https://doi.org/10.7494/automat.2018.22.2.29

V.C. Prasad*

Novel method to simplify Boolean functions

Abstract: Most methods for determining the prime implicants of a Boolean function depend on the minterms of the function. Deviating from this philosophy, this paper presents a method that depends on maxterms (the minterms of the complement of the function) for this purpose. Normally, maxterms are used to get prime implicates and not prime implicants. It is shown that all prime implicants of a Boolean function can be obtained by expanding and simplifying any product of sums form of the function appropriately. No special form of the product of the sums is required. What is more, prime implicants can generally be generated from any form of the function by converting it into a POS using well-known techniques. The prime implicants of a product of Boolean functions can be obtained from the prime implicants of individual Boolean functions. This allows us to handle big functions by breaking them into the products of smaller functions. A simple method is presented to obtain one minimal set of prime implicants from all prime implicants without using minterms. Similar statements also hold for prime implicates. In particular, all prime implicates can be obtained from any sum of a product's form. Twelve variable examples are solved to illustrate the methods.

Keywords: *Boolean functions, digital electronics, prime implicants*

1. Introduction

The minimization of a Boolean function is an important problem in Boolean algebra/combinational digital circuits [1–13]. This often involves determining all of the prime implicants/implicates of the function in the first instance. Then, a minimal set of prime implicants/implicates is selected to realize the function using gates. Several authors have studied this problem over the past seventy years or more. Many books on digital circuits discuss this topic extensively, and it is widely taught to undergraduate students of several disciplines of engineering and the physical sciences [1–4]. This paper addresses these problems in a new way. It gives a new method for generating all prime implicants. One minimal set of prime implicants is then determined from them without using minterms.

The most popular methods use minterms. Methods like the Karnaugh map and the Quine–McCluskey method start with the minterms of the function [1–4]. They are systematic; however,

*Department of Electrical Engineering, Faculty of Engineering, Dayalbagh Educational Institute, Dayalbagh, Agra 282005, India (Retired from Indian Institute of Technology, Delhi), e-mail: prasadvishdelhi@gmail.com

there are usually too many minterms. Algebraic techniques exist in Boolean algebra to simplify Boolean functions [1–4]; however, they are only used to partially simplify a Boolean function. They are not used to get all prime implicants. This is because (i) there is no systematic way in the literature to use them to achieve the objective and (ii) it is tedious to determine whether the product terms that are so generated are prime implicants or not. Even when it gives prime implicants, we do not know whether all of the prime implicants are generated or not. It is the purpose of this paper to use simple algebraic techniques in a systematic way to get all prime implicants/implicates. Starting with any product of sum's (POS) form, a Boolean function can be simplified to get all prime implicants using simple Boolean rules like distributive law, idempotent law, etc. A similar procedure on the sum of products (SOP) gives all prime implicates. Interestingly, this is achieved without using minterms for prime implicants and maxterms for prime implicates. This is contrary to what is normally done. Any POS (SOP) will do for prime implicants (implicates). Since any form of the function can be converted into a POS or SOP using well-known techniques, starting with a POS/SOP is not a restriction at all. The new approach has two important advantages: (i) it can start from any form – it does not require minterms or maxterms (which are usually too many), and (ii) it allows us to break up a big function into smaller functions and manipulate them.

Once all of the prime implicants are generated, the next task is to obtain one minimal set of them so as to cover all minterms of the function. Several methods for doing this exist in the literature [1–4]; however, most of them select prime implicants so that each minterm is covered. But, the new method does not start with minterms. Consequently, we need to generate a minimal set without using minterms. A new method is presented in this paper to achieve this.

The new methods for determining all prime implicants/implicates are described in the next section. They do not use all rules of Boolean algebra. The theorems are proven to show the correctness of the methods. The examples and remarks highlight the various features and intricacies of the new approach. Twelve variable examples are solved to illustrate the potential of the methods. Section III discusses a method to get one minimal set from all prime implicants. Section IV summarizes the contributions of this paper.

Implicant, implicate, prime implicant, prime implicate, minterm and maxterm are well--known terms in Boolean algebra [1–4]. We will use the following definitions for these terms, as they are more suitable for the present context.

Definition 1 (Prime implicant). Consider a Boolean function f that is zero for at least one value of its variables. A product (AND) of the literals of f is an implicant if $f = 1$ whenever all literals of the product are 1. It is a prime implicant if no proper subset of it is an implicant. Note that this definition does not require f to be a sum of the products (SOP) form. Let $f = (A'+B)(C+D)$; then, $A'BC$ is an implicant because $f = 1$ if $A' = 1, B = 1$, and $C = 1$. However, it is not a prime implicant because its subset BC is also an implicant. BC is a prime implicant because (i) it is an implicant and (ii) its subsets B and C are not implicants. An implicant is called a minterm if every variable is present in it in complemented or uncomplemented form. Note that $f = 0$ has no prime implicants. For example, $f = x'y'(x+y)$ has no prime implicants.

Definition 2 (Prime implicate). Consider a Boolean function f that is one for at least one value of its variables. A sum (OR) of the literals of f is an implicate if $f = 0$ whenever all literals of the sum are zero. It is a prime implicate if no proper subset of it is an implicate. Note that f need not be a product of sum (POS) to use this definition. Let $f = A'C + A'D + BC + BD$; then, $A' + B + C$ is an implicate because $f = 0$ if $A' = 0, B = 0$, and $C = 0$. However, it is not a prime implicate because its subset $A' + B$ is also an implicate. $A' + B$ is a prime implicate because (i) it is an implicate and (ii) its subsets A' and B are not implicates. An implicate is called a maxterm if every variable is present in it in complemented or uncomplemented form. Note that $f = 1$ has no prime implicates. For example, $AB + AB' + A'$ has no prime implicates. It is clear from these definitions that a prime implicant (implicate) of a function f is a prime implicate (implicant) of its dual $\hat{f}$ Furthermore, the complement of a prime implicant (implicate) of f is a prime implicate (implicant) of $\bar{f}$.

2. All prime implicants and prime implicates

In this section, the methods are presented to determine all prime implicants/implicates of Boolean function f of variables $x_1, x_2, \ldots, x_n$. The generation of all prime implicants is discussed first.

2.1. Prime implicants

Let $\bar{f}$ be the SOP form of its minterms; then, $f = \bar{\bar{f}}$ is a POS form. Each sum (factor) of f is an OR expression of the variables of the function. Each variable is present in the complemented or uncomplemented form but not both; i.e., each factor is a maxterm. Let there be m factors in POS; number them from 1 to m.

Method I. (Multiply out the POS form and simplify using the following rules)

$$X(Y+Z) = X \cdot Y + X \cdot Z \tag{1}$$

$$X \cdot Y = Y \cdot X \tag{2}$$

$$X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z \tag{3}$$

$$X \cdot X = X \tag{4}$$

$$X + X = X \tag{5}$$

$$X \cdot X' = 0 \tag{6}$$

$$X \cdot 0 = 0 \tag{7}$$

$$X + 0 = X \tag{8}$$

$$X + XY = X \tag{9}$$

The resulting simplified f is in SOP form. Let F denote this form; then, F has the prime implicants of f. The following theorem is about these prime implicants.

Theorem 1. *Every product term of F is a prime implicant of f. Furthermore, all prime implicants of f are generated.*

Proof. Note that Rules (1) through (9) are also there in Boolean algebra; hence, f and F are identical in a Boolean sense. So, we can use any rule of Boolean algebra when we talk about f and F in a Boolean sense. We will use Rules (1) through (9) for other situations. Let $P = y_1 y_2 \ldots y_\alpha$ be any product term of F where y_j is either a complemented or uncomplemented form of a variable of f for $j = 1, 2, \ldots, \alpha$. We first note that there is at least one literal in P from each factor of f. It is clear from Rules (1) through (9) that a literal from a factor may not appear in P if it gets eliminated like Y in Rule (9), X or X' in Rule (6), or X in Rule (7). First, consider Rule (6). $X(X')$ of a factor j gets eliminated if a factor k has $X'(X)$ only. Then, $f = 0$ for all values of its variables if $X(X')$ is the only literal of factor j. Since this is not true, factor j must have a literal different from this. This will give a nonzero P, and the literal of a factor will not be eliminated because of Rule (6). Next, consider Rule (7); zero in this rule can occur because of Rule (6) only (e.g., $YXX' = Y \cdot 0 = 0$). Therefore, elimination due to Rule (7) is not possible. Lastly, consider Rule (9). Let $y_1 = 1, y_2 = 1, \ldots, y_\alpha = 1$ and evaluate using the required rules of Boolean algebra. This gives $P = 1$ and $F = 1 = f$ (i.e., each factor of f is 1). Let y_1 refer to an x_{a1} variable. Consider a factor of f. Since every factor has every variable, this factor has x_{a1} or x'_{a1} (i.e., the factor has $y1$ or $y'1$). If the factor has all complemented y variables, then the factor is zero when $y_1 = y_2 = \ldots = y_\alpha = 1$. Therefore, this will be possible only if every factor has one of the literals of P; i.e., literals of P cannot get eliminated because of Rule (9). Thus, P has at least one literal from each factor. Conversely, every literal of P must lie in at least one factor. This is because we are expanding factors using Rule 1. So, literals on the right hand side of Rule 1 are from factors.

We next note that every product term P of F is a minimal term; i.e., a subset of the literals of P cannot cover all factors (a factor is covered if there is at least one literal from it in P). If this is not true, let $y_2, \ldots, y_\alpha$ cover all factors; then, this or a subset of it will appear in F, and Rule (9) will eliminate the P that has not happened. So, P must be minimal.

Let $P = y_1 y_2 \ldots y_\alpha$ be a minimal product term of F. Take $y_1 = 1, y_2 = 1, \ldots, y_\alpha = 1$ and use the required rules of Boolean algebra to evaluate F. This gives $F = 1 = f$. Therefore, P is an implicant of f from the definition. If P is not a prime implicant of f, let a literal (say, y_1) be redundant (i.e., $y_2 = 1, \ldots, y_\alpha = 1$ is an implicant of f). Every factor of f is 1 for these values of the variables. $f = 1 = F$ implies that $y_2 y_3 \ldots y_\alpha$ or a subset of it is a product term of F, as F is an SOP. If this is true, Rule (9) will eliminate the P that has not happened. Therefore, every product term of F must be a prime implicant of f.

We next prove that F has all prime implicants of f. Let $P = y_1 y_2 \ldots y_\alpha$ be any prime implicant of f; then, $f = 1 = F$ for $y_1 = y_2 = \ldots = y_\alpha = 1$ using the relevant rules of Boolean algebra. $f = 1$ implies that every factor of f is 1; i.e., every factor of f will have at least one of these literals as explained in the first paragraph of the proof, and so $y_1 y_2 \ldots y_\alpha$ will appear as a product term in F unless Rule (9) eliminates it. However, Rule (9) can eliminate it only if a subset of these variables is a product term in F. This requires that $F = f = 1$ when the

variables of this subset are 1. If this is true, then this subset will be an implicant of f (whereas a subset of a prime implicant cannot be an implicant). Therefore, Rule (9) cannot eliminate it, and P appears in F. This proves the theorem.

■

Now, consider any POS form of f (i.e., the factors of f need not have all variables). For example, let X be a factor in the POS of f. Let y,z be the variables missing in this factor; then,

$$X = (X+y+z)(X+y+z')(X+y'+z)(X+y'+z'). \tag{10}$$

This is true in Boolean algebra. We will now show that this is also true according to Rules (1) through (9). Expand the first two factors and the third and fourth factors using Rules (1), (2), (3), and (4).

$$\begin{aligned} R.H.S =& [(X+y)+z(X+y)+z'(X+y)+zz'][(X+y')+z(X+y')+z'(X+y')+zz'] \\ =& (X+y)(X+y') \quad \text{using Rules } (6),(8), \text{and } (9) \\ =& X+Xy+Xy'+yy' = X \quad \text{using Rules } (2),(4),(6),(8), \text{and } (9). \end{aligned}$$

Thus, Rule (10) is allowed in the above theorem. Take any factor of f and replace it by maxterms using Rule (10). This transforms any POS of f into $\Pi_1 \cdot \Pi_2$, where Π_1 is the POS of the maxterms of f and Π_2 is the product of the repeating maxterms, if any. Rules (2) and (3) can be used to achieve this if necessary; however, repeating factors can be ignored according to Rule (4). Therefore, $f = \Pi_1$, and Theorem 1 can be applied. Using Method I on this POS, we get all prime implicants of f. This proves the following theorem.

Theorem 2. *Let f be the given Boolean function in a POS form. Let F be the SOP form of this function using Method I. Then, every product term of F is a prime implicant of f. Furthermore, every prime implicant of f is a product term of F.*

Remarks:

1) Popular methods like the Karnaugh map [1–4], Quine-McCluskey's method [1–4], and many others start with minterms. A function usually has a large number of minterms, whereas the number of factors in a POS is relatively much less, as every variable need not be present in the factors. This is an important advantage of the new method.
2) Well-known methods [1–13] use minterms of f, whereas the new method uses POS (which depends on the minterms of $\bar{f}$). Thus, it is interesting to note that the basic philosophy of the new method is widely different from some popular methods. Normally, the minterms of $\bar{f}$ are used to getting prime implicates and not prime implicants.
3) As a special case, let f be a POS of all prime implicates of f. Then, Th. 2 shows that all prime implicates can be multiplied and simplified to get all prime implicants.

4) In general, f can also have don't care conditions (also called optional combinations). Include them in f (i.e., do not take them as minterms of $\bar{f}$ to get a POS of f). Get all prime implicants using the above method. Ignore a prime implicant if it only covers the don't care conditions.
5) Note that every literal of a factor need not appear in some product term of F. For example, let $f = (x+y)(x'+y)$. Then, $f = xx'+x'y+xy+y = y = F$ using Rules (1) through (9). There are no product terms in F that contain x or x'.
6) Consider $f = (x+y)(x'+y')(x'+y)(x+y')$. This function has no prime implicants because $f = 0$. Expand and simplify it using Rules (1) through (9); this gives $f = (x'y+xy')(xy+x'y') = 0 = F$, showing that the function also has no prime implicants according to the method.
7) Let $f = (x+y)(y+z)$; expand and simplify using Rules (1) through (8) but not Rule (9). This gives $f = F = xy+y+xz+yz$. Both y and xz are prime implicants, whereas xy and yz are not prime implicants (i.e., F contains all prime implicants). However, every product term is not a prime implicant; thus, Rule (9) does not help in the generation of prime implicants. It helps eliminate terms that are not prime implicants.

Example 1. Various features of the method concerning factors are illustrated below.

i. Consider $f = (x+y)(x'+z)(y+z)$; simplify using Rules (1) through (9). This gives $f = (x'y+xz+yz)(y+z) = x'y+xyz+yz+x'yz+xz+yz = x'y+xz+yz$. Note that $x'yz+xyz = yz$ using $x+x' = 1$. However, this rule is not there in Rules (1) through (9). Still, $x'yz, xyz$ are eliminated using Rule (9). Next, consider $f = (x+y)(x'+z)$. Using Rules (1) through (9), we get $f = x'y+xz+yz$. This function is the same as the above function. This implies that $(y+z)$ is not required; thus, all factors may not contribute to the prime implicants (and yet, we do not need to eliminate the unwanted factors).
ii. Consider $f = (x+y)(x'+z)(x+y)$. Using Rules (1) through (9), we get $f = (x'y+xz+yz)(x+y) = xz+xyz+x'y+xyz+yz = xz+x'y+yz$. Factor $(x+y)$ is repeating. We could have eliminated it using Rule (4), but we did not do this. Even then, the method correctly gives the prime implicants.
iii. Let $f = (x+y)(x'+z) = (x+y+z)(x+y+z')(x'+y+z)(x'+y'+z)$; i.e., take the POS of the maxterms and simplify using Rules (1) through (9). This gives $f = [z(x+y)+z'(x+y)+(x+y)+zz'][y(x'+z)+y'(x'+z)+(x'+z)+yy'] = (x+y)(x'+z) = xz+x'y+yz$. Thus, we need not start with a POS of the maxterms to get all prime implicants. This is a huge advantage because there are usually too many maxterms.
iv. Let $f = (x+y)(x'+y+z)(x'+y'+z)$; i.e., some factors are maxterms and others are not. Using Rules (1) through (9), we get $f = F = xz+x'y+yz$, which gives all prime implicants. Thus, the POS can have a mixture of maxterms and other factors.
v. Let $f = (x+y)(x'+z)(x+y+z)$. Factors $(x+y)$ and $(x+y+z)$ have a common maxterm (i.e., some maxterms are repeating implicitly). Even then, the method correctly gives all prime implicants. Simplifying, we get $F = xz+x'y+yz$.

All of these examples show that POS can be in any form. No special form of POS is required to get all prime implicants. This is an important advantage because we need not simplify or do checks on the given POS form.

Example 2. Consider $f = (x+z)(x'+y)(u+v')$. Using Rules (1) through (9), expand and simplify. This gives $f = (x'z+xy+yz)(u+v') = x'uz+xyu+uyz+x'v'z+xyv'+v'yz$. Every term in this is a prime implicant. Now, let us use Boolean theorem $x'z+xy+yz = x'z+xy$; this gives $f = (x'z+xy)(u+v') = x'uz+xyu+x'v'z+xyv'$. This misses prime implicants uyz and $v'yz$. Next, consider

$$\begin{aligned} f &= (x+z)(x'+y)(u+y) = (x'z+xy+yz)(u+y) \\ &= x'yz+xy+yz+x'zu+xyu+uyz. \end{aligned}$$

If we don't take yz using the above Boolean theorem, then $x'yz$ will not be eliminated. This shows that we cannot blindly use all of the rules of Boolean algebra in the above method.

Example 3. Let $f = (x+x'y)(x+y)$. Using Rules (1) through (9), $f = x+xy+x'y = x+x'y$; however, $x'y$ is not a prime implicant. This happened because $(x+x'y)$ is not an allowed factor. A factor is required to only be the sum of literals. If we take $x+y$ in place of $(x+x'y)$, the method correctly gives prime implicants.

Example 4. Consider the following Boolean function of twelve variables: $f(x_1, x_2, \ldots, x_{12}) = (x_1+x_2+x_3+x_4+x_5)(x'_6+x_7+x'_8+x'_9+x'_{10})(x'_1+x_2)(x_1+x_{11})(x_1+x'_{12})$.

We want to determine all of the prime implicants of this function using Boolean Rules (1) through (9). Multiplying factors 1 & 3 and simplifying, we get $x_2+x'_1x_3+x'_1x_4+x'_1x_5$. The last two factors give $x_1+x_{11}x'_{12}$. Multiply these two partial SOPs with factor 2 and simplify; this gives the following:

$$\begin{aligned} f = F = {} & x_1x_2x'_6+x_1x_2x_7+x_1x_2x'_8+x_1x_2x'_9+x_1x_2x'_{10}+x_2x'_6x_{11}x'_{12}+x'_1x_3x'_6x_{11}x'_{12} \\ & +x'_1x_4x'_6x_{11}x'_{12}+x'_1x_5x'_6x_{11}x'_{12}+x_2x_7x_{11}x'_{12}+x'_1x_3x_7x_{11}x'_{12} \\ & +x'_1x_4x_7x_{11}x'_{12}+x'_1x_5x_7x_{11}x'_{12}+x_2x'_8x_{11}x'_{12}+x'_1x_3x'_8x_{11}x'_{12} \\ & +x'_1x_4x'_8x_{11}x'_{12}+x'_1x_5x'_8x_{11}x'_{12}+x_2x'_9x_{11}x'_{12}+x'_1x_3x'_9x_{11}x'_{12} \\ & +x'_1x_4x'_9x_{11}x'_{12}+x'_1x_5x'_9x_{11}x'_{12}+x_2x'_{10}x_{11}x'_{12}+x'_1x_3x'_{10}x_{11}x'_{12} \\ & +x'_1x_4x'_{10}x_{11}x'_{12}+x'_1x_5x'_{10}x_{11}x'_{12}. \end{aligned}$$

Each product term is a prime implicant, and all of the prime implicants of f are obtained. Any method that uses minterms (e.g., Quine-McCluskey's method) requires enormous effort to get these prime implicants.

Theorem 3. *Let $f = f_1f_2\ldots f_a$, where $f_1, f_2, \ldots, f_a$ are SOPs of all prime implicants of the respective functions. Expand the product and simplify using Rules (1) through (9). Every product term of the resulting SOP is a prime implicant of f, and every prime implicant of f is a product term of the SOP.*

Proof. Express $f_1, f_2, \ldots, f_a$ in the POS form of their maxterms and apply Th.1 to each of them. Then, f_j will be the SOP of all of its prime implicants. This is true for all $j = 1, 2, \ldots, a$. Applying Th.1 to f as a product of all of these maxterms, we get all of the prime implicants of f. However, the POS of the SOPs of all prime implicants of f_1 to f_a is an intermediate step in the expansion and simplification using Rules (1) through (9); hence, the result. ∎

Example 5. Let $f = f_1 f_2$, where $f_1 = x + x'y$ and $f_2 = xy + yz$. $f = (x + x'y)(xy + yz) = xy + xyz + x'yz = xy + x'yz$ using Rules (1) through (9). However, $x'yz$ is not a prime implicant and Theorem 3 is not violated. This is because $x'y$ is not a prime implicant of f_1, whereas the theorem requires that f_1 and f_2 should be the SOPs of their prime implicants.

Remark:

> It follows from Th. 3 that a large function can be handled by breaking it into smaller functions. This is an important advantage. A similar thing is not possible in methods that are based on the minterms of f (like Quine-McCluskey's method). If the minterms of f are grouped into sub functions, we get $f = f_1 + f_2 + \ldots$ In this case, a prime implicant of f_1 may combine with some minterms of f_2 to give a smaller implicant. Therefore, a prime implicant of f_1 may not be a prime implicant of f. Consequently, this type of division into sub functions does not work.

The method presented above requires f in the POS form; however, any form of f can easily be converted into a POS form using the well-known techniques of Boolean algebra [1–4]. Therefore, it is possible to generate all prime implicants from any form of f. The following example illustrates this:

Example 6. Let $f = x_1x_2x_3' + x_3x_4 + (x_1 + x_2x_3)(x_4 + x_5x_6)$. This is neither SOP nor POS. The factors in the product are themselves the sum of the products. We can get all of the prime implicants of this function by first converting it into a POS form, where each factor is sum of literals. This is done as follows: Let $F = YZ + F_1$, where F is the given function, YZ is a product term of F, and F_1 is the remaining part of F. Then, $YZ + F_1 = (Y + F_1)(Z + F_1)$. This converts a sum into a product. By using this repeatedly, we can convert any sum into a product. $x_1 + x_2x_3 = (x_1 + x_2)(x_1 + x_3), x_4 + x_5x_6 = (x_4 + x_5)(x_4 + x_6)$. Let F_1 be the product of these factors. Then, $f = x_1x_2x_3' + F_2$, where $F_2 = x_3x_4 + F_1$. $f = (x_1 + F_2)(x_2 + F_2)(x_3' + F_2)$.

$$
\begin{aligned}
x_3' + F_2 &= x_3' + x_4 + F_1 \\
&= (x_3' + x_4 + x_1 + x_2)(x_3' + x_4 + x_1 + x_3)(x_3' + x_4 + x_4 + x_5)(x'3 + x_4 + x_4 + x_6) \\
&= (x_1 + x_2 + x_3' + x_4)(x_3' + x_4 + x_5)(x_3' + x_4 + x_6)
\end{aligned}
$$

$$
\begin{aligned}
x_1 + F_2 &= x_1 + x_3x_4 + F_1 = (x_1 + x_3x_4 + x_1 + x_2)(x_1 + x_3x_4 + x_1 + x_3)(x_1 + x_3x_4 + x_4 + x_5) \\
&\quad (x_1 + x_3x_4 + x_4 + x_6)
\end{aligned}
$$

However, $x_1 + x_2 + x_3x_4 = (x_1 + x_2 + x_3)(x_1 + x_2 + x_4)$. This gives $x_1 + F_2 = (x_1 + x_2 + x_4)(x_1 + x_3)(x_1 + x_4 + x_5)(x_1 + x_4 + x_6)$.

Similarly, $x_2 + F_2 = (x_1 + x_2 + x_3)(x_1 + x_2 + x_4)(x_2 + x_4 + x_5)(x_2 + x_4 + x_6)$.

Therefore, $f = (x_1 + x_2 + x_4)(x_1 + x_3)(x_1 + x_4 + x_5)(x_1 + x_4 + x_6)(x_2 + x_4 + x_5)(x_2 + x_4 + x_6)(x_1 + x_2 + x_3' + x_4)(x_3' + x_4 + x_5)(x_3' + x_4 + x_6)$, which is the required POS form. Now, expand and simplify using Rules (1) through (9); this gives all prime implicants of f. They are $x_1x_4, x_3x_4, x_1x_2x_3', x_1x_5x_6$, and $x_2x_3x_5x_6$.

2.2. Prime implicates

Take f in any SOP form. Let $\hat{f}$ be its dual. It is in POS form. Expand $\hat{f}$ and simplify using Rules (1) through (9). Let $\hat{F}$ denote the resulting SOP form. Let F be its dual. Since $\hat{f} = \hat{F}$, $f = F$ in a Boolean sense. Theorem 2 can be applied to $\hat{f}$ and $\hat{F}$. This proves the following theorem.

Theorem 4. *Take any SOP form of f , form its dual, expand, and simplify it using Rules (1) through (9). This gives F. Then, each factor of F is a prime implicate of f. Furthermore, each prime implicate of f is a factor of F.*

Theorem 5. *Let $f = f_1 + f_2 + \ldots + f_a$, where $f_1, f_2, \ldots, f$ a are the POSs of all prime implicates of the respective functions. Expand the product and simplify using Rules (1) through (9) on the product of the duals of $f_1, f_2, \ldots, f_a$. The dual of every product term of the resulting SOP is a prime implicate of f, and every prime implicate of f is a dual of a product term of the SOP.*

Proof. The proof follows from Th. 3 on the duals of functions $f_1, f_2, \ldots, f_a$. Comments similar to those in the remarks and examples above also hold for the prime implicates. ∎

Example 7. Consider function $f(x_1, x_2, \ldots, x_{12}) = x_1x_2x_5'x_6x_8' + x_3'x_4'x_7x_9x_{11}'x_{12} + x_1x_2' + x_2x_{10}' + x_2x_{11}$. We wish to determine all prime implicates of this function using Rules (1) through (9) on its dual $\hat{f}$.

$$\hat{f} = (x_1 + x_2 + x_5' + x_6 + x_8')(x_3' + x_4' + x_7 + x_9 + x_{11}' + x_{12})(x_1 + x_2')(x_2 + x_{10}')(x_2 + x_{11}).$$

Multiply factors 1 and 3 and 4 and 5 and simplify them using Rules (1) through (9). This gives $(x_1 + x_2'x_5' + x_2'x_6 + x_2'x_8')(x_2 + x_{10}'x_{11})(x_3' + x_4' + x_7 + x_9 + x_{11}' + x_{12})$. Expand and simplify to get

$$\begin{aligned}\hat{f} = \hat{F} = {} & x_1x_2x_{11}' + x_1x_2x_3' + x_1x_3'x_{10}'x_{11} + x_2'x_3'x_5'x_{10}'x_{11} + x_2'x_3'x_6x_{10}'x_{11} \\ & + x_2'x_3'x_8'x_{10}'x_{11} + x_1x_2x_4' + x_1x_4'x_{10}'x_{11} + x_2'x_4'x_5'x_{10}'x_{11} \\ & + x_2'x_4'x_6x'10x_{11} + x_2'x_4'x_8'x'10x_{11} + x_1x_2x_7 + x_1x_7x'10x_{11} + x_2'x_5'x_7x'10x_{11} \\ & + x_2'x_6x_7x'10x_{11} + x_2'x_7x_8'x'10x_{11} + x_1x_2x9 + x_1x_9x'10x_{11} + x_2'x_5'x_9x'10x_{11} \\ & + x_2'x_6x_9x'10x_{11} + x_2'x_8'x_9x'10x_{11} + x_1x_2x_{12} + x_1x_{10}'x_{11}x_{12} + x_2'x_5'x_{10}'x_{11}x_{12} \\ & + x_2'x_6x_{10}'x_{11}x_{12} + x_2'x_8'x_{10}'x_{11}x_{12}\end{aligned}$$

$$
\begin{aligned}
f = F = {} & (x_1 + x_2 + x'11)(x_1 + x_2 + x_3')(x_1 + x_3' + x_{10}' + x_{11})(x_2' + x_3' + x_5' + x_{10}' + x_{11}) \\
& (x_2' + x_3' + x_6 + x_{10}' + x_{11})(x_2' + x_3' + x_8' + x_{10}' + x_{11})(x_1 + x_2 + x_4') \\
& (x_1 + x_4' + x_{10}' + x_{11})(x_2' + x_4' + x_5' + x_{10}' + x_{11})(x_2' + x_4' + x_6 + x_{10}' + x_{11}) \\
& (x_2' + x_4' + x_8' + x_{10}' + x_{11})(x_1 + x_2 + x_7)(x_1 + x_7 + x_{10}' + x_{11}) \\
& (x_2' + x_5' + x_7 + x_{10}' + x_{11})(x_2' + x_6 + x_7 + x_{10}' + x_{11})(x_2' + x_7 + x_8' + x_{10}' + x_{11}) \\
& (x_1 + x_2 + x_9)(x_1 + x_9 + x_{10}' + x_{11})(x_2' + x_5' + x_9 + x_{10}' + x_{11}) \\
& (x_2' + x_6 + x_9 + x_{10}' + x_{11})(x_2' + x_8' + x_9 + x_{10}' + x_{11})(x_1 + x_2 + x_{12}) \\
& (x_1 + x_{10}' + x_{11} + x_{12})(x_2' + x_5' + x_{10}' + x_{11} + x_{12})(x_2' + x_6 + x_{10}' + x_{11} + x_{12}) \\
& (x_2' + x_8' + x_{10}' + x_{11} + x_{12}).
\end{aligned}
$$

Each sum term (factor) is a prime implicate, and all of the prime implicates of f are obtained. Any method that uses the maxterms of f (i.e., minterms of $\bar{f}$ [e.g., Quine-McCluskey's method]) requires enormous effort to get these prime implicates.

3. Minimal sets of prime implicants

Once all of the prime implicants are obtained, we wish to determine one minimal set of prime implicants that covers the function. This topic is well-studied [1–4, 11–13]; e.g., one can use Prasad's method [11] to get one minimal set. However, such methods cover the function by covering its minterms. Since the method of Section II does not start with minterms, it is convenient to use a method that does not depend on minterms. Furthermore, a function usually has many minterms. In this section, we will describe a method to get one minimal set of prime implicants from all prime implicants.

Let f be the given function and $F = P_1 + P_2 + \ldots + P_l$ be the sum of all prime implicants of f. The following method gives a minimal set of prime implicants covering all of the minterms of f. This is achieved by deleting the redundant prime implicants from F.

Method II. (Minimal set of prime implicants)

1) $j = 1$.
2) Consider P_j. Choose the literals in P_j as 1 so that P_j is 1. Take out P_j from F and evaluate $\bar{F}$ for these values of the variables. P_j is redundant if $\bar{F} = 0$. Otherwise, put it back in F.
3) $j = j + 1$. If $j = l + 1$, the algorithm terminates. Otherwise, go to Step 2.

Theorem 6. *F at the end of Method II gives a minimal set of prime implicants covering all of the minterms of f.*

Proof. Initially, $f = F$. So P_1 to P_l cover all of the minterms of f. In Step 2, a redundant prime implicant is deleted. Therefore, the remaining F covers all of the minterms of f. This implies that F at the end of the algorithm covers all of the minterms of f. Let S_m be the set of all prime implicants in F at the end. Let P_j be any prime implicant of S_m. Let $F_{\cdot j}$ be the F without

P_j when P_j was examined in Step 2 for deletion. It was not deleted because it has at least one minterm that is not covered by the prime implicants of F_{-j}. The prime implicants in S_m without P_j are a subset of those in F_{-j}. Therefore, this subset cannot cover P_j if F_{-j} cannot do it. Thus, P_j has at least one minterm not covered by the remaining prime implicants in S_m. This proves the theorem. ∎

The following example illustrates Method II.

Example 8. Let $f = (x_1 + x_3 + x_4)(x_1 + x_2' + x_3' + x_4)(x_1' + x_2 + x_3')(x_2 + x_3' + x_4)$. Using Method I, $F = x_1x_2 + x_2x_4 + x_1x_3' + x_3'x_4 + x_1'x_4$. To determine a minimal set of prime implicants, start with x_1x_2 and check if it is redundant. Take $x_1 = 1, x_2 = 1$ and evaluate $\overline{x_2x_4 + x_1x_3' + x_3'x_4 + x_1'x_4}$. This is nonzero, showing that x_1x_2 is not redundant. So, retain it in F. Consider x_2x_4. Take $x_2 = 1$ and $x_4 = 1$ and evaluate $\overline{x_1x_2 + x_1x_3' + x_3'x_4 + x_1'x_4}$. This is zero. So, x_2x_4 is redundant. Deleting it, $F = x_1x_2 + x_1x_3' + x_3'x_4 + x_1'x_4$. Next, consider x_1x_3'. It is not redundant. Proceeding like this, $x_3'x_4$ is redundant and $x_1'x_4$ is not redundant. This gives $F = x_1x_2 + x_1x_3' + x_1'x_4$; i.e., $S_m = x_1x_2, x_1x_3', x_1'x_4$.

4. Conclusion

It is shown that all prime implicants (implicates) can be determined from any POS (SOP) form of a Boolean function. This is like using maxterms (minterms) for prime implicants (implicates) contrary to the normal practice. Thus it is widely different from many well-known methods.

References

[1] Marcus M.P., *Switching Circuits for Engineers*, Prentice-Hall, Englewood Cliffs, New Jersey 1969.

[2] Givone D.D., *Digital Principles and Design*, International Edition, McGraw-Hill, New York 2003.

[3] Kohavi Z., Jha N.K., *Switching and Finite Automata Theory*, Cambridge University Press, New York 2010, www.cambridge.org/9780521857482.

[4] Marcovitz A.B., *Introduction to logic design*, International Edition, McGraw-Hill, New York 2002.

[5] Biswas N.N., *Minimization of Boolean functions*, IEEE Trans. On Computers, Vol. C-20/8, Aug. 1971, pp. 925–929.

[6] Suresh Ch., *Minimization of switching functions – A fast technique*, IEEE Trans. On Computers, Vol. C-24, 1975, pp. 735–756.

[7] Rhyne N., McKinney P., *A new technique for the fast minimization of switching functions*, IEEE Trans. On Computers, Vol. C – 26, 1977, pp. 757–764.

[8] Gurunath B., Biswas N.N., *An algorithm for multiple output minimization*, IEEE Trans. On CAD of Integrated circuits, Vol. 8, 1989, pp. 1007–1013.

[9] Minato S., *Fast generation of prime irredundant covers from Binary decision Diagrams*, IEICE Trans. Fundamentals, Vol. E 76 – A, No. 6, June 1993, pp. 967–973.

[10] Tomaszewski S.P., Celix I.U., Antoniou G.E., *WWW – Based Boolean function minimization*, Intl. J. Appl. Math. Comput. Sci., Vol. 13/1, 2003, pp. 577–583.

[11] Prasad V.C., *Efficient minimization of Boolean functions*, International Journal Of Electrical Engineering Education, 45(4), Oct. 2008, pp. 321–327.
[12] Prasad V.C., *Quality of minimal sets of prime implicants of Boolean functions*, Intl. Journal of Electronics and Telecommunications (Poland), Vol. 63/2, 2017, pp. 165–169.
[13] Prasad V.C., *Generalized Karnaugh Map method for Boolean functions of many variables*, IETE Journal of Education (India), Vol. 58/1, 2017, pp. 11–19.

Nowa metoda upraszczania funkcji boolowskich

Streszczenie: Większość metod wyznaczania implikantów prostych funkcji boolowskiej wykorzystuje mintermy funkcji. Niniejszy artykuł prezentuje odmienną metodę, która stosuje do tego celu makstermy (mintermy dopełnienia funkcji). Jest to podejście niestandardowe, gdyż zazwyczaj na podstawie makstermów uzyskuje się implicenty proste, a nie implikanty proste. W artykule pokazano, że wszystkie implikanty proste funkcji boolowskiej mogą być otrzymane przez odpowiednie rozwinięcie i uproszczenie funkcji podanej w dowolnej postaci typu iloczyn sum. Nie jest przy tym wymagana żadna szczególna postać tego iloczynu. Co więcej, implikanty proste mogą być zwykle utworzone na podstawie funkcji w dowolnej postaci, przez przekształcenie jej do postaci typu iloczyn sum z wykorzystaniem znanych metod. Implikanty proste iloczynu funkcji boolowskich można uzyskać z implikantów prostych poszczególnych funkcji. To pozwala operować na dużych funkcjach przez rozbicie ich na iloczyn mniejszych. Artykuł pokazuje prostą metodę uzyskania jednego zbioru minimalnego implikantów prostych na podstawie zbioru wszystkich implikantów prostych bez korzystania z mintermów. Podobne stwierdzenia mają zastosowanie także w przypadku implicentów prostych. W szczególności, wszystkie implicenty proste mogą być otrzymane z dowolnej formy typu suma iloczynów. Zaproponowana w artykule metoda została zilustrowana licznymi przykładami.

Słowa kluczowe: *funkcje boolowskie, elektronika cyfrowa, implikanty proste*