Synchronous Sequential Circuits

Text book:

Morris Mano M. and Michael D. Ciletti., Digital Design: With an introduction to the Verilog HDL (5e), PHI Learning 2007.
Chapter 5 and 6

- The digital circuits considered thus far have been combinational, i.e., the outputs at any instant of time are entirely dependent upon the inputs present at that time.

- Although every digital system is likely to have combinational circuits, most systems encountered in practice also include memory elements, which require that the system be described in terms of sequential.

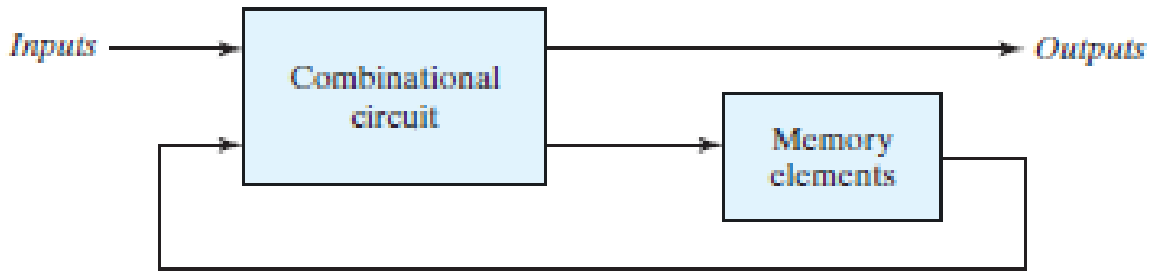


FIGURE 5.1
Block diagram of sequential circuit

- In the diagram, the memory elements (Storage Elements) are devices capable of storing binary information within them.
- The binary information stored in the memory elements at any given time defines the state of the sequential circuit.
- The sequential circuit receives binary information from external inputs. These inputs, together with the present state of the memory elements, determine the binary value at the output terminals.
- They also determine the condition for changing the state in the memory elements.
- The block diagram demonstrates that the external outputs in a sequential circuit are a function not only of external inputs, but also of the present state of the memory elements. The next state of the memory elements is also a function of external inputs and the present state.

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

Two main types of sequential circuits. :

- Synchronous sequential circuit
- Asynchronous sequential circuit

A synchronous sequential circuit is a system Whose behavior can be defined from the knowledge of its signals at discrete instants of time. Synchronization is achieved by a timing device called a clock generator, which provides a clock signal having the form of a periodic train of clock pulses. The activity within the circuit and the resulting updating of stored values is synchronized to the occurrence of clock pulses.

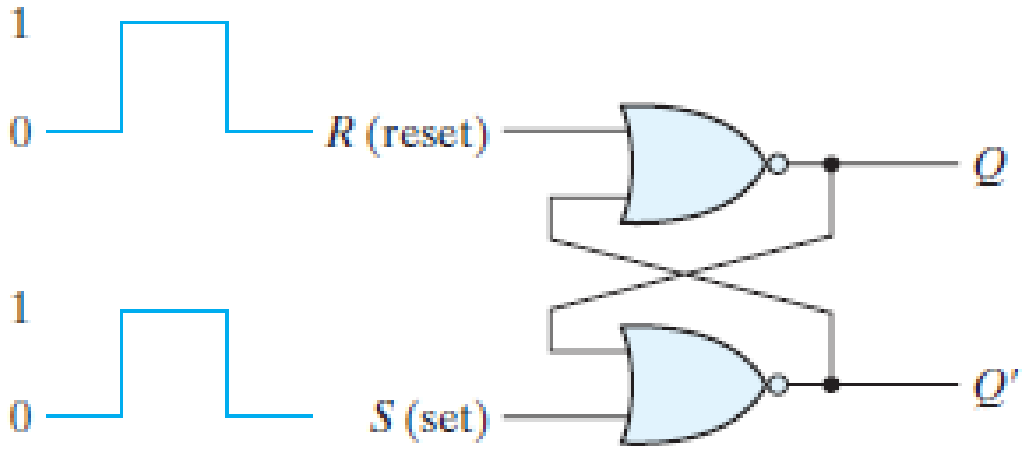
The behavior of an asynchronous sequential circuit depends upon the order in which its input signals change and can be affected at any instant of time.

- A synchronous sequential circuit employs signals that affect the storage elements only at discrete instants of time. Synchronization is achieved by a timing device called a ***clock generator*** which produces a periodic train of ***clock pulses***.
- The pulses are distributed throughout the system in such a way that synchronous storage elements are affected only in some specified relationship to every pulse.
- In practice, the clock pulses are applied with other signals that specify the required change in the storage elements. The outputs of storage elements can change their value only in the presence of clock pulses.
- Synchronous sequential circuits that use clock pulses as inputs for storage elements are called ***clocked sequential circuits*****.**

Storage Elements

- 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.
- *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* .
- 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.
- Because they are the building blocks of flip-flops, however, we will consider the fundamental storage mechanism used in latches

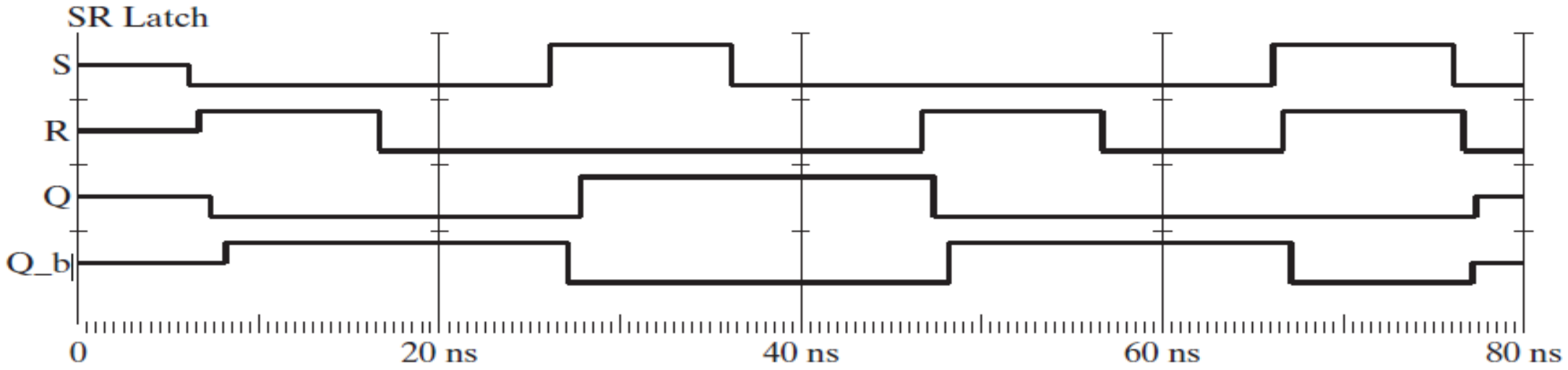
SR Latch with NOR gates



(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)

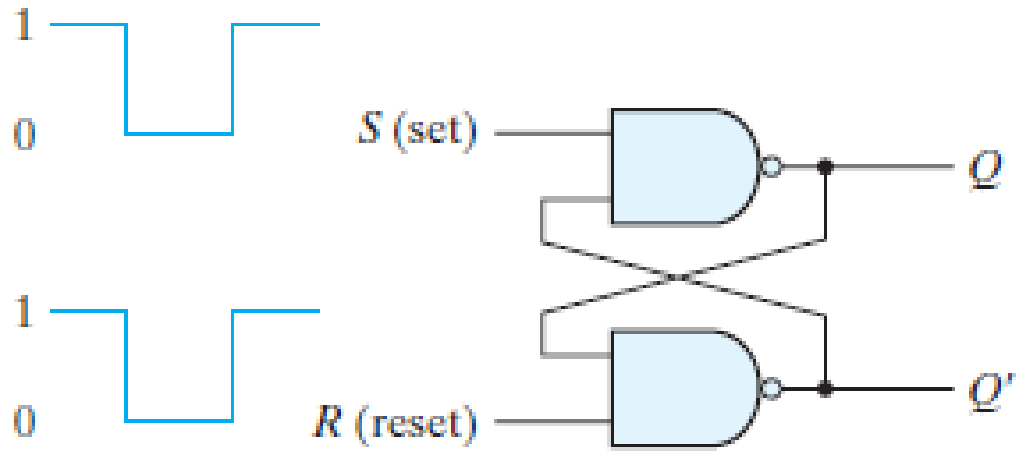
(b) Function table



- It has two useful states. When output Q = 1 and Q' = 0, it 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. The binary state of the FF is taken to be the value of the normal output.
- 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. Consequently, in practical applications, setting both inputs to 1 is forbidden.

- Under normal conditions, both inputs remain at 0 unless the state has to be changed. The application of a momentary 1 to the *S* input causes the FF to go to the set state. The *S* input must go back to 0 before any other changes take place, in order to avoid the occurrence of an undefined next state that results from the forbidden input condition.
- When both inputs S and R are equal to 0, the FF can be in either the set or the reset state, depending on which input was most recently a 1.
- If a 1 is applied to both the S and R inputs, both outputs go to 0. This action produces an undefined next state, because the state that results from the input transitions depends on the order in which they return to 0. It also violates the requirement that outputs be the complement of each other. In normal operation, this condition is avoided by making sure that 1's are not applied to both inputs simultaneously.

SR latch with NAND gates ($\overline{S}\,\overline{R}$ Latch)



(a) Logic diagram

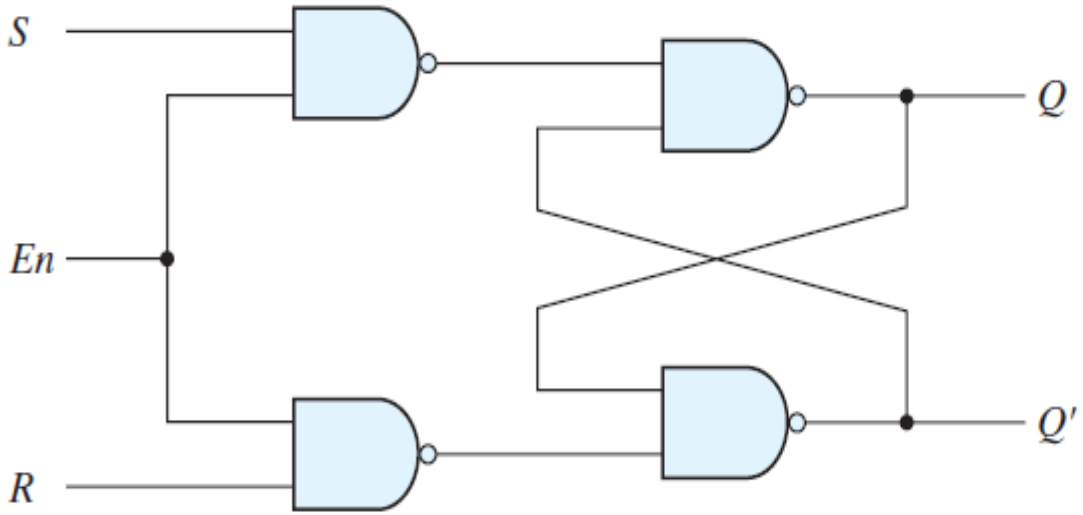
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

(b) Function table

- The output of a NAND gate is 1 if any input is 0, and that the output is 0 only when all the inputs are 1.

- It operates with both inputs normally at 1, unless the state has to be changed. The application of 0 to the *S* input causes output *Q* to go to 1, putting the FF in the set state.
- When the *S* input goes back to 1, the circuit remains in the set state. After both inputs go back to 1, we are allowed to change the state of the FF by placing a 0 in the *R* input. This action causes the circuit to go to the reset state and stay there even after both inputs return to 1.
- The condition that is forbidden for the NAND FF is both inputs being equal to 0 at the same time, an input combination that should be avoided.

SR latch with Control input



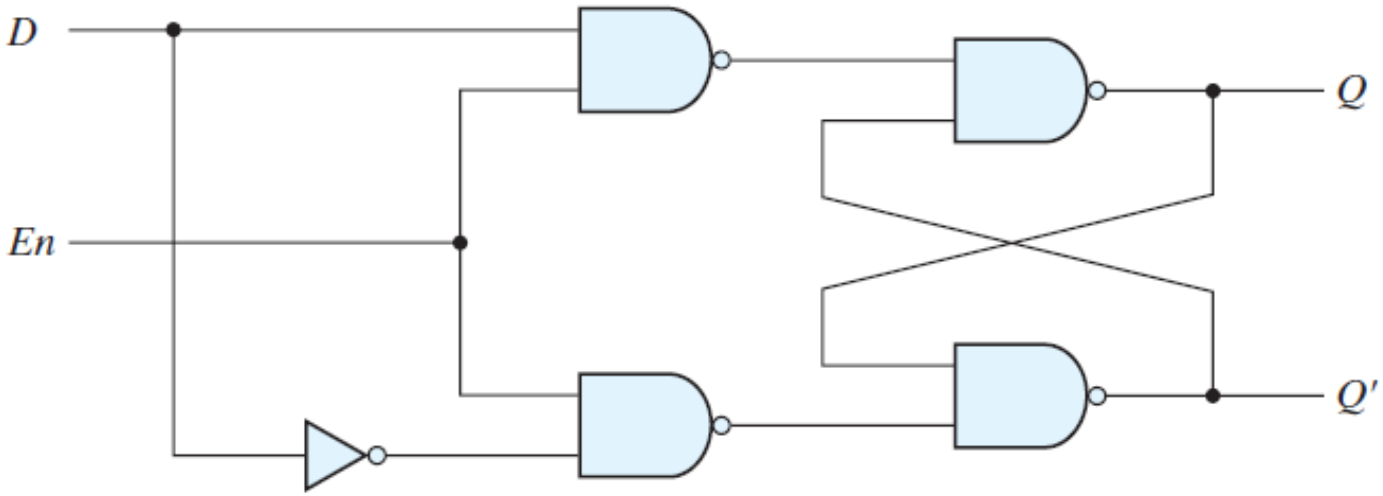
(a) Logic diagram

En	*S*	*R*	Next state of *Q*
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

(b) Function table

D Latch (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



(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

Storage Element : Flip Flop

- The state of a latch or flip-flop is switched by a change in the control input. This momentary change is called a *trigger*, and the transition it causes is said to trigger the flip-flop.
- The *D* latch with pulses in its control input is essentially a flip-flop that is triggered every time the pulse goes to the logic-1 level.
- As long as the pulse input remains at this level, any changes in the data input will change the output and the state of the latch.

- When latches are used for the storage elements, a serious difficulty arises. The state transitions of the latches start as soon as the clock pulse changes to the logic-1 level. The new state of a latch appears at the output while the pulse is still active.
- This output is connected to the inputs of the latches through the combinational circuit.
- If the inputs applied to the latches change while the clock pulse is still at the logic-1 level, the latches will respond to new values and a new output state may occur.
- The result is an unpredictable situation, since the state of the latches may keep changing for as long as the clock pulse stays at the active level.
- Because of this unreliable operation, the output of a latch cannot be applied directly or through combinational logic to the input of the same or another latch when all the latches are triggered by a common clock source.
- **Flip-flop circuits are constructed in such a way as to make them operate properly when they are part of a sequential circuit that employs a common clock**

Flip Flop

A flip-flop 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 flip-flops are in the number of inputs they possess and in the manner in which the inputs affect the binary state. The most common types of flip-flops are discussed in what follows.

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.

- The key to the proper operation of a flip-flop is to trigger it only during a signal *transition* .
- A clock pulse goes through two transitions: from 0 to 1 and the return from 1 to 0.

(a) Response to positive level

Pulse Triggered

(b) Positive-edge response

Positive edge Triggered

(c) Negative-edge response

Negative edge Triggered

Edge Triggered D Flip Flop

A common way to create a flip-flop is to connect two latches which is often referred to as a *master–slave* flip-flop.

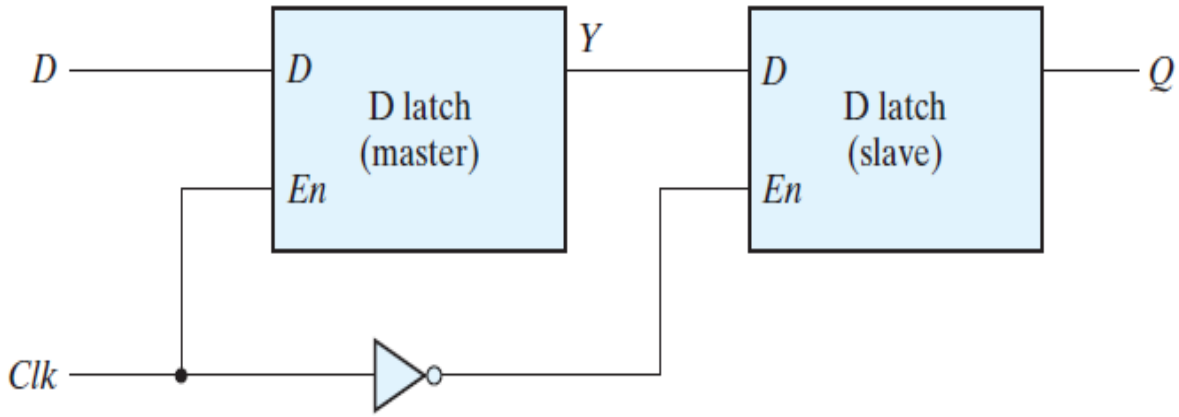


FIGURE 5.9
Master–slave *D* flip-flop

- The circuit samples the *D* input and changes its output *Q* only at the negative edge of the synchronizing or controlling clock (designated as *Clk*).
- When the clock is 0, the output of the inverter is 1. The slave latch is enabled, and its output *Q* is equal to the master output *Y* .
- The master latch is disabled because *Clk* = 0.
- When the input pulse changes to the logic-1 level, the data from the external *D* input are transferred to the master.
- The slave, however, is disabled as long as the clock remains at the 1 level, because its *enable* input is equal to 0.
- Any change in the input changes the master output at *Y*, but cannot affect the slave output.
- When the clock pulse returns to 0, the master is disabled and is isolated from the *D* input.
- At the same time, the slave is enabled and the value of *Y* is transferred to the output of the flip-flop at *Q* .
- Thus, *a change in the output of the flip-flop can be triggered only by and during the transition of the clock from 1* to 0.

- The behavior of the master–slave flip-flop just described dictates that

  (1) the output may change only once,

  (2) a change in the output is triggered by the negative edge of the clock, an

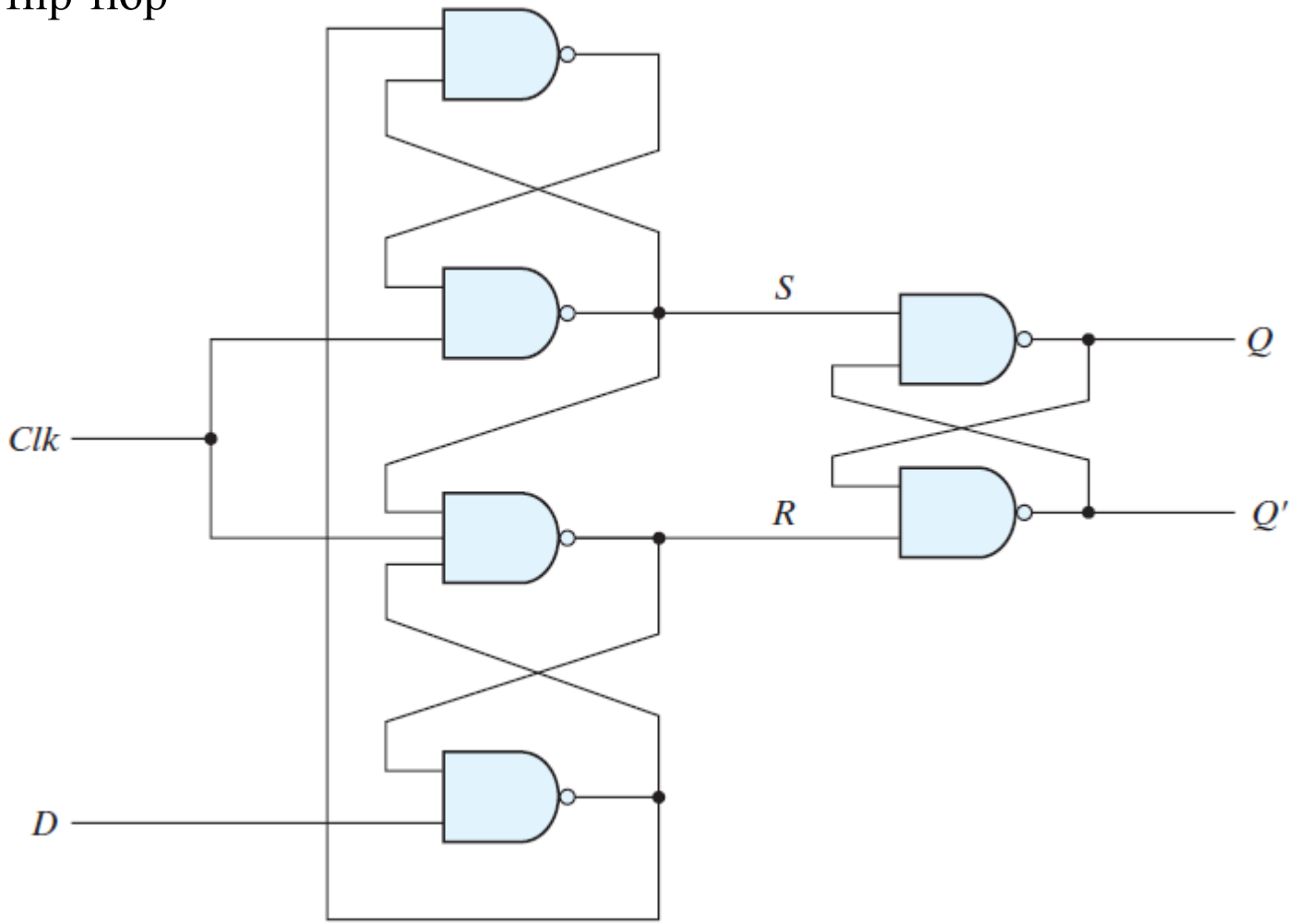
  (3) the change may occur only during the clock's negative level.

The value that is produced at the output of the flip-flop is the value that was *stored in the master stage immediately before the negative edge occurred* .

Edge Triggered D FF

Another construction of an edge-triggered *D* flip-flop uses three *SR* latches

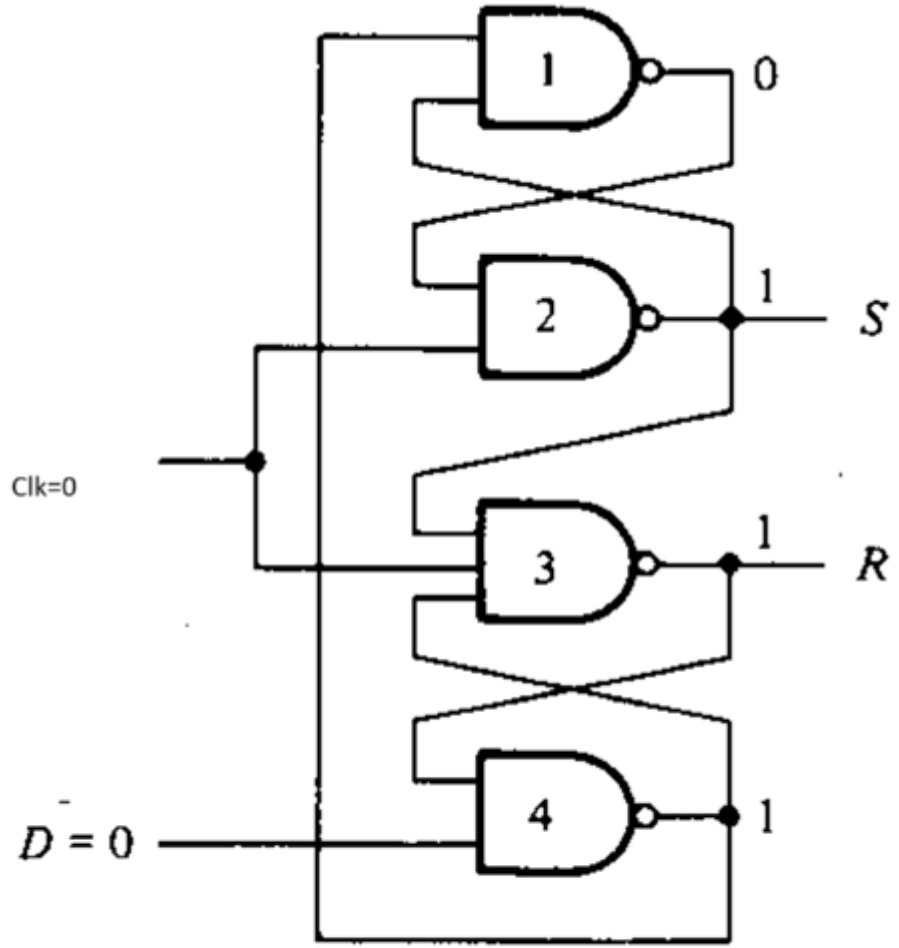
- Two latches respond to the external *D* (data) and *Clk* (clock) inputs.
- The third latch provides the outputs for the flip-flop.

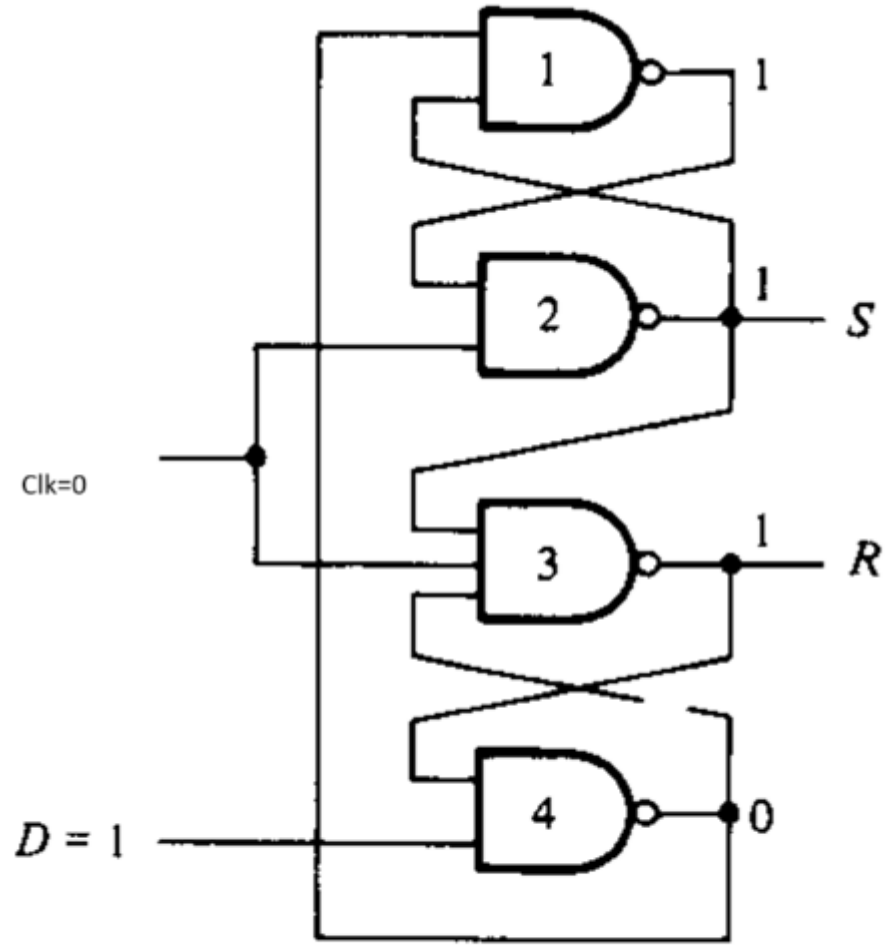


S	R	Q	Q′	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

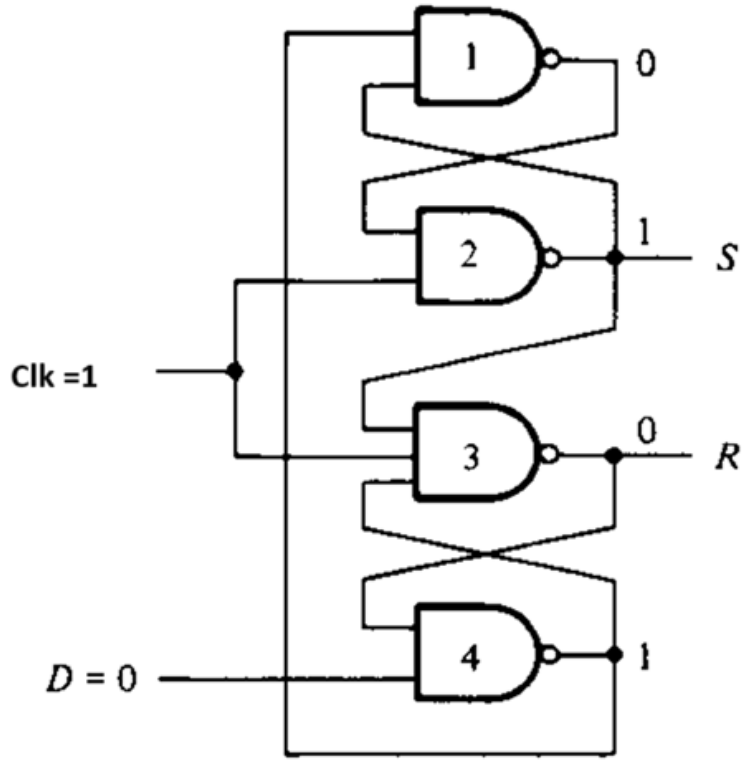
(b) Function table

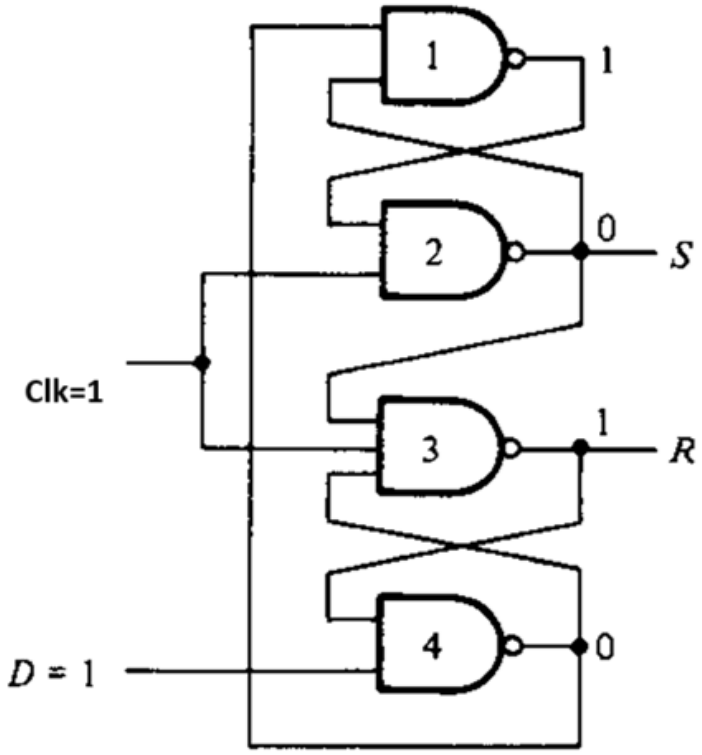
- The *S* and *R* inputs of the output latch are maintained at the logic-1 level when *Clk* = 0.
- This causes the output to remain in its present state. Input *D* may be equal to 0 or 1.
- If *D* = 0 when *Clk* becomes 1, *R* changes to 0. This causes the flip-flop to go to the reset state, making *Q* = 0.
- If there is a change in the *D* input while *Clk* = 1, terminal *R* remains at 0 because *Q* is 0.
- Thus, the flip-flop is locked out and is unresponsive to further changes in the input.
- When the clock returns to 0, *R* goes to 1, placing the output latch in the quiescent condition without changing the output.
- Similarly, if *D* = 1 when *Clk* goes from 0 to 1, *S* changes to 0.
- This causes the circuit to go to the set state, making *Q* = 1. Any change in *D* while *Clk* = 1 does not affect the output.





(a) With Clk=0





(b) With Clk=1

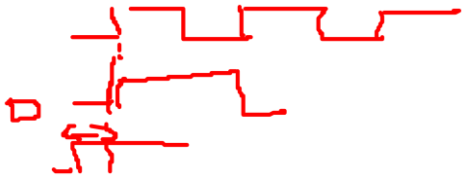
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

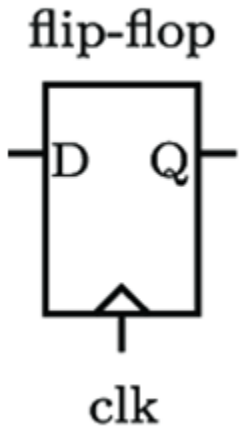
(b) Function table

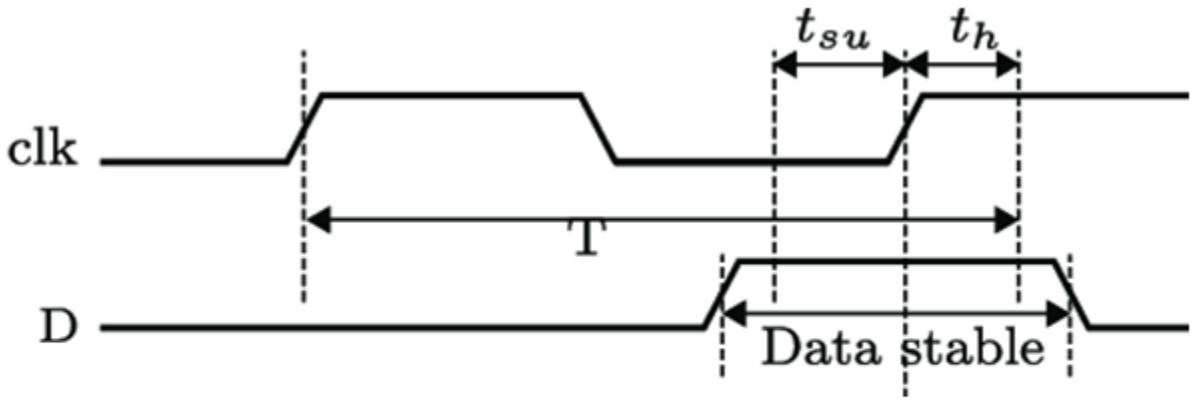
- When the input clock in the positive-edge-triggered flip-flop makes a positive transition, the value of *D* is transferred to *Q* . A negative transition of the clock (i.e., from 1 to 0) does not affect the output, nor is the output affected by changes in *D* when *Clock* is in the steady logic-1 level or the logic-0 level. Hence, this type of flip-flop responds to the transition from 0 to 1 and nothing else.
- There is a minimum time called the *setup time* during which the *D* input must be maintained at a constant value prior to the occurrence of the clock transition. It is equal to the propagation delay through the gates 4 and 1 since a change in D causes a change in the outputs of these two gates
- Similarly, there is a minimum time called the *hold time* during which the *D* input must not change after the application of the positive transition of the clock. When D=0, it is the propagation delay of gate 3, since it must be ensured that R becomes 0 in order to maintain the output of gate 4 at 1, regardless the value of D. When D=1, it is the propagation delay of gate 2, since it must be ensured that S becomes 0 in order to maintain the output of gate 1 at 1, regardless the value of D
- The propagation delay time of the flip-flop is defined as the interval between the trigger edge and the stabilization of the output to a new state

SET UP Time

- **Setup time** is the minimum amount of time a synchronous data input should be held steady before the clock event so that the data input is reliably sampled by the clock event.
- Setup time is defined as the minimum amount of time before the clock's active edge that the data must be stable for it to be latched correctly. Any violation may cause incorrect data to be captured, which is known as setup violation.



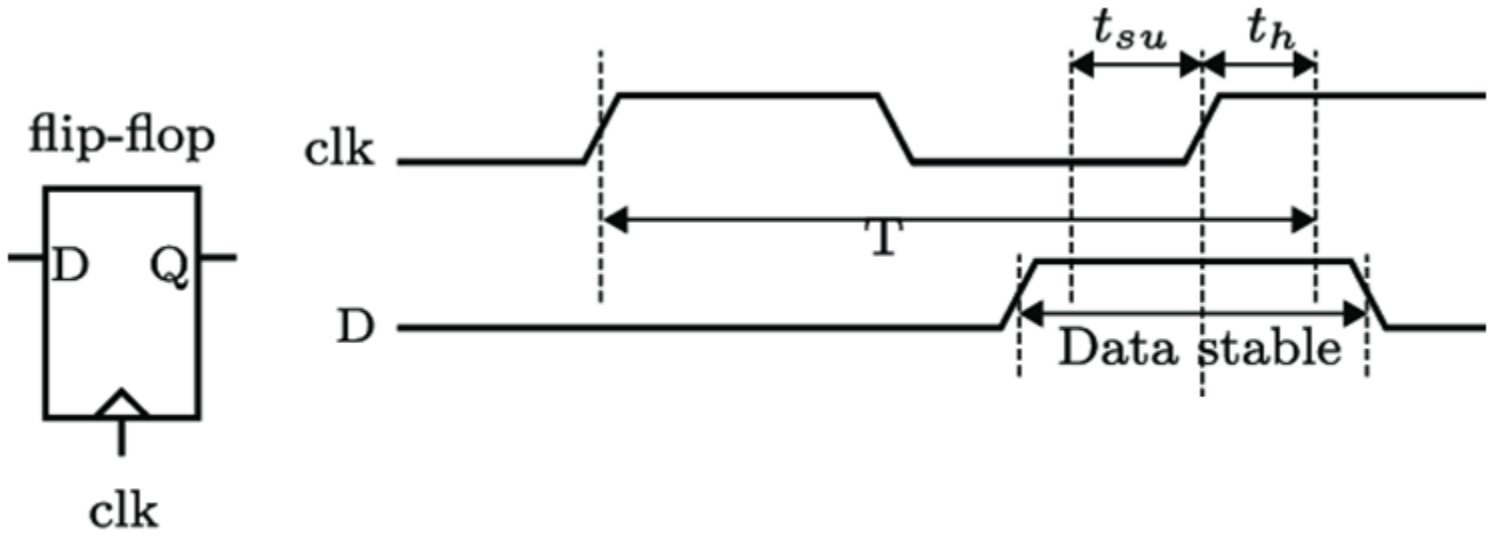




HOLD Time

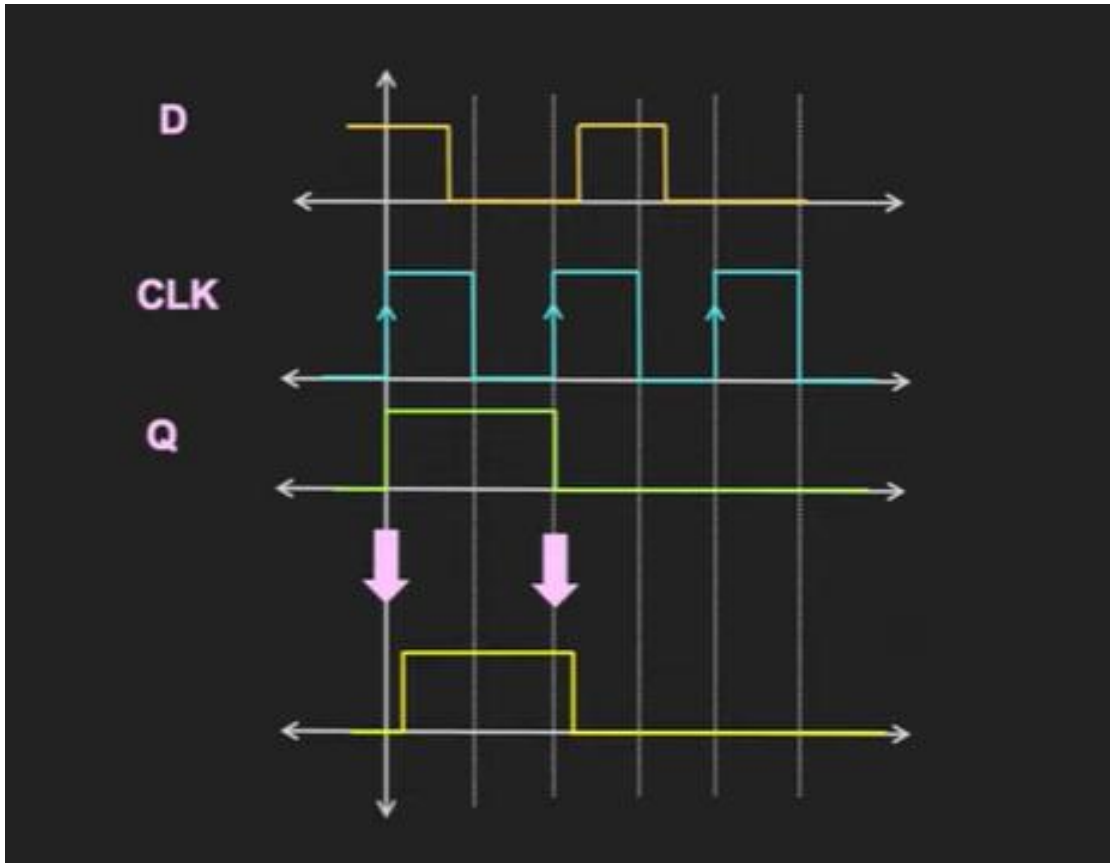
Hold time is the minimum amount of time a synchronous data input should be held steady after the clock event so that the data input is reliably sampled by the clock event.

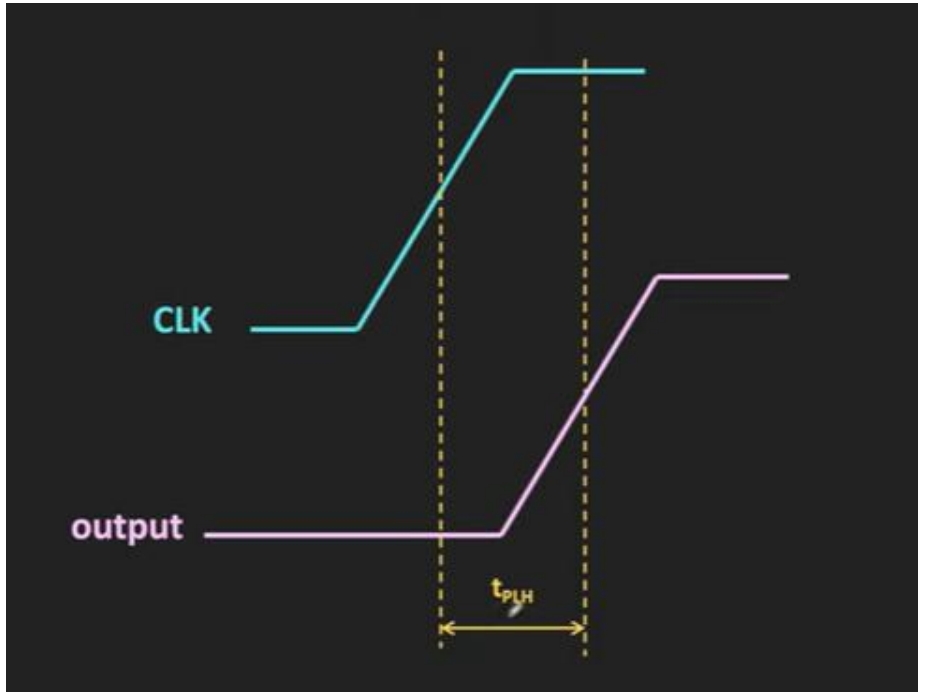
Hold time is defined as the minimum amount of time after the clock's active edge during which data must be stable. Violation in this case may cause incorrect data to be latched, which is known as a hold violation. Note that setup and hold time is measured with respect to the active clock edge only.

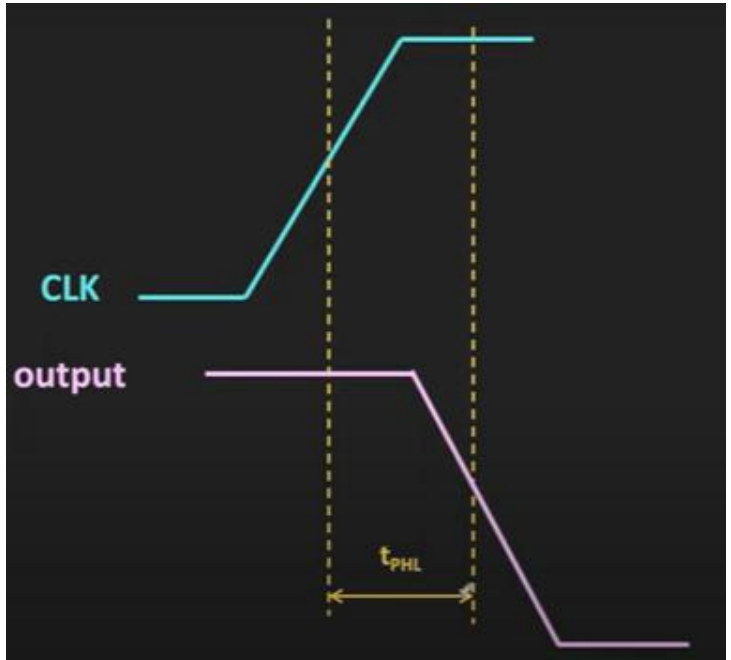


Propagation Delay

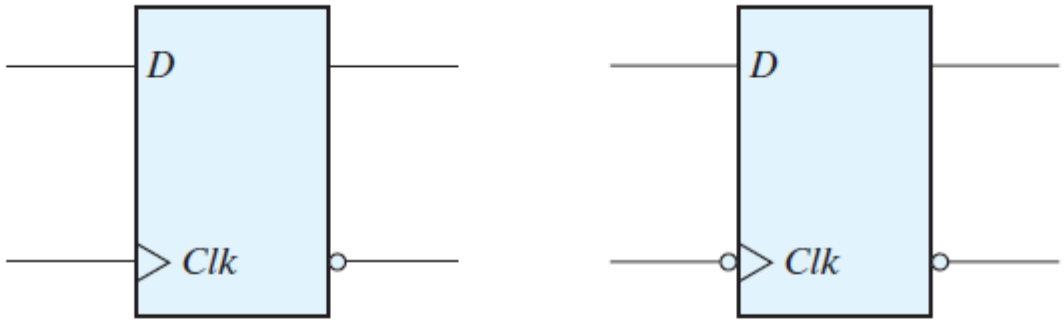
The propagation delay of the flip flops means a small delay occurs between the clock edge and the flip flop output, Q







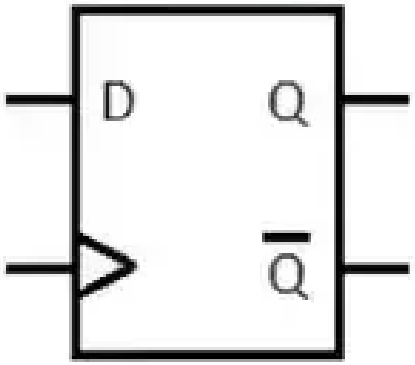
- In sum, when the input clock in the positive-edge-triggered flip-flop makes a positive transition, the value of *D* is transferred to *Q* .
- A negative transition of the clock (i.e., from 1 to 0) does not affect the output, nor is the output affected by changes in *D* when *Clk* is in the steady logic-1 level or the logic-0 level.
- Hence, this type of flip-flop responds to the transition from 0 to 1 and nothing else.



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

FIGURE 5.11
Graphic symbol for edge-triggered *D* flip-flop

D Flip Flop

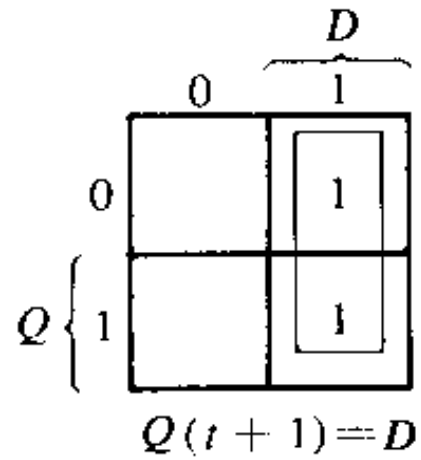


Graphical Symbol

CLK	D	Q^+
↑	0	0
↑	1	1

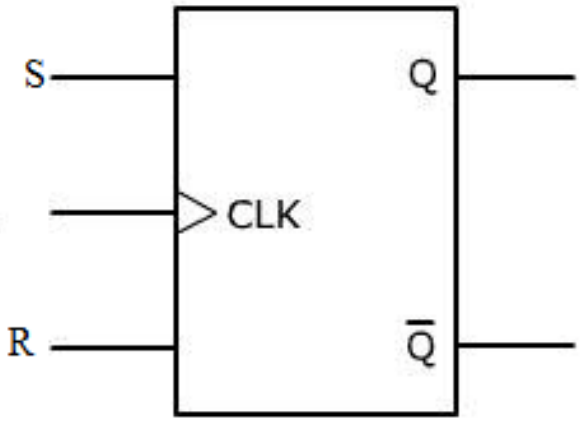
Q	D	$Q(t+1)$
0	0	0
0	1	1
1	0	0
1	1	1

(b) Characteristic table



(c) Characteristic equation

S R Flip Flop

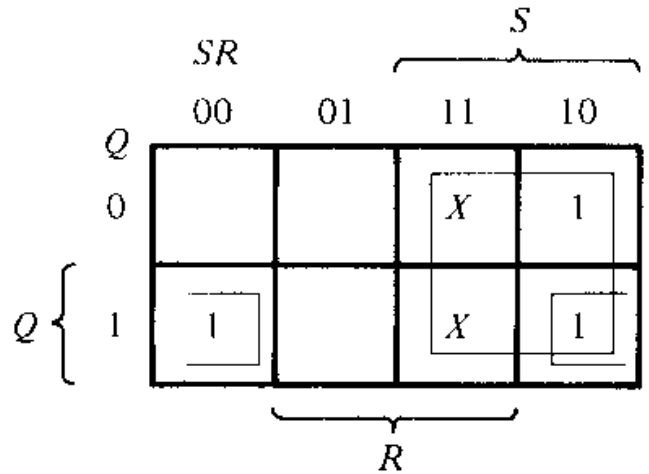


Graphical Symbol

CLK	S	R	Q^+
↑	0	0	Q
↑	0	1	0
↑	1	0	1
↑	1	1	Forbidden

Q	S	R	Qt+1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	Indeterminate
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	Indeterminate

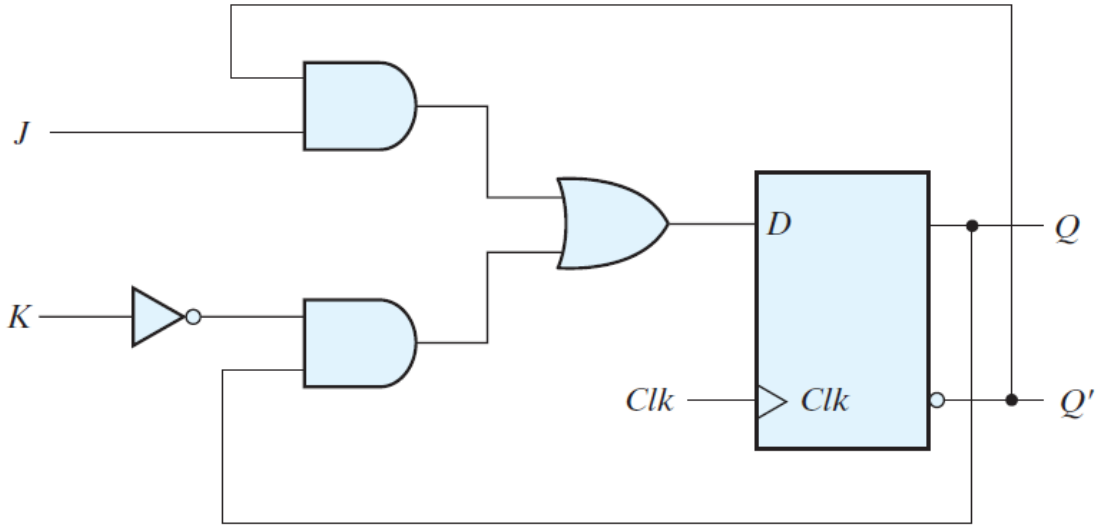
b) Characteristic Table



$$Q(t+1) = S + R'Q$$
$$SR = 0$$

(c) Characteristic equation

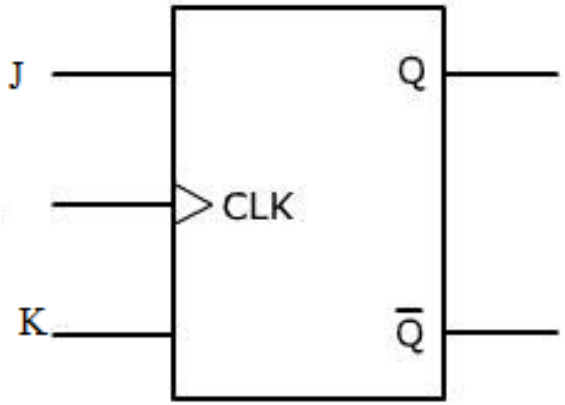
J K Flip Flop



(a) Circuit diagram

CLK	J	K	Q^+
↑	0	0	Q
↑	0	1	0
↑	1	0	1
↑	1	1	$\overline{Q}$

JK Flip flop

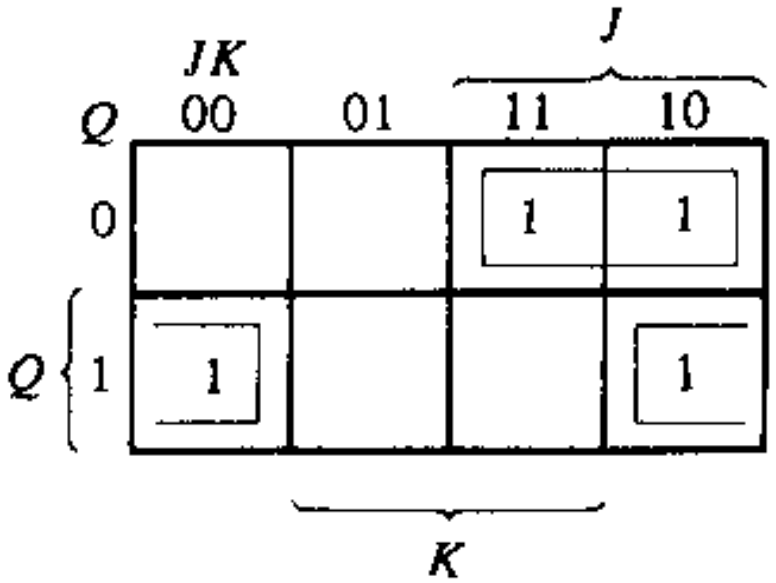


Q	J	K	$Q(t+1)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

(b) Characteristic table

Q	J	K	$Q(t+1)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

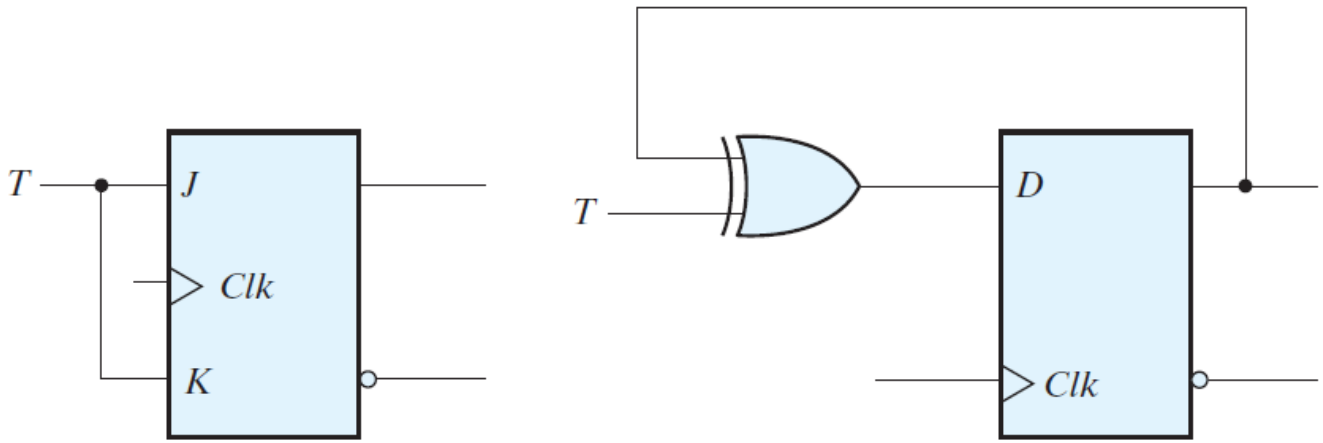
(b) Characteristic table



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

(c) Characteristic equation

T Flip Flop



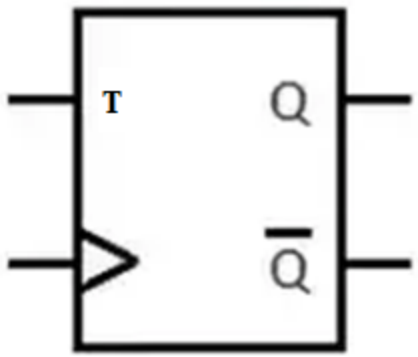
(a) From *JK* flip-flop

(b) From *D* flip-flop

CLK	T	Q^+
↑	0	Q
↑	1	$\overline{Q}$

T FF Logic Diagram

T Flip flop



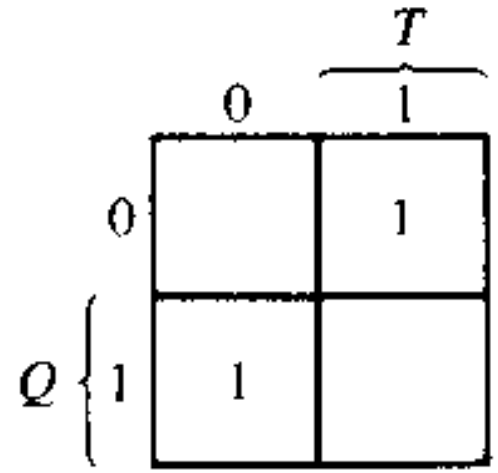
Graphical Symbol

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

(b) Characteristic table

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

(b) Characteristic table



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

(c) Characteristic equation

Excitation Table

J	K	Q	Q^+
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

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Excitation Table

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0