EC 3352 – DIGITAL SYSTEM DESIGN

UNIT – I : BASIC CONCEPTS

1.4 Quine-McCluskey Method of Minimization

Karnaugh Map provides a good method of minimizing a logic function. However, it depends on our ability to observe appropriate patterns and identify the necessary implicants. If the number of variables increases beyond five, K-map or its variant Variable Entered Map can become very messy and there is every possibility of committing a mistake. What we require is a method that is more suitable for implementation on a computer, even if it is inconvenient for paper- and-pencil procedures. The concept of tabular minimisation was originally formulated by Quine in 1952. This method was later improved upon by McClusky in 1956, hence the name Quine-McClusky

This Learning Unit is concerned with the Quine-McClusky method of minimisation. This method is tedious, time-consuming and subject to error when performed by hand. But it is better suited to implementation on a digital computer.

Principle of Quine-McClusky Method

The Quine-McClusky method is a two stage simplification process. Generate prime implicants of the given Boolean function by a special tabulation process. Determine the minimal set of implicants is determined from the set of implicants generated in the first stage. The tabulation process starts with a listing of the specified minterms for the 1s (or 0s) of a function and don't-cares (the unspecified minterms) in a particular format. All the prime implicants are generated from them using the simple logical adjacency theorem, namely, AB’ + AB = A. The main feature of this stage is that we work with the equivalent binary number of the product terms. For example in a four variable case, the minterms A’ BCD’ and A’ BC’ D’ are entered as 0110 and 0100. As the two logically adjacent minterms A’ BCD’ and A’ BC’ D’ can be combined to form a product term A’ BD’

,the two binary terms 0110 and 0100 are combined to form a term represented as "01-0", where ‘-‘ (dash) indicates the position where the combination took place. Stage two involves creating a prime implicant table. This table provides a means of identifying, by a special procedure, the smallest number of prime implicants that represents the original Boolean function. The selected prime implicants are combined to form the simplified expression in the SOP form. While we confine our discussion to the creation of minimal SOP expression of a Boolean function in the canonical form, it is easy to extend the procedure to functions that are givenin the standard or any other forms.

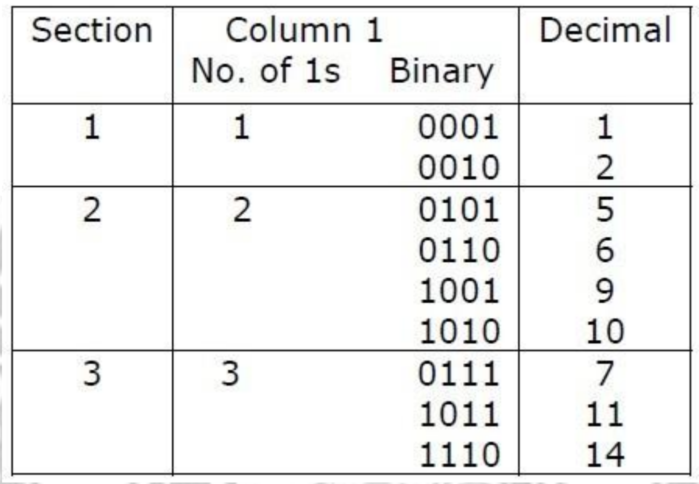
Generation of Prime Implicants

The process of generating prime implicants is best presented through an example.

Example 1: F = Σ (1,2,5,6,7,9,10,11,14)

All the minterms are tabulated as binary numbers in sectionalised format, so that each section consists of the equivalent binary numbers containing the same number of 1s, and the number of 1s in the equivalent binary numbers of each section is always more than that in its previous section. This process is illustrated in the table as below.

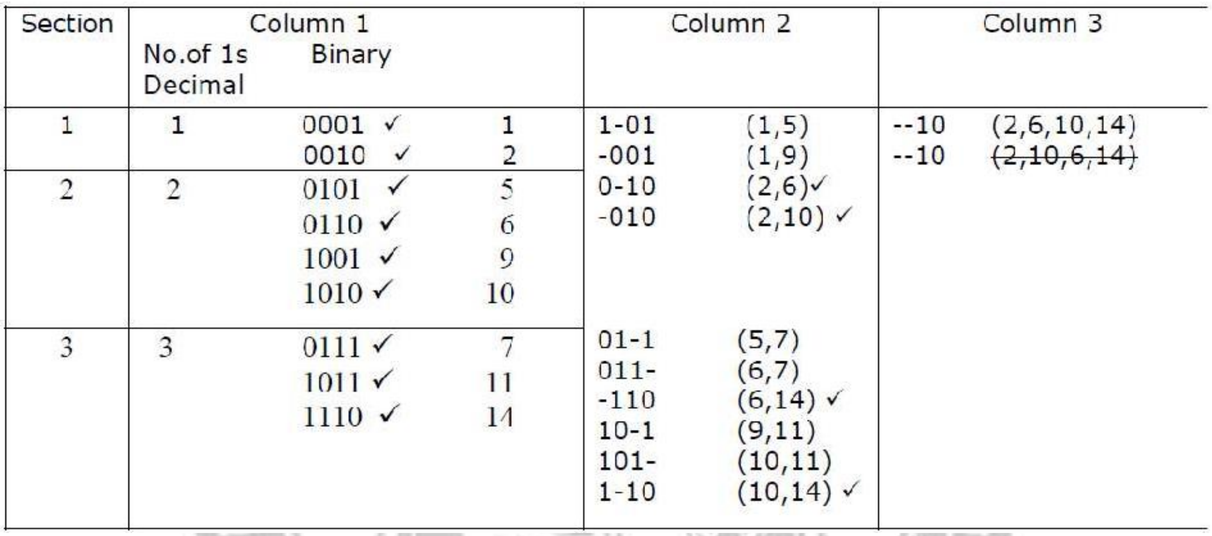
Table: 1.5 – No of 1's available in each binary



Section	Column 1		Decimal
	No. of 1s	Binary	
1	1	0001	1
		0010	2
2	2	0101	5
		0110	6
		1001	9
		1010	10
3	3	0111	7
		1011	11
		1110	14

The next step is to look for all possible combinations between the equivalent binary numbers in the adjacent sections by comparing every binary number in each section with every binary number in the next section. The combination of two terms in the adjacent sections is possible only if the two numbers differ from each other with respect to only one bit. For example 0001 (1) in section 1 can be combined with 0101 (5) in section 2 to result in 0-01 (1, 5). Notice that combinations cannot occur among the numbers belonging to the same section. The results of such combinations are entered into another column, sequentially along with their decimal equivalents indicating the binary equivalents from which the result of combination came, like (1, 5) as mentioned above. The second column also will get sectionalised based on the number of 1s. The entriesof one section in the second column can again be combined together with entriesin the next section, in a similar manner. These combinations are illustrated in theTable below

Table:1.6 – 1st and 2nd level reduction



Section	Column 1			Column 2	Column 3
	No.of 1s	Binary	Decimal		
1	1	0001 ✓	1	1-01 (1,5)	--10 (2,6,10,14)
		0010 ✓	2	-001 (1,9)	--10 ~~(2,10,6,14)~~
2	2	0101 ✓	5	0-10 (2,6)✓	
		0110 ✓	6	-010 (2,10) ✓	
		1001 ✓	9		
		1010 ✓	10		
3	3	0111 ✓	7	01-1 (5,7)	
		1011 ✓	11	011- (6,7)	
		1110 ✓	14	-110 (6,14) ✓	
				10-1 (9,11)	
				101- (10,11)	
				1-10 (10,14) ✓	

All the entries in the column which are paired with entries in the next section are checked off. Column 2 is again sectionalised with respect t the number of 1s. Column 3 is generated by pairing off entries in the first section of the column 2 with those items in the second section. In principle this pairing could continue until no further combinations can take place. All those entries that are paired can be checked off. It may be noted that combination of entries in column 2 can only take place if the corresponding entries have the dashes at the same place. This rule is applicable for generating all other columns as well. After the tabulation is completed, all those terms which are not checked off constitute the set of prime implicants of the given function. The repeated terms, like --10 in the column 3, should be eliminated. Therefore, from the above tabulation procedure, we obtain seven prime implicants (denoted by their decimal equivalents) as (1,5), (1,9), (5,7), (6,7), (9,11), (10,11), (2,6,10,14). The next stage is to determine theminimal set of prime implicants.

Determination of the Minimal Set of Prime Implicants

The prime implicants generated through the tabular method do not constitute the minimal set. The prime implicants are represented in so called "prime implicant table". Each column in the table represents a decimal equivalent of the minterm. A row is placed for each prime implicant with its corresponding product appearing to the left and the decimal group to the right side. Asterisks are entered at those intersections where the columns of binary equivalents intersect the row that covers them. The prime implicant table for the function under consideration is shown in the figure.

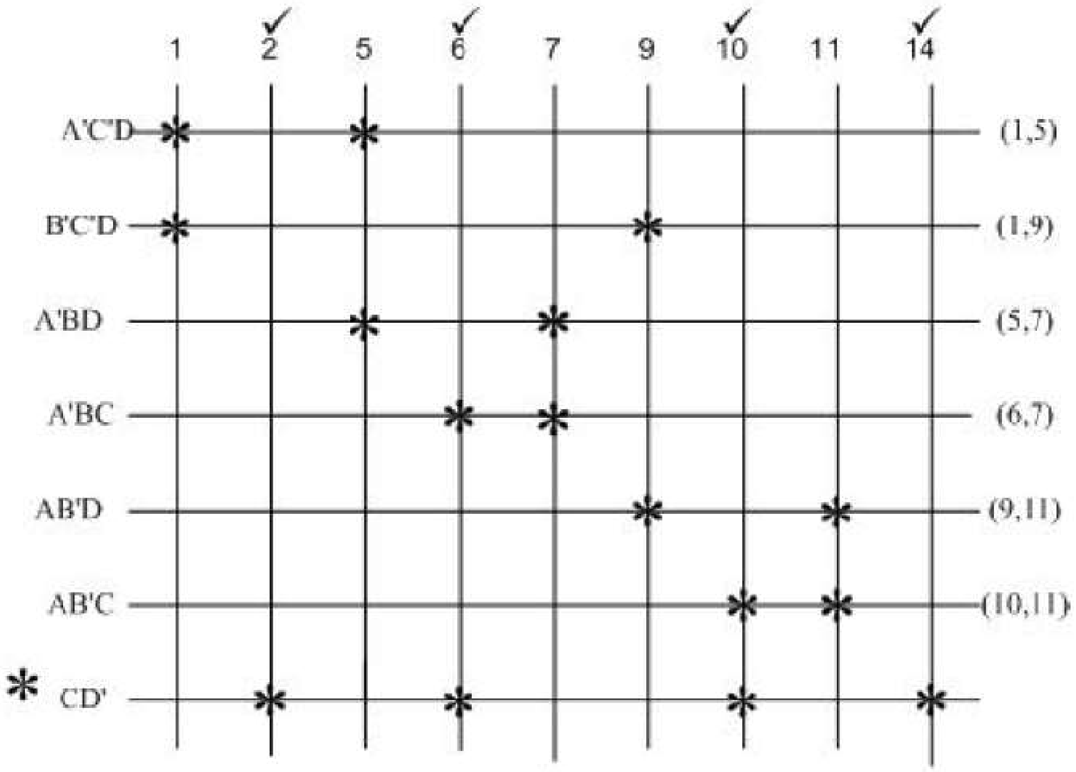


Fig 1. 1 – Prime implicants mapping

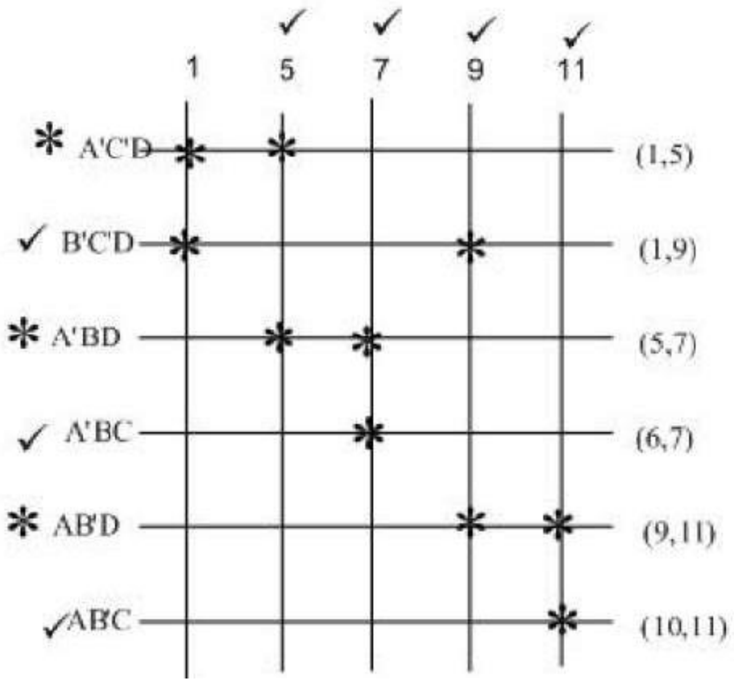


Fig 1. 2 – Essential Prime implicants mapping

We then select dominating prime implicants, which are the rows that have more asterisks than others. For example, the row A’ BD includes the minterm 7, which is the only one included in the row represented by A’BC. A’BD is dominant implicant over A’BC, and hence A’ BC can be eliminated. Mark A’BD by an asterisk and check off the column 5 and 7. We then choose AB’ D as the dominating row over the row represented by AB’ C. Consequently, we mark the row AB’ D by an asterisk, and eliminate the row AB’ C and the columns 9 and 11 by checking them off. Similarly, we select A’ C’ D as the dominating one over B’ C’ D. However, B’ C’ D can also be chosen as the dominating prime implicant and eliminate the implicant A’ C’ D. Retaining A’ C’ D as the dominant prime implicant the minimal set of prime implicants is {CD’ , A’ C’ D, A’ BD and AB’ D). The corresponding minimal SOP expression for the Boolean function is:

F = CD’ + A’ C’ D + A’BD + AB’ D

If we choose B’ C’ D instead of A’ C’ D, then the minimal SOP expression for theBoolean function is:

F = CD’ + B’C’ D + A’BD + AB’ D