11 yield undefined Q.

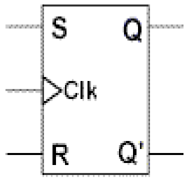
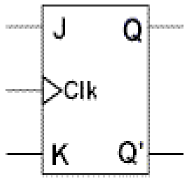
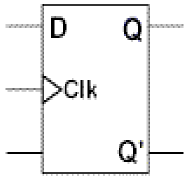
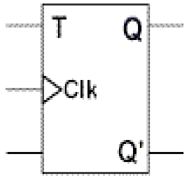
Feature: S and R only have effect when C=1. We can design outside circuit so SR=11 never happens when C=1. Problem: avoiding SR=11 can be a burden.

Feature: SR can't be 11 if D is stable before and while C=1, and will be 11 for only a brief glitch even if D changes while C=1. Problem: C=1 too long propagates new values through too many latches: too short may not enable a store.

Feature: Only loads D value present at rising clock edge, so values can't propagate to other flip-flops during same clock cycle. Tradeoff: uses more gates internally than D latch, and requires more external gates than SR – but gate count is less of an issue today.

- We considered increasingly better bit storage until we arrived at the robust D flip-flop bit storage

Summary of the Types of Flip-flop Behavior

FLIP-FLOP NAME	FLIP-FLOP SYMBOL	CHARACTERISTIC TABLE	CHARACTERISTIC EQUATION	EXCITATION TABLE
SR			$Q_{(next)} = S + R'Q$ $SR = 0$	
JK			$Q_{(next)} = JQ' + K'Q$	
D			$Q_{(next)} = D$	
T			$Q_{(next)} = TQ' + T'Q$	

S	R	$Q_{(next)}$
0	0	Q
0	1	0
1	0	1
1	1	?

Q	$Q_{(next)}$	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

J	K	$Q_{(next)}$
0	0	Q
0	1	0
1	0	1
1	1	Q'

Q	$Q_{(next)}$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

D	$Q_{(next)}$
0	0
1	1

Q	$Q_{(next)}$	D
0	0	0
0	1	1
1	0	0
1	1	1

T	$Q_{(next)}$
0	Q
1	Q'

Q	$Q_{(next)}$	T
0	0	0
0	1	1
1	0	1
1	1	0

State Tables and State Diagrams

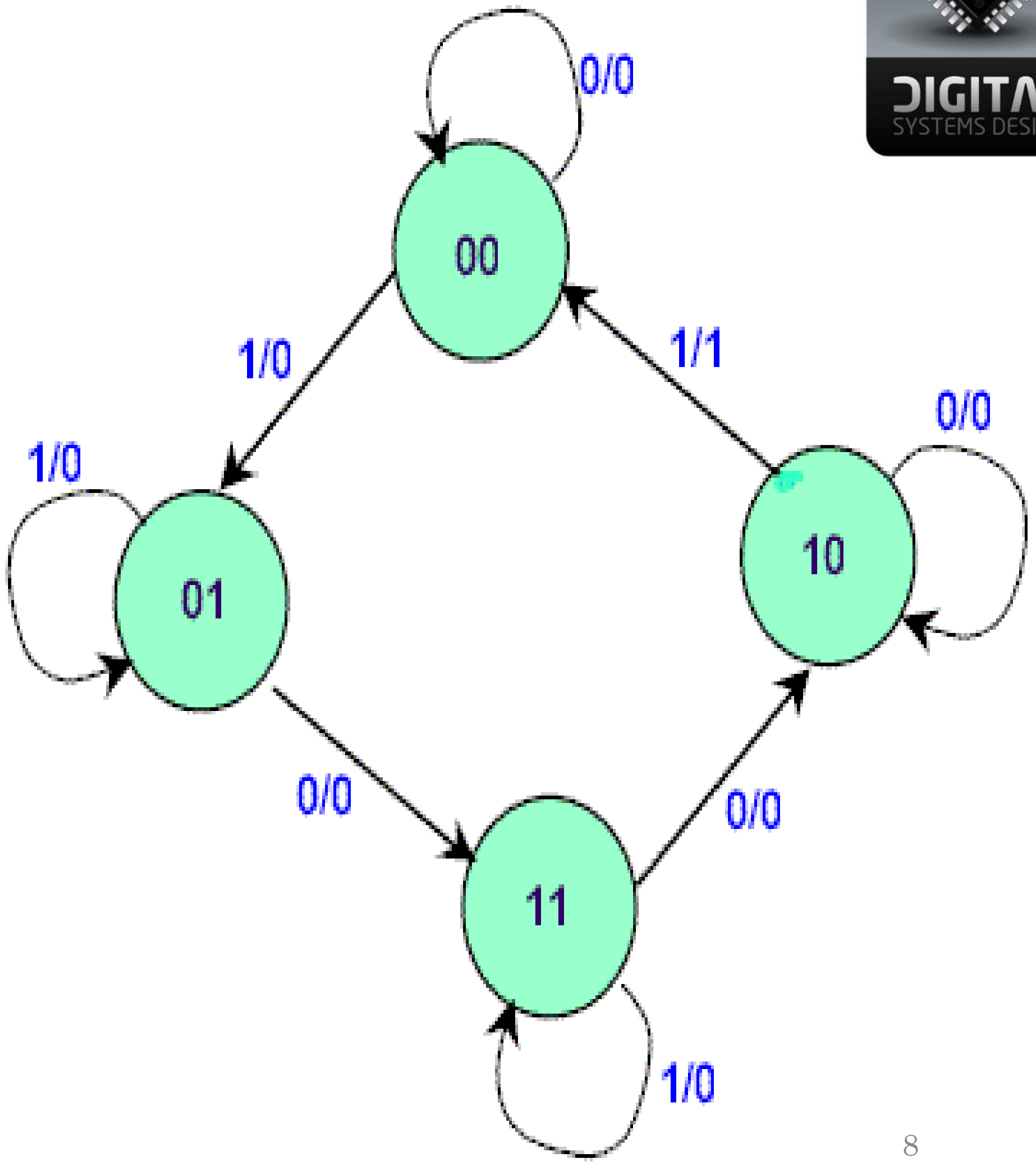
- We have examined a general model for sequential circuits. In this model the effect of all previous inputs on the outputs is represented by a state of the circuit. Thus, the output of the circuit at any time depends upon its current state and the input. These also determine the next state of the circuit.
- *The relationship that exists among the inputs, outputs, present states and next states can be specified by either the **state table** or the **state diagram**.*

State Table

- The state table representation of a sequential circuit consists of three sections labeled *present state*, *next state* and *output*.
- *The present state designates the state of flip-flops before the occurrence of a clock pulse.*
- The next state shows the states of flip-flops after the clock pulse, and the output section lists the value of the output variables during the present state.

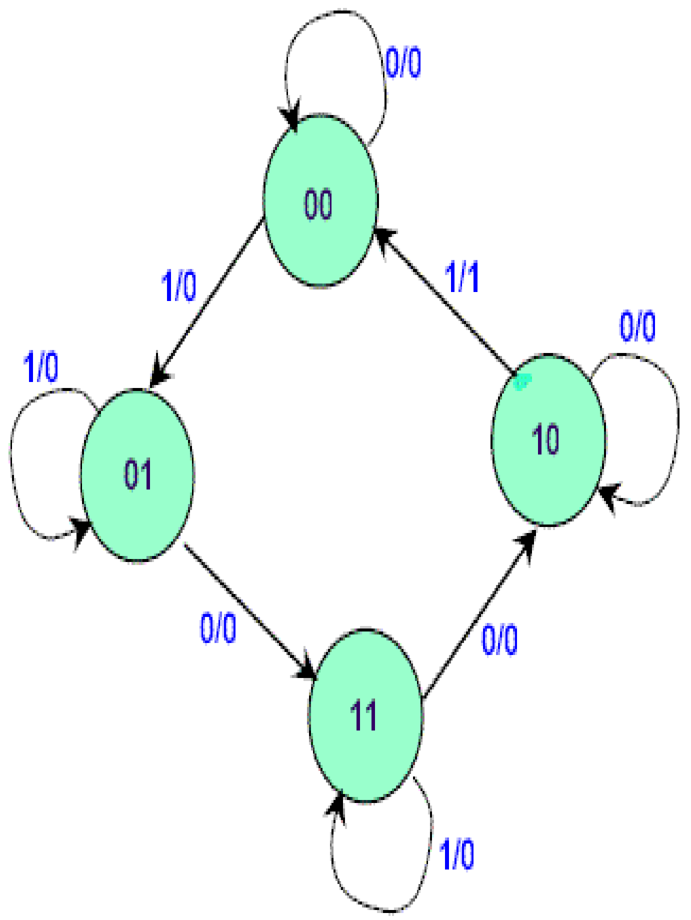
State Diagram

- In addition to graphical symbols, tables or equations, flip-flops can also be represented graphically by a state diagram.
- In this diagram, a state is represented by a circle, and the transition between states is indicated by directed lines (or arcs) connecting the circles.
- **An example of a state diagram is shown in Figure below.**



State Diagram

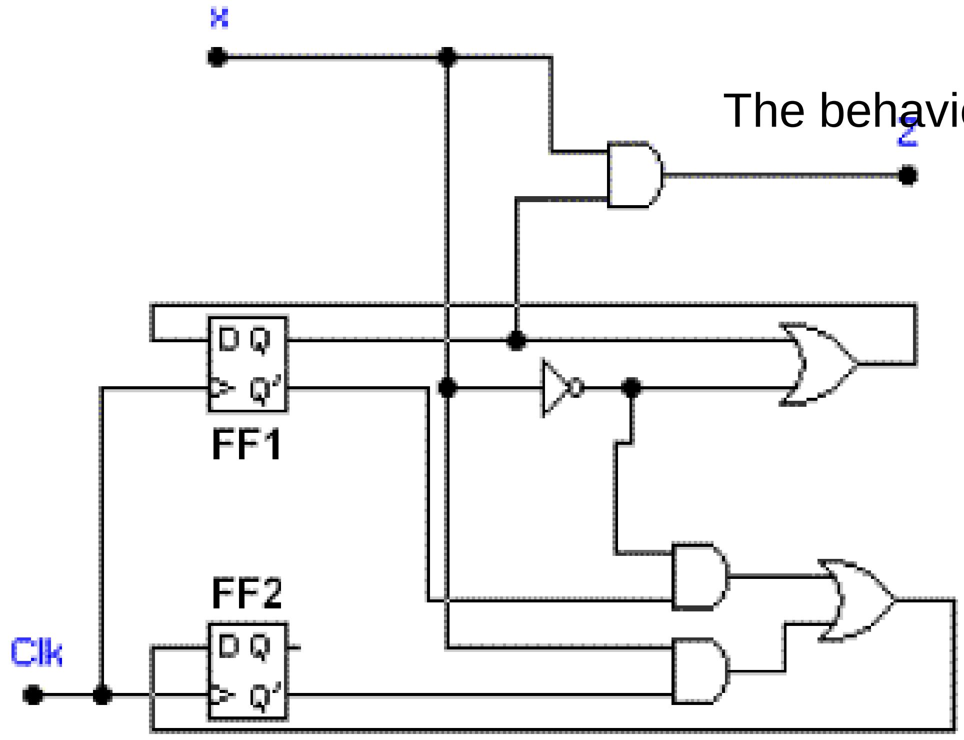
- The binary number inside each circle identifies the state the circle represents. The directed lines are labeled with two binary numbers separated by a slash (/). The input value that causes the state transition is labeled first. The number after the slash symbol / gives the value of the output.
- **For example, the directed line from state 00 to 01 is labeled 1/0, meaning that, if the sequential circuit is in a present state and the input is 1, then the next state is 01 and the output is 0. If it is in a present state 00 and the input is 0, it will remain in that state.**
- **A directed line connecting a circle with itself indicates that no change of state occurs. The state diagram provides exactly the same information as the state table and is obtained directly from the state table.**



Example:

- Consider a sequential circuit shown in Figure.
- It has one input x, one output Z and two state variables Q_1Q_2

(thus having four possible present states 00, 01, 10, 11)



The behavior of the circuit is determined by the following Boolean expressions:

$$Z = x * Q_1$$

$$D_1 = x' + Q_1$$

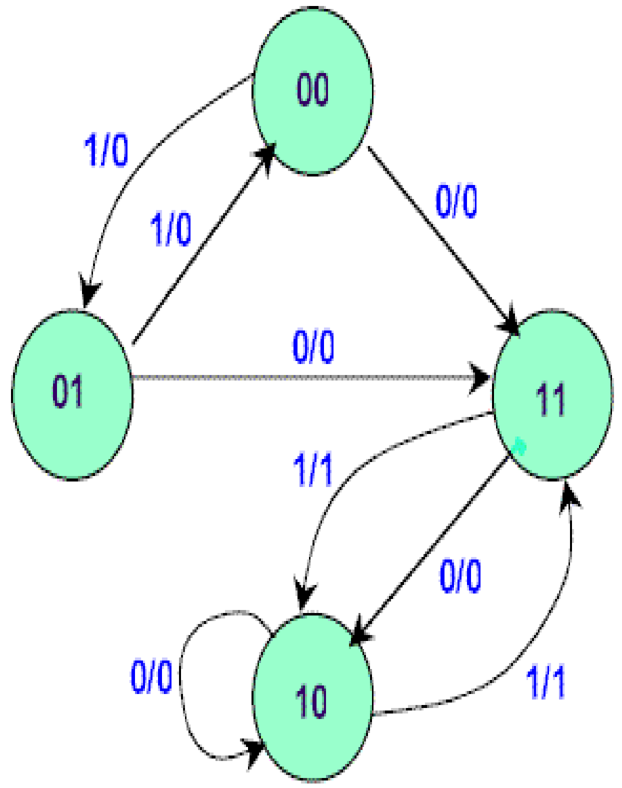
$$D_2 = x * Q_2' + x' * Q_1'$$

This example is taken from P. K. Lala, *Practical Digital Logic Design and Testing*, Prentice Hall, 1996, p.155

Example:

- These equations can be used to form the state table. Suppose the present state (i.e. Q1Q2) = 00 and input x = 0.

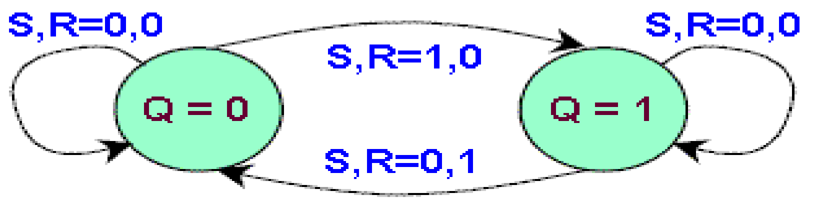
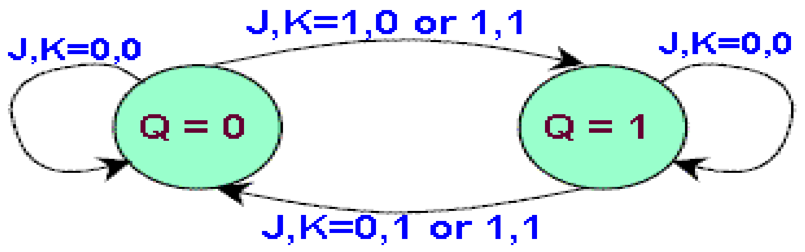
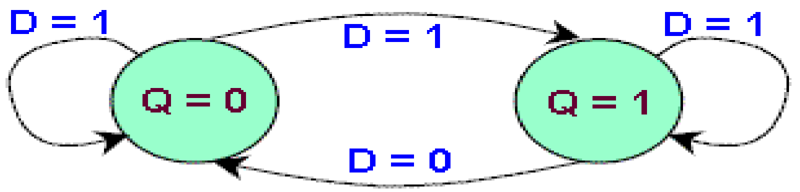
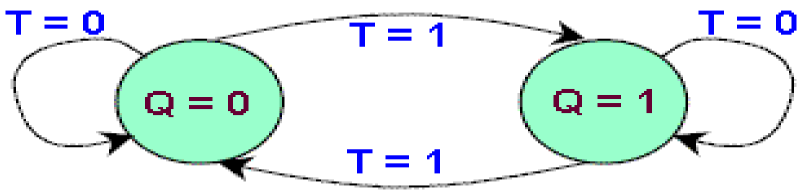
  Under these conditions, we get Z = 0, D1 = 1, and D2 = 1.
- Thus the next state of the circuit D1D2 = 11, and this will be the present state after the clock pulse has been applied.
- The output of the circuit corresponding to the present state Q1Q2 = 00 and x = 1 is Z = 0.
- This data is entered into the state table as shown in Table 2. The state diagram for the sequential circuit is shown in Figure.



Analysis of Sequential circuits

In order to analysis the sequential circuits we should draw the table or graph for the sequential time series of Inputs and outputs and internal states
the sequential circuits we should follow this rules
For each FF we should put the name
The current state is (t)
The next state is (t+1)
The Q and Q' out puts are shows with the same indication

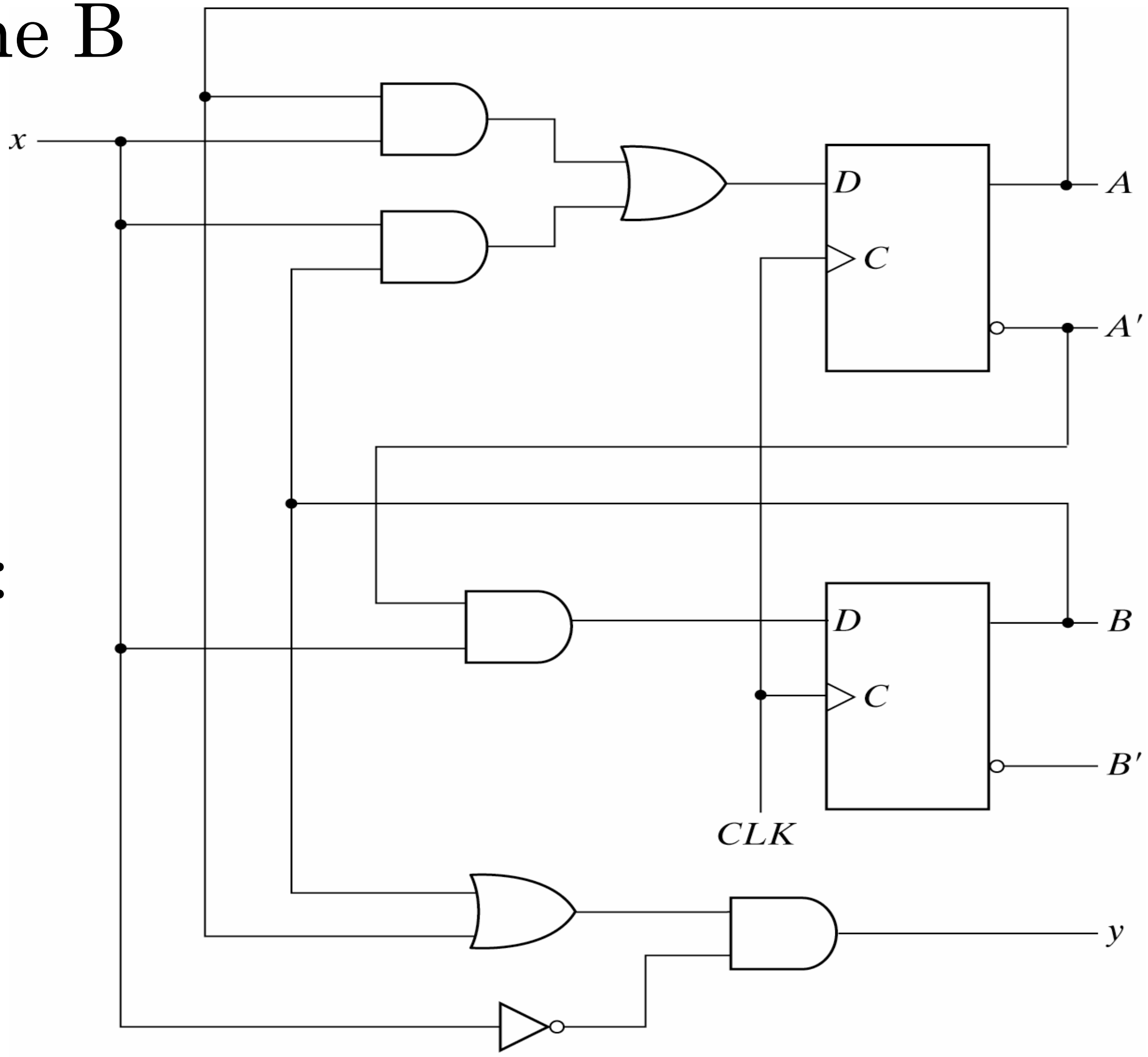
State Diagrams of Various Flip-flops

NAME	STATE DIAGRAM
SR	
JK	 J,K=1,0 or 1,1 J,K=0,0 J,K=0,0 Q = 0 Q = 1 J,K=0,1 or 1,1
D	
T	

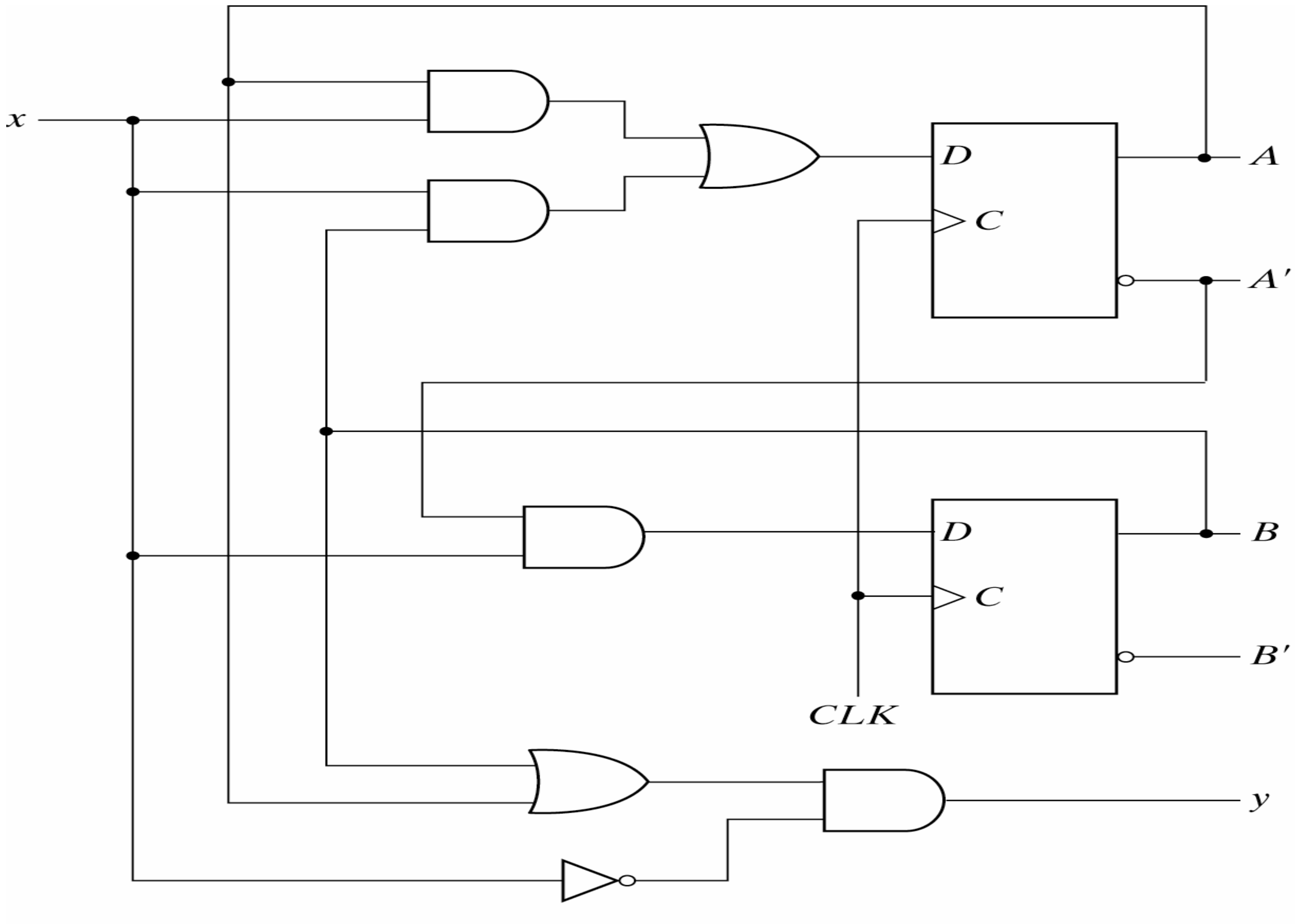
State table and state diagram represents exactly the same information.

Example of sequential circuits

- If we call the first FF A and second one B
  - A(t+1) = A(t) x(t) + B(t) x(t)
  - B(t+1) = A'(t) x(t)
- In the simplest form we can have
  - A(t+1) = Ax + Bx
  - B(t+1) = A'x
- Then the ckt out put can be shown as:
  - y(t) = [A(t) + B(t)] x'(t)
  - y = (A + B) x'



State table



Present		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

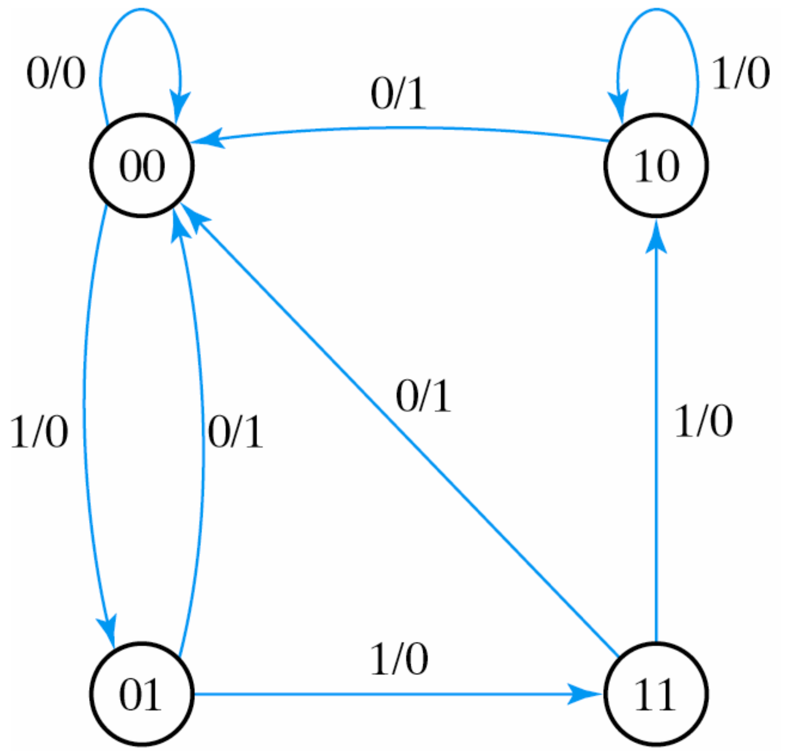
State Diagram

- The change between stats, input and Output of a sequential circuit can be shown in diagram called state diagram
- This diagram can be draw form state table or directly form sequential circuits
- The vertices from one node to other can be labed with input/out put

State diagram

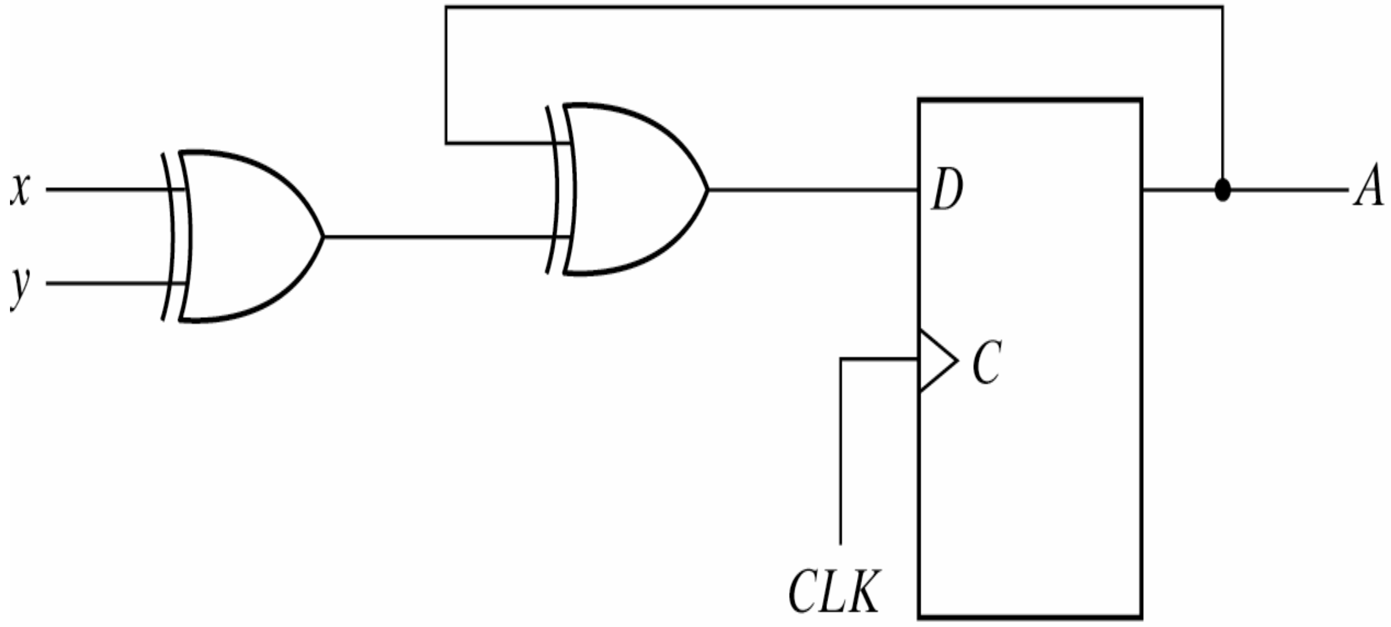
- Draw the state diagram form state table

Present		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

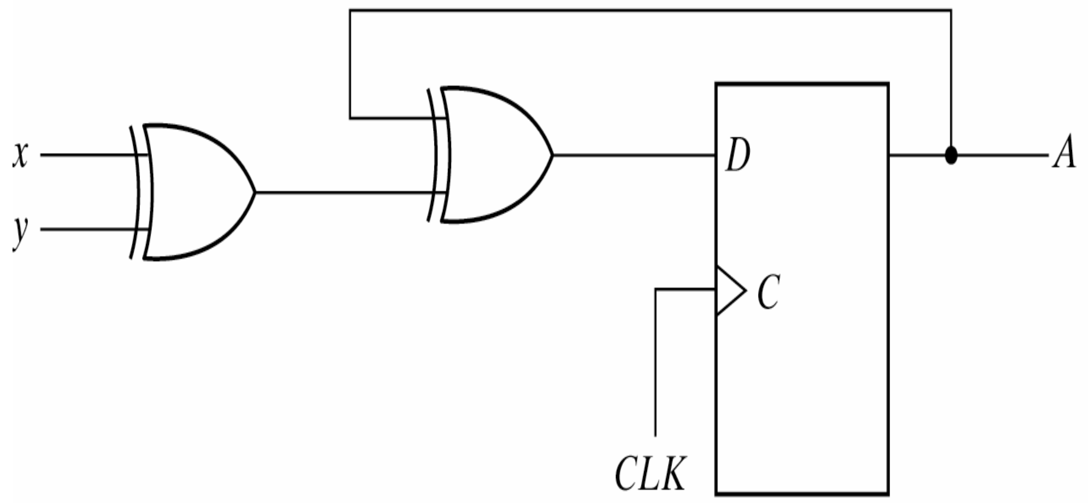


Example

The figure is sequential circuit, For the mentioned diagram show the state table and state diagram



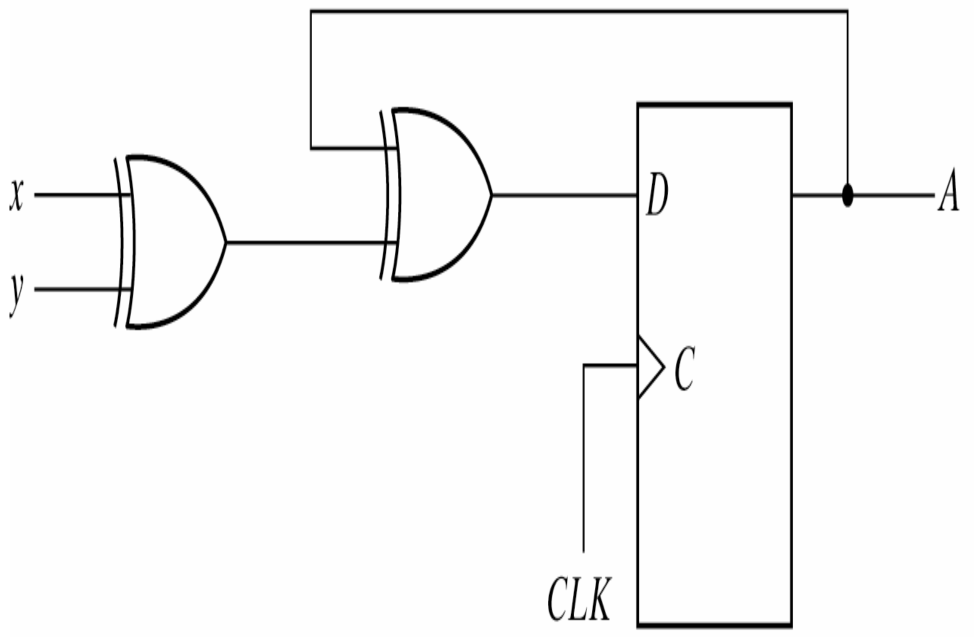
Example

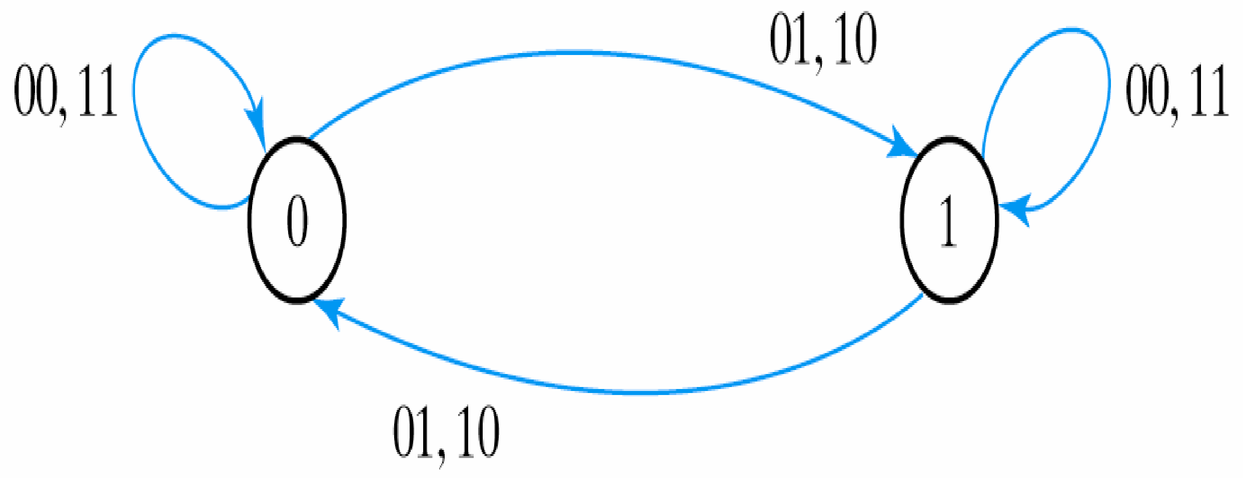


Present state	Inputs		Next state
A	x	y	A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Example

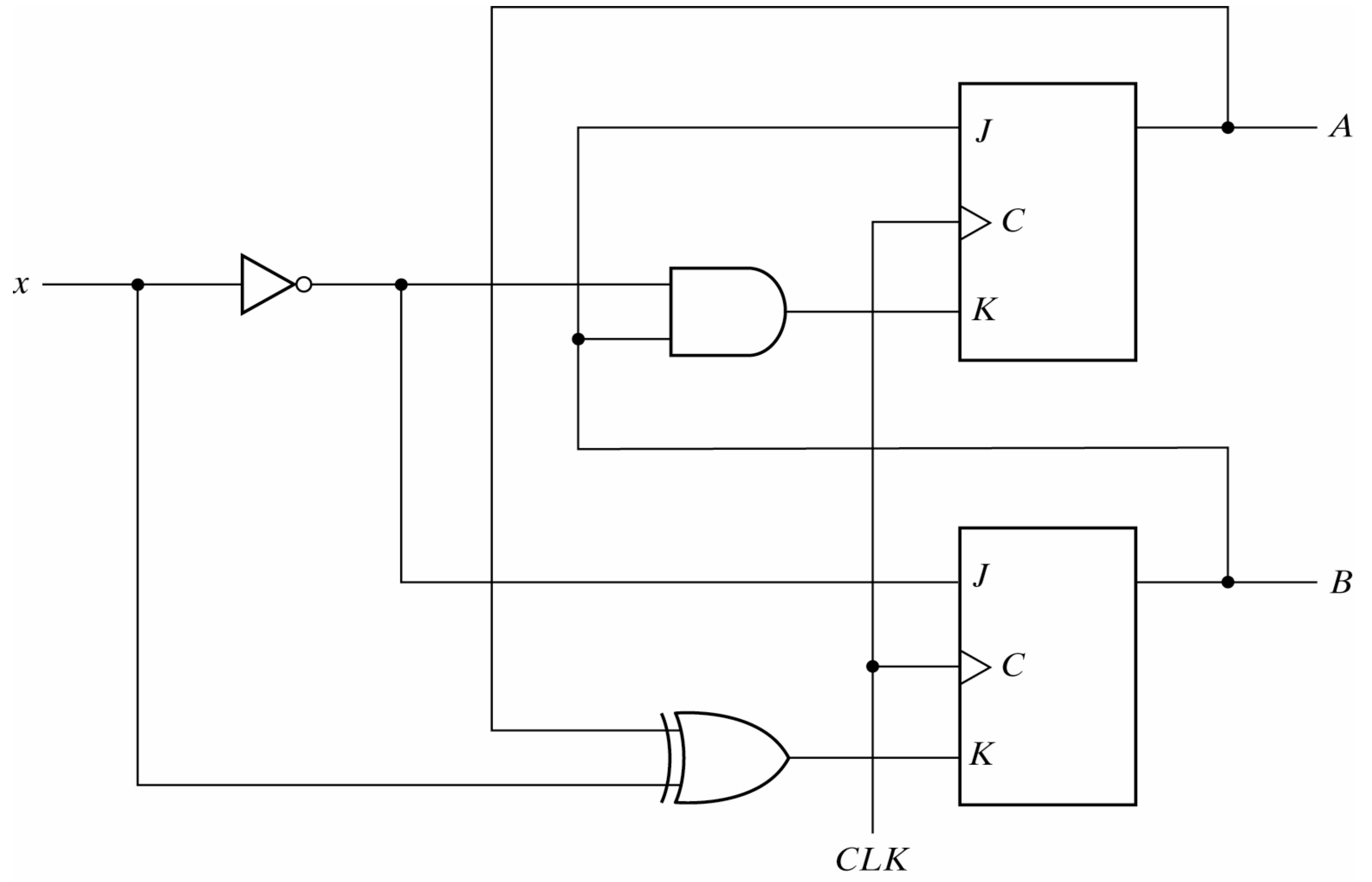
- State Diagram





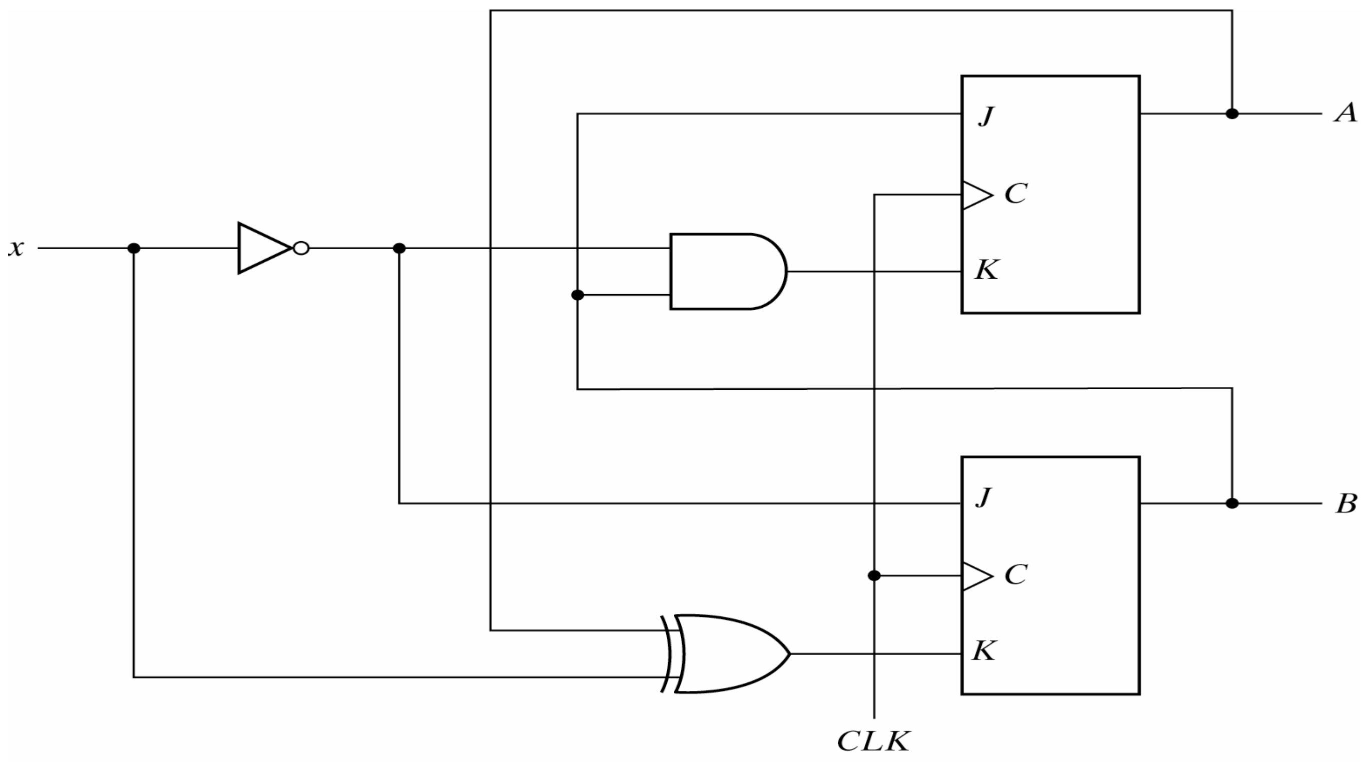
Example

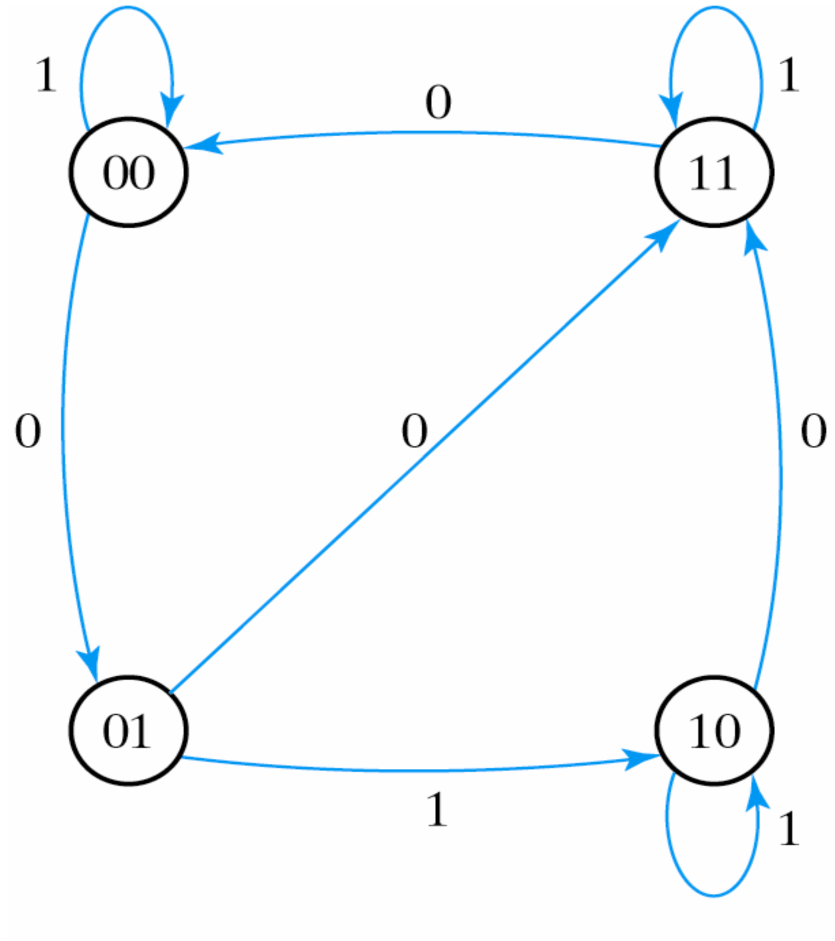
- The figure is sequential circuit, For the mentioned diagram show the state diagram



Example(continue)

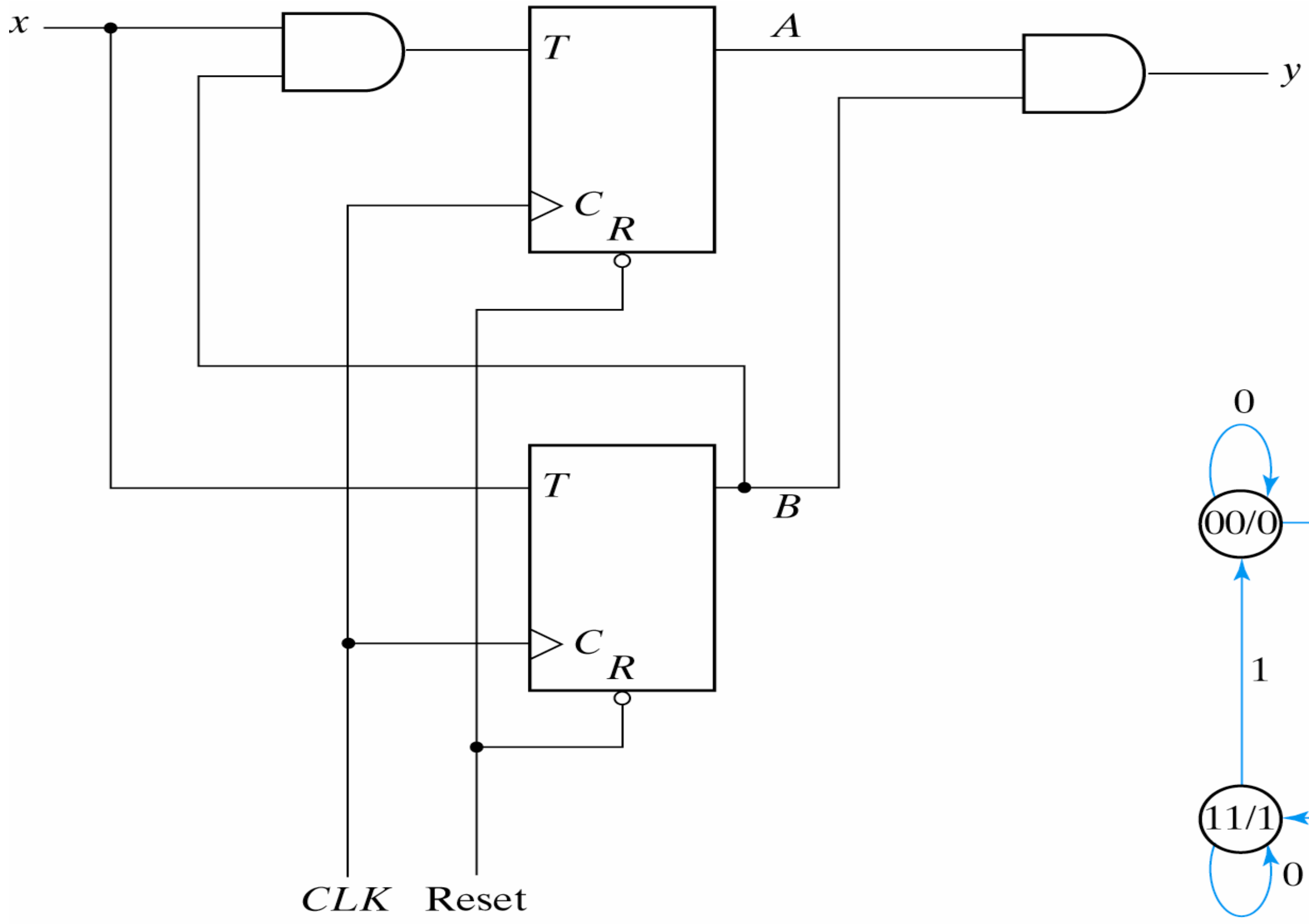
- State Diagram

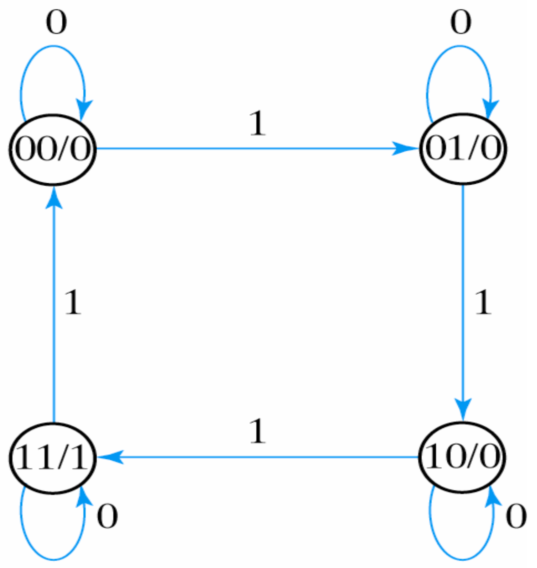




Example

- The figure is sequential circuit, For the mentioned diagram show the state diagram





State table of FF

- This table shows the situation of input ff when the present and future status of FF are specified
- The excitation table of RS FF are as follows

Q(t)	Q(t+1)	S	R	
0	0	0	X	0 0 0 1
0	1	1	0	1 0
1	0	0	1	0 1
1	1	X	0	0 0 1 0

Excitation table of JK_FF

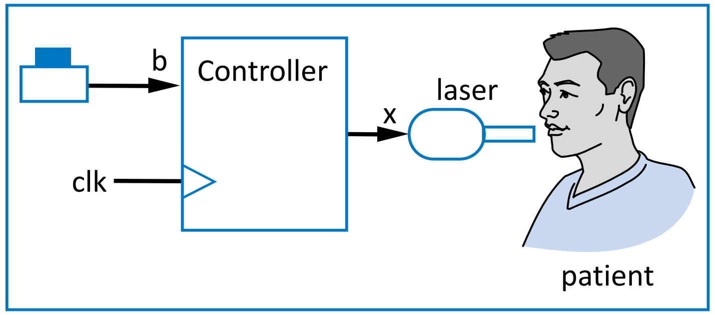
Q(t)	Q(t+1)	S	R
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

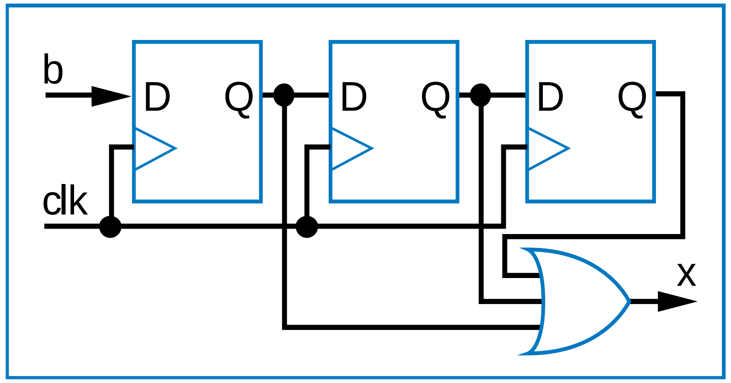
Q(t)	Q(t+1)	R
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table of T_FF

Finite-State Machines (FSMs) and Controllers

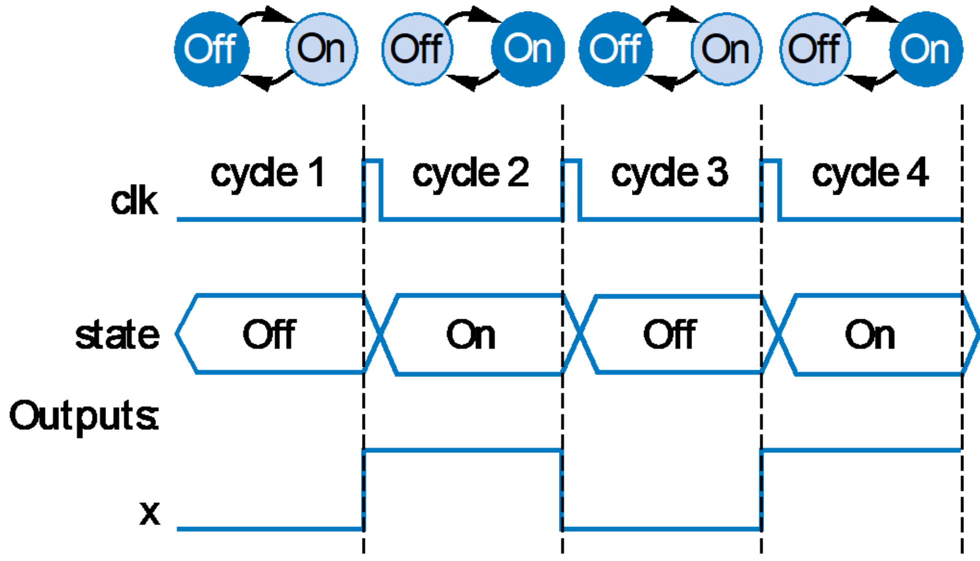
- Want sequential circuit with particular behavior over time
- Example: Laser timer
  - Push button: x=1 for 3 clock cycles
  - How? Let's try three flip-flops
    - b=1 gets stored in first D flip-flop
    - Then 2nd flip-flop on next cycle, then 3rd flip-flop on next
    - OR the three flip-flop outputs, so x should be 1 for three cycles

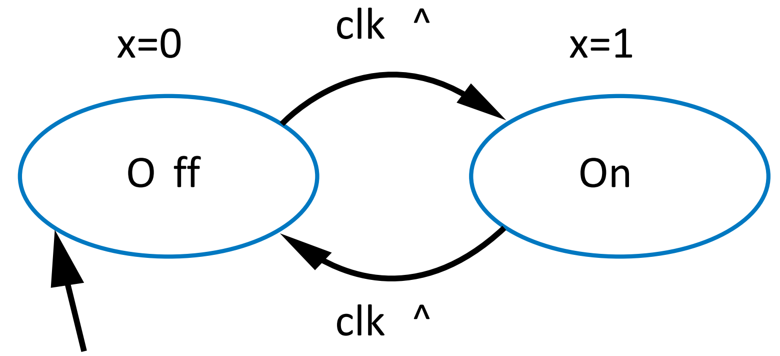




Describing Behavior of Sequential Circuit: FSM

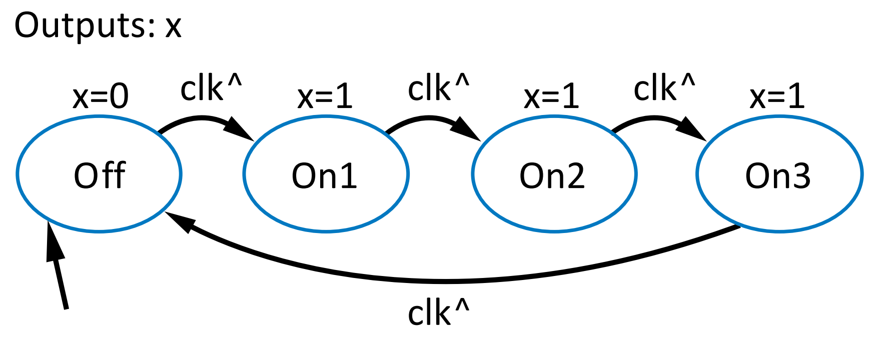
- Finite-State Machine (FSM)
- A way to describe desired behavior of sequential circuit
  - Akin to Boolean equations for combinational behavior
- List states, and transitions among states
  - Example: Make x change toggle (0 to 1, or 1 to 0) every clock cycle
  - Two states: “Off” (x=0), and “On” (x=1)
  - 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)

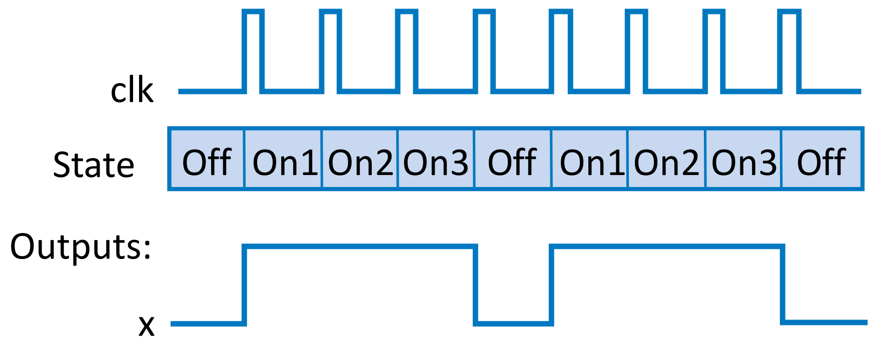




FSM Example: 0,1,1,1,repeat

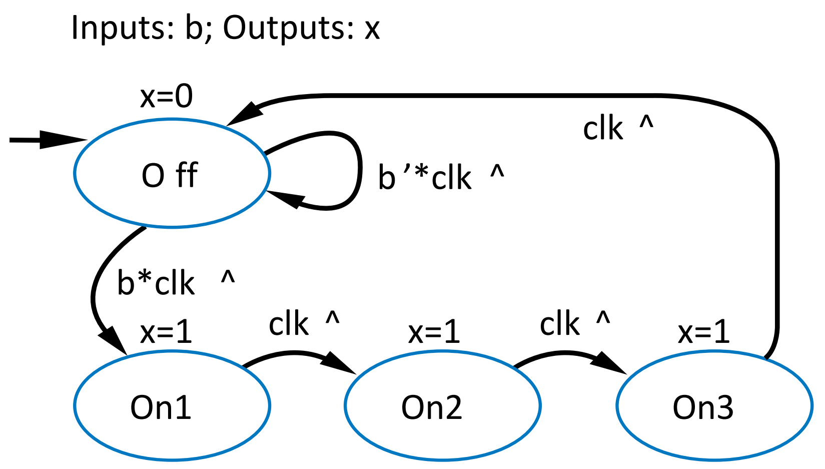
- Want 0, 1, 1, 1, 0, 1, 1, 1, …
  - Each value for one clock cycle
- Can describe as FSM
  - Four states
  - Transition on rising clock edge to next state

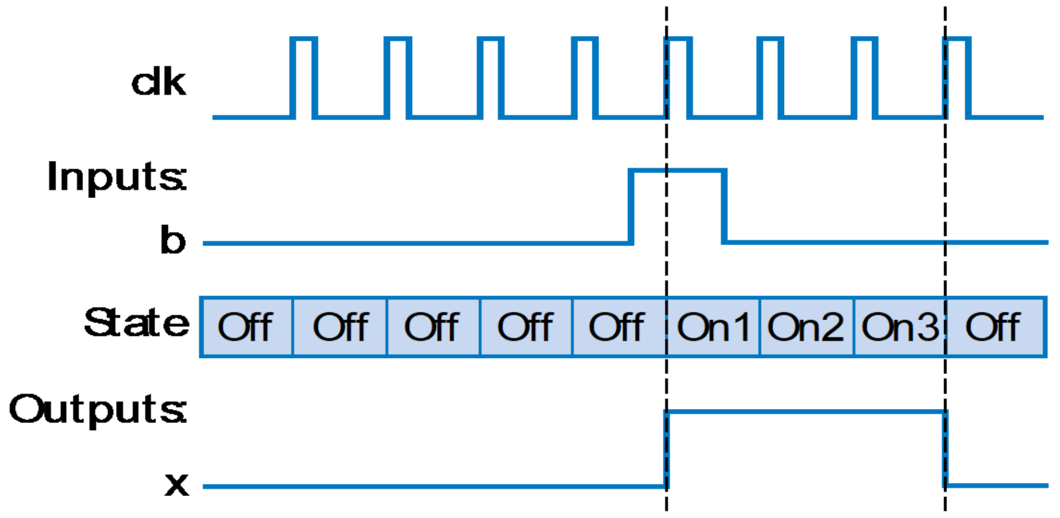




Extend FSM to Three-Cycles High Laser Timer

- Four states
- Wait in "Off" state while b is 0 (b')
- When b is 1 (and 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

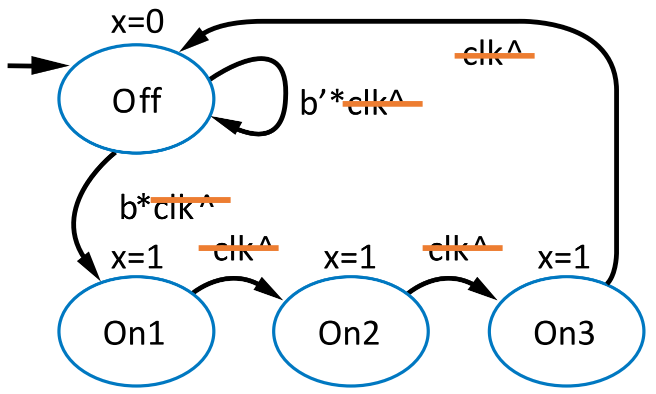




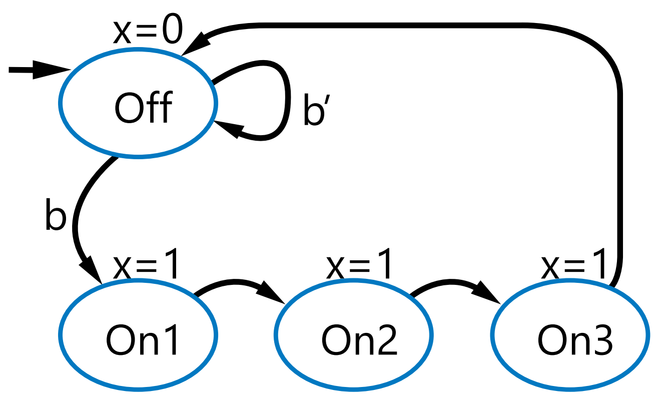
FSM Simplification: Rising Clock Edges Implicit

- Showing rising clock on every transition: cluttered
  - Make implicit -- assume every edge has rising clock, even if not shown
  - What if we wanted a transition *without* a rising edge
    - We don't consider such asynchronous FSMs -- less common, and advanced topic
    - Only consider ***synchronous*** FSMs -- rising edge on *every* transition

Inputs: b; Outputs: x



Inputs: b; Outputs: x

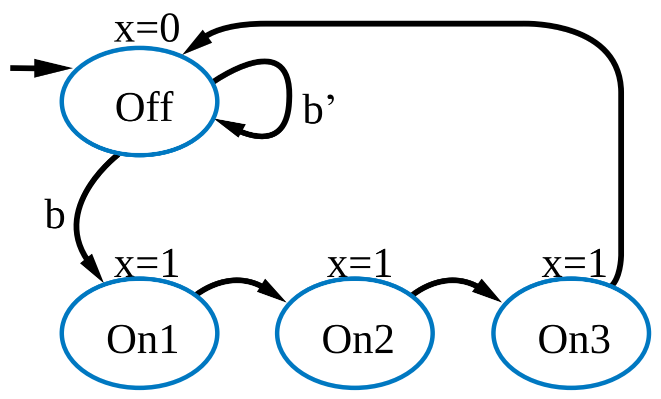


Note: Transition with no associated condition thus transistions to next state on next clock cycle

FSM Definition

- FSM consists of
  - 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

Inputs: b; Outputs: x

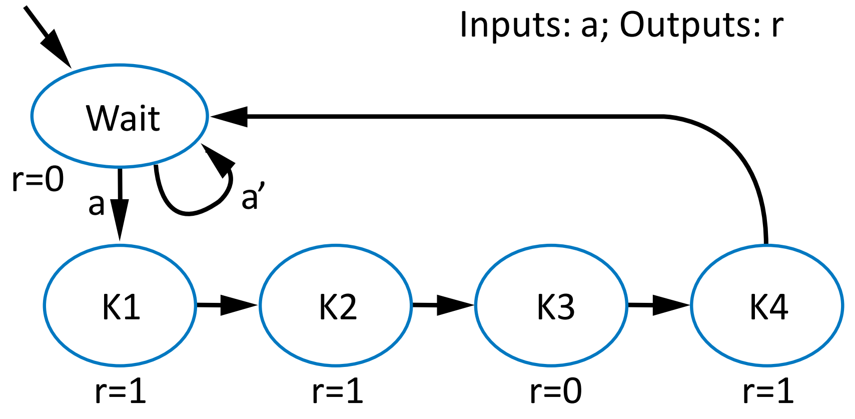


We often draw FSM graphically, known as ***state diagram***

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

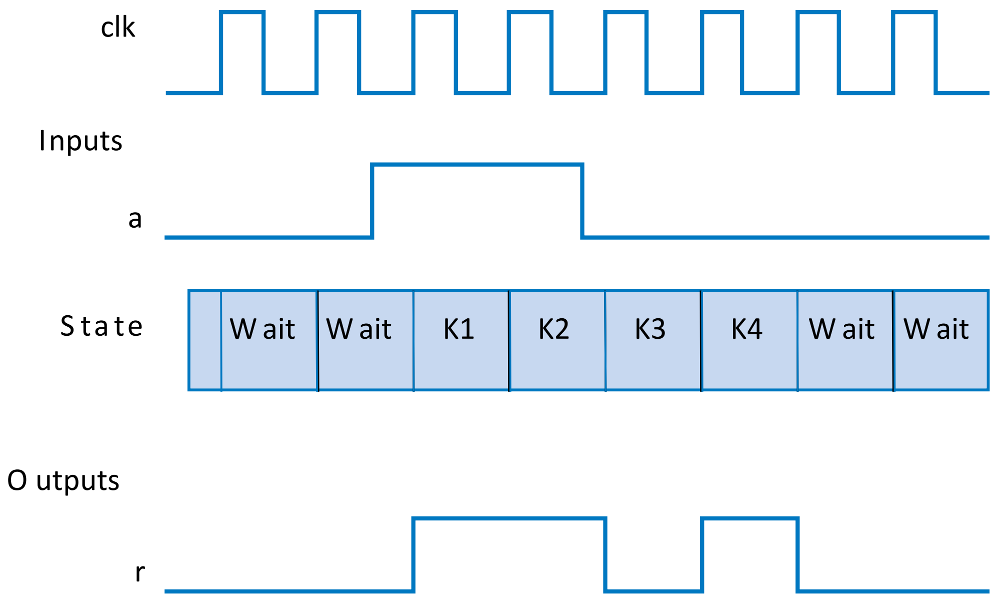
FSM Example: Secure Car Key

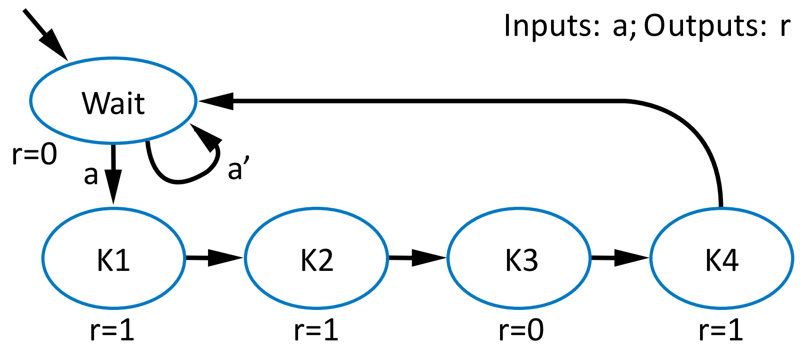
- Many new car keys include tiny computer chip
  - When car starts, car's computer (under engine hood) requests identifier from key
  - Key transmits identifier
    - If not, computer shuts off car
- FSM
  - Wait until computer requests ID (a=1)
  - Transmit ID (in this case, 1101)



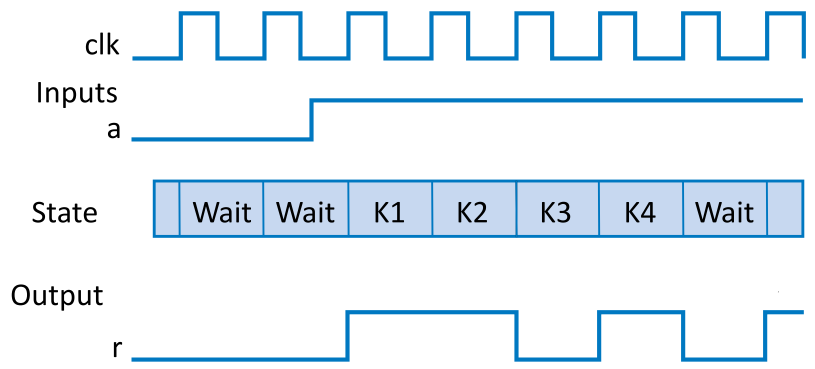
FSM Example: Secure Car Key (cont.)

- Nice feature of FSM
  - Can evaluate output behavior for different input sequence
  - Timing diagrams show states and output values for different input waveforms



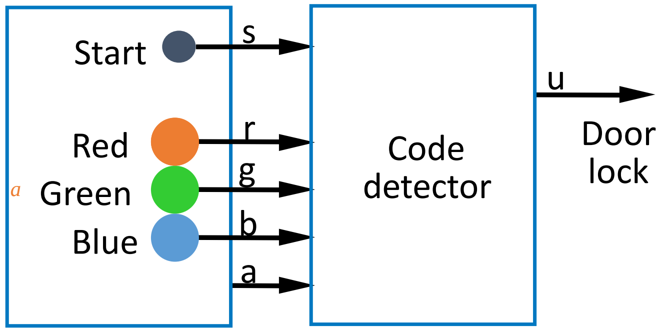


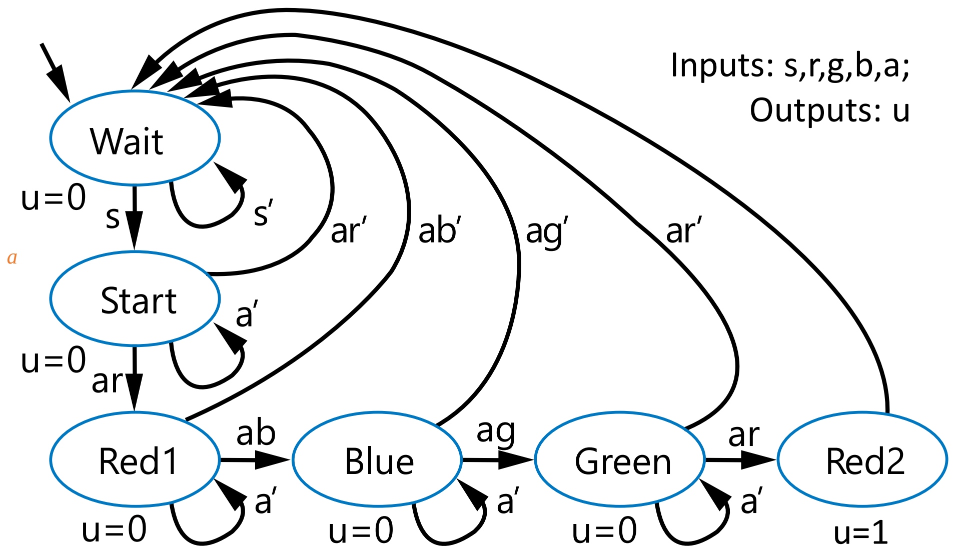
Q: Determine states and r value for given input waveform:



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
  - 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", without opening door

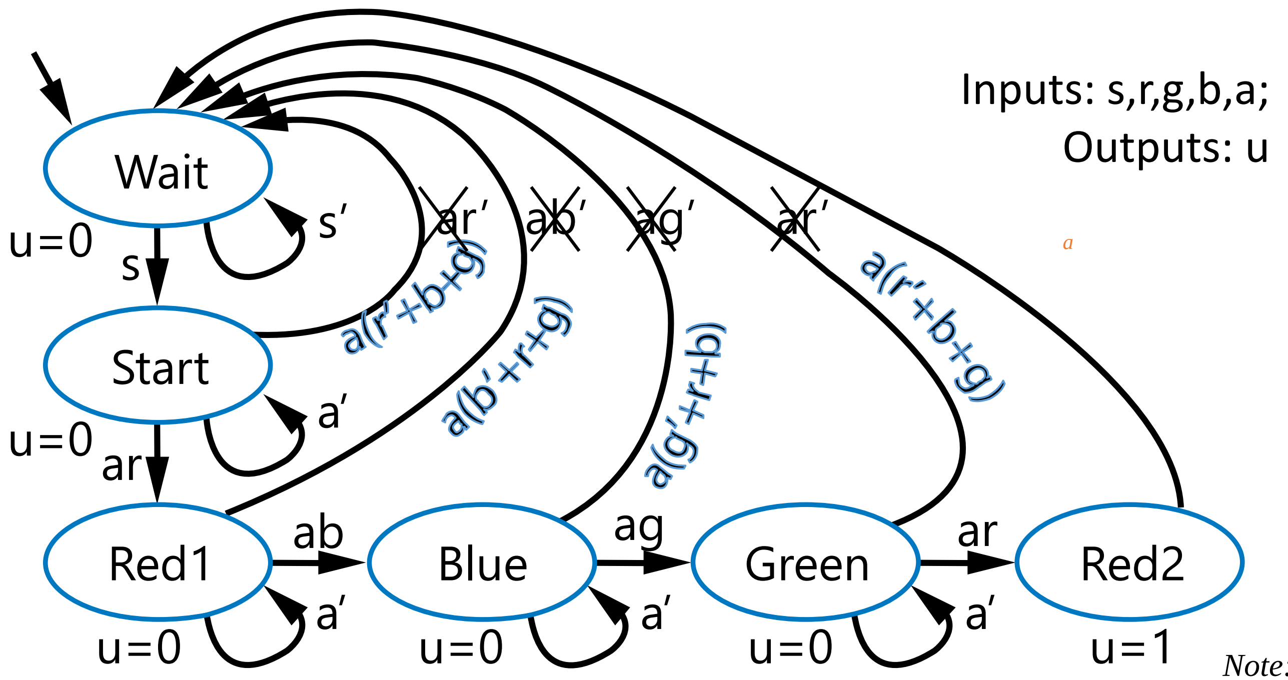




Q: Can you trick this FSM to open the door, without knowing the code?

A: Yes, hold all buttons simultaneously

Improve FSM for Code Detector

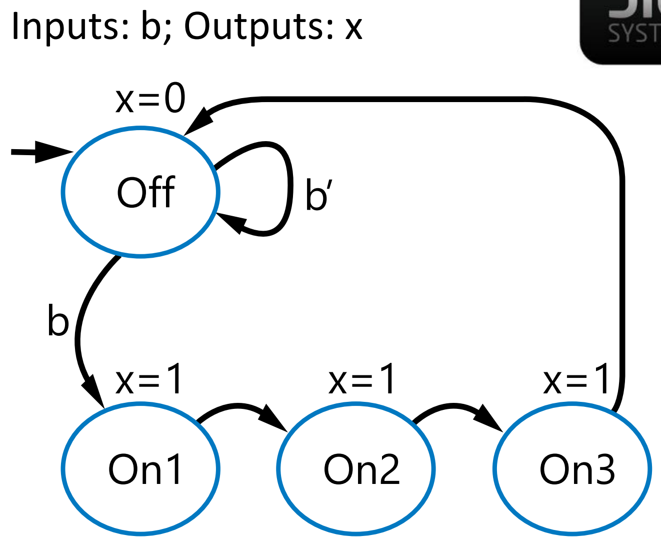


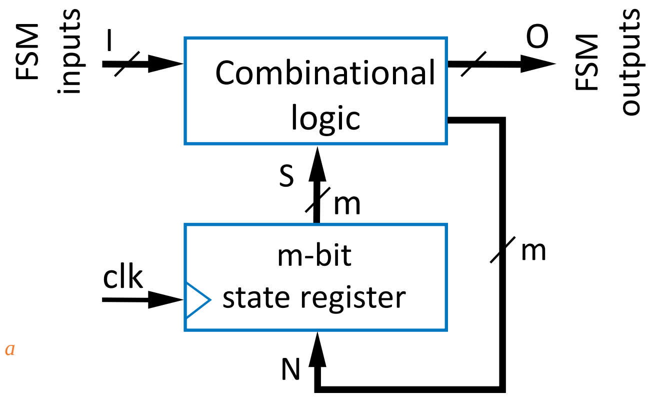
Note: small problem still remains; we'll discuss later

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

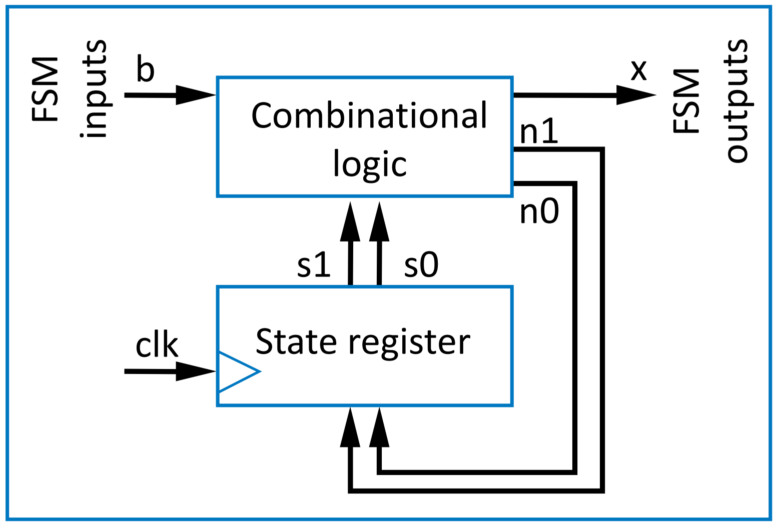
Standard Controller Architecture

- How implement FSM as sequential circuit?
  - Use standard architecture
    - State register -- to store the present state
    - Combinational logic -- to compute outputs, and next state
    - For laser timer FSM
      - 2-bit state register, can represent four states
      - Input b, output x
  - Known as ***controller***





General version



Refrence

- http://osp.mans.edu.eg/cs212/Seq_circuits_analysis.htm