Code Converters

Gauri Rao

Binary To Gray Code Converter

Gray Code system is a binary number system in which every successive pair of numbers differs in only one bit. It is used in applications in which the normal sequence of binary numbers generated by the hardware may produce an error or ambiguity during the transition from one number to the next.

For example, the states of a system may change from 3(011) to 4(100) as- 011 — 001 — 101 — 100. Therefore there is a high chance of a wrong state being read while the system changes from the initial state to the final state.
This could have serious consequences for the machine using the information. The Gray code eliminates this problem since only one bit changes its value during any transition between two numbers.

Binary to Gray Code converter Truth table

Binary				Gray Code			
b_3	b_2	b_1	b_0	g_3	g_2	g_1	g_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Let be the bits representing the binary numbers, where b0 is the LSB and b3 is the MSB, and
Let be the bits representing the gray code of the binary numbers, where g0 is the LSB and g3 is the MSB.

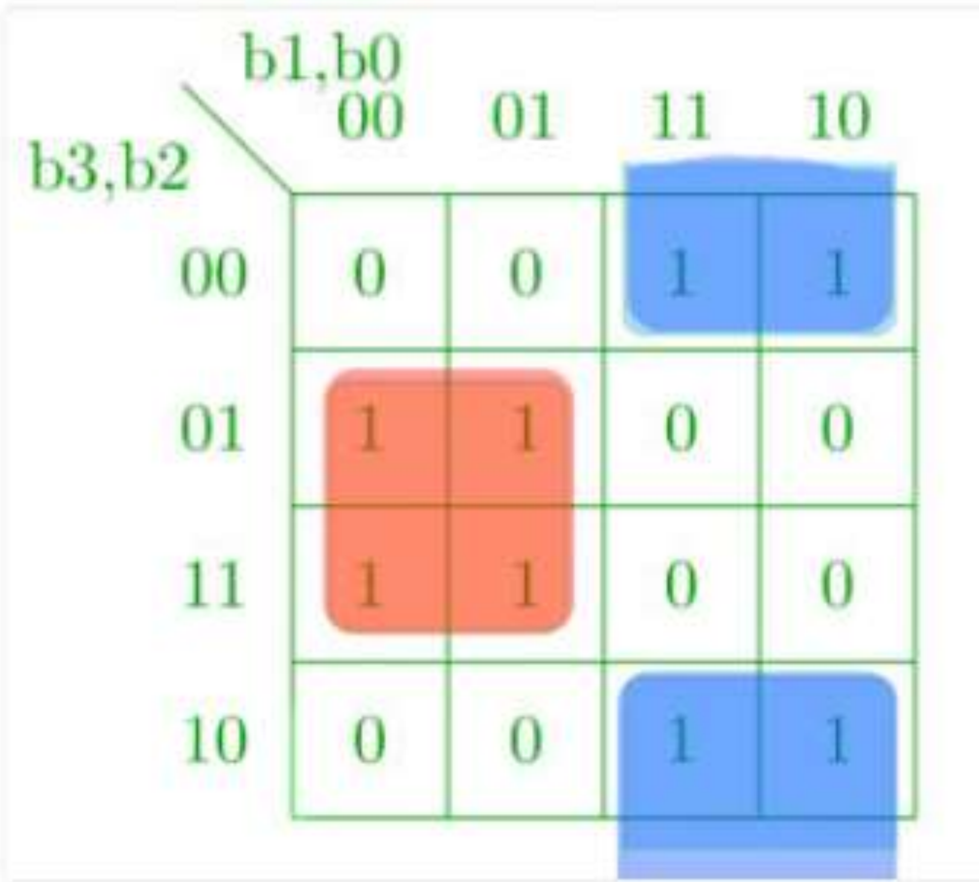
To find the corresponding digital circuit, we will use the K-Map technique for each of the gray code bits as output with all of the binary bits as input.

K map for g0

b3,b2 \ b1,b0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

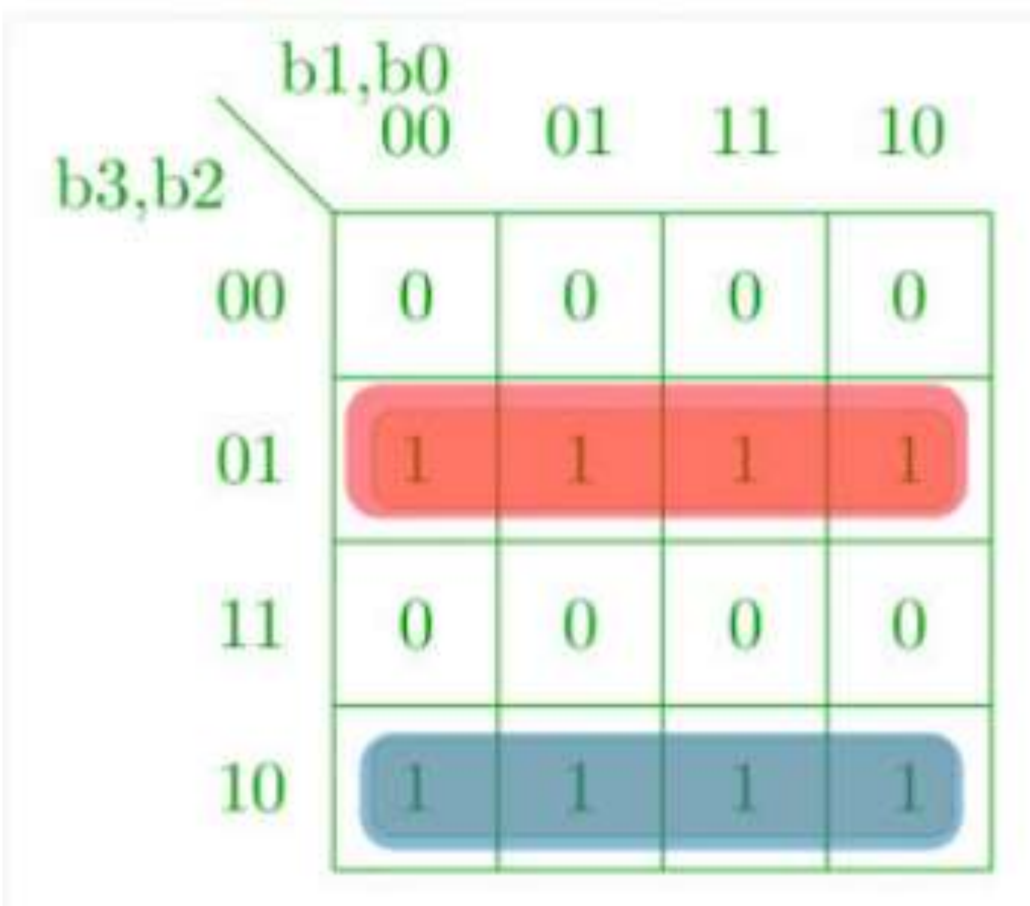
$$g_0 = b_0 b_1' + b_1 b_0' = b_0 \oplus b_1$$

K map for g1



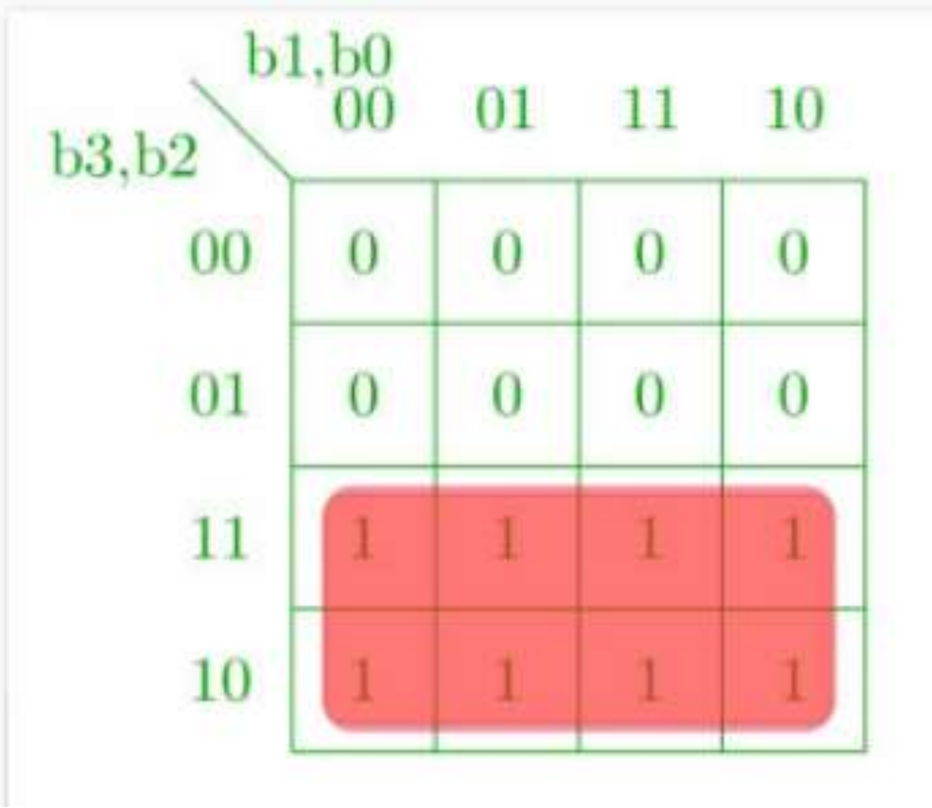
$$g_1 = b_2 b_1' + b_1 b_2' = b_1 \oplus b_2$$

K map for g2



$$g_2 = b_2 b_3' + b_3 b_2' = b_2 \oplus b_3$$

K map for g3



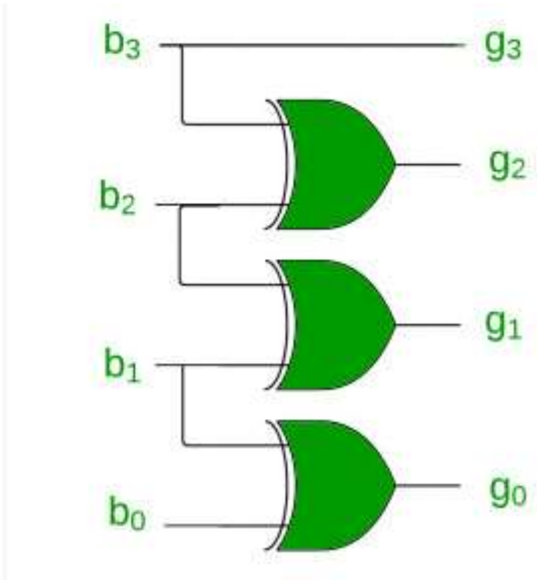
$$g_3 = b_3$$

Corresponding minimized boolean expressions for gray code bits

$$g_0 = b_0 b_1' + b_1 b_0' = b_0 \oplus b_1$$

$$g_1 = b_2 b_1' + b_1 b_2' = b_1 \oplus b_2$$
$$g_2 = b_2 b_3' + b_3 b_2' = b_2 \oplus b_3$$
$$g_3 = b_3$$

Code converter

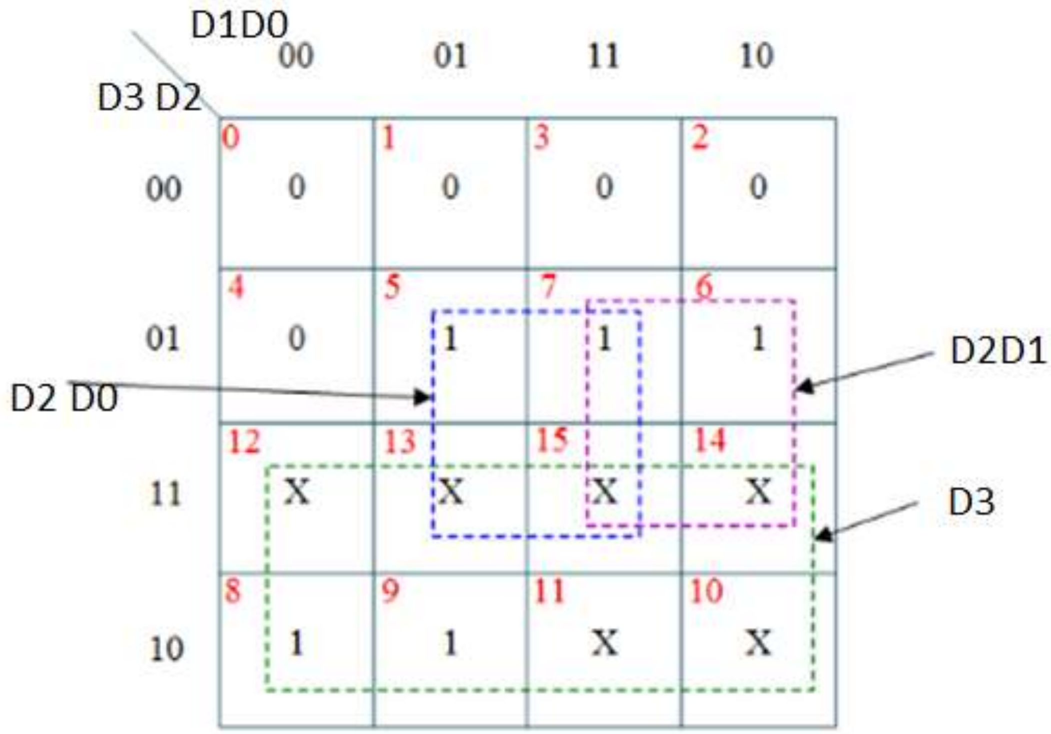


BCD to excess-3 code converter

As is clear by the name, a BCD digit can be converted to it's corresponding Excess-3 code by simply adding 3 to it.
Let $A,\ B,\ C,\ and\ D$ be the bits representing the binary numbers, where D is the LSB and A is the MSB, and
Let $w,\ x,\ y,\ and\ z$ be the bits representing the gray code of the binary numbers, where z is the LSB and w is the MSB.
The truth table for the conversion is given below. The X's mark don't care conditions.

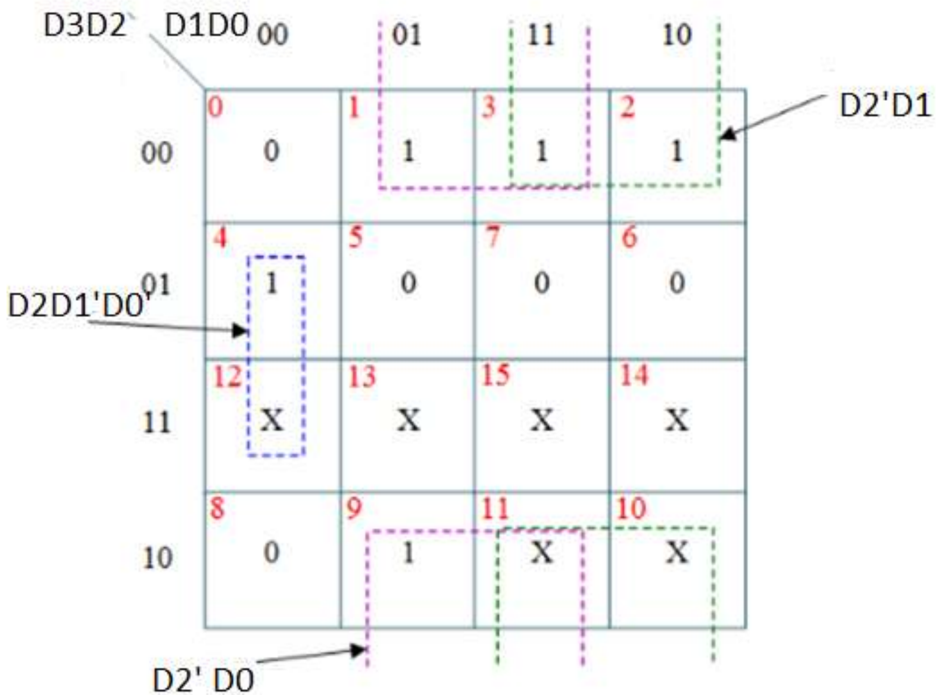
BCD(8421)				Excess-3			
D3	D2	D1	D0	E3	E2	E1	E0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

K map for E3



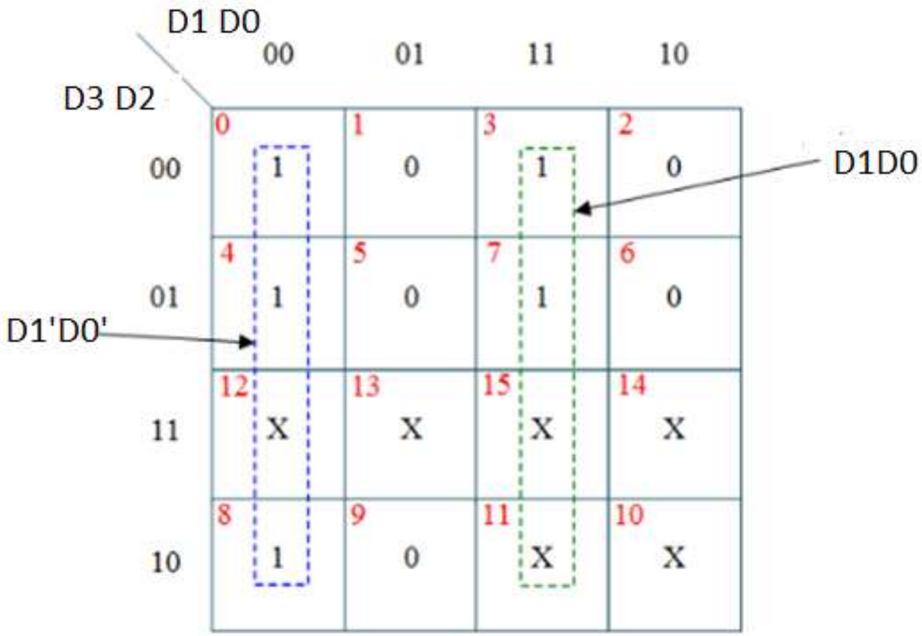
E3= D3+ D2D0+ D2D1

K map for E2



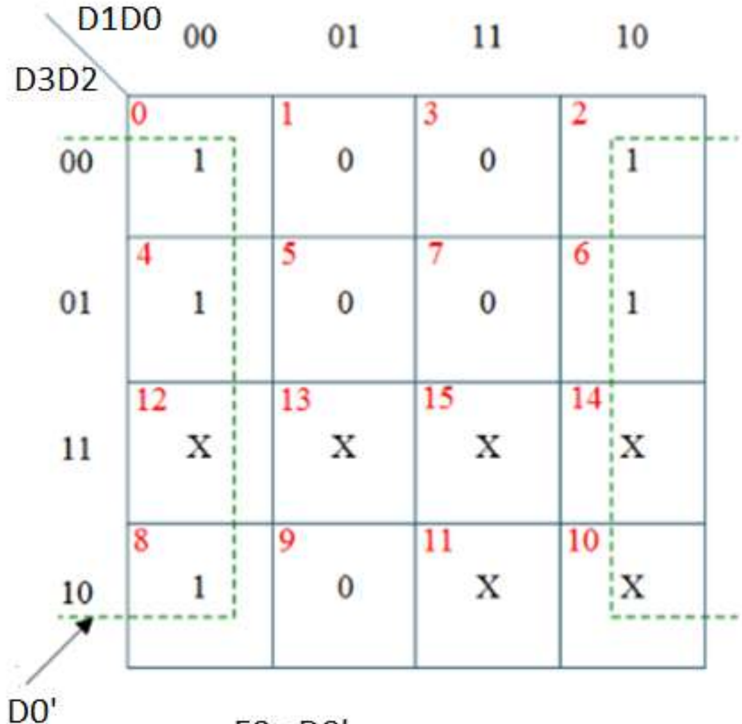
E2= D2'D1+D2'D0
+D2D1'D0'

K map for E1



E1=D1D0+D1'Do'

K map for z



E0= D0'

Minimized expressions

$E3 = D3 + D2D0 + D2D1$

$E2 = D2'D1 + D2'D0 + D2D1'D0'$

$E1 = D1D0 + D1'Do'$

$E0 = D0'$

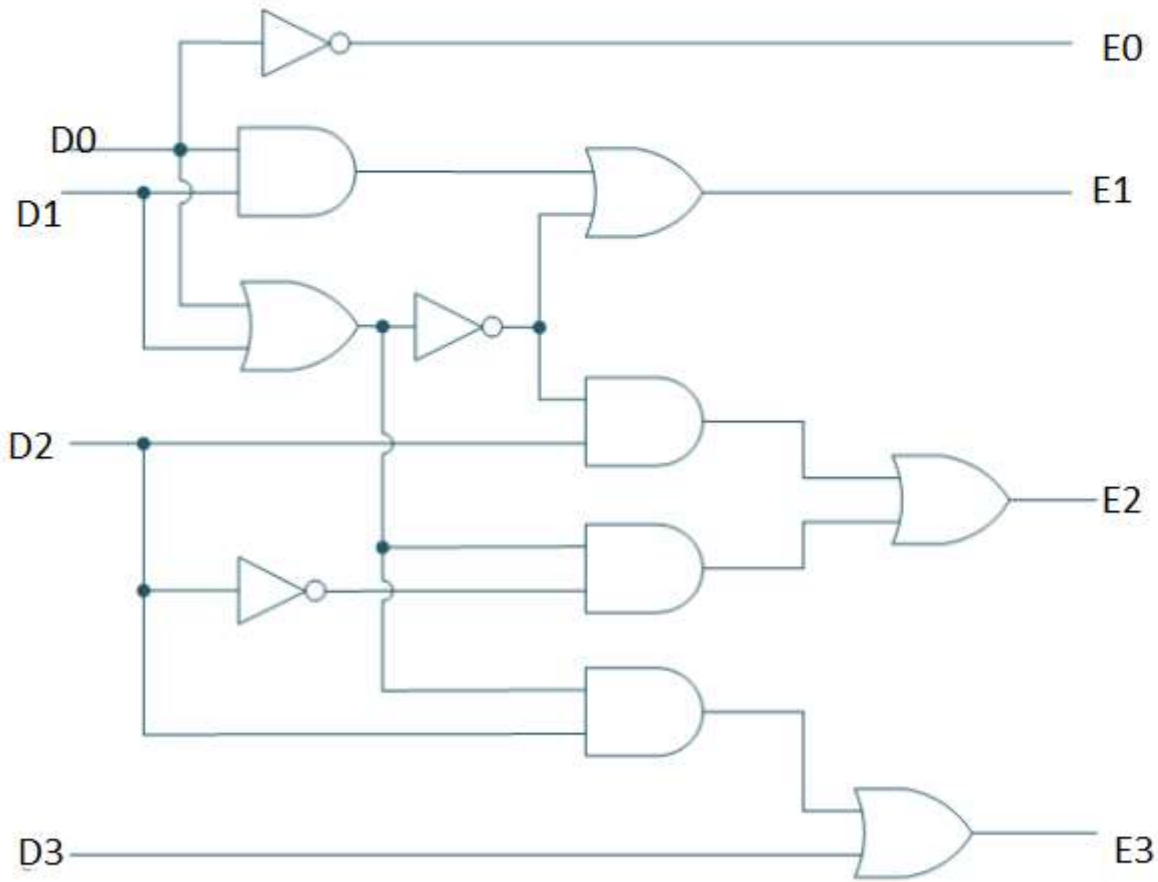
E3= D3+D2(D0+D1)

E2=D2’(D1+D0)+ D2D1’D0’

E1= D0 EXNOR D1

E0= D0’

BCD to Excess-3 code converter



Practice Problem

BCD to Gray code converter

Truth table

BCD code				Gray code			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1

K map simplification

For G_0

B_3B_2 \ B_1B_0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	X	X	X	X
10	0	1	X	X

$$G_0 = \overline{B}_1B_0 + B_1\overline{B}_0 = B_1 \oplus B_0$$

For G_1

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	x	x	x	x
10	0	0	x	x

$$G_1 = B_2\overline{B}_1 + \overline{B}_2B_1 = B_2 \oplus B_1$$

For G_2

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

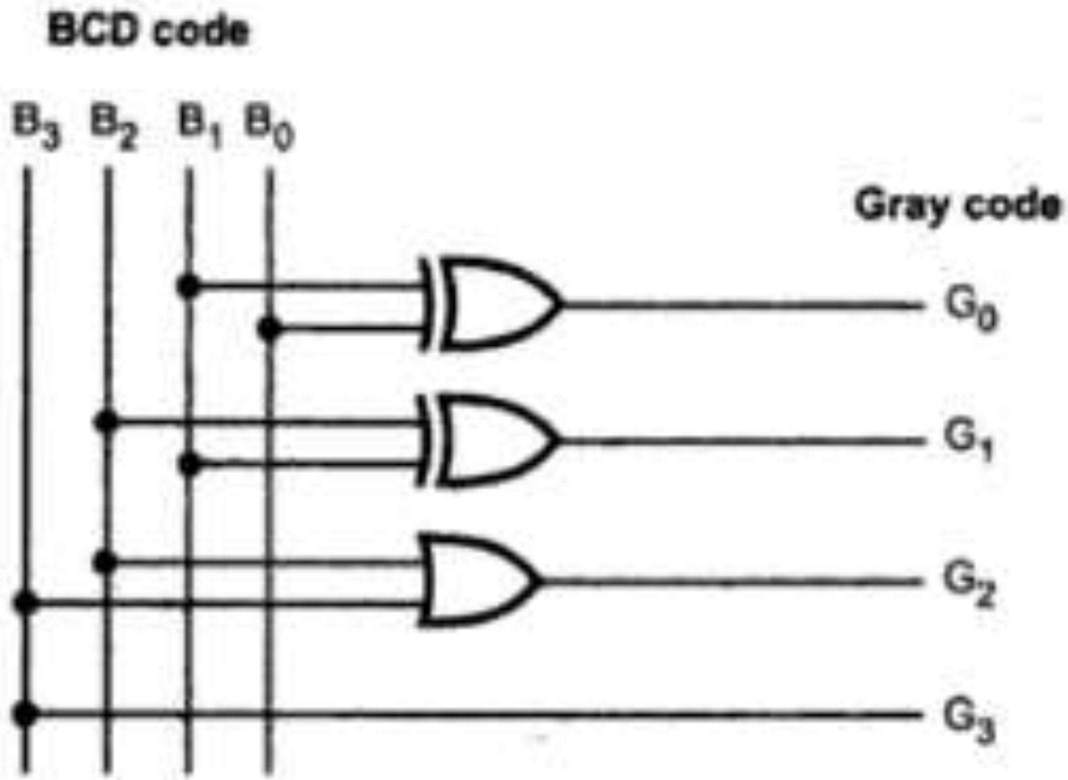
$$G_2 = B_2 + B_3$$

For G_3

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	1	1	X	X

$$G_3 = B_3$$

Circuit diagram



Thank you