$$B = \bar{X}\,Y + Z\,\overline{(X \oplus Y)} = \bar{X}\,(X + Y) + \overline{(X \oplus Y)}\,[(X \oplus Y) + Z]$$

$$= \overline{\overline{\overline{X + \overline{(X + Y)}} + \overline{(X \oplus Y) + \overline{X \oplus Y + Z}}}}$$

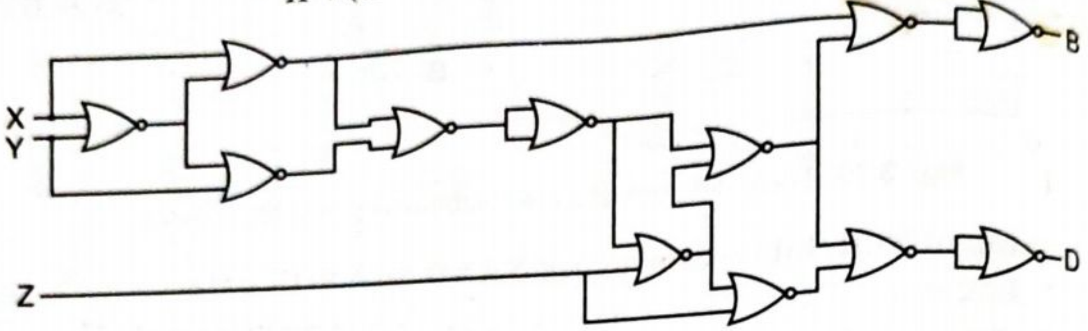


Fig. 3.41. Logic diagram of a full subtractor using NOR gates.

3.15 PARALLEL BINARY ADDERS

A full adder is capable of adding two 1-bits binary numbers and a carry-in. When two n-bit binary numbers are to be added, the number of full-adders will be equal to the number of bit n in each number. The addition of LSBs can be done by using either a half adder or a full adder with C_{in} terminal grounded. The carry-out of each full adder is connected to the carry-in of the next higher order adder. A parallel adder is used to add two numbers in parallel form and to produce the sum-bits as parallel outputs. A block diagram of 4-bit parallel adder capable of adding two 4-bit number *i.e.* $A_3\,A_2\,A_1\,A_0$ and $B_3\,B_2\,B_1\,B_0$ is shown in figure 3.39 and resulting outputs sum bits are $S_3\,S_2\,S_1\,S_0$ and C_{out} as carry bit e.g. if $A_3\,A_2\,A_1\,A_0$ 1101 and $B_3\,B_2\,B_1\,B_0 = 0101$.

A_0 B_0 C_{in} S_0 C_{out} — Least significant stage

$1 + 1 + 0 = 0$ 1

A_1 B_1 C_{in} S_1 C_{out}

$0 + 0 + 1 = 1$ 0

A_2 B_2 C_{in} S_2 C_{out}

$1 + 1 + 0 = 0.$ 1

A_3 B_3 C_{in} S_3 C_{out} — Most significant stage

$1 + 0 + 1 = 0$ 1

$S_4 \rightarrow$ Carry out

Therefore, we can see that $S_3\,S_2\,S_1\,S_0 = 0010$, since the carry-out from the most significant stage is as 1, we have an overflow, *i.e.* the sum (10010) must be expressed in 5-bits.

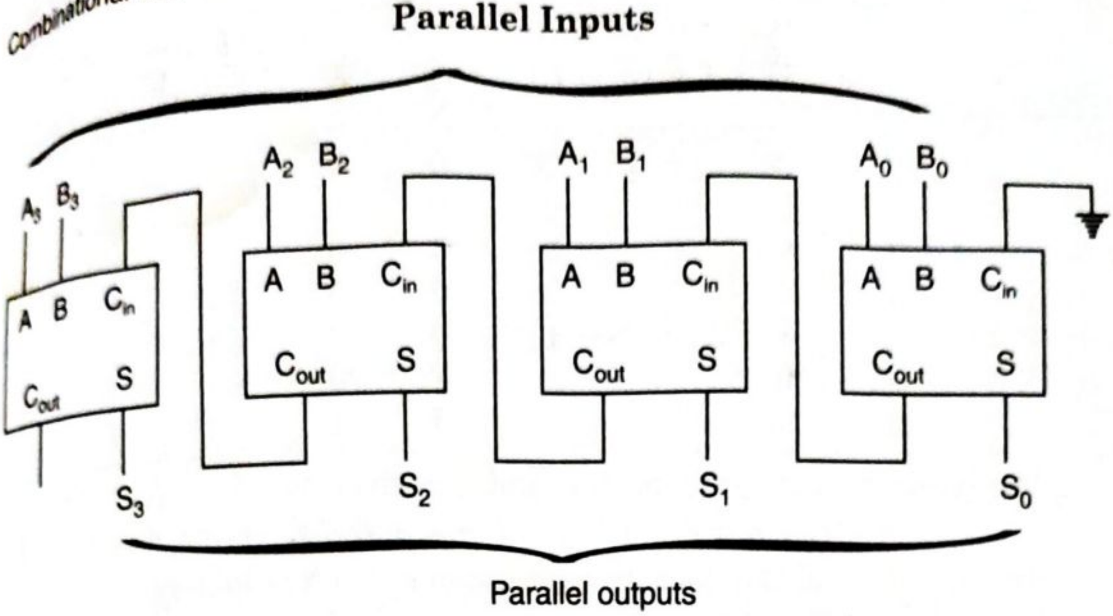


Fig. 3.42. A 4-bit parallel binary adder.

3.16 ENCODERS

An encoder is considered to be circuit which has multiple inputs and generates particular address at its output. An encoder has a number of input lines, only one of which is activated at a given time and produces an *N*-bit output code depending on which input is activated.

In Encoder the inputs are decimal digits and/or alphabetic characters and whose outputs are the coded representation of those inputs. Encoders performs the operation of encoding which is a process of converting numbers or symbols into a coded format. A block diagram of an encoder with *M* inputs and *N* outputs is shown in figure 3.43.

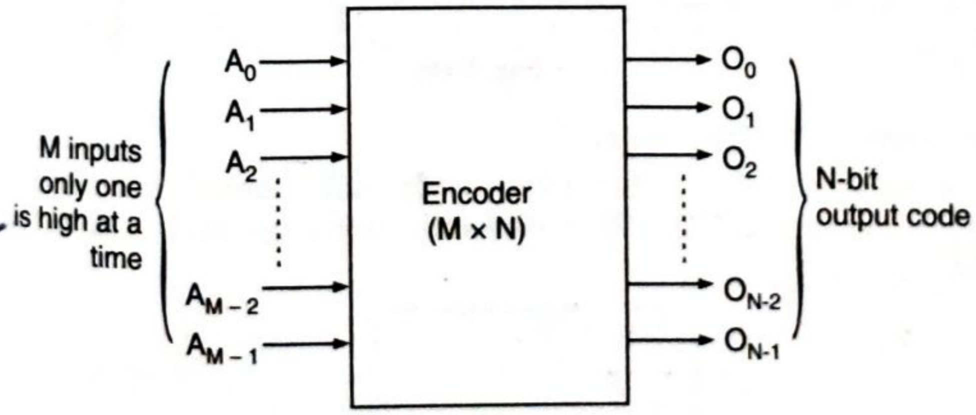


Fig. 3.43.

Types of encoders are :

(1) Octal-to-Binary Encoder

An octal to binary encoder (8 line to 3 line encoder) accepts 8 inputs lines and produces a 3-bit output code corresponding to the activated input. The eight inputs are 0, 1, 2, 3, 4, 5, 6, 7. The truth table is given as follows :

Input (Octal)								Output (Binary)		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

The reduced expression for A, B and C is given as

$$A = I_4 + I_5 + I_6 + I_7 \quad B = I_2 + I_3 + I_6 + I_7 \quad C = I_1 + I_3 + I_5 + I_7$$

The realization of the Boolean expression is given as follows :

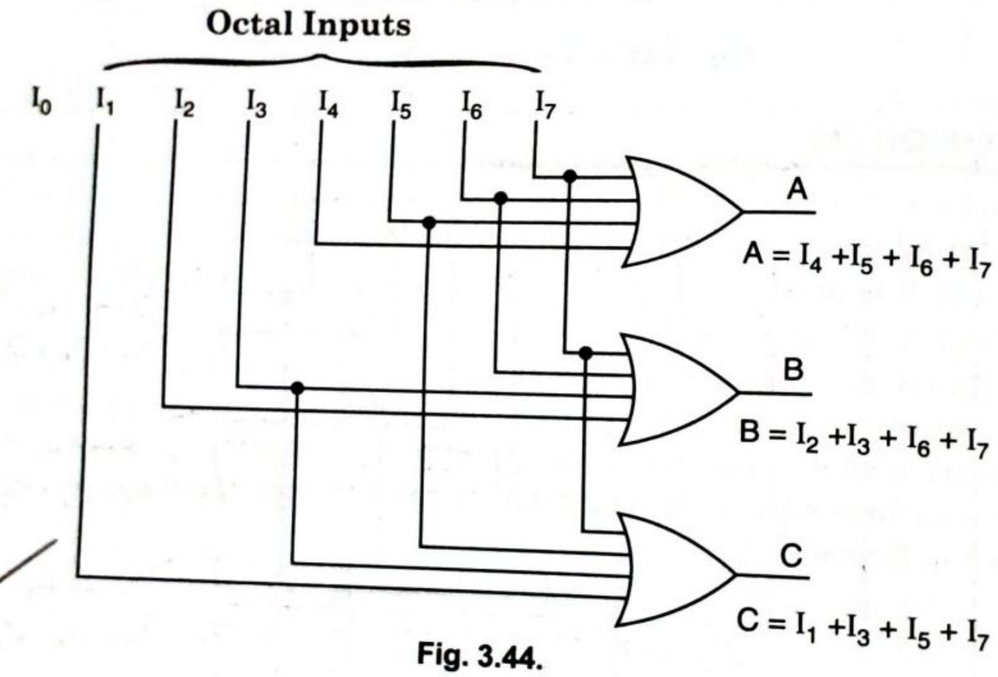


Fig. 3.44.

(2) Decimal to BCD Encoder

This type of encoder has 10 inputs–one for each decimal digit and 4 outputs. Corresponding to the BCD code as shown in block diagram. It is 10 line to 4 line encoder.

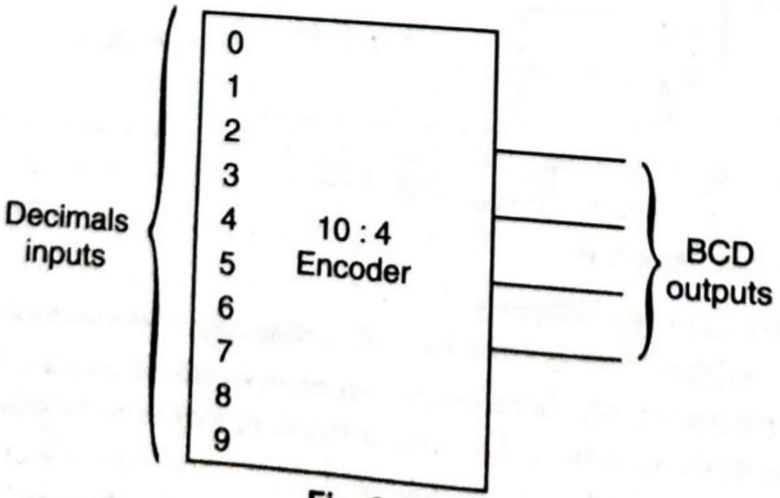


Fig. 3.45.

The truth table is represented as follows :

Decimal Inputs		Binary Outputs			
		A_3	A_2	A_1	A_0
D_0	0	0	0	0	0
D_1	1	0	0	0	1
D_2	2	0	0	1	0
D_3	3	0	0	1	1
D_4	4	0	1	0	0
D_5	5	0	1	0	1
D_6	6	0	1	1	0
D_7	7	0	1	1	1
D_8	8	1	0	0	0
D_9	9	1	0	0	1

The logic diagram is given below :

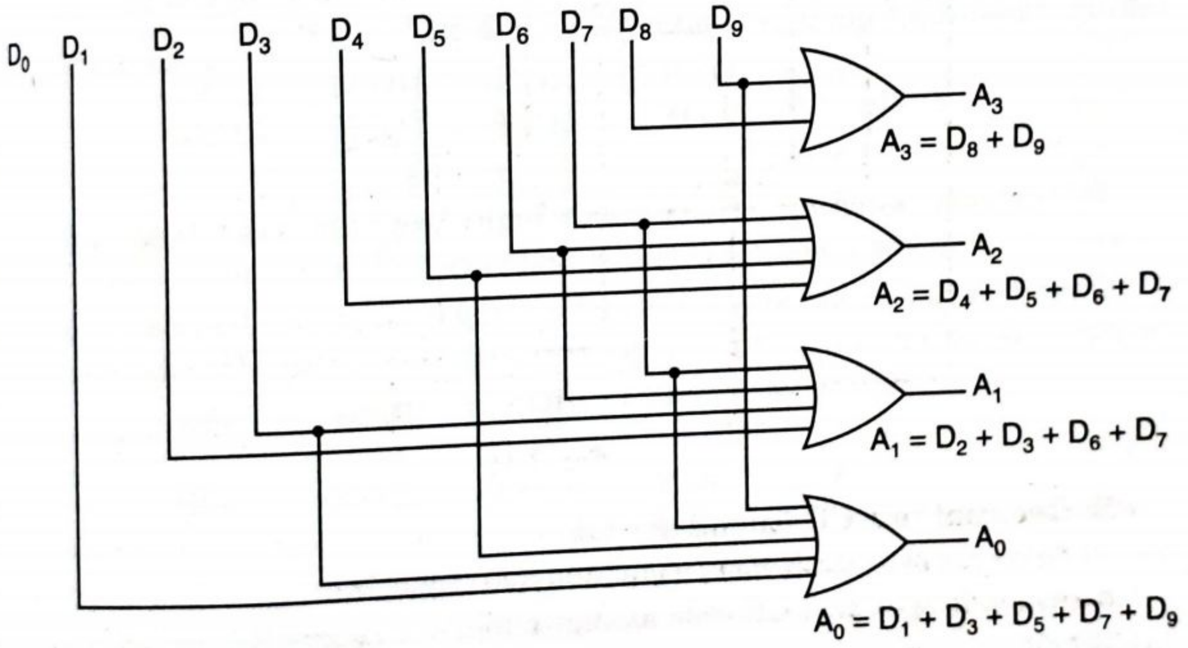


Fig. 3.46.

Code Converters

A type of combinational circuits that convert one code to another code. Main purpose of code converters is to encode the information in such a way that other person is enable to decode your private/secret information.

Hence, code converters more specifically encoders and decoders are used to protect private information benefits of using code converters.

1. **Encryption of data** more precisely called as cryptography so that private information can be protected. e.g. When we speak into cell phone, an encoder converts the sound of your voice into electrical signal which can travel very fast over very long distances. When the electrical signal

reach receivers cell phone,a decoder converts that electrical signal back to sound of your voice. in this way transmitted information remains recure.

2. **Code converters are also used to enhance data portablity and tractability.**

   Enhanced portability means information can be easily transported from one end to another end.

   Enhanced tractability means information can be easily stored as well as managed.

3. We can also do **hardware reduction** after doing the code conversion.
4. We can also **enhance speed of operation** and **decrease power dissipation**. Lets take one example to illustrate this:

Lets take the example of a counter which counts in binary and take the situation of a 4 bit binary counter. When it counters 8 after 7 i.e., transition from 0111 to 1000. All the 4 switches (Flip-flops) takes transition and change their status.

Lets assume that taking transition from 1 → 0 indicates switch off and taking transition from 0 → 1 indicates switch on.

7 ⇒ O(OFF) |(ON) |(ON) |(ON)
↓ ON ↓OFF ↓OFF ↓ OFF
8 ⇒ | 0 0 0

It indicates switching activity is very high. As we know that high switching activity, lead to low speed of operation and very high power dissipation.

Now if we use a code converters and uv gray coding, then biggest advantage of gray code isthat, only 1 bit changes between 2 consecutive numbers. e..g.,

7 ⇒ O(OFF) |(ON) |(ON) |(ON)
↓
8 ⇒ |(ON) 0(OFF) 0(OFF) 0 OFF

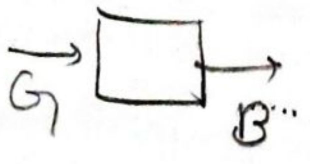
we see that here only one switch takes a transition from 0 to 1 and rest of 3 switches wants change. Hence, switching activity is less.

In this way speed of operation becomes high and power dissipation becomes low.

(1) Gray to Binary code converter

To design a circuit for Gray to binary code converter we proceed as follows:

The conversion table for binary to Gray Code is written first and then interchange the inputs means Gray is input and output is binary code. In this case we consider the four bits of Gray code as input (G_1, G_2, G_3 and G_4) and B_1, B_2, B_3, B_4 as the output bits.



Truth table for Gray to Binary Code is given as follows :

Decimal Number	Gray Codes				Binary			
	G_1	G_2	G_3	G_4	B_1	B_2	B_3	B_4
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	1	0	0	1	0
3	0	0	1	0	0	0	1	1
4	0	1	1	0	0	1	0	0
5	0	1	1	1	0	1	0	1
6	0	1	0	1	0	1	1	0
7	0	1	0	0 7	0	1	1	1 7
8	1	1	0	0 8	1	0	0	0 8
9	1	1	0	1	1	0	0	1
10	1	1	1	1	1	0	1	0
11	1	1	1	0	1	0	1	1
12	1	0	1	0	1	1	0	0
13	1	0	1	1	1	1	0	1
14	1	0	0	1	1	1	1	0
15	1	0	0	0	1	1	1	1

The output B_1, B_2, B_3 and B_4 can be expressed as interms of minterms are as follows :

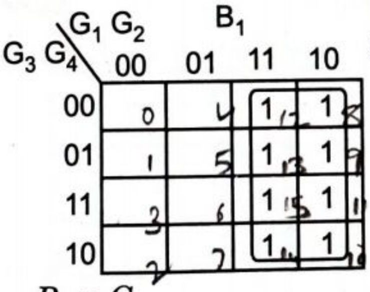
12, 13, 14, 15

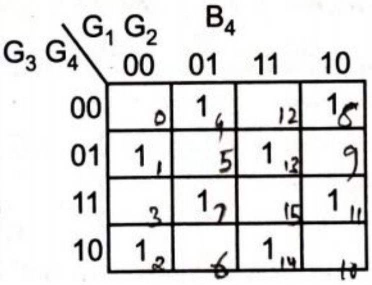
$B_1 = \Sigma m$ (8, 9, 10, 11, 12, 13, 14, 15) $B_2 = \Sigma m$ (4, 5, 6, 7, ~~8, 9, 10, 11~~)

$B_3 = \Sigma m$ (2, 3, ~~4, 5~~, ~~8, 9~~, 14, 15) $B_4 = \Sigma m$ (~~1, 2, 4, 7, 8, 11, 13, 14~~)

6, 7, 10, 11 1, 3, 5, 7, 9, 11, 13, 15

The K-maps for these variables are as follows :





$B_1 = G_1$

$$B_4 = \bar{G}_1 G_2 \bar{G}_3 \bar{G}_4 + G_1 \bar{G}_2 \bar{G}_3 \bar{G}_4 + \bar{G}_1 \bar{G}_2 \bar{G}_3 G_4 + G_1 G_2 \bar{G}_3 G_4 + \bar{G}_1 G_2 G_3 G_4 + G_1 \bar{G}_2 G_3 G_4 + \bar{G}_1 \bar{G}_2 G_3 \bar{G}_4 + G_1 G_2 G_3 \bar{G}_4$$

$$\Rightarrow \bar{G}_1 G_2 \bar{G}_3 \bar{G}_4 + \bar{G}_1 \bar{G}_2 \bar{G}_3 G_4 + G_1 \bar{G}_2 \bar{G}_3 \bar{G}_4 + G_1 G_2 \bar{G}_3 G_4 + \bar{G}_1 G_2 G_3 G_4 + \bar{G}_1 \bar{G}_2 G_3 \bar{G}_4 + G_1 \bar{G}_2 G_3 G_4 + G_1 G_2 G_3 \bar{G}_4$$

$$\Rightarrow \bar{G}_1 \bar{G}_3 [G_2 \bar{G}_4 + \bar{G}_2 G_4] + G_1 \bar{G}_3 [\bar{G}_2 \bar{G}_4 + G_2 G_4] + \bar{G}_1 G_3 [G_2 G_4 + \bar{G}_2 \bar{G}_4] + G_1 G_3 [\bar{G}_2 G_4 + G_2 \bar{G}_4]$$

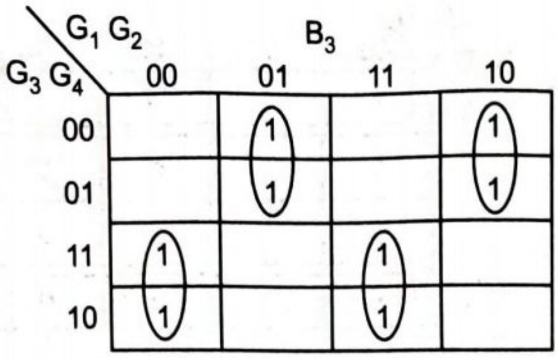
$$\Rightarrow \bar{G}_1 \bar{G}_3 [G_2 \oplus G_4] + G_1 \bar{G}_3 [G_2 \odot G_4] + \bar{G}_1 G_3 [G_2 \odot G_4] + G_1 G_3 [G_2 \oplus G_4]$$

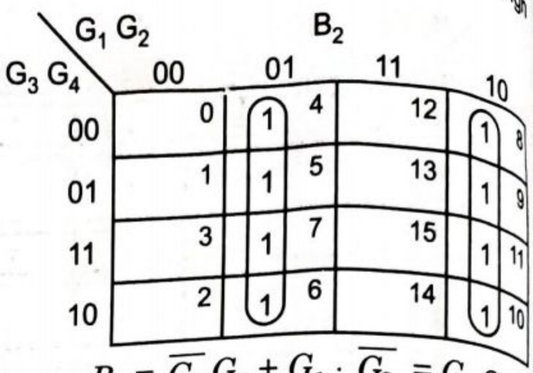
$$\Rightarrow \bar{G}_1 \bar{G}_3 [G_2 \oplus G_4] + G_1 G_3 [G_2 \oplus G_4] + G_1 \bar{G}_3 [G_2 \odot G_4] + \bar{G}_1 G_3 [G_2 \odot G_4]$$

$$\Rightarrow [\bar{G}_1 \bar{G}_3 + G_1 G_3][G_2 \oplus G_4] + [G_1 \bar{G}_3 + \bar{G}_1 G_3][G_2 \odot G_4]$$

$$= [G_1 \odot G_3][G_2 \oplus G_4] + [G_1 \oplus G_3][G_2 \odot G_4]$$

$$= G_1 \oplus G_2 \oplus G_3 \oplus G_4$$





$$B_3 = \overline{G_3}\,\overline{G_1}\;G_2 + \overline{G_1}\,\overline{G_2}\,G_3 + G_1\overline{G_2}\;\overline{G_3} + G_1\,G_2\,G_3 = G_1 \oplus G_2 \oplus G_3$$

$$B_2 = \overline{G_1}\,G_2 + G_1 \cdot \overline{G_2} = G_1 \oplus G_2$$

Designing of the Circuit

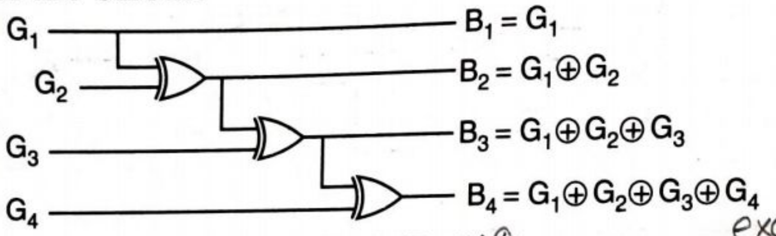
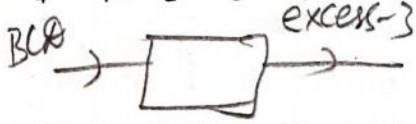


Fig. 3.47.

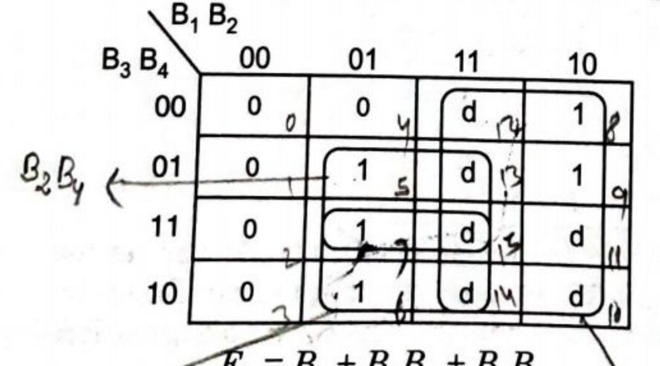


(2) BCD to Excess-3 code converter

In order to convert the BCD to Excess-3, we first write down the truth table of BCD to excess-3 code. The corresponding to decimal numbers 10, 11, 12, 13, 14 and 15 are considered as don't care terms. The truth table is as follows :

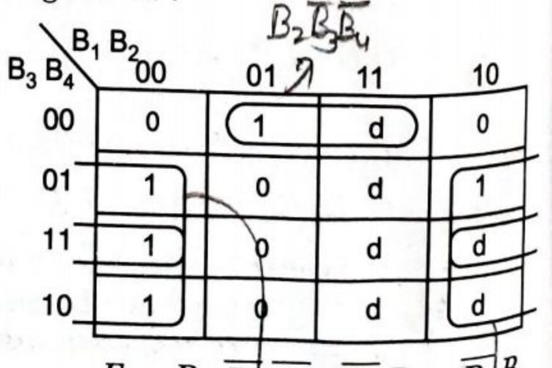
Decimal Number	*BCD Codes*				*Excess-3 Code*			
	B_1	B_2	B_3	B_4	E_1	E_2	E_3	E_4
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

The reduced expression for K-map is given as :



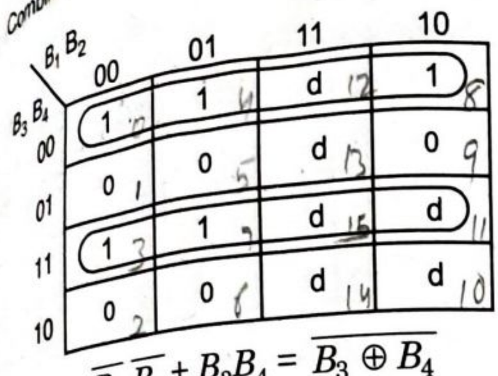
$$E_1 = B_1 + B_2B_4 + B_2B_3$$

B_2B_3 B_1

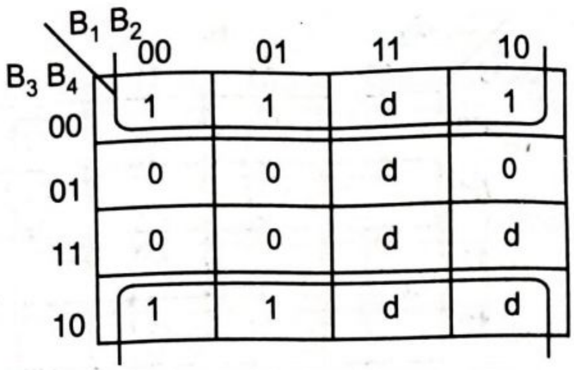


$$E_2 = B_2\;\overline{B_3}\;\overline{B_4} + \overline{B_2}\,B_4 + \overline{B_2}\,B_3$$

$\overline{B_2}B_4$ $\overline{B_2}B_3$



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



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

$$E_3 = \overline{B_3}\,\overline{B_4} + B_3B_4 = \overline{B_3 \oplus B_4}$$

$$E_4 = \overline{B_4}$$

The circuit realization is given as follows :

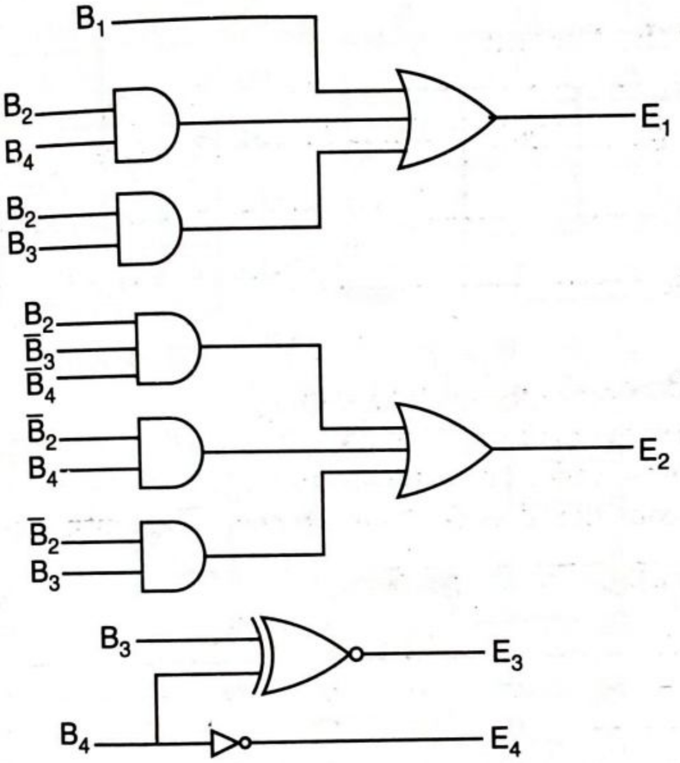


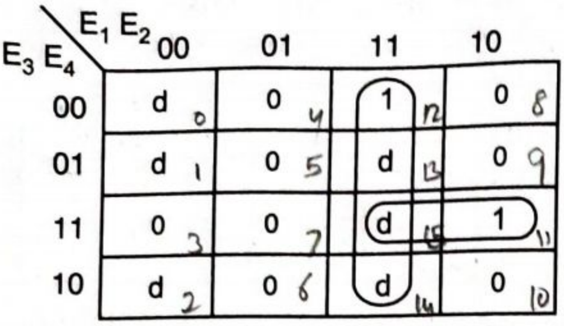
Fig. 3.48.

(3) Excess-3 to BCD code converter

Truth table is as follows:

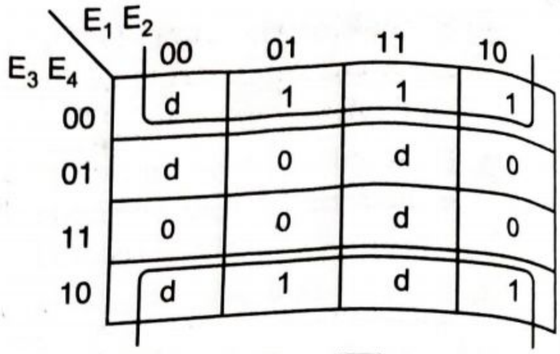
	E_1	E_2	E_3	E_4		B_1	B_2	B_3	B_4
3	0	0	1	1	—	0	0	0	0
4	0	1	0	0	—	0	0	0	1
5	0	1	0	1	—	0	0	1	0
6	0	1	1	0	—	0	0	1	1
7	0	1	1	1	—	0	1	0	0
8	1	0	0	0	—	0	1	0	1
9	1	0	0	1	—	0	1	1	0
10	1	0	1	0	—	0	1	1	1
11	1	0	1	1	—	1	0	0	0
12	1	1	0	0	—	1	0	0	1

In this encoder Excess-3 is input and BCD code is output. As the terms corresponding to decimal number 0, 1, 2, 13, 14, 15 will not appear in Excess-3 code, these terms are considered as don't care terms (*d*). The Boolean expression is given as after solving K-maps are :



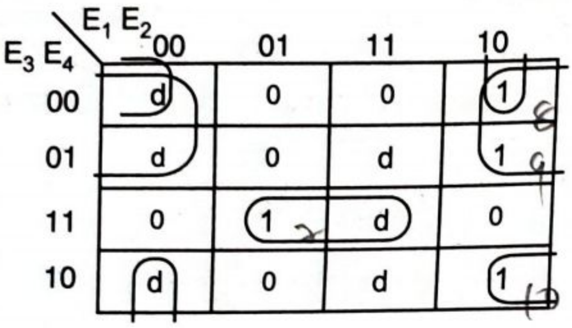
E_3E_4 \ E_1E_2	00	01	11	10
00	d 0	0 4	1 12	0 8
01	d 1	0 5	d 13	0 9
11	0 3	0 7	d 15	1 11
10	d 2	0 6	d 14	0 10

$$B_1 = E_1E_2 + E_1E_3E_4$$



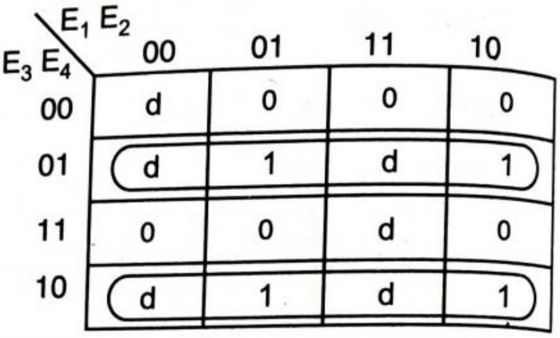
E_3E_4 \ E_1E_2	00	01	11	10
00	d	1	1	1
01	d	0	d	0
11	0	0	d	0
10	d	1	d	1

$$B_4 = \overline{E_4}$$



E_3E_4 \ E_1E_2	00	01	11	10
00	d	0	0	1 8
01	d	0	d	1 9
11	0	1 2	d	0
10	d	0	d	1 10

$$B_2 = \bar{E}_2\,\bar{E}_{\underset{3}{4}} + \bar{E}_4\,\bar{E}_2 + E_2E_3E_4$$



E_3E_4 \ E_1E_2	00	01	11	10
00	d	0	0	0
01	d	1	d	1
11	0	0	d	0
10	d	1	d	1

$$B_3 = \overline{E_3}\,E_4 + E_3\overline{E_4} = E_3 \oplus E_4$$

The circuit realization is given as follows :

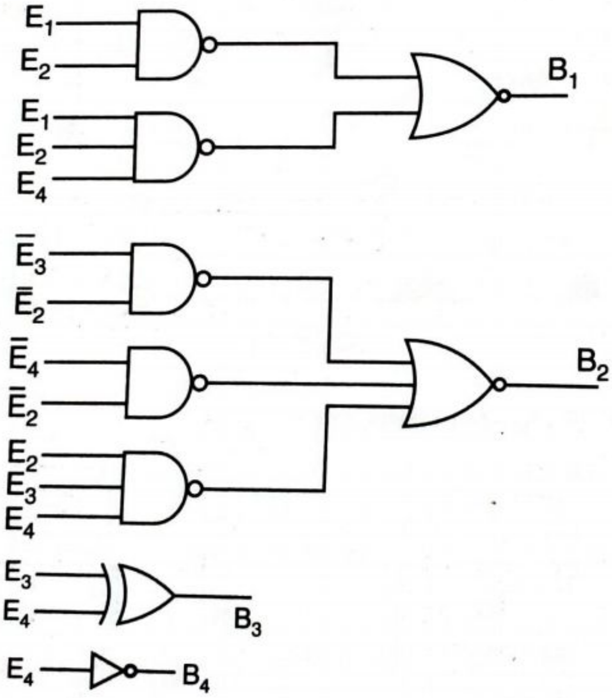


Fig. 3.49.

3.17 DESIGN PROCEDURE

We can design any combinational circuit by determination of boolean functions. Steps of designing are as follows :

1. Find out the number of inputs and outputs by provided specifications about the combinational circuit. Assign arbitrary variables to all inputs and outputs.

2. By making the input combinations based on arbitrary input variables, we can make the truth table corresponding to relationship between input and output variables.
3. Find out the boolean function corresponding to each output variable as a function of input variable. K-map approach can be utilized to obtain simplied boolean functions.
4. Based on boolean functions, draw the logic diagram.
5. Verify obtained circuit design manually or by using logic simulation.

Let us explain it with an example. Suppose we want to convert '3' bit binary code to gray code. It requires 3 bits B_3, B_2, B_1 for input and three bits for output i.e., G_3, G_2, G_1. For '3' bits the combinations of inputs ($2^3 = 8$) are 000 to 111. Next step is to draw the truth table taking input and output values. It is given as

Binary code			Gray code		
B_3	B_2	B_1	G_3	G_2	G_1
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Next step is to determine the value of G_3, G_2 and G_1. It is obtained by taking B_3, B_2 and B_1(inputs) and solving the k map corresponding to each outputs G_3, G_2 and G_1. The three inputs are there, so three variable k-map is used. It is given as

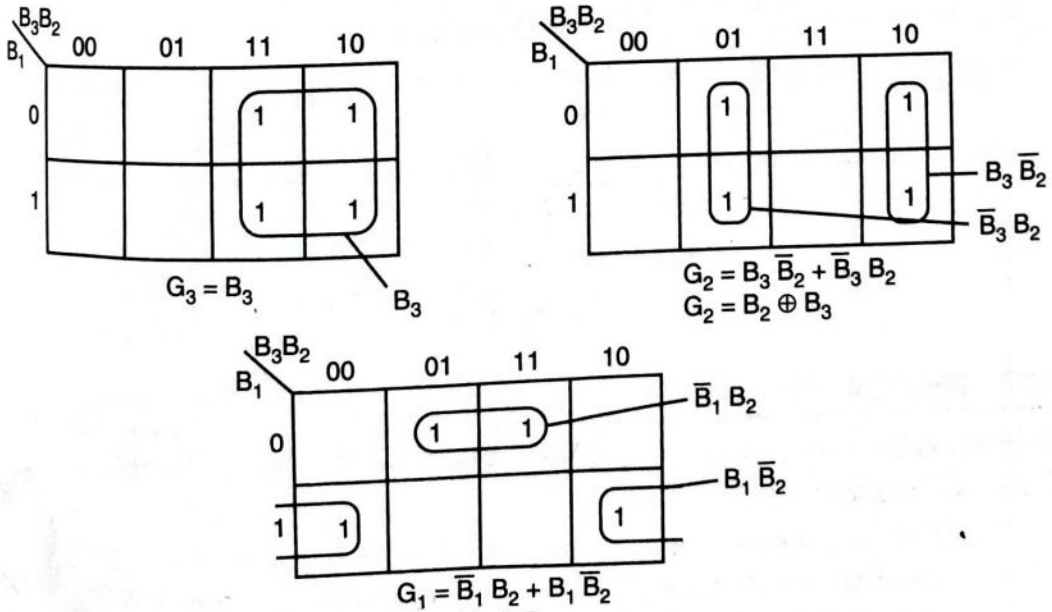


Fig. 3.50.