The designing is as follows :

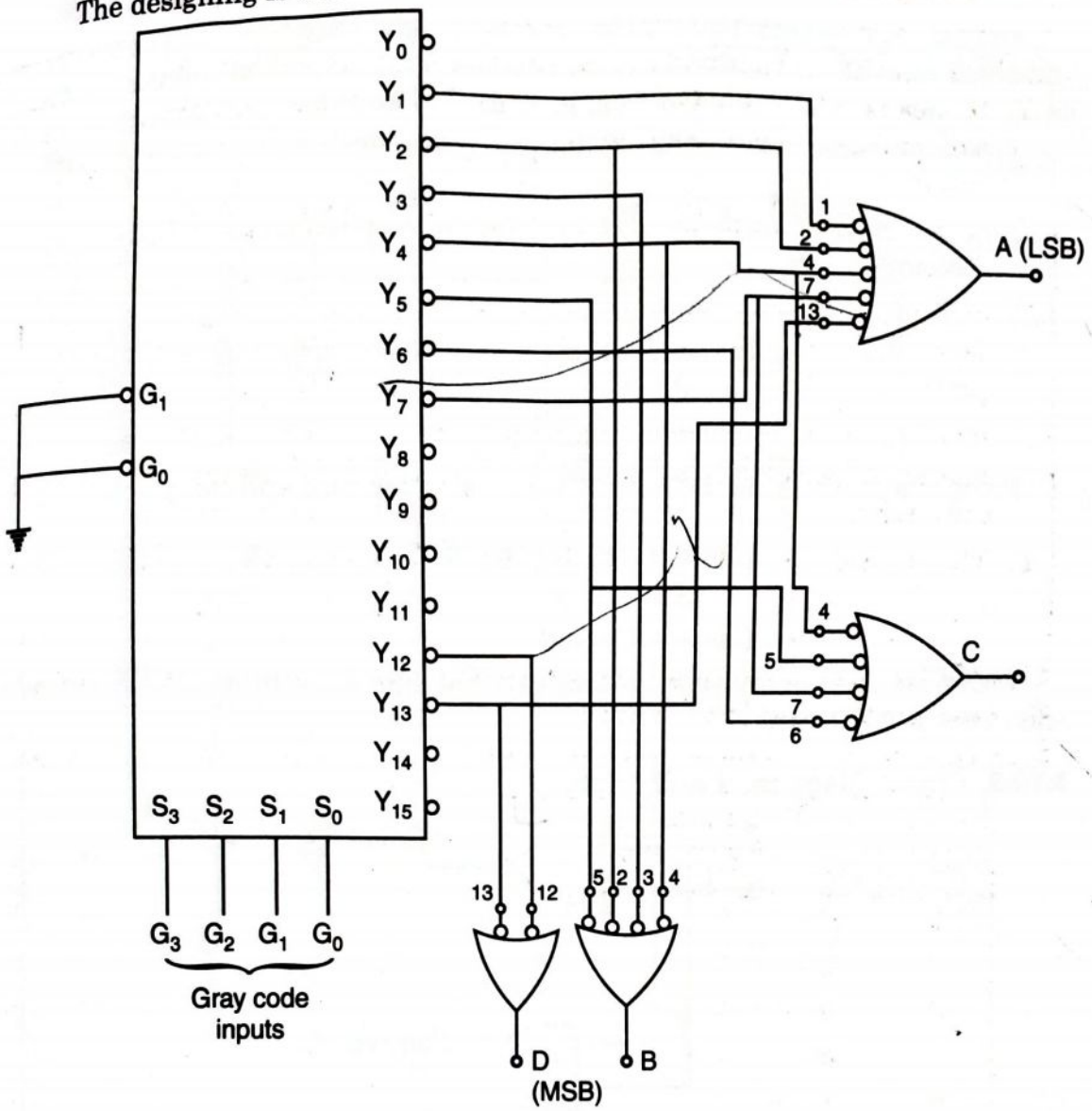



Fig. 3.23.

## 3.10 ADDERS

Adders are the combinational circuit that perform addition of bits. Addition of two binary digits is most basic arithmetic operation. Table format the simple addition consists of four possible operations, which are

$$0 + 0 = 0,$$
$$0 + 1 = 1,$$
$$1 + 0 = 1$$

and $$1 + 1 = 10.$$

The first three operations produce a sum of one digit and last operation produces sum consists of two digits. The higher significant bit of this result is called carry. Adders is classified into two types :

(*i*) Half Adder

(*ii*) Full Adder

### 3.10.1. Half Adder

A combinational circuit that performs the addition of two bits is known as Half Adder.

### 3.10.2. Designing of Half Adder

In this logic circuit, two inputs and two outputs are present. Let the input variables augend and addend are denoted as $A$ and $B$ and outputs are denoted as $S$ and $C$ where '$S$' is known as sum and '$C$' is known as carry.

The truth table involves the following operations *i.e.*

| Input variables | | Output | |
|---|---|---|---|
| A | B | S | C |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

From the truth table the output expression for sum and carry

$$\text{Sum} = S = \bar{A}B + A\bar{B}$$

$$S = A \oplus B$$

$$\text{Carry} = C = AB$$

From the logic expression we can say that sum is nothing but EX-OR gate and carry is AND gate.

### 3.10.3. Circuit Diagram of Half Adder

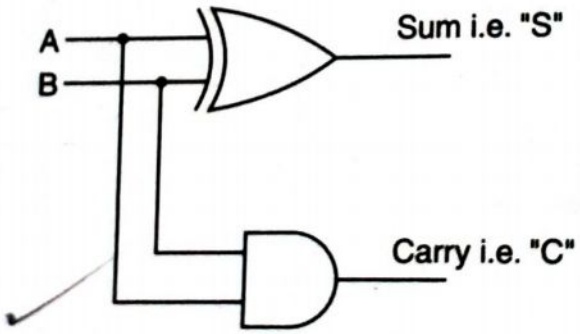



**Fig. 3.24.**

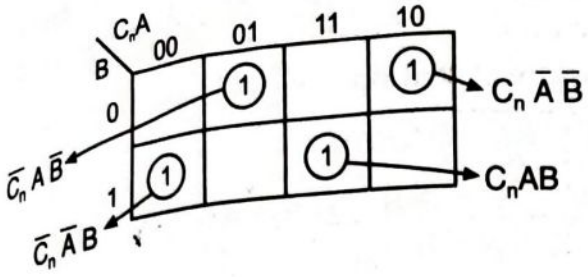
### 3.10.4. Full Adder

In full adder, there is an addition of three bits *i.e.* augend, addend and previous carry and produces the outputs sum and carry. Let us denote the input variable augend by $A$, addend by $B$ and previous carry by $C_n$ and outputs sum by $S$ and carry by $C$. The truth table for three variable input is given as follows:

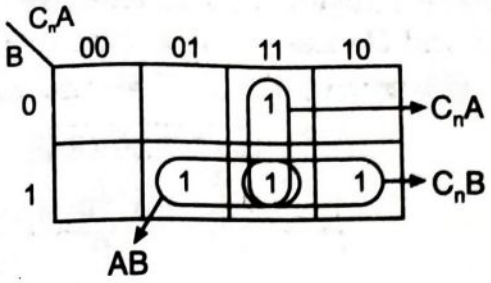
| Input Variables | | | Outputs | |
|---|---|---|---|---|
| $C_n$ | A | B | S | C |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

The K map realization for sum and carry output is given as follows :

**For Sum**




**For Carry**




$$S = \overline{C_n} A \overline{B} + C_n \overline{A} \overline{B} + \overline{C_n} \overline{A} B + C_n AB$$

$$C = C_n A + AB + C_n B$$

The circuit can be realized using three inputs 'AND' gate and 'OR' gate for sum output and two-input 'AND' gate and 'OR' gate for carry output.

The designing of the circuit is as follows :

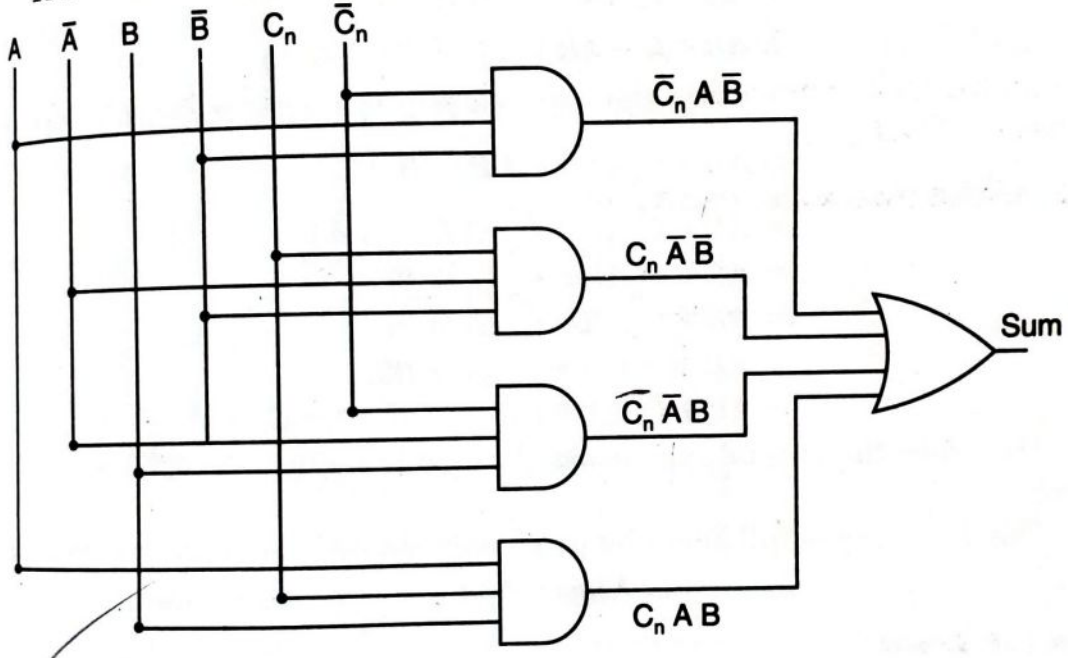



**Fig. 3.25.** Output representation of sum function.

The designing of full adder circuit for carry output is given as follows :

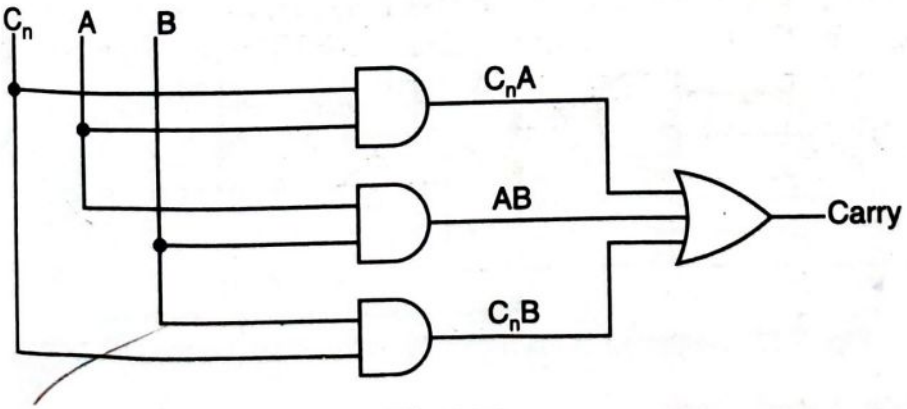



**Fig. 3.26.**

$AB + BC + AC \Rightarrow AB(C + \bar{C}) + BC(A + \bar{A}) + AC(B + \bar{B})$

$\Rightarrow ABC + AB\bar{C} + ABC + \bar{A}BC + ABC + A\bar{B}C$

$\Rightarrow AB\bar{C} + ABC + \bar{A}BC + ABC$

**Realization of Full Adder Circuit by Two Half Adder and OR Gate**

Since the expression for sum is given as :

$$S = \overline{C_n}\, A \bar{B} + C_n\, \bar{A}\, \bar{B} + \overline{C_n}\, \bar{A} B + C_n AB$$

$$= (\overline{C_n} A + C_n \bar{A})\, \bar{B} + (\overline{C_n}\, \bar{A} + AC_n)\, B$$

$$= \bar{B}\,(A \oplus C_n) + B\,\overline{(A \oplus C_n)}$$

$$= C_n \oplus A \oplus B \qquad [\because\ \bar{A} B + \bar{B} A = A \oplus B]$$

[as $A \oplus B = B \oplus A$]

Therefore the sum expression is reduced to EX-OR gates

[As in case of Half Adder sum is generated by EX-OR gate]

The expression for carry is given as :

$$C = AB + C_n B + C_n A$$

$$= AB + C_n\,(A + B)$$

$$= AB + C_n\,(A\,(B + \bar{B})) + B\,(A + \bar{A}))$$

$$= AB + C_n\,(AB + A\bar{B} + BA + B\bar{A})$$

$[\because\ A + \bar{A} = 1,\ B + \bar{B} = 1]$ $[AB + BA = AB + AB = AB]$

$$= AB + C_n\,(AB + A\bar{B} + B\bar{A})$$

$$= AB + C_n AB + C_n\,(A\bar{B} + B\bar{A})$$

$$= AB + C_n AB + C_n\,(A \oplus B)$$

$$= AB + C_n AB + C_n\,(A \oplus B)$$

$$= AB\,(1 + C_n) + C_n\,(A \oplus B)$$

$$= AB + C_n\,(A \oplus B) \qquad [\because\ 1 + C_n = 1]$$

Therefore the output expression of carry is reduced to AND and EX-OR gate.

The designing of full adder by using only Half Adder is given as follows :

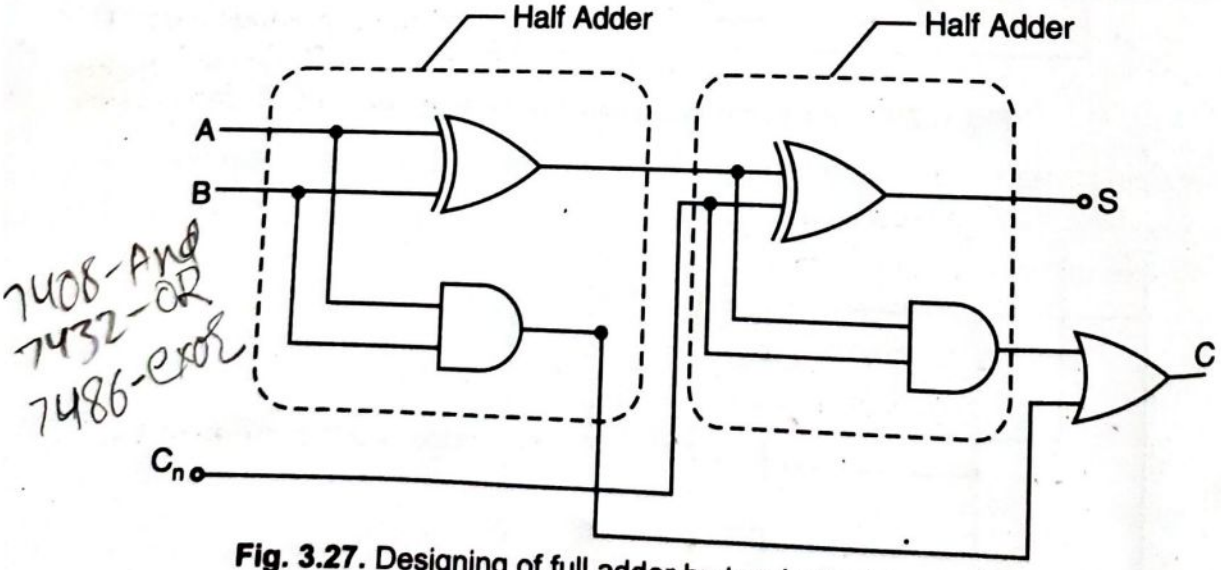



**Fig. 3.27.** Designing of full adder by two half adder and OR gate.

### 3.10.5. Look Ahead Carry

Look ahead carry is a concept introduced to calculate the previous carry in multistage address so as to reduce the propagation delay of gate and increase speed of adder circuit.

The carry generated in adder circuit is dependent only on given bits, it can be seen in drawn below.

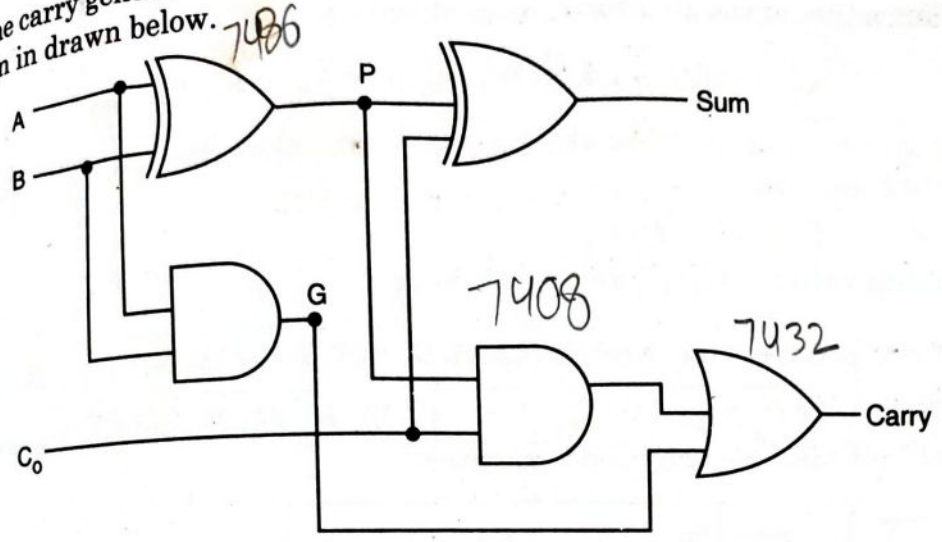
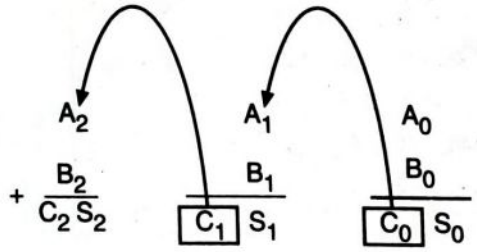



**Fig. 3.28.**

As see above the carry generated in adder depends on the value of $A$ & $B$ and $C_0$ (previous carry) $A$ & $B$ are given in case of multi bit addition and previous carry is the calculated so that no more delay is present in the circuit.

### Need of Calculation of Previous Carry

Suppose we have to add two 3 bit numbers and result is sum and final carry as shown below




Final result with carry is $C_2\ S_2\ S_1\ S_0$

The calculation of $A_1$, $B_1$ adder required the value of previous carry $C_0$ which is completely depend on $A_0$, $B_0$. Similarly the value of $C_1$ depend on $A_1$, $B_1$. Similarly the value of $C_2$ depends on $A_2$, $B_2$. Hence value of $C_1$ depends on $A_0$, $B_0$, $A_1$, $B_1$.

### Calculation of Previous Carry :

From figure 3.28 , we have

$$P_n = A_n \oplus B_n \quad G_n = A_n B_n$$
$$S_n = P_n \oplus C_{n-1} = A_n \oplus B_n \oplus C_{n-1}$$
$$C_n = G_n + P_n C_{n-1}$$

If we take example of 3 bit adder and have to calculate the value of $C_2$, $C_1$, $C_0$ then using above equations.

$$P_0 = A_0 \oplus B_0, \quad G_0 = A_0 B_0 \quad C_0 = G_0 + P_0 C_{-1}$$

Putting value of $P_0$, $B_0$

$$\boxed{C_0 = A_0 B_0 + (A_0 \oplus B_0) C_{-1}}$$

Hence $C_0$ depends on $A_0$, $B_0$ & $C_{-1}$, all values are given, so $C_0$ can be calculated separately.

Now $C_1 = G_1 + P_1 C_0$

Place value of $G_1$, $P_1$ and $C_0$ from above equation's we have.

$$C_1 = A_1 B_1 + (A_1 \oplus B_1)\,[A_0 B_0 + (A_0 \oplus B_0)\, C_{-1}]$$

All value used in calculating $C_1$ is given hence the value of $C_1$ is also calculated separately.

Now $C_2 = G_2 + P_2 C_1$

Putting value of $G_2$, $P_2$ and $C_1$ we have

$$C_2 = A_2 B_2 + (A_2 \oplus B_2)\,[A_1 B_1 + (A_1 \oplus B_1)\,\{A_0 B_0 + (A_0 \oplus B_0)\, C_{-1}\}]$$

The value of $C_2$ depends on $A_2$, $A_1$, $A_0$, $B_2$, $B_1$, $B_1$, $B_0$ and $C_{-1}$ means given quantity and can be calculated separately.

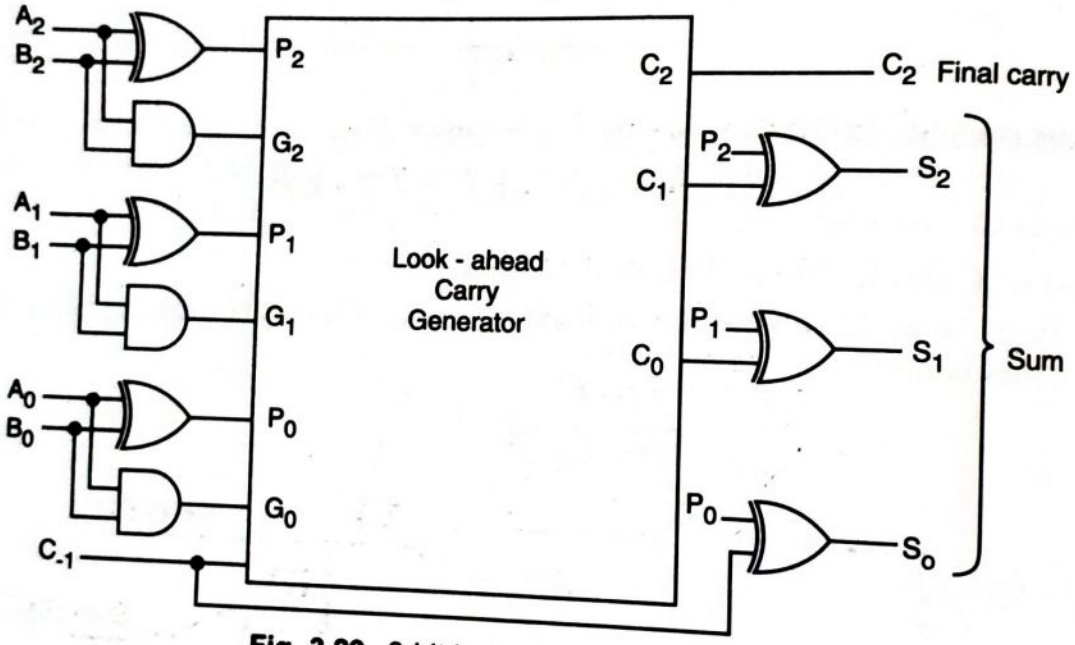



**Fig. 3.29.** 3-bit look-ahead carry adder.

## Designing of Half Adder using NAND and NOR gates

The expression for sum and carry is given as follows :

Sum $S = A\bar{B} + \bar{A}B$

$= A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B}$

$= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B})$

$= A \cdot \overline{AB} + B \cdot \overline{AB}$ $[\because A\bar{A}$ and $B\bar{B} = 0]$

$= \overline{\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}}$ $[\because \overline{AB} = \bar{A} + \bar{B}]$

$\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}$

Carry $C = AB = \overline{\overline{AB}}$

**Designing using NAND gates**

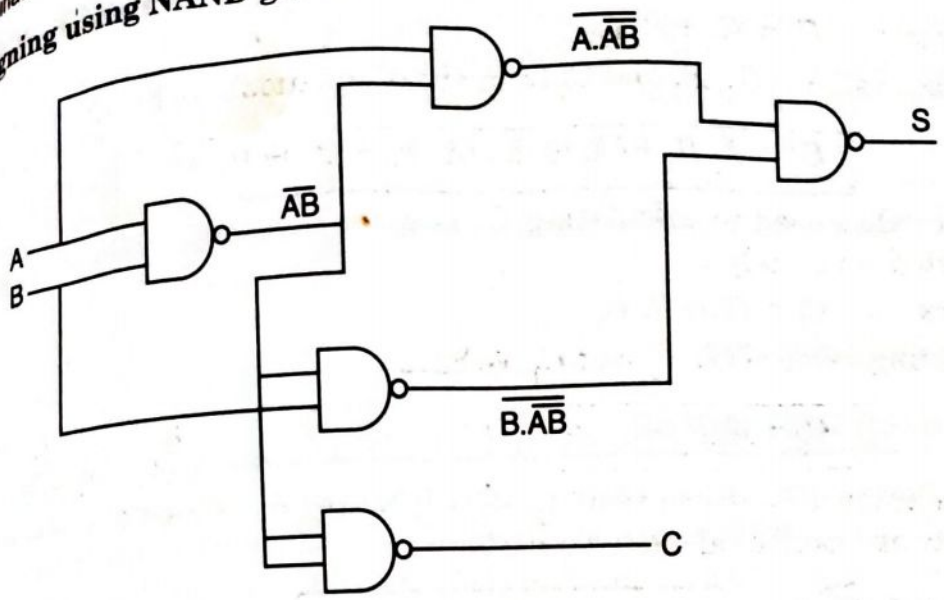



Fig. 3.30.

**Designing of Half Adder using NOR gates**

$$S = A\bar{B} + \bar{A}B = A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B}$$

$$= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B})$$

$$= (A + B)(\bar{A} + \bar{B})$$

$$= \overline{\overline{A + B}} + \overline{\overline{A} + \overline{B}}$$

$$C = AB = \overline{\overline{AB}} = \overline{\bar{A} + \bar{B}}$$

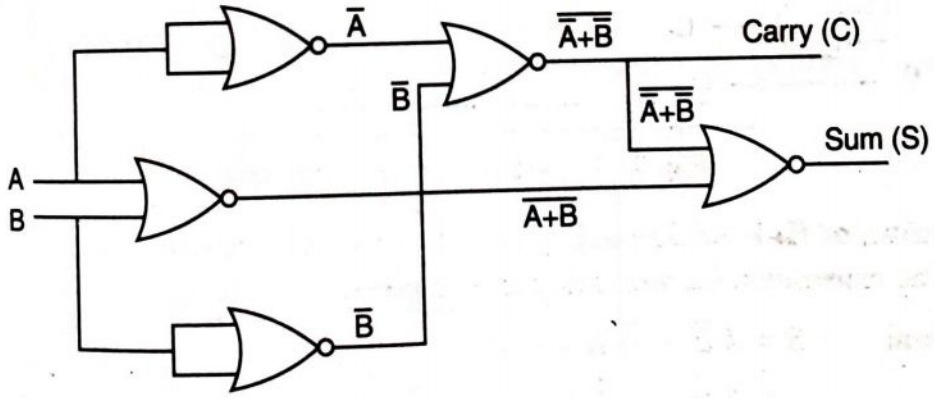



Fig. 3.31.

## 3.11 SUBTRACTORS

The combinational circuit which is going to perform the subtraction of bits is known as subtractors. Subtraction of two binary digits consists of four possible combinations are,

$0 - 0 = 0,$

$0 - 1 = 1$ (With borrow of 1)

and

$1 - 0 = 1$

$1 - 1 = 0.$

The first, third and fourth operations produce a subtraction of one digit, but the second operation produces difference bit and borrow bit. The borrow bit is used for subtraction of next higher significant bit. The digit from which other digit is subtracted is called minuend and the digit which is to be subtracted is called subtrahend. When minuend and subtrahend contains more significant digits, the borrow obtained from the subtraction of two bits is subtracted from the next higher order pair of significant bits. Subtractors is classified into two parts :

(*i*) Half Subtractor
(*ii*) Full Subtractor.

## 3.12 HALF SUBTRACTOR

In half subtractor, the subtraction of two bits are taking place and two outputs are produced *i.e.* Difference and Borrow. Let the input variables Minuend and subtrahend is denoted by *X* and *Y* and outputs. Difference and Borrow is denoted by *D* and *B*. The truth table for Half Subtractor is given as follows :

| *Input variables* | | *Output* | |
|---|---|---|---|
| ***X*** | ***Y*** | ***D*** | ***B*** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

The Boolean expressions for *D* and *B* is given as follows :

$$D = \bar{X}Y + X\bar{Y} = X \oplus Y$$
$$B = \bar{X}Y$$

The designing is as follows :

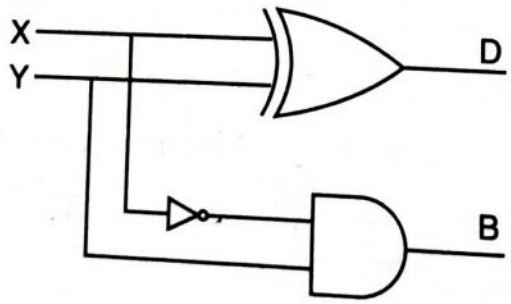


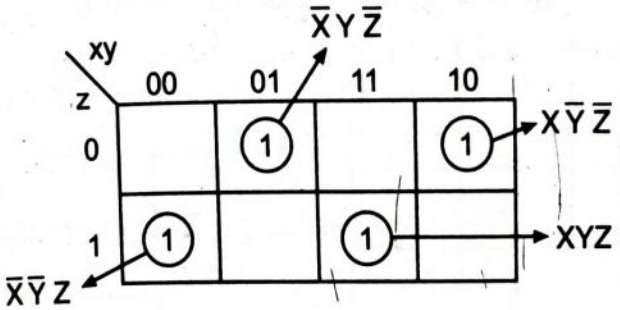

**Fig. 3.32.**

## 3.13 FULL SUBTRACTOR

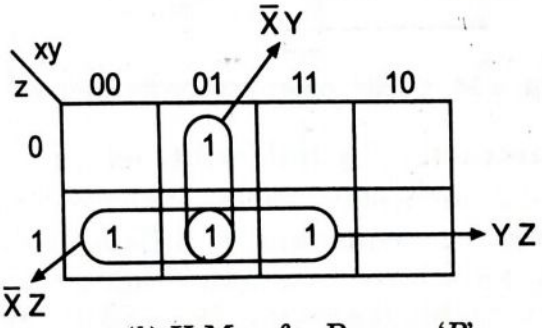
In full subtractor the subtraction of three bits *i.e.* minuend, subtrahend and borrow is taking place and two outputs difference and borrow is produced. Let us denote the minuend by *X*, subtrahend by *Y* and previous borrow by *Z* and outputs difference by *D* and borrow by *B*. The truth table corresponding to eight possible inputs are given as follows :

| Input Variables | | | Outputs | |
|---|---|---|---|---|
| X | Y | Z | D | B |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

The realization of the truth tables with the help of 'K' map. The 'K' map for Difference and Borrow is given as follows :




(*a*) K-Map for Difference '*D*'




(*b*) K-Map for Borrow '*B*'

Output expression for Difference and Borrow is given as

$$D = \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z + XYZ$$

$$B = \bar{X}Y + \bar{X}Z + YZ$$

## 3.14 REALIZATION OF LOGIC EXPRESSION OF D AND B BY GATES

The difference output requires three input AND gate and one 'OR' gate. Similarly in case of Borrow, it requires two input 'AND' and one 'OR' gate.

The designing is as follows :

$$D = \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z + XYZ$$

$$B = \bar{X}Y + \bar{X}Z + YZ$$

$\bar{X}Y(Z+\bar{Z}) + \bar{X}Z(Y+\bar{Y}) + YZ(X+\bar{X})$

$\bar{X}YZ + \bar{X}Y\bar{Z} + \bar{X}Z\bar{Y} + \bar{X}Z\bar{Y} + YZX + \bar{X}ZY$

$\bar{X}YZ + \bar{X}Y\bar{Z} + \bar{X}Z\bar{Y} + XYZ$

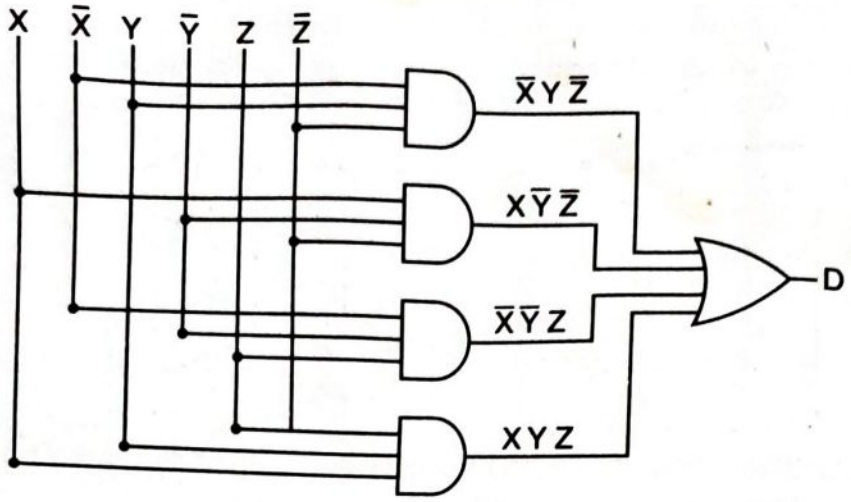



**Fig. 3.33.** Circuit diagram for difference.

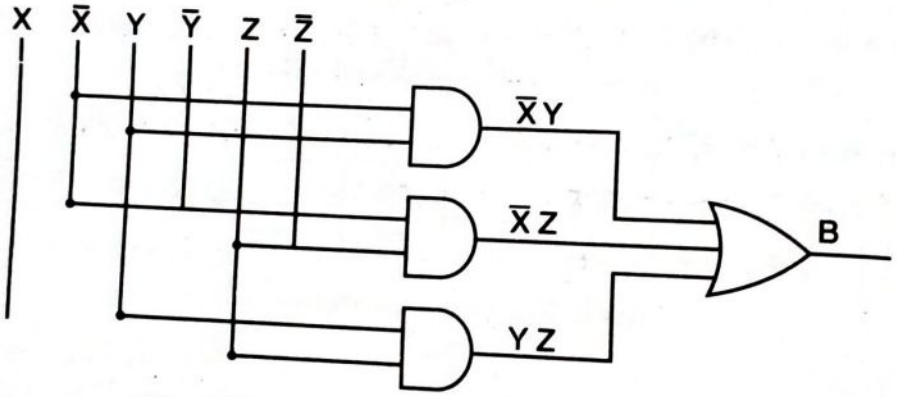



**Fig. 3.34.** Circuit representation for borrow.

### Designing of Full Subtractors by Half Subtractors

In order to design the circuit of full subtractor by Half Subtractor, we have to reduce the following logic equations of Difference and Borrow of full subtractors, by the gates which are used in Half Subtractors. The equations is modified first in order to realize the circuit. The equation are as follows :

$$D = \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z + XYZ$$

$$= \bar{X}Y\bar{Z} + \bar{X}\bar{Y}Z + X\bar{Y}\bar{Z} + XYZ$$

$$= \bar{X}(Y\bar{Z} + \bar{Y}Z) + X(\bar{Y}\bar{Z} + YZ)$$

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

$$B = \bar{X}Y + \bar{X}Z + YZ = \bar{X}Y + Z(\bar{X} + Y)$$

$$= \bar{X}Y + Z[\bar{X}(Y + \bar{Y}) + Y(X + \bar{X})]$$

$$= \bar{X}Y + Z(\bar{X}Y + \bar{X}\bar{Y} + YX + Y\bar{X}) \quad (\bar{X}Y + Y\bar{X} = \bar{X}Y)$$

$$= \bar{X}Y + Z(\bar{X}Y + \bar{X}\bar{Y} + XY)$$

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

$$= \overline{XY}(1 + Z) + Z\overline{(X \oplus Y)} = \bar{X}Y + Z\overline{(X \oplus Y)} \quad (1 + Z = 1)$$

$[\because \bar{X}\bar{Y} + XY = \overline{X \oplus Y}]$

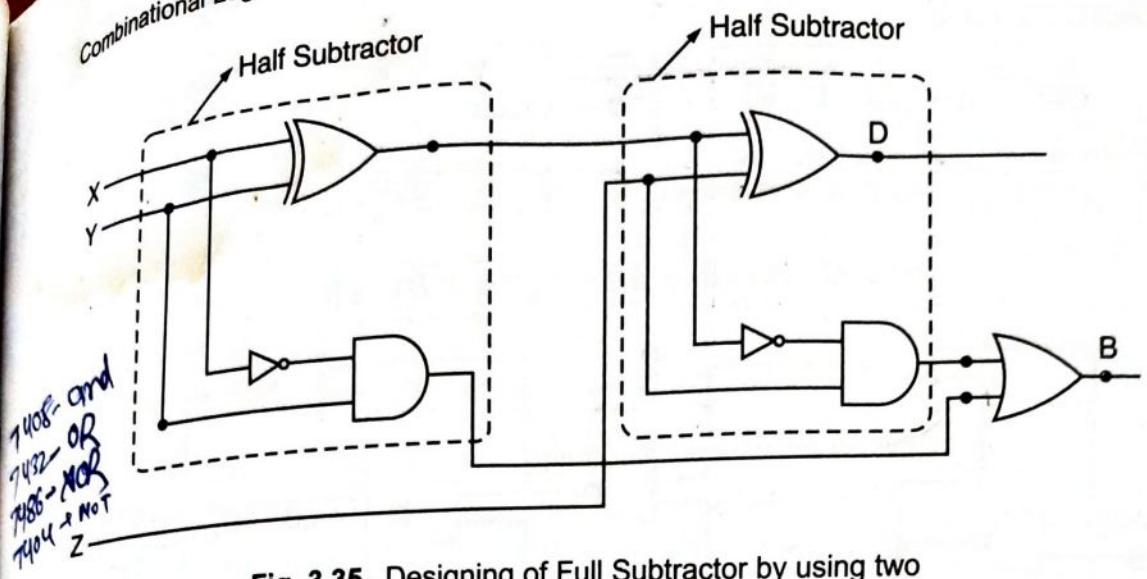



**Fig. 3.35.** Designing of Full Subtractor by using two half subtractor and OR gate.

### Designing of Full Adder using NAND gates and NOR gates

The logical expression for sum and carry is given as follows :

$$S = \bar{A}\,\bar{B}\,C_n + \bar{A}\,B\,\overline{C_n} + A\,\bar{B}\,\overline{C_n} + AB\,C_n = A \oplus B \oplus C_n$$

$$C = AB + AC_n + BC_n$$

### NOR Circuit

Let $A \oplus B = X = \overline{\overline{A + B} + \overline{\bar{A} \cdot \bar{B}}}$

$$S = A \oplus B \oplus C_n$$

$$= X \oplus C_n$$

$$= \overline{\overline{X + C_n} + \overline{\bar{X} \cdot \bar{C}_n}}$$

$$C = AB + C_n\,(A \oplus B)$$

$$= \overline{\overline{A + B}} + \overline{\overline{C_n} + \overline{A \oplus B}}$$

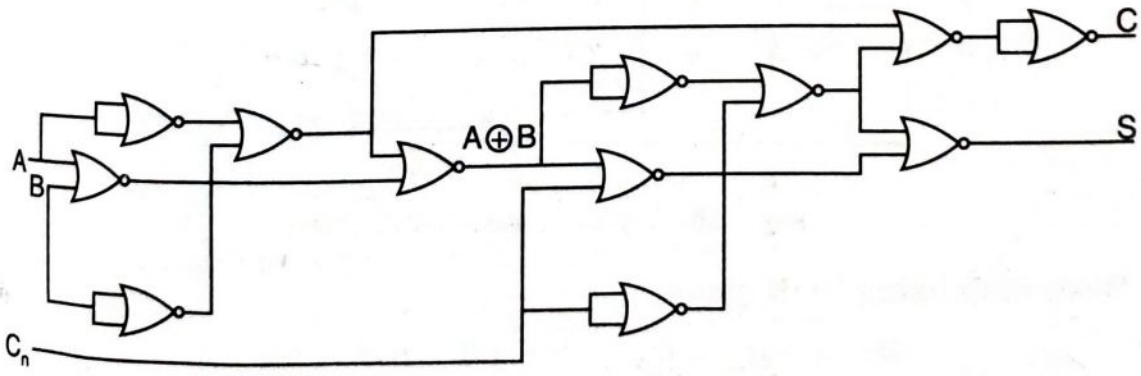



**Fig. 3.36.**