CO
Unit-3

Part-1

Synchronous Sequential Logic

Syllabus

- Introduction to Sequential circuits
- latches
- Flip-Flops

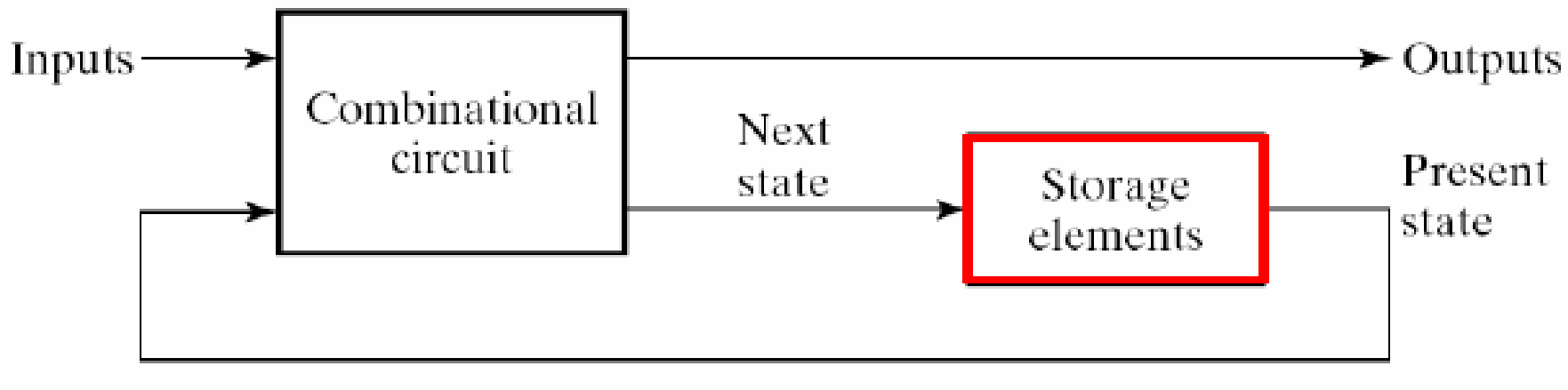
Syllabus

- Introduction to Sequential circuits
- latches
- Flip-Flops

Introduction

- The digital circuits considered thus far have been combinational i.e, their output depends only and immediately on their inputs—they have no memory, i.e., dependence on past values of their inputs.
- Sequential circuits act as storage elements and have memory.
- They can store, retain, and then retrieve information when needed at a later time.

Sequential Circuits



Sequential Circuits

- 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.

Sequential Circuits

- The block diagram demonstrates that the outputs in a sequential circuit are a function not only of the inputs, but also of the present state of the storage elements.
- The next state of the storage elements is also a function of external inputs and the present state.
- **A sequential circuit is specified by a time sequence of inputs, outputs, and internal states.**
- The outputs of combinational logic depend only on the present values of the inputs.

Types of Sequential Circuits

- There are two main types of sequential circuits, and their classification is a function of the timing of their signals.
- A synchronous sequential circuit is a system whose behavior can be defined from the knowledge of its signals at discrete instants of time.
- Storage elements: e.g., flip-flops

Types of Sequential Circuits

- The behavior of an 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.
- Storage elements: e.g., latches, feedback paths
- Adv.: May have better performance.
- Disadv.: May be unstable & difficult to design

Synchronous 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 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.

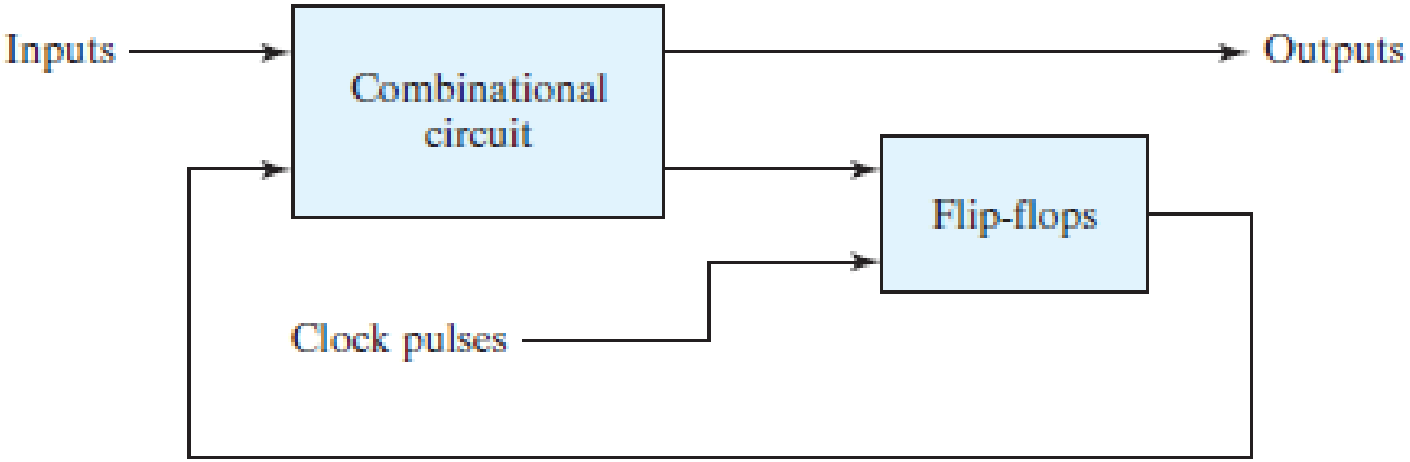
Synchronous Sequential Circuits

- Synchronous sequential circuits that use clock pulses to control storage elements are called clocked sequential circuits.
- 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.
- Advantage:-The design of synchronous circuits is feasible because they seldom manifest instability problems and their timing is easily broken down into independent discrete steps, each of which can be considered separately.

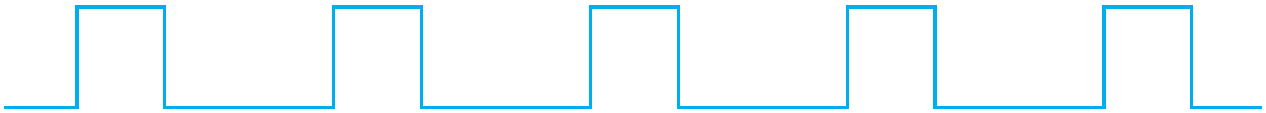
Memory Elements

- 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.
- The block diagram of a synchronous clocked sequential circuit is shown in below Fig.

Memory Elements



(a) Block diagram



(b) Timing diagram of clock pulses

Fig: Synchronous clocked Sequential Circuits

Memory Elements

- The outputs are formed by a combinational logic function of the inputs to the circuit or the values stored in the flip-flops (or both).
- The value that is stored in a flip-flop when the clock pulse occurs is also determined by the inputs to the circuit or the values presently stored in the flip-flop (or both).
- The new value is stored (i.e., the flip-flop is updated) when a pulse of the clock signal occurs.
- Prior to the occurrence of the clock pulse, the combinational logic forming the next value of the flip-flop must have reached a stable value.

Memory Elements

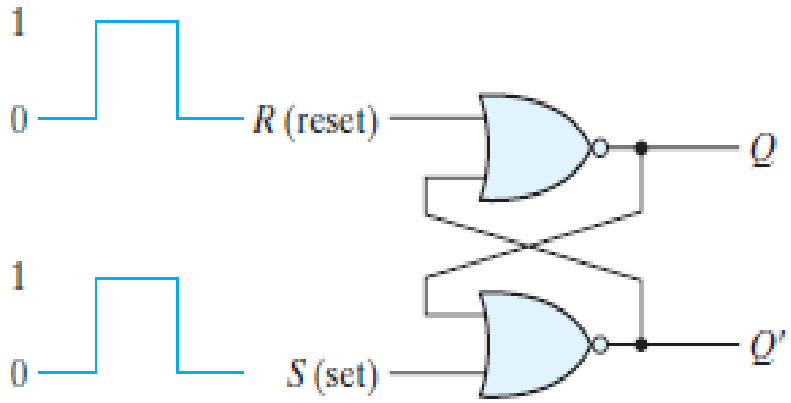
- A change in state of the flip-flops is initiated only by a clock pulse transition—for example, when the value of the clock signals changes from 0 to 1.
- The transition from one state to the next occurs only at predetermined intervals dictated by the clock pulses.

STORAGE ELEMENTS: LATCHES

- Storage elements that operate with signal levels 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.
- 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.
  - Avoid to use latches as possible in synchronous sequential circuits to avoid design problems

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 SR latch constructed with two cross-coupled NOR gates is shown in below Fig.



(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

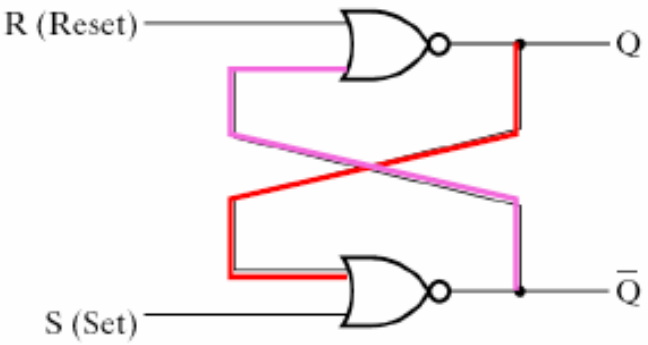
FIGURE 5.3
SR latch with NOR gates

SR Latch

- SR latch: w/ NOR gates

Q: the normal output
$\overline{Q}$: the complement output

Async Seq Ckt



(a) Logic diagram

S	R	Q	$\overline{Q}$	
1	0	1	0	Set state
0	0	1	0	
0	1	0	1	Reset state
0	0	0	1	
1	1	0	0	Undefined

(b) Function table

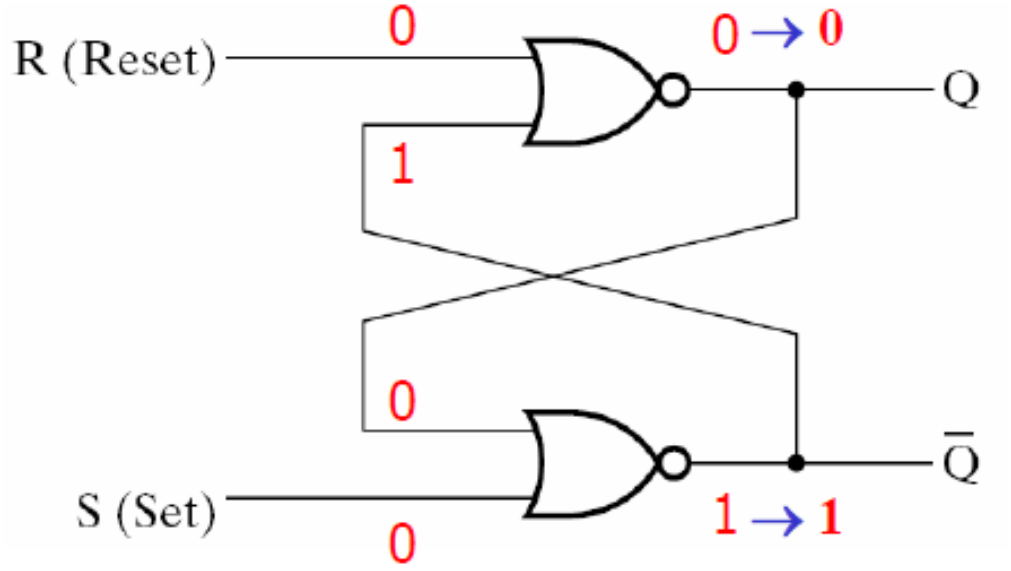
- Useful states:
  - Set state: $Q = 1, \overline{Q} = 0$
  - Reset state: $Q = 0, \overline{Q} = 1$
- Undefined states: $Q = \overline{Q}$

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

- Two useful states:
  - S=1, R=0 → set state (Q will become to 1)
  - S=0, R=1 → reset state (Q will become to 0)
- When S=0 and R=0 → keep the current value

(Case 1)

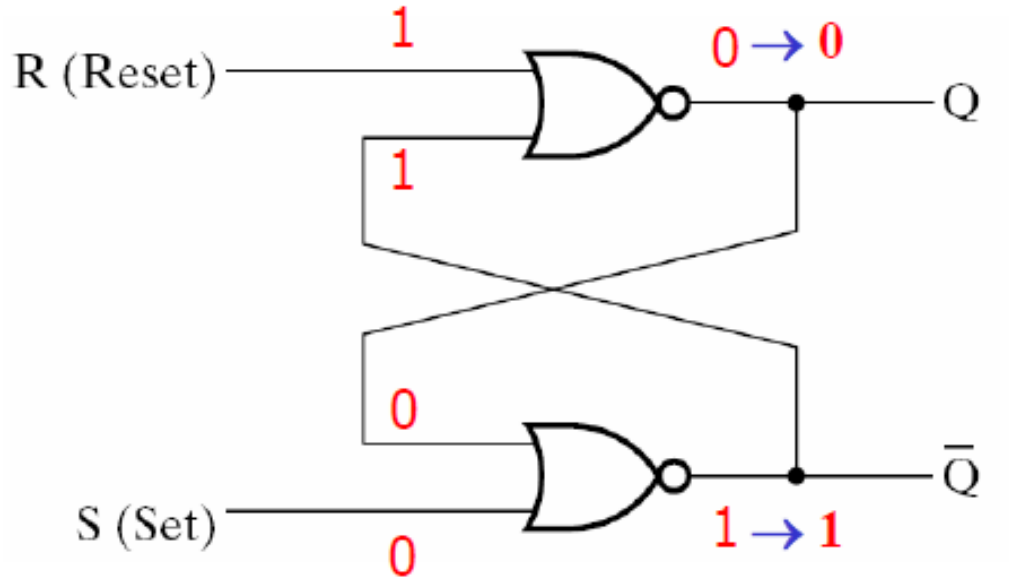


S	R	Q	$\overline{Q}$	Q^+	$\overline{Q}^+$	
0	0	0	1	0	1	✓
		1	0	1	0	✓
0	1	0	1	0	1	
		1	0	0	1	
1	0	0	1	1	0	
		1	0	1	0	
1	1	0	1	0	0	
		1	0	0	0	

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

(Case 2)

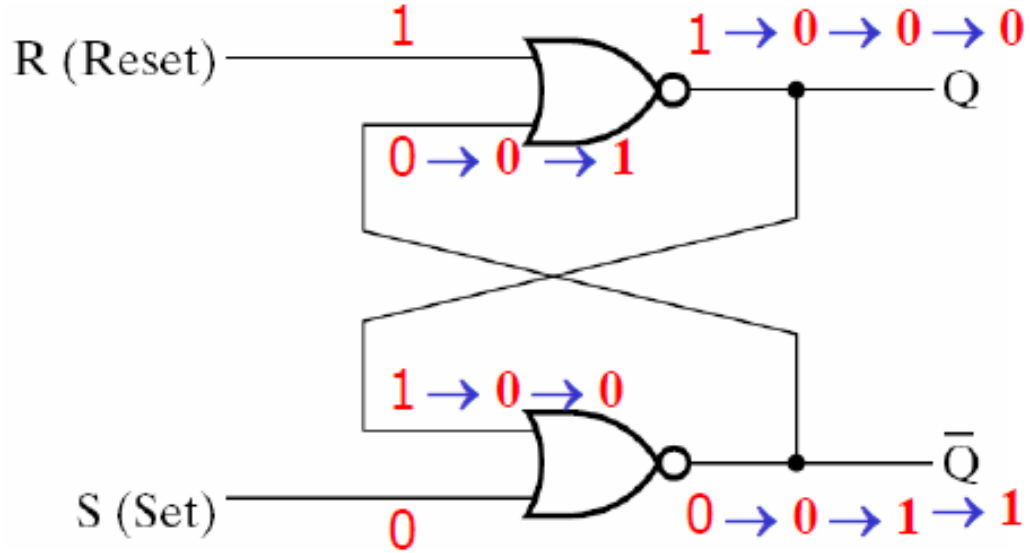


S	R	Q	$\overline{Q}$	Q^+	$\overline{Q}^+$	
0	0	0	1	0	1	
		1	0	1	0	
0	**1**	**0**	**1**	**0**	**1**	✓
		1	0	0	1	
1	0	0	1	1	0	
		1	0	1	0	✓
1	1	0	1	0	0	
		1	0	0	0	

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

(Case 3)

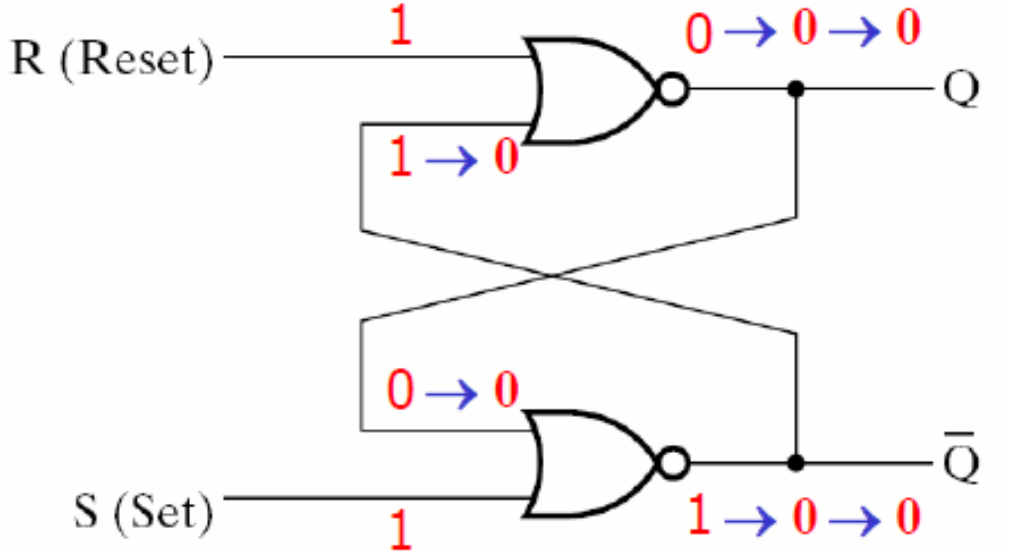


S R	Q $\overline{Q}$	Q^+ $\overline{Q}^+$	
0 0	0 1	0 1	
	1 0	1 0	
0 1	0 1	0 1	
	1 0	**0 1**	✓
1 0	0 1	1 0	✓
	1 0	1 0	
1 1	0 1	0 0	
	1 0	0 0	

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

(Case 4)

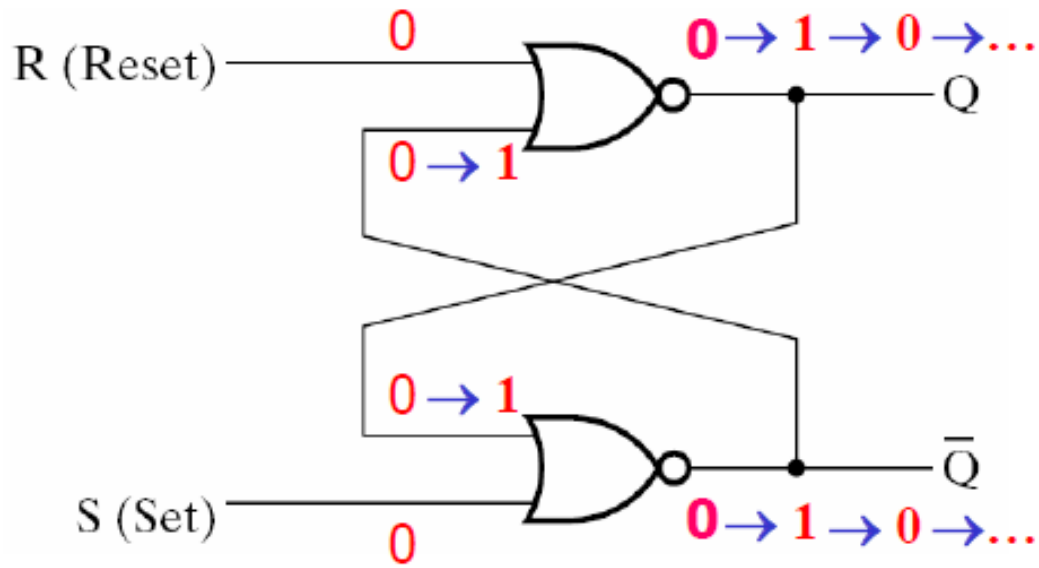


S R	Q $\overline{Q}$	Q^+ $\overline{Q}^+$
0 0	0 1 1 0	0 1 1 0
0 1	0 1 1 0	0 1 0 1
1 0	0 1 1 0	1 0 1 0
1 1	**0 1** 1 0	**0 0** 0 0

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

(Case 4+)

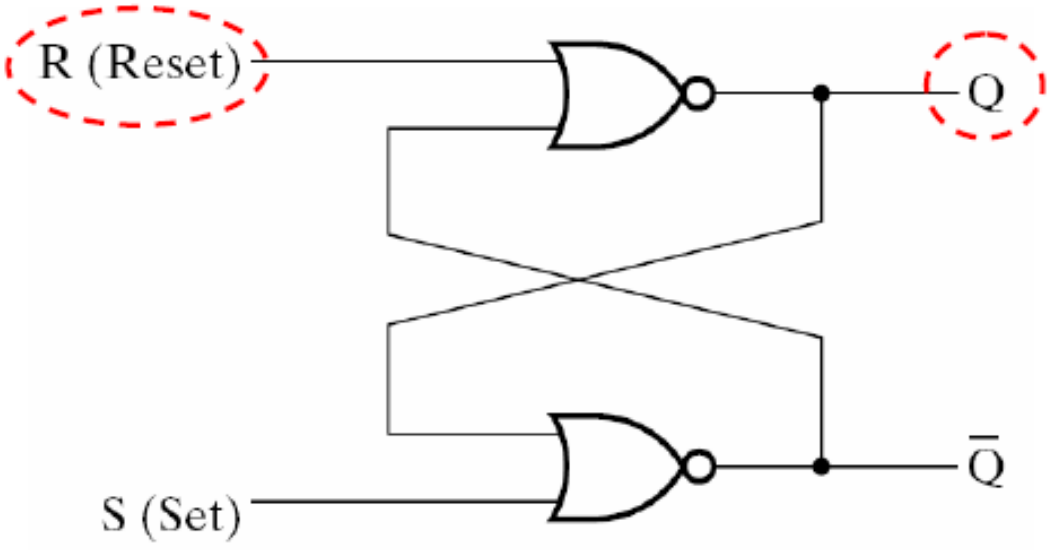


S R	Q $\bar{Q}$	Q^+ $\bar{Q}^+$
0 0	0 1 1 0	0 1 1 0
0 1	0 1 1 0	0 1 0 1
1 0	0 1 1 0	1 0 1 0
1 1	0 1 1 0	0 0 0 0

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

- Summary:



S R	Q $\overline{Q}$	Q^+ $\overline{Q}^+$
0 0	0 1 1 0	0 1 1 0
0 1	0 1 1 0	0 1 0 1
1 0	0 1 1 0	1 0 1 0
1 1	0 1 1 0	0 0 0 0

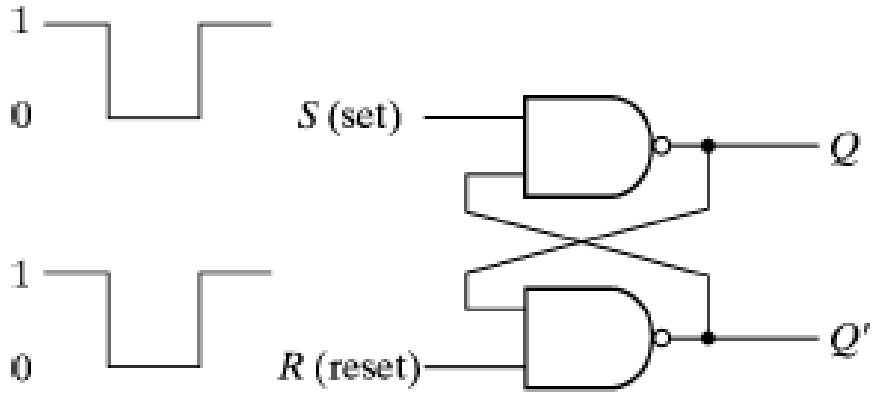
$$Q^+ = (R + \overline{Q})'$$
$$\overline{Q}^+ = (S + Q)'$$

S	R	Q^+
0	0	No change ($Q^+ = Q$)
0	1	Reset ($Q^+ = 0$)
1	0	Set ($Q^+ = 1$)
1	1	Indeterminate

Q^+: next state of Q

SR Latch with NAND Gates

- The SR latches constructed with two cross-coupled NAND gates are **active-low**
  - S=1, R=0 → reset state (Q will become to 0)
  - S=0, R=1 → set state (Q will become to 1)
  - S=1, R=1 → unchanged



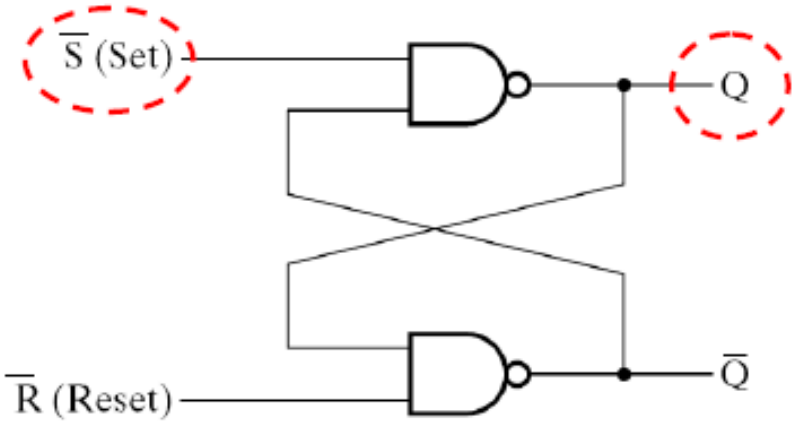
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	

(a) Logic diagram (b) Function table

Fig. 5-4 *SR* Latch with NAND Gates

$\overline{S}$ $\overline{R}$ Latch

- $\overline{S}$ $\overline{R}$ latch: w/ NAND gates

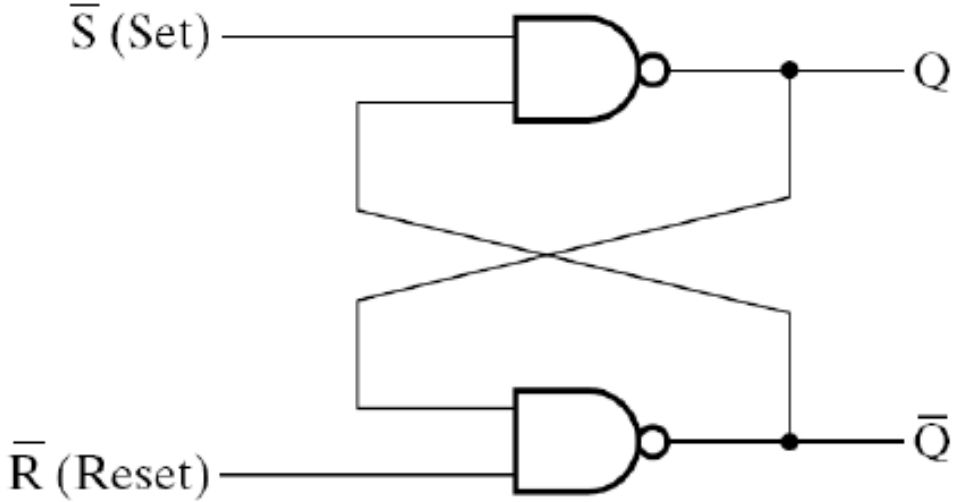


(a) Logic diagram

$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$	
0	1	1	0	Set state
1	1	1	0	
1	0	0	1	Reset state
1	1	0	1	
0	0	1	1	Undefined

(b) Function table

$\overline{S}$	$\overline{R}$	Q^+
1	1	No change ($Q^+ = Q$)
1	0	Reset ($Q^+ = 0$)
0	1	Set ($Q^+ = 1$)
0	0	Indeterminate



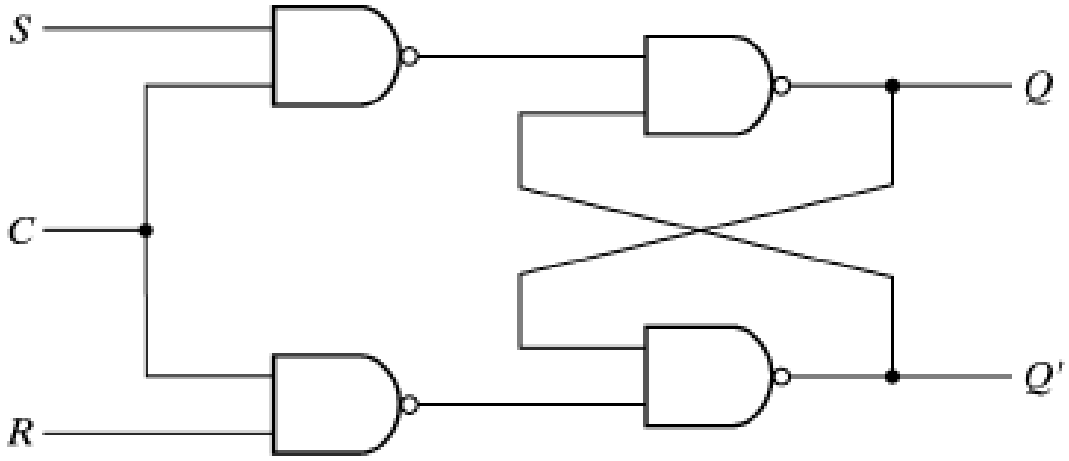
$\overline{S}$ $\overline{R}$	Q $\overline{Q}$	Q^+ $\overline{Q}^+$
1 1	0 1 1 0	0 1 1 0
0 1	0 1 1 0	0 1 0 1
1 0	0 1 1 0	1 0 1 0
0 0	0 1 1 0	1 1 1 1

$$Q^+ = (\overline{S} \cdot \overline{Q})'$$
$$\overline{Q}^+ = (\overline{R} \cdot Q)'$$

S	R	Q^+
1	1	No change ($Q^+ = Q$)
1	0	Reset ($Q^+ = 0$)
0	1	Set ($Q^+ = 1$)
0	0	Indeterminate

SR Latch with Control Input

- Add an additional control input to determine when the state of the latch can be changed
- C=0: S and R are disabled (no change at outputs)
- C=1: S and R are active-high



(a) Logic diagram

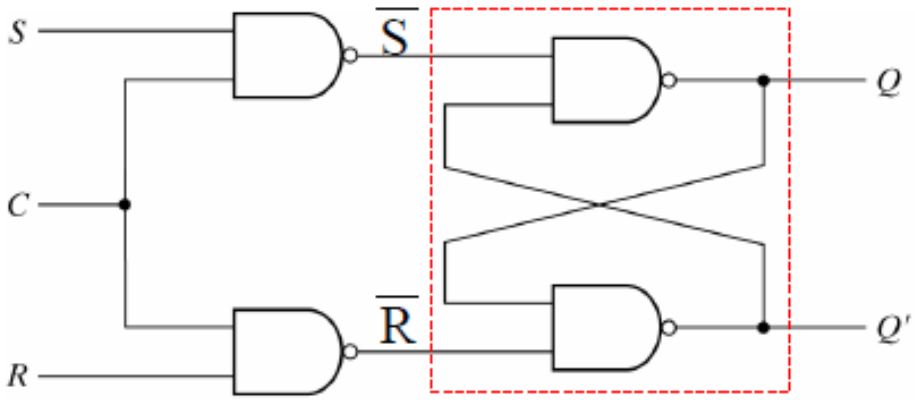
C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; Reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

(b) Function table

Fig. 5-5 SR Latch with Control Input

SR Latch w/ Control Input

- SR latch w/ control input:



(a) Logic diagram

enable signal

$\overline{S} = \overline{R} = 1$

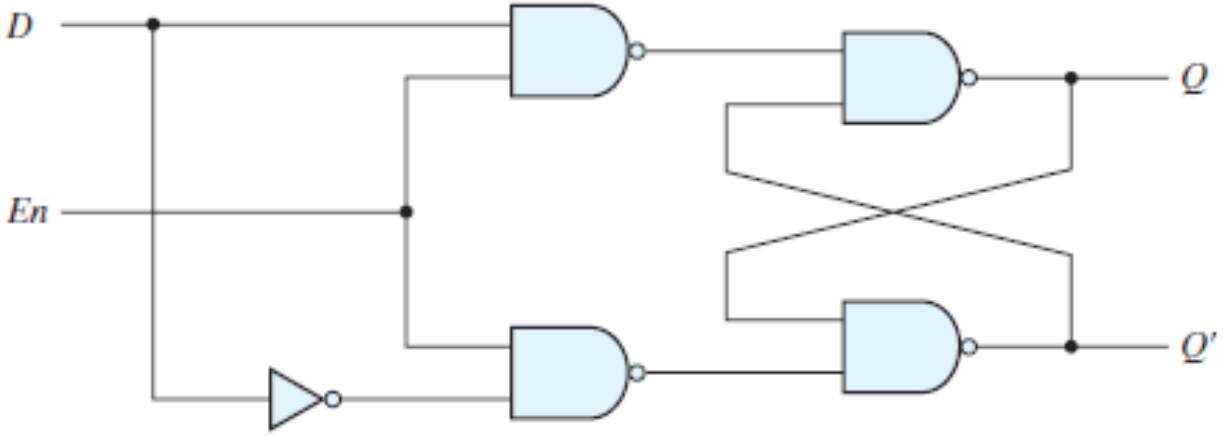
C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0; Reset state
1	1	0	Q = 1; set state
1	1	1	Indeterminate

(b) Function table

- The indeterminate condition makes this ckt difficult to manage & it is seldom used in practice.
- It is an important ckt (other latches & flip-flops are constructed from it)

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, shown in below Fig.



(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

FIGURE 5.6
D latch

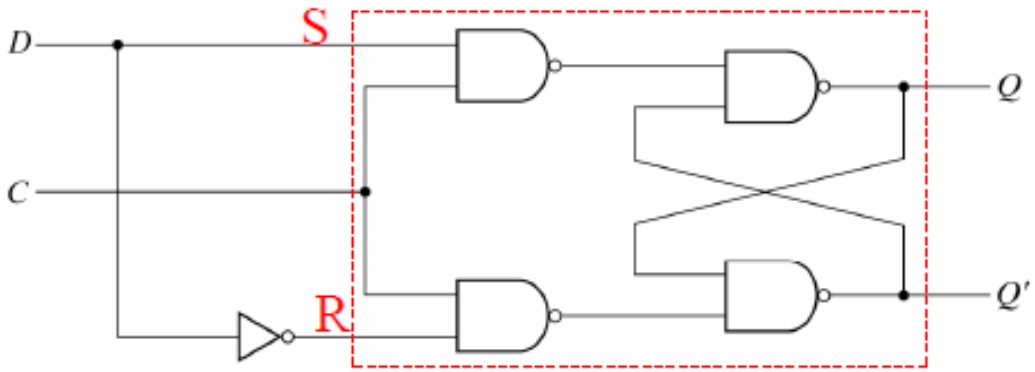
D Latch (Transparent Latch)

- 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.
- The D input is sampled when En = 1.
- If D = 1, the Q output goes to 1, placing the circuit in the set state.
- If D = 0, output Q goes to 0, placing the circuit in the reset state.

D Latch

C D	C S R	Q+
0 X	0 X X	Q (No change)
1 0	1 0 1	0 (Reset)
1 1	1 1 0	1 (Set)

- D latch (w/ control input):



(a) Logic diagram

C D	Next state of Q
0 X	No change
1 0	Q = 0; Reset state
1 1	Q = 1; Set state

(b) Function table

- Disadv. of a latch w/ control input:
  - The state of latch may keep changing for as long as the control input stays in the active level.

Graphic Symbols for latches

A latch is designated by a rectangular block with inputs on the left and outputs on the right. One output designates the normal output, and the other designates the complement output.

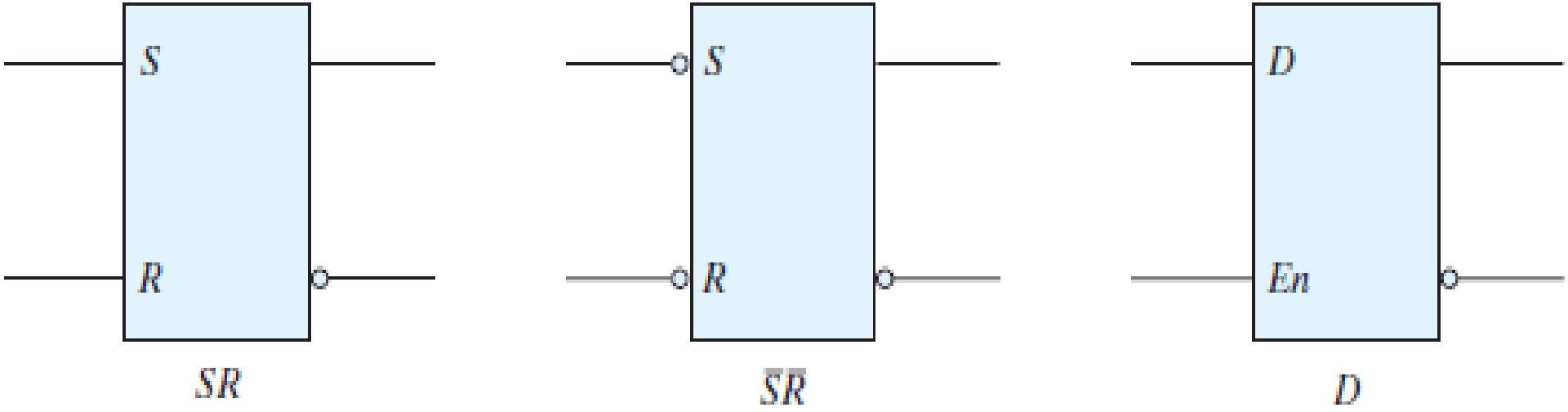


FIGURE 5.7
Graphic symbols for latches

STORAGE ELEMENTS: FLIP-FLOPS

- 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 cause 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 in the level, any changes in the data input will change the output and the state of the latch.

STORAGE ELEMENTS: FLIP-FLOPS

- Flip-flop vs. Latch:
  - Latch w/ control input:
    - can change state in response to the inputs anytime the control input is in the active level.

      $\Rightarrow$ The latch is controlled by the "level" of its control input.
  - Flip-flop:
    - responses only to a "transition" of a triggering input called the "clock."
    - Positive-edge trigger: $0 \rightarrow 1$
    - Negative-edge trigger: $1 \rightarrow 0$

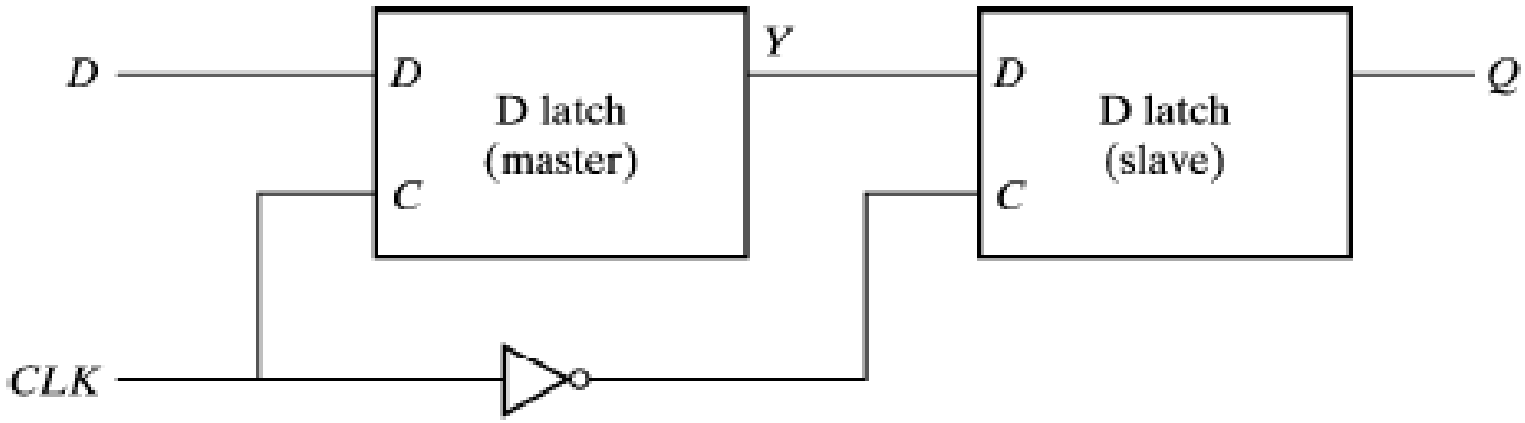
Clock response in latch and flip-flop

(a) Response to positive level

(b) Positive-edge response

(c) Negative-edge response

Edge-Triggered D Flip-Flop



- Constructed with two D latches and an inverter
- The first latch (master) is enabled when CLK=1
  - It reads the input changes but stops before the second one
- The second latch (slave) is enabled when CLK=0
  - Close the first latch to isolate the input changes
  - Deliver the final value at the moment just before CLK changes
- The circuit samples the D input and changes its output Q only at the **negative-edge** of the controlling clock

Edge-Triggered D Flip-Flop

- If only SR latches are available, three latches are required
- Two latches are used for locking the two inputs (CLK & D)
- The final latch provides the output of the flip-flop

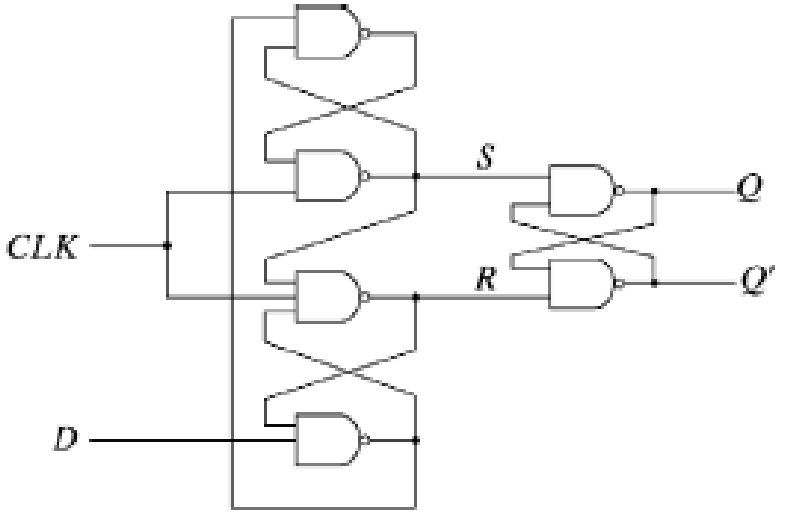
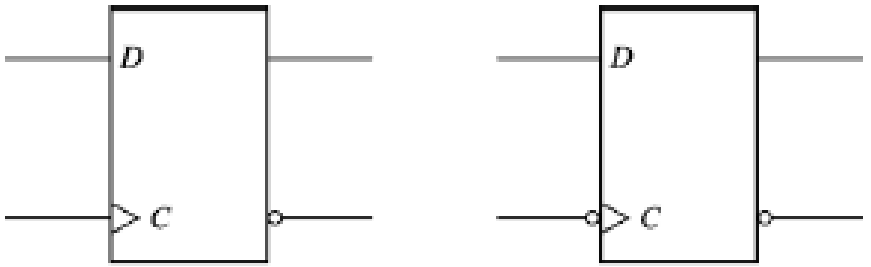


Fig. 5-10 *D*-Type Positive-Edge-Triggered Flip-Flop



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

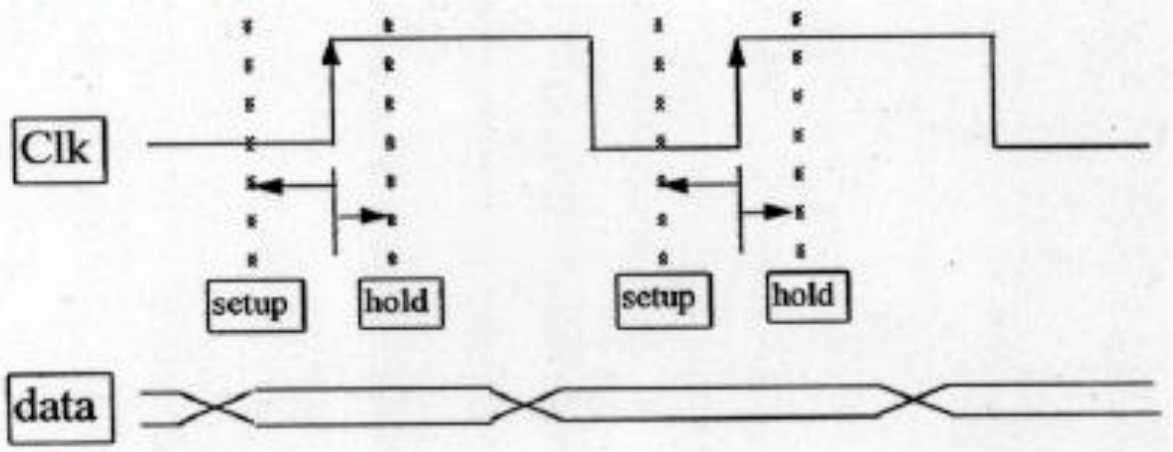
Fig. 5-11 Graphic Symbol for Edge-Triggered *D* Flip-Flop

Setup & Hold Times

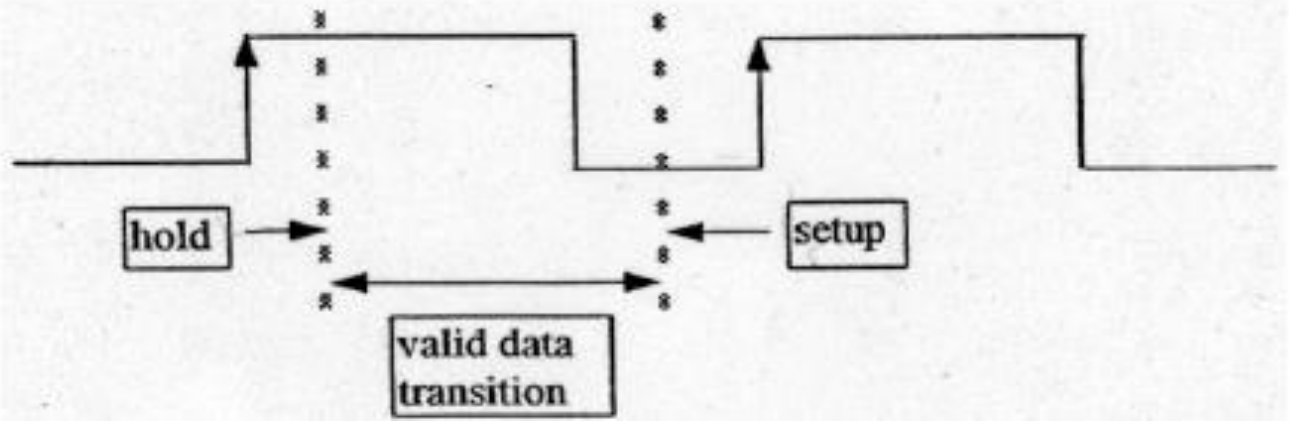
- The response time of a flip-flop to input changes must be taken into consideration
- Setup Time: The length of time that data must stabilize before the clock transition
  - The maximum data path is used to determine if the setup time is met
- Hold Time: The length of time that data must remain stable at the input pin after the active clock transition
  - The minimum data path is used to determine if hold time is met

Setup & Hold Times

- Timing Diagram



- Valid Data Transition

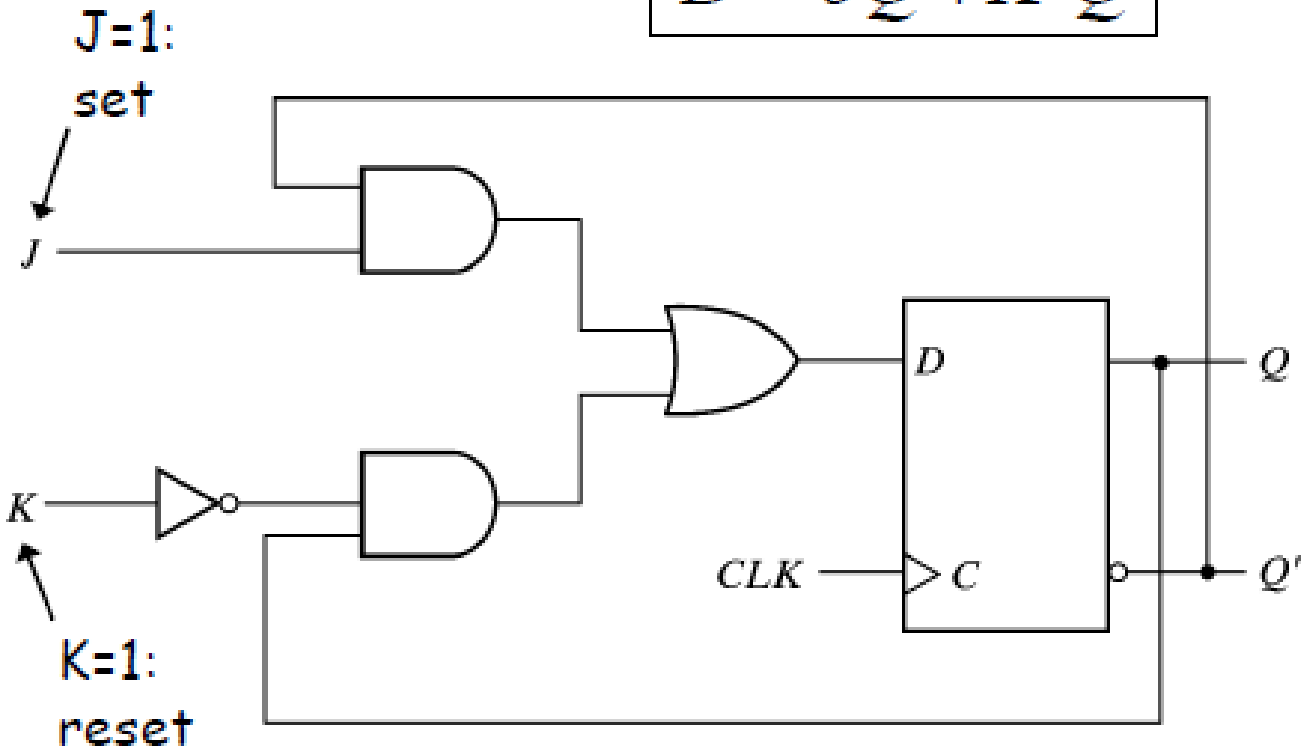


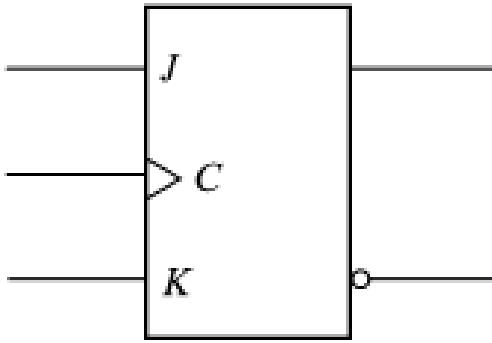
Other Flip-Flops

- The most economical and efficient flip-flop is the edge-triggered D flip-flop
  - It requires the smallest number of gates
- Other types of flip-flops can be constructed by using the D flip-flop and external logic
  - JK flip-flop
  - T flip-flops
- Three major operations that can be performed with a flip-flop:
  - Set it to 1
  - Reset it to 0
  - Complement its output

Edge-Triggered JK Flip-Flop

$$D = JQ' + K'Q$$





J=0,K=0: hold
J=1,K=1: toggle

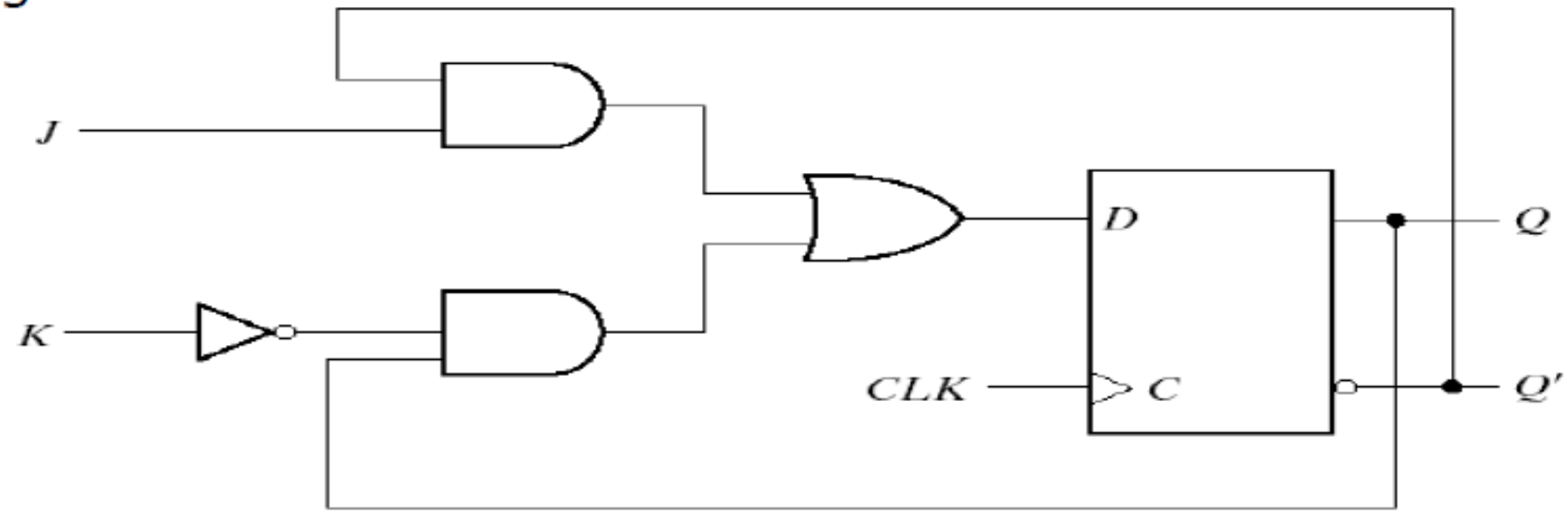
(a) Circuit diagram

(b) Graphic symbol

Fig. 5-12 *JK* Flip-Flop

Other Flip-Flops JK Flip-Flop

There are three operations that can be performed with a flip-flop: set it to 1, reset it to 0, or complement its output. The JK flip-flop performs all three operations. The circuit diagram of a JK flip-flop constructed with a D flip-flop and gates.

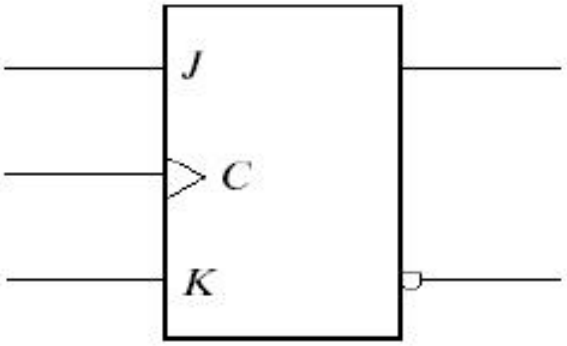


(a) Circuit diagram

JK Flip-Flop

The J input sets the flip-flop to 1, the K input resets it to 0, and when both inputs are enabled, the output is complemented. This can be verified by investigating the circuit applied to the D input:

$$D = J\,Q` + K`\,Q$$



(b) Graphic symbol

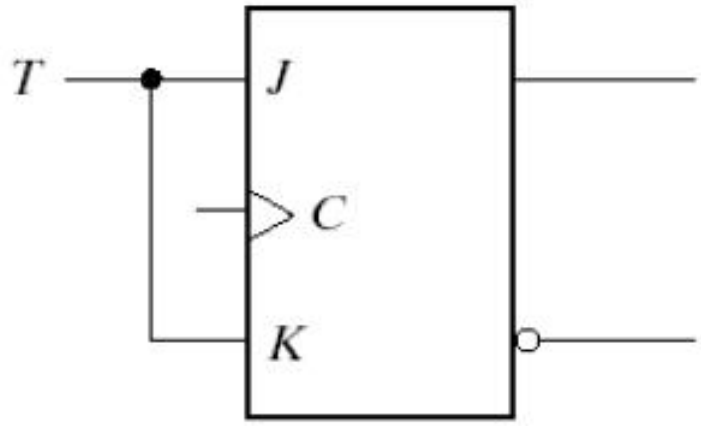
Flip-Flop Characteristic Tables

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

T Flip-Flop

The T(toggle) flip-flop is a complementing flip-flop and can be obtained from a JK flip-flop when inputs J and K are tied together.



(a) From *JK* flip-flop

D Flip-Flop

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

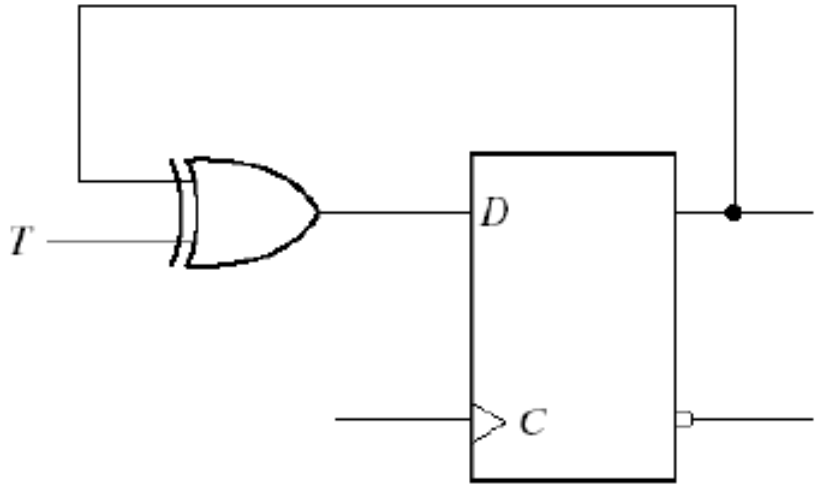
T Flip-Flop

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

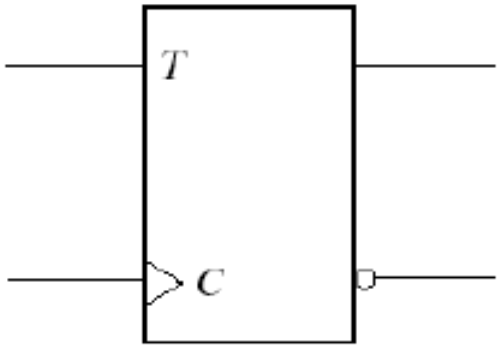
T Flip-Flop

The T flip-flop can be constructed with a D flip-flop and an exclusive-OR gates as shown in Fig. (b). The expression for the D input is

$$D = T \oplus Q = TQ` + T`Q$$



(b) From *D* flip-flop



(c) Graphic symbol

Characteristic Tables

- Define the logical properties in tabular form

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

D Flip-Flop

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

T Flip-Flop

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

Characteristic Equations

- Algebraically describe the next state
- Can be derived from characteristic tables
- D flip-flop:

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

- JK flip-flop:

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

- T flip-flop:

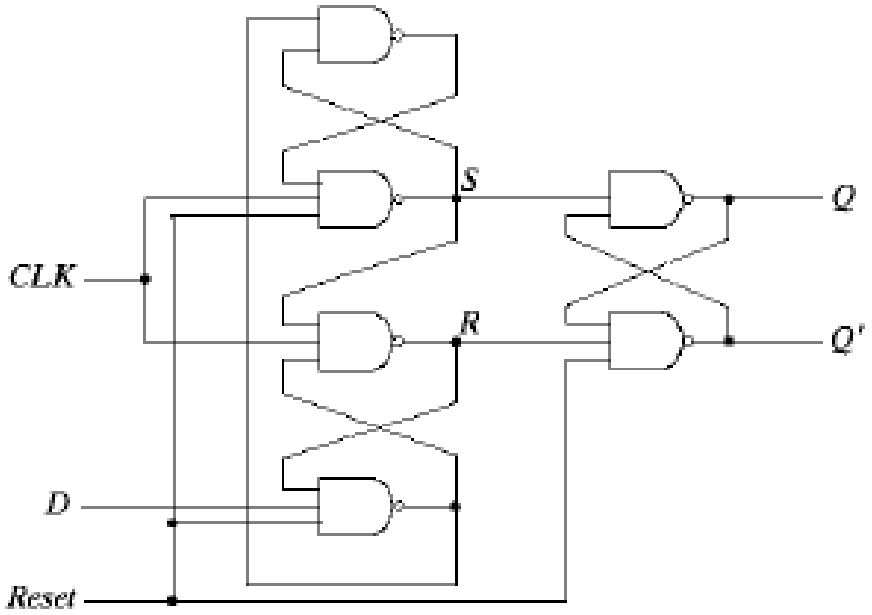
$$Q(t+1) = T \oplus Q = TQ' + T'Q$$

Direct Inputs

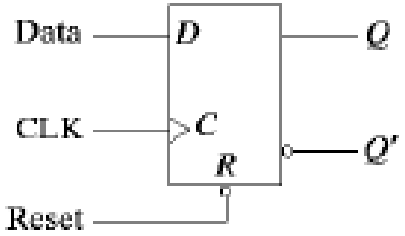
- Force the flip-flop to a particular state immediately
  - Independent of clock signal
  - Have higher priority than any other inputs
  - Useful to bring all flip-flops from unknown into known state while power up
- The input that sets the flip-flop to 1 is called **preset** or **direct set**
- The input that clears the flip-flop to 0 is called **clear** or **direct reset**
- Also called **asynchronous** set/reset

D F/F with Asynchronous Reset

directly change the internal states of all three latches



(a) Circuit diagram



(b) Graphic symbol

R	C	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

(b) Function table

→ other inputs have no effects