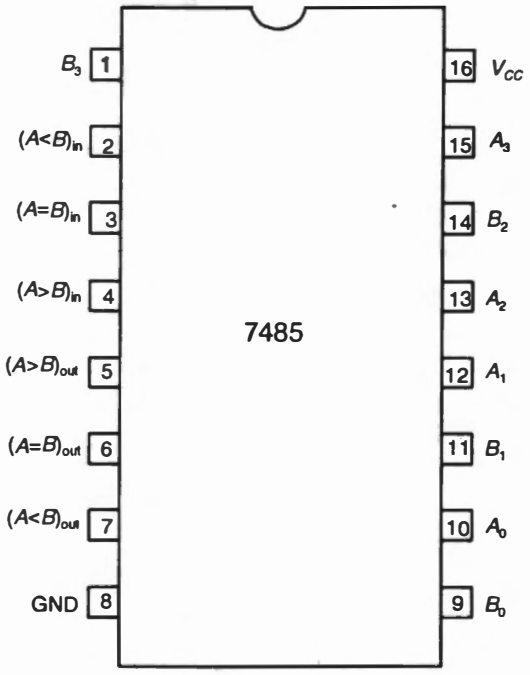
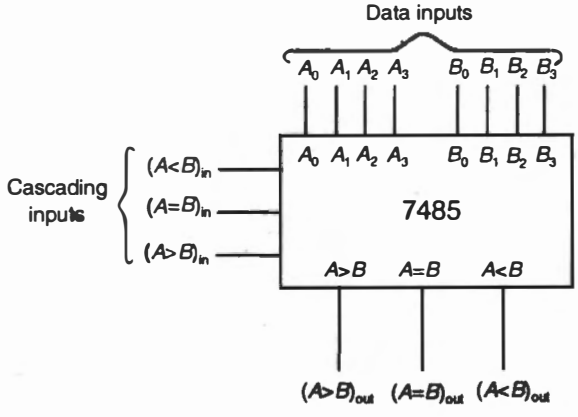
The 7485 compares two unsigned 4-bit binary numbers (i.e., words). One of them is $A_3\ A_2\ A_1 A_0$ (i.e., word A); the other is $B_3\ B_2\ B_1\ B_0$ (i.e., word B). Three fully decoded decisions about two 4-bit words



(a) Pinout for 7485



(b) Logic symbol for 7485

Fig. 5.12 The pinout and logic symbol for 7485

(A, B) are made and are externally available at three outputs. Output $A < B$ will be HIGH when the magnitude of word A is less than the magnitude of word B. Output $A = B$ will be HIGH when word A and B are identical. Output $A > B$ will be HIGH when the magnitude of word A is greater than the magnitude of word B. Words of greater length may be compared by connecting comparators in cascade.

Implementation of Boolean Function using Multiplexers

A Boolean function can be implemented using a multiplexer. If we have a Boolean function of $n + 1$ variables, we take n of these variables and connect them to the selection lines of a multiplexer. The remaining single variable of the function is used for the inputs of the multiplexer. If A is this single variable, the inputs of the multiplexer are chosen to be either A or $\bar{A}$ or 1 or 0. In this way it is possible to implement any Boolean function of $n + 1$ variables with a 2^n-to-1 multiplexer. We demonstrate the procedure with the following example.

EXAMPLE 5.2: Implement the function $F(A, B, C) = \Sigma(2, 3, 5, 6)$ using multiplexer.

SOLUTION: The given Boolean function is a 3-variable function. Hence we need a multiplexer with two (i.e., $3 - 1 = 2$) selection lines and four (i.e., 2^2) inputs. We take two variables B and C, and connect them to the selection lines S_1 and S_0, respectively.

The truth table shown in Fig. 5.26(a) of the given function reveals that the first half of the minterms $\left(m_0 = \bar{A}\bar{B}\bar{C},\ m_1 = \bar{A}\bar{B}C,\ m_2 = \bar{A}B\bar{C} \text{ and } m_3 = \bar{A}BC\right)$ are associated with $\bar{A}$ and the second half $\left(m_4 = A\bar{B}\bar{C},\ m_5 = A\bar{B}C,\ m_6 = AB\bar{C} \text{ and } m_7 = ABC\right)$ with A, since the variable A is in the highest order position in the sequence of variables.

Now we go for preparing implementation table from which we can find values for the multiplexer inputs. The procedure is as follows. First, list the inputs of the multiplexer, I_0 through I_3, and under them

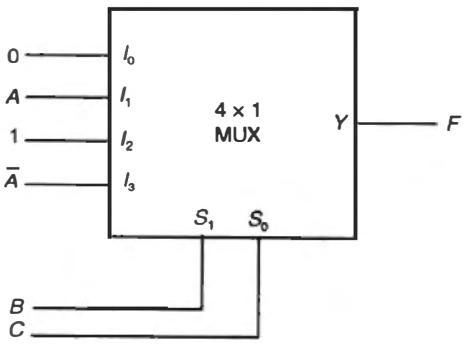
Minterm	A	B	C	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

(a) Truth table

	I_0	I_1	I_2	I_3
$\bar{A}$	0	1	②	③
A	4	⑤	⑥	7
	0	A	1	$\bar{A}$

Values for the multiplexer inputs

(b) Implementation table



(c) Multiplexer implementation

Fig. 5.26 Implementing $F(A, B, C) = \Sigma(2, 3, 5, 6)$ with a multiplexer

5.8 Implement a 4 × 64 decoder using one 2 × 4 decoder and four 4 × 16 decoders.

5.9 Implement the following two functions using a dual 4 × 1 multiplexer chip (74153).

$$f_1(A, B, C) = ABC + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}C$$

$$f_2(A, B, C) = \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + A\bar{B}C$$

5.10 Implement a 16 × 1 multiplexer using two 8 × 1 multiplexers and an OR gate.

Edge-Triggered *T* Flip-Flop

The *T* flip-flop is a single-input version of the *J-K* flip-flop. As shown in Fig. 6.19, the *T* flip-flop is obtained from a *J-K* flip-flop if both the inputs *J* and *K* are tied together. It receives designation *T* from its ability to 'toggle'. Its truth table is given in Table 6.6.

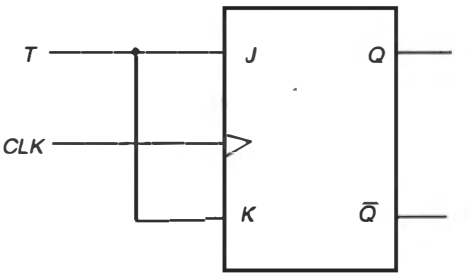


Fig. 6.19 *J-K* flip-flop converted into a *T* flip-flop

Table 6.6 Truth table for a positive edge-triggered *T* flip-flop

Inputs		Output	Comments
T	*CLK*	*Q*	
0	↑	Q_p	No change
1	↑	$\bar{Q}_p$	Toggle

Note: ↑ = PGT, Q_p = Prior output state

From the truth table it is clear that if $T = 1$ the flip-flop acts as a toggle switch. The logic symbols for edge triggered *T* flip-flops are shown in Fig. 6.20.

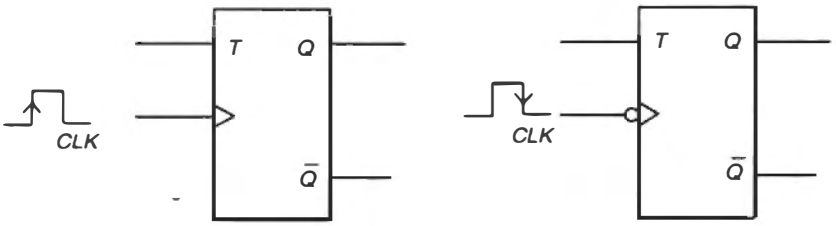


Fig. 6.20 Logic symbols for edge triggered *T* flip-flops that trigger on: (a) PGT (b) NGT

The output *Q* of an edge-triggered *T* flip-flop for the *T* input and the clock is shown in Fig. 6.21. It is assumed that the flip-flop is initially RESET.