Digital Circuit & Logic Design

(CSPC-201)

Synchronous Sequential Logic

Synchronous Sequential Logic

The digital circuits considered thus far have been combinational—their output depends only and immediately on their inputs—they have no memory, i.e., dependence on past values of their inputs.

Sequential circuits, however, act as storage elements and have memory. They can store, retain, and then retrieve information when needed at a later time.

Sequential Circuits

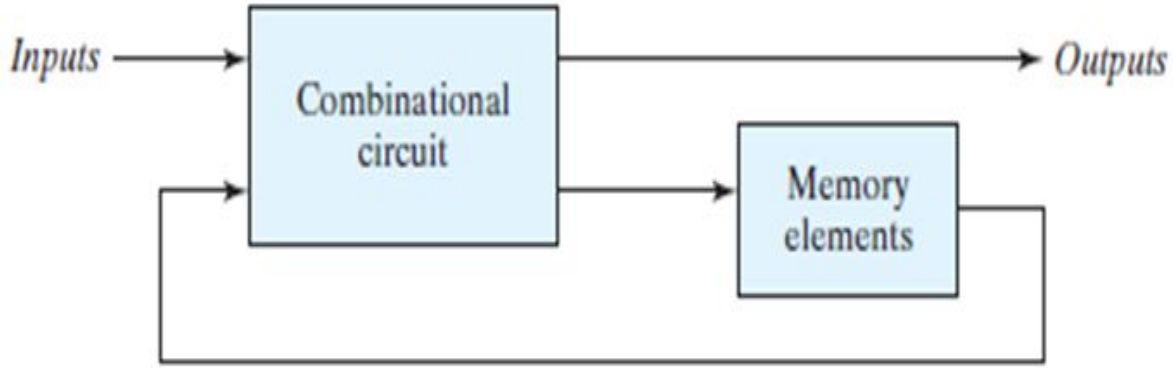
Consists of a combinational circuit to which storage elements are connected to form a feedback path.

The storage elements are devices capable of storing binary information.

The binary information stored in these elements at any given time defines the *state* of the sequential circuit at that time.

The sequential circuit receives binary information from external inputs that, together with the present state of the storage elements, determine the binary value of the outputs.

These external inputs also determine the condition for changing the state in the storage elements.



Thus, *a sequential circuit is specified by a **time sequence of inputs**, **outputs**, and **internal states**.*

Types of Sequential Circuits

There are two main types of sequential circuits, and their classification is a function of the timing of their signals.

Synchronous sequential circuit is a system whose behavior can be defined from the knowledge of its signals at discrete instants of time.

Asynchronous sequential circuit depends upon the input signals at any instant of time and the order in which the inputs change. The storage elements commonly used in asynchronous sequential circuits are time-delay devices.

The storage elements commonly used in asynchronous sequential circuits are time-delay devices. The storage capability of a time-delay device varies with the time it takes for the signal to propagate through the device.

In practice, the internal propagation delay of logic gates is of sufficient duration to produce the needed delay, so that actual delay units may not be necessary. In gate-type asynchronous systems, the storage elements consist of logic gates whose propagation delay provides the required storage. Thus, an asynchronous sequential circuit may be regarded as a combinational circuit with feedback. Because of the feedback among logic gates, an asynchronous sequential circuit may become unstable at times.

Clock Signal Sequential Circuits

A synchronous sequential circuit employs signals that affect the storage elements at only discrete instants of time. Synchronization is achieved by a timing device called a *clock generator*, which provides a clock signal (identifiers clock and clk) having the form of a periodic train of clock pulses.

The clock pulses are distributed throughout the system in such a way that storage elements are affected only with the arrival of each pulse. In practice, the clock pulses determine when computational activity will occur within the circuit, and other signals (external inputs and otherwise) determine what changes will take place affecting the storage elements and the outputs.

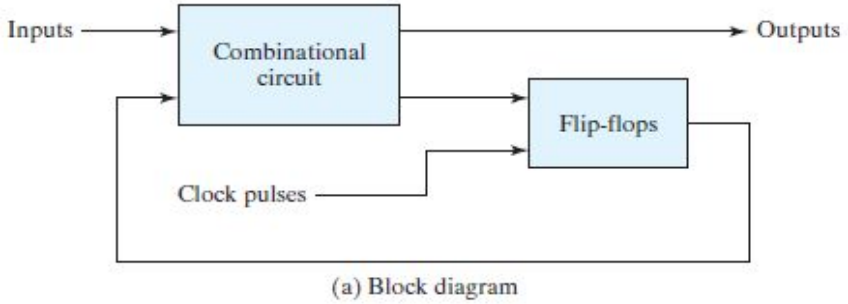
Synchronous sequential circuits that use clock pulses to control storage elements are called *clocked sequential circuits* and are the type most frequently encountered in practice. They are called synchronous circuits because the activity within the circuit and the resulting updating of stored values is synchronized to the occurrence of clock pulses.

Storage Elements: Flip-flops

The storage elements (memory) used in clocked sequential circuits are called ***flip-flops***.

A flip-flop is a binary storage device capable of storing one bit of information. In a stable state, the output of a flip-flop is either 0 or 1.

A sequential circuit may use many flip-flops to store as many bits as necessary.



(a) Block diagram

(b) Timing diagram of clock pulses

Flip-flops

Propagation delays play an important role in determining the minimum interval between clock pulses that will allow the circuit to operate correctly.

A change in state of the flip-flops is initiated only by a clock pulse transition(when signals changes from 0 to 1 or vice versa).

When a clock pulse is not active, the feedback loop between the value stored in the flip-flop and the value formed at the input to the flip-flop is effectively broken because the flipflop outputs cannot change even if the outputs of the combinational circuit driving their inputs change in value. Thus, the transition from one state to the next occurs only at predetermined intervals dictated by the clock pulses.

Storage Elements: Latches

A storage element in a digital circuit can maintain a binary state indefinitely (as long as power is delivered to the circuit), until directed by an input signal to switch states. The major differences among various types of storage elements are in the number of inputs they possess and in the manner in which the inputs affect the binary state.

Storage elements that operate with signal levels (rather than signal transitions) are referred to as latches; those controlled by a clock transition are flip-flops.

Flip-flops vs. Latches

Latches are said to be level sensitive devices; flip-flops are edge-sensitive devices.

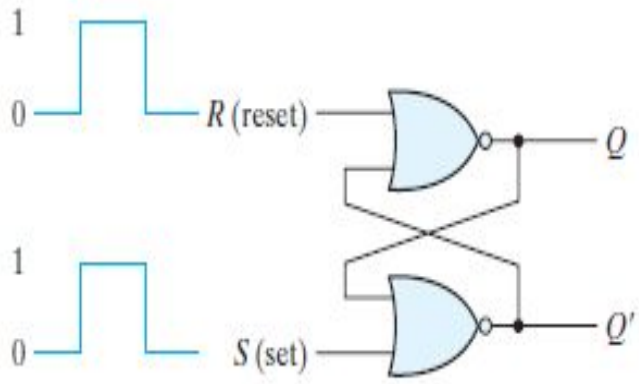
The two types of storage elements are related because latches are the basic circuits from which all flip-flops are constructed.

Although latches are useful for storing binary information and for the design of asynchronous sequential circuits, they are not practical for use as storage elements in synchronous sequential circuits. However, studied as the building blocks of flip flops.

SR Latch

The SR latch is a circuit with two cross-coupled NOR gates or two cross-coupled NAND gates, and two inputs labeled S for set and R for reset.

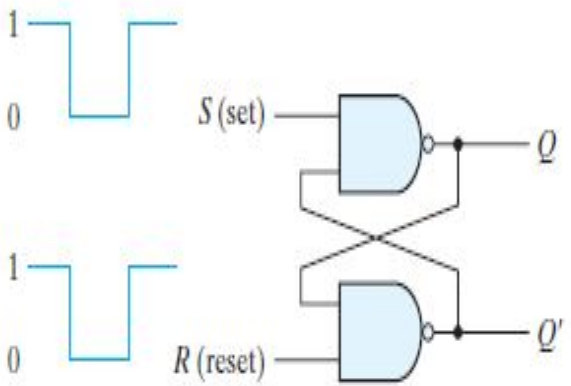
The latch has two useful states. When output Q = 1 and Q' = 0, the latch is said to be in the set state. When Q = 0 and Q' = 1, it is in the reset state. Outputs Q and Q' are normally the complement of each other.



(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



(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

However, when both inputs are equal to 1 at the same time, a condition in which both outputs are equal to 0 (rather than be mutually complementary) occurs. If both inputs are then switched to 0 simultaneously, the device will enter an unpredictable or undefined state or a metastable state.

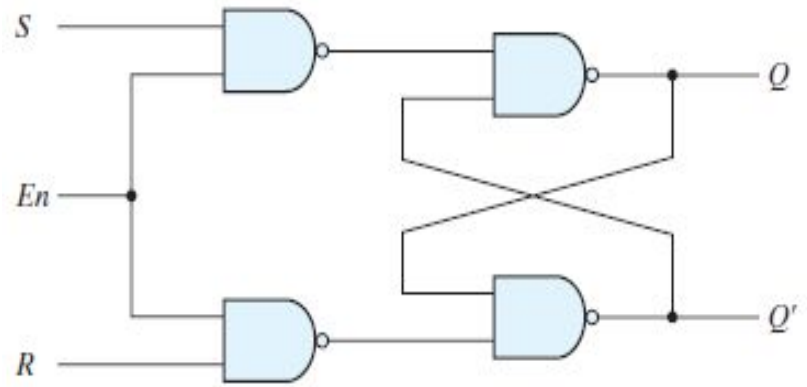
Under normal conditions, both inputs of the latch remain at 0 unless the state has to be changed.

In NOR latch, the condition of making sure that 1’s are not applied to both inputs simultaneously is avoided. The condition that is forbidden for the NAND latch is both inputs being equal to 0 at the same time, an input combination that should be avoided.

The operation of the basic SR latch can be modified by providing an additional input signal that determines (controls) when the state of the latch can be changed by determining whether S and R (or S' and R') can affect the circuit.

An SR latch with a control input consists of the basic SR latch and two additional NAND gates.

The control input E_n acts as an enable signal for the other two inputs. The outputs of the NAND gates stay at the logic-1 level as long as the enable signal remains at 0. This is the quiescent condition for the SR latch.

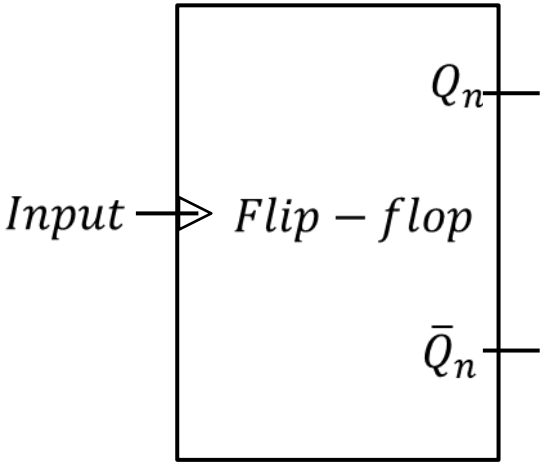


(a) Logic diagram

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

Flip-flop

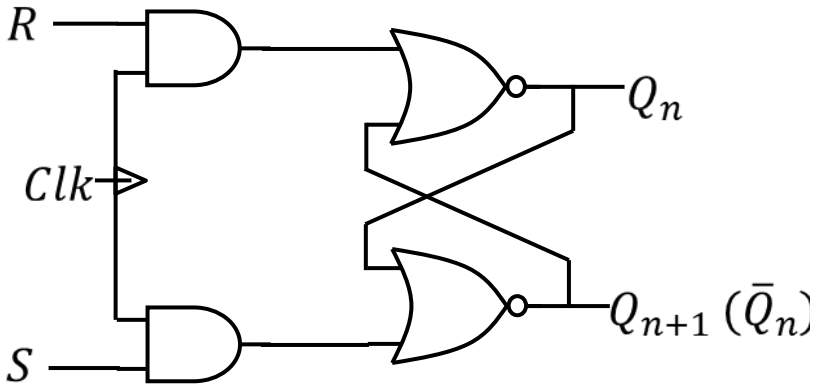


- Capable of storing ***ONE*** bit
- Bi-stable device (because of only two ***stable*** states)
- Functionality type: 4
  1. **RS flip-flop**
  2. **JK flip-flop**

  Two Inputs
  3. **T flip-flop**
  4. **D flip-flop**

  One Input
- Operation modes: 2
  1. **Latch Mode** (No changes in *o/p* when *i/p* (control pins) changes)
  2. **Transparent Mode** (Transparent to control pins *i.e. o/p* changes when *i/p* changed)
- Clock Synchronization: ±ve Level, ±ve Edge
- Construction: NOR & NAND gates

SR Flip-flop

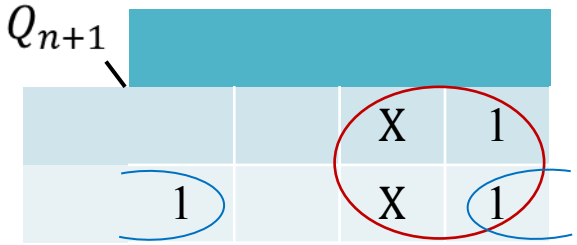


Function Table

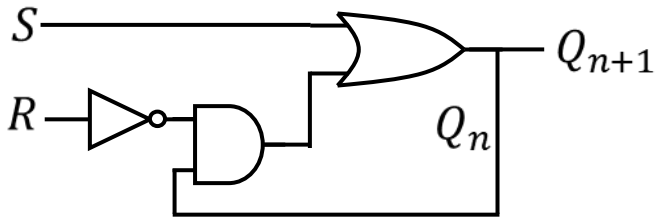
0	0	
0	1	0
1	0	1
1	1	X

Characteristics Table (Truth Table)

0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X



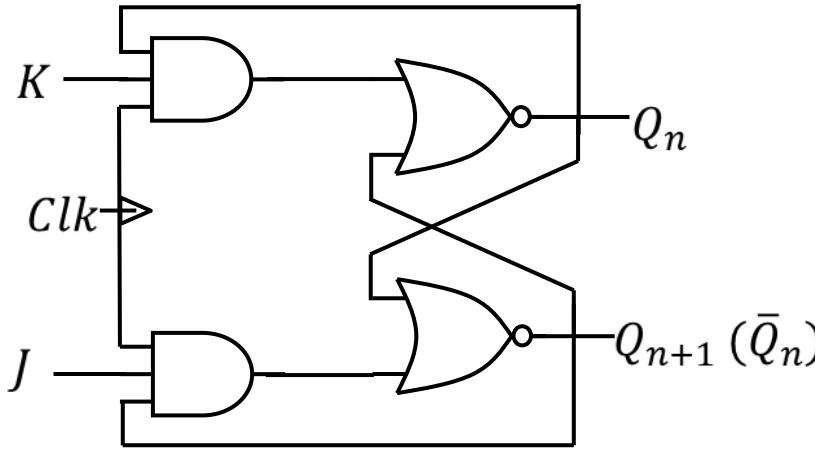
$Charatersics\ Eq^n$: $Q_{n+1} = S + \bar{R}Q_n$

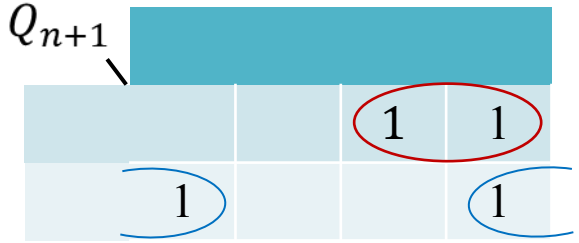


Excitation Table (Inverse of TT)

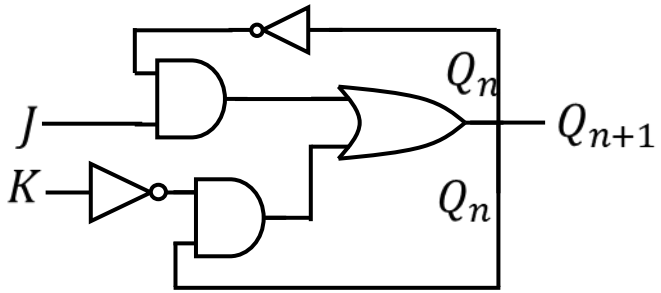
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	1

JK Flip-flop





$Charatersics\ Eq^n$: $Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$



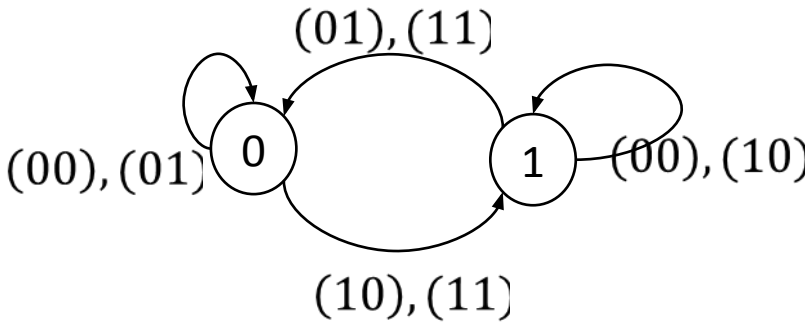
Function Table

0	0	
0	1	0
1	0	1
1	1	

Characteristics Table (Truth Table)

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

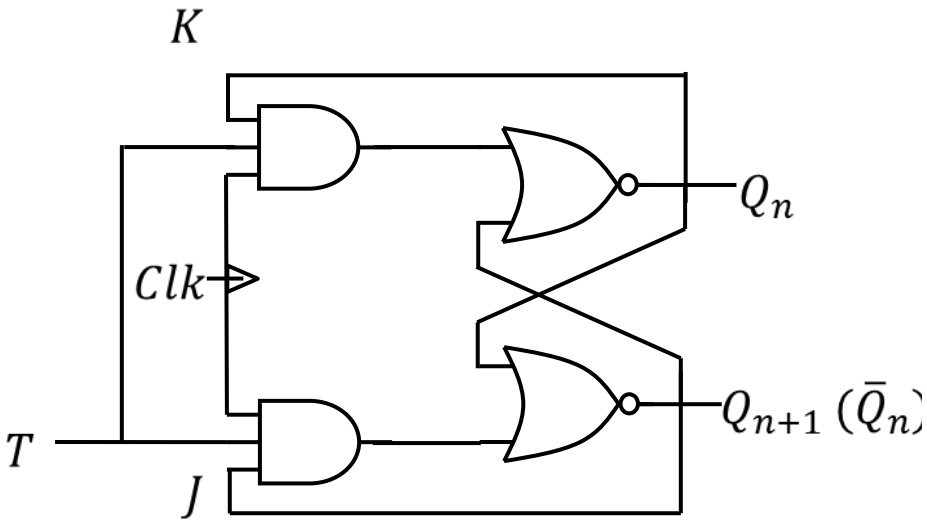
State Diagram



Excitation Table (Inverse of TT)

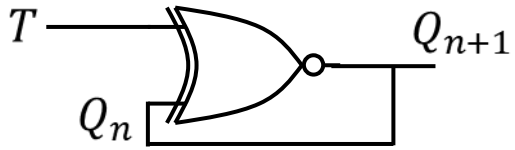
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

T Flip-flop



Q_{n+1}		
		1
	1	

$Charatersics\ Eq^n$: $Q_{n+1} = T\bar{Q}_n + \bar{T}Q_n$



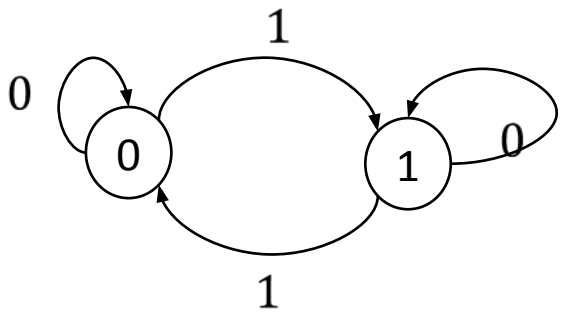
Function Table

0	
1	

Characteristics Table (Truth Table)

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

State Diagram



Excitation Table (Inverse of TT)

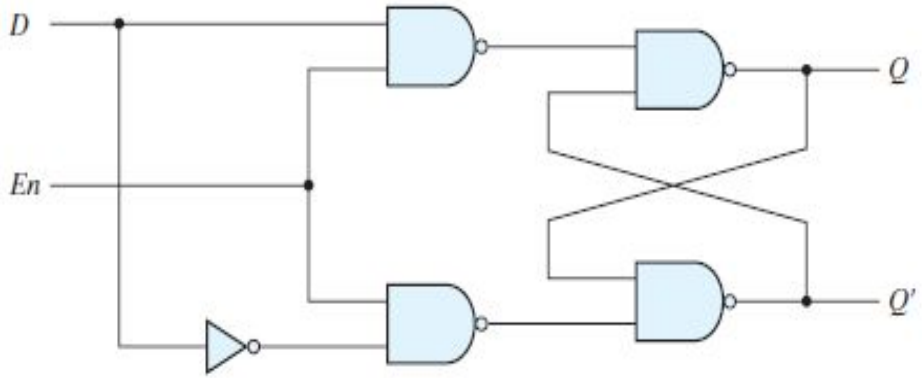
0	0	0
0	1	1
1	0	1
1	1	0

D Latch (Transparent Latch)

One way to eliminate the undesirable condition of the indeterminate state in the SR latch is to ensure that inputs S and R are never equal to 1 at the same time. This is done in the **D latch**.

This latch has only two inputs: D (data) and En (enable). The D input goes directly to the S input, and its complement is applied to the R input.

As long as the enable input is at 0, the cross-coupled SR latch has both inputs at the 1 level and the circuit cannot change state regardless of the value of D .

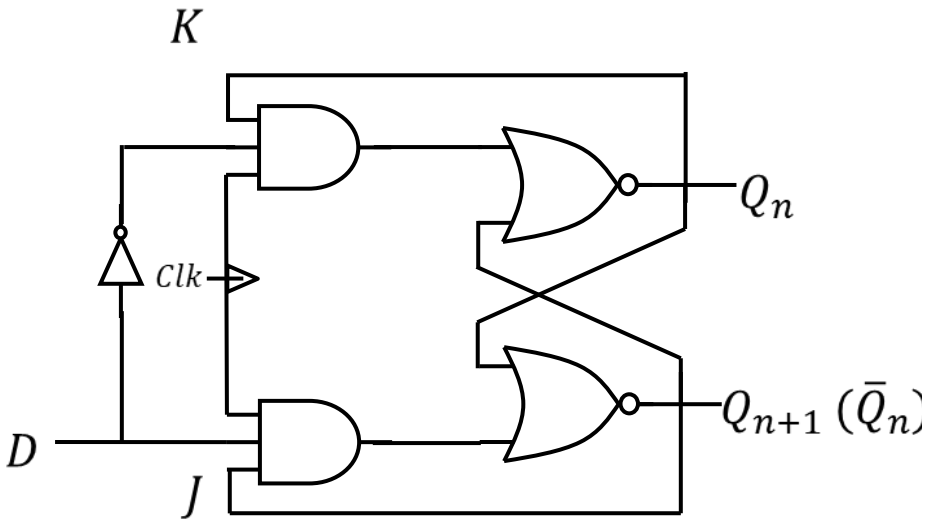


(a) Logic diagram

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

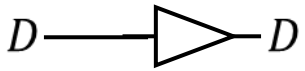
(b) Function table

D Flip-flop



Q_{n+1}	
	1
	1

$Charatersics\ Eq^n$: $Q_{n+1} = \mathrm{D}$



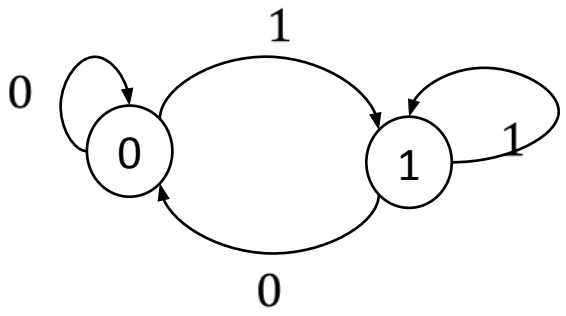
Function Table

0	
1	1

Characteristics Table (Truth Table)

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

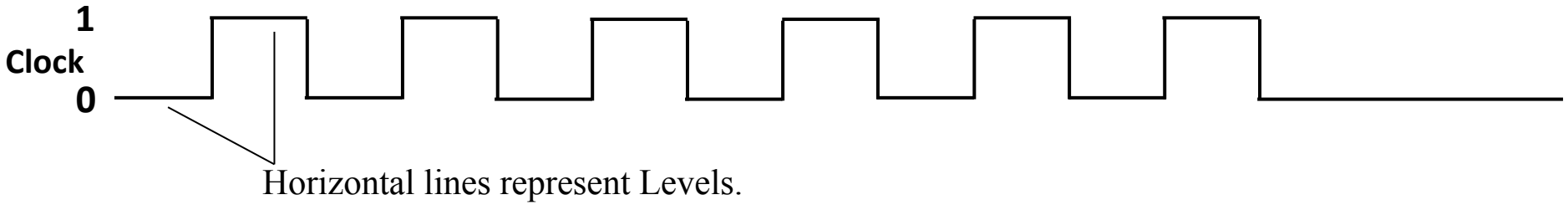
State Diagram



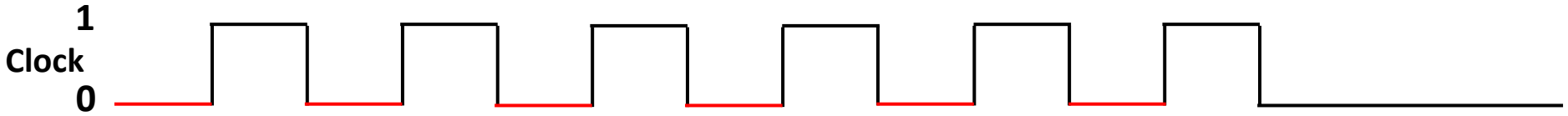
Excitation Table (Inverse of TT)

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

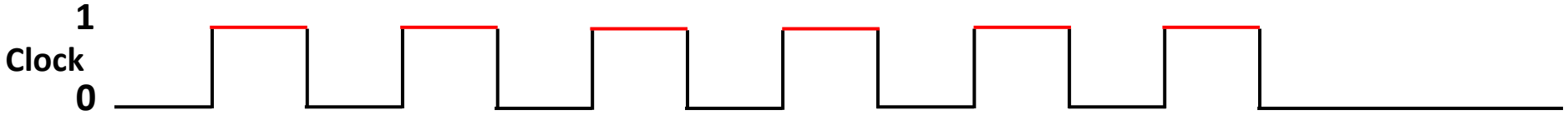
Clock Triggering



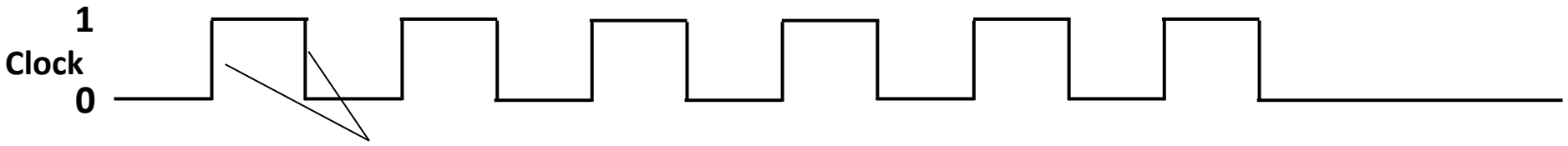
-ve Level Clock Triggering: When clock level is ***Zero***



+ve Level Clock Triggering: When clock level is ***One***

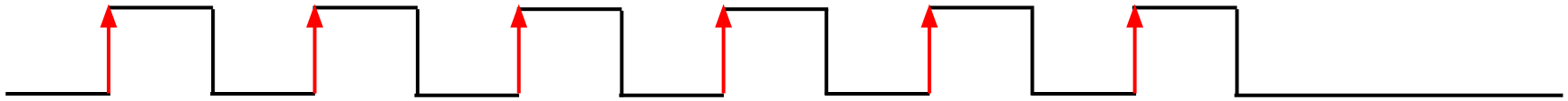


Clock Triggering



Vertical lines represent Edges.

+ve Edge Clock Triggering: When clock level is changes from ***Zero*** to ***One***



-ve Edge Clock Triggering: When clock level is changes from ***One*** to ***Zero***