The three boolean expression of G_3, G_2, G_1 are realized using gates.

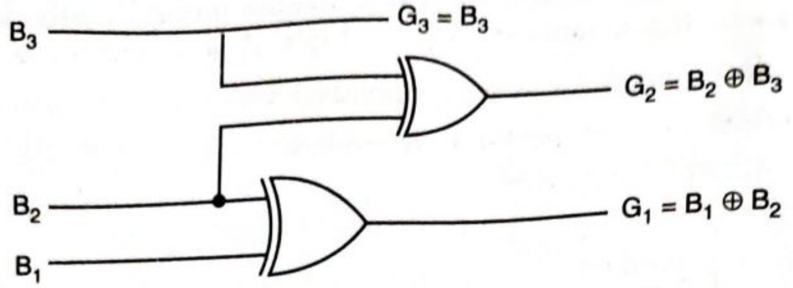


Fig. 3.51. Binary to gray code converter logic diagram

The logic diagram can be verified manually by putting the values of B_3, B_2 and B_1 in the gates. If B_3, B_2 and B_1 is 000 then $G_3 = 0$ and G_2 is obtained by taking Ex-OR operation of B_2 and B_3, which is again '0'. It means that $G_2 = 0$. The value of G_1 is obtained by taking the EX-OR operation of B1 and B2 which produces '0' result. So G1 is zero. The values of G_3 G_2 G_1 = 000 corresponding to B_3 B_2 B_1 = 000 matched with the truth table. Hence, the logic diagram is verified.

3.18 PRIORITY ENCODER

When feeding data/program into a computer it is possible that more than one key is pressed simultaneously. A priority function means that the encoder will give priority to]e highest order decimal digit in the inputs and ignore all other *e.g.* in a priority encoder decimals 8 and 4 are pressed together (*i.e.* both 8 and 4 inputs are high), the encoder will convert the decimal 8 to the output and ignore 4.

Decimal to BCD Priority Encoder

The logic circuit for incorporating a priority in encoding must incorporate a feature to prevent a lower order digit input from disrupting the encoding of higher order digit. This is done by using inhibit (enable) gates.

The truth table for decimal to BCD is given as :

Decimal Number	***BCD Codes***			
	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

It is seen from the truth table that Y_0 is high when inputs 1, 3, 5, 7, 9 are high. Priority can be added if digit 1 input activates gates Y_0 only if no higher digit (other than which also activate Y_0) are high. This is express as :

(1) Y_0 is high if 1 is high and 2, 4, 6, 8 are low

(2) Y_0 is high if 3 is high and 4, 6, 8 are low

(3) Y_0 is high if 5 is high and 6, 8 are low

(4) Y_0 is high if 7 is high and 8 is low

(5) Y_0 is high if 9 is high.

The 5 statements describe the priority for encoding for BCD bit Y_0. Thus Y_0 is high if any of above statements are true.

The logic equation is expressed as :

$$Y_0 = 1.\bar{2}.\bar{4}.\bar{6}.\bar{8} + 3.\bar{4}.\bar{6}.\bar{8} + 5.\bar{6}.\bar{8} + 7.\bar{8} + 9$$

The logic statement for Y_1 is given as : 2,3,6,7

(1) Y_1 is high if 2 is high and 4, 5, 8, 9 are low.

(2) Y_1 is high if 3 is high and 4, 5, 8, 9 are low.

(3) Y_1 is high if 6 is high and 8, 9 are low

(4) Y_1 is high if 7 is high and 8, 9 are low

The above statements can be expressed as

$$Y_1 = 2.\bar{4}.\bar{5}.\bar{8}.\bar{9} + 3.\bar{4}.\bar{5}.\bar{8}.\bar{9} + 6.\bar{8}.\bar{9} + 7.\bar{8}.\bar{9}$$

The output Y_2 can be described by following statements :

4567

(1) Y_2 is high if 4 is high and 8, 9 are low

(2) Y_2 is high if 5 is high and 8, 9 are low

(3) Y_2 is high if 6 is high and 8, 9 are low

(4) Y_2 is high if 7 is high and 8, 9 arelow.

It is expressed as logic equation

$$Y_2 = 4.\bar{8}.\bar{9} + 5.\bar{8}.\bar{9} + 6.\bar{8}.\bar{9} + 7.\bar{8}.\bar{9}$$

The statement for Y_3 is given as :

(1) Y_3 is high if 8 is high and 9 is low. 89

(2) Y_3 is high if 9 is high.

Logical equation is given as for Y_3 *i.e.*

$$Y_3 = 8.\bar{9} + 9 \text{ or } \quad 8 + 9 \quad [\because A\bar{B} + B = A + B]$$

Truth table realization is given as :

Inputs										*Output*			
9	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	Y_3	Y_2	Y_1	Y
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	×	0	0	0	1
0	0	0	0	0	0	0	1	×	×	0	0	1	0
0	0	0	0	0	0	1	×	×	×	0	0	1	1
0	0	0	0	0	1	×	×	×	×	0	1	0	0
0	0	0	0	1	×	×	×	×	×	0	1	0	1
0	0	0	1	×	×	×	×	×	×	0	1	1	0
0	0	1	×	×	×	×	×	×	×	0	1	1	1
0	1	×	×	×	×	×	×	×	×	1	0	0	0
1	×	×	×	×	×	×	×	×	×	1	0	0	1

Logic circuit for output bit Y_0, Y_1, Y_2, Y_3 is given as follows :

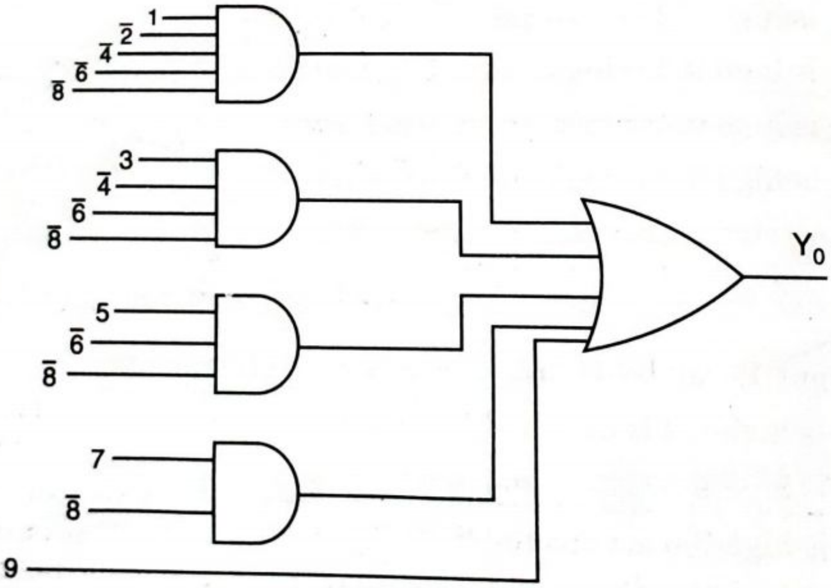


Fig. 3.52. Logic circuit for output bit Y_0 of decimal to BCD Priority encoder.

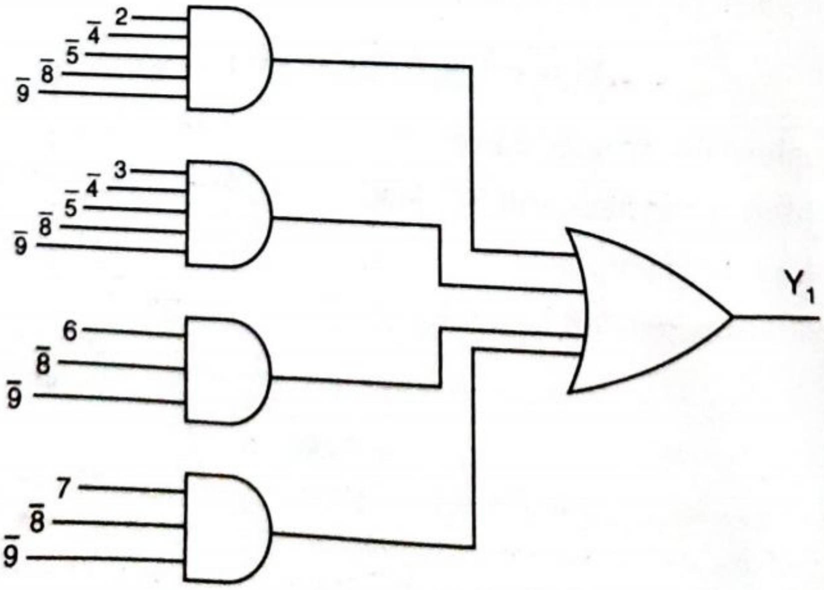


Fig. 3.53. Logic circuit for output bit Y_1.

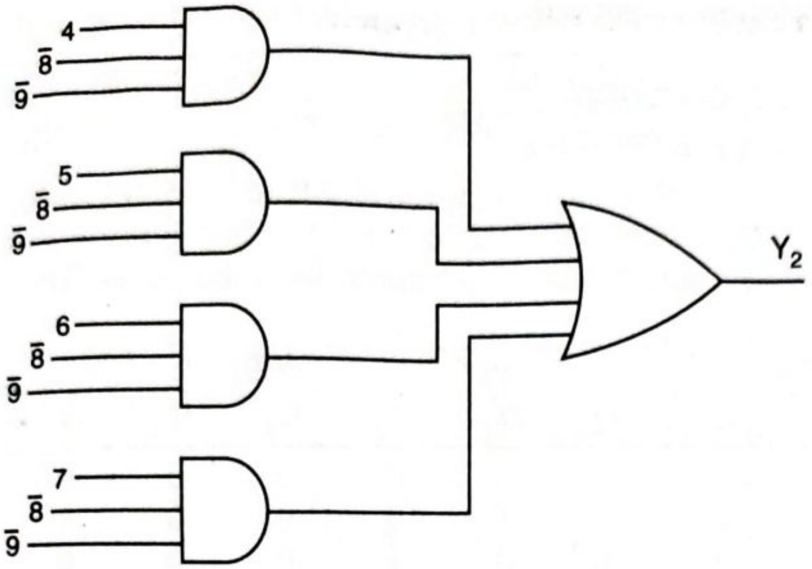


Fig. 3.54. Logic circuit for output bit Y_2.

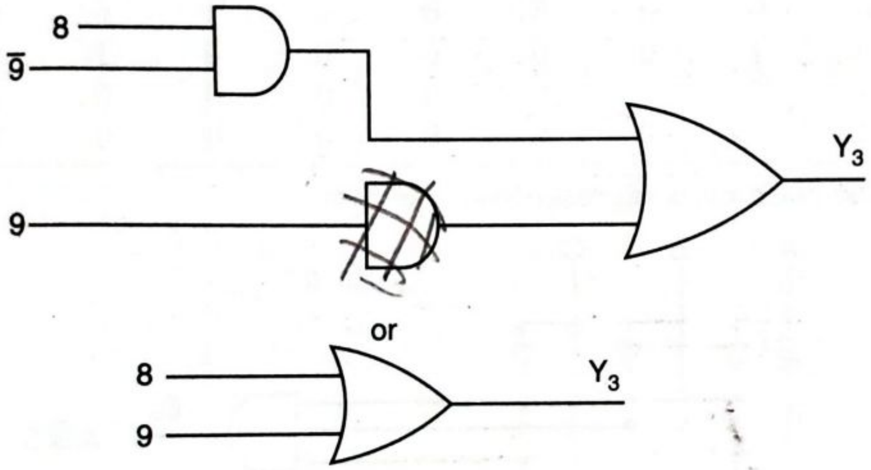


Fig. 3.55. Logic circuit output of Y_3.

3.19 DECODER

A decoder is a combinational circuit that converts an N-bit binary input code into M output lines such that only one output line is activated for each one of the possible combinations of inputs. The block diagram for decoder is shown as in figure 3.55.

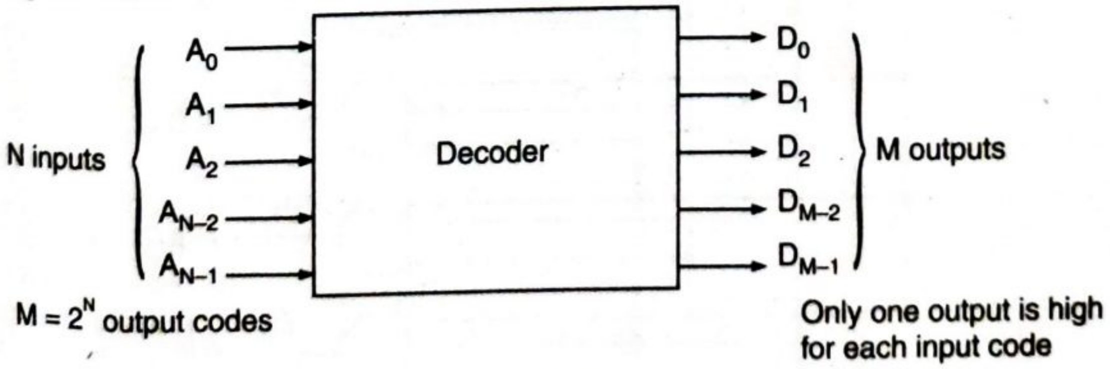


Fig. 3.56.

Since each of N inputs can be 0 or 1, there are 2^N possible input combinations or codes. For each of these input combinations, only one of M outputs will be active (High), all the other outputs will remain inactive (Low). Some decoders are designed to produce active low output, while all the other outputs remain high. Some decoders do not utilize 2^N possible input codes *e.g.* a BCD to decimal decoder.

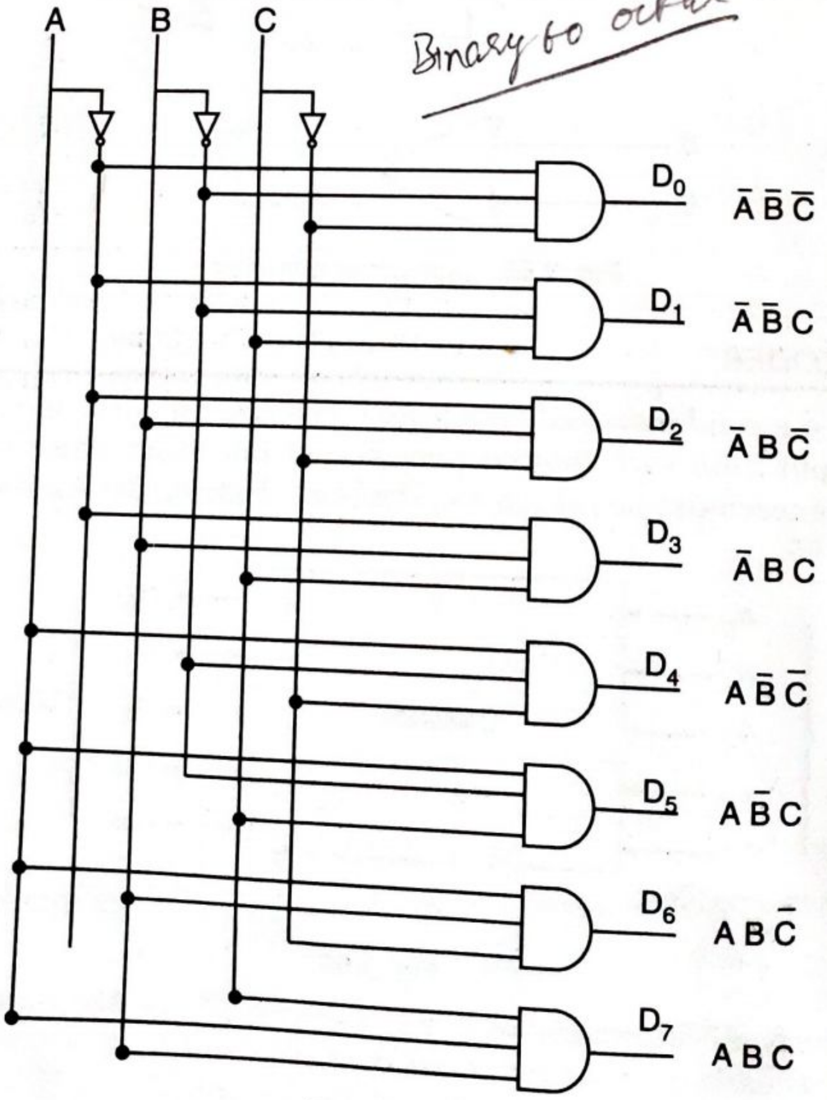
Examples of decoders are :

(1) 3 Line to 8 Line Decoder

For this decoder, it has three inputs and 8 outputs. It uses all AND gates, therefore the outputs are active high. For active low outputs NAND gates are used. This decoder is also known as binary to octal decoder. The truth table is given as :

Inputs			*Outputs*							
A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

The logic diagram is represented as follows :



Binary to octal

Fig. 3.57.

(2) BCD to Decimal Decoder 4:10

In this decoder BCD is input and decimal number is output. It is also known as 4 to 10 line decoder. The truth table is given as follows :

Input BCD Code					*Actual Output*
A_3	A_2	A_1	A_0		
0	0	0	0	———	$\overline{D_0}$
0	0	0	1	———	$\overline{D_1}$
0	0	1	0	———	$\overline{D_2}$
0	0	1	1	———	$\overline{D_3}$
0	1	0	0	———	$\overline{D_4}$
0	1	0	1	———	$\overline{D_5}$
0	1	1	0	———	$\overline{D_6}$
0	1	1	1	———	$\overline{D_7}$
1	0	0	0	———	$\overline{D_8}$
1	0	0	1	———	$\overline{D_9}$

For input combinations that are invalid for BCD, none of the outputs will be activated. In this case, we assume the output should be active low, for this we use NAND gates instead of AND gate. The circuit representation is given as follows :

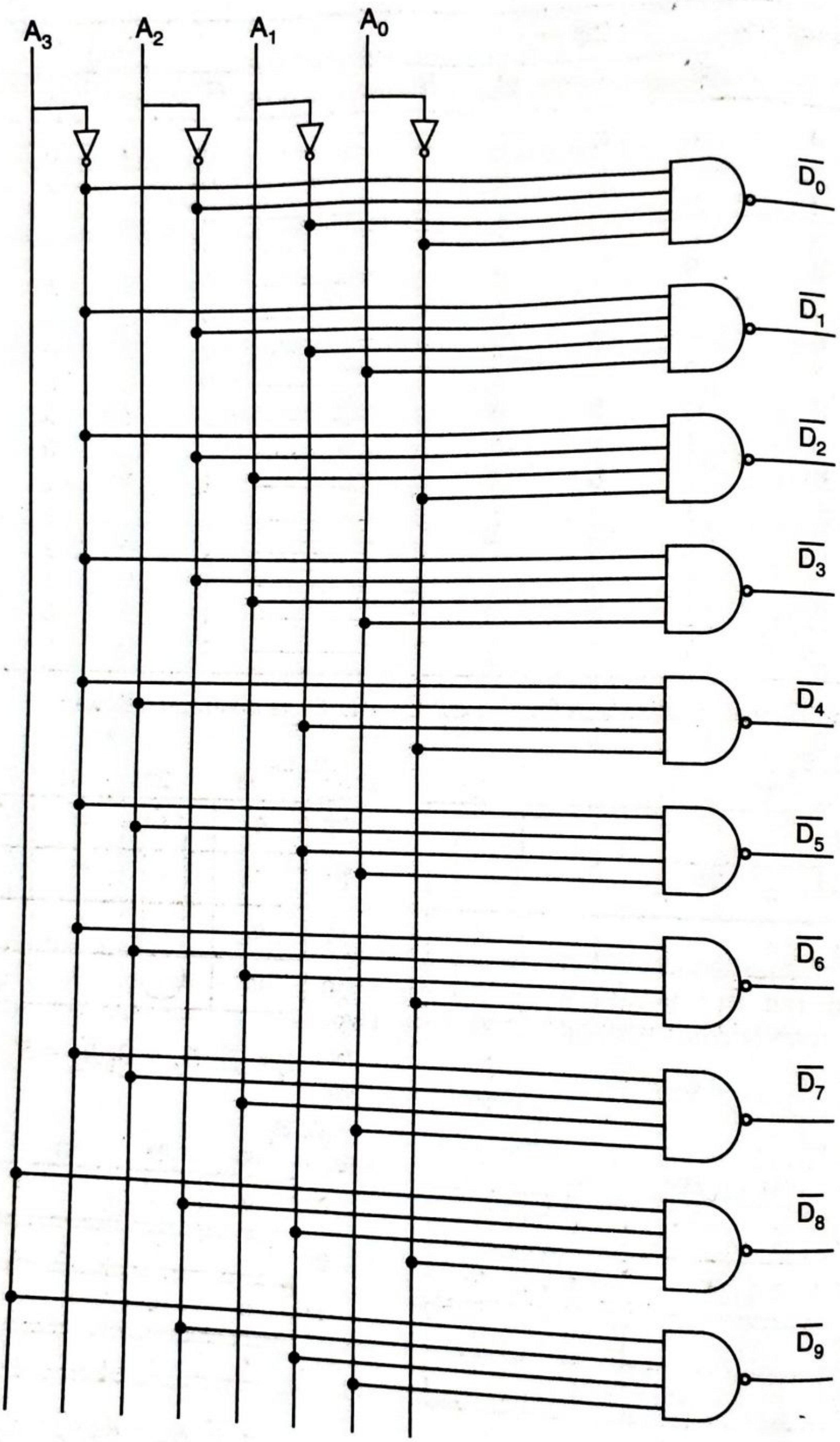


Fig. 3.58. 4 Line to 10 line decoder.

(3) Binary to Gray Code converter

The conversion table for binary to Gray Code is given as follows :

Decimal No.	Binary					Gray			
	B_1	B_2	B_3	B_4		G_1	G_2	G_3	G_4
0	0	0	0	0	———	0	0	0	0
1	0	0	0	1	———	0	0	0	1
2	0	0	1	0	———	0	0	1	1
3	0	0	1	1	———	0	0	1	0
4	0	1	0	0	———	0	1	1	0
5	0	1	0	1	———	0	1	1	1
6	0	1	1	0	———	0	1	0	1
7	0	1	1	1	———	0	1	0	0
8	1	0	0	0	———	1	1	0	0
9	1	0	0	1	———	1	1	0	1
10	1	0	1	0	———	1	1	1	1
11	1	0	1	1	———	1	1	1	0
12	1	1	0	0	———	1	0	1	0
13	1	1	0	1	———	1	0	1	1
14	1	1	1	0	———	1	0	0	1
15	1	1	1	1	———	1	0	0	0

The K-map realization for G_1, G_2, G_3 and G_4 is given as follows :

B_3B_4 \ B_1B_2	00	01	11	10
00	0 0	0 4	1 12	1 8
01	0 1	0 5	1 13	1 9
11	0 3	0 7	1 15	1 11
10	0 2	0 6	1 14	1 10

$G_1 = B_1$

B_3B_4 \ B_1B_2	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$G_2 = \overline{B_1}\, B_2 + B_2\overline{B_2} = B_1 \oplus B_2$

B_3B_4 \ B_1B_2	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1	0	0	1
10	1	0	0	1

$G_3 = \overline{B_2}\, B_3 + B_2\overline{B_3} = B_2 \oplus B_3$

B_3B_4 \ B_1B_2	00	01	11	10
00	0	0		0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$G_4 = \overline{B_3}\, B_4 + B_3\overline{B_4} = B_3 \oplus B_4$

The circuit realization of binary to Gray Code converter is given as :