## 4.2 Decoders

A decoder is a multiple-input, multiple-output logic circuit which converts coded inputs into coded outputs, where the input and output codes are different. The input code generally has fewer bits than the output code. Each input code word produces a different output code word, i.e., there is one-to-one mapping from input code words into output code words. This one-to-one mapping can be expressed in a truth table.

The Fig. 4.6 shows the general structure of the decoder circuit. As shown in the Fig. 4.6, the encoded information is presented as n inputs producing $2^n$ possible outputs. The $2^n$ output values are from 0 through $2^n - 1$. Sometimes an n-bit binary code is truncated to represent fewer output values than $2^n$. For example, in the BCD code, the 4-bit combinations 0000 through 1001 represent the decimal digits 0-9, and combinations 1010 through 1111 are not used. Usually, a decoder is provided with enable inputs to activate decoded output based on data inputs. When any one enable input is unasserted, all outputs of decoder are disabled.

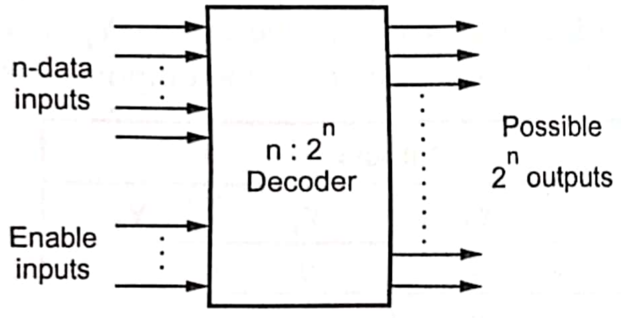


Fig. 4.6 General structure of decoder

### 4.2.1 Binary Decoder

A decoder which has an n-bit binary input code and a one activated output out-of-$2^n$ output code is called **binary decoder**. A binary decoder is used when it is necessary to activate exactly one of $2^n$ outputs based on an n-bit input value.

Fig. 4.7 shows 2 to 4 decoder. Here, 2 inputs are decoded into four outputs, each output representing one of the minterms of the 2 input variables. The two inverters provide the complement of the inputs, and each one of four AND gates generates one of the minterms.

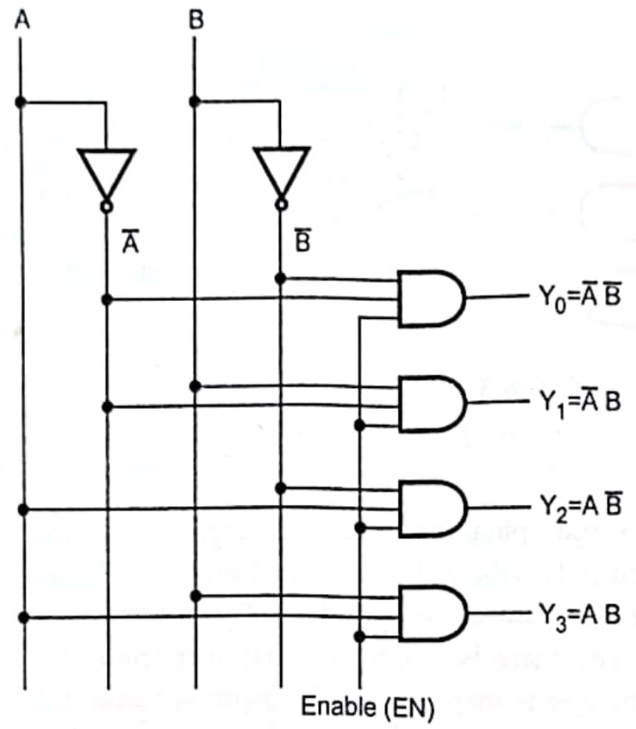


**Fig. 4.7 2-to-4 line decoder**

The Table 4.2 shows the truth table for a 2-to-4 decoder. As shown in the truth table, if enable input is 1 (EN = 1), one, and only one, of the outputs $Y_0$ to $Y_3$, is active for a given input. The output $Y_0$ is active, i.e. $Y_0$ = 1 when inputs A = B = 0, the output $Y_1$ is active when inputs A = 0 and B = 1. If enable input is 0, i.e. EN = 0, then all the outputs are 0.

| Inputs | | | Outputs | | | |
|---|---|---|---|---|---|---|
| EN | A | B | $Y_3$ | $Y_2$ | $Y_1$ | $Y_0$ |
| 0 | X | X | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 |

**Table 4.2 Truth table for a 2-to-4 decoder**

**Example 4.3 :** *Draw the circuit for 3-to-8 decoder and explain.*

**Solution :** Fig. 4.8 shows 3-to-8 line decoder. Here, 3 inputs are decoded into eight outputs, each output represent one of the minterms of the 3-input variables. The three inverters provide the complement of the inputs, and each one of the eight AND gates generates one of the minterms. Enable input is provided to activate decoded output based on data inputs A, B, and C. The table shows the truth table for 3 to 8 decoder.

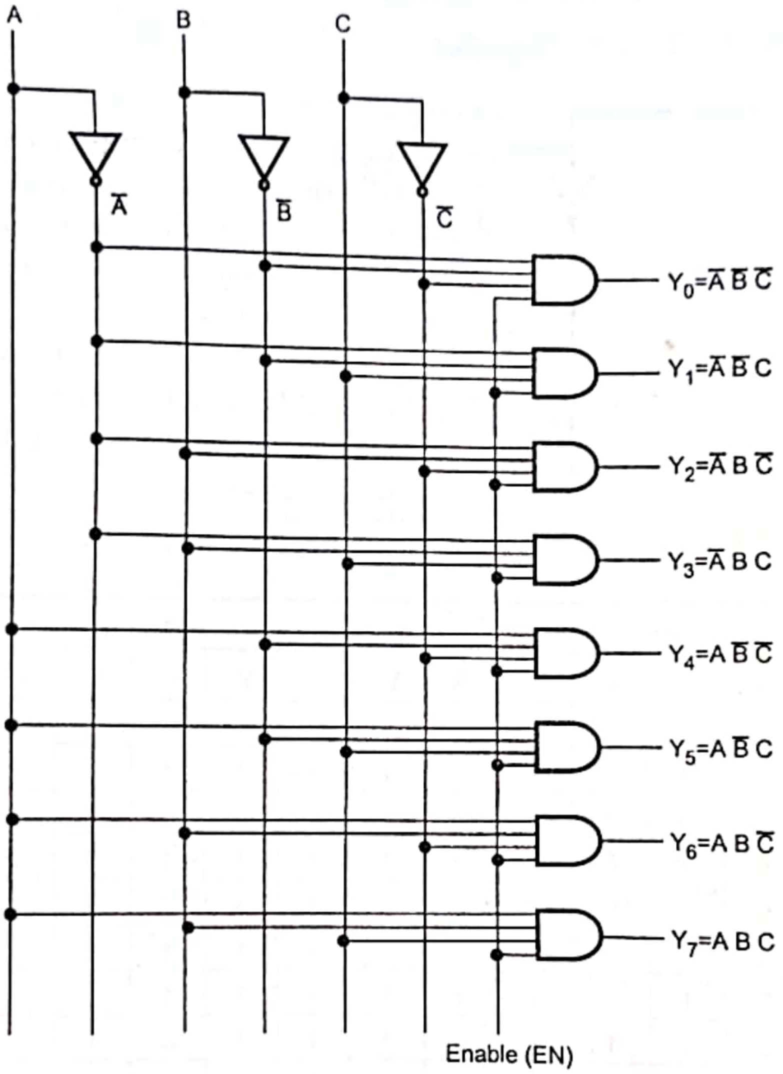


**Fig. 4.8 3:8 line decoder**

| Inputs | | | | Outputs | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| EN | A | B | C | $Y_7$ | $Y_6$ | $Y_5$ | $Y_4$ | $Y_3$ | $Y_2$ | $Y_1$ | $Y_0$ |
| 0 | X | X | X | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 4.3 Truth table for a 3-to-8 decoder**

### 4.2.2 The 74X138 3-to-8 Decoder

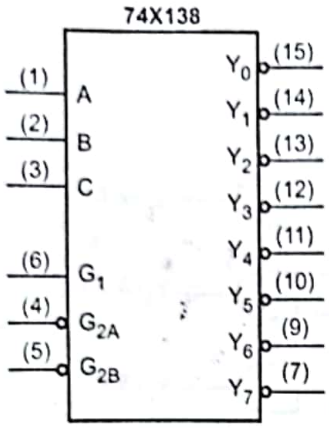


Fig. 4.9 Logic symbol

The 74X138 is a commercially available 3-to-8 decoder. It accepts three binary inputs (A, B, C) and when enabled, provides eight individual active low outputs ($Y_0$ - $Y_7$). The device has three enable inputs : two active low ($\overline{G}_{2A}, \overline{G}_{2B}$) and one active high ($G_1$). Fig. 4.9 and Table 4.4 show logic symbol and function table respectively.

| Inputs | | | | | | Outputs | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $G_{2B}$ | $G_{2A}$ | $G_1$ | C | B | A | $\overline{Y_7}$ | $\overline{Y_6}$ | $\overline{Y_5}$ | $\overline{Y_4}$ | $\overline{Y_3}$ | $\overline{Y_2}$ | $\overline{Y_1}$ | $\overline{Y_0}$ |
| 1 | X | X | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| X | 1 | X | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| X | X | 0 | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Table 4.4 Function table

### 4.2.3 The 74X139 Dual 2-to-4 Decoder

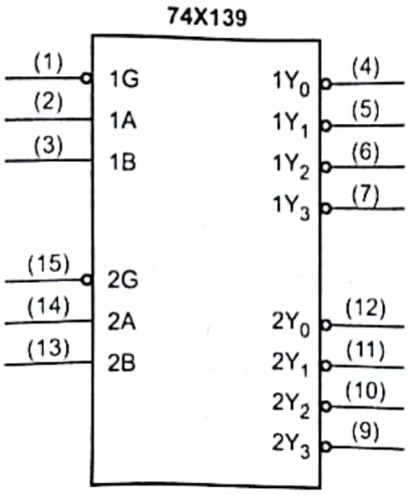


Fig. 4.10 Logic symbol for IC 74X139

The 74X139 consists of two independent and identical 2-to-4 decoders. The enable inputs and outputs of IC 74X139 are active low. Fig. 4.10 shows the logic symbol and Table 4.11 shows the function table for IC 74X139. The Table 4.5 shows the truth table for one half of a 74X139 dual 2-to-4 decoder. The truth table for other half is same as first half.

| Inputs | | | Outputs | | | |
|---|---|---|---|---|---|---|
| $\overline{G}$ | B | A | $\overline{Y}_3$ | $\overline{Y}_2$ | $\overline{Y}_1$ | $\overline{Y}_0$ |
| 1 | x | x | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 |

Table 4.5 Truth table for one half of a 74X139

### 4.2.4 Cascading Binary Decoders

Binary decoder circuits can be connected together to form a larger decoder circuit. Fig. 4.11 shows the 4 × 16 decoder using two 3 × 8 decoders.

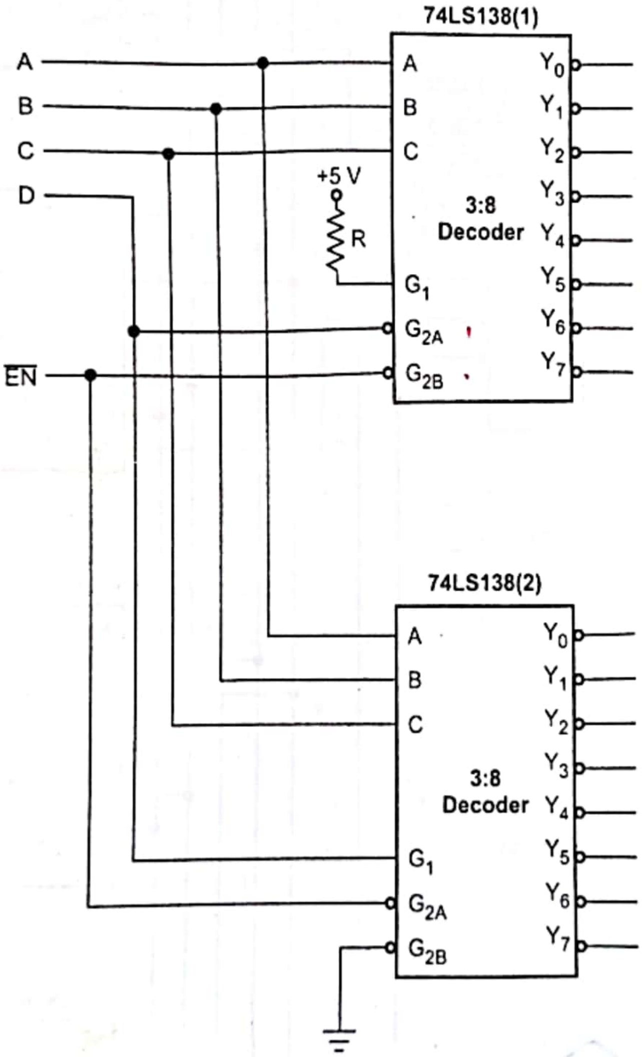


Fig. 4.11 4:16 decoder using two 74LS138 ICs (3 : 8 decoder)

Here, one input line (D) is used to enable/disable the decoders. When D = 0, the top decoder is enabled and the other is disabled. Thus the bottom decoder outputs are all 1s, and the top eight outputs generate minterms 0 0 0 0 to 0 1 1 1. When D=1, the enable conditions are reversed and thus bottom decoder outputs generate minterms 1000 to 1111, while the outputs of the top decoder are all 1s.

➡ **Example 4.4 :** *Design 5-to-32 decoder using one 2-to-4 and four 3-to-8 decoder ICs.*

**Solution :** The Fig. 4.12 shows the construction of 5-to-32 decoder using four 74LS138s and half 74LS139. The half section of 74LS139 IC is used as a 2-to-4 decoder to decode the two higher order inputs, D and E. The four outputs of this decoder are used to enable one of

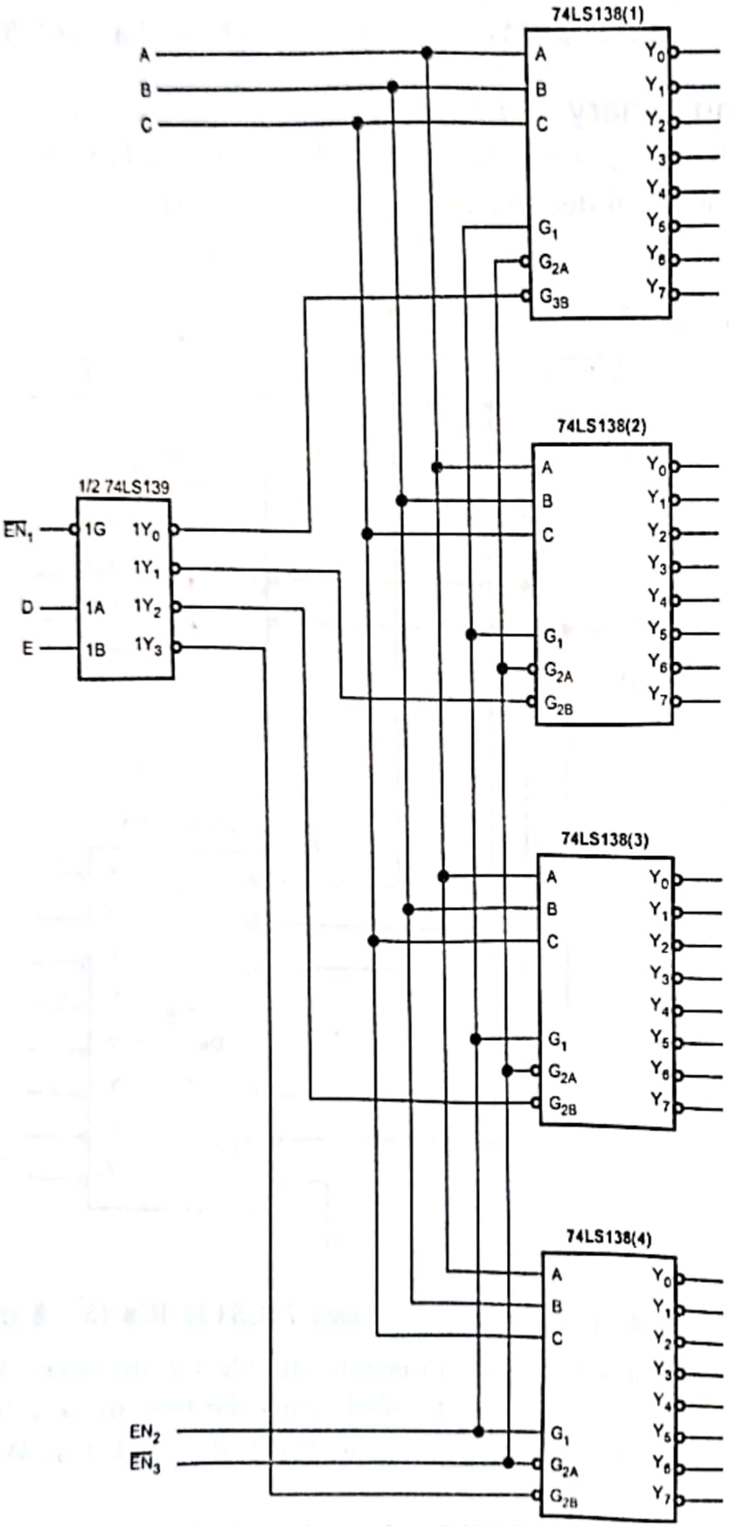


**Fig. 4.12 5-to-32 decoder using 74LS138 and 74LS139**

the four 3-to-8 decoders. The three lower order inputs A, B and C are connected in parallel to four 3-to-8 decoders. This means that the same output pin of each of the four 3-to-8 decoders is selected but only one is enabled. The remaining enable signals of four 3-to-8 decoder ICs are connected in parallel to construct enable signals for 5-to-32 decoder.

### 4.2.5 Realization of Multiple Output Function using Binary Decoder

The combination of decoder and external logic gates can be used to implement single or multiple output functions. We know that decoder can have one of the two output states; either active low or active high. Let us see the significance of these output states in the implementation of binary function.

### For Active High Output

#### SOP Function Implementation

When decoder output is active high, it generates minterms (product terms) for input variables; i.e. it makes selected output logic 1. In such case to implement SOP function we have to take sum of selected product terms generated by decoder. This can be implemented by ORing the selected decoder outputs, as shown in the Fig. 4.13 (a). The Fig. 4.13 (a) shows the implementation of function $f = \sum M(1, 2, 3, 7)$ using 3 : 8 decoder with active high outputs.

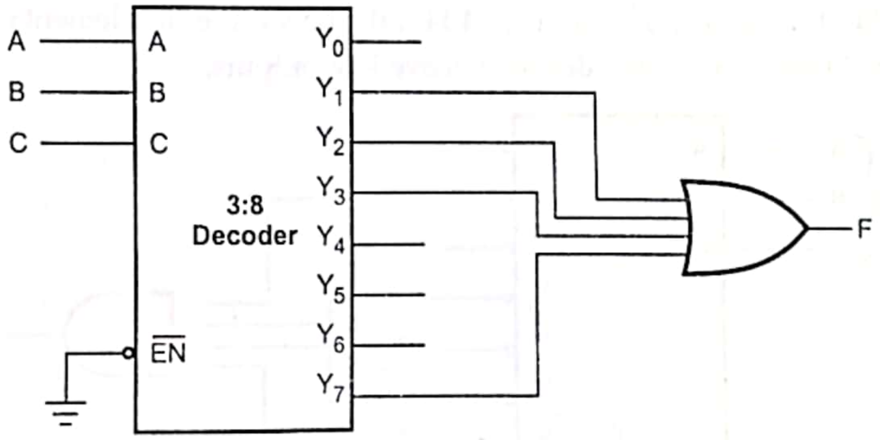


**Fig. 4.13 (a) Single output function implementation using decoder and gate**

#### POS Function Implementation

When decoder output is active high, we can implement the POS function in similar manner as for SOP function except function output is complemented. This can be achieved by connecting NOR gate instead of OR gate. This is illustrated in Fig. 4.13 (b). The Fig. 4.13 (b) shows the implementation of function $f = \pi M(1, 3, 5, 7)$ using 3 : 8 decoder with active high outputs.

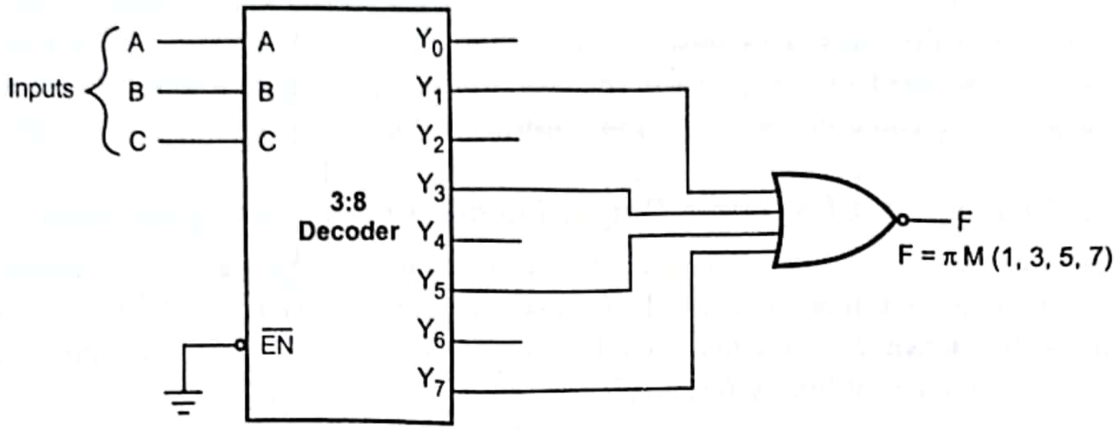


Fig. 4.13 (b) Implementation of POS function using decoder (with active high outputs) and gate

## For Active Low Output

### POS Function Implementation

When decoder output is active low, the output is in complemented form, i.e., it generates maxterms (sum terms) for input variables. It makes selected output logic 0. In such case to implement POS function we have to take product of selected sum terms generated by decoder. This can be achieved by ANDing the selected decoder outputs, as shown in the Fig. 4.14 (a). The Fig. 4.14 (a) shows the implementation of function $f = \pi M(1,3,5,7)$ using 3 : 8 decoder with active low outputs.

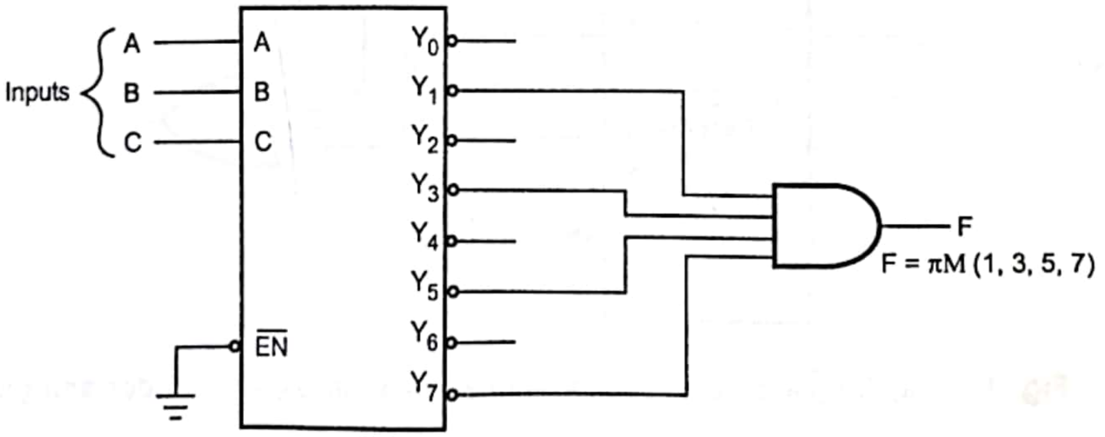


Fig. 4.14 (a) Implementation of POS function using decoder (with active low ouputs) and gate

### SOP Function Implementation

When decoder output is active low, we can implement the SOP function in similar manner as for POS function except function output is complemented. This can be achieved by connecting NAND gate instead of AND gate. This is illustrated in Fig. 4.14 (b). The Fig. 4.14 (b) shows the implementation of function $f = \sum m(1,2,5,7)$ using 3 : 8 decoder with active low outputs.

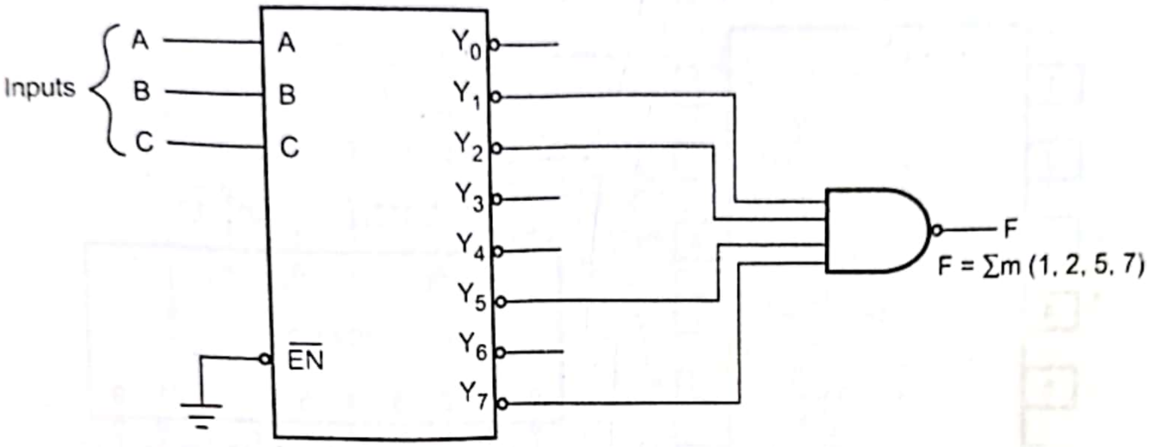


**Fig. 4.14 (b) Implementation of SOP function using decoder (with active low outputs) and gate**

**Example 4.5 :** *Implement following multiple output function using 74LS138 and external gates.* $F_1(A, B, C) = \sum m(1, 4, 5, 7)$ $F_2(A, B, C) = \pi M(2, 3, 6, 7)$

**Solution :** In this example, we use IC 74LS138, 3 : 8 decoder to implement multiple output function. The outputs of 74LS138 are active low, therefore, SOP function (function $F_1$) can be implemented using NAND gate and POS function (function $F_2$) can be implemented using AND gate, as shown in Fig. 4.15.

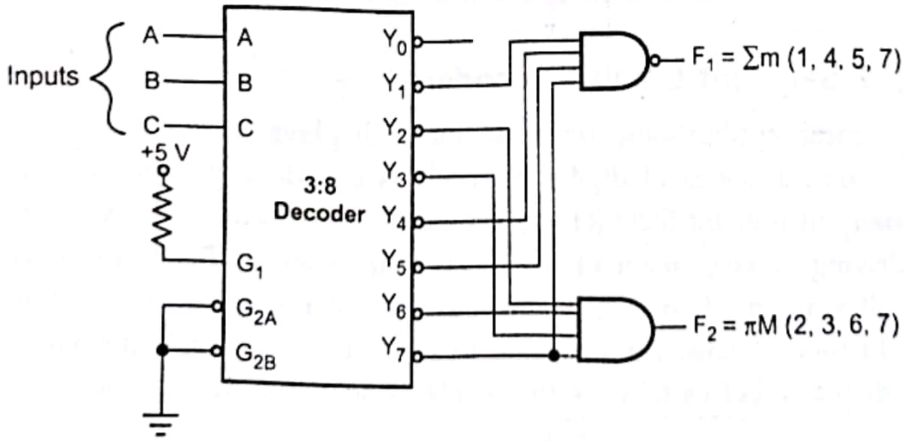
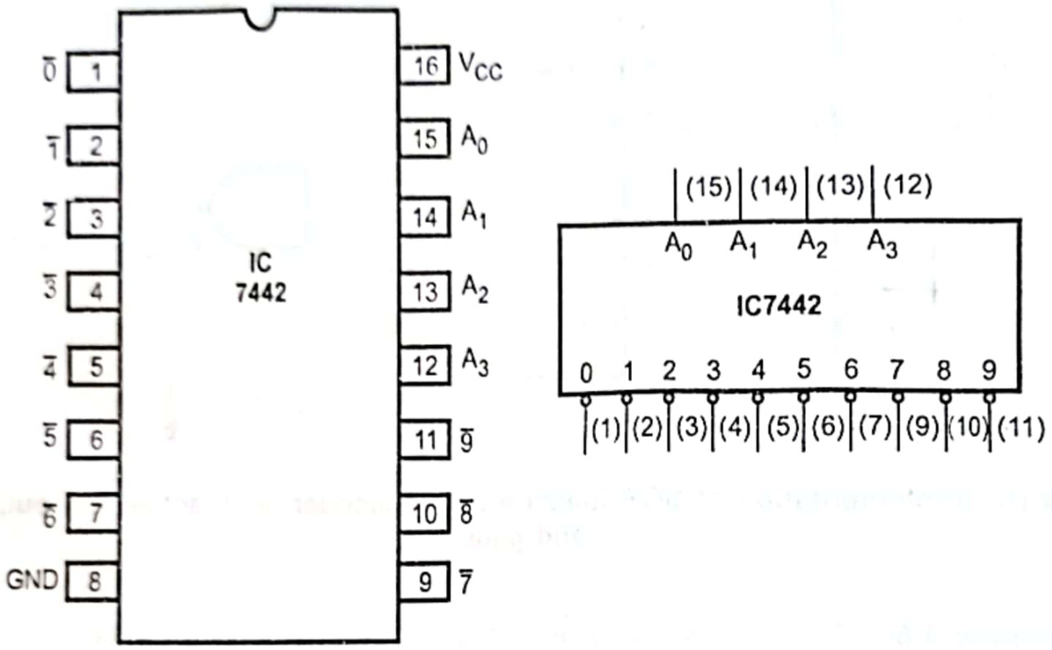


**Fig. 4.15**

### 4.2.6 BCD to Decimal Decoder

BCD decoders have four inputs and 10 outputs. The four-bit BCD input is decoded to activate one of the ten outputs. The 74XX42 is a BCD to decimal decoder. It accepts four active high BCD inputs and provides 10 independent active low outputs. Fig. 4.16 (a) shows pin diagram and logic symbol for IC 7442. The active low outputs of IC 7442 facilitates addressing other MSI units with active low input enables.



**(a) Pin diagram** **(b) Logic symbol**

**Fig. 4.16**

The logic design of 7442 ensures that all outputs are high when binary codes greater than nine are applied to the inputs. The most significant input, $A_3$ produces a useful inhibit function when the 7442 is used as a 3 to 8 decoder.

### 4.2.7 BCD to 7 Segment Display Decoder

In most practical applications, seven segment displays are used to give a visual indication of the output states of digital ICs such as decade counters, latches etc. These outputs are usually in four bit BCD (binary coded decimal) form, and are thus not suitable for directly driving seven segment displays. The special BCD to seven segment decoder/driver ICs are used to convert the BCD signal into a form suitable for driving these displays. In this sections, we are going to study LED and LCD decoders/drivers for seven segment displays. Let us tabulate the segments activated during each digit display.

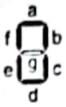
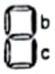
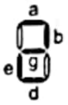
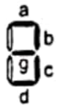
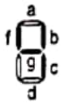
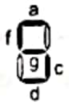
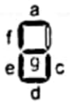
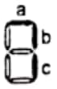
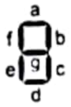
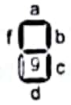
| Digit | Segments Activated | Display |
|---|---|---|
| 0 | a, b, c, d, e, f |  |
| 1 | b, c |  |
| 2 | a, b, d, e, g |  |
| 3 | a, b, c, d, g |  |
| 4 | b, c, f, g |  |
| 5 | a, c, d, f, g |  |
| 6 | a, c, d, e, f, g |  |
| 7 | a, b, c |  |
| 8 | a, b, c, d, e, f, g |  |
| 9 | a, b, c, d, f, g |  |

Table 4.6

From the above Table 4.6 we can determine the truth table for BCD-to-7 segment decoder/driver. This truth table also depends on the construction of 7-segment display. If 7-segment display is common anode, the segment driver output must be active low to glow the segment. In case of common cathode type 7-segment display, the segment driver output must be active high to glow the segment. Table 4.7 and 4.8 show the truth tables for both BCD to 7 segment decoder/driver with common cathode display and with common anode display respectively.

| Digit | A | B | C | D | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 3 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 5 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 7 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

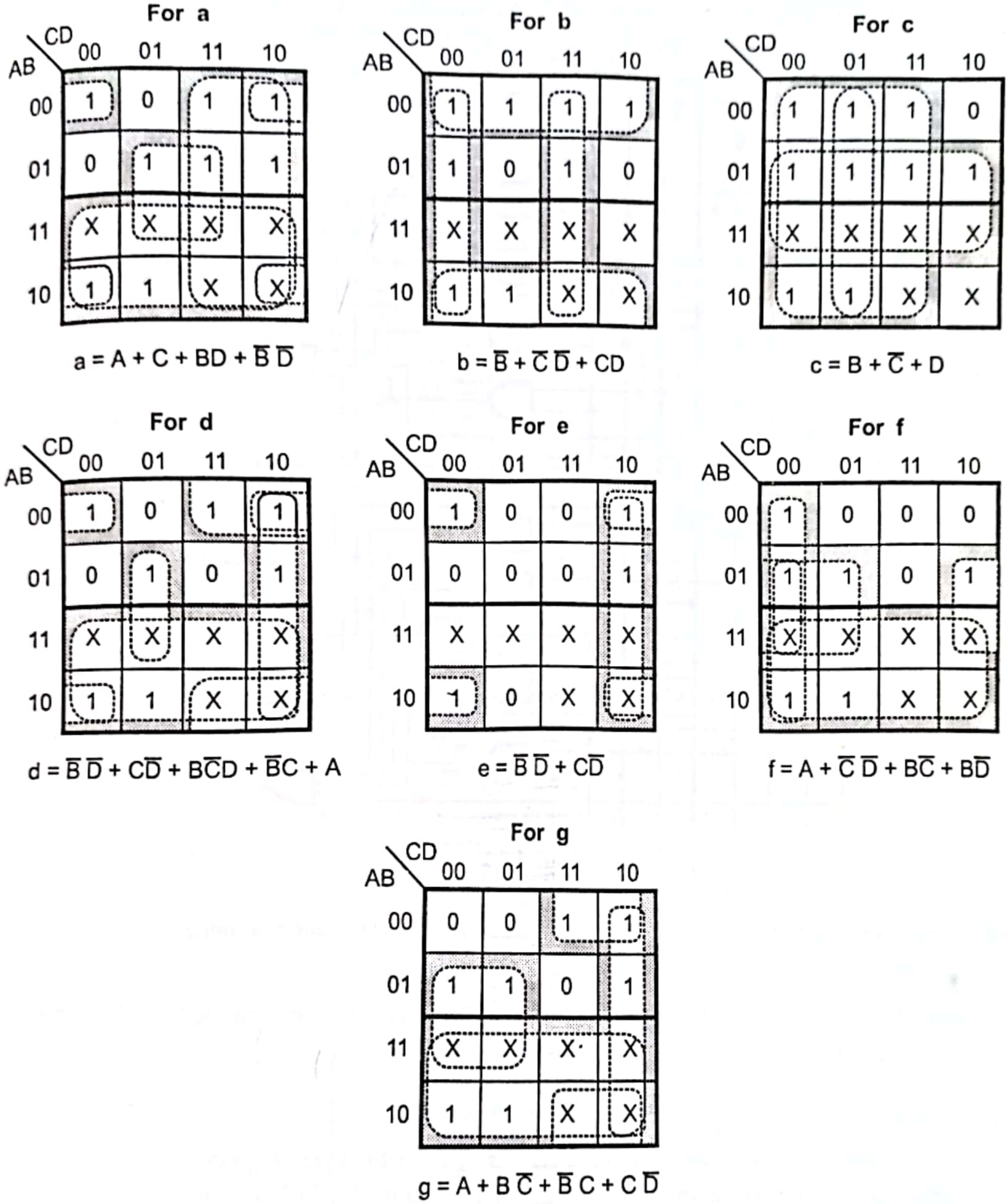
**Table 4.7 Truth table for BCD-to-common-cathode 7-segment decoder/driver**

| Digit | A | B | C | D | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

**Table 4.8 Truth table for BCD-to-common-anode 7-segment decoder/driver**

Let us design the combinational circuit for common cathode 7-segment display/driver.

K-map simplification



For a

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 1 | 1 |
| 01 | 0 | 1 | 1 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$a = A + C + BD + \bar{B}\,\bar{D}$

For b

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 0 | 1 | 0 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$b = \bar{B} + \bar{C}\,\bar{D} + CD$

For c

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 1 | 1 | 0 |
| 01 | 1 | 1 | 1 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$c = B + \bar{C} + D$

For d

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 1 | 1 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$d = \bar{B}\,\bar{D} + C\bar{D} + B\bar{C}D + \bar{B}C + A$

For e

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 0 | 1 |
| 01 | 0 | 0 | 0 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 0 | X | X |

$e = \bar{B}\,\bar{D} + C\bar{D}$

For f

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 0 | 0 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$f = A + \bar{C}\,\bar{D} + B\bar{C} + B\bar{D}$

For g

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 1 | 1 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | X | X | X | X |
| 10 | 1 | 1 | X | X |

$g = A + B\,\bar{C} + \bar{B}\,C + C\,\bar{D}$

Fig. 4.17

**Logic diagram**

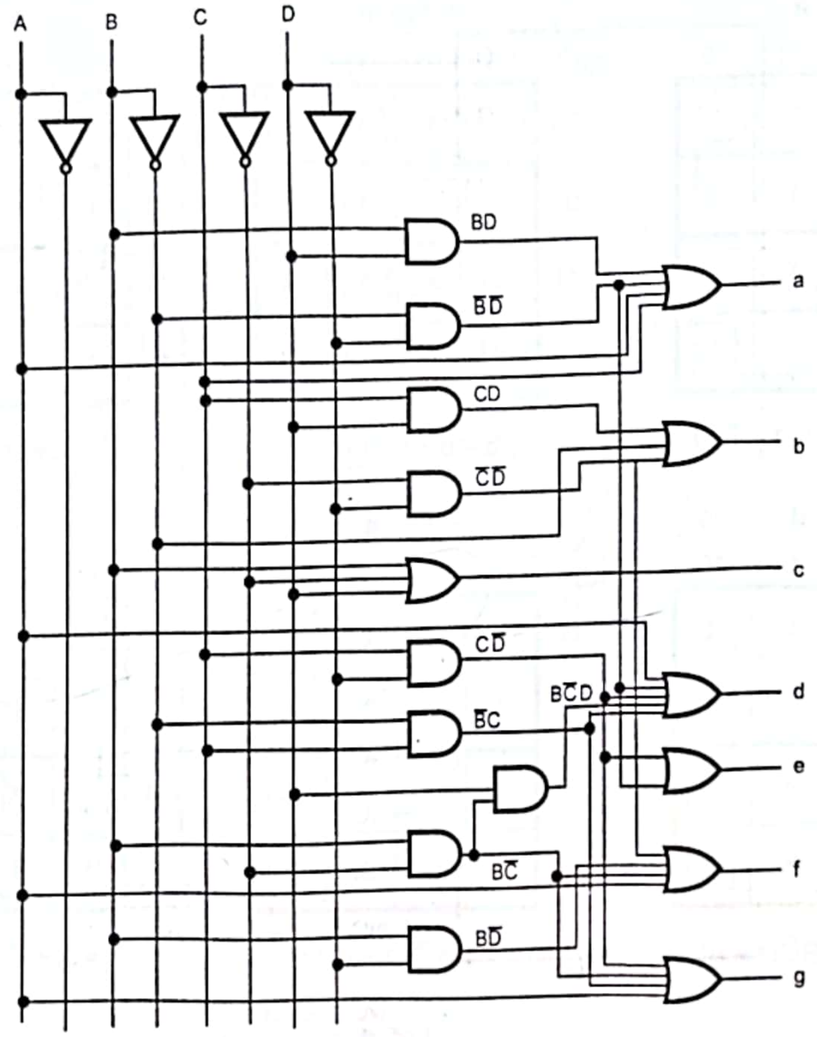


Fig. 4.18

**Example 4.6 :** *Implement BCD-to-7 segment decoder for common anode using 4 : 16 decoder.*

**Solution :** Referring the truth table from Table 4.7 we can implement BCD-to-7 segment decoder for common anode as shown in the Fig. 4.19.

### 4.2.7.1 Basic Connection for Driving 7-Segment Displays

Fig. 4.20 and 4.21 show the basic connections of BCD to seven segment decoder/driver for common-anode and common-cathode displays, respectively. In both the circuits, current limiting resistors are placed in series with each display segment. Looking at the figures, we can observe that common anode decoder/driver sinks current whereas common-cathode decoder/driver source the current to each display segment.

Now we will see the practical decoder/driver ICs, their pin connections, functional description and features.

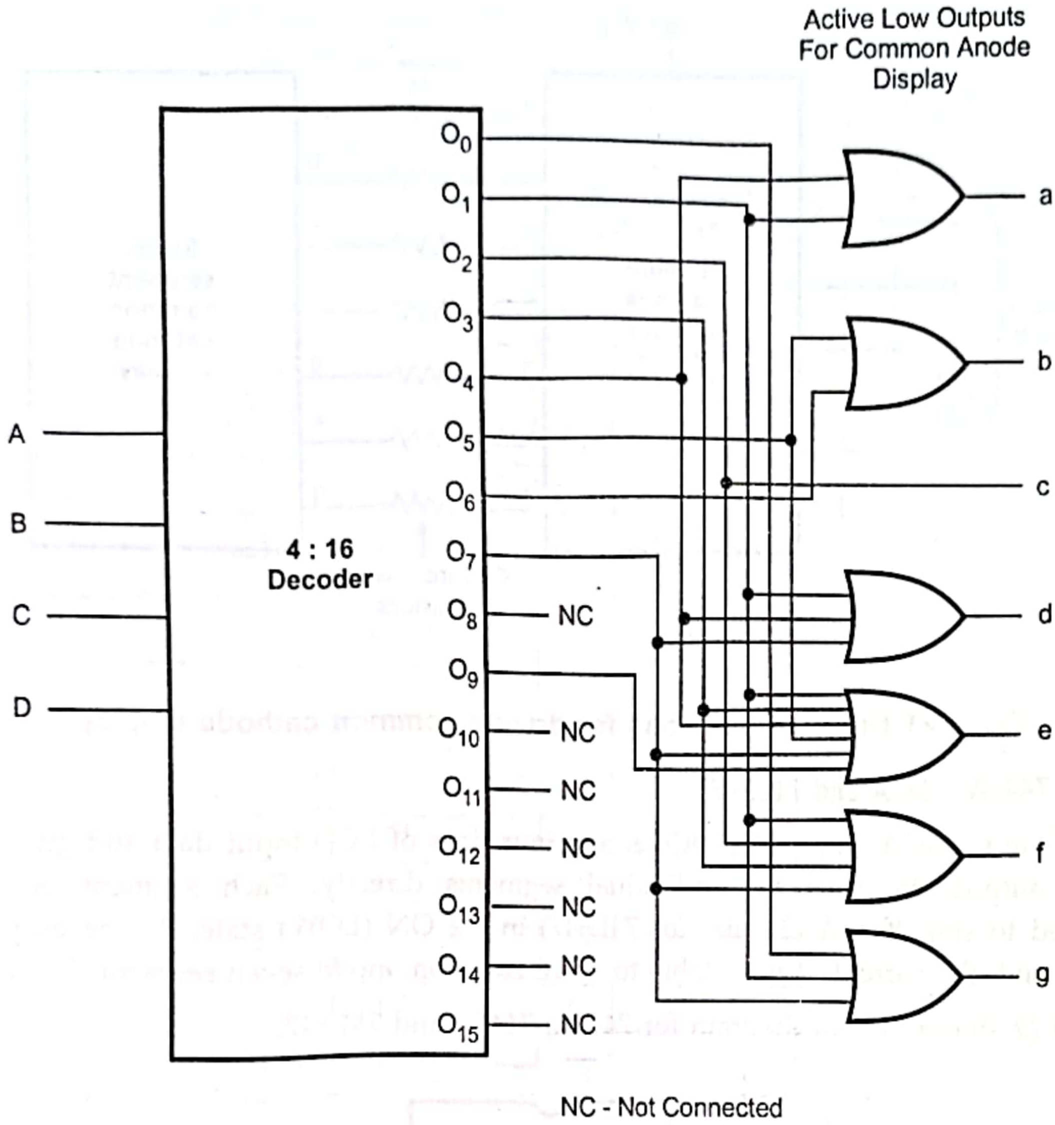


Fig. 4.19 BCD-to-7 segment decoder using 4 : 16 decoder

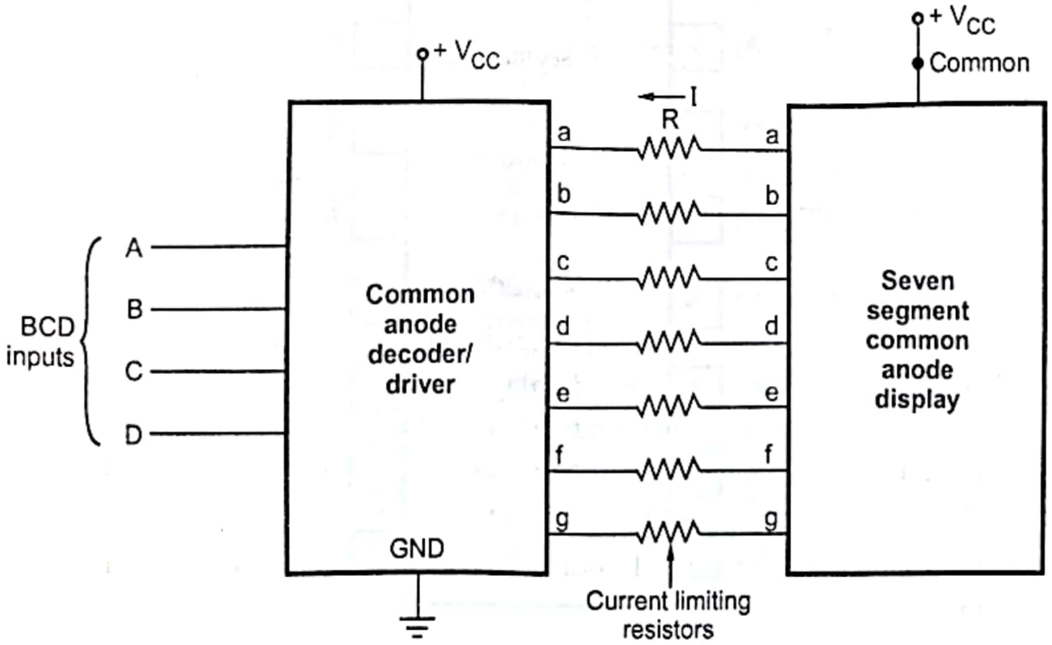


Fig. 4.20 Basic connections for driving common anode display

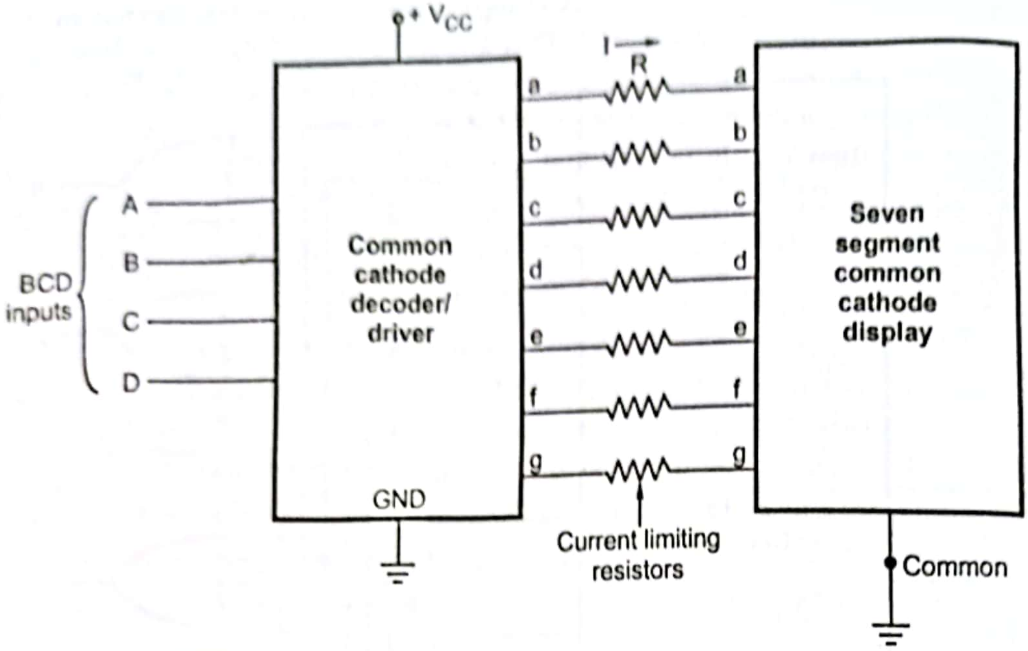

+ $V_{CC}$
BCD inputs
A
B
C
D
Common cathode decoder/ driver
GND
I
R
a
b
c
d
e
f
g
Current limiting resistors
Seven segment common cathode display
Common

Fig. 4.21 Basic connections for driving common cathode display

### 4.2.7.2 IC 7446A, 7447A and 74LS47

The 7446A, 7447A and 74LS47 ICs accept four lines of BCD input data and gives open collector outputs to drive the individual segments directly. Each segment output is guaranteed to sink 40 mA (24 mA for 74LS47) in the ON (LOW) state. As the outputs of these ICs sink the current, it is suitable to drive common anode seven segment displays.

Fig. 4.22 shows the pin diagram for 7446A, 7447A and 74LS47.

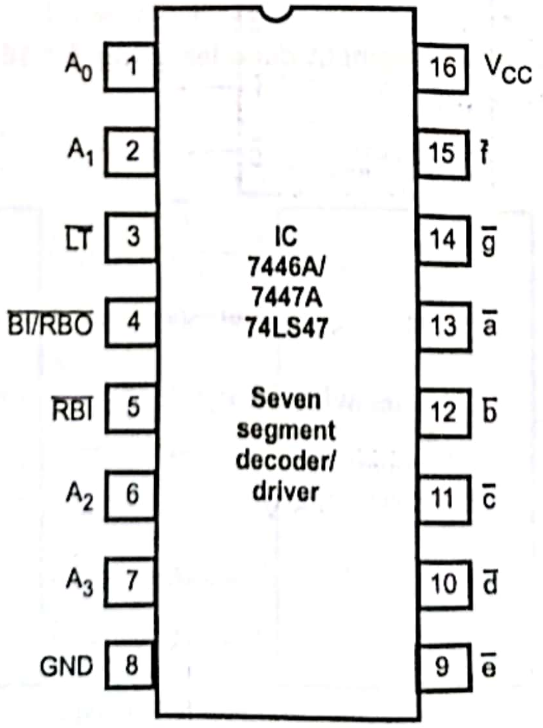


Fig. 4.22 Pin diagram for 7446A, 7447A and 74LS47

Pins $A_0$, $A_1$, $A_2$ and $A_3$ represent BCD inputs with $A_0$ as a least significant bit (LSB) and $A_3$ as a most significant bit (MSB). Pins $\bar{a}$ through $\bar{g}$ are the seven segment outputs. These are active low outputs, i.e. when segment output goes low (active state), segment is made 'ON'. The test lamp pin ($\overline{LT}$) is provided to test whether all segments are working properly or not. When $\overline{LT}$ pin is held low with $\overline{RBO}$ pin open or at logic high, IC drives all display terminals ON (active low). When the $\overline{BI/RBO}$ pin is pulled low, all outputs are blanked ; this pin also functions as a ripple-blanking output terminal. $\overline{BI/RBO}$ along with RBI can be used to provide ripple blanking feature discussed later.

Fig. 4.23 shows a circuit to drive a single, seven segment, common anode LED display. For common anode, when anode is connected to positive supply, a low voltage is applied to a cathode to turn it on. Here, BCD to seven segment decoder, IC 7447 is used to apply low voltages at cathodes according to BCD input applied to 7447. To limit the current through LED segments, resistors are connected in series with the segments. This circuit connection is referred to as a static or non-multiplex display because current is being passed through the display at all times.

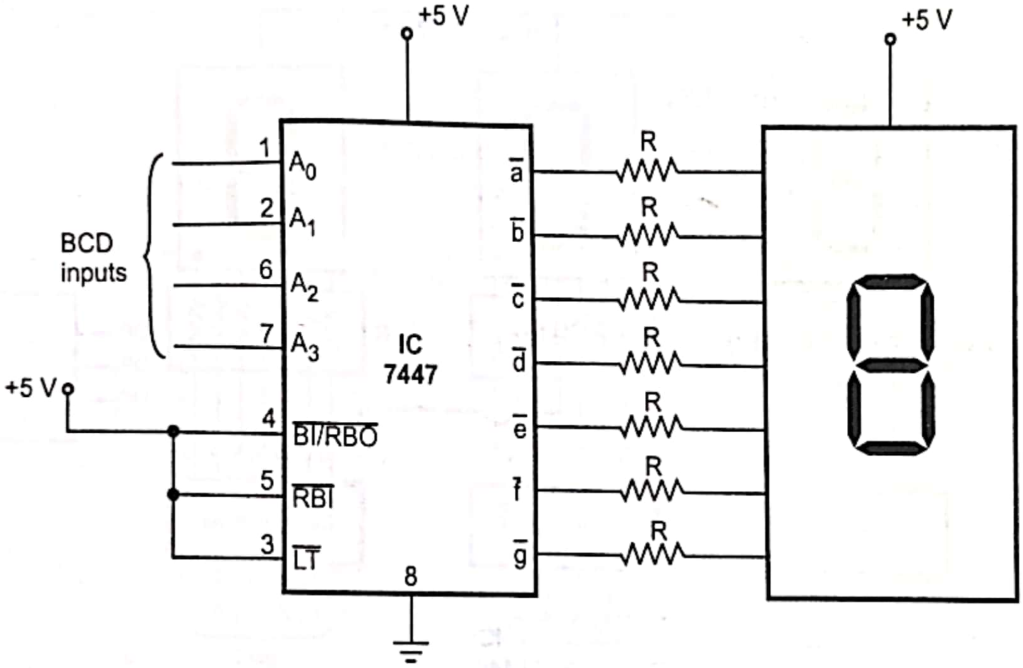


**Fig. 4.23 Circuit for driving single seven segment display using 7446/7447**

The value of the resistor in series with the segment can be calculated as follows :

We know, $V_{CC}$ – drop across LED segment – IR = 0

Drop across LED segment is nearly 1.5 V.

$$\begin{aligned} IR &= V_{CC} - 1.5 \text{ V} \\ &= 5 - 1.5 \text{ V} \\ &= 3.5 \text{ V} \end{aligned}$$

Each LED segment requires a current of between 5 and 30 mA to light. Let us assume that current through LED segment is 15 mA

$$R = \frac{3.5\text{ V}}{1.5\text{ mA}}$$

$$= 233\ \Omega$$

In practice, the voltage drop across the LED and the output of 7447 are not exactly predictable and the exact current through the LED is not critical as long as we do not exceed its maximum current rating. Therefore, a standard value 220 Ω can be used.

**(a) Cascaded Non Multiplexed Displays**

Several sets of seven segment displays and associated decoder/drivers ICs can be cascaded to make multi-digit display system. Fig. 4.24 shows the connection for cascaded multi-digit display system.

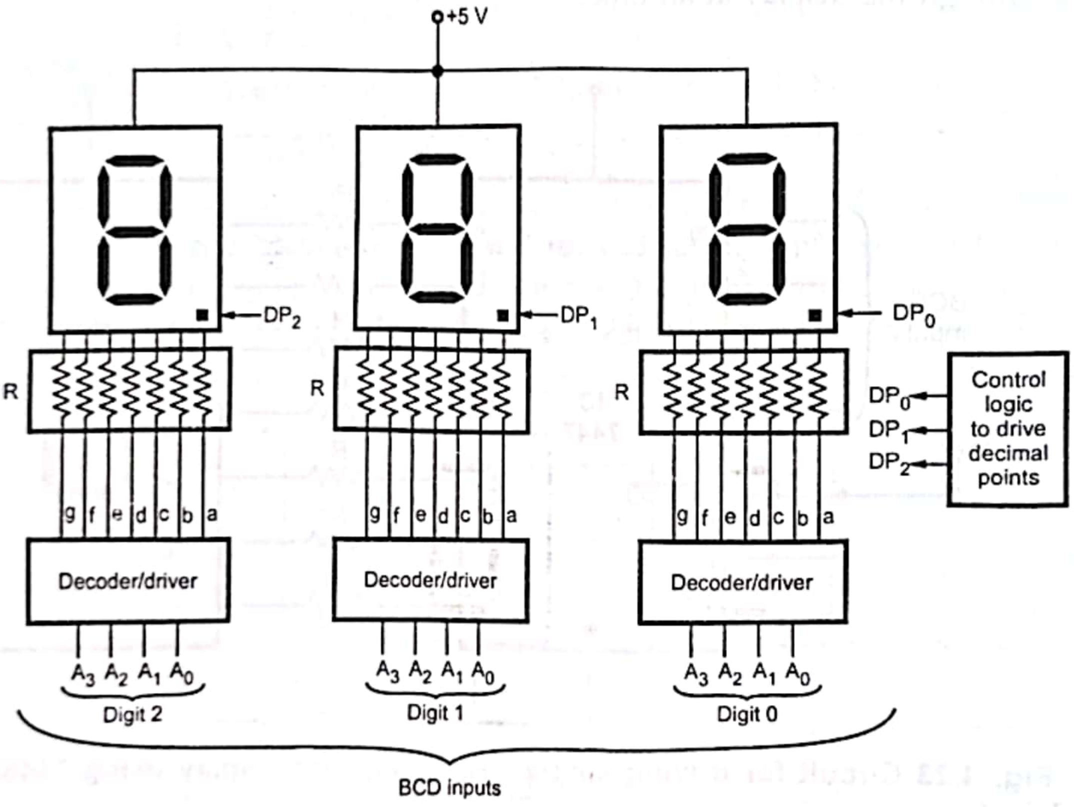


**Fig. 4.24 Cascaded multi-digit non multiplexed display system**

**(b) Ripple Blanking in Multi-digit Displays**

If the display system shown in Fig. 4.24 is used to display count of 8, then it actually gives the reading 008. Similarly, if it is used to display 0.2, it actually gives reading 0.20. These leading and trailing zeroes can be automatically blanking using $\overline{RBI}$ and $\overline{RBO}$ signals of the IC 7447 decoder/driver. The technique of blanking leading and trailing zeroes is called **ripple blanking**.

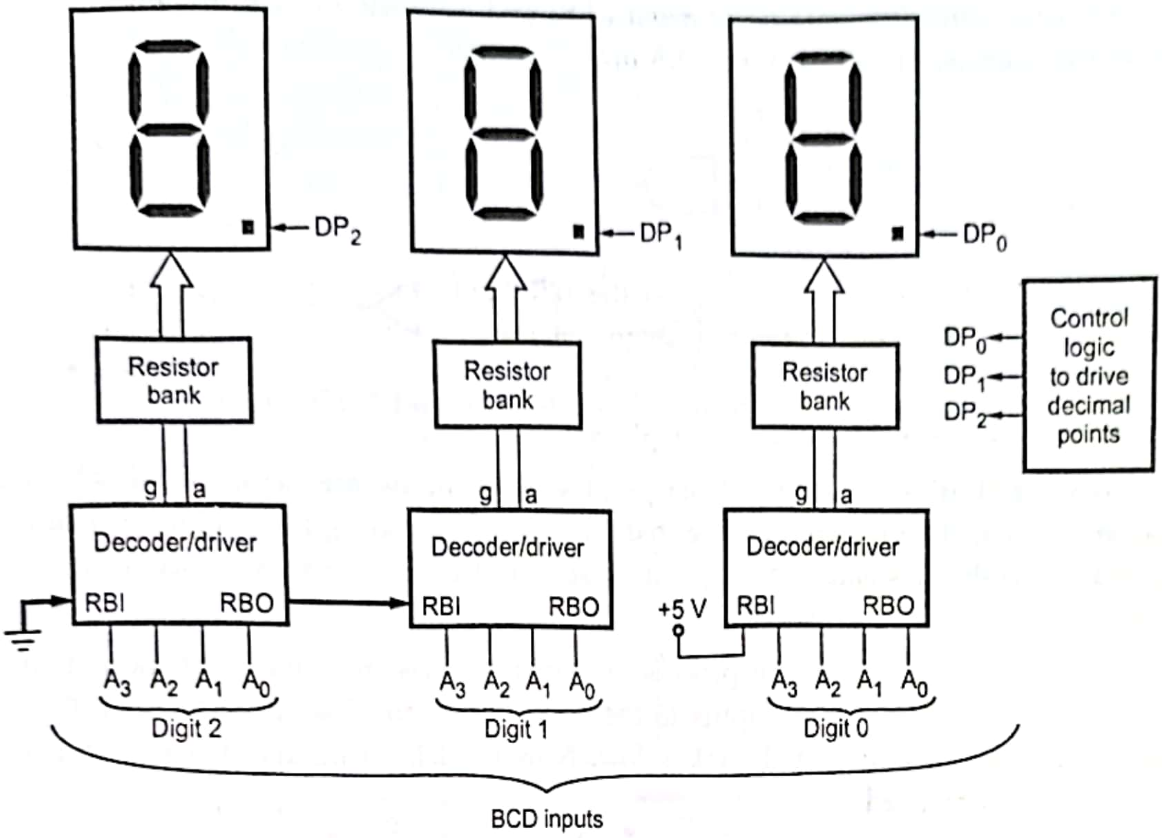


**Fig. 4.25 Circuit for blanking leading zeroes**

Fig. 4.25 and 4.26 show circuits to provide ripple blanking for leading zeroes and trailing zeroes.

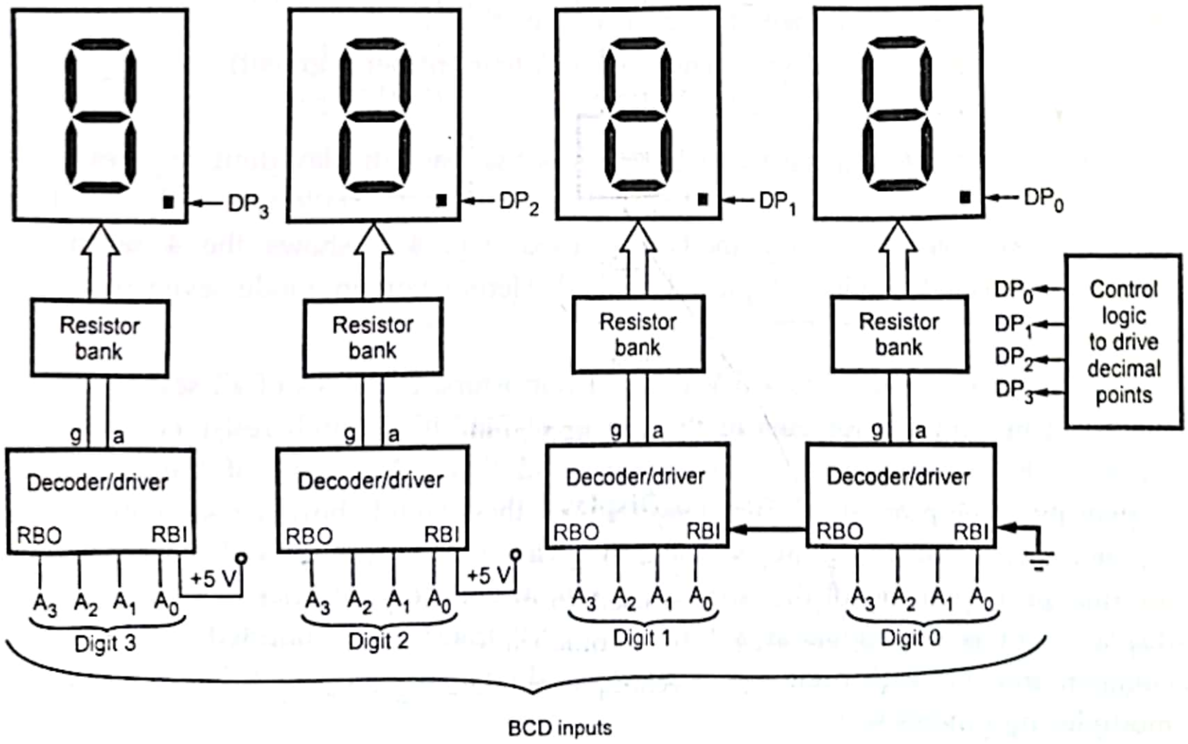


**Fig. 4.26 Circuit for blanking trailing zeroes**

To understand the working of signals RBI and RBO we will see the internal circuit for these two signals, as shown in Fig. 4.27.

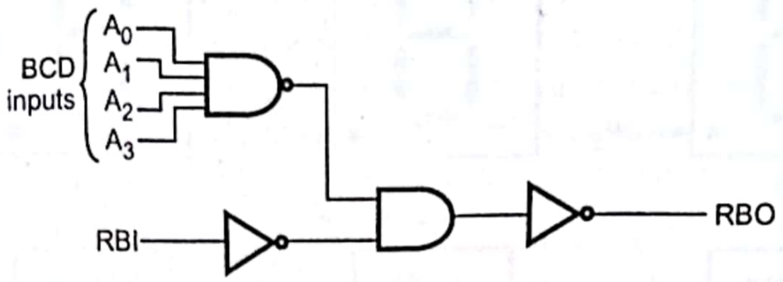


**Fig. 4.27 Internal circuit for RBI and RBO signals**

When RBI is low (activated) and all the BCD inputs are zero, then RBO goes low. Looking at Fig. 4.25 we can observe that when there is leading zero, digit 2 is blanked and it sets RBI zero for adjacent digit (digit 1), so that if BCD inputs for adjacent digit (digit 1) are zero, the digit is blanked.

In case of trailing zero the process of blanking starts from the right most digit. Here, digit 0 is blanked if all BCD inputs to the digit 0 are zero. If so, RBO for digit 0 goes low and therefore RBI input for digit 1 is low. Now if all BCD inputs of the digit 1 are zero then digit 1 is blanked.

### (c) Multiplexed Displays

Untill now we have seen static or non-multiplexed display circuits. These circuits work well for driving up to four LED digits. However, these circuits are not suitable for driving more LED digits, say 8 digits. When there are more number of digits, the first problem is a power consumption. For worst-case calculations, assume that all eight digits with all segments are lit. Therefore, worst case current required is

$$I = 8 \text{ (digits)} \times 7 \text{ (segments)} \times 15 \text{ mA (current per segment)}$$
$$= 840 \text{ mA}$$

A second problem of the static approach is that each display digit requires a separate BCD to 7 segment decoder. To solve the problems of the static non-multiplexed display approach multiplexed display method is used. Fig. 4.28 shows the 4 seven segment displays connected using multiplexed method. Here, common anode seven segment LEDs are used.

Anodes are connected to + 5 V through transistors. Cathodes of all seven segments are connected in parallel and then to the output of 7447 IC through resistors. Looking at the Fig. 4.27, the question may occur in our mind that, "Aren't all of the digits going to display the same number ?" The answer is that they would show the same number only if all the digits are turned-on at the same time. However, in multiplexed display the segment information is sent for all digits on the common lines (output lines of 7447), but only one display digit is turned on at a time. The PNP transistors connected in series with the common anode of each digit acts as an ON and OFF switch for that digit. Here is how the multiplexing process works.

The BCD code for digit 0 is first applied to the 7447. The 7447, BCD to seven segment decoder outputs the corresponding seven segment code on the segment bus lines. The transistor $Q_0$ connected to digit 0 is then turned on by corresponding control signals. All of the rest of the control lines are made high to ensure no other digits are turned on. After 2 ms, digit 0 is turned-off by making all control inputs high. The BCD code for digit 1 is then applied and the control input for digit 1 is made low to turn it ON. After next 2 ms, digit 1 is turned-off and the process is repeated for digit 2 and digit 4. After completion of turn for each digit, all the digits are lit again in turn.

With 4 digits and 2 ms per digit we get back to digit 1 every 8 ms or about 125 times a second. This refresh rate is fast enough that, to our eye and due to persistence of all digits will appear to be lit all the time.

In multiplexed display, the segment current is kept in between 40 mA to 60 mA so that they will appear as bright as they would, if not multiplexed. Even with this increased segment current, multiplexing gives a large saving in power and hardware components.

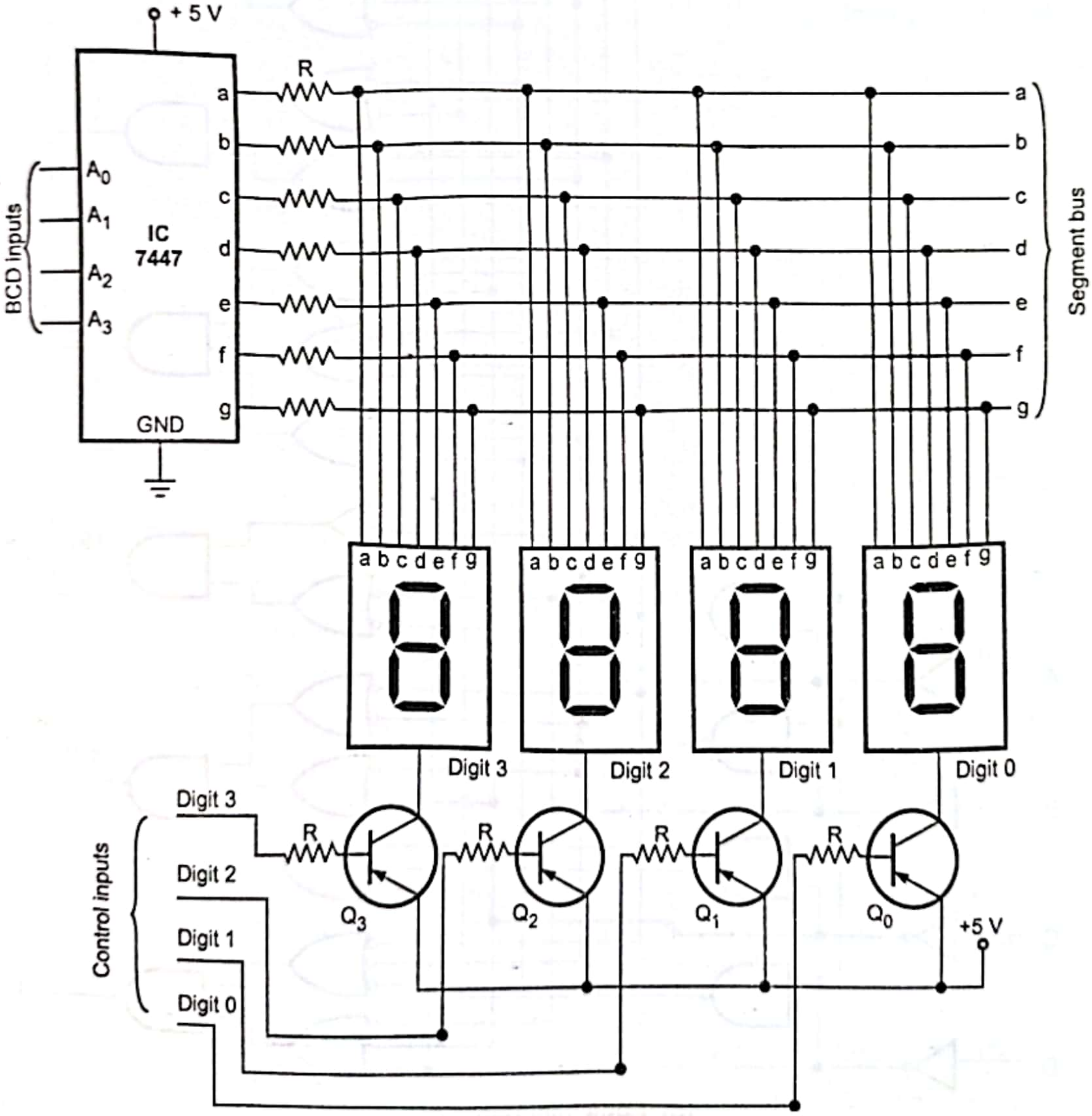
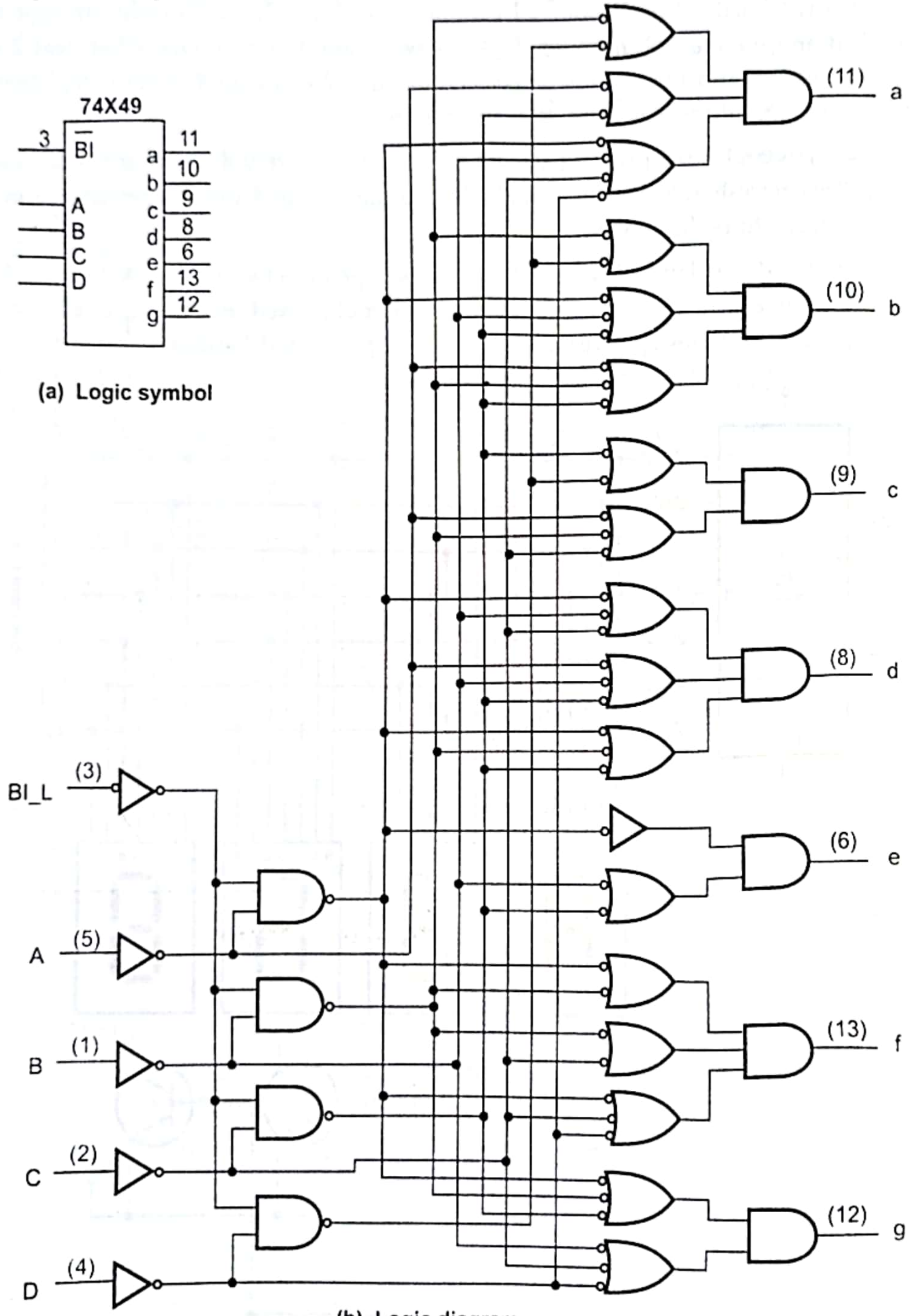


Fig. 4.28 Seven segment display in multiplexed connection

### 4.2.7.3 IC 74X49 Seven Segment Decoder

The Fig. 4.29 shows the pinouts and the logic diagram for IC 74X49, 4-bit BCD to seven segment decoder. It has active-low blanking $\overline{BL}$ input. When $\overline{BL}$ input is activated, the outputs are logic 0. The Table 4.9 shows the truth table for IC 74X49.



(a) Logic symbol

(b) Logic diagram

**Fig. 4.29 The 74X49 Seven segment decoder**

| Inputs | | | | | Outputs | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\overline{BI}$ | D | C | B | A | a | b | c | d | e | f | g |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Table 4.9 Truth Table for IC 74X49**