An Efficient Technique For Switching Functions Simplification

Authors: Sahadev Roy
Dept. of ECE, NIT Arunachal Pradesh, Yupia, 791112, India
e-mail:sdr.ece@nitap.in

Subject Category: Engineering
Sub Category: Digital Electronics

Editor: Md. Abdul Alim Sheikh

Volume 2 Issue 1 January 2017
Received on: 07.01.2017
Revised on: 28.01.2017

Keywords: Algorithm for Boolean function minimization;
Combinational logic;
Octal coded minterms;
Multiple inputs logic circuit minimization;
Switching circuit simplification;
Sum-of-Products (SOP);
Weighted sum.
An Efficient Technique For Switching Functions Simplification

Sahadev Roy

Abstract

In this paper, we propose a new approach for switching function simplification using octal coded minterms. This novel method of two-level minimization techniques effectively reduces the number of steps in comparison with other existing methods. A chart for octal based minimization for manual minimization process is also proposed herein. Proposed minimization method factorized minterms in a group of three input variables which are represented in octal code. These groups of minterms further paired with adjacent minterms. For easy detection, an octal chart also proposed here. The proposed method effectively reduces the complexity of multiple input minimization process. The results, which are verified by comparison with available experimental data, indicated that the minimal product of some was achieved with minimal effort.

1. Introduction

Investigation on combinational logic circuit minimization is quite old but not obsolete [1]. The requirement of efficient multiple input logic synthesis in different fields like automatic control system design, many areas of FPGA design flow [2-4], VLSI synthesis [5], the design of BIST, feature reduction [6] and biological logic circuits designing [7]. It is possible to design the same functional combinational circuit with various design styles. Design efficiency depends on target technology. Minimal sum-of-product realization of any switching function is the basic criteria for design consideration. Many optimal and elegant algorithms are available for this purpose. Combinational switching circuit designed technique is classified into categories, two-level, and multi-level implementations. The switching action of any two level circuits is fast in comparison with the multilevel design, but it’s required a large number of logic gates. Power consumptions are increased with the number of gates increased [8]. There are two main approaches to logic synthesis, exact algorithms, and heuristic algorithms. Heuristic algorithms are based on informal methods and solutions are near minimal whereas exact algorithms are complex in nature. The paper represents an efficient minimization processed using octal code representation which is shorter and easily converted into a binary number which reduces the complexity of minimization. This technique is also useful for multiple inputs multiple outputs digital circuit simplification. In Section 2, octal minterms are introduced here. Octal adjacency chart proposed in Section 3 along with simplification rules for easy minimization. Applications of octal minterms are describe in Section 4 using suitable example.

2. Octal Minterms and Logic Adjacency

All logic function is represented in two canonical forms: Sum of Product (SOP) and Product of Sum (POS) [9]. the SOP form can be obtained from POS or vice versa by using the "duality" principal. The POS is related with maxterm and SOP is related with minterms. Minterms are defined as any one of the $2^n$ distinct intersections (ORing) where maxterm is defined as any one of the $2^n$ distinct union (ANDing) of all n variables. Both minterm and maxterm these literals are used either non-complimented or complemented form but not both [10]. Literals are marked either high (1) for non-negation or low (0) for negation [11]. Input variables are arranged arbitrarily, and positional weight of power 2 are assigned to each variable which from a truth table. By using binary to decimal number conversion technique, minterms are converted into the decimal number from here designer obtains Decimal Coded of Minterm (DCM). DCM is not suitable for minimization because all simplifications approached are based on binary representation or Gray code. But binary numbers are not suitable for realization due to a large number of digits. Reconversions from decimal into binary number increases unnecessary complexity in minimization of multiple inputs logic synthesis [12]. Few minimization algorithms are reported using DCM like as 'Minterm Set Method' [13], 'RAD Tree' [14]. Karnaugh map [15] also manual minimization technique using DCM but proper minimization method becomes complicated for more than four input switching system. Let an eight variable system minterms are expressed in binary form as Eq. (1).

\[ f_1(A, B, C, D, E, F, G, H) = \sum (00011000, 00011001, 00011100, 00011101, 01101001, 11101001) \]  

By using binary to octal conversion method, the same function can be expressed as Eq. (2). Suffix 8 is used to indicate octal minterms.

\[ f_2(A, B, C, D, E, F, G, H) = \sum (030, 031, 034, 035, 131, 351)_8 \]  

Gray code has one of the interested property i.e. one-bit changes from its nearest bit which served as logic adjacency [7]. Another property is reflection property. Here, Bold purple colour indicate Octal indices.

3. Simplification Using Octal Minterms

Propose octal chart (Fig. 1) is useful to determine logic adjacency for manual technique. The link shows possible adjacent pairs. A possible pair of four adjacent minterms is those minterms complete ring mainly 0-1-3-2, 0-1-5-4, 4-5-7-6, 1-3-7-5, 3-2-7-6, 0-2-6-4 and an only possible pair of eight minterms forms 0 to 7. Realization of binary from of lowest and highest minterms of the pair and bit change difference by X which indicates complimented literals. This technique is also applicable for a pair of four minterms.

![Figure 1. Octal Chart for manual synthesis of combinational logic circuit.](image1)

This technique improves the speed of simplification. In Fig. 2, dotted links represented all simplified product term. Double ‘X’ indicates pairing of four minterms.

![Figure 2. Links represent logic adjacency with simplified product terms.](image2)
This technique effectively reduces complexity. Logic functions with large number of input variables are basically octal minterms breakup variables into many parts by grouping three variables starting from least significant bit. The pairing of three least significant variables we called here Least Significant Pair (LSP). Pairing of a most significant bit we called here as Most Significant Pair (MSP). MSP may be containing three or less number of variables. This pairing technique automatically executes when any minterm represents in octal code [17]. We propose here the rules of minimization using octal minterms.

a) Rule 1

Simplified product term is obtained directly by realizing binary from of lowest and highest minterms of an adjacent pair of two, four or eight by changing bit difference by X which indicates complimented literals.

Example: For pair 4 and 5 binary representation 100 and 101 and difference of bit at LSB so simplified product term is 10X.

b) Rule 2

All possible Adjacent LSP with common MSP, further simplification are possible and simplified product term will contain bit pattern of MSP with right padding of simplified LSP bit pattern which is obtained by rule 1. This rule is also applicable for common LSP with adjacent MSP.

Example: Let us consider two octal minterms 45 and 65. Here, LSP are common, and 4 and 6 are adjacent, and the simplified term is 1X0 so, \((45-65) \rightarrow (4-6)5 \rightarrow 1X05 \rightarrow 1X0101\).

Another example XX14 and XX15: Here bit pattern of MSP are the same and 4-5 are adjacent thus the simplified term is 10X so, \((XX14-XX15) \rightarrow XX1(4-5) \rightarrow XX110X\).

c) Rule 3

For any pair of adjacentLSP, the bit pattern of MSP would not match due to the difference of bit X with the presence of either 1 or 0 in other group pairing are allowed between them considering that X as 1 or 0 as per requirement. Changing of X into 1 or 0 allows only for a particular group contain a large number of X with rest. In this case, some the simplified terms must be two: one with the same bit pattern which contains that X and other is a non-minimized MSP bit pattern with the minimized LSP bit pattern. This rule is also applicable for the identical bit pattern of LSP.

Example: Let us consider two partial minimized term XX14 and X11 5. Here 4 and 5 logic adjacent LSP but MSP possesses different bit pattern. This difference is abolished if middle MSP bit X is considered as 1 because X can be any value 0 or 1. We divided its pattern of XX14 as X014 and X114 to match with X115. So pairing is done as X014-X114 and X114-X115. So, X014-X114 \(\rightarrow\) XX14 \(\rightarrow\) XX1100 same bit patterns. Other pairs are also minimized using rule 2 as X114-X115 \(\rightarrow\) X11(4-5) \(\rightarrow\) X1110X, so non-minimized bit pattern follows MSP portion and LSP with minimized bit pattern. This rule is also applicable for multiple bit changes only when in a group like XX14 and 1115, and its minimized terms are XX1100 and 11110X.

Now, we consider case like XX15, 1XX7 i.e. decimal coded minterm are 13, 29, 45, 61, 39, 47, 55, 63. Further pairing is not possible because of third-bit from MSB of LSP 7, X not permitted to change 1 to match with another group.

For simplification, required steps are listed below,

Step 1: Make a group of all minterms according to ascending order of LSP while listing only remaining digits.

Step 2: For subgroup considering if a number of the digit is more than one. Subgroups are formed still MSPs is separated.

Step 3: Consider all MSP in a group as minterms of an independent function and minimized using Fig. 2. Minimization also is possible using K-map [8]. For computer implementation ‘Minterm Set method’ is useful [4]. If any minterms can not pair i.e. isolated minterm with other, remain unchanged keep that same. Write minimized bit pattern of MSP term using rule 2.

Step 4: Consider group numbers (LSP) are minterm and minimized using Rule 3.

Step 5: Selection of the smallest number of prime implicants is done by using Prime Implicants Chart in the tabular method.
Step 6: Lastly variable assignment to each simplified product terms as 1 replaced with proper variable according to position without complemented form, 0 replaced with complemented form and X eliminated from the expression.

4. Examples of the Proposed Simplification

a) Example 1

\[ f(A, B, C, D, E, F) = \sum m(0, 2, 7, 8, 10, 13, 16, 18, 24, 26, 29, 31, 32, 34, 37, 39, 40, 42, 45, 47, 48, 50, 53, 55, 56, 58, 61, 63) \]

We follow binary to octal conversion i.e. direct realization from truth table; so in case of complexity analysis we consider equivalent octal minterms,

\[ f(A, B, C, D, E, F) = \sum m_8(00, 02, 07, 10, 12, 15, 20, 22, 30, 32, 34, 37, 40, 42, 45, 47, 50, 52, 55, 57, 60, 62, 65, 67, 70, 72, 75, 77) \]

Step 1: Groups are formed by checking the last digit of minterms presented in Table 1. In the case of the tabular method, we have to count all 1 present in each minterms for grouping only but in proposed method only the last digit. The proposed method effectively reduces simplification complexity.

<table>
<thead>
<tr>
<th>Table 1: Grouping of minterms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 0</td>
</tr>
<tr>
<td>Group 2</td>
</tr>
<tr>
<td>Group 5</td>
</tr>
<tr>
<td>Group 7</td>
</tr>
</tbody>
</table>

Step 2: No sub-grouping is required for this example.

Step 3: Minimization of group 0 and group 2 yields same result XXX, because it is contained all number from 0 to 7.

For minterms 1-3-7-5, 1-5-4-6 is adjacent pair of four minterms. 0-4 and 3-7 are logic adjacent Fig.3.

To determine minimized product terms, we use tricks by realizing binary from of lowest and highest minterms of the pair and change bit difference by X which indicates complemented literals. Minimized literal with group digit tabulated is Table 2.
Table 2: Minimization Table of MSP part

<table>
<thead>
<tr>
<th>Group of LSP</th>
<th>Pairing of minterms</th>
<th>Minimization of MSP part</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 0</td>
<td>0, 1, 2, 3, 4, 5, 6, 7</td>
<td>XXX</td>
</tr>
<tr>
<td>Group 2</td>
<td>0, 1, 2, 3, 4, 5, 6, 7</td>
<td>XXX</td>
</tr>
<tr>
<td>Group 5</td>
<td>1, 3, 5, 7</td>
<td>XX1</td>
</tr>
<tr>
<td></td>
<td>4, 5, 6, 7</td>
<td>1XX</td>
</tr>
<tr>
<td>Group 7</td>
<td>0, 4</td>
<td>X00</td>
</tr>
<tr>
<td></td>
<td>3, 7</td>
<td>X11</td>
</tr>
</tbody>
</table>

Step 4: Identification of adjacent LSP is done by fig.2. Here 0-2 and 5-7 are adjacent numbers. MSP bit pattern of group 0 and 2 are same; so further simplification is possible using rule 2 and minimized product term as XXX(0-2) → XXX0X0. Similarly 1XX(5-7) → 1XX1X1. XX15-X117 simplified using rule 3, and simplified product terms are XX1101, X111X1. MSP pattern X00 of group 7 does not match with other, so further simplification not possible. So simplified product terms are X00111.

The final expression of is presented below,

\[ f(A, B, C, D, E, F) = \overline{D} \overline{F} + CD\overline{E}F + BCDF + ADF + \overline{B}\overline{C}DEF. \]

Cost and expression are same as in K-Map simplification method [9].

Example 2

\[ f(P, Q, R, S, T, U) = \sum m(0, 1, 3, 7, 8, 11, 12, 16, 20, 23, 24, 27, 28, 31) \]

Equivalent octal expression of (5) as below,

\[ f(P, Q, R, S, T, U) = \sum m_8(00, 01, 03, 07, 10, 13, 14, 20, 23, 24, 27, 30, 33, 34, 37) \]

Table 3: Minimization Table of MSP part

<table>
<thead>
<tr>
<th>Group of LSP</th>
<th>Grouping of Minterms</th>
<th>Minimization of MSP part</th>
<th>Minimization of LSP part</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 0</td>
<td>0, 1, 2, 3, 4, 5, 6, 7</td>
<td>XXX</td>
<td>XXX(0-2)</td>
</tr>
<tr>
<td>Group 2</td>
<td>0, 1, 2, 3, 4, 5, 6, 7</td>
<td>XXX</td>
<td>→ XXX0X0</td>
</tr>
<tr>
<td>Group 5</td>
<td>1, 3, 5, 7</td>
<td>XX1</td>
<td>XX15-X117</td>
</tr>
<tr>
<td></td>
<td>4, 5, 6, 7</td>
<td>1XX</td>
<td>→XX1101(ii), X111X1(iii), 1XX(5-7) →1XX1X1(iv), X00111(v)</td>
</tr>
<tr>
<td>Group 7</td>
<td>0, 4</td>
<td>X00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3, 7</td>
<td>X11</td>
<td></td>
</tr>
</tbody>
</table>

The yield of step 4 is tabulated in Table 3.

Step 5: Variable assignment to each simplified product terms which are marked with a roman number under the first bracket in the fourth column of Table 5 are replaced with proper variable according to position without complemented form, 0 replaced with complemented form and X eliminate.

Table 4: Minimization Table of Example 2

<table>
<thead>
<tr>
<th>Group of MSP</th>
<th>LSP</th>
<th>Partial Minimization Using LSP</th>
<th>Minimization by pairing MSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0, 1, 3, 7</td>
<td>(0-1) 00X  (3, 7) X11</td>
<td>0000X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1-3) 0X1</td>
<td>000X1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(0X11-1011) 00X11, 0X011</td>
</tr>
<tr>
<td>1</td>
<td>0, 3, 4</td>
<td>(0, 4) X00, 011</td>
<td>(0X11-2011) 00X11, X011</td>
</tr>
<tr>
<td>2</td>
<td>0, 4, 7</td>
<td>(0, 4) X00, 111</td>
<td>(2-3)X00 1XX00</td>
</tr>
<tr>
<td>3</td>
<td>0, 3, 4, 7</td>
<td>(0, 4) X00, (3, 7) X11</td>
<td>(1-3)X00 X1X00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11X11</td>
</tr>
</tbody>
</table>

Using prime implicant chart the essential prime implicants are determined from Table 4 and essential prime implicants are 1XX00, X1X00, 0000X, 11X11, X0111, 0X011.

5. Comparison and Analysis

In general for traditional approach, K map required representing the minterms into decimal number and the resulted input combinations are placed into the map. In this proposed method, octal minterms are required which are easier to represent and manipulation in comparisons with decimal coded minterms. Figure 4(a), illustrate Karnaugh Map for four variables and Figure 4(b), and represent the circular octal chart.

![Figure 4](image_url)

Octal minterms are easier to represent and easily converted into binary number which are useful for minimization in comparison with decimal coded minterms. Another advantage is less number of comparisons is required for minimization in comparison with decimal coded minterms. For seven and eight inputs system required the number of map level are eight and sixteen respectively and also identification of adjacent cell become more difficult. This process also increases complexity minimization of multiple inputs logic synthesis. Decimal coded minterms are not suitable for minimization because all simplifications approached are based on binary representation or Gray code. Decimal coded minterms is not played any significance role in minimization process except representations of input conditions. These difficulties successfully overcome using proposed octal representation and minimization technique. Since given minterms are in decimal coded, we have to convert in equivalent octal number. This step was initially avoided when designer realized decimal coded minterms from the truth table. Conversion from binary to decimal conversion requires more operation in comparison with binary to octal conversion. We do not follow the way binary to octal realization via decimal conversion. The method easily overcome or minimized all those difficulties successfully. If minterms are given in decimal number then all minterms first convert into the binary number for further process. Quine-McCluskey minimization method also deals with binary coded minterms. Representation of minterms in binary coding also required more space and not user-friendly.

6. Conclusion

Here we proposed a new approach for simplification of switching circuit based on the octal representation of minterms. This process reduces complexity effectively. Representations of minterms of multiple inputs are the big issues. The proposed octal minterms based minimization method is required less number of the comparison. This method can easily be used for any number of variables. By using decimal coded minterms, it is hard to realize minimized product terms. The results, which are verified with available experimental data, indicated that the minimal sum-of-product terms are achieved with minimal effort.

References


