

Latches, Flip-Flops, and Timers

CHAPTER OUTLINE

- 7-1 Latches
- 7-2 Flip-Flops
- 7-3 Flip-Flop Operating Characteristics
- 7-4 Flip-Flop Applications
- 7-5 One-Shots
- 7-6 The Astable Multivibrator
- 7-7 Troubleshooting
Applied Logic

CHAPTER OBJECTIVES

- Use logic gates to construct basic latches
- Explain the difference between an S-R latch and a D latch
- Recognize the difference between a latch and a flip-flop
- Explain how D and J-K flip-flops differ
- Understand the significance of propagation delays, set-up time, hold time, maximum operating frequency, minimum clock pulse widths, and power dissipation in the application of flip-flops
- Apply flip-flops in basic applications
- Explain how retriggerable and nonretriggerable one-shots differ
- Connect a 555 timer to operate as either an astable multivibrator or a one-shot
- Describe latches, flip-flops, and timers using VHDL
- Troubleshoot basic flip-flop circuits

KEY TERMS

Key terms are in order of appearance in the chapter.

- Latch
- Bistable
- SET
- RESET

- Clock
- Edge-triggered flip-flop
- D flip-flop
- Synchronous
- J-K flip-flop
- Toggle
- Preset
- Clear
- Propagation delay time
- Set-up time
- Hold time
- Power dissipation
- One-shot
- Monostable
- Timer
- Astable

VISIT THE WEBSITE

Study aids for this chapter are available at <http://www.pearsonglobaleditions.com/floyd>

INTRODUCTION

This chapter begins a study of the fundamentals of sequential logic. Bistable, monostable, and astable logic devices called *multivibrators* are covered. Two categories of bistable devices are the latch and the flip-flop. Bistable devices have two stable states, called SET and RESET; they can retain either of these states indefinitely, making them useful as storage devices. The basic difference between latches and flip-flops is the way in which they are changed from one state to the other. The flip-flop is a basic building block for counters, registers, and other sequential control logic and is used in certain types of memories. The monostable multivibrator, commonly known as the one-shot, has only one stable state. A one-shot produces a single controlled-width pulse when activated or triggered. The astable multivibrator has no stable state and is used primarily as an oscillator, which is a self-sustained waveform generator. Pulse oscillators are used as the sources for timing waveforms in digital systems.

7-1 Latches

The **latch** is a type of temporary storage device that has two stable states (bistable) and is normally placed in a category separate from that of flip-flops. Latches are similar to flip-flops because they are bistable devices that can reside in either of two states using a feedback arrangement, in which the outputs are connected back to the opposite inputs. The main difference between latches and flip-flops is in the method used for changing their state.

After completing this section, you should be able to

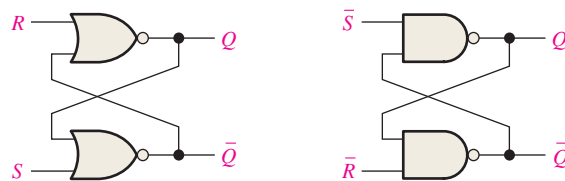
- ◆ Explain the operation of a basic S-R latch
- ◆ Explain the operation of a gated S-R latch
- ◆ Explain the operation of a gated D latch
- ◆ Implement an S-R or D latch with logic gates
- ◆ Describe the 74HC279A and 74HC75 quad latches

InfoNote

Latches are sometimes used for multiplexing data onto a bus. For example, data being input to a computer from an external source have to share the data bus with data from other sources. When the data bus becomes unavailable to the external source, the existing data must be temporarily stored, and latches placed between the external source and the data bus may be used to do this.

The S-R (SET-RESET) Latch

A latch is a type of **bistable** logic device or **multivibrator**. An active-HIGH input S-R (SET-RESET) latch is formed with two cross-coupled NOR gates, as shown in Figure 7-1(a); an active-LOW input $\bar{S}\bar{R}$ latch is formed with two cross-coupled NAND gates, as shown in Figure 7-1(b). Notice that the output of each gate is connected to an input of the opposite gate. This produces the regenerative **feedback** that is characteristic of all latches and flip-flops.



(a) Active-HIGH input S-R latch

(b) Active-LOW input $\bar{S}\bar{R}$ latch



FIGURE 7-1 Two versions of SET-RESET (S-R) latches. Open files F07-01(a) and (b) and verify the operation of both latches. A *Multisim* tutorial is available on the website.

To explain the operation of the latch, we will use the NAND gate $\bar{S}\bar{R}$ latch in Figure 7-1(b). This latch is redrawn in Figure 7-2 with the negative-OR equivalent symbols used for the NAND gates. This is done because LOWs on the \bar{S} and \bar{R} lines are the activating inputs.

The latch in Figure 7-2 has two inputs, \bar{S} and \bar{R} , and two outputs, Q and \bar{Q} . Let's start by assuming that both inputs and the Q output are HIGH, which is the normal latched state. Since the Q output is connected back to an input of gate G_2 , and the \bar{R} input is HIGH, the output of G_2 must be LOW. This LOW output is coupled back to an input of gate G_1 , ensuring that its output is HIGH.

When the Q output is HIGH, the latch is in the **SET** state. It will remain in this state indefinitely until a LOW is temporarily applied to the \bar{R} input. With a LOW on the \bar{R} input and a HIGH on \bar{S} , the output of gate G_2 is forced HIGH. This HIGH on the \bar{Q} output is coupled back to an input of G_1 , and since the \bar{S} input is HIGH, the output of G_1 goes LOW. This LOW on the Q output is then coupled back to an input of G_2 , ensuring that the \bar{Q} output remains HIGH even when the LOW on the \bar{R} input is removed. When the Q output is LOW, the latch is in the **RESET** state. Now the latch remains indefinitely in the RESET state until a momentary LOW is applied to the \bar{S} input.

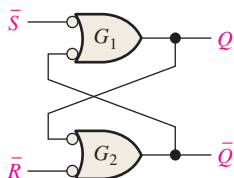


FIGURE 7-2 Negative-OR equivalent of the NAND gate $\bar{S}\bar{R}$ latch in Figure 7-1(b).

A latch can reside in either of its two states, **SET** or **RESET**.

In normal operation, the outputs of a latch are always complements of each other.

When Q is HIGH, \bar{Q} is LOW, and when Q is LOW, \bar{Q} is HIGH.

An invalid condition in the operation of an active-LOW input $\bar{S}\text{-}\bar{R}$ latch occurs when LOWs are applied to both \bar{S} and \bar{R} at the same time. As long as the LOW levels are simultaneously held on the inputs, both the Q and \bar{Q} outputs are forced HIGH, thus violating the basic complementary operation of the outputs. Also, if the LOWs are released simultaneously, both outputs will attempt to go LOW. Since there is always some small difference in the propagation delay time of the gates, one of the gates will dominate in its transition to the LOW output state. This, in turn, forces the output of the slower gate to remain HIGH. In this situation, you cannot reliably predict the next state of the latch.

SET means that the Q output is HIGH.

RESET means that the Q output is LOW.

Figure 7-3 illustrates the active-LOW input $\bar{S}\text{-}\bar{R}$ latch operation for each of the four possible combinations of levels on the inputs. (The first three combinations are valid, but the last is not.) Table 7-1 summarizes the logic operation in truth table form. Operation of the active-HIGH input NOR gate latch in Figure 7-1(a) is similar but requires the use of opposite logic levels.

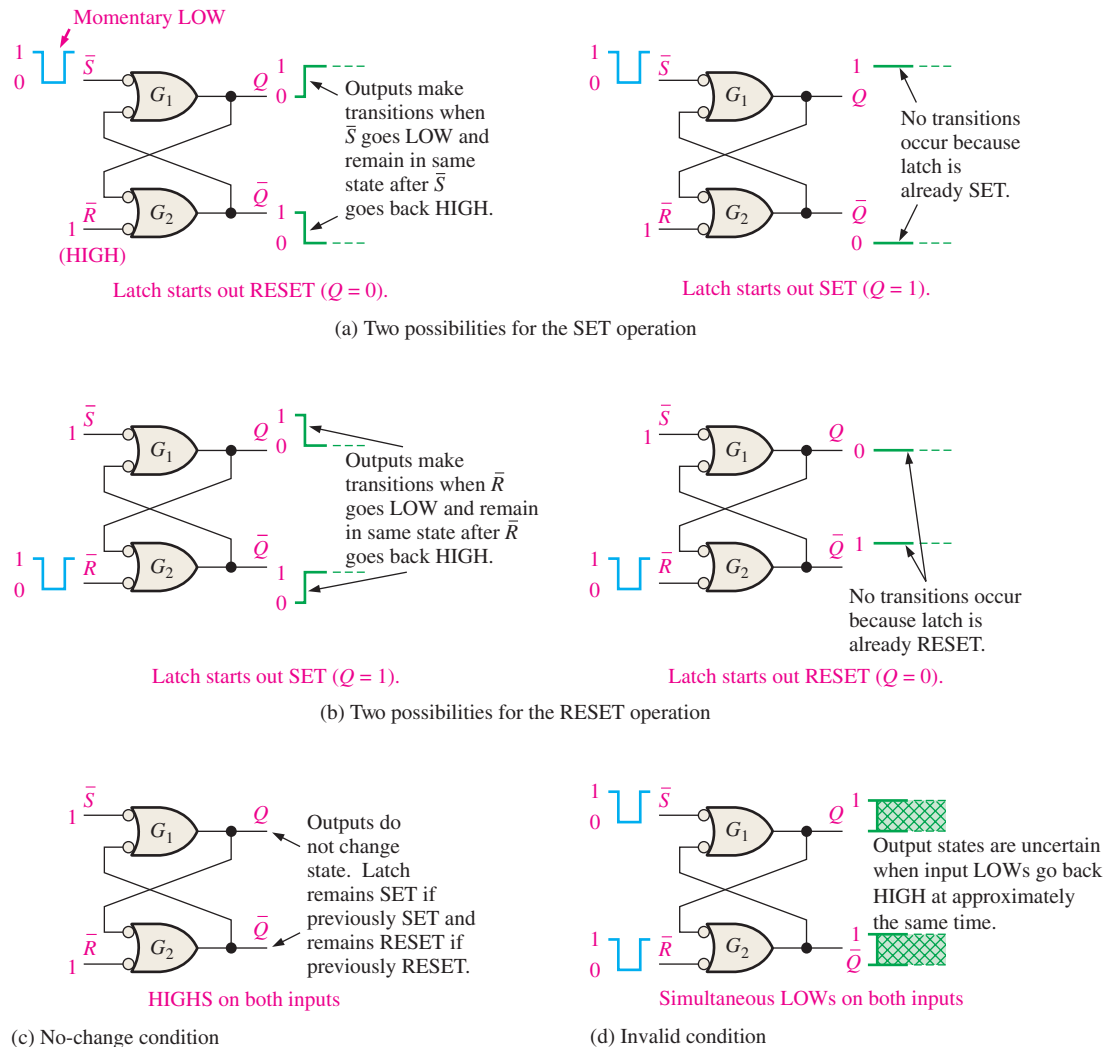


FIGURE 7-3 The three modes of basic $\bar{S}\text{-}\bar{R}$ latch operation (SET, RESET, no-change) and the invalid condition.

TABLE 7-1

Truth table for an active-LOW input $\bar{S}\text{-}\bar{R}$ latch.

Inputs		Outputs		Comments
\bar{S}	\bar{R}	Q	\bar{Q}	
1	1	NC	NC	No change. Latch remains in present state.
0	1	1	0	Latch SET.
1	0	0	1	Latch RESET.
0	0	1	1	Invalid condition

Logic symbols for both the active-HIGH input and the active-LOW input latches are shown in Figure 7-4.

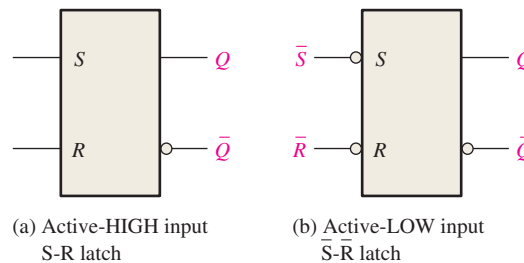


FIGURE 7-4 Logic symbols for the S-R and $\bar{S}\text{-}\bar{R}$ latch.

Example 7-1 illustrates how an active-LOW input $\bar{S}\text{-}\bar{R}$ latch responds to conditions on its inputs. LOW levels are pulsed on each input in a certain sequence and the resulting Q output waveform is observed. The $\bar{S} = 0, \bar{R} = 0$ condition is avoided because it results in an invalid mode of operation and is a major drawback of any SET-RESET type of latch.

EXAMPLE 7-1

If the \bar{S} and \bar{R} waveforms in Figure 7-5(a) are applied to the inputs of the latch in Figure 7-4(b), determine the waveform that will be observed on the Q output. Assume that Q is initially LOW.

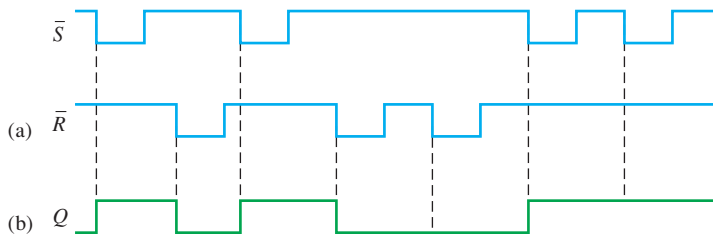


FIGURE 7-5

Solution

See Figure 7-5(b).

Related Problem*

Determine the Q output of an active-HIGH input S-R latch if the waveforms in Figure 7-5(a) are inverted and applied to the inputs.

*Answers are at the end of the chapter.

An Application

The Latch as a Contact-Bounce Eliminator

A good example of an application of an $\bar{S}\text{-}\bar{R}$ latch is in the elimination of mechanical switch contact “bounce.” When the pole of a switch strikes the contact upon switch closure, it physically vibrates or bounces several times before finally making a solid contact. Although these bounces are very short in duration, they produce voltage spikes that are often not acceptable in a digital system. This situation is illustrated in Figure 7–6(a).

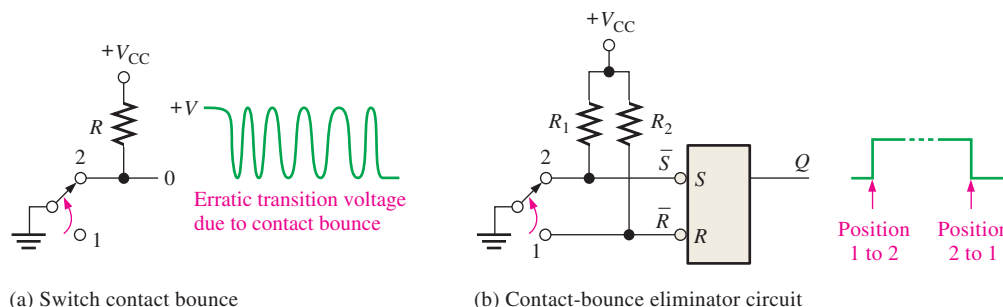
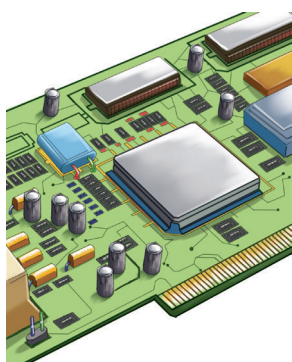


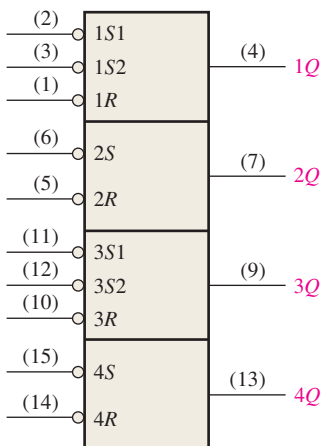
FIGURE 7-6 The $\bar{S}\text{-}\bar{R}$ latch used to eliminate switch contact bounce.

An $\bar{S}\text{-}\bar{R}$ latch can be used to eliminate the effects of switch bounce as shown in Figure 7–6(b). The switch is normally in position 1, keeping the \bar{R} input LOW and the latch RESET. When the switch is thrown to position 2, \bar{R} goes HIGH because of the pull-up resistor to V_{CC} , and \bar{S} goes LOW on the first contact. Although \bar{S} remains LOW for only a very short time before the switch bounces, this is sufficient to set the latch. Any further voltage spikes on the \bar{S} input due to switch bounce do not affect the latch, and it remains SET. Notice that the Q output of the latch provides a clean transition from LOW to HIGH, thus eliminating the voltage spikes caused by contact bounce. Similarly, a clean transition from HIGH to LOW is made when the switch is thrown back to position 1.

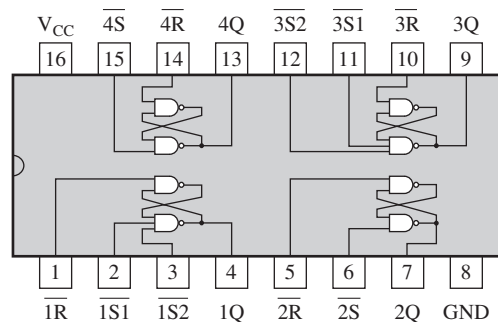
IMPLEMENTATION: $\bar{S}\text{-}\bar{R}$ LATCH



Fixed-Function Device The 74HC279A is a quad $\bar{S}\text{-}\bar{R}$ latch represented by the logic diagram of Figure 7–7(a) and the pin diagram in part (b). Notice that two of the latches each have two \bar{S} inputs.



(a) Logic diagram



(b) Pin diagram

FIGURE 7-7 The 74HC279A quad $\bar{S}\text{-}\bar{R}$ latch.

Programmable Logic Device (PLD) An \overline{S} - \overline{R} latch can be described using VHDL and implemented as hardware in a PLD. VHDL statements and keywords not used in previous chapters are introduced in this chapter. These are **library**, **use**, **std_logic**, **all**, and **inout**. The data flow approach is used in this program to describe a single \overline{S} - \overline{R} latch. (The blue comments are not part of the program.)



```

entity SRLatch is
  port (SNot, RNot: in std_logic; Q, QNot: inout std_logic);
end entity SRLatch;

architecture LogicOperation of SRLatch is
begin
  Q <= QNot nand SNot; } Boolean expressions
  QNot <= Q nand RNot; } define the outputs
end architecture LogicOperation;
    
```

SNot: SET complement
RNot: RESET complement
Q: Latch output
QNot: Latch output complement

The two inputs SNot and RNot are defined as **std_logic** from the IEEE library. The **inout** keyword allows the Q and QNot outputs of the latch to be used also as inputs for cross-coupling.

The Gated S-R Latch

A gated latch requires an enable input, *EN* (*G* is also used to designate an enable input). The logic diagram and logic symbol for a gated S-R latch are shown in Figure 7–8. The *S* and *R* inputs control the state to which the latch will go when a HIGH level is applied to the *EN* input. The latch will not change until *EN* is HIGH; but as long as it remains HIGH, the output is controlled by the state of the *S* and *R* inputs. The gated latch is a *level-sensitive* device. In this circuit, the invalid state occurs when both *S* and *R* are simultaneously HIGH and *EN* is also HIGH.

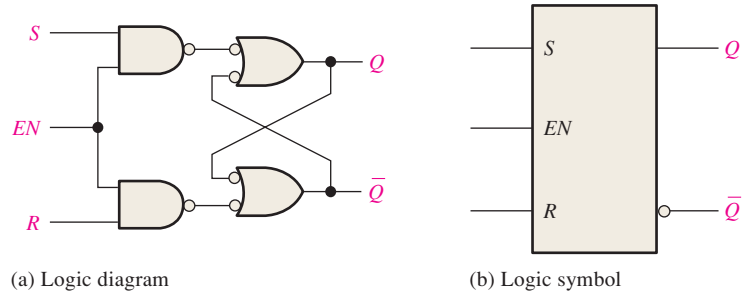


FIGURE 7-8 A gated S-R latch.

EXAMPLE 7-2

Determine the *Q* output waveform if the inputs shown in Figure 7–9(a) are applied to a gated S-R latch that is initially RESET.

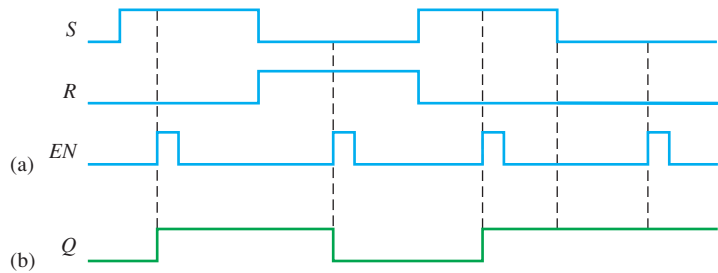


FIGURE 7-9

Solution

The Q waveform is shown in Figure 7–9(b). When S is HIGH and R is LOW, a HIGH on the EN input sets the latch. When S is LOW and R is HIGH, a HIGH on the EN input resets the latch. When both S and R are LOW, the Q output does not change from its present state.

Related Problem

Determine the Q output of a gated S-R latch if the S and R inputs in Figure 7–9(a) are inverted.

The Gated D Latch

Another type of gated latch is called the D latch. It differs from the S-R latch because it has only one input in addition to EN . This input is called the D (data) input. Figure 7–10 contains a logic diagram and logic symbol of a D latch. When the D input is HIGH and the EN input is HIGH, the latch will set. When the D input is LOW and EN is HIGH, the latch will reset. Stated another way, the output Q follows the input D when EN is HIGH.

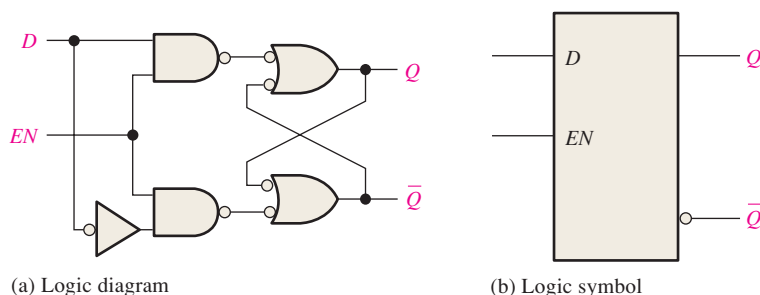


FIGURE 7-10 A gated D latch. Open file F07-10 and verify the operation.

**EXAMPLE 7-3**

Determine the Q output waveform if the inputs shown in Figure 7–11(a) are applied to a gated D latch, which is initially RESET.

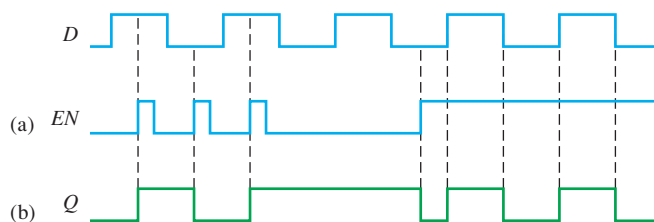


FIGURE 7-11

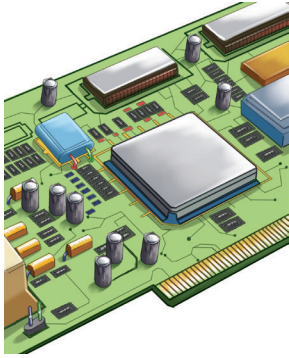
Solution

The Q waveform is shown in Figure 7–11(b). When D is HIGH and EN is HIGH, Q goes HIGH. When D is LOW and EN is HIGH, Q goes LOW. When EN is LOW, the state of the latch is not affected by the D input.

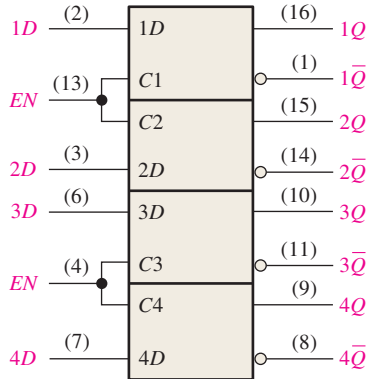
Related Problem

Determine the Q output of the gated D latch if the D input in Figure 7–11(a) is inverted.

IMPLEMENTATION: GATED D LATCH



Fixed-Function Device An example of a gated D latch is the 74HC75 represented by the logic symbol in Figure 7–12(a). The device has four latches. Notice that each active-HIGH *EN* input is shared by two latches and is designated as a control input (*C*). The truth table for each latch is shown in Figure 7–12(b). The X in the truth table represents a “don’t care” condition. In this case, when the *EN* input is LOW, it does not matter what the *D* input is because the outputs are unaffected and remain in their prior states.



(a) Logic symbol

Inputs		Outputs		Comments
<i>D</i>	<i>EN</i>	<i>Q</i>	\bar{Q}	
0	1	0	1	RESET
1	1	1	0	SET
X	0	Q_0	\bar{Q}_0	No change

Note: Q_0 is the prior output level before the indicated input conditions were established.

(b) Truth table (each latch)

FIGURE 7-12 The 74HC75 quad D latch.

Programmable Logic Device (PLD) The gated D latch can be described using VHDL and implemented as hardware in a PLD. The data flow approach is used in this program to describe a single D latch.



```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity DLatch1 is
    port (D, EN: in std_logic; Q, QNot: inout std_logic);
end entity DLatch1;
```

```
architecture LogicOperation of DLatch1 is
begin
```

```
    Q <= QNot nand (D nand EN);
    QNot <= Q nand (not D nand EN);
end architecture LogicOperation;
```

D: Data input
EN: Enable
Q: Latch output
QNot: Latch output complement

SECTION 7-1 CHECKUP

Answers are at the end of the chapter.

1. List three types of latches.
2. Develop the truth table for the active-HIGH input S-R latch in Figure 7–1(a).
3. What is the *Q* output of a D latch when $EN = 1$ and $D = 1$?

7-2 Flip-Flops

Flip-flops are synchronous bistable devices, also known as *bistable multivibrators*. In this case, the term *synchronous* means that the output changes state only at a specified point (leading or trailing edge) on the triggering input called the **clock** (CLK), which is designated as a control input, C ; that is, changes in the output occur in synchronization with the clock. Flip-flops are edge-triggered or edge-sensitive whereas gated latches are level-sensitive.

After completing this section, you should be able to

- ◆ Define *clock*
- ◆ Define *edge-triggered flip-flop*
- ◆ Explain the difference between a flip-flop and a latch
- ◆ Identify an edge-triggered flip-flop by its logic symbol
- ◆ Discuss the difference between a positive and a negative edge-triggered flip-flop
- ◆ Discuss and compare the operation of D and J-K edge-triggered flip-flops and explain the differences in their truth tables
- ◆ Discuss the asynchronous inputs of a flip-flop

An **edge-triggered flip-flop** changes state either at the positive edge (rising edge) or at the negative edge (falling edge) of the clock pulse and is sensitive to its inputs only at this transition of the clock. Two types of edge-triggered flip-flops are covered in this section: D and J-K. The logic symbols for these flip-flops are shown in Figure 7-13. Notice that each type can be either positive edge-triggered (no bubble at C input) or negative edge-triggered (bubble at C input). The key to identifying an edge-triggered flip-flop by its logic symbol is the small triangle inside the block at the clock (C) input. This triangle is called the *dynamic input indicator*:

The dynamic input indicator \triangleright means the flip-flop changes state only on the edge of a clock pulse.

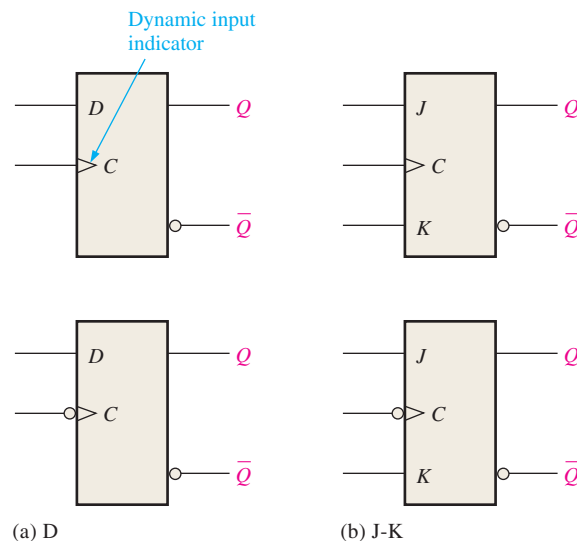


FIGURE 7-13 Edge-triggered flip-flop logic symbols (top: positive edge-triggered; bottom: negative edge-triggered).

The D Flip-Flop

The D input of the **D flip-flop** is a **synchronous** input because data on the input are transferred to the flip-flop's output only on the triggering edge of the clock pulse. When D is HIGH, the Q output goes HIGH on the triggering edge of the clock pulse, and the flip-flop

D flip-flop but D as variable.

InfoNote

Semiconductor memories consist of large numbers of individual cells. Each storage cell holds a 1 or a 0. One type of memory is the Static Random Access Memory or SRAM, which uses flip-flops for the storage cells because a flip-flop will retain either of its two states indefinitely as long as dc power is applied, thus the term *static*. This type of memory is classified as a *volatile* memory because all the stored data are lost when power is turned off. Another type of memory, the Dynamic Random Access Memory or DRAM, uses capacitance rather than flip-flops as the basic storage element and must be periodically refreshed in order to maintain the stored data.

is SET. When D is LOW, the Q output goes LOW on the triggering edge of the clock pulse, and the flip-flop is RESET.

This basic operation of a positive edge-triggered D flip-flop is illustrated in Figure 7–14, and Table 7–2 is the truth table for this type of flip-flop. Remember, *the flip-flop cannot change state except on the triggering edge of a clock pulse*. The D input can be changed at any time when the clock input is LOW or HIGH (except for a very short interval around the triggering transition of the clock) without affecting the output. Just remember, Q follows D at the triggering edge of the clock.

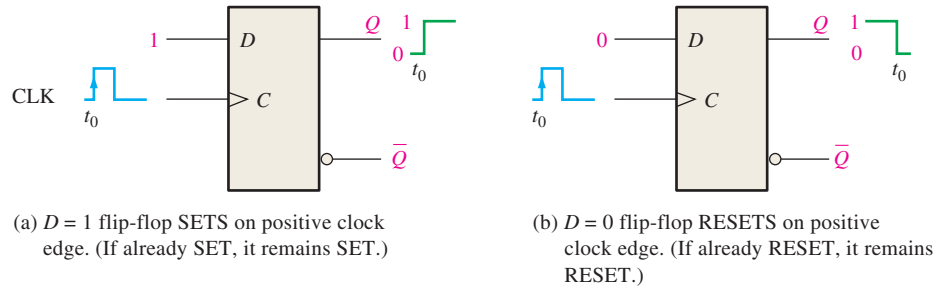


FIGURE 7-14 Operation of a positive edge-triggered D flip-flop.

TABLE 7-2

Truth table for a positive edge-triggered D flip-flop.

Inputs		Outputs		Comments
D	CLK	Q	\bar{Q}	
0	↑	0	1	RESET
1	↑	1	0	SET

↑ = clock transition LOW to HIGH

The operation and truth table for a negative edge-triggered D flip-flop are the same as those for a positive edge-triggered device except that the falling edge of the clock pulse is the triggering edge.

EXAMPLE 7-4

Determine the Q and \bar{Q} output waveforms of the flip-flop in Figure 7–15 for the D and CLK inputs in Figure 7–16(a). Assume that the positive edge-triggered flip-flop is initially RESET.

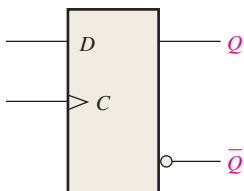


FIGURE 7-15

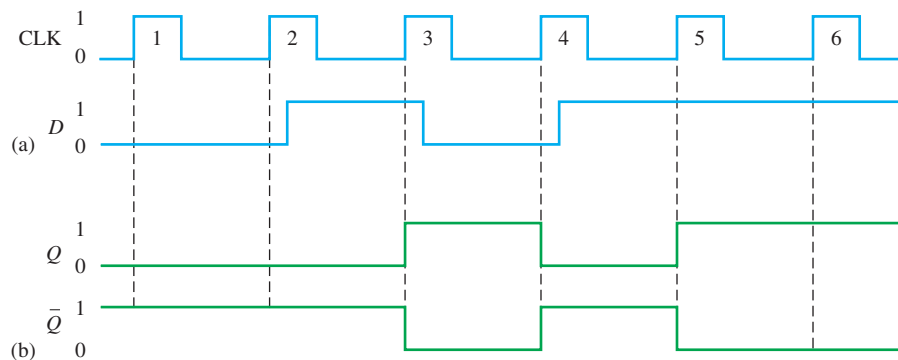


FIGURE 7-16

Solution

1. At clock pulse 1, D is LOW, so Q remains LOW (RESET).
2. At clock pulse 2, D is LOW, so Q remains LOW (RESET).
3. At clock pulse 3, D is HIGH, so Q goes HIGH (SET).
4. At clock pulse 4, D is LOW, so Q goes LOW (RESET).
5. At clock pulse 5, D is HIGH, so Q goes HIGH (SET).
6. At clock pulse 6, D is HIGH, so Q remains HIGH (SET).

Once Q is determined, \bar{Q} is easily found since it is simply the complement of Q . The resulting waveforms for Q and \bar{Q} are shown in Figure 7–16(b) for the input waveforms in part (a).

Related Problem

Determine Q and \bar{Q} for the D input in Figure 7–16(a) if the flip-flop is a negative edge-triggered device.

The J-K Flip-Flop

The J and K inputs of the **J-K flip-flop** are synchronous inputs because data on these inputs are transferred to the flip-flop's output only on the triggering edge of the clock pulse. When J is HIGH and K is LOW, the Q output goes HIGH on the triggering edge of the clock pulse, and the flip-flop is SET. When J is LOW and K is HIGH, the Q output goes LOW on the triggering edge of the clock pulse, and the flip-flop is RESET. When both J and K are LOW, the output does not change from its prior state. When J and K are both HIGH, the flip-flop changes state. This called the **toggle** mode.

This basic operation of a positive edge-triggered flip-flop is illustrated in Figure 7–17, and Table 7–3 is the truth table for this type of flip-flop. Remember, *the flip-flop cannot change state except on the triggering edge of a clock pulse*. The J and K inputs can be changed at any time when the clock input is LOW or HIGH (except for a very short interval around the triggering transition of the clock) without affecting the output.

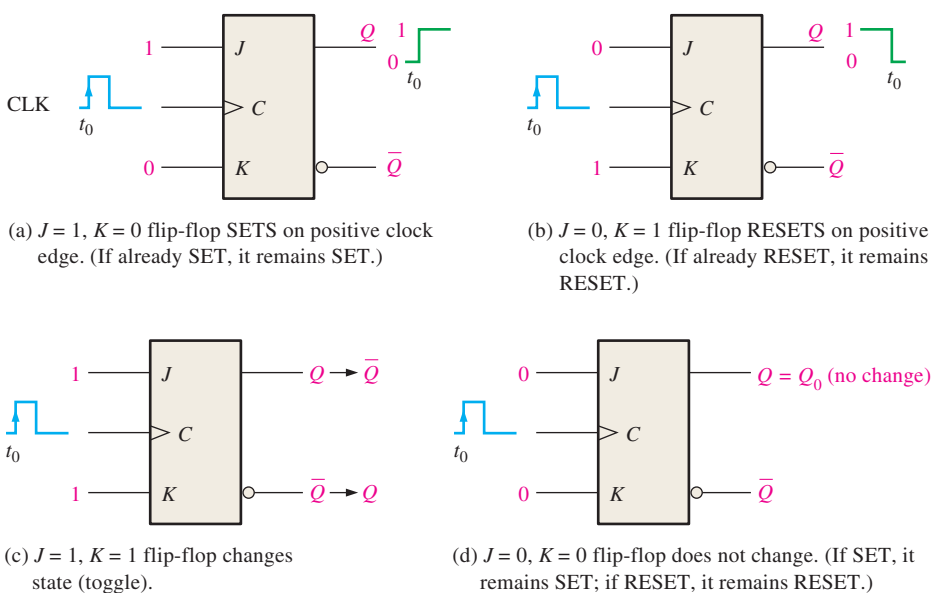


FIGURE 7-17 Operation of a positive edge-triggered J-K flip-flop.

TABLE 7-3

Truth table for a positive edge-triggered J-K flip-flop.

Inputs			Outputs		Comments
<i>J</i>	<i>K</i>	CLK	<i>Q</i>	\bar{Q}	
0	0	↑	Q_0	\bar{Q}_0	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	\bar{Q}_0	Q_0	Toggle

↑ = clock transition LOW to HIGH
 Q_0 = output level prior to clock transition

EXAMPLE 7-5

The waveforms in Figure 7-18(a) are applied to the *J*, *K*, and clock inputs as indicated. Determine the *Q* output, assuming that the flip-flop is initially RESET.

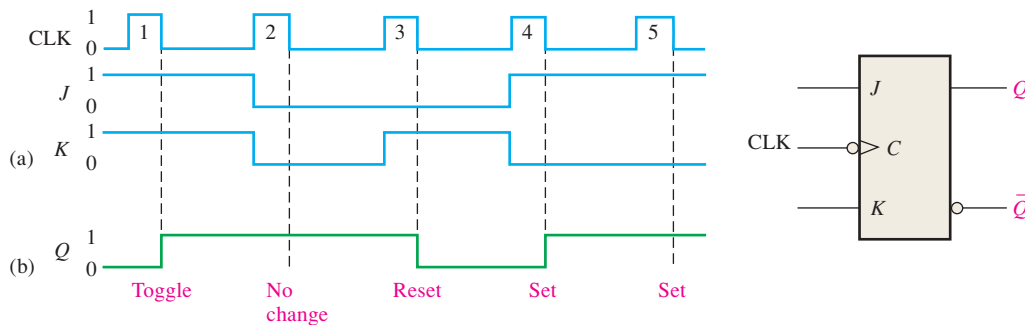


FIGURE 7-18

Solution

Since this is a negative edge-triggered flip-flop, as indicated by the “bubble” at the clock input, the *Q* output will change only on the negative-going edge of the clock pulse.

1. At the first clock pulse, both *J* and *K* are HIGH; and because this is a toggle condition, *Q* goes HIGH.
2. At clock pulse 2, a no-change condition exists on the inputs, keeping *Q* at a HIGH level.
3. When clock pulse 3 occurs, *J* is LOW and *K* is HIGH, resulting in a RESET condition; *Q* goes LOW.
4. At clock pulse 4, *J* is HIGH and *K* is LOW, resulting in a SET condition; *Q* goes HIGH.
5. A SET condition still exists on *J* and *K* when clock pulse 5 occurs, so *Q* will remain HIGH.

The resulting *Q* waveform is indicated in Figure 7-18(b).

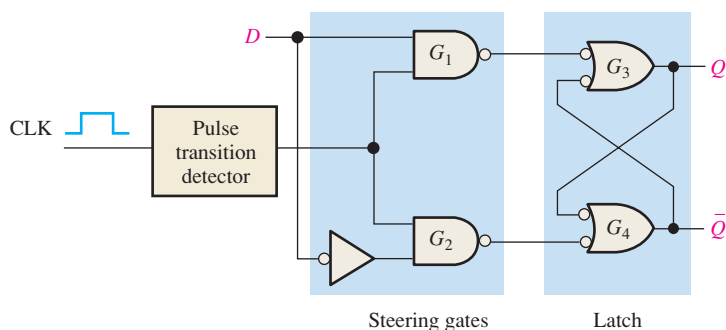
Related Problem

Determine the *Q* output of the J-K flip-flop if the *J* and *K* inputs in Figure 7-18(a) are inverted.

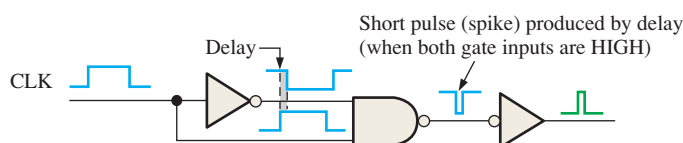
Edge-Triggered Operation

D Flip-Flop

A simplified implementation of an edge-triggered D flip-flop is illustrated in Figure 7-19(a) and is used to demonstrate the concept of edge-triggering. Notice that the basic D flip-flop differs from the gated D latch only in that it has a pulse transition detector.



(a) A simplified logic diagram for a positive edge-triggered D flip-flop



(b) A type of pulse transition detector

FIGURE 7-19 Edge triggering.

One basic type of pulse transition detector is shown in Figure 7-19(b). As you can see, there is a small delay through the inverter on one input to the NAND gate so that the inverted clock pulse arrives at the gate input a few nanoseconds after the true clock pulse. This circuit produces a very short-duration spike on the positive-going transition of the clock pulse. In a negative edge-triggered flip-flop the clock pulse is inverted first, thus producing a narrow spike on the negative-going edge.

The circuit in Figure 7-19(a) is partitioned into two sections, one labeled Steering gates and the other labeled Latch. The steering gates direct, or steer, the clock spike either to the input to gate G_3 or to the input to gate G_4 , depending on the state of the D input. To understand the operation of this flip-flop, begin with the assumptions that it is in the RESET state ($Q = 0$) and that the D and CLK inputs are LOW. For this condition, the outputs of gate G_1 and gate G_2 are both HIGH. The LOW on the Q output is coupled back into one input of gate G_1 and gate G_2 , making the \bar{Q} output HIGH. Because \bar{Q} is HIGH, both inputs to gate G_3 are HIGH (remember, the output of gate G_1 is HIGH), holding the Q output LOW. If a pulse is applied to the CLK input, the outputs of gates G_1 and G_2 remain HIGH because they are disabled by the LOW on the D input; therefore, there is no change in the state of the flip-flop—it remains in the RESET state.

Let's now make D HIGH and apply a clock pulse. Because the D input to gate G_1 is now HIGH, the output of gate G_1 goes LOW for a very short time (spike) when CLK goes HIGH, causing the Q output to go HIGH. Both inputs to gate G_4 are now HIGH (remember, gate G_2 output is HIGH because D is HIGH), forcing the \bar{Q} output LOW. This LOW on \bar{Q} is coupled back into one input of gate G_3 , ensuring that the Q output will remain HIGH. The flip-flop is now in the SET state. Figure 7-20 illustrates the logic level transitions that take place within the flip-flop for this condition.

Next, let's make D LOW and apply a clock pulse. The positive-going edge of the clock produces a negative-going spike on the output of gate G_2 , causing the \bar{Q} output to go HIGH. Because of this HIGH on \bar{Q} , both inputs to gate G_3 are now HIGH (remember, the output of gate G_1 is HIGH because of the LOW on D), forcing the Q output to go LOW. This LOW on Q is coupled back into one input of gate G_4 , ensuring that \bar{Q} will remain HIGH. The flip-flop is now in the RESET state. Figure 7-21 illustrates the logic level transitions that occur within the flip-flop for this condition.

InfoNote

All logic operations that are performed with hardware can also be implemented in software. For example, the operation of a J-K flip-flop can be performed with specific computer instructions. If two bits were used to represent the J and K inputs, the computer would do nothing for 00, a data bit representing the Q output would be set (1) for 10, the Q data bit would be cleared (0) for 01, and the Q data bit would be complemented for 11. Although it may be unusual to use a computer to simulate a flip-flop, the point is that all hardware operations can be simulated using software.

The Q output of a D flip-flop assumes the state of the D input on the triggering edge of the clock.

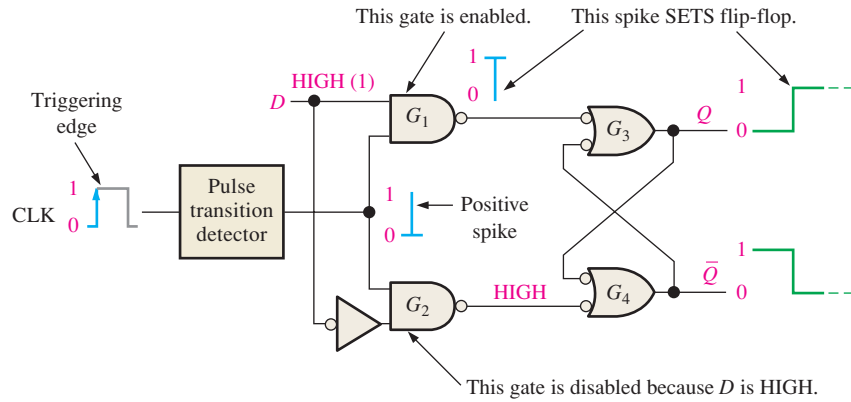


FIGURE 7-20 Flip-flop making a transition from the RESET state to the SET state on the positive-going edge of the clock pulse.

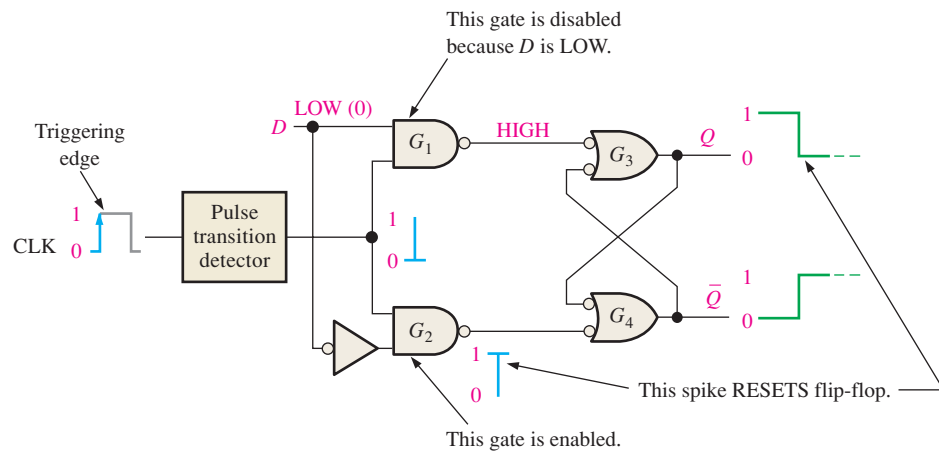


FIGURE 7-21 Flip-flop making a transition from the SET state to the RESET state on the positive-going edge of the clock pulse.

EXAMPLE 7-6

Given the waveforms in Figure 7-22(a) for the D input and the clock, determine the Q output waveform if the flip-flop starts out RESET.

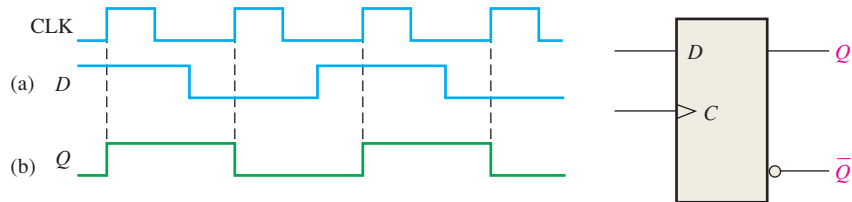


FIGURE 7-22

Solution

The Q output goes to the state of the D input at the time of the positive-going clock edge. The resulting output is shown in Figure 7-22(b).

Related Problem

Determine the Q output for the D flip-flop if the D input in Figure 7-22(a) is inverted.

J-K Flip-Flop

Figure 7–23 shows the basic internal logic for a positive edge-triggered J-K flip-flop. The Q output is connected back to the input of gate G_2 , and the \bar{Q} output is connected back to the input of gate G_1 . The two control inputs are labeled J and K in honor of Jack Kilby, who invented the integrated circuit. A J-K flip-flop can also be of the negative edge-triggered type, in which case the clock input is inverted.

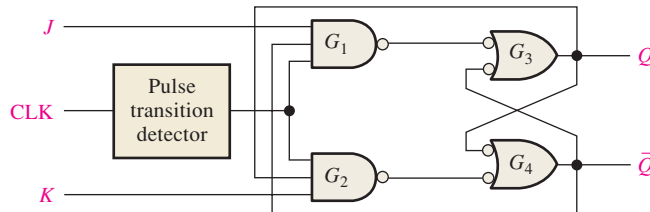


FIGURE 7-23 A simplified logic diagram for a positive edge-triggered J-K flip-flop.

Let's assume that the flip-flop in Figure 7–24 is RESET and that the J input is HIGH and the K input is LOW rather than as shown. When a clock pulse occurs, a leading-edge spike indicated by ① is passed through gate G_1 because \bar{Q} is HIGH and J is HIGH. This will cause the latch portion of the flip-flop to change to the SET state. The flip-flop is now SET.

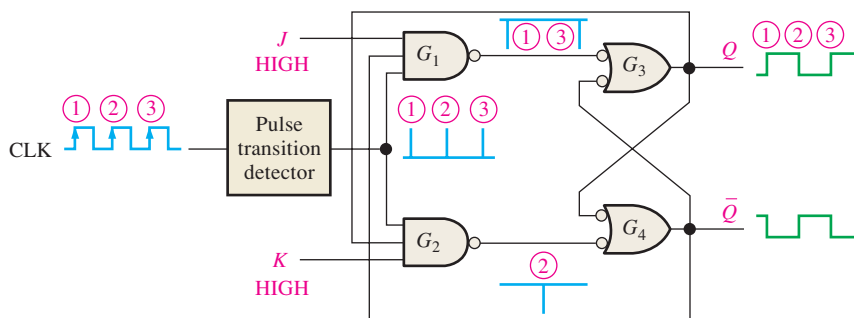


FIGURE 7-24 Transitions illustrating flip-flop operation.

If you make J LOW and K HIGH, the next clock spike indicated by ② will pass through gate G_2 because Q is HIGH and K is HIGH. This will cause the latch portion of the flip-flop to change to the RESET state.

If you apply a LOW to both the J and K inputs, the flip-flop will stay in its present state when a clock pulse occurs. A LOW on both J and K results in a *no-change* condition.

When both the J and K inputs are HIGH and the flip-flop is RESET, the HIGH on the \bar{Q} enables gate G_1 ; so the clock spike indicated by ③ passes through to set the flip-flop. Now there is a HIGH on Q , which allows the next clock spike to pass through gate G_2 and reset the flip-flop.

As you can see, on each successive clock spike, the flip-flop toggles to the opposite state. Figure 7–24 illustrates the transitions when the flip-flop is in the toggle mode. A J-K flip-flop connected for toggle operation is sometimes called a *T flip-flop*.

Asynchronous Preset and Clear Inputs

For the flip-flops just discussed, the D and J - K inputs are called *synchronous inputs* because data on these inputs are transferred to the flip-flop's output only on the triggering edge of the clock pulse; that is, the data are transferred synchronously with the clock.

In the toggle mode, a J-K flip-flop changes state on every clock pulse.

An active preset input makes the Q output HIGH (SET).

An active clear input makes the Q output LOW (RESET).

Most integrated circuit flip-flops also have **asynchronous** inputs. These are inputs that affect the state of the flip-flop *independent of the clock*. They are normally labeled **preset** (\overline{PRE}) and **clear** (\overline{CLR}), or *direct set* (S_D) and *direct reset* (R_D) by some manufacturers. An active level on the preset input will set the flip-flop, and an active level on the clear input will reset it. A logic symbol for a D flip-flop with preset and clear inputs is shown in Figure 7–25. These inputs are active-LOW, as indicated by the bubbles. These preset and clear inputs must both be kept HIGH for synchronous operation. In normal operation, preset and clear would not be LOW at the same time.

Figure 7–26 shows the logic diagram for an edge-triggered D flip-flop with active-LOW preset (\overline{PRE}) and clear (\overline{CLR}) inputs. This figure illustrates basically how these inputs work. As you can see, they are connected so that they override the effect of the synchronous input, D and the clock.

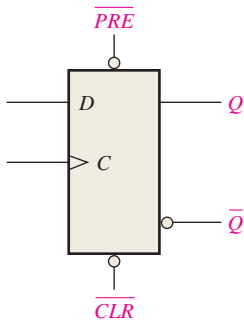


FIGURE 7-25 Logic symbol for a D flip-flop with active-LOW preset and clear inputs.

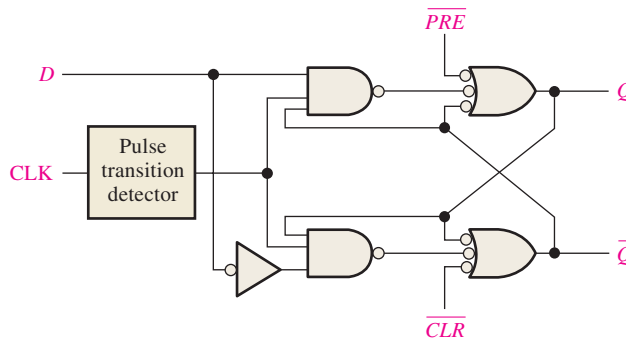


FIGURE 7-26 Logic diagram for a basic D flip-flop with active-LOW preset and clear inputs.

EXAMPLE 7-7

For the positive edge-triggered D flip-flop with preset and clear inputs in Figure 7–27, determine the Q output for the inputs shown in the timing diagram in part (a) if Q is initially LOW.

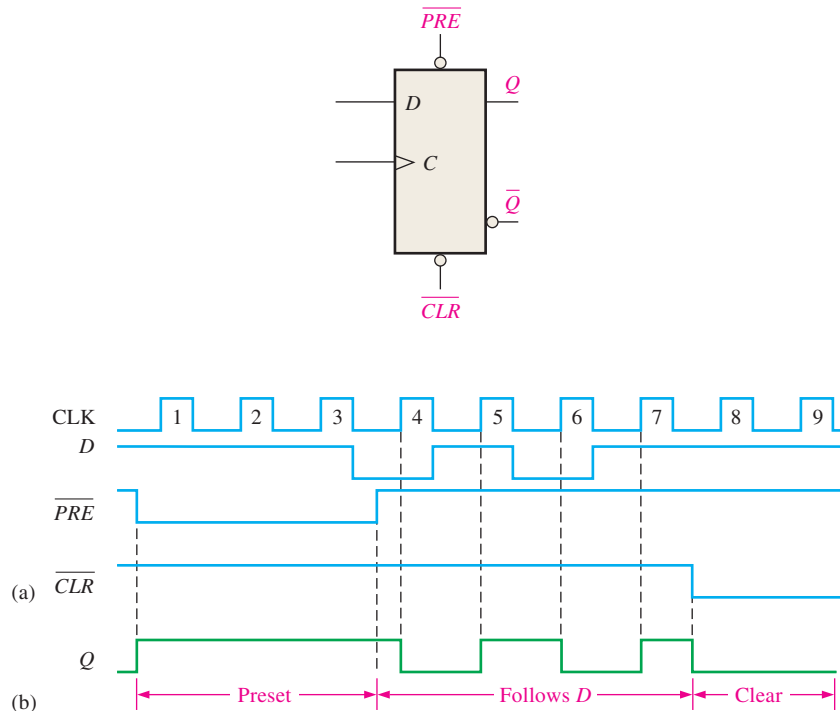


FIGURE 7-27 Open file F07-27 to verify the operation.



Solution

1. During clock pulses 1, 2, and 3, the preset (\overline{PRE}) is LOW, keeping the flip-flop SET regardless of the synchronous D input.
2. For clock pulses 4, 5, 6, and 7, the output follows the input on the clock pulse because both \overline{PRE} and \overline{CLR} are HIGH.
3. For clock pulses 8 and 9, the clear (\overline{CLR}) input is LOW, keeping the flip-flop RESET regardless of the synchronous inputs.

The resulting Q output is shown in Figure 7–27(b).

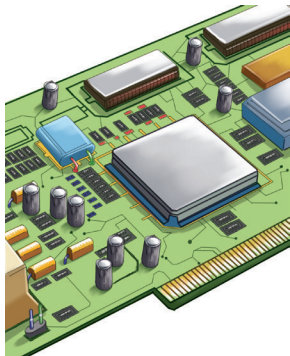
Related Problem

If you interchange the \overline{PRE} and \overline{CLR} waveforms in Figure 7–27(a), what will the Q output look like?

Let’s look at two specific edge-triggered flip-flops. They are representative of the various types of flip-flops available in fixed-function IC form and, like most other devices, are available in CMOS and in bipolar (TTL) logic families.

Also, you will learn how VHDL is used to describe the types of flip-flops.

IMPLEMENTATION: D FLIP-FLOP



Fixed-Function Device The 74HC74 dual D flip-flop contains two identical D flip-flops that are independent of each other except for sharing V_{CC} and ground. The flip-flops are positive edge-triggered and have active-LOW asynchronous preset and clear inputs. The logic symbols for the individual flip-flops within the package are shown in Figure 7–28(a), and an ANSI/IEEE standard single block symbol that represents the entire device is shown in part (b). The pin numbers are shown in parentheses.

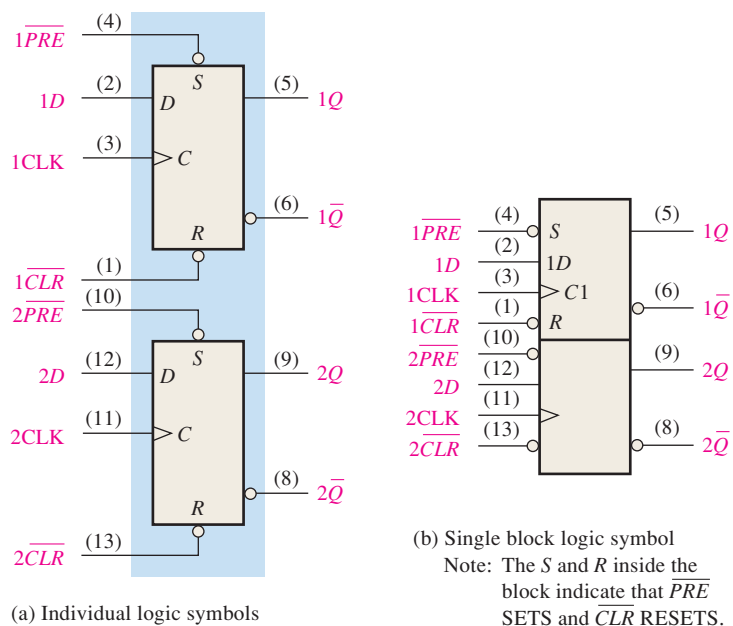


FIGURE 7-28 The 74HC74 dual positive edge-triggered D flip-flop.

Programmable Logic Device (PLD) The positive edge-triggered D flip-flop can be described using VHDL and implemented as hardware in a PLD. In this program, the behavioral approach will be used for the first time because it lends itself to describing sequential operations. A new VHDL statement, **wait until rising edge**, is introduced. This statement allows the program to wait for the rising edge of a clock pulse to process the *D* input to create the desired results. Also the **if then else** statement is introduced. The keyword **process** is a block of code placed between the **begin** and **end** statements of the architecture to allow statements to be sequentially processed. The program code for a single D flip-flop is as follows:



```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity dffl is
    port (D, Clock, Pre, Clr: in std_logic; Q: inout std_logic);
end entity dffl;
```

D: Flip-flop input
Clock: System clock
Pre: Preset input
Clr: Clear input
Q: Flip-flop output

```
architecture LogicOperation of dffl is
begin
    process
    begin
        wait until rising_edge (Clock);
        if Clr = '1' then
            if Pre = '1' then
                if D = '1' then
                    Q <= '1';
                else
                    Q <= '0';
                end if;
            else
                Q <= '1';
            end if;
        else
            Q <= '0';
        end if;
    end process;
end architecture LogicOperation;
```

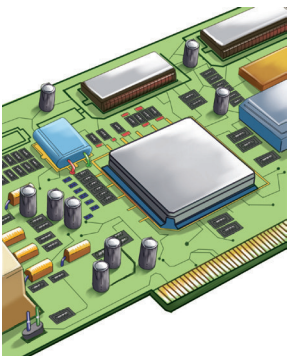
} Check for Preset and Clear conditions

Q input follows D input when Clr and Pre inputs are HIGH.

Q is set HIGH when Pre input is LOW.

Q is set LOW when Clr input is LOW.

IMPLEMENTATION: J-K FLIP-FLOP



Fixed-Function Device The 74HC112 dual J-K flip-flop has two identical flip-flops that are negative edge-triggered and have active-LOW asynchronous preset and clear inputs. The logic symbols are shown in Figure 7–29.

Programmable Logic Device (PLD) The negative edge-triggered J-K flip-flop can be described using VHDL and implemented as hardware in a PLD. In this program, the behavioral approach will be used. A new VHDL statement, **if falling edge then**, is introduced. This statement allows the program to wait for the falling edge of a clock pulse

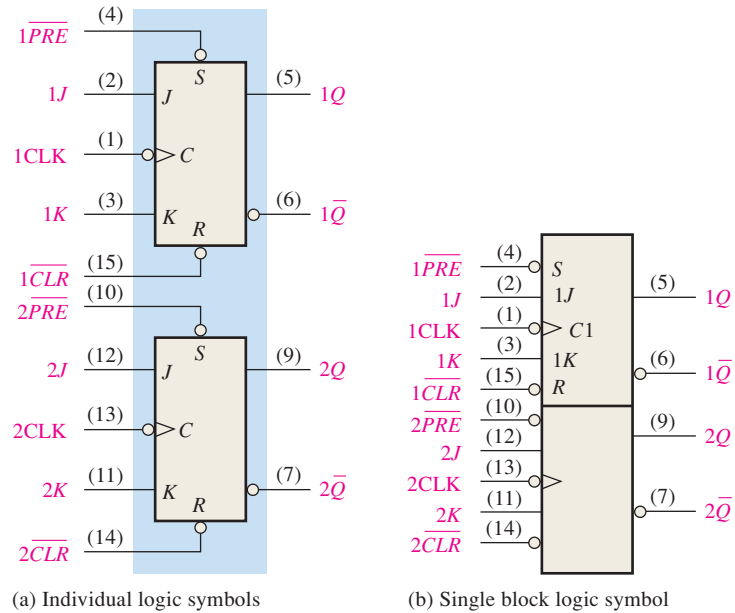


FIGURE 7-29 The 74HC112 dual negative edge-triggered J-K flip-flop.

to process the J and K inputs to create the desired results. The following program code describes a single J-K flip-flop with no preset or clear inputs.



```

library ieee;
use ieee.std_logic_1164.all;
entity JKFlipFlop is
    port (J, K, Clock: in std_logic; Q, QNot: inout std_logic); } Inputs and outputs
end entity JKFlipFlop; } declared

architecture LogicOperation of JKFlipFlop is
    signal J1, K1: std_logic;

begin
    process (J, K, Clock, J1, K1, Q, QNot)
    begin
        if falling_edge(Clock) and Clock = '0' then } Identifies with Boolean expressions
            J1 <= not (J and not Clock and QNot); } the inputs (J1 and K1) to the latch
            K1 <= not (K and not Clock and Q); } portion of the flip-flop
        end if;
            Q <= J1 nand QNot; } Defines the outputs in terms of J1 and
            QNot <= K1 nand Q; } K1 with Boolean expressions
    end process;
end architecture LogicOperation;
    
```

EXAMPLE 7-8

The $1J$, $1K$, $1CLK$, $1\overline{PRE}$, and $1\overline{CLR}$ waveforms in Figure 7-30(a) are applied to one of the negative edge-triggered flip-flops in a 74HC112 package. Determine the $1Q$ output waveform.

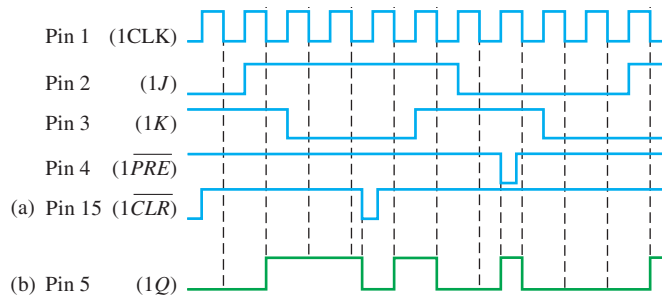


FIGURE 7-30

Solution

The resulting $1Q$ waveform is shown in Figure 7-30(b). Notice that each time a LOW is applied to the $1\overline{PRE}$ or $1\overline{CLR}$, the flip-flop is set or reset regardless of the states of the other inputs.

Related Problem

Determine the $1Q$ output waveform if the waveforms for $1\overline{PRE}$ and $1\overline{CLR}$ are interchanged.

SECTION 7-2 CHECKUP

1. Describe the main difference between a gated D latch and an edge-triggered D flip-flop.
2. How does a J-K flip-flop differ from a D flip-flop in its basic operation?
3. Assume that the flip-flop in Figure 7-22 is negative edge-triggered. Describe the output waveform for the same CLK and D waveforms.

7-3 Flip-Flop Operating Characteristics

The performance, operating requirements, and limitations of flip-flops are specified by several operating characteristics or parameters found on the data sheet for the device. Generally, the specifications are applicable to all CMOS and bipolar (TTL) flip-flops.

After completing this section, you should be able to

- ◆ Define *propagation delay time*
- ◆ Explain the various propagation delay time specifications
- ◆ Define *set-up time* and discuss how it limits flip-flop operation
- ◆ Define *hold time* and discuss how it limits flip-flop operation
- ◆ Discuss the significance of maximum clock frequency
- ◆ Discuss the various pulse width specifications
- ◆ Define *power dissipation* and calculate its value for a specific device
- ◆ Compare various series of flip-flops in terms of their operating parameters

Propagation Delay Times

A **propagation delay time** is the interval of time required after an input signal has been applied for the resulting output change to occur. Four categories of propagation delay times are important in the operation of a flip-flop:

1. Propagation delay t_{PLH} as measured from the triggering edge of the clock pulse to the LOW-to-HIGH transition of the output. This delay is illustrated in Figure 7–31(a).
2. Propagation delay t_{PHL} as measured from the triggering edge of the clock pulse to the HIGH-to-LOW transition of the output. This delay is illustrated in Figure 7–31(b).

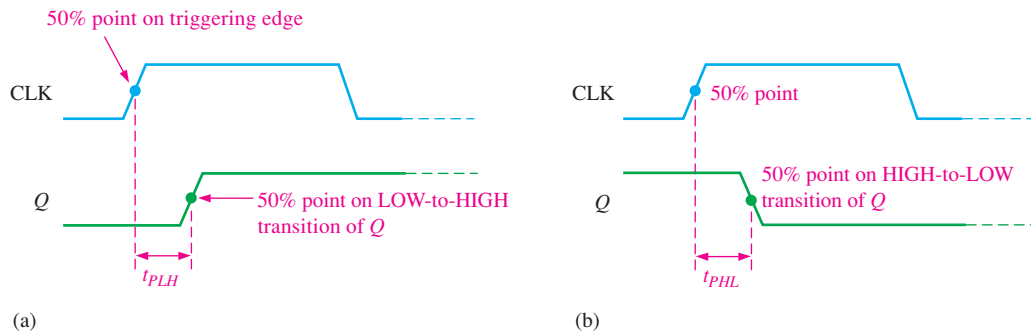


FIGURE 7-31 Propagation delays, clock to output.

3. Propagation delay t_{PLH} as measured from the leading edge of the preset input to the LOW-to-HIGH transition of the output. This delay is illustrated in Figure 7–32(a) for an active-LOW preset input.
4. Propagation delay t_{PHL} as measured from the leading edge of the clear input to the HIGH-to-LOW transition of the output. This delay is illustrated in Figure 7–32(b) for an active-LOW clear input.

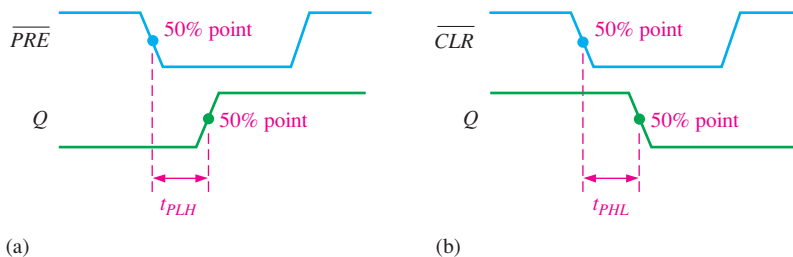


FIGURE 7-32 Propagation delays, preset input to output and clear input to output.

Set-up Time

The **set-up time** (t_s) is the minimum interval required for the logic levels to be maintained constantly on the inputs (J and K , or D) prior to the triggering edge of the clock pulse in order for the levels to be reliably clocked into the flip-flop. This interval is illustrated in Figure 7–33 for a D flip-flop.

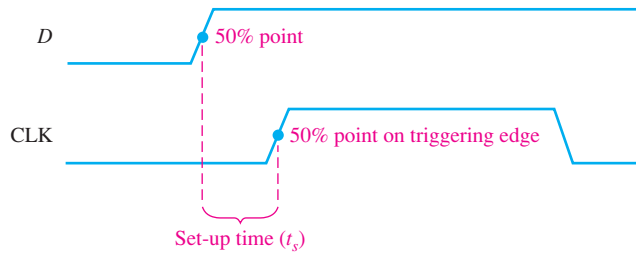


FIGURE 7-33 Set-up time (t_s). The logic level must be present on the D input for a time equal to or greater than t_s before the triggering edge of the clock pulse for reliable data entry.

Hold Time

The **hold time** (t_h) is the minimum interval required for the logic levels to remain on the inputs after the triggering edge of the clock pulse in order for the levels to be reliably clocked into the flip-flop. This is illustrated in Figure 7-34 for a D flip-flop.

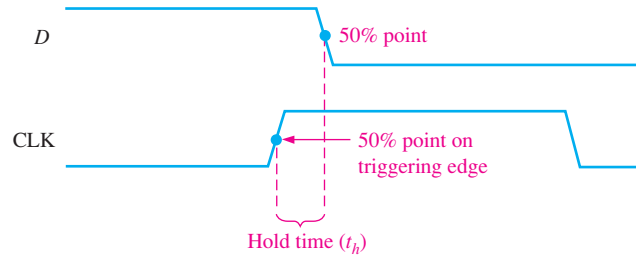


FIGURE 7-34 Hold time (t_h). The logic level must remain on the D input for a time equal to or greater than t_h after the triggering edge of the clock pulse for reliable data entry.

Maximum Clock Frequency

The maximum clock frequency (f_{max}) is the highest rate at which a flip-flop can be reliably triggered. At clock frequencies above the maximum, the flip-flop would be unable to respond quickly enough, and its operation would be impaired.

Pulse Widths

Minimum pulse widths (t_w) for reliable operation are usually specified by the manufacturer for the clock, preset, and clear inputs. Typically, the clock is specified by its minimum HIGH time and its minimum LOW time.

Power Dissipation

The **power dissipation** of any digital circuit is the total power consumption of the device. For example, if the flip-flop operates on a +5 V dc source and draws 5 mA of current, the power dissipation is

$$P = V_{CC} \times I_{CC} = 5 \text{ V} \times 5 \text{ mA} = 25 \text{ mW}$$

The power dissipation is very important in most applications in which the capacity of the dc supply is a concern. As an example, let's assume that you have a digital system that requires a total of ten flip-flops, and each flip-flop dissipates 25 mW of power. The total power requirement is

$$P_T = 10 \times 25 \text{ mW} = 250 \text{ mW} = 0.25 \text{ W}$$



An advantage of CMOS is that it can operate over a wider range of dc supply voltages (typically 2 V to 6 V) than bipolar and, therefore, less expensive power supplies that do not have precise regulation can be used. Also, batteries can be used as secondary or primary sources for CMOS circuits. In addition, lower voltages mean that the IC dissipates less power. The drawback is that the performance of CMOS is degraded with lower supply voltages. For example, the guaranteed maximum clock frequency of a CMOS flip-flop is much less at $V_{CC} = 2\text{ V}$ than at $V_{CC} = 6\text{ V}$.

This tells you the output capacity required of the dc supply. If the flip-flops operate on +5 V dc, then the amount of current that the supply must provide is

$$I = \frac{250\text{ mW}}{5\text{ V}} = 50\text{ mA}$$

You must use a +5 V dc supply that is capable of providing at least 50 mA of current.

Comparison of Specific Flip-Flops

Table 7–4 provides a comparison, in terms of the operating parameters discussed in this section, of four CMOS and bipolar (TTL) flip-flops of the same type but with different IC families (HC, AHC, LS, and F).

TABLE 7–4

Comparison of operating parameters for four IC families of flip-flops of the same type at 25°C.

Parameter	CMOS		Bipolar (TTL)	
	74HC74A	74AHC74	74LS74A	74F74
t_{PHL} (CLK to Q)	17 ns	4.6 ns	40 ns	6.8 ns
t_{PLH} (CLK to Q)	17 ns	4.6 ns	25 ns	8.0 ns
t_{PHL} (\overline{CLR} to Q)	18 ns	4.8 ns	40 ns	9.0 ns
t_{PLH} (\overline{PRE} to Q)	18 ns	4.8 ns	25 ns	6.1 ns
t_s (set-up time)	14 ns	5.0 ns	20 ns	2.0 ns
t_h (hold time)	3.0 ns	0.5 ns	5 ns	1.0 ns
t_W (CLK HIGH)	10 ns	5.0 ns	25 ns	4.0 ns
t_W (CLK LOW)	10 ns	5.0 ns	25 ns	5.0 ns
t_W ($\overline{CLR}/\overline{PRE}$)	10 ns	5.0 ns	25 ns	4.0 ns
f_{max}	35 MHz	170 MHz	25 MHz	100 MHz
Power, quiescent	0.012 mW	1.1 mW		
Power, 50% duty cycle			44 mW	88 mW

SECTION 7–3 CHECKUP

- Define the following:
 - set-up time
 - hold time
- Which specific flip-flop in Table 7–4 can be operated at the highest frequency?

7–4 Flip-Flop Applications

In this section, three general applications of flip-flops are discussed to give you an idea of how they can be used. In Chapters 8 and 9, flip-flop applications in registers and counters are covered in detail.

After completing this section, you should be able to

- ◆ Discuss the application of flip-flops in data storage
- ◆ Describe how flip-flops are used for frequency division
- ◆ Explain how flip-flops are used in basic counter applications

Parallel Data Storage

A common requirement in digital systems is to store several bits of data from parallel lines simultaneously in a group of flip-flops. This operation is illustrated in Figure 7–35(a) using four flip-flops. Each of the four parallel data lines is connected to the D input of a flip-flop. The clock inputs of the flip-flops are connected together, so that each flip-flop is triggered by the same clock pulse. In this example, positive edge-triggered flip-flops are used, so the data on the D inputs are stored simultaneously by the flip-flops on the positive edge of the clock, as indicated in the timing diagram in Figure 7–35(b). Also, the asynchronous reset (R) inputs are connected to a common \overline{CLR} line, which initially resets all the flip-flops.

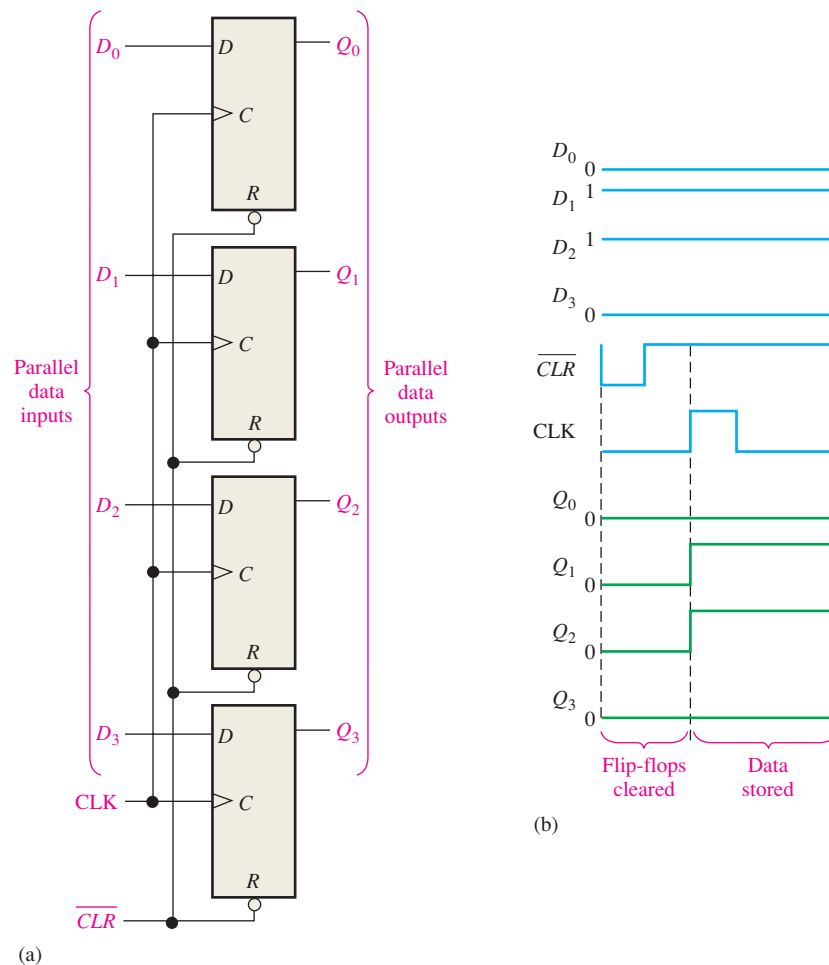


FIGURE 7-35 Example of flip-flops used in a basic register for parallel data storage.

This group of four flip-flops is an example of a basic register used for data storage. In digital systems, data are normally stored in groups of bits (usually eight or multiples thereof) that represent numbers, codes, or other information. Registers are covered in Chapter 8.

Frequency Division

Another application of a flip-flop is dividing (reducing) the frequency of a periodic waveform. When a pulse waveform is applied to the clock input of a D or J-K flip-flop that is connected to toggle ($D = \bar{Q}$ or $J = K = 1$), the Q output is a square wave with one-half the frequency of the clock input. Thus, a single flip-flop can be applied as a divide-by-2 device, as is illustrated in Figure 7-36 for both a D and a J-K flip-flop. As you can see in part (c), the flip-flop changes state on each triggering clock edge (positive edge-triggered in this case). This results in an output that changes at half the frequency of the clock waveform.

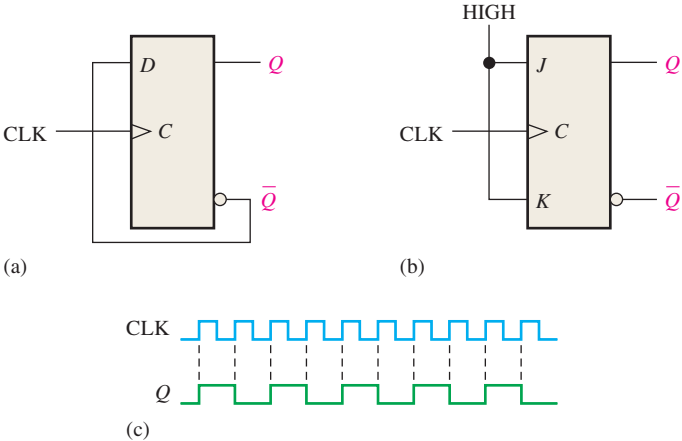


FIGURE 7-36 The D flip-flop and J-K flip-flop as a divide-by-2 device. Q is one-half the frequency of CLK. Open file F07-36 and verify the operation.



Further division of a clock frequency can be achieved by using the output of one flip-flop as the clock input to a second flip-flop, as shown in Figure 7-37. The frequency of the Q_A output is divided by 2 by flip-flop B. The Q_B output is, therefore, one-fourth the frequency of the original clock input. Propagation delay times are not shown on the timing diagrams.

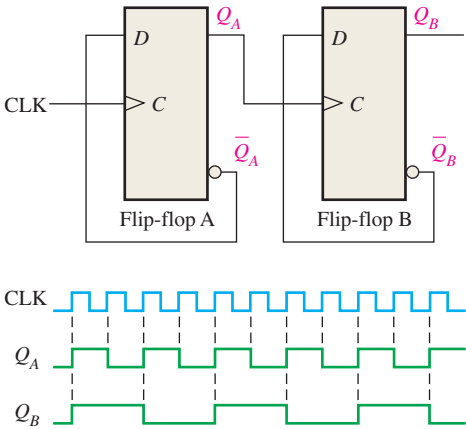


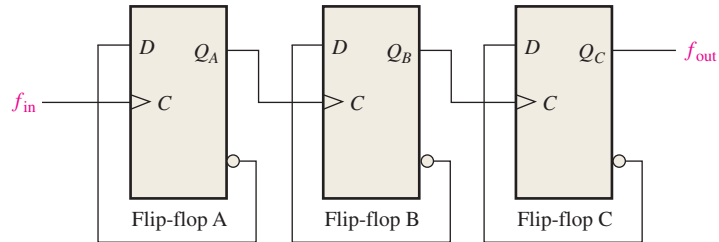
FIGURE 7-37 Example of two D flip-flops used to divide the clock frequency by 4. Q_A is one-half and Q_B is one-fourth the frequency of CLK. Open file F07-37 and verify the operation.



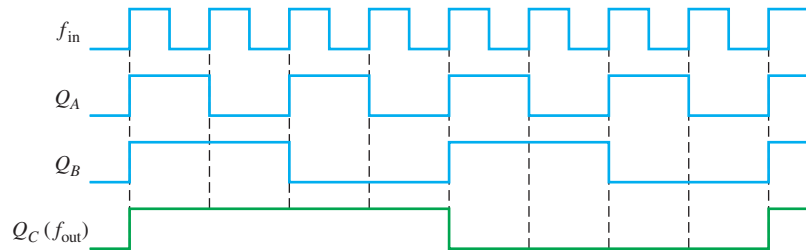
By connecting flip-flops in this way, a frequency division of 2^n is achieved, where n is the number of flip-flops. For example, three flip-flops divide the clock frequency by $2^3 = 8$; four flip-flops divide the clock frequency by $2^4 = 16$; and so on.

EXAMPLE 7-9

Develop the f_{out} waveform for the circuit in Figure 7-38 when an 8 kHz square wave input is applied to the clock input of flip-flop A.

**FIGURE 7-38****Solution**

The three flip-flops are connected to divide the input frequency by eight ($2^3 = 8$) and the Q_C (f_{out}) waveform is shown in Figure 7-39. Since these are positive edge-triggered flip-flops, the outputs change on the positive-going clock edge. There is one output pulse for every eight input pulses, so the output frequency is 1 kHz. Waveforms of Q_A and Q_B are also shown.

**FIGURE 7-39****Related Problem**

How many flip-flops are required to divide a frequency by thirty-two?

Counting

Another important application of flip-flops is in digital counters, which are covered in detail in Chapter 9. The concept is illustrated in Figure 7-40. Negative edge-triggered J-K flip-flops are used for illustration. Both flip-flops are initially RESET. Flip-flop A toggles on the negative-going transition of each clock pulse. The Q output of flip-flop A clocks flip-flop B, so each time Q_A makes a HIGH-to-LOW transition, flip-flop B toggles. The resulting Q_A and Q_B waveforms are shown in the figure.

Observe the sequence of Q_A and Q_B in Figure 7-40. Prior to clock pulse 1, $Q_A = 0$ and $Q_B = 0$; after clock pulse 1, $Q_A = 1$ and $Q_B = 0$; after clock pulse 2, $Q_A = 0$ and $Q_B = 1$; and after clock pulse 3, $Q_A = 1$ and $Q_B = 1$. If we take Q_A as the least significant bit, a 2-bit sequence is produced as the flip-flops are clocked. This binary sequence repeats every four clock pulses, as shown in the timing diagram of Figure 7-40. Thus, the flip-flops are counting in sequence from 0 to 3 (00, 01, 10, 11) and then recycling back to 0 to begin the sequence again.

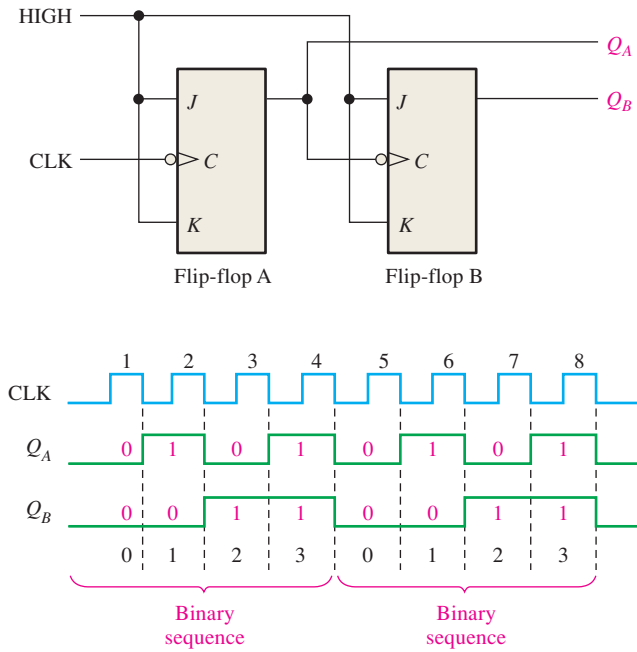
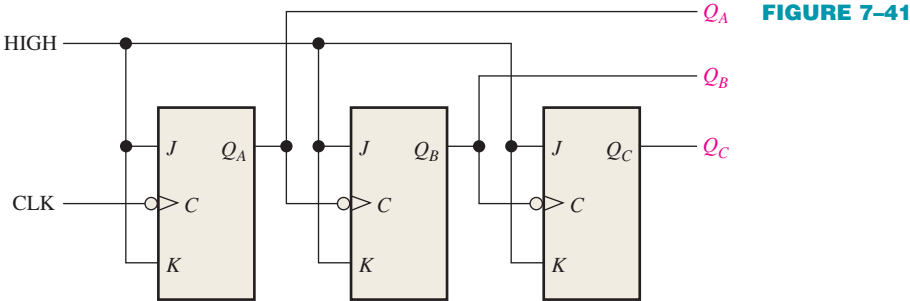


FIGURE 7-40 J-K flip-flops used to generate a binary count sequence (00, 01, 10, 11). Two repetitions are shown.

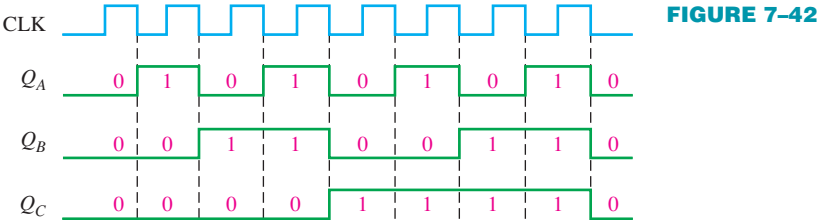
EXAMPLE 7-10

Determine the output waveforms in relation to the clock for Q_A , Q_B , and Q_C in the circuit of Figure 7-41 and show the binary sequence represented by these waveforms.



Solution

The output timing diagram is shown in Figure 7-42. Notice that the outputs change on the negative-going edge of the clock pulses. The outputs go through the binary sequence 000, 001, 010, 011, 100, 101, 110, and 111 as indicated.



Related Problem

How many flip-flops are required to produce a binary sequence representing decimal numbers 0 through 15?

SECTION 7-4 CHECKUP

1. What is a group of flip-flops used for data storage called?
2. How must a D flip-flop be connected to function as a divide-by-2 device?
3. How many flip-flops are required to produce a divide-by-64 device?

7-5 One-Shots

The **one-shot**, also known as a **monostable** multivibrator, is a device with only one stable state. A one-shot is normally in its stable state and will change to its unstable state only when triggered. Once it is triggered, the one-shot remains in its unstable state for a predetermined length of time and then automatically returns to its stable state. The time that the device stays in its unstable state determines the pulse width of its output.

After completing this section, you should be able to

- ◆ Describe the basic operation of a one-shot
- ◆ Explain how a nonretriggerable one-shot works
- ◆ Explain how a retriggerable one-shot works
- ◆ Set up the 74121 and the 74LS122 one-shots to obtain a specified output pulse width
- ◆ Recognize a Schmitt trigger symbol and explain basically what it means
- ◆ Describe the basic elements of a 555 timer
- ◆ Set up a 555 timer as a one-shot

A one-shot produces a single pulse each time it is triggered.

Figure 7-43 shows a basic one-shot (monostable multivibrator) that is composed of a logic gate and an inverter. When a pulse is applied to the **trigger** input, the output of gate G_1 goes LOW. This HIGH-to-LOW transition is coupled through the capacitor to the input of inverter G_2 . The apparent LOW on G_2 makes its output go HIGH. This HIGH is connected back into G_1 , keeping its output LOW. Up to this point the trigger pulse has caused the output of the one-shot, Q , to go HIGH.

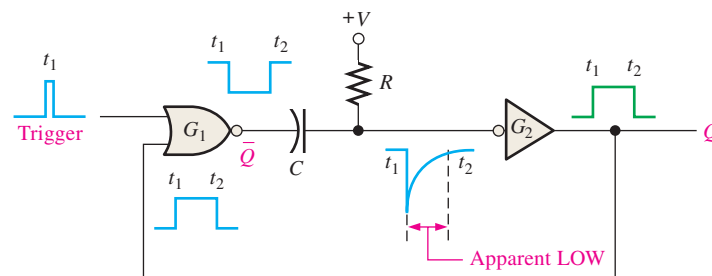


FIGURE 7-43 A simple one-shot circuit.

The capacitor immediately begins to charge through R toward the high voltage level. The rate at which it charges is determined by the RC time constant. When the capacitor charges to a certain level, which appears as a HIGH to G_2 , the output goes back LOW.

To summarize, the output of inverter G_2 goes HIGH in response to the trigger input. It remains HIGH for a time set by the RC time constant. At the end of this time, it goes LOW. A single narrow trigger pulse produces a single output pulse whose time duration is controlled by the RC time constant. This operation is illustrated in Figure 7-43.

A typical one-shot logic symbol is shown in Figure 7-44(a), and the same symbol with an external R and C is shown in Figure 7-44(b). The two basic types of IC one-shots are nonretriggerable and retriggerable.

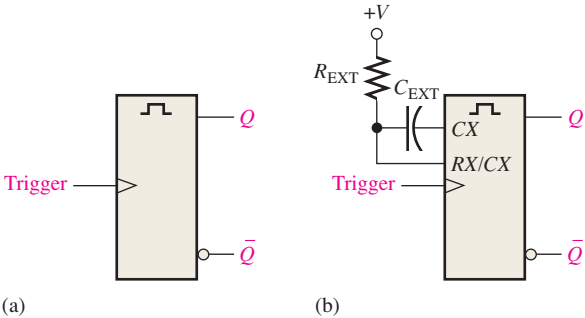


FIGURE 7-44 Basic one-shot logic symbols. CX and RX stand for external components.

A nonretriggerable one-shot will not respond to any additional trigger pulses from the time it is triggered into its unstable state until it returns to its stable state. In other words, it will ignore any trigger pulses occurring before it times out. The time that the one-shot remains in its unstable state is the pulse width of the output.

Figure 7-45 shows the nonretriggerable one-shot being triggered at intervals greater than its pulse width and at intervals less than the pulse width. Notice that in the second case, the additional pulses are ignored.

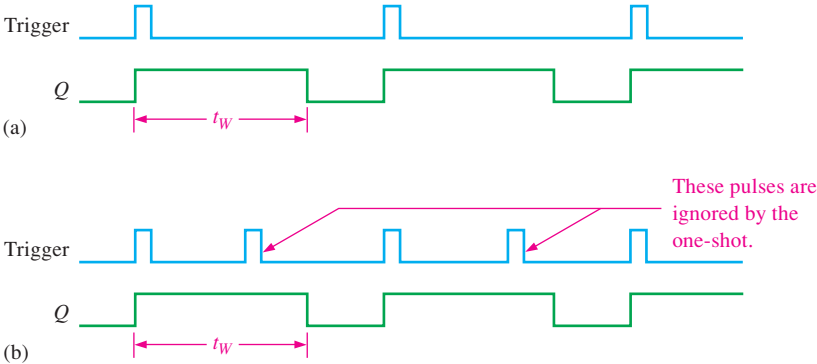


FIGURE 7-45 Nonretriggerable one-shot action.

A retriggerable one-shot can be triggered before it times out. The result of retriggering is an extension of the pulse width as illustrated in Figure 7-46.

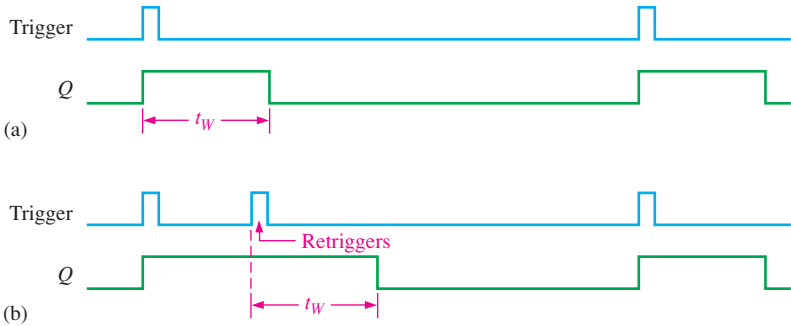


FIGURE 7-46 Retriggerable one-shot action.

Nonretriggerable One-Shot

The 74121 is an example of a nonretriggerable IC one-shot. It has provisions for external R and C , as shown in Figure 7–47. The inputs labeled A_1 , A_2 , and B are gated trigger inputs. The R_{INT} input connects to a 2 k Ω internal timing resistor.

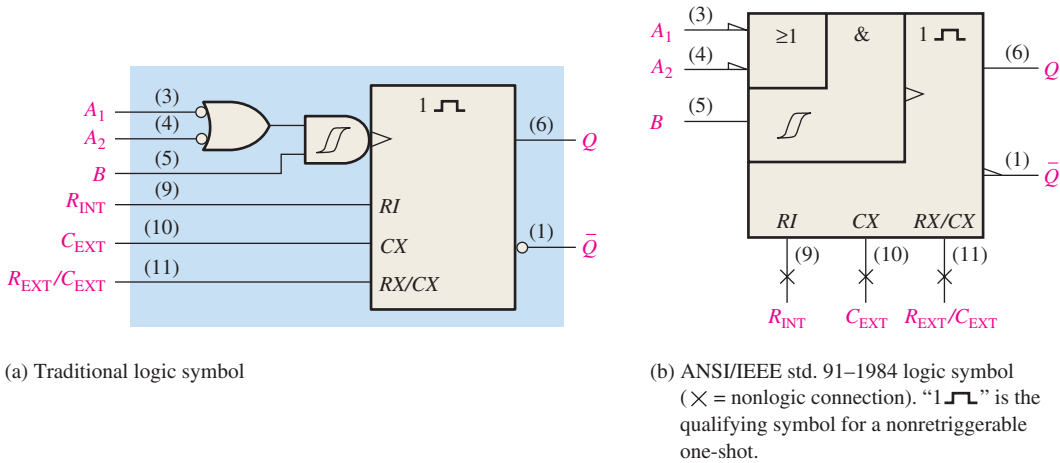


FIGURE 7-47 Logic symbols for the 74121 nonretriggerable one-shot.

Setting the Pulse Width

A typical pulse width of about 30 ns is produced when no external timing components are used and the internal timing resistor (R_{INT}) is connected to V_{CC} , as shown in Figure 7–48(a). The pulse width can be set anywhere between about 30 ns and 28 s by the use of external components. Figure 7–48(b) shows the configuration using the internal resistor (2 k Ω) and an external capacitor. Part (c) shows the configuration using an external resistor and an external capacitor. The output pulse width is set by the values of the resistor ($R_{INT} = 2 \text{ k}\Omega$, and R_{EXT} is selected) and the capacitor according to the following formula:

$$t_W = 0.7RC_{EXT} \quad \text{Equation 7-1}$$

where R is either R_{INT} or R_{EXT} . When R is in kilohms (k Ω) and C_{EXT} is in picofarads (pF), the output pulse width t_W is in nanoseconds (ns).

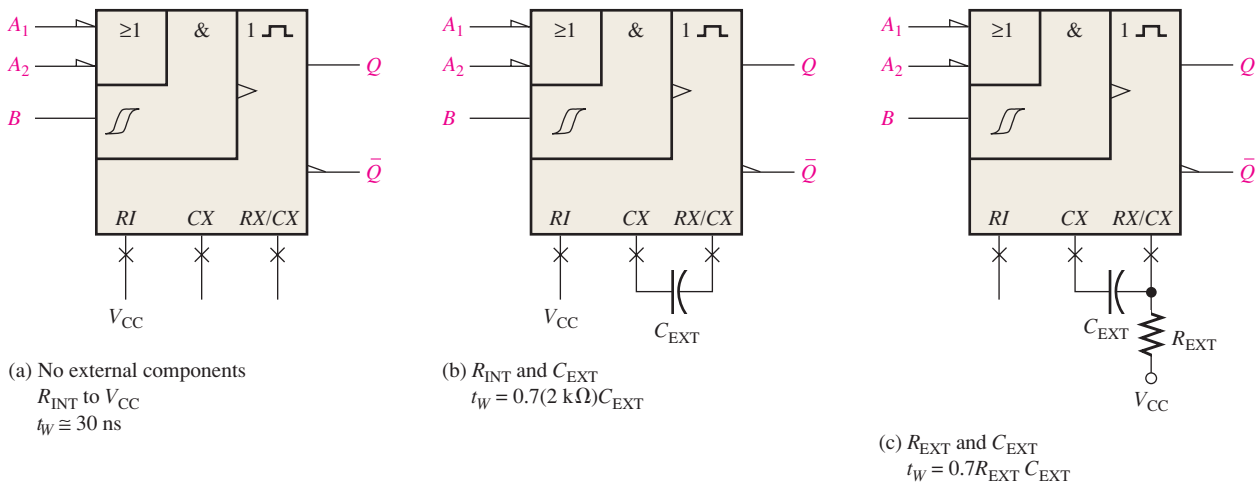


FIGURE 7-48 Three ways to set the pulse width of a 74121.

The Schmitt-Trigger Symbol

The symbol \int indicates a Schmitt-trigger input. This type of input uses a special threshold circuit that produces **hysteresis**, a characteristic that prevents erratic switching between states when a slow-changing trigger voltage hovers around the critical input level. This allows reliable triggering to occur even when the input is changing as slowly as 1 volt/second.

Retriggerable One-Shot

The 74LS122 is an example of a retriggerable IC one-shot with a clear input. It also has provisions for external R and C , as shown in Figure 7–49. The inputs labeled A_1 , A_2 , B_1 , and B_2 are the gated trigger inputs.

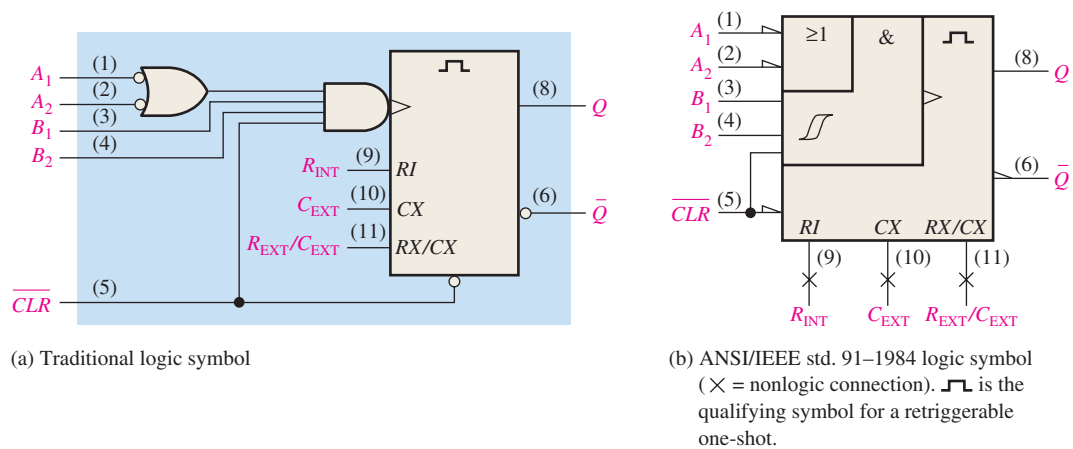


FIGURE 7-49 Logic symbol for the 74LS122 retriggerable one-shot.

A minimum pulse width of approximately 45 ns is obtained with no external components. Wider pulse widths are achieved by using external components. A general formula for calculating the values of these components for a specified pulse width (t_w) is

$$t_w = 0.32RC_{\text{EXT}} \left(1 + \frac{0.7}{R} \right) \quad \text{Equation 7-2}$$

where 0.32 is a constant determined by the particular type of one-shot, R is in $k\Omega$ and is either the internal or the external resistor, C_{EXT} is in pF, and t_w is in ns. The internal resistance is 10 $k\Omega$ and can be used instead of an external resistor. (Notice the difference between this formula and that for the 74121, shown in Equation 7-1.)

EXAMPLE 7-11

A certain application requires a one-shot with a pulse width of approximately 100 ms. Using a 74121, show the connections and the component values.

Solution

Arbitrarily select $R_{\text{EXT}} = 39 \text{ k}\Omega$ and calculate the necessary capacitance.

$$t_w = 0.7R_{\text{EXT}}C_{\text{EXT}}$$

$$C_{\text{EXT}} = \frac{t_w}{0.7R_{\text{EXT}}}$$

where C_{EXT} is in pF, R_{EXT} is in $k\Omega$, and t_w is in ns. Since $100 \text{ ms} = 1 \times 10^8 \text{ ns}$,

$$C_{\text{EXT}} = \frac{1 \times 10^8 \text{ ns}}{0.7(39 \text{ k}\Omega)} = 3.66 \times 10^{-6} \text{ pF} = \mathbf{3.66 \mu\text{F}}$$

A standard $3.3\ \mu\text{F}$ capacitor will give an output pulse width of 91 ms. The proper connections are shown in Figure 7-50. To achieve a pulse width closer to 100 ms, other combinations of values for R_{EXT} and C_{EXT} can be tried. For example, $R_{\text{EXT}} = 68\ \text{k}\Omega$ and $C_{\text{EXT}} = 2.2\ \mu\text{F}$ gives a pulse width of 105 ms.

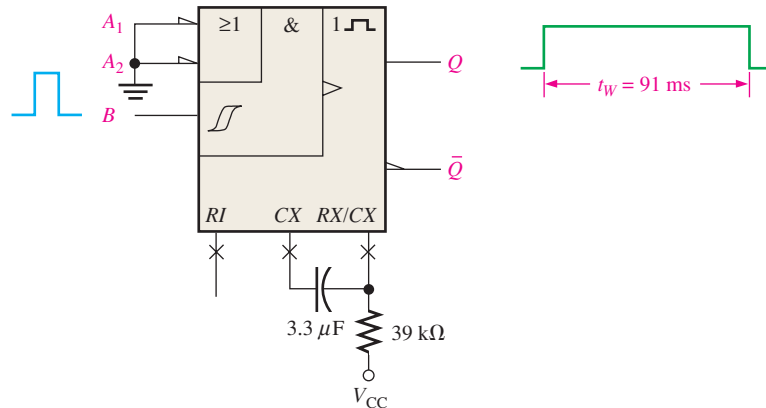


FIGURE 7-50

Related Problem

Use an external capacitor in conjunction with R_{INT} to produce an output pulse width of $10\ \mu\text{s}$ from the 74121.

EXAMPLE 7-12

Determine the values of R_{EXT} and C_{EXT} that will produce a pulse width of $1\ \mu\text{s}$ when connected to a 74LS122.

Solution

Assume a value of $C_{\text{EXT}} = 560\ \text{pF}$ and then solve for R_{EXT} . The pulse width must be expressed in ns and C_{EXT} in pF. R_{EXT} will be in $\text{k}\Omega$.

$$\begin{aligned} t_w &= 0.32R_{\text{EXT}}C_{\text{EXT}} \left(1 + \frac{0.7}{R_{\text{EXT}}} \right) = 0.32R_{\text{EXT}}C_{\text{EXT}} + 0.7 \left(\frac{0.32R_{\text{EXT}}C_{\text{EXT}}}{R_{\text{EXT}}} \right) \\ &= 0.32R_{\text{EXT}}C_{\text{EXT}} + (0.7)(0.32)C_{\text{EXT}} \\ R_{\text{EXT}} &= \frac{t_w - (0.7)(0.32)C_{\text{EXT}}}{0.32C_{\text{EXT}}} = \frac{t_w}{0.32C_{\text{EXT}}} - 0.7 \\ &= \frac{1000\ \text{ns}}{(0.32)560\ \text{pF}} - 0.7 = \mathbf{4.88\ \text{k}\Omega} \end{aligned}$$

Use a standard value of **4.7 kΩ**.

Related Problem

Show the connections and component values for a 74LS122 one-shot with an output pulse width of $5\ \mu\text{s}$. Assume $C_{\text{EXT}} = 560\ \text{pF}$.

An Application

One practical one-shot application is a sequential timer that can be used to illuminate a series of lights. This type of circuit can be used, for example, in a lane change directional indicator for highway construction projects or in sequential turn signals on automobiles.

Figure 7–51 shows three 74LS122 one-shots connected as a sequential timer. This particular circuit produces a sequence of three 1 s pulses. The first one-shot is triggered by a switch closure or a low-frequency pulse input, producing a 1 s output pulse. When the first one-shot (OS 1) times out and the 1 s pulse goes LOW, the second one-shot (OS 2) is triggered, also producing a 1 s output pulse. When this second pulse goes LOW, the third one-shot (OS 3) is triggered and the third 1 s pulse is produced. The output timing is illustrated in the figure. Variations of this basic arrangement can be used to produce a variety of timed outputs.

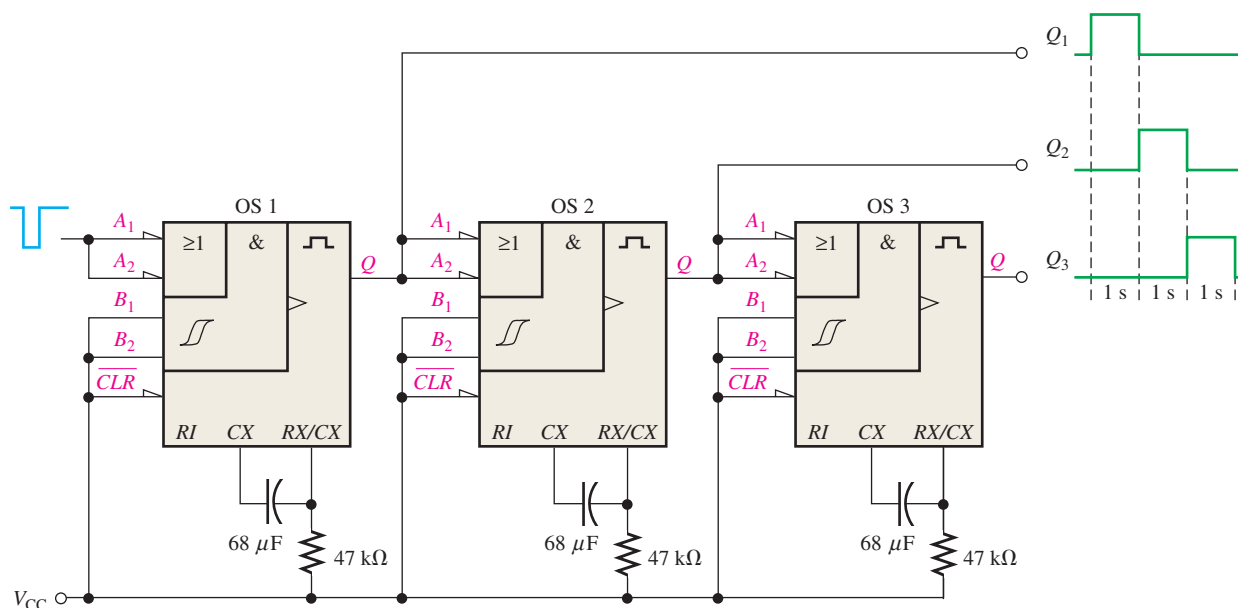


FIGURE 7-51 A sequential timing circuit using three 74LS122 one-shots.

The 555 Timer as a One-Shot

The 555 **timer** is a versatile and widely used IC device because it can be configured in two different modes as either a monostable multivibrator (one-shot) or as an astable multivibrator (pulse oscillator). The astable multivibrator is discussed in Section 7–6.

The 555 Timer Operation

A functional diagram showing the internal components of a 555 timer is shown in Figure 7–52. The comparators are devices whose outputs are HIGH when the voltage on the positive (+) input is greater than the voltage on the negative (–) input and LOW when the – input voltage is greater than the + input voltage. The voltage divider consisting of three 5 kΩ resistors provides a trigger level of $\frac{1}{3} V_{CC}$ and a threshold level of $\frac{2}{3} V_{CC}$. The control voltage input (pin 5) can be used to externally adjust the trigger and threshold levels to other values if necessary. When the normally HIGH trigger input momentarily goes below $\frac{1}{3} V_{CC}$, the output of comparator B switches from LOW to HIGH and sets the S-R latch, causing the output (pin 3) to go HIGH and turning the discharge transistor Q_1 off. The output will stay HIGH until the normally LOW threshold input goes above $\frac{2}{3} V_{CC}$ and causes the output of comparator A to switch from LOW to HIGH. This resets the latch, causing the output to go back LOW and turning the discharge transistor on. The external reset input can be used to reset the latch independent of the threshold circuit. The trigger and threshold inputs (pins 2 and 6) are controlled by external components connected to produce either monostable or astable action.

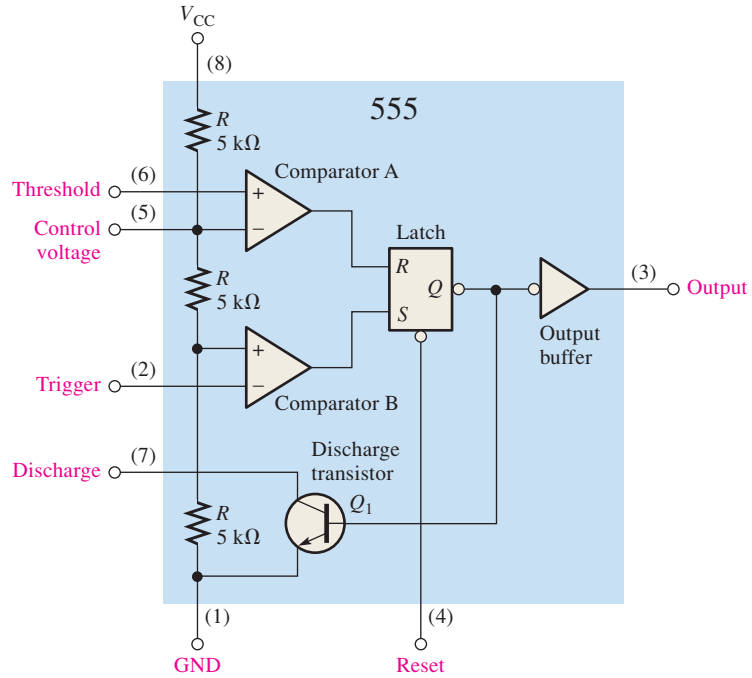


FIGURE 7-52 Internal functional diagram of a 555 timer (pin numbers are in parentheses).

Monostable (One-Shot) Operation

An external resistor and capacitor connected as shown in Figure 7-53 are used to set up the 555 timer as a nonretriggerable one-shot. The pulse width of the output is determined by the time constant of R_1 and C_1 according to the following formula:

$$t_w = 1.1R_1C_1 \quad \text{Equation 7-3}$$

The control voltage input is not used and is connected to a decoupling capacitor C_2 to prevent noise from affecting the trigger and threshold levels.

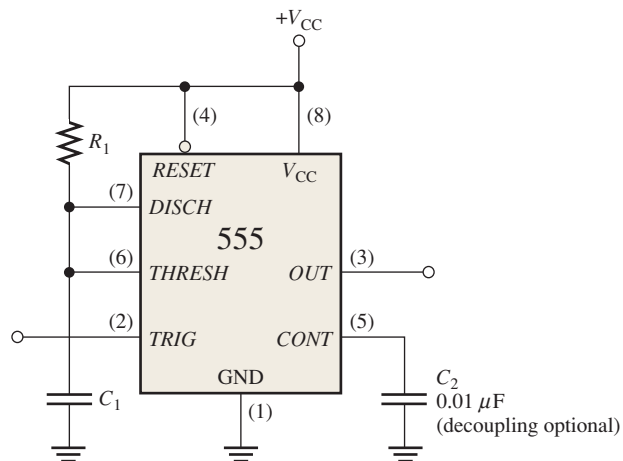


FIGURE 7-53 The 555 timer connected as a one-shot.

Before a trigger pulse is applied, the output is LOW and the discharge transistor Q_1 is *on*, keeping C_1 discharged as shown in Figure 7-54(a). When a negative-going trigger pulse is applied at t_0 , the output goes HIGH and the discharge transistor turns *off*, allowing capacitor C_1 to begin charging through R_1 as shown in part (b). When C_1 charges to $\frac{1}{3} V_{CC}$,

the output goes back LOW at t_1 and Q_1 turns *on* immediately, discharging C_1 as shown in part (c). As you can see, the charging rate of C_1 determines how long the output is HIGH.

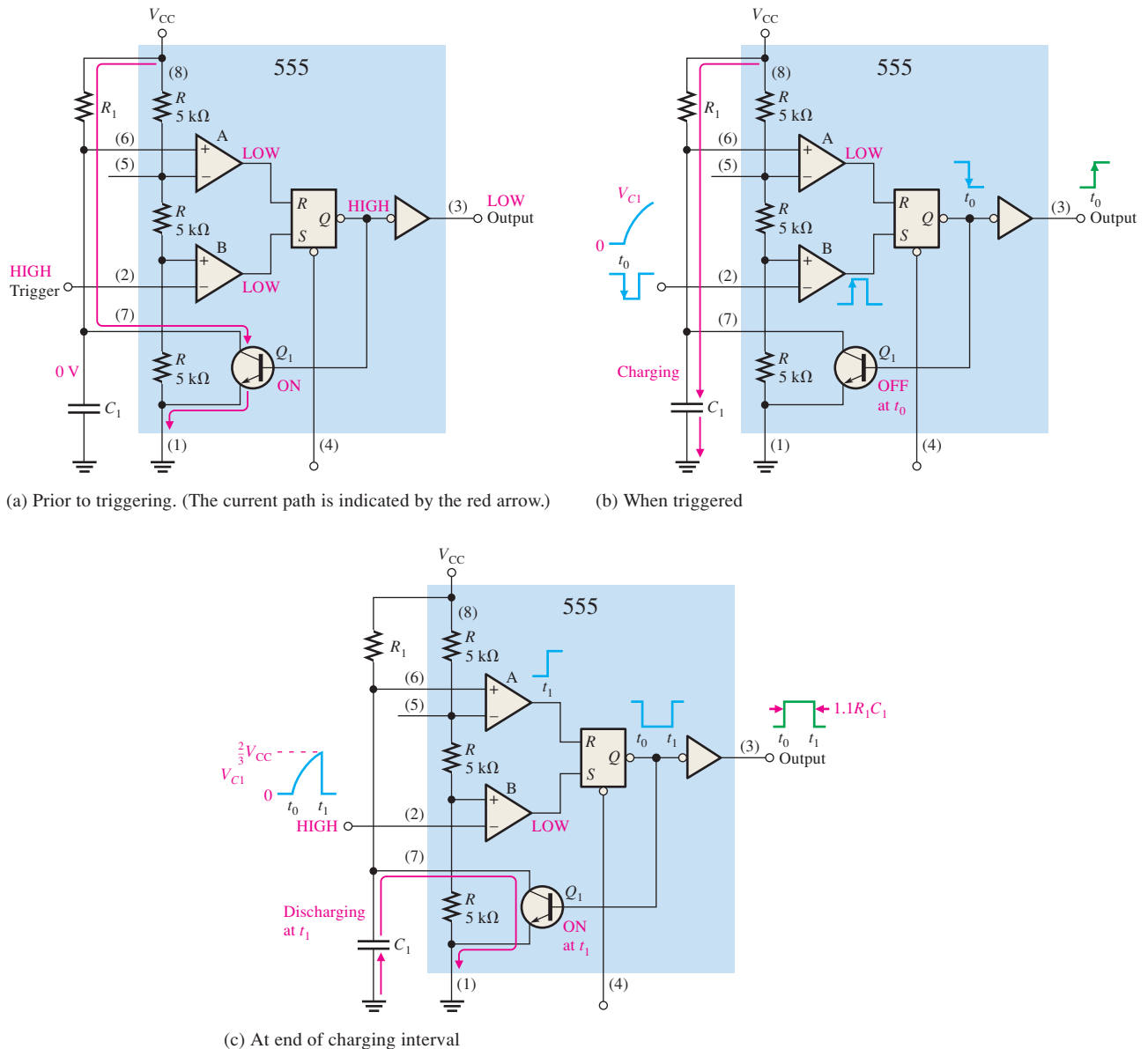


FIGURE 7-54 One-shot operation of the 555 timer.

EXAMPLE 7-13

What is the output pulse width for a 555 monostable circuit with $R_1 = 2.2 \text{ k}\Omega$ and $C_1 = 0.01 \text{ }\mu\text{F}$?

Solution

From Equation 7-3 the pulse width is

$$t_W = 1.1R_1C_1 = 1.1(2.2 \text{ k}\Omega)(0.01 \text{ }\mu\text{F}) = \mathbf{24.2 \text{ }\mu\text{s}}$$

Related Problem

For $C_1 = 0.01 \text{ }\mu\text{F}$, determine the value of R_1 for a pulse width of 1 ms.

One-Shot with VHDL

An example of a VHDL program code for a one-shot is as follows:



```

library ieee;
use ieee.std_logic_1164.all;

entity OneShot is
  port (Enable, Clk: in std_logic;
        Duration: in integer range 0 to 25;
        QOut: buffer std_logic);
end entity OneShot;

architecture OneShotBehavior of OneShot is
begin
  Counter: process (Enable, Clk, Duration)
  variable Flag      : boolean := true;
  variable Cnt      : integer range 0 to 25;
  variable SetCount : integer range 0 to 25;
  begin
    SetCount := Duration;
    if (Clk'EVENT and Clk = '1') then
      if Enable = '0' then
        Flag := true;
      end if;

      if Enable = '1' and Flag then
        Cnt := 1;
        Flag :=False;
      end if;

      if cnt = SetCount then
        Qout <= '0';
        Cnt := 0;
        Flag := false;
      else
        if Cnt > 0 then
          Cnt := Cnt + 1;
          Qout <= '1';
        end if;
      end if;
    end if;
  end process;
end architecture OneShotBehavior;

```



In normal operation, a one-shot produces only a single pulse, which can be difficult to measure on an oscilloscope because the pulse does not occur regularly. To obtain a stable display for test purposes, it is useful to trigger the one-shot from a pulse generator that is set to a longer period than the expected pulse width and trigger the oscilloscope from the same pulse. For very long pulses, either store the waveform using a digital storage oscilloscope or shorten the time constant by some known factor. For example, replace a 1000 μF capacitor with a 1 μF capacitor to shorten the time by a factor of 1000. A faster pulse is easier to see and measure with an oscilloscope.

SECTION 7-5 CHECKUP

1. Describe the difference between a nonretriggerable and a retriggerable one-shot.
2. How is the output pulse width set in most IC one-shots?
3. What is the pulse width of a 555 timer one-shot when $C = 1 \mu\text{F}$ and $R = 10 \text{ k}\Omega$?

7-6 The Astable Multivibrator

An **astable** multivibrator is a device that has no stable states; it changes back and forth (oscillates) between two unstable states without any external triggering. The resulting output is typically a square wave that is used as a clock signal in many types of sequential logic circuits. Astable multivibrators are also known as pulse **oscillators**.

After completing this section, you should be able to

- ◆ Describe the operation of a simple astable multivibrator using a Schmitt trigger circuit.
- ◆ Set up a 555 timer as an astable multivibrator.

Figure 7-55(a) shows a simple form of astable multivibrator using an inverter with hysteresis (Schmitt trigger) and an RC circuit connected in a feedback arrangement. When power is first applied, the capacitor has no charge; so the input to the Schmitt trigger inverter is LOW and the output is HIGH. The capacitor charges through R until the inverter input voltage reaches the upper trigger point (UTP), as shown in Figure 7-55(b). At this point, the inverter output goes LOW, causing the capacitor to discharge back through R , shown in part (b). When the inverter input voltage decreases to the lower trigger point (LTP), its output goes HIGH and the capacitor charges again. This charging/discharging cycle continues to repeat as long as power is applied to the circuit, and the resulting output is a pulse waveform, as indicated.

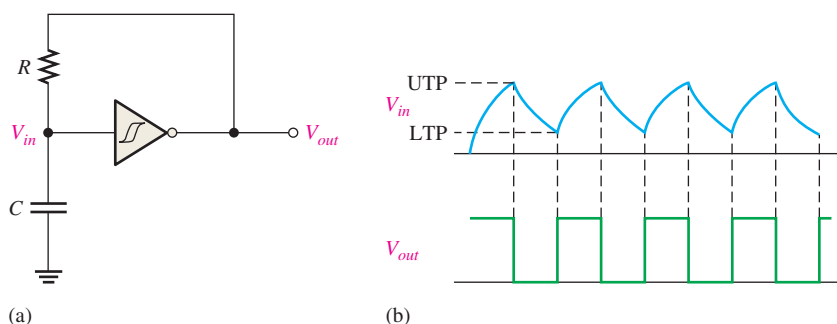


FIGURE 7-55 Basic astable multivibrator using a Schmitt trigger.

The 555 Timer as an Astable Multivibrator

A 555 timer connected to operate as an astable multivibrator is shown in Figure 7-56. Notice that the threshold input (*THRESH*) is now connected to the trigger input (*TRIG*). The external components R_1 , R_2 , and C_1 form the timing network that sets the frequency of oscillation. The $0.01 \mu\text{F}$ capacitor, C_2 , connected to the control (*CONT*) input is strictly for decoupling and has no effect on the operation; in some cases it can be left off.

InfoNote

Most systems require a timing source to provide accurate clock waveforms. The timing section controls all system timing and is responsible for the proper operation of the system hardware. The timing section usually consists of a crystal-controlled oscillator and counters for frequency division. Using a high-frequency oscillator divided down to a lower frequency provides for greater accuracy and frequency stability.

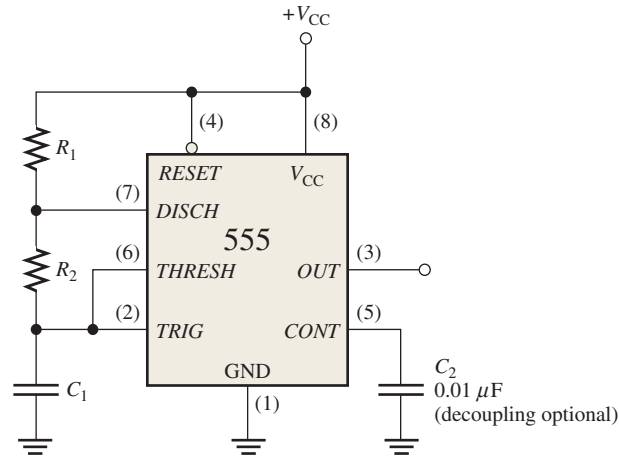


FIGURE 7-56 The 555 timer connected as an astable multivibrator (oscillator).

Initially, when the power is turned on, the capacitor (C_1) is uncharged and thus the trigger voltage (pin 2) is at 0 V. This causes the output of comparator B to be HIGH and the output of comparator A to be LOW, forcing the output of the latch, and thus the base of Q_1 , LOW and keeping the transistor off. Now, C_1 begins charging through R_1 and R_2 , as indicated in Figure 7-57. When the capacitor voltage reaches $\frac{1}{3} V_{CC}$, comparator B switches to its LOW output state; and when the capacitor voltage reaches $\frac{2}{3} V_{CC}$, comparator A switches to its HIGH output state. This resets the latch, causing the base of Q_1 to go HIGH and turning on the transistor. This sequence creates a discharge path for the capacitor through R_2 and the transistor, as indicated. The capacitor now begins to discharge, causing comparator A to go LOW. At the point where the capacitor discharges down to $\frac{1}{3} V_{CC}$, comparator B switches HIGH; this sets the latch, making the base of Q_1 LOW and turning off the transistor. Another charging cycle begins, and the entire process repeats. The

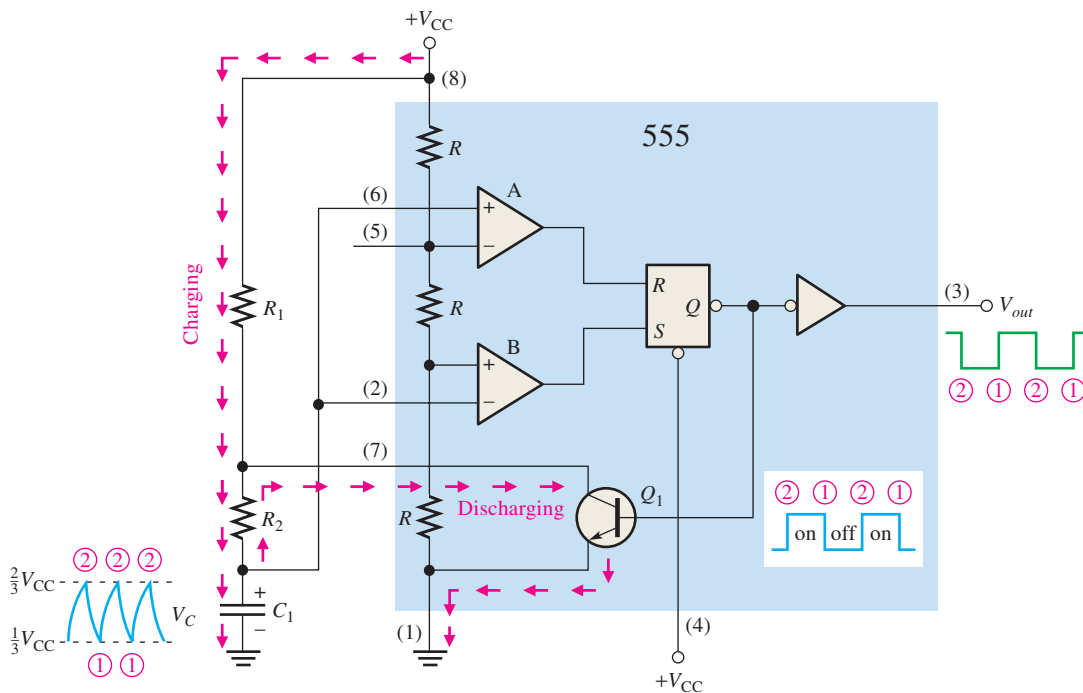


FIGURE 7-57 Operation of the 555 timer in the astable mode.

result is a rectangular wave output whose duty cycle depends on the values of R_1 and R_2 . The frequency of oscillation is given by the following formula, or it can be found using the graph in Figure 7-58.

$$f = \frac{1.44}{(R_1 + 2R_2)C_1} \quad \text{Equation 7-4}$$

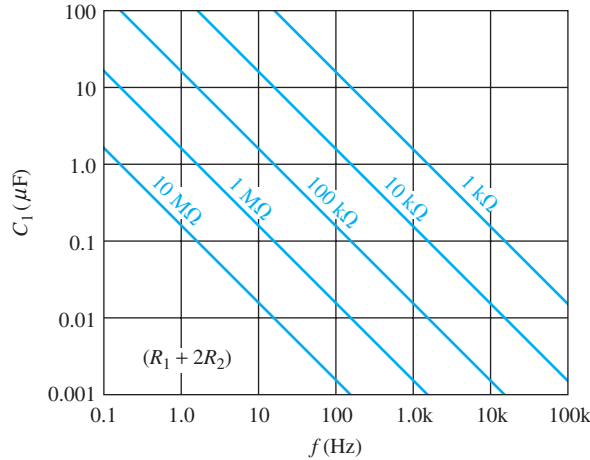


FIGURE 7-58 Frequency of oscillation as a function of C_1 and $R_1 + 2R_2$. The sloped lines are values of $R_1 + 2R_2$.

By selecting R_1 and R_2 , the duty cycle of the output can be adjusted. Since C_1 charges through $R_1 + R_2$ and discharges only through R_2 , duty cycles approaching a minimum of 50 percent can be achieved if $R_2 \gg R_1$ so that the charging and discharging times are approximately equal.

An expression for the duty cycle is developed as follows. The time that the output is HIGH (t_H) is how long it takes C_1 to charge from $\frac{1}{3} V_{CC}$ to $\frac{2}{3} V_{CC}$. It is expressed as

$$t_H = 0.7(R_1 + R_2)C_1 \quad \text{Equation 7-5}$$

The time that the output is LOW (t_L) is how long it takes C_1 to discharge from $\frac{1}{3} V_{CC}$ to $\frac{2}{3} V_{CC}$. It is expressed as

$$t_L = 0.7R_2C_1 \quad \text{Equation 7-6}$$

The period, T , of the output waveform is the sum of t_H and t_L . This is the reciprocal of f in Equation 7-4.

$$T = t_H + t_L = 0.7(R_1 + 2R_2)C_1$$

Finally, the duty cycle is

$$\begin{aligned} \text{Duty cycle} &= \frac{t_H}{T} = \frac{t_H}{t_H + t_L} \\ \text{Duty cycle} &= \left(\frac{R_1 + R_2}{R_1 + 2R_2} \right) 100\% \end{aligned} \quad \text{Equation 7-7}$$

To achieve duty cycles of less than 50 percent, the circuit in Figure 7-56 can be modified so that C_1 charges through only R_1 and discharges through R_2 . This is achieved with a diode, D_1 , placed as shown in Figure 7-59. The duty cycle can be made less than 50 percent by making R_1 less than R_2 . Under this condition, the expression for the duty cycle is

$$\text{Duty cycle} = \left(\frac{R_1}{R_1 + R_2} \right) 100\% \quad \text{Equation 7-8}$$

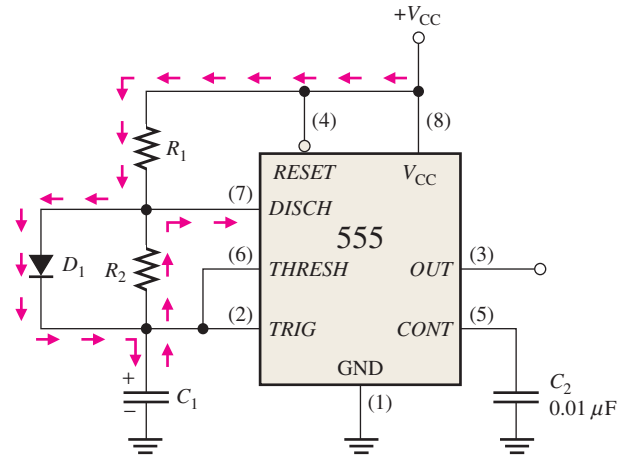


FIGURE 7-59 The addition of diode D_1 allows the duty cycle of the output to be adjusted to less than 50 percent by making $R_1 < R_2$.

EXAMPLE 7-14

A 555 timer configured to run in the astable mode (pulse oscillator) is shown in Figure 7-60. Determine the frequency of the output and the duty cycle.

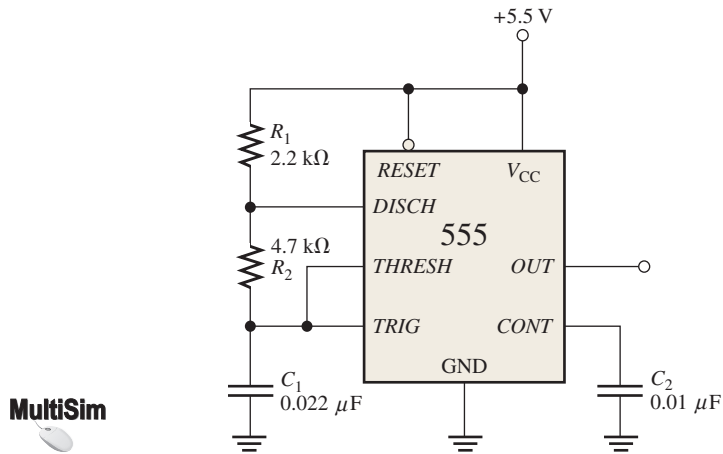


FIGURE 7-60 Open file F07-60 to verify operation.

Solution

Use Equations 7-4 and 7-7.

$$f = \frac{1.44}{(R_1 + 2R_2)C_1} = \frac{1.44}{(2.2 \text{ k}\Omega + 9.4 \text{ k}\Omega)0.022 \text{ }\mu\text{F}} = \mathbf{5.64 \text{ kHz}}$$

$$\text{Duty cycle} = \left(\frac{R_1 + R_2}{R_1 + 2R_2} \right) 100\% = \left(\frac{2.2 \text{ k}\Omega + 4.7 \text{ k}\Omega}{2.2 \text{ k}\Omega + 9.4 \text{ k}\Omega} \right) 100\% = \mathbf{59.5\%}$$

Related Problem

Determine the duty cycle in Figure 7-60 if a diode is connected across R_2 as indicated in Figure 7-59.

SECTION 7-6 CHECKUP

1. Explain the difference in operation between an astable multivibrator and a monostable multivibrator.
2. For a certain astable multivibrator, $t_H = 15$ ms and $T = 20$ ms. What is the duty cycle of the output?