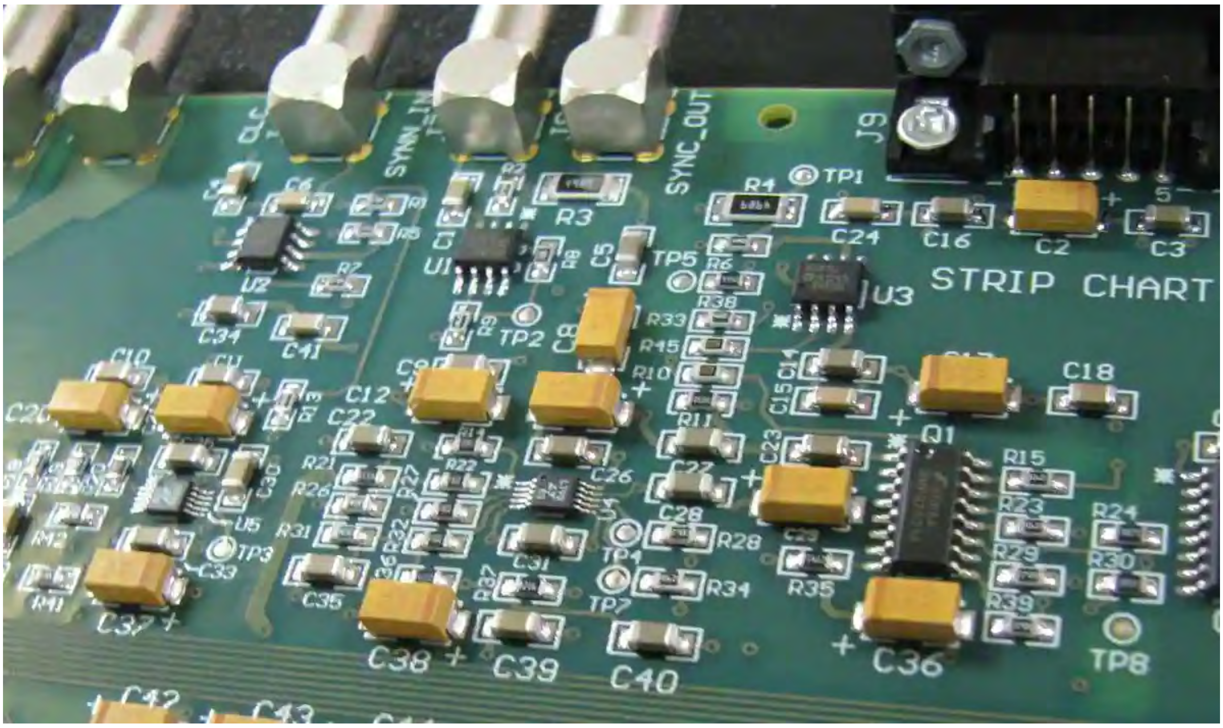


Pulse & Digital Circuits

Karnaugh Maps and Arithmetic Circuits

Karnaugh Maps & , Arithmetic Circuits

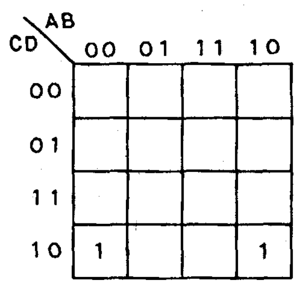
Karnaugh Maps

Karnaugh map (or K-map) provides a structured means of achieving maximum possible simplification of any logic function. This map is a sort of matrix of cells, where each cell corresponds to a unique combination of the set of literals. Thus for 4 variables (A, B, C, D) there are 2^4 =16 cells, which are arranged as per the Gray Code. This is illustrated in figure below. K-map can be used to obtain simplified logic functions either sop or pos forms directly.

SOP FORM SIMPLIFICATION USING K-MAP

In order to obtain simplified expression in sop form (AND-OR configuration), corresponding to each minterm in the given function, '1' is entered in the corresponding cell of the *K*-map.

Consider the term $\overline{B}, C, \overline{D}$. Enter 1 in the two cells with B = 0, C = 1 and D = 0 but A can be either 0 or 1. These entries are indicated in given figure. Such entries are made for all terms of a sum-of-products expression.

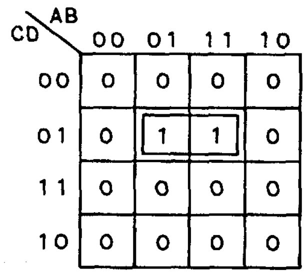


Simplification proceeds by combining 1's of adjacent cells. Two cells are said to be adjacent if (i) these are vertically above each other or are in the top and bottom cells of a column and (ii) these are horizontally side by side or in left and right most cells of a row. In combining adjacent cells it is to be noted that these differ in one variable only because of the use of the Gray code.

1's of the *K*-map are combined in groups of 2^i where i = 1, 2, …..n,; n being the number of variables. Various types of combinations for simplification are illustrated below.

The pair

The sum-of-products terms corresponding to a pair of adjacent 1's can be combined to eliminate the variable, which appears in complemented form in these terms; this results from the use of the Gray code.

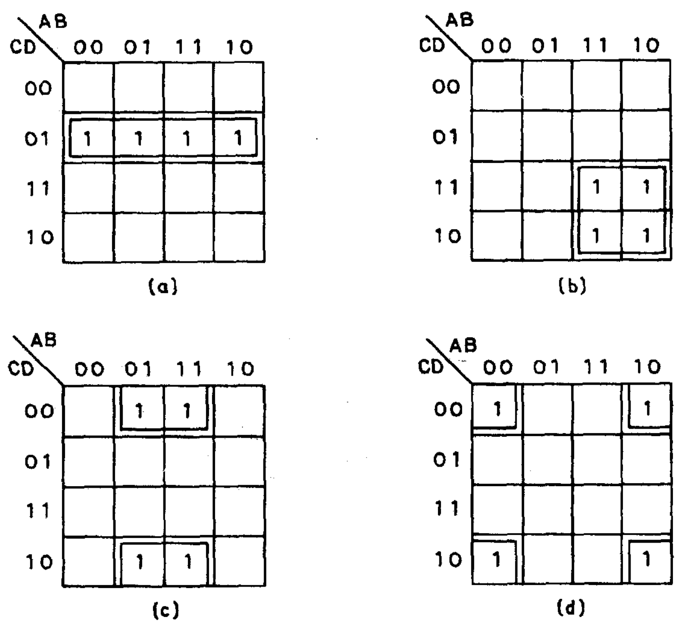


Thus in figure given above for the pair of 1's.

$$Y = \overline{A}B\overline{C}D + AB\overline{C}D = B\overline{C}D(\overline{A} + A) = B\overline{C}D;\ A \text{ is eliminated}$$

The quad

A quad is a group of four adjacent 1's in a K-map. It can appear in various ways as indicated in figure given below. This group called a quad which leads to elimination of two variables. Thus Y's corresponding to the four quads of figure given below as



Quads in K-map

$Y = \overline{C}D$ fig.(a)

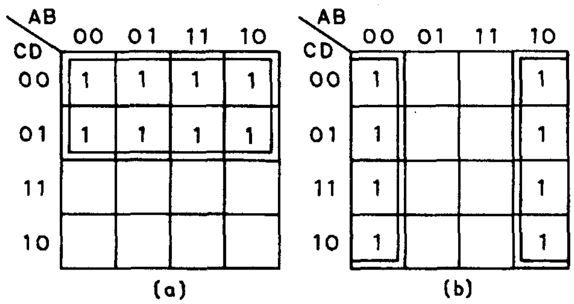
$Y = AC$ fig.(b)

$Y = B\overline{D}$ fig.(c)

$Y = \overline{B}\overline{D}$ fig.(d)

The octet

An octet is a group of eight adjacent 1's as shown in figure given below. It leads to elimination of three variables. Thus



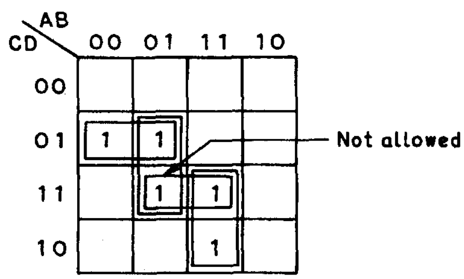
Octat in K-maps

$Y = \overline{C}$ fig.(a)

$Y = \overline{B}$ fig.(b)

It is to be noted that in a K-map there may be more than one pair or a quad or an octet. Because of simplification afforded identification must proceed first as octet, followed by quads and then pairs. The 1's which cannot be grouped must also be encircled. The Boolean equation is then obtained by ORing the products corresponding to the encircled groups.

While forming groups, it is to be noted that overlapping of groups is allowed, i.e., two groups can have one or more 1's in common. At the same time, redundancy is not allowed i.e. a group whose all 1's are overlapped by other groups. Both these grouping are shown in figure given below.



SUMMARY: K-MAP SIMPLIFICATION

1. Enter a 1 on the K-map for each fundamental product that corresponds to output 1 in the truth table. Blanks left out stand for 0's.
2. Encircle the 1's as octets, quads and pairs in that order. Encircle the isolated 1's also if any.
3. Eliminate redundant groups if they exist.
4. Write the Boolean equation by ORing the products.
5. Draw the equivalent logic circuit.

Σ AND π NOMENCLATURE

Σ and π notations are respectively used to represent sum-of-products and product-of-sums Boolean expressions. We will illustrate these notations with the help of examples. Let us consider the following Boolean function:

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

We will represent this function using Σ notation. The first step is to write the expanded sum-of-products given by

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

$$= A\bar{B}\bar{C}D + A\bar{B}\bar{C}\bar{D} + ABCD + \bar{A}B\bar{C}D + \bar{A}\bar{B}\bar{C}D$$

Different terms are then arranged in ascending order of the binary numbers represented by various terms, with true variables representing a 1 and a complemented variable representing a '0'. The expression becomes

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

The different terms represent 0001, 0101, 1000, 1001 and 1111. The decimal equivalent of these terms enclosed in the Σ then gives the Σ notation for the given Boolean function. That is, f(A, B, C, D) = Σ 1,5,8,9,15.

The complement of f(A, B, C, D), that is f'(A. B, C, D), can be directly determined from Σ notation by including the left-out entries from the list of all possible numbers for a four-variable function. That is,

$$f'(A,B,C,D) = \sum 0,2,3,4,6,7,10,11,12,13,14$$

Let us now take the case of a product-of-sums Boolean function and its representation in π nomenclature. Let us consider the Boolean function

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

The expanded product-of-sums form is given by

$$(A+B+\bar{C}+\bar{D})(\bar{A}+B+\bar{C}+\bar{D})(\bar{A}+\bar{B}+C+D)(A+\bar{B}+\bar{C}+\bar{D})$$

The binary numbers represented by the different sum terms are 0011, 1011, 1100 and 0111 (true and complemented variables here represent 0 and 1 respectively). When arranged in ascending order, these numbers are 0011, 0111, 1011 and 1100. Therefore,

$$f(A,B,C,D) = \pi 3,7,11,12 \; and \; f'(A,B,C,D) = \pi 0,1,2,4,5,6,8,9,10,13,14,15$$

An interesting corollary of what we have discussed above is that, if a given Boolean function f(A.B,C) is given by f(A,B, C) = Σ 0,1,4,7, then

f(A,B,C) = π 2,3,5,6 and f'(A,B,C) = Σ 2, 3, 5,6 = π 0,1,4,7

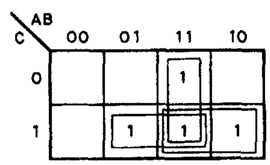
Example

For the truth table of table given below obtain the simplified sum-of-products expression using K-map and realize it using only NAND gates. Observe that this is the output of a majority voting circuit.

A	*B*	*C*	*Y*
0	*0*	*0*	*0*
0	*0*	*1*	*0*
0	*1*	*0*	*0*
0	*1*	*1*	*1*
1	*0*	*0*	*0*
1	*0*	*1*	*1*
1	*1*	*0*	*1*
1	*1*	*1*	*1*

Solution

The 3-variable K-map is shown in figure given above with 1's corresponding to Y = 1.



Here, an octet or a quad is not possible. But 3 pair are possible as shown. The sum-of-products expression is

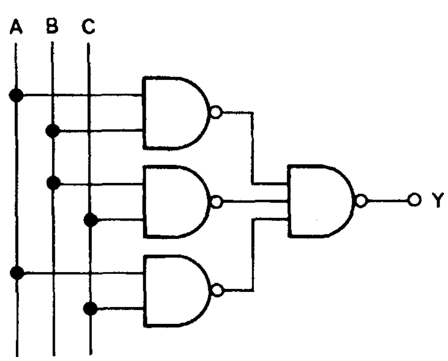
$$Y = AB + BC + AC$$

Note that this expression cannot be simplified any further. Applying De Morgan's theorem we can write the output as

$$\overline{Y} = \overline{AB + BC + AC} = \overline{AC}.\overline{BC}.\overline{AC}$$

$$Y = \overline{\overline{AB}.\overline{BC}.\overline{AC}}$$

Which can be synthesized using NAND gates only as shown in figure given below.



Example

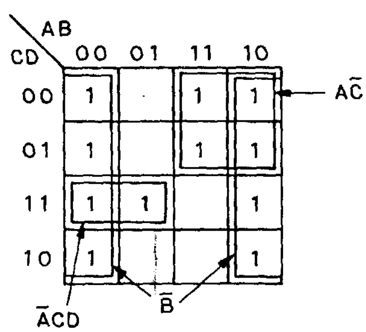
Determine the minimized sum-of-products expression for the truth-table given in table given below

A	*B*	*C*	*D*	*Y*
0	*0*	*0*	*0*	*1*
0	*0*	*0*	*1*	*1*
0	*0*	*1*	*0*	*1*
0	*0*	*1*	*1*	*1*
0	*1*	*0*	*0*	*0*
0	*1*	*0*	*1*	*0*
0	*1*	*1*	*0*	*0*
0	*1*	*1*	*1*	*1*
1	*0*	*0*	*0*	*1*
1	*0*	*0*	*1*	*1*
1	*0*	*1*	*0*	*1*
1	*0*	*1*	*1*	*1*
1	*1*	*0*	*0*	*1*
1	*1*	*0*	*1*	*1*
1	*1*	*1*	*0*	*0*
1	*1*	*1*	*1*	*0*

Solution

The 4-variables K-map variable is shown in fig given below with 1; entered at Y = 1. By following the minimization steps, we obtain the minimized expression as

$$Y = \overline{B} + A\overline{C} + \overline{A}CD$$



Example

Map the function

$$f(A,B,C,D) = A\overline{B}\overline{C}D + AB\overline{C}\,\overline{D} + \overline{A}\,C\overline{D} + \overline{A}\,\overline{C}\,\overline{D} + \overline{B}\,\overline{C}\,\overline{D}$$

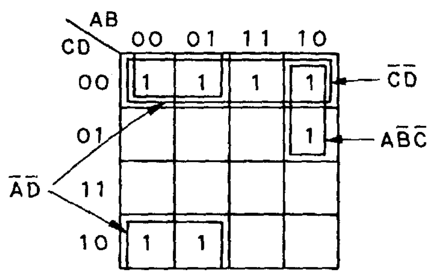
and obtain the minimal sum-of-products expression.

Solution

The K-map is drawn in figure given below with 1's entered corresponding to each of the terms in sum-of-product expression of f(A,B,C,D). Note that corresponding terms like ACD (three variables) the fourth variable can have any value (0/1) and two 1's are therefore entered, one for ACD with B = 0 and the other for B = 1.

Combining terms as dictated by encirclements, we can immediately write the simplified expression as

$$f(A,B,C,D) = \overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{D}$$



Example

Minimize the following expression using the map method:

$$(B\overline{C} + \overline{A}B + \overline{B}C\overline{D} + \overline{A}\overline{B}D + A\overline{B}\overline{C}D)$$

Solution

$$F = B\overline{C} + \overline{A}B + BC\overline{D} + \overline{A}\overline{B}D + A\overline{B}\overline{C}D$$

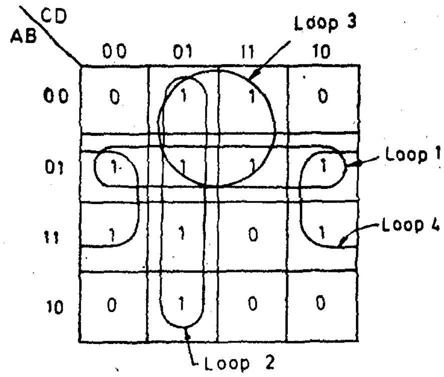
The expanded sum of products[1] is obtained as

$$F = AB\overline{C}D + AB\overline{C}\overline{D} + \overline{A}B\overline{C}D + AB\overline{C}\overline{D} + \overline{A}BCD + \overline{A}BC\overline{D} + ABC\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}D + A\overline{B}\overline{C}D$$

The above function is shown by given figure.

---

[1] In expanded sum pf products, each product term contains all variables, either uncomplemented or complemented. To obtain the expanded sum of product from a sum of products, the missing variables are added in all possible combinations to each product.



The function can be simplified by forming four loops.

The expressions for loops 1, 2, 3 and 4 are $\bar{A}B, \bar{C}D, \bar{A}D$ and $B\bar{C}$.

Problem

Minimize the 4-variable logic function using K-map

$$f = ACD + ABC + BD + CD + D$$

Answer: ABC + D

Problem

Minimize the following functions

(i) $f_1 = \sum m(1,4,6,9,10,11,14,15)$

(ii) $f_2 = \sum m(1,2,3,4,5,6,7,9,10,11,13,14,15)$

Answer: (i) $AC + \bar{A}B\bar{D} + \bar{B}\bar{C}D$ *(ii)* $C + D + \bar{A}B$

Hint: m = minterm defined by gray code. For the case for literal see gray code given in table of gray codes already discussed. For example from the table of gray codes

m(4) = 0110 = $\bar{A}BC\bar{D}$

Don't care conditions

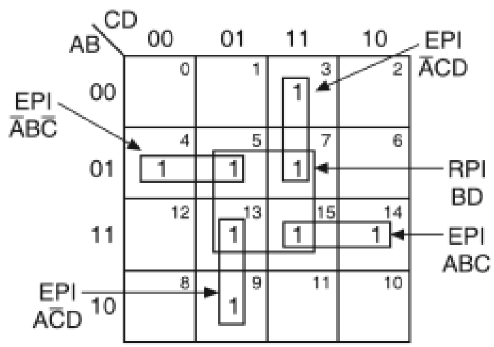
Sometimes we come across cases in which certain input combinations never exist. For instance, in BCD (binary coded decimal) systems the nibbles (1 nibble = 4 bits) 1010 to 1111 are never used. Also for some functions the output corresponding to certain combinations of input variables do not matter. In such situations, we represent the output as an x mark symbolizing don't-care. The corresponding entry can be used either as 0 or 1 depending upon whichever leads to a simpler expression. This is illustrated in the following example.

Prime Implicants, Essential Prime Implicants, Redundant Prime Implicants and Selective Prime Implicants

Each square or rectangle made up of the bunch of adjacent minterms is called a subcube. Each of these subcubes is called a *prime implicant (PI).*The prime implicant which contains at least one 1 which cannot be covered by any other prime implicant is called an *essential prime implicant (EPI)*. The prime implicant whose each 1 is covered at least by one EPI is called a *redundant prime implicant (RPI).* A prime implicant which is neither an essential prime implicant nor a redundant prime implicant is called a *selective prime implicant (SPI).*

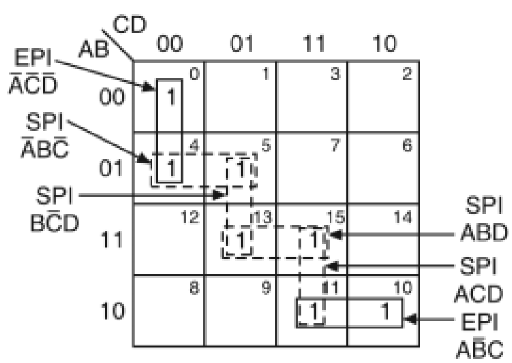
The function mapped in following figure has a unique MSP comprising EPIs given by

$$F(A,B,C,D) = \bar{A}CD + ABC + A\bar{C}D + \bar{A}B\bar{C}$$



The RPI 'BD' may be included without changing the function but the resulting expression would not be in minimal sum of products (MSP) form.

In following figure representing K(A, B, C, D) = Σm(0. 4, 5, 10, 11, 13, 15) SPIs are marked by dotted squares. This shows that the MSP form of a function need not he unique.



Example

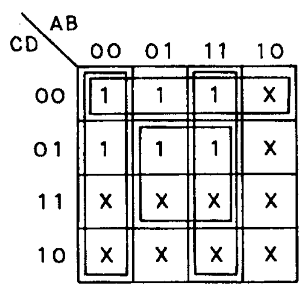
For the truth table of Table (given below) obtain the simplified Boolean equation.

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1

0	*0*	*1*	*0*	*x*
0	*0*	*1*	*1*	*x*
0	*1*	*0*	*0*	*1*
0	*1*	*0*	*1*	*1*
0	*1*	*1*	*0*	*x*
0	*1*	*1*	*1*	*x*
1	*0*	*0*	*0*	*x*
1	*0*	*0*	*1*	*x*
1	*0*	*1*	*0*	*x*
1	*0*	*1*	*1*	*x*
1	*1*	*0*	*0*	*1*
1	*1*	*0*	*1*	*1*
1	*1*	*1*	*0*	*x*
1	*1*	*1*	*1*	*x*

Solution

The K-map for table given below



This figure shows the most efficient way to encircle the groups. First, we visualize as many x's as possible to be 1's and try to form the largest groups that include the real 1's. This gives us 3 quads. Once, all the real 1's have been encircled, we visualize the remaining x's as 0's. In this way the x's are used to the best advantage. Care should be taken that a group should contain at least a single real 1. Given figure leads us to the simplified Boolean equation

$$Y = \overline{A}\,\overline{B} + BD + AB + \overline{C}\,\overline{D}$$

Example

Consider a digital system for minority logic. there are three inputs a , B and C. The output Y is equal to 1 if two or three inputs are 0.

(a) Write the truth-table.

(b) From the truth-table obtain the Boolean expression for Y.

(c) Minimize Y and draw the logic block diagram using NAND gates.

Solution

(a)

A	*B*	*C*	*Y*
0	*0*	*0*	*1*
0	*0*	*1*	*1*
0	*1*	*0*	*1*
0	*1*	*1*	*0*
1	*0*	*0*	*1*
1	*0*	*1*	*0*
1	*1*	*0*	*0*
1	*1*	*1*	*0*

(b) $$Y = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C}$$

(c) The Karnaugh map is drawn in figure (a). The minimal expression for output is given as

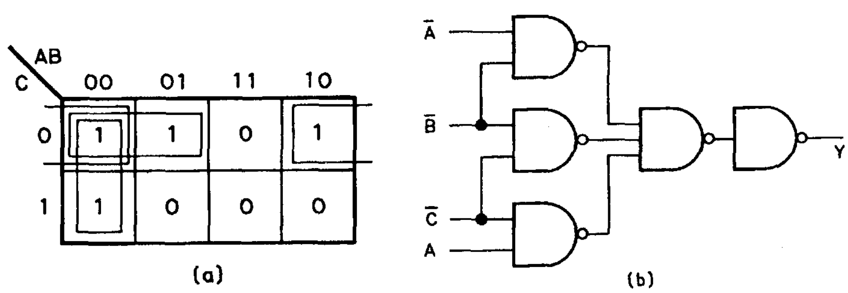
$$Y = \bar{A}\bar{B} + A\bar{C} + \bar{B}\bar{C}$$

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

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

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

The logic diagram is drawn in given figure (b).



Problem

For the logic equation

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

(a) Obtain the standard sum-of-products equation.

(b) Make a truth table

(c) *Simplify using K-map.*

(d) *Realize the simplified expression obtained from (c) using NAND gates.*

Answer: (a) $\sum m(4,6,7,9,11,12)$ *(c)* $\overline{A}B\overline{D} + A\overline{B}\overline{D} + ACD + B\overline{C}D$

Example

Simplify the following expressions:

(i) F(A, B, C) = Σ(2, 3, 5, 4),

(ii) F = Σ(3, 4, 6, 7),

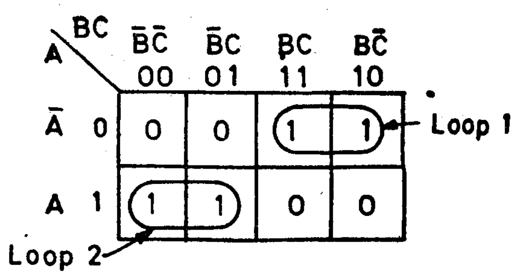
(iii) $F = \overline{A}C + \overline{A}B + A\overline{B}C + BC$,

(vi) $F = (A+B+C)(A+B+\overline{C})(A+\overline{B}+C)(\overline{A}+B+\overline{C})(\overline{A}+B+\overline{C})$

Solution

(i) F = Σ(2, 3, 4, 5) = $\overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C} + A\overline{B}C$.

Karnaugh map in figure represents the above function.

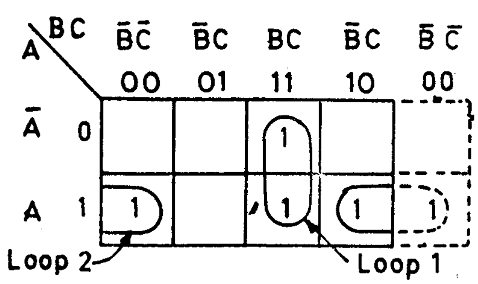


The l's can be looped as shown. For loop l, the expression is $\overline{A}B$ as C can be eliminated. Variable C can be eliminated from expression for loop 2 also.

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

(ii) F = Σ(3, 4, 6, 7) $= \overline{A}BC + A\overline{B}\overline{C} + AB\overline{C} + ABC$

Karnaugh map in given figure represents the above function.



Within loop l, variable A changes and can be eliminated from the expression for loop l. Loop 2 formed by looping l's in the corner squares of the bottom row. The first column is in fact a column adjacent to the last column as only one variable, namely B, has different values for these two columns. We can draw the first column besides the last column. This arrangement is shown in dotted lines in the figure. Variable B is not constant within loop 2 and can hence be eliminated and the expression for loop 2 is $A\overline{C}$. The simplified expression is given by

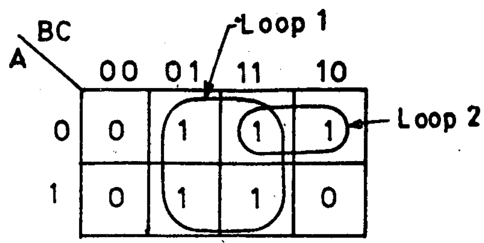
$$F = BC + A\overline{C}$$

(iii) $$F = \overline{A}C + \overline{A}B + A\overline{B}C + BC.$$

The above expression is expressed first as the expanded sum of products.

$$F = \overline{A}BC + \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}C + ABC.$$

Karnaugh map in given figure represents the above function.

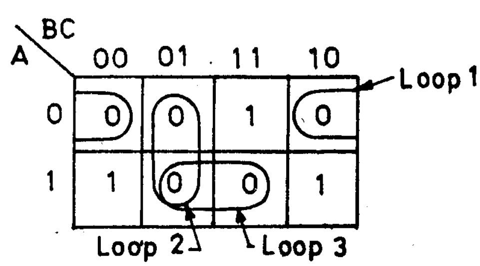


Here loop l contains four l's and loop 2 two l's. Four squares lying adjacent can be grouped as shown. Within loop l, both A and B are not constant and can be eliminated from the expression for loop l. If two squares are grouped, one variable can be eliminated. If four squares are looped, two variables can be eliminated. The variables that are eliminated are those which vary for the minterms represented by the squares within the loop. Here the expression for loop l is as both A and B can be eliminated. The expression for loop 2 is $\overline{A}B$. Hence

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

(iv) $$F = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + \overline{C})(\overline{A} + \overline{B} + \overline{C}).$$

The squares corresponding to the maxterms of the above expression are marked in given figure.



We can loop adjacent 0's and form the simplified product of sums expression. There are three loops marked in the figure. For loop l, the variable B can be eliminated. The simplified sum term is then given by (A + C). For loop l, the simplified product term is $\bar{A}\bar{C}$. We can get the sum term by complementing $\bar{A}\bar{C}$. The complement of $\bar{A}\bar{C}$ is (A +C). Similarly, the sum terms for loop 2 and loop 3 are $(B+\bar{C})$ and $(\bar{A}+\bar{C})$. The simplified expression in the form of product of sums is given by

$$F=(A+C)(B+\bar{C})(\bar{A}+\bar{C}).$$

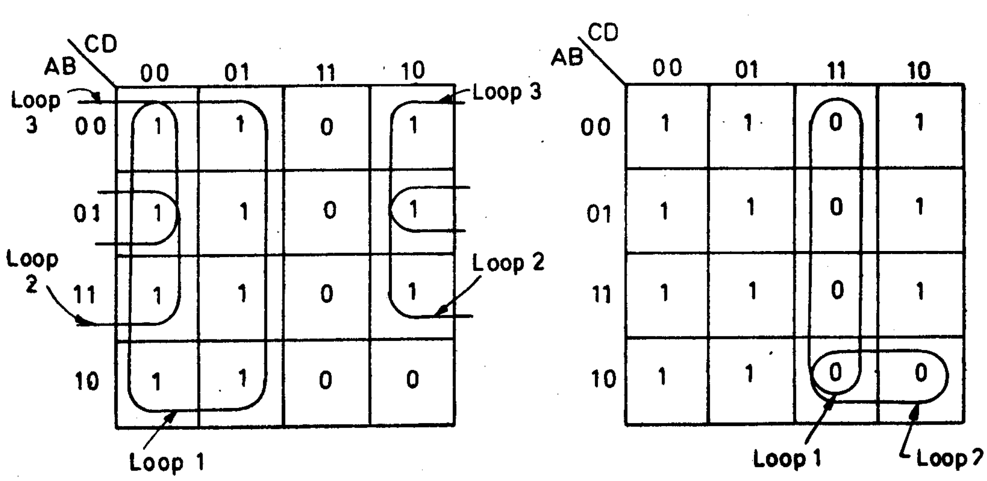
Example

Simplify the function F(A, B, C, D) where F = Σ(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14).

Solution

The map in Fig. (a) shows the given function. It is seen that there that there are 3 loops. Loop 1 contains 8 squares and the variables A, B and D are not constant within loop l and they are eliminated. Hence

Expression for loop l = $\bar{C}$.



(a) (b)

Loop 2 contains 4 variables. It is known that the first column and the last column vary only in one variable and the squares in the first and the last columns can be combined if they are of the same row. Hence

Expression for loop $2 = B\bar{D}$.

Similarly,

Expression for loop $3 = \bar{A}\bar{D}$. The simplified function is $(\bar{C}+B\bar{D}+\bar{A}\bar{D})$.

The same map is reproduced in fig. (b).

Here we solve the problem by looping zeroes. There are 5 zeroes and two loops can be formed. Equation for loop l is $(\bar{C}+\bar{D})$. expression for loop 2 is $(\bar{A}+B+\bar{C})$. Hence

$$F = \left(\bar{C} + \bar{D}\right)\left(\bar{A} + B + \bar{C}\right).$$

By multiplying out, we get

$$F = \bar{A}\bar{C} + \bar{A}\bar{D} + B\bar{C} + B\bar{D} + \bar{C}$$

$$= \bar{C}\left(1 + \bar{A} + B\right)\bar{A}\bar{D} + B\bar{D}$$

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

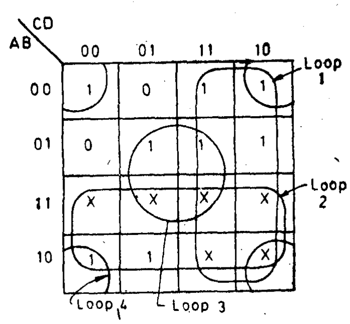
We get the same expression as that obtained by simplifying Fig. (a). Of course, we must get the same expression.

Example

Write the Boolean function f(A, B, C, D) = Σ(0, 2, 3, 5, 6, 7, 8, 9) with 10, 11, 12, 13, 14, 15 as don't cares.

Solution

The given function and the don't care terms are marked in given figure.



We can form four loops. Loop l is expressed by the function C, loop 2 by A, loop 3 by BD and 4 by $\bar{B}\bar{D}$. The simplified function is given by

$$F = A + C + BD + \bar{B}\bar{D}.$$

Problem

Simplify

(i) $F(A, B, C) = \Sigma(5, 7)$

(ii) $F = \Sigma(0, 2, 4, 5, 6)$

Answer: (i) $F = AC$ (ii) $F = A\bar{B} + \bar{C}$

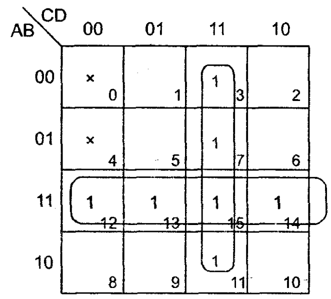
Example (AMIE Summer 2010, 6 marks)

Minimise the following switching function on a Karnaugh map

$$F = \sum(3, 7, 11, 12, 13, 14, 15) + \sum_d (0, 4)$$

Solution

Karnaugh map is given below



Reduced function is

$$F = AB + CD$$

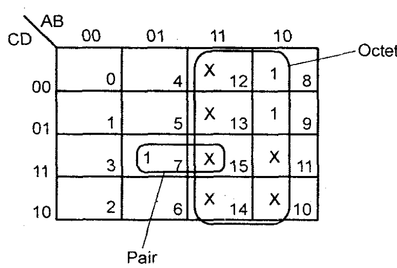
Example (AMIE, Winter 2010, 10 marks)

Simplify the following using K map:

$$f(A,B,C,D) = \sum m(7,8,9) + \sum d(10,11,12,13,14,15)$$

Solution

See following K map



On making groupings, we get one octet and one pair. Hence simplified function will be

$$f = A + BCD$$

LIMITATIONS OF KARNAUGH MAP

The map method of simplification is convenient as long as the number of variables does not exceed five or six. As the number of variables increases it is difficult to make judgements about which combinations form the minimum expression. In case of complex problem with 7, 8, or even 10 variables it is almost an impossible task to simplify expression by the mapping method. Another important point is that the K-map simplification is manual technique and simplification process is heavily depends on the human abilities. To meet this need, W. V. Quine and E. J. McCluskey developed an exact tabular method to simplify the boolean expression. This method is called the Quine-McCluskey, or tabular method.

THE TABULATION METHOD

The Karnaugh map method is a very useful and convenient tool for simplification of Boolean functions as long as the number of variables does not exceed four (at the most six). But if the number of variables increases, the visualization and selection of patterns of adjacent cells in the Karnaugh map becomes complicated and difficult. The tabular method, also known as the Quine-McCluskey method, overcomes this difficulty. It is a specific step-by-step procedure to achieve guaranteed, simplified standard form of expression for a function.

The following steps are followed for simplification by the tabular or Quine-McCluskey method.

1. An exhaustive search is done to find the terms that may be included in the simplified functions. These terms are called *prime implicants*.
2. Form the set of prime implicants, essential prime implicants are determined by preparing a prime implicants chart.
3. The minterms that are not covered by the essential prime implicants, are taken into consideration by selecting some more prime implications to obtain an optimized Boolean expression.

Determination of Prime Implicants

The prime implicants are obtained by the following procedure:

1. Each minterm of the function is expressed by its binary representation.
2. The minterms are arranged according to increasing index (index is defined as the number of 1s in a minterm). Each set of minterms possessing the same index are separated by lines.
3. Now each of the minterms is compared with the minterms of a higher index. For each pair of terms that can combine, the new terms are formed. If two minterms are differed by only one variable, that variable is replaced by a '-' (dash) to form the new term with one less number of literals. A line is drawn in when all the minterms of one set is compared with all the minterms of a higher index.
4. The same process is repeated for all the groups of minterms. A new list of terms is obtained after the first stage of elimination is completed.
5. At the next stage of elimination two terms from the new list with the '-' of the same position differing by only one variable are compared and again another new term is formed with a less number of literals.
6. The process is to be continued until no new match is possible.
7. All the terms that remain unchecked i.e., where no match is found during the process, are considered to be the prime implicants.

Prime Implicant Chart

1. After obtaining the prime implicants, a chart or table is prepared where rows are represented by the prime implicants and the columns are represented by the minterms of the function.
2. Crosses are placed in each row to show the composition of the minterms that makes the prime implicants.
3. A completed prime implicant table is to be inspected for the columns containing only a single cross. Prime implicants that cover the minterms with a single cross are called the essential prime implicants.

Example

Obtain the minimal sum of the products for the function

$$F(A, B, C, D) = \Sigma(1, 4, 6, 7, 8, 9, 10, 11, 15).$$

Solution

The table in given figure shows the step-by-step procedure the Quine-McCluskey method uses to obtain the simplified expression of the above function.

I	*II*					*III*			*IV*	
Decimal equivalent	*Binary equivalent* *A*	*B*	*C*	*D*		*ABCD*			*ABCD*	
1	0	0	0	1	√	1,9	–001		8,9,10,11	10– –
4	0	1	0	0	√	4,6	01–0		8,10,9,11	10– –
8	1	0	0	0	√	8,9	100–	√		
						8,10	10–0	√		
6	0	1	1	0	√	6,7	011–			
9	1	0	0	1	√	9,11	10–1	√		
10	1	0	1	0	√	10,11	101–	√		
7	0	1	1	1	√	7,15	–111			
11	1	0	1	1	√	11,15	1–11			
15	1	1	1	1	√					

Column I consists of the decimal equivalent of the function or the minterms and column II is the corresponding binary representation. They are grouped according to their index i.e., number of 1s in the binary equivalents. In column III, two minterms are grouped if they are differed by only a single variable and equivalent terms are written with a '-' in the place where the variable changes its logic value. As an example, minterms 1 (0001) and 9 (1001) are grouped and written as 1,9 {- 001) and so on for the others. Also, the terms of column II, which are considered to form the group in column III, are marked with √.

The terms which are not marked with √ are the *Prime implicants*. To express the prime implicants algebraically, variables are to be considered as true form in place of 1s, as complemented form in place of 0s, and no variable if '-' appears. Here the prime implicants are B'CD, A'BD', A'BC, BCD, ACD (from column III), and AB' (from column IV). So the Boolean expression of the given function can be written as

F = AB' + B'CD + A'BD' + A'BC + BCD + ACD

But the above expression *may not be* of minimized form, as all the prime implicants may not be necessary. To find out the essential prime implicants, the following steps are carried out. A table or chart consisting of prime implicants and the decimal equivalent of minterms as given in the expression, is prepared.

Prime Implicants	*1*	*4*	*6*	*7*	*8*	*9*	*10*	*11*	*15*
√ AB′					X	X	X	X	
√ B′C′D	X					X			
√ A′BD′		X	X						
A′BC			X	X					
BCD				X					X
ACD								X	X
	√	√	√		√	√	√	√	

In the table, the prime implicants are listed in the 1st column and Xs are placed against the corresponding minterms. The completed prime implicant table is now inspected for the columns containing only a single X. As in above figure, the minterm 1 is represented by only a single prime implicant B'CD, and only a single X in that column, it should be marked as well as the corresponding column should be marked. Similarly, the prime implicants AB' and ABTD' are marked. These are the essential prime implicants as they are absolutely necessary to form the minimized Boolean expression.

Now all the other minterms corresponding to these prime implicants are marked at the end of the columns i.e., the minterms 1, 4, 6, 8, 9, 10, and 11 are marked. Note that the terms A'BC, BCD, and ACD are not marked. So they are not the essential prime implicants. However, the minterms 7 and 15 are still unmarked and both of them are covered by the term BCD and are included in the Boolean expression. Therefore, the simplified Boolean expression of the given function can be written as

F = AB' + B'CD + A'BD' + BCD

This simplified expression is in the *sum of products form*. The Quine-McClusky method can also be adopted to derive the simplified expression in *product of sums form*. In the Karnaugh map method the complement of the function was considered by taking 0s from the initial list of the minterms.

Similarly the tabulation method or Quine-McClusky method may be carried out by considering the 0s of the function to derive the sum of products form. Finally, by making the complement again, we obtain the simplified expression in the form of product of sums.

Example

Obtain the minimal sum of the products for the function F (A,B,C,D) = Σ(1, 2, 3, 7, 8, 9, 10, 11, 14, 15) by the Quine-McClusky method.

Solution

The first step is to find out the prime implicants as described by the table in following figure.

I	*II*					*III*			*IV*	
Decimal equivalent	*Binary equivalent* *A*	*B*	*C*	*D*			*ABCD*			*ABCD*
1	0	0	0	1	√	1,3	00–1	√	1,3,9,11	–0–1
2	0	0	1	0	√	1,9	–001	√	2,3,10,11	–01–
8	1	0	0	0	√	2,3	001–	√	8,9,10,11	10– –
						2,10	–010	√	3,7,11,15	– –11
						8,9	100–	√	10,11,14,15	1–1–
						8,10	10–0	√		
3	0	0	1	1	√	3,7	0–11	√		
9	1	0	0	1	√	3,11	–011	√		
10	1	0	1	0	√	9,11	10–1	√		
						10,11	101–	√		
						10,14	1–10	√		
7	0	1	1	1	√	7,15	–111	√		
11	1	0	1	1	√	11,15	1–11	√		
14	1	1	1	0	√	14,15	111–	√		
15	1	1	1	1	√					

The prime implicants are BD, B'C, AB', CD, and AC. The prime implicant table is prepared as in following figure.

Prime Implicants	1	2	3	7	8	9	10	11	14	15
√ B′D	X		X			X		X		
√ B′C		X	X				X	X		
√ AB′					X	X	X	X		
√ CD			X	X				X		X
√ AC							X	X	X	X
	√	√	√	√	√	√	√	√	√	√

All the prime implicants are essential. So the simplified Boolean expression of the given function is

$$P = B'D + B'C + AB' + CD + AC$$

Example

Using the Quine-McClusky method, obtain the minimal sum of the products expression for the function F(A, B, C, D) = X (1, 3, 4, 5, 9, 10, 11) + ϕ(6, 8).

Solution

The prime implicants are obtained from the table in following figure.

I	*II*					*III*			*IV*	
Decimal equivalent	*Binary equivalent* *A*	*B*	*C*	*D*			*ABCD*			*ABCD*
1	0	0	0	1	√	1,3	00–1	√	1,3,9,11	–0 –1
4	0	1	0	0	√	1,5	0–01		8,9,10,11	10– –
8	1	0	0	0	√	1,9	–001	√		
						4,5	010–			
						4,6	01–0			
						8,9	100–	√		
						8,10	10–0	√		
3	0	0	1	1	√	3,11	–011	√		
5	0	1	0	1	√	9,11	10–1	√		
6	0	1	1	0	√	10,11	101–	√		
9	1	0	0	1	√					
10	1	0	1	0	√					
11	1	0	1	1	√					

The prime implicants are A'CD, ABC, A'BD', B'D, and AB'. The prime implicant table is prepared as in following figure.

Prime Implicants	*1*	*3*	*4*	*5*	*9*	*10*	*11*
A′C′D	X			X			
A′BC′			X	X			
A′BD′			X				
√ B′D	X	X			X		X
√ AB′					X	X	X
	√	√			√	√	√

From the table, we obtain the essential prime implicants B'D and AB'. The minterms 4 and 5 are not marked in the table. The term A'BC is considered, which covers both the minterms 4 and 5. So the simplified Boolean expression for the given function is

P = ABC + B'D + AB'

Arithmetic Circuits

A logic gate is an electronic circuit, which accepts a binary input and produces a binary output namely 0 and 1. The inverter (NOT) logic gate has one input and one output, but a logic gate in general accepts one or more inputs and produces one output. Apart from the NOT gate there are six other types of logic gates.

Input to a gate will be designated by binary variables A, B, C etc. and the output will be indicated by binary variable Y. As stated earlier, a binary variable can take on values 0 and 1 which are electronically represented by LOW and HIGH voltage levels. In terms of boolean algebra the function of a logic gate will be represented by a binary expression.

Basic Circuits have been discussed in the chapter " Boolean Algebra & Karnaugh Maps".

HALF ADDER

This circuit adds two binary variables, yields a carry but does not accept carry from another circuit(adder). The truth table of half adder is given in below.

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

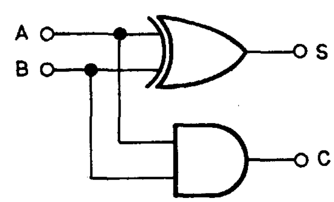
From this table

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

$$C = AB$$

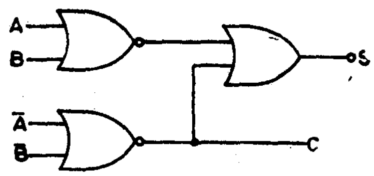
S is *Sum* and C is *Carry*.

Half adder logic circuit is shown in given figure.



Circuit using XOR

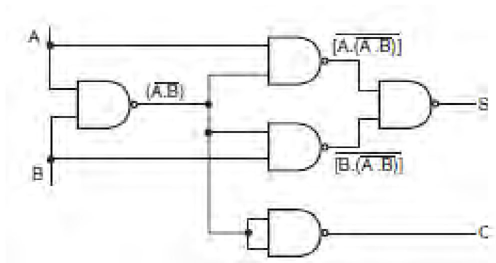
Half adder circuit using NOR gates is shown in given figure.



Half adder circuit using NOR gates

Here $S = \bar{S} = \overline{AB + \bar{A}\bar{B}} = \overline{(\bar{A}+\bar{B}) + (\bar{A}+\bar{B})}$

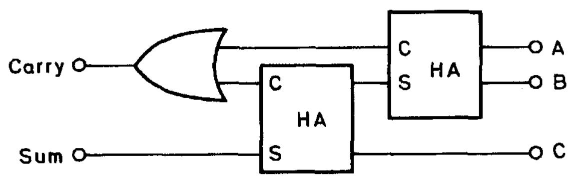
Half adder circuit using NAND gates is shown in given figure



A close look at the logic diagram of above figure reveals that one part of the circuit implements a two-input EX-OR gate with two-input NAND gates.

FULL ADDER

This circuit can add two binary numbers, accept a carry and yield a carry. Such a circuit can easily be visualized by means of two half adders(HA) and an OR as in given figure.



Full adder using two half adders

To synthesize the full adder circuit, we should proceed from the truth table as in given table.

A	B	C_i	S	C_0
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

It can be immediately written from this table that

$$S = \bar{A}\bar{B}C_i + \bar{A}B\bar{C}_i + A\bar{B}\bar{C}_i + ABC_i \quad (1)$$

and $$C_0 = \bar{A}BC_i + A\bar{B}C_i + AB\bar{C}_i + ABC_i \quad (2)$$

Recognizing that $Y + Y + Y + ... + Y = Y$ and adding ABC_i twice to right hand side of eq. (2), we can write

$$C_0 = (\bar{A}BC_i + ABC_i) + (A\bar{B}C_i + ABC_i) + (AB\bar{C}_i + ABC_i)$$

Observing that $Y + \bar{Y} = 1$, we get

$$C_0 = BC_i + AC_i + AB \quad (3)$$

The following is easily established

$$S = [(\overline{AC_i + BC_i + AB}) + ABC_i](A + B + C_i) \quad (4)$$

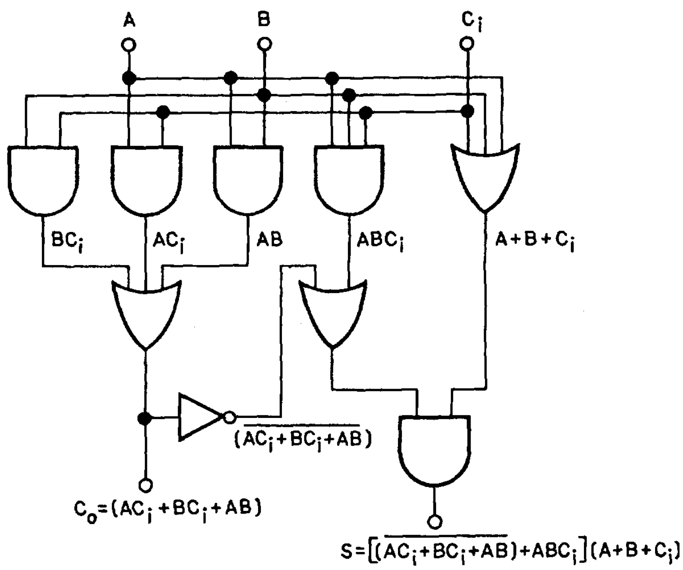
$$= [(\overline{AC_i}.\overline{BC_i}.\overline{AB}) + ABC_i](A + B + C_i)$$

$$= (\overline{AC_i}.\overline{BC_i}.\overline{AB})(A + B + C_i) + ABC_i$$

$$= (\overline{A} + C_i)(\overline{B} + \overline{C_i})(\overline{A} + \overline{B})(A + B + C_i) + ABC_i$$

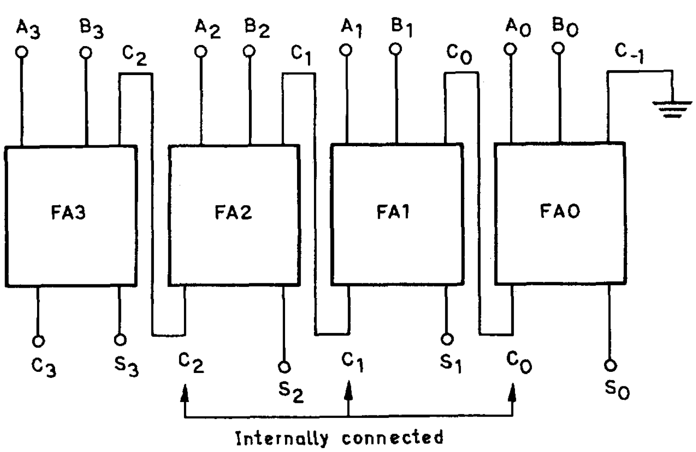
$$= \overline{A}\overline{B}C_i + \overline{A}B\overline{C_i} + A\overline{B}\overline{C_i} + ABC_i = S$$

Using AND OR INVERT(AOI) gates, equations (3) and (4) are implemented in figure given, which gives outputs S and C_0.



Full Adder

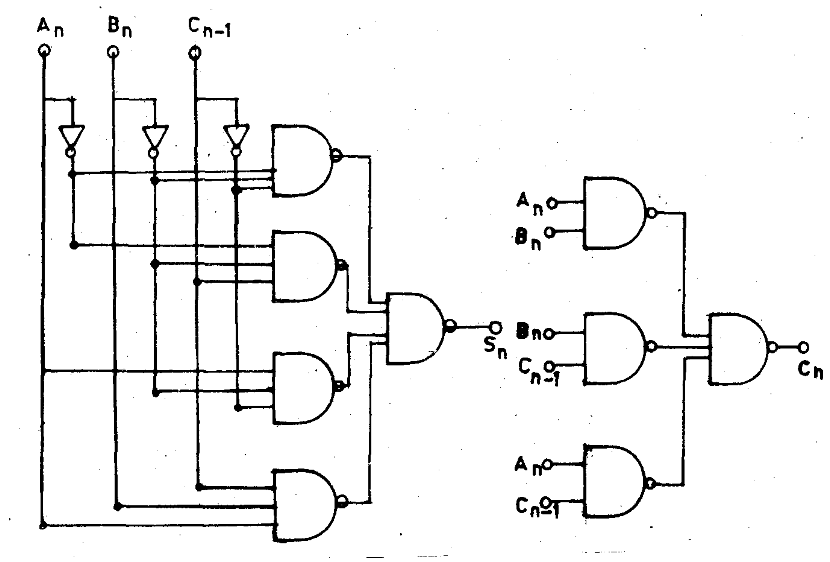
The schematic diagram of 4-bit parallel adder is shown in given figure which is self explanatory.



4 -bit parallel binary adder(cascaded)

FULL ADDER USING NAND GATES

See following diagram for full adder using NAND gates.



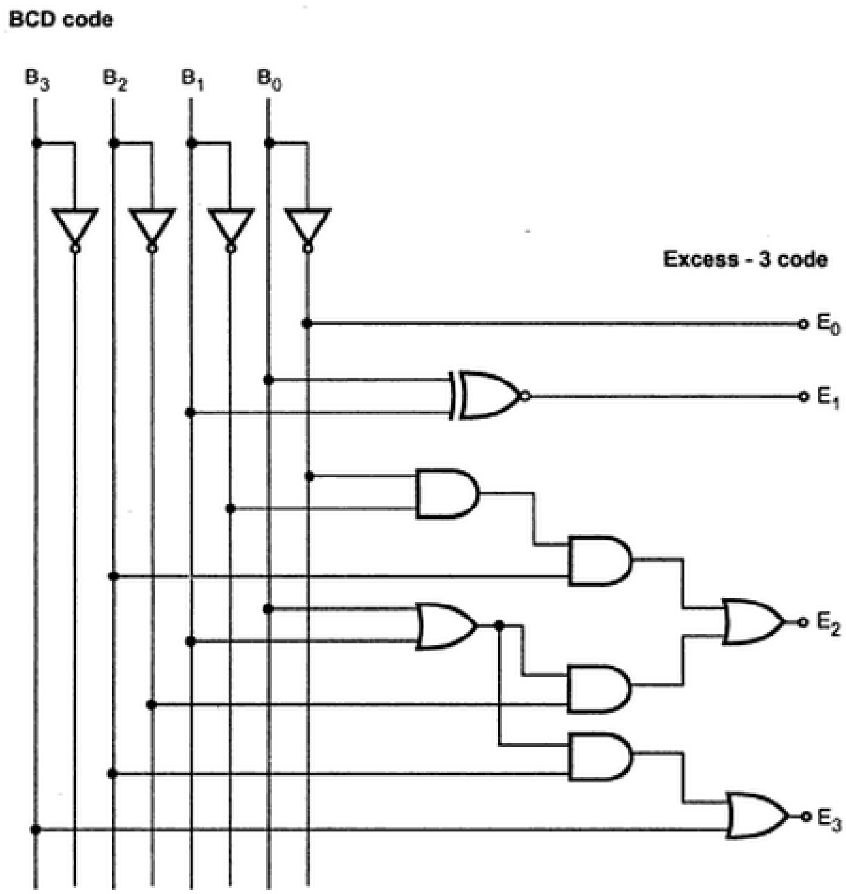
BCD TO EXCESS-3 CONVERTER

Exccss-3 code is a modified form of a BCD number. The Excess-3 code can be derived from the natural BCD code by adding 3 to each coded number. For example, decimal 12 can be represented in BCD as 0001 0010. Now adding 3 to each digit we get Excess-3 code as 0100 0101 (12 in decimal).

Truth table for BCD to Excess-3 code converter is given below.

Decimal	B_3	B_2	B_1	B_0	E_3	E_2	E_1	E_0
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

Logic diagram is given below.


BCD code
B3
B2
B1
B0
Excess - 3 code
E0
E1
E2
E3

ASSIGNMENT

KARNAUGH MAPS

Q.1. (AMIE S13, 6 marks): Define the following:

(i) Karnaugh map

(ii) Quine McClusky table

(iii) Negative OR logic gate

Q.2. (AMIE S15, 2 marks): State the limitations of Karnaugh maps.

Q.3. (AMIE W14, 10 marks): Minimise the expression $f = AB + \bar{A}\bar{B}CD$ using Karnaugh map method.

Q.4. (AMIE W12, 8 marks): Minimise the following switching functions using Karnaugh map. List all prime implicants and essential prime implicants (non redundant group).

(i) F = Σ(1, 3, 5, 6, 7)

(ii) F = Σ(0, 1, 3, 6, 14, 15)

Answer: (i) F = AB + C, AB and C are prime implicants. Both are also essential prime implicants.

(ii) $F = \bar{A}\bar{B}C + A\bar{B}D + BC\bar{D} + ABC$, here all these four terms are prime implicants and also essential prime implicants.

Q.5. (AMIE S10, 10 marks): Simplify the following using K-map

$$f(A,B,C,D) = \sum m(7,8,9) + \sum d(10,11,12,13,14,15)$$

Answer: f = A + BCD

Q.6. (AMIE S10, 4 marks): A truth table has a low output for the first three input conditions: 000, 001 and 010. If all other outputs are high, what are the product of sum (POS) form represented by the truth table?

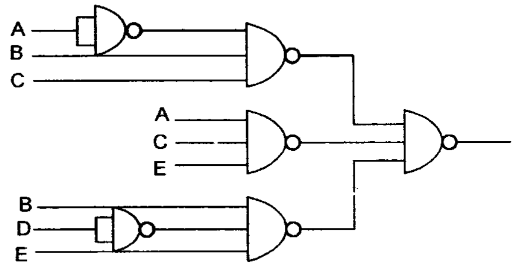
Answer: $(A+B+C)(A+B+\bar{C})(A+\bar{B}+C)$, simplified form is (A+B)(A+C)

Q.7. (AMIE W10, 10 marks): Simplify the following using K map:

F(A,B,C,D,E) = Σ(0,2,4,6,9,13,21,23,25,29,31)

Implement using only NAND gates.

Answer: $F = \bar{A}B\bar{E} + ACE + B\bar{D}E$

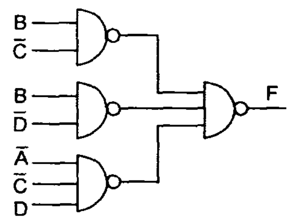


Q.8. (AMIE S11, 10 marks): Reduce the expression using K map in SOP and POS forms

Σm(1, 5, 6, 12, 13, 14) + d(2, 4)

Also, implement the minimal expression in universal logic.

Answer: SOP form is $F = B\overline{C} + B\overline{D} + \overline{A}\overline{C}D$; POS form is $F = (\overline{A} + B)(A + C + D)(\overline{C} + \overline{D})$



Q.9. (AMIE W11, S15, 12 marks): Simplify the following Boolean expression using Karnaugh map method. Implement the simplified expression using NAND gates only:

f = Σm (0, 1, 2, 5, 7, 9, 12, 14, 21, 23, 26, 29) + Σd(6, 15, 28, 30, 31)

Answer: $f = \overline{W}XZ + WXY + WX\overline{Y}\overline{Z} + \overline{VWXY} + \overline{VW}Y\overline{Z} + \overline{VXY}Z + VWY\overline{Z}$

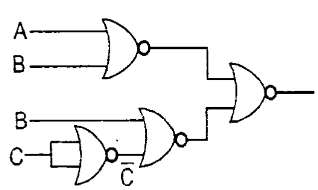
Q.10. (AMIE S13, 10 marks): Minimise the following switching function using Karnaugh map method. Implement the function using NAND gates only.

f(A, B, C, D, E) = Σm(0, 1, 4, 9, 11, 16, 22, 25, 31) + Σd(3, 10, 24, 30)

Answer: $\overline{A}\overline{B}\overline{D}\overline{E} + \overline{B}\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{D} + ACD\overline{E} + ABCD + B\overline{C}\overline{D}E + \overline{A}B\overline{C}D$

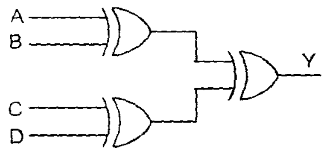
Q.11. (AMIE S12, 5 marks): Implement πM (0, 1, 2, 3, 10, 11) using only NOR gates in simplified form.

Answer: $(A + B)(B + \overline{C})$



Q.12. (AMIE S12, 10 marks): Identify the function Y = Σ m(1, 2, 4, 7, 8, 11, 13, 14) in simplest form and implement using only three logic gates.

Answer: $(A \oplus B) \oplus (C \oplus D)$



QUINE McCLUSKY METHOD

Q.13. (AMIE S15, 8 marks): Define the terms “prime implement”, “non prime implement”, essential prime implicant” and “non essential prime implicant”.

Q.14. (AMIE S10, 10 marks): Minimise the function

f = Σ m (0, 2, 3, 6, 7, 8, 9, 10, 13)

using Quine McClusky method.

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

Q.15. (AMIE W10, 10 marks): Minimise the following using Tabular method:

$$Y = \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + AB\bar{C}\bar{D} + AB\bar{C}D + A\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D}$$

Answer: $F = B\bar{C} + A\bar{C}D + \bar{A}\bar{B}CD$

Q.16. (AMIE S11, 10 marks): Obtain the set of prime implicants for

Σm(0,1,6,7,8,9,13,14,15)

Answers: $A\bar{C}D, ABD, \bar{B}\bar{C}, BC$

Q.17. (AMIE W11, 10 marks): Reduce the following function using Quine McClusky method. Implement the reduced function using NAND gates only:

f = Σm (1, 3, 4, 7, 9, 11, 12, 14) + Σd (2, 6, 13)

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

Q.18. (AMIE W12, 12 marks): Simplify the following Boolean functions by Quine McClusky method:

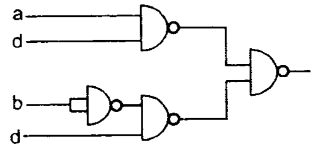
(i) F = Σ(1, 3, 5, 8, 10, 14)

(ii) F = Σ(1, 9, 10, 16,20) + Σ_d(14, 29, 30)

Answer: (i) $\bar{A}\bar{B}D + \bar{A}\bar{C}D + A\bar{B}D + AC\bar{D}$ (ii) $\bar{A}\bar{C}\bar{D}E + A\bar{B}\bar{D}\bar{E} + \bar{A}BD\bar{E}$

Q.19. (AMIE W09, 10 marks): Using Quine-McClusky method, reduce Y = (a, b, c, d) = Σm(1,3, 13, 15) + Σd(8, 9, 10,11) and implement using NAND gate only.

Answer: $f = \bar{b}d + ad$



Q.20. (AMIE W13, 8 marks): Using tabular method, obtain prime implicate and minimal expression for

$F = \pi M(2, 3, 8, 12, 13).d(10,14)$

Hint: $F = \pi M(2, 3, 8, 12, 13).d(10,14) = \Sigma m(0, 1, 4, 5, 6, 7, 9, 11, 15) + d(10, 14)$

Answer: $\bar{A}\bar{C} + \bar{A}B + AC + \bar{B}\bar{C}D$

Q.21. (AMIE S15, 10 marks): Using the tabular method, find the minimal expression for

$f = \Sigma[m(6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15)$

ARITHMETIC CIRCUITS

Q.22. (AMIE S15, 10 marks): What is half adder? Design a half adder using suitable logic gates.

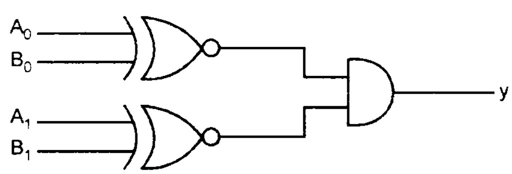
Q.23. (AMIE S13, 10 marks): Design a 3-bit adder using suitable logic gates.

Q.24. (AMIE W13, 10 marks): Design and implement a 2 bit by 2 bit binary multiplier using half adders.

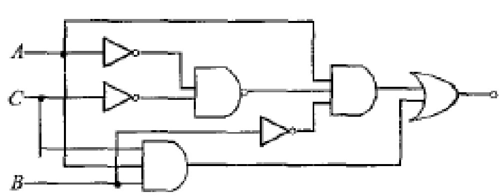
Q.25. (AMIE S15, 10 marks): Derive minimised Boolean expressions required for operation of full adder.

Q.26. (AMIE W13, 10 marks): Consider two binary numbers represented as A_1A_0 and B_1B_0. design a logic circuit whose output will be 1 when A_1A_0 is equal to B_1B_0.

Answer:



Q.27. (AMIE W14, 10 marks): Simplify the logic circuit shown below, where a mix of 1, 2 and 3 input logic gates are used.



(For online support such as eBooks, video lectures, audio lectures, unsolved papers, quiz, test series and course updates, visit www.amiestudycircle.com)