

Fig. 5-1 4-Bit Register

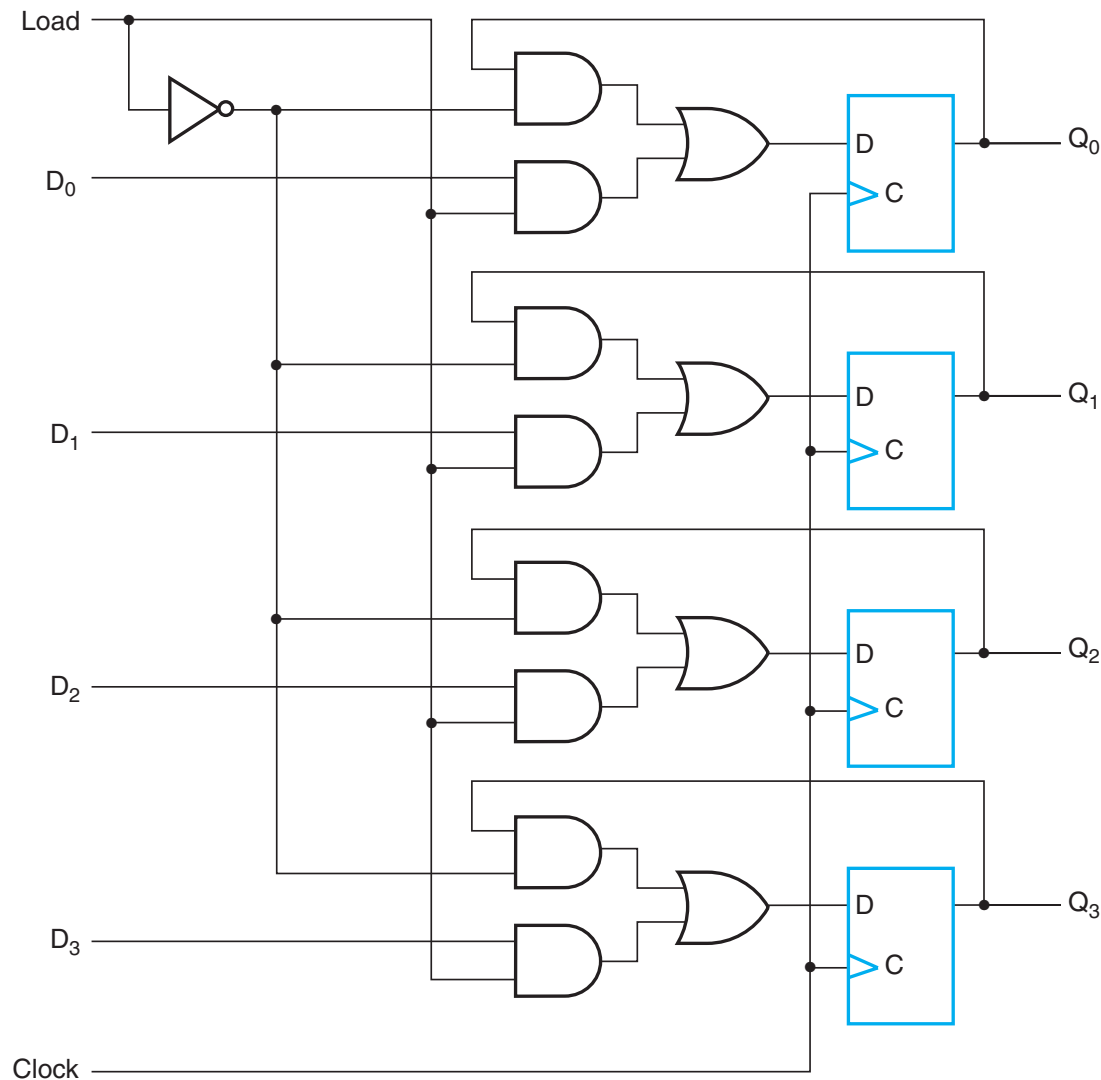


Fig. 5-2 4-Bit Register with Parallel Load

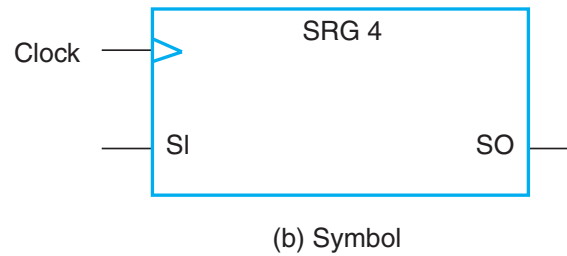
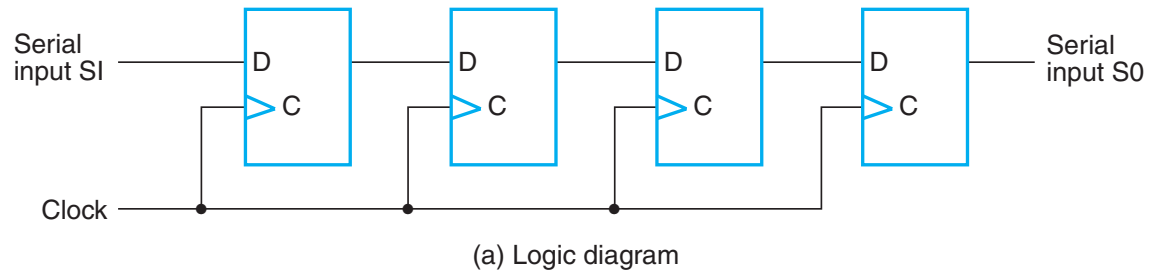


Fig. 5-3 4-Bit Shift Register

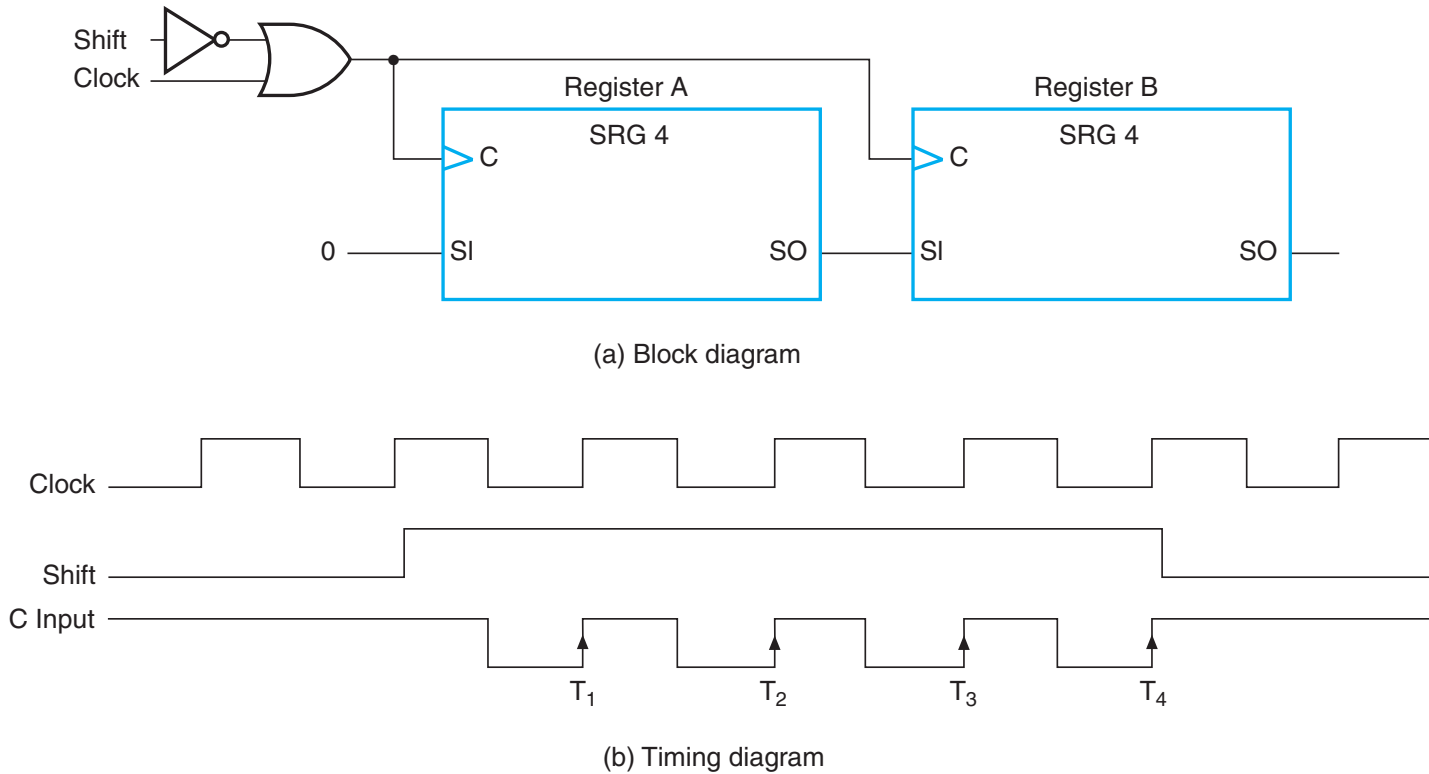


Fig. 5-4 Serial Transfer

TABLE 5-1
Example of Serial Transfer

Timing pulse	Shift Register A				Shift Register B			
Initial value	1	0	1	1	0	0	1	0
After T_1	0	1	0	1	1	0	0	1
After T_2	0	0	1	0	1	1	0	0
After T_3	0	0	0	1	0	1	1	0
After T_4	0	0	0	0	1	0	1	1

Table 5-1 Example of Serial Transfer

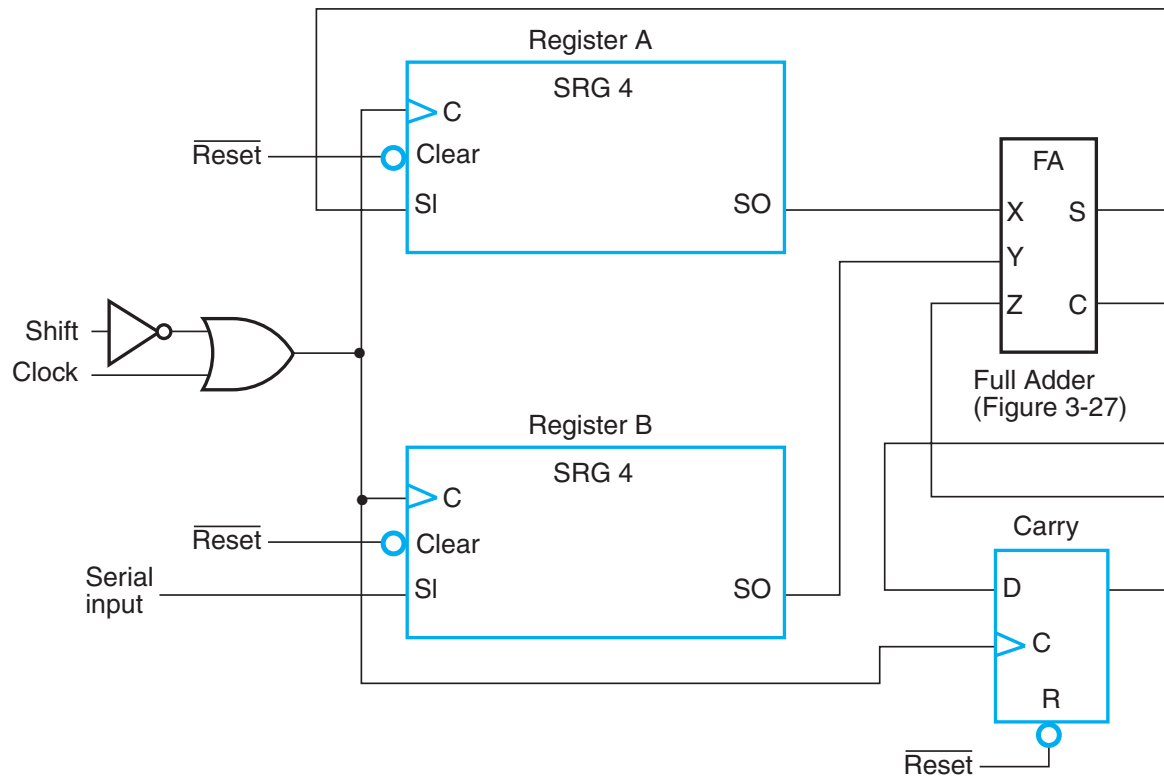


Fig. 5-5 Serial Addition

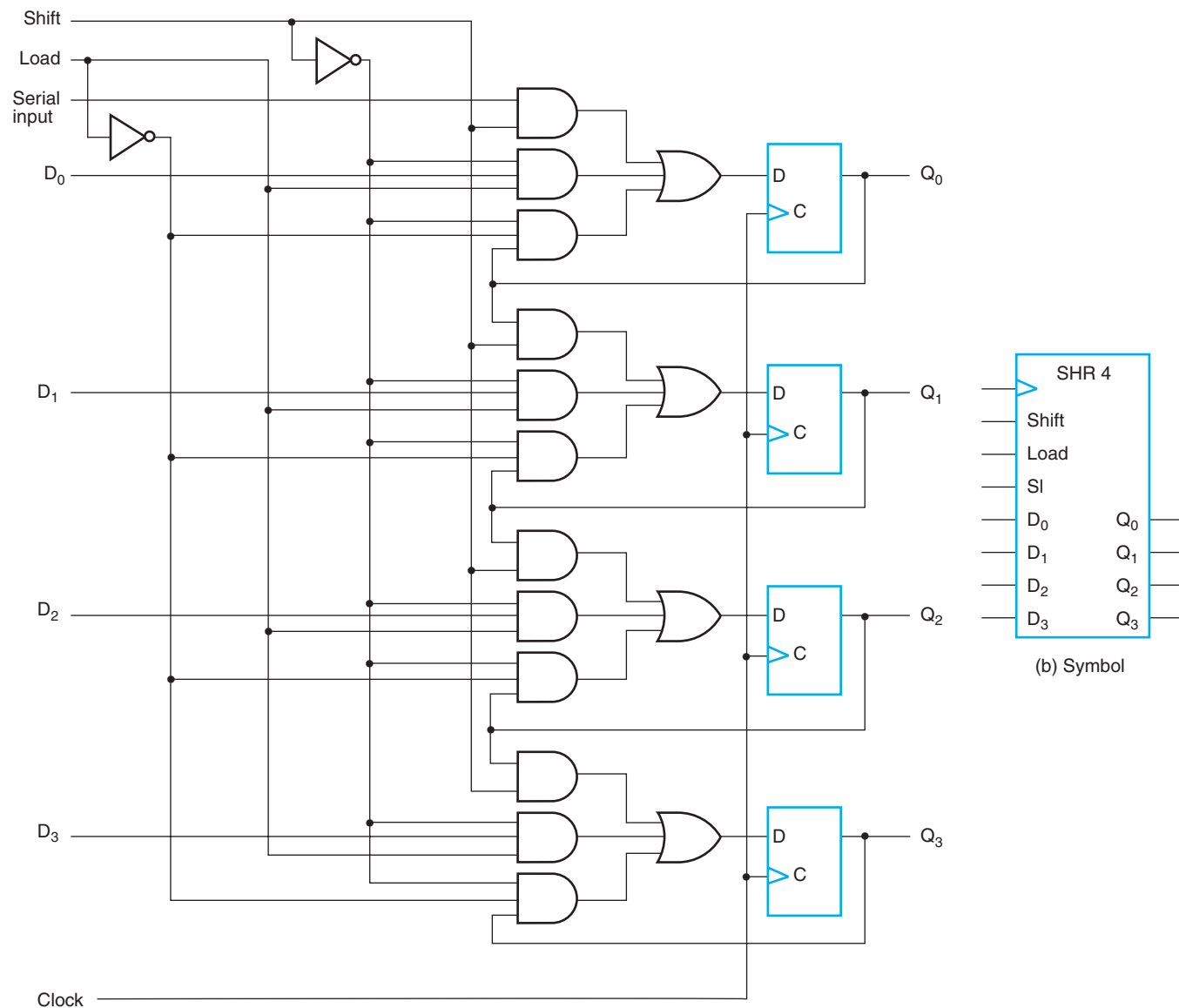
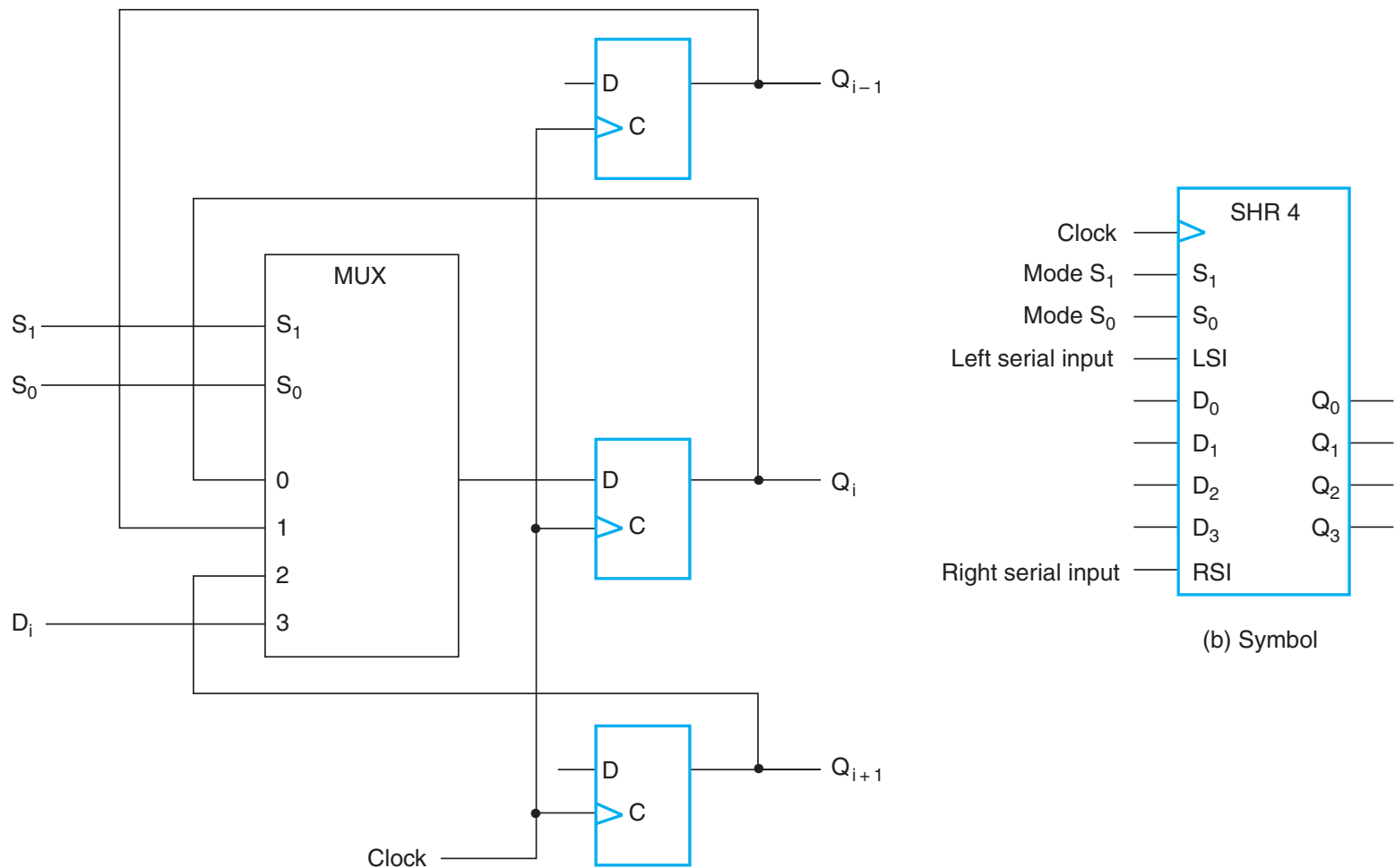


Fig. 5-6 Shift Register with Parallel Load

□ TABLE 5-2
Function Table for the Register of Figure 5-6

Shift	Load	Operation
0	0	No change
0	1	Load parallel data
1	×	Shift down from Q_0 to Q_3

Table 5-2 Function Table for the Register of Figure 5-6



(a) Logic diagram of one typical stage

(b) Symbol

Fig. 5-7 Bidirectional Shift Register with Parallel Load

□ TABLE 5-3
Function Table for the Register of Figure 5-7

Mode control		Register Operation
S_1	S_0	
0	0	No change
0	1	Shift down
1	0	Shift up
1	1	Parallel load

Table 5-3 Function Table for the Register of Figure 5-7

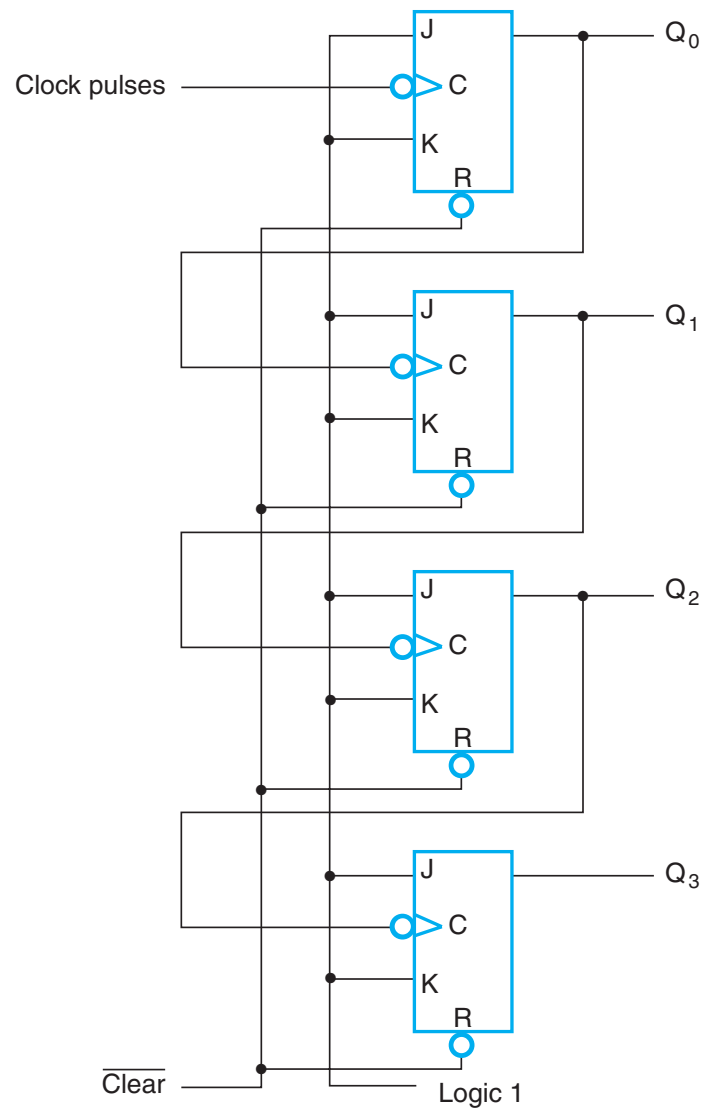


Fig. 5-8 4-Bit Ripple Counter

TABLE 5-4
Counting Sequence of Binary Counter

Upward Counting Sequence				Downward Counting Sequence			
Q ₃	Q ₂	Q ₁	Q ₀	Q ₃	Q ₂	Q ₁	Q ₀
0	0	0	0	1	1	1	1
0	0	0	1	1	1	1	0
0	0	1	0	1	1	0	1
0	0	1	1	1	1	0	0
0	1	0	0	1	0	1	1
0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	0	0
1	0	0	0	0	1	1	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	0
1	1	0	0	0	0	1	1
1	1	0	1	0	0	1	0
1	1	1	0	0	0	0	1
1	1	1	1	0	0	0	0

Table 5-4 Counting Sequence of Binary Counter

TABLE 5-5
State Table and Flip-Flop Inputs for Binary Counter

Present state				Next state				Flip-flop inputs							
Q ₃	Q ₂	Q ₁	Q ₀	Q ₃	Q ₂	Q ₁	Q ₀	J _{Q3}	K _{Q3}	J _{Q2}	K _{Q2}	J _{Q1}	K _{Q1}	J _{Q0}	K _{Q0}
0	0	0	0	0	0	0	1	0	×	0	×	0	×	1	×
0	0	0	1	0	0	1	0	0	×	0	×	1	×	×	1
0	0	1	0	0	0	1	1	0	×	0	×	×	0	1	×
0	0	1	1	0	1	0	0	0	×	1	×	×	1	×	1
0	1	0	0	0	1	0	1	0	×	×	0	0	×	1	×
0	1	0	1	0	1	1	0	0	×	×	0	1	×	×	1
0	1	1	0	0	1	1	1	0	×	×	0	×	0	1	×
0	1	1	1	1	0	0	0	1	×	×	1	×	1	×	1
1	0	0	0	1	0	0	1	×	0	0	×	0	×	1	×
1	0	0	1	1	0	1	0	×	0	0	×	1	×	×	1
1	0	1	0	1	0	1	1	×	0	0	×	×	0	1	×
1	0	1	1	1	1	0	0	×	0	1	×	×	1	×	1
1	1	0	0	1	1	0	1	×	0	×	0	0	×	1	×
1	1	0	1	1	1	1	0	×	0	×	0	1	×	×	1
1	1	1	0	1	1	1	1	×	0	×	0	×	0	1	×
1	1	1	1	0	0	0	0	×	1	×	1	×	1	×	1

Table 5-5 State Table and Flip-Flop Inputs for Binary Counter

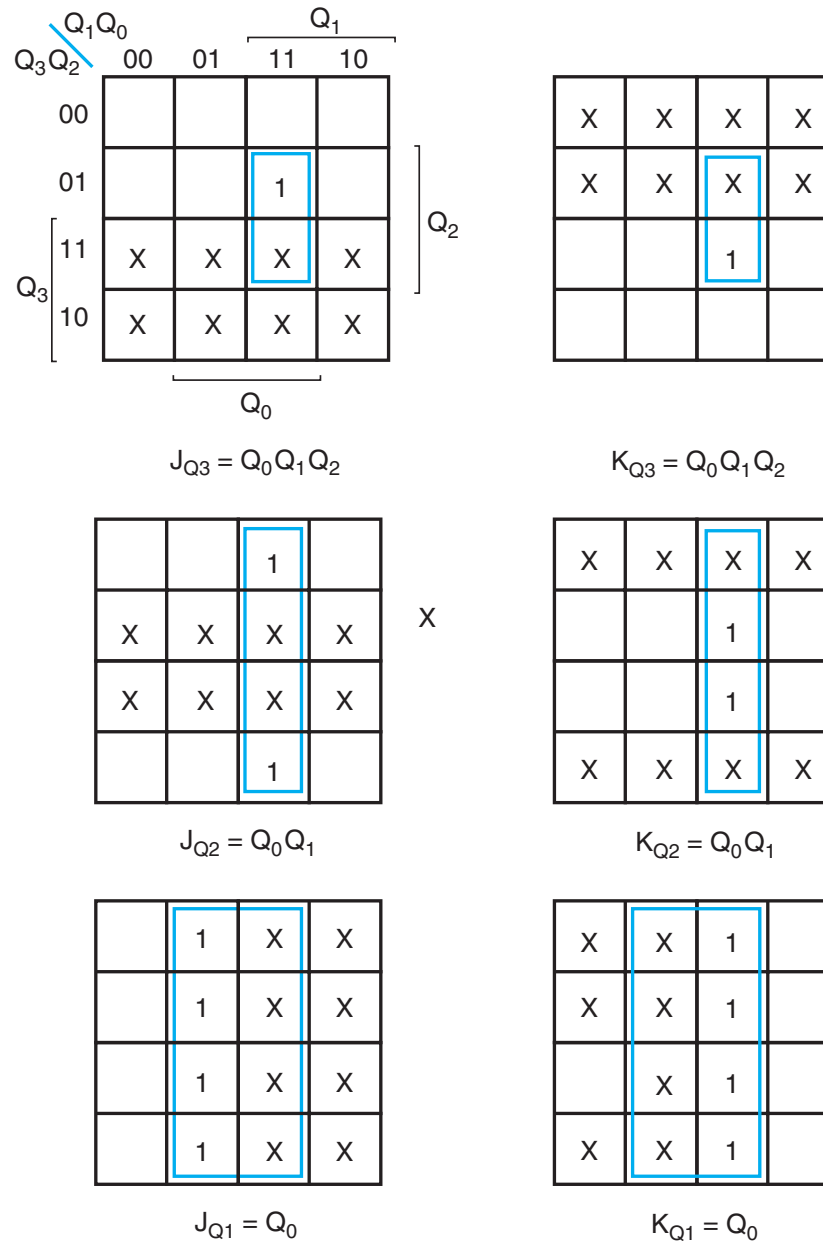


Fig. 5-9 Maps for Input Equations of a Binary Counter

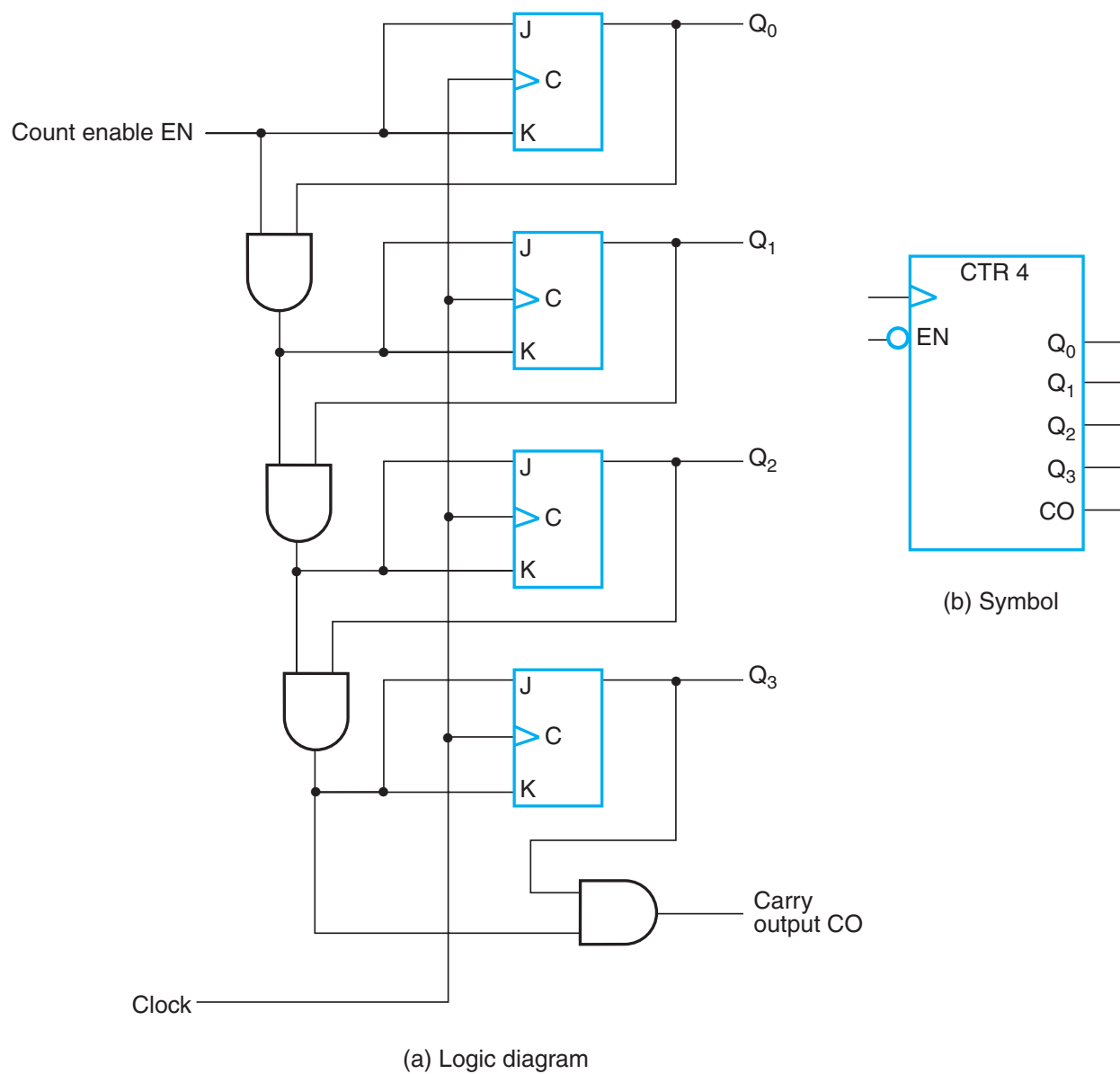
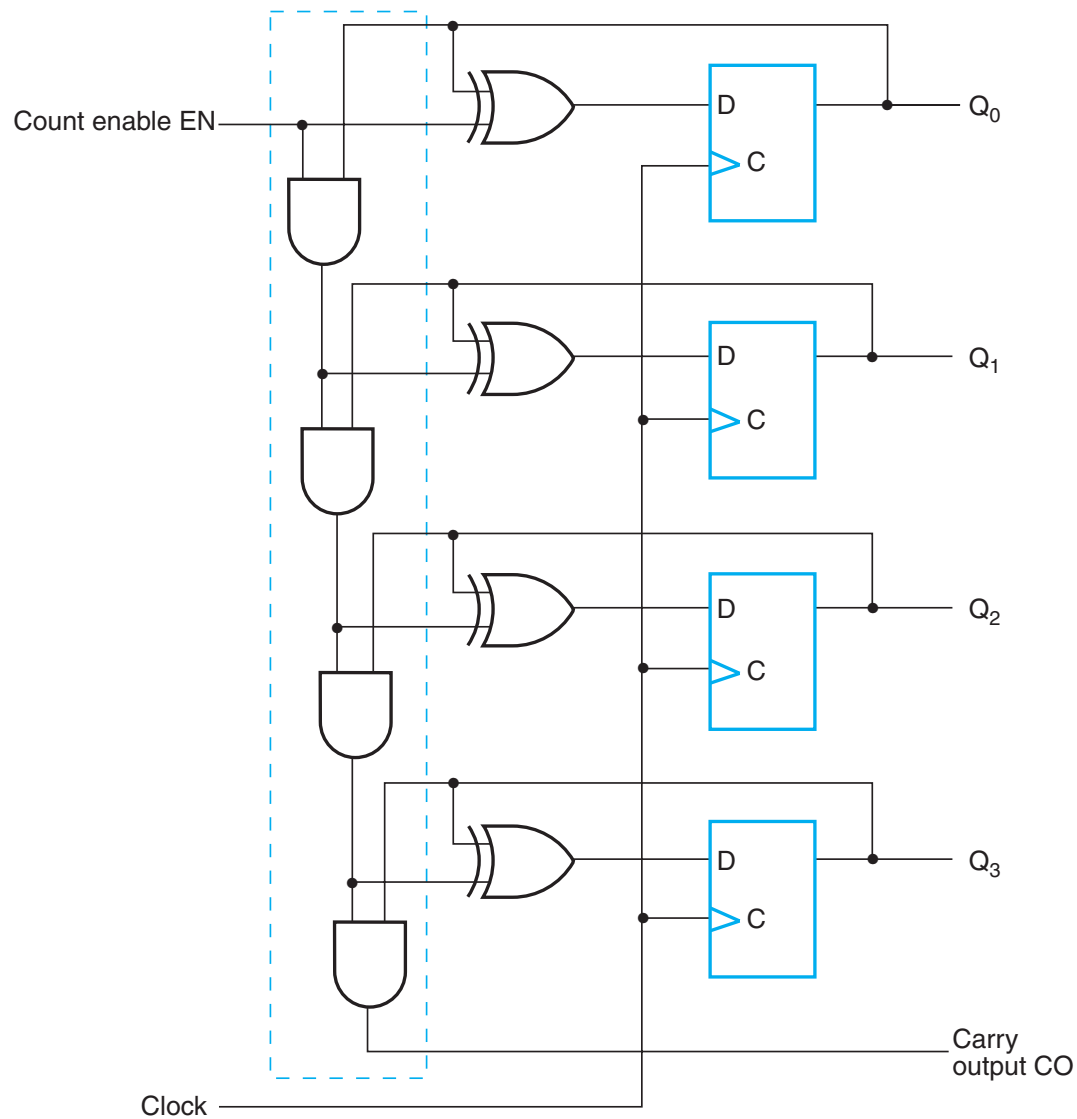
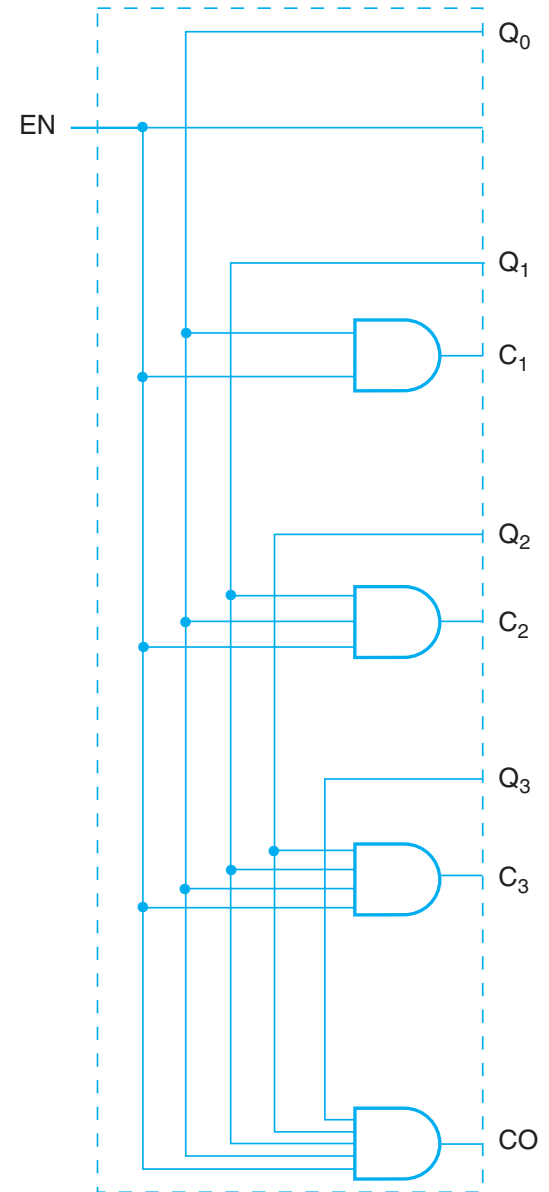


Fig. 5-10 4-Bit Synchronous Binary Counter



(a) Serial gating



(b) Parallel gating

Fig. 5-11 4-Bit Binary Counter with *D* Flip-Flops

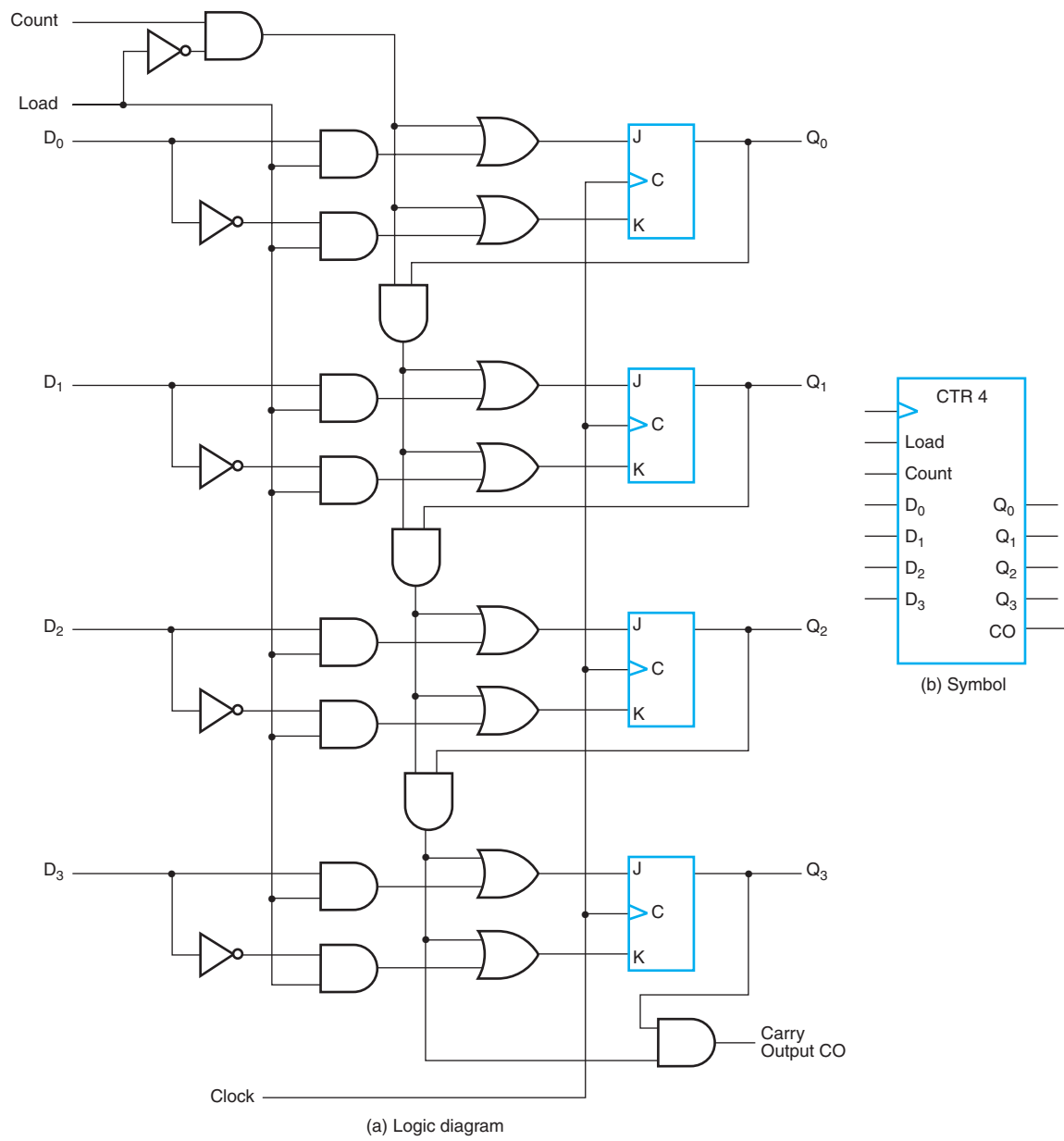


Fig. 5-12 4-Bit Binary Counter with Parallel Load

TABLE 5-6
State Table and Flip-Flop Inputs for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q ₈	Q ₄	Q ₂	Q ₁	Q ₈	Q ₄	Q ₂	Q ₁	Y	T _{Q8}	T _{Q4}	T _{Q2}	T _{Q1}
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

Table 5-6 State Table and Flip-Flop Inputs for BCD Counter

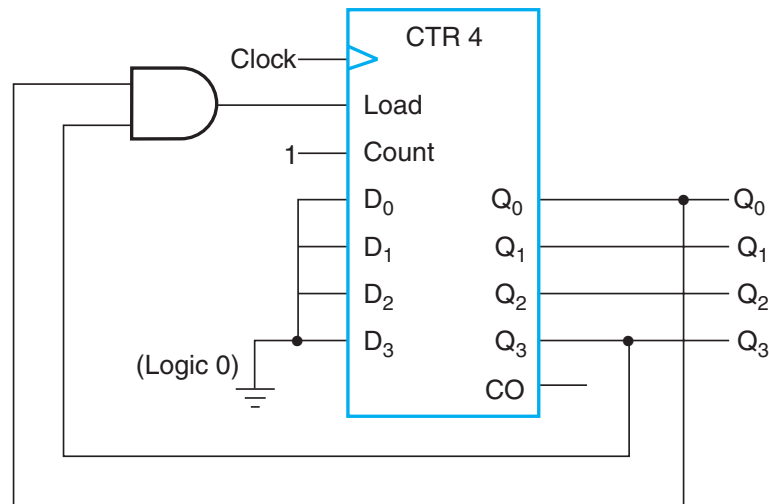
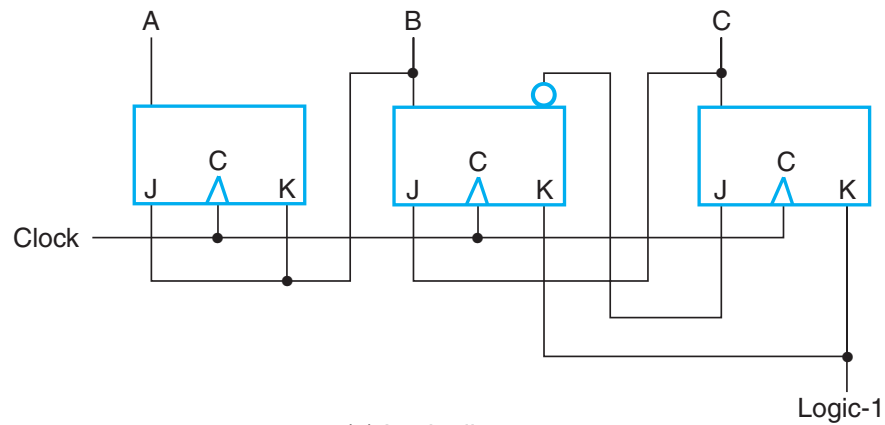


Fig. 5-13 BCD Counter

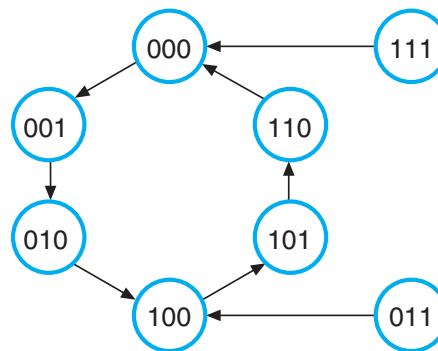
TABLE 5-7
State Table and Flip-Flop Inputs for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q ₈	Q ₄	Q ₂	Q ₁	Q ₈	Q ₄	Q ₂	Q ₁	Y	T _{Q8}	T _{Q4}	T _{Q2}	T _{Q1}
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

Table 5-7 State Table and Flip-Flop Inputs for BCD Counter



(a) Logic diagram



(b) State diagram

Fig. 5-14 Counter with Arbitrary Count

TABLE 5-8
State Table and Flip-Flop Inputs for Counter

Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J _A	K _A	J _B	K _B	J _C	K _C
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	0	1	0	0	×	1	×	×	1
0	1	0	1	0	0	1	×	×	1	0	×
1	0	0	1	0	1	×	0	0	×	1	×
1	0	1	1	1	0	×	0	1	×	×	1
1	1	0	0	0	0	×	1	×	1	0	×

Table 5-8 State Table and Flip-Flop Inputs for Counter

```

-- 4-bit Shift Register with Reset
-- (See Figure 5-3)
library ieee;
use ieee.std_logic_1164.all;

entity srg_4_r is
  port(CLK, RESET, SI : in std_logic;
        Q : out std_logic_vector(3 downto 0);
        SO : out std_logic);
end srg_4_r;

architecture behavioral of srg_4_r is
  signal shift : std_logic_vector(3 downto 0);
begin
  process (RESET, CLK)
  begin
    if (RESET = '1') then
      shift <= "0000";
    elsif (CLK'event and (CLK = '1')) then
      shift <= shift(2 downto 0) & SI;
    end if;
  end process;
  Q <= shift;
  SO <= shift(3);
end behavioral;

```

Fig. 5-15 Behavioral VHDL Description of 4-bit Left Shift Register with Direct Reset

```

-- 4-bit Binary Counter with Reset
-- (See Figure 5-10)
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity count_4_r is
    port (CLK, RESET, EN : in std_logic;
          Q : out std_logic_vector(3 downto 0);
          CO : out std_logic);
end count_4_r;

architecture behavioral of count_4_r is
    signal count : std_logic_vector(3 downto 0);
begin
    process (RESET, CLK)
    begin
        if (RESET = '1') then
            count <= "0000";
        elsif (CLK'event and (CLK = '1') and (EN = '1')) then
            count <= count + "0001";
        end if;
    end process;
    Q <= count;
    CO <= '1' when count = "1111" and EN = '1' else '0';
end behavioral;

```

Fig. 5-16 Behavioral VHDL Description of 4-bit Binary Counter with Direct Reset


```

// 4-bit Shift Register with Reset
// (See Figure 5-3)

module srg_4_r_v (CLK, RESET, SI, Q, SO);
    input CLK, RESET, SI;
    output [3:0] Q;
    output SO;

    reg [3:0] Q;

    assign SO = Q[3];

    always@(posedge CLK or posedge RESET)
    begin
        if (RESET)
            Q <= 4'b0000;
        else
            Q <= {Q[2:0], SI};
    end
endmodule

```

Fig. 5-17 Behavioral Verilog Description of 4-bit Left Shift Register with Direct Reset

```

// 4-bit Binary Counter with Reset
// (See Figure 5-10)

module count_4_r_v (CLK, RESET, EN, Q, CO);
    input CLK, RESET, EN;
    output [3:0] Q;
    output CO;

    reg [3:0] Q;

    assign CO = (count == 4'b1111 && EN == 1'b1) ? 1 : 0;
    always@(posedge CLK or posedge RESET)
    begin
        if (RESET)
            Q <= 4'b0000;
        else if (EN)
            Q <= Q + 4'b0001;
        end
    endmodule

```

Fig. 5-18 Behavioral Verilog Description of 4-bit Binary Counter with Direct Reset