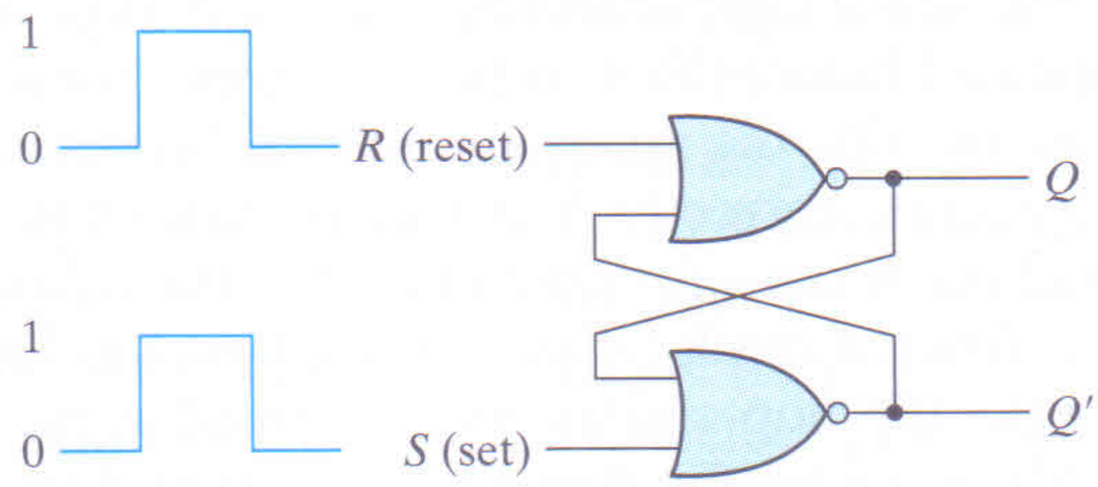
5.3 STORAGE ELEMENTS: LATCHES

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.* 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 before considering flip-flops in the next section.

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 Fig. 5.3. 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. 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. Consequently, in practical applications, setting both inputs to 1 is forbidden.

Under normal conditions, both inputs of the latch remain at 0 unless the state has to be changed. The application of a momentary 1 to the S input causes the latch 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. As shown in the function table of Fig. 5.3(b), two input conditions cause the

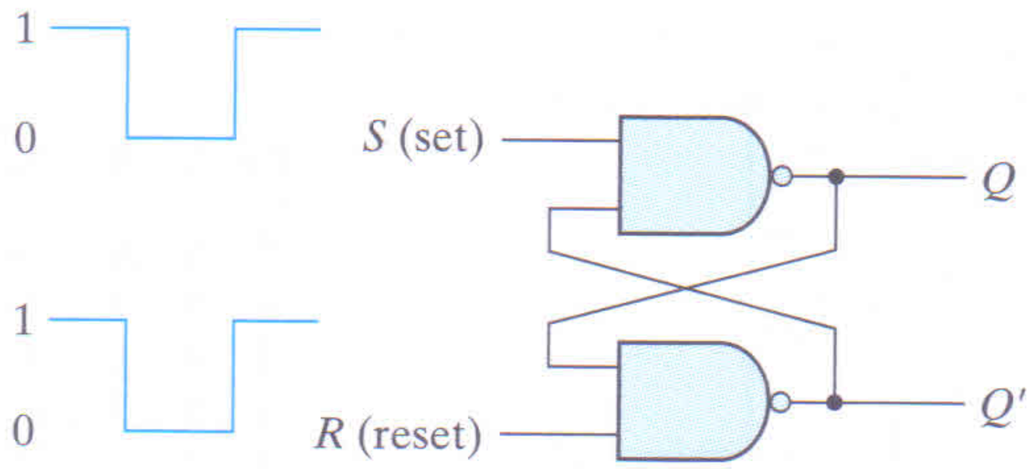


(a) Logic diagram

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**



(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

FIGURE 5.4
***SR* latch with NAND gates**

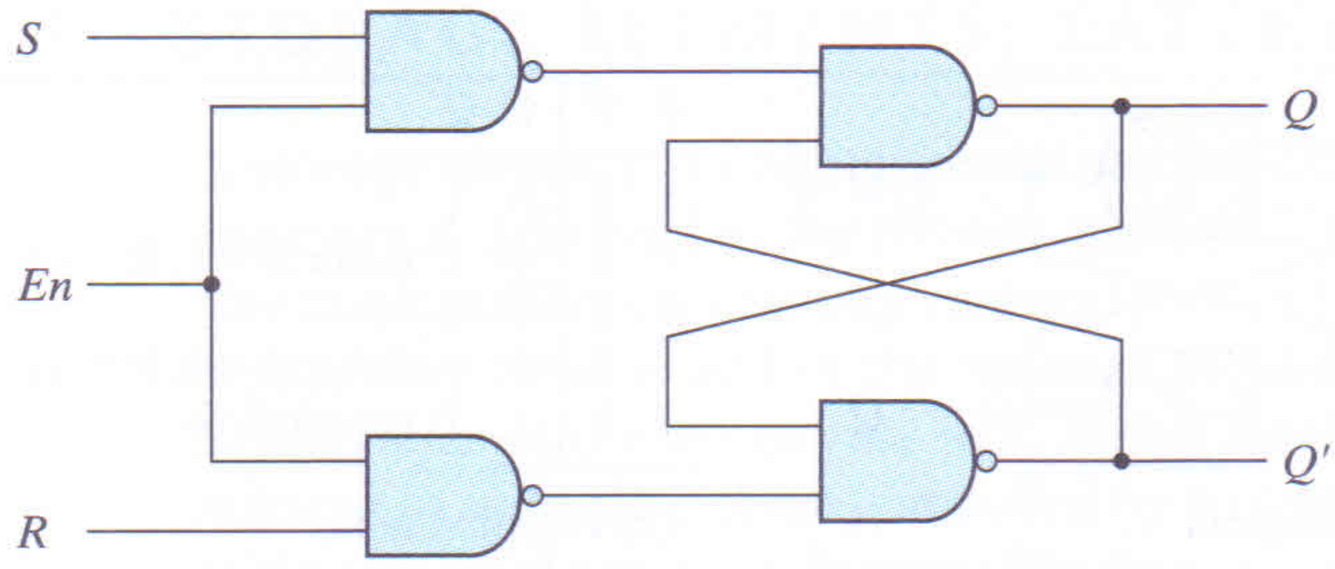
circuit to be in the set state. The first condition ($S = 1, R = 0$) is the action that must be taken by input S to bring the circuit to the set state. Removing the active input from S leaves the circuit in the same state. After both inputs return to 0, it is then possible to shift to the reset state by momentary applying a 1 to the R input. The 1 can then be removed from R, whereupon the circuit remains in the reset state. Thus, when both inputs S and R are equal to 0, the latch 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 of the latch, 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.

The SR latch with two cross-coupled NAND gates is shown in Fig. 5.4. It operates with both inputs normally at 1, unless the state of the latch has to be changed. The application of 0 to the S input causes output Q to go to 1, putting the latch 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 latch 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 latch is both inputs being equal to 0 at the same time, an input combination that should be avoided.

In comparing the NAND with the NOR latch, note that the input signals for the NAND require the complement of those values used for the NOR latch. Because the NAND latch requires a 0 signal to change its state, it is sometimes referred to as an $S'R'$ latch. The primes (or, sometimes, bars over the letters) designate the fact that the inputs must be in their complement form to activate the circuit.

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 is shown in Fig. 5.5. It consists of the basic SR latch and two additional NAND gates. The control input En 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. When the enable input goes to 1, information from the S or R input is allowed to affect the latch. The set state is reached with $S = 1, R = 0$,



(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

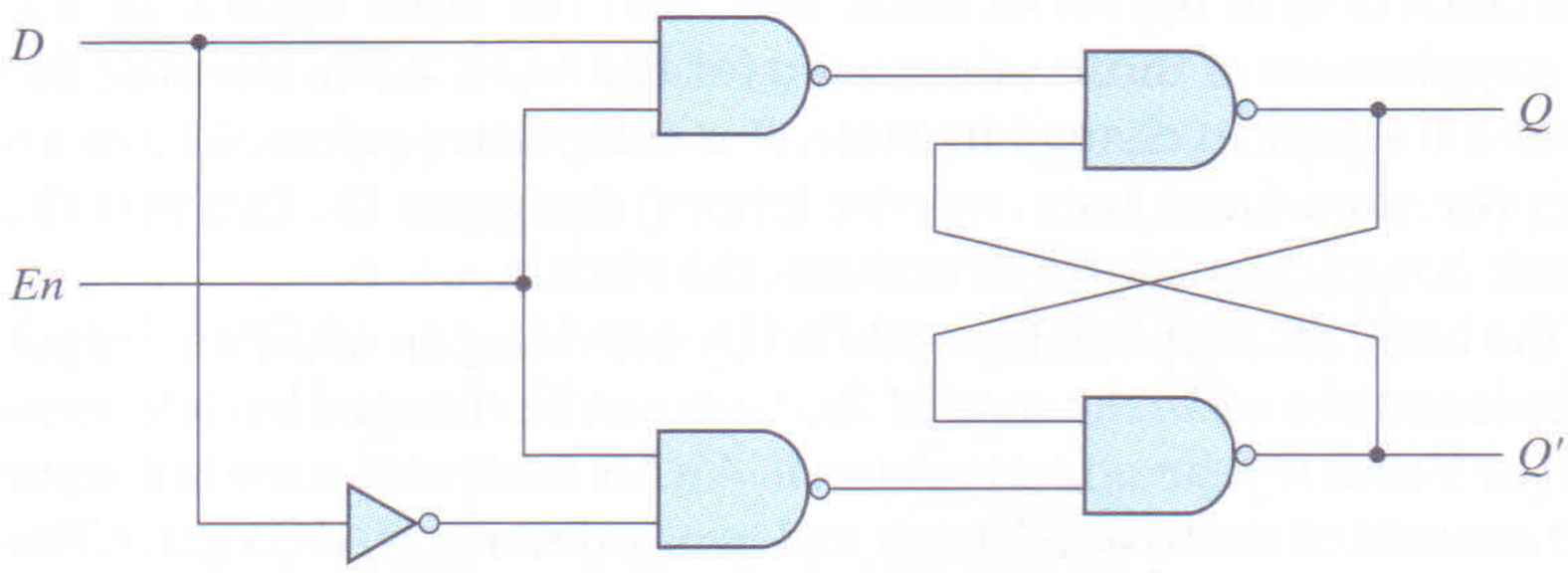
FIGURE 5.5
***SR* latch with control input**

and $En = 1$ (active-high enabled). To change to the reset state, the inputs must be $S = 0$, $R = 1$, and $En = 1$. In either case, when *En* returns to 0, the circuit remains in its current state. The control input disables the circuit by applying 0 to *En*, so that the state of the output does not change regardless of the values of *S* and *R*. Moreover, when $En = 1$ and both the *S* and *R* inputs are equal to 0, the state of the circuit does not change. These conditions are listed in the function table accompanying the diagram.

An indeterminate condition occurs when all three inputs are equal to 1. This condition places 0's on both inputs of the basic *SR* latch, which puts it in the undefined state. When the enable input goes back to 0, one cannot conclusively determine the next state, because it depends on whether the *S* or *R* input goes to 0 first. This indeterminate condition makes this circuit difficult to manage, and it is seldom used in practice. Nevertheless, the *SR* latch is an important circuit because other useful latches and 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 Fig. 5.6. This latch has only two inputs: *D* (data) and



(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**

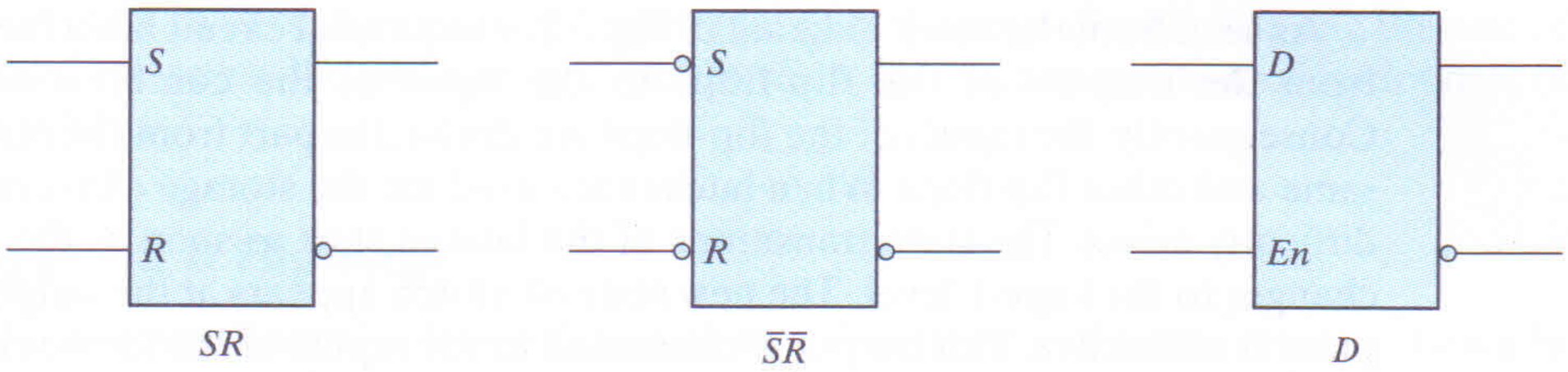


FIGURE 5.7
Graphic symbols for latches

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*. 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.

The *D* latch receives that designation from its ability to hold *data* in its internal storage. It is suited for use as a temporary storage for binary information between a unit and its environment. The binary information present at the data input of the *D* latch is transferred to the *Q* output when the enable input is asserted. The output follows changes in the data input as long as the enable input is asserted. This situation provides a path from input *D* to the output, and for this reason, the circuit is often called a *transparent* latch. When the enable input signal is de-asserted, the binary information that was present at the data input at the time the transition occurred is retained (i.e., stored) at the *Q* output until the enable input is asserted again. Note that an inverter could be placed at the enable input. Then, depending on the physical circuit, the external enabling signal will be a value of 0 (active low) or 1 (active high).

The graphic symbols for the various latches are shown in Fig. 5.7. 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 (with the bubble designation) designates the complement output. The graphic symbol for the *SR* latch has inputs *S* and *R* indicated inside the block. In the case of a NAND gate latch, bubbles are added to the inputs to indicate that setting and resetting occur with a logic-0 signal. The graphic symbol for the *D* latch has inputs *D* and *En* indicated inside the block.

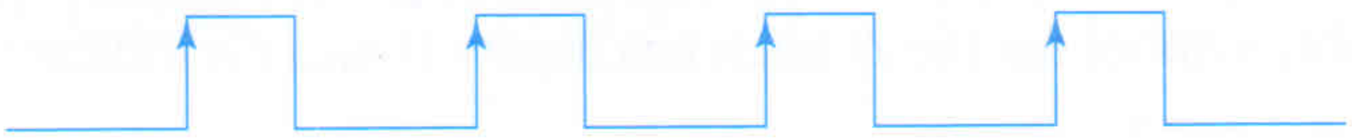
5.4 STORAGE ELEMENTS: FLIP-FLOPS

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 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.

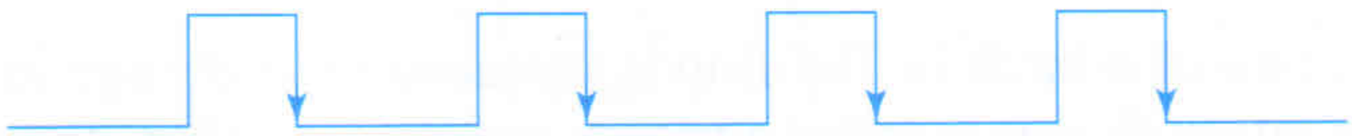
As seen from the block diagram of Fig. 5.2, a sequential circuit has a feedback path from the outputs of the flip-flops to the input of the combinational circuit. Consequently, the inputs of the flip-flops are derived in part from the outputs of the same and other flip-flops. When latches are used for 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. The problem with the latch is that it responds to a change in the *level* of a clock pulse. As shown in Fig. 5.8(a), a positive level response in the enable input allows changes in the output when the *D* input changes while the clock pulse stays at logic 1. The key to the proper operation of a flip-flop is to trigger it only during a signal *transition*. This can be accomplished by eliminating the feedback path that is inherent in the operation of the sequential circuit using latches. A clock pulse goes through two transitions: from 0 to 1 and the return from 1 to 0. As shown in Fig. 5.8, the positive transition is defined as the positive edge and the negative transition as the negative edge. There are two ways that a latch can be modified to form a flip-flop. One way is to employ two latches in a special configuration that isolates the output of the flip-flop and prevents it from being affected while the input to the flip-flop is changing. Another way is to produce a flip-flop that triggers only during a signal transition (from 0 to 1

(a) Response to positive level



(b) Positive-edge response



(c) Negative-edge response

FIGURE 5.8
Clock response in latch and flip-flop

or from 1 to 0) of the synchronizing signal (clock) and is disabled during the rest of the clock pulse. We will now proceed to show the implementation of both types of flip-flops.

Edge-Triggered *D* Flip-Flop

The construction of a *D* flip-flop with two *D* latches and an inverter is shown in Fig. 5.9. The first latch is called the master and the second the slave. 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, and (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*. It is also possible to design the circuit so that the flip-flop output changes on the positive edge of the clock. This happens in a flip-flop that has an additional inverter between the *Clk* terminal and the junction between the other inverter and input *En* of the master latch. Such a flip-flop is triggered with a negative pulse, so that the negative edge of the clock affects the master and the positive edge affects the slave and the output terminal.

Another construction of an edge-triggered *D* flip-flop uses three *SR* latches as shown in Fig. 5.10. Two latches respond to the external *D* (data) and *Clk* (clock)

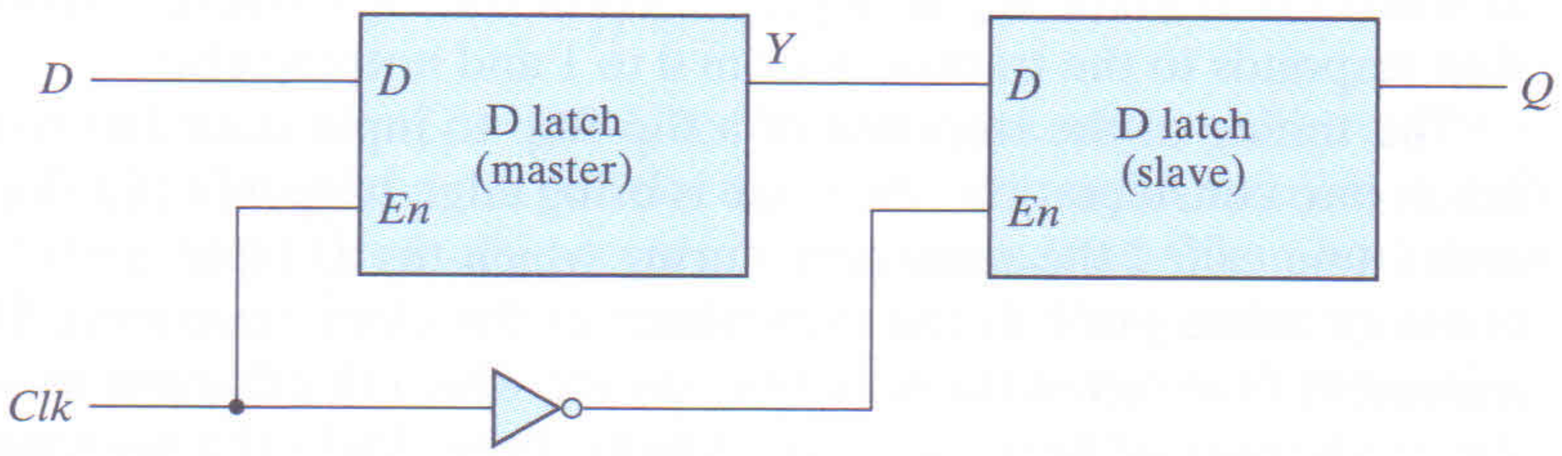


FIGURE 5.9
Master–slave *D* flip-flop

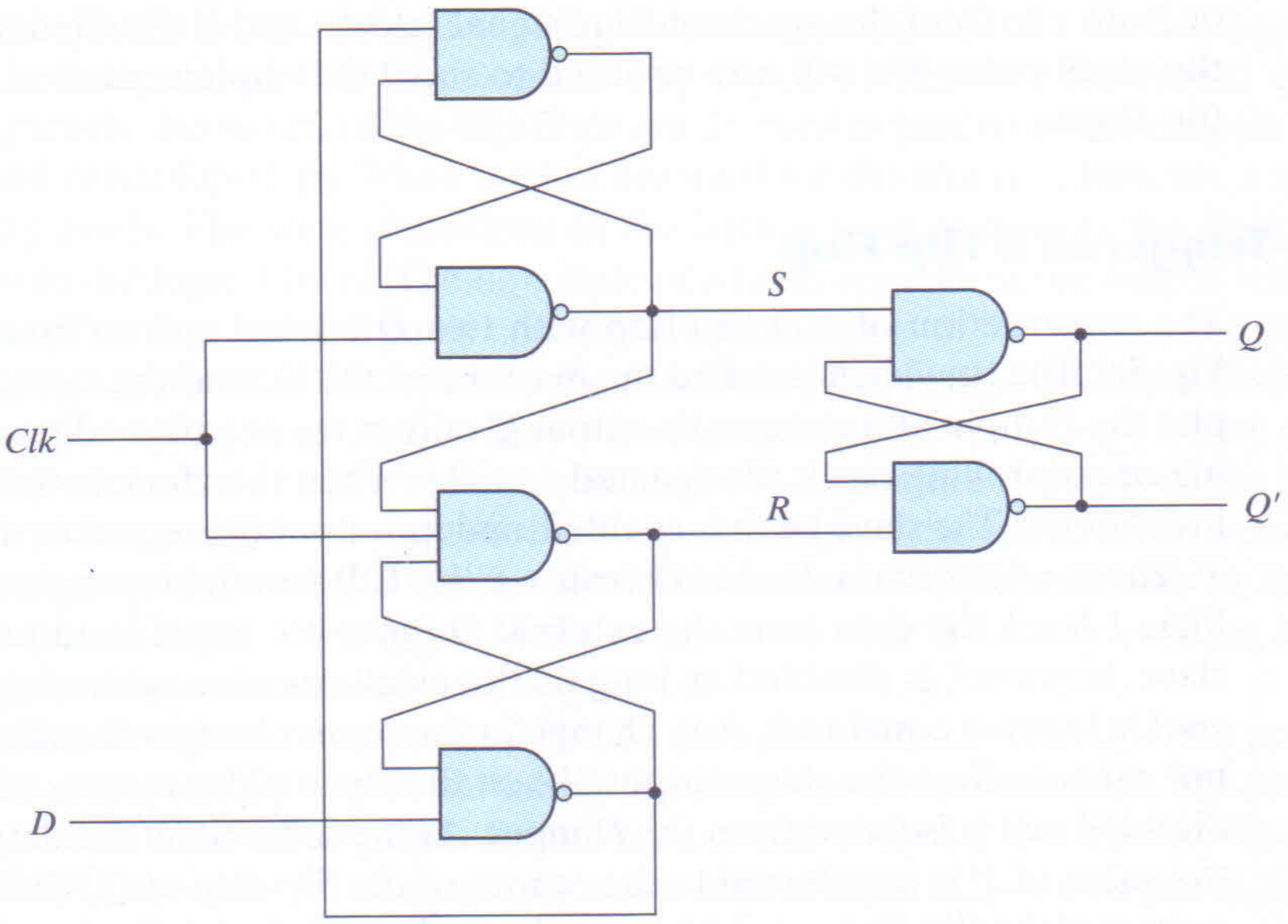


FIGURE 5.10
D-type positive-edge-triggered flip-flop

inputs. The third latch provides the outputs for the flip-flop. 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.

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.

The timing of the response of a flip-flop to input data and to the clock must be taken into consideration when one is using edge-triggered flip-flops. 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. 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. 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. These and other parameters are specified in manufacturers' data books for specific logic families.