Computational Simulation of Quine-McCluskey Algorithm using C/C++

Digital Logic Circuits (EEE2040.01-00), Yonsei University

Taewan Ham[1]

wani7@yonsei.ac.kr

Ⅰ. Introduction

Karnaugh Map method is often used to simplify Boolean functions. It is an intuitive and effective method. However, when the number of variables become larger, additional method is required due to its complexity. Quine-McCluskey algorithm provides more systematic approach than Karnaugh Map method. Therefore, it enables the usage of computer. The goal of this project is to create a computational simulation of Quine-McCluskey algorithm using C/C++.

Ⅱ. Theoretical explanation of Quine-McCluskey Algorithm

Quine-McCluskey algorithm is consisted in two big steps. The first step is figuring out the prime implicants. Prime implicants can be found by representing min-terms in binary notation and repetitively combining two min-terms if they have only one different variable. To reduce the calculation, each min-terms are grouped based on the number of 1's according to their binary notation. The combination continues until there is no available implicants to combine. Implicants that can't be combined anymore is called prime implicants.

The second step is selecting the minimum set of prime implicants. After the first step, set of prime implicants is selected. Based on this prime implicants, we can construct a prime implicant chart. The row of the chart means prime implicants. The column of the chart means min-terms. If a prime implicant is the only implicant covering the min-term, the prime implicant covering it is called 'essential prime implicant'. After finding the essential prime implicant, we must find the minimum set of prime implicants that can cover all the remaining min-terms. The set of prime implicants can be found by using combination (trial and error) or Petrick's method. The selected set of prime implicants can be expressed in sum-of-products form.

Following is an example of using Quine-McCluskey algorithm on four variable Boolean function. Let's say f(A, B, C, D) = Σm(4, 8, 9, 10, 11, 12, 14, 15). Each min-terms are represented in binary notation and grouped according to their number of 1's.

min-term	Binary notation	Number of 1's (Group)
m(4)	0100	1
m(8)	1000	

[1] *Undergraduate, School of Electrical and Electronic Engineering, Yonsei University*

m(9)	1001	2
m(10)	1010	
m(12)	1100	
m(11)	1011	3
m(14)	1110	
m(15)	1111	4

Table 1. Initial procedure

After the initial procedure each min-terms are combined if only one variable differs. The different digit is notated as '_'.

size 2		size 4	
m(4, 12)	_100	m(8,9,10,11)	10__
m(8, ,9)	100_	m(8,10,12,14)	1__0
m(8, 10)	10_0	m(10,11,14,15)	1_1_
m(8, 12)	1_00		
m(9, 11)	10_1		
m(10, 11)	10_1		
m(10, 14)	1_10		
m(12, 14)	11_0		
m(11, 15)	1_11		
m(14, 15)	111_		

Table 2. Combining min-terms and repeating it

After finding the prime implicants, we need to construct a prime implicant chart.

	4	**8**	**9**	**10**	**11**	**12**	**14**	**15**
m(4, 12)	x					x		
m(8, 9, 10, 11)		x	x	x	x			
m(8, 10, 12, 14)		x		x		x	x	
m(10, 11, 14, 15)				x	x		x	x

Table 3. Prime implicant chart

Based on this prime implicant chart, we conduct either trial/error method or Petrick's method to solve the minimum sum-of-products of the Boolean function. In trial/error method, we pick the minimum set of min-terms that can eliminate the 'x marks' in the prime implicant chart. In Petrick's method, we make a logical function which is consisted of sum-of-products of Boolean expression that makes each column true. Then, we simplify the logical function by using X + X = X and X + XY = X.

Ⅲ. Implementation of Quine-McCluskey Algorithm in C/C++ code

The overall of the code is consisted in three steps. The first step is getting input from the user. I made error exception using while loop. The second step is finding the prime implicants. The third step is finding the essential prime implicants. For further explanation, please check out the code comment.

Ⅳ. Results of the Simulation

Function (1) - (6) are given test functions. The result of each test functions is shown in Table 4. Screenshots of the result is attached in the end of the paper.

<table>
<tr><th>(1) $F(a, b, c, d) = \sum m(2, 4, 5, 6, 9, 10, 11, 12, 13, 15)$</th><th>(2) $F(a, b, c, d) = \sum m(0, 3, 4, 5, 7, 9, 11, 13)$</th></tr>
<tr><td>----QM operation----
A B C D g u f
----
0 0 1 0 1 2 1 0 ||| m(2)
0 1 0 0 1 3 1 0 ||| m(4)
0 1 0 1 2 2 1 0 ||| m(5)
0 1 1 0 2 2 1 0 ||| m(6)
1 0 0 1 2 2 1 0 ||| m(9)
1 0 1 0 2 2 1 0 ||| m(10)
1 0 1 1 3 3 1 0 ||| m(11)
1 1 0 0 2 2 1 0 ||| m(12)
1 1 0 1 3 4 1 0 ||| m(13)
1 1 1 1 4 2 1 0 ||| m(15)

----QM operation----
A B C D g u f
----
0 -1 1 0 1 0 1 0 ||| m(2 6)
-1 0 1 0 1 0 1 0 ||| m(2 10)
0 1 0 -1 1 1 1 0 ||| m(4 5)
0 1 -1 0 1 0 1 0 ||| m(4 6)
-1 1 0 0 1 1 1 0 ||| m(4 12)
-1 1 0 1 2 1 1 0 ||| m(5 13)
1 0 -1 1 2 1 1 0 ||| m(9 11)
1 -1 0 1 2 1 1 0 ||| m(9 13)
1 0 1 -1 2 0 1 0 ||| m(10 11)
1 1 0 -1 2 1 1 0 ||| m(12 13)
1 -1 1 1 3 1 1 0 ||| m(11 15)
1 1 -1 1 3 1 1 0 ||| m(13 15)

----QM operation----
A B C D g u f

-1 1 0 -1 1 0 1 0 ||| m(4 5 12 13)
1 -1 -1 1 2 0 1 0 ||| m(9 11 13 15)

----List of Prime Implicants----
m(2 6)
m(2 10)
m(4 6)
m(10 11)
m(4 5 12 13)
m(9 11 13 15)

----Prime Implicants Chart----
2 4 5 6 9 10 11 12 13 15
----
2 0 0 2 0 0 0 0 0 0</td><td>----QM operation----
A B C D g u f
----
0 0 0 0 0 1 1 0 ||| m(0)
0 0 1 1 2 2 1 0 ||| m(3)
0 1 0 0 1 2 1 0 ||| m(4)
0 1 0 1 2 3 1 0 ||| m(5)
0 1 1 1 3 2 1 0 ||| m(7)
1 0 0 1 2 2 1 0 ||| m(9)
1 0 1 1 3 2 1 0 ||| m(11)
1 1 0 1 3 2 1 0 ||| m(13)

----QM operation----
A B C D g u f
----
0 -1 0 0 0 0 1 0 ||| m(0 4)
0 1 0 -1 1 0 1 0 ||| m(4 5)
0 -1 1 1 2 0 1 0 ||| m(3 7)
-1 0 1 1 2 0 1 0 ||| m(3 11)
0 1 -1 1 2 0 1 0 ||| m(5 7)
-1 1 0 1 2 0 1 0 ||| m(5 13)
1 0 -1 1 2 0 1 0 ||| m(9 11)
1 -1 0 1 2 0 1 0 ||| m(9 13)

----List of Prime Implicants----
m(0 4)
m(4 5)
m(3 7)
m(3 11)
m(5 7)
m(5 13)
m(9 11)
m(9 13)

----Prime Implicants Chart----
0 3 4 5 7 9 11 13
----
1 0 2 0 0 0 0 0
0 0 2 3 0 0 0 0
0 2 0 0 2 0 0 0
0 2 0 0 0 0 2 0
0 0 0 3 2 0 0 0
0 0 0 3 0 0 0 2
0 0 0 0 0 2 2 0
0 0 0 0 0 2 0 2

----Essential Prime Implicants----
A'C'D'</td></tr>
</table>

<table>
<tr><td>2 0 0 0 0 2 0 0 0 0
0 2 0 2 0 0 0 0 0 0
0 0 0 0 0 2 2 0 0 0
0 2 1 0 0 0 0 1 2 0
0 0 0 0 1 0 2 0 2 1

----Essential Prime Implicants----
BC'
AD</td><td></td></tr>
<tr><td>$F(a, b, c, d) = \sum m(0, 1, 2, 3, 5, 7, 8, 10, 12, 13, 15)$</td><td>$F(a, b, c, d, e) = \sum m(0, 1, 2, 4, 8, 11, 13, 14, 15, 17, 18, 20, 21, 26, 27, 30, 31)$</td></tr>
<tr><td>----QM operation----
A B C D g u f
----
0 0 0 0 0 3 1 0 ||| m(0)
0 0 0 1 1 3 1 0 ||| m(1)
0 0 1 0 1 3 1 0 ||| m(2)
0 0 1 1 2 3 1 0 ||| m(3)
0 1 0 1 2 3 1 0 ||| m(5)
0 1 1 1 3 3 1 0 ||| m(7)
1 0 0 0 1 3 1 0 ||| m(8)
1 0 1 0 2 2 1 0 ||| m(10)
1 1 0 0 2 2 1 0 ||| m(12)
1 1 0 1 3 3 1 0 ||| m(13)
1 1 1 1 4 2 1 0 ||| m(15)

----QM operation----
A B C D g u f
----
0 0 0 -1 0 1 1 0 ||| m(0 1)
0 0 -1 0 0 2 1 0 ||| m(0 2)
-1 0 0 0 0 1 1 0 ||| m(0 8)
0 0 -1 1 1 2 1 0 ||| m(1 3)
0 -1 0 1 1 1 1 0 ||| m(1 5)
0 0 1 -1 1 1 1 0 ||| m(2 3)
-1 0 1 0 1 1 1 0 ||| m(2 10)
1 0 -1 0 1 1 1 0 ||| m(8 10)
1 -1 0 0 1 0 1 0 ||| m(8 12)
0 -1 1 1 2 1 1 0 ||| m(3 7)
0 1 -1 1 2 2 1 0 ||| m(5 7)
-1 1 0 1 2 1 1 0 ||| m(5 13)
1 1 0 -1 2 0 1 0 ||| m(12 13)
-1 1 1 1 3 1 1 0 ||| m(7 15)
1 1 -1 1 3 1 1 0 ||| m(13 15)

----QM operation----
A B C D g u f
----
0 0 -1 -1 0 0 1 0 ||| m(0 1 2 3)
-1 0 -1 0 0 0 1 0 ||| m(0 2 8 10)
0 -1 -1 1 1 0 1 0 ||| m(1 3 5 7)
-1 1 -1 1 2 0 1 0 ||| m(5 7 13 15)

----List of Prime Implicants----
m(8 12)
m(12 13)
m(0 1 2 3)
m(0 2 8 10)
m(1 3 5 7)</td><td>----QM operation----
A B C D E g u f
----
0 0 0 0 0 0 4 1 ||| m(0)
0 0 0 0 1 1 2 1 ||| m(1)
0 0 0 1 0 1 2 1 ||| m(2)
0 0 1 0 0 1 2 1 ||| m(4)
0 1 0 0 0 1 1 1 ||| m(8)
0 1 0 1 1 3 2 1 ||| m(11)
0 1 1 0 1 3 1 1 ||| m(13)
0 1 1 1 0 3 2 1 ||| m(14)
0 1 1 1 1 4 4 1 ||| m(15)
1 0 0 0 1 2 2 1 ||| m(17)
1 0 0 1 0 2 2 1 ||| m(18)
1 0 1 0 0 2 2 1 ||| m(20)
1 0 1 0 1 3 2 1 ||| m(21)
1 1 0 1 0 3 3 1 ||| m(26)
1 1 0 1 1 4 3 1 ||| m(27)
1 1 1 1 0 4 3 1 ||| m(30)
1 1 1 1 1 5 3 1 ||| m(31)

----QM operation----
A B C D E g u f
----
0 0 0 0 -1 0 0 1 ||| m(0 1)
0 0 0 -1 0 0 0 1 ||| m(0 2)
0 0 -1 0 0 0 0 1 ||| m(0 4)
0 -1 0 0 0 0 0 1 ||| m(0 8)
-1 0 0 0 1 1 0 1 ||| m(1 17)
-1 0 0 1 0 1 0 1 ||| m(2 18)
-1 0 1 0 0 1 0 1 ||| m(4 20)
1 0 -1 0 1 2 0 1 ||| m(17 21)
1 -1 0 1 0 2 0 1 ||| m(18 26)
1 0 1 0 -1 2 0 1 ||| m(20 21)
0 1 -1 1 1 3 1 1 ||| m(11 15)
-1 1 0 1 1 3 1 1 ||| m(11 27)
0 1 1 -1 1 3 0 1 ||| m(13 15)
0 1 1 1 -1 3 1 1 ||| m(14 15)
-1 1 1 1 0 3 1 1 ||| m(14 30)
1 1 0 1 -1 3 1 1 ||| m(26 27)
1 1 -1 1 0 3 1 1 ||| m(26 30)
-1 1 1 1 1 4 2 1 ||| m(15 31)
1 1 -1 1 1 4 2 1 ||| m(27 31)
1 1 1 1 -1 4 2 1 ||| m(30 31)

----QM operation----
A B C D E g u f
--------------------</td></tr>
</table>

<table>
<tr>
<td>m(5 7 13 15)

----Prime Implicants Chart----
0 1 2 3 5 7 8 10 12 13 15
----
0 0 0 0 0 0 2 0 2 0 0
0 0 0 0 0 0 0 0 2 2 0
2 2 2 2 0 0 0 0 0 0 0
2 0 2 0 0 0 2 1 0 0 0
0 2 0 2 2 2 0 0 0 0 0
0 0 0 0 2 2 0 0 0 2 1

----Essential Prime Implicants----
B'D'
BD</td>
<td>-1 1 -1 1 1 3 0 1 ||| m(11 15 27 31)
-1 1 1 1 -1 3 0 1 ||| m(14 15 30 31)
1 1 -1 1 -1 3 0 1 ||| m(26 27 30 31)

----List of Prime Implicants----
m(0 1)
m(0 2)
m(0 4)
m(0 8)
m(1 17)
m(2 18)
m(4 20)
m(17 21)
m(18 26)
m(20 21)
m(13 15)
m(11 15 27 31)
m(14 15 30 31)
m(26 27 30 31)

----Prime Implicants Chart----
0 1 2 4 8 11 13 14 15 17 18 20 21 26 27 30 31
----
4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0
0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0 0
0 0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0
0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0
0 0 0 0 0 0 1 0 3 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 3 0 0 0 0 0 2 0 3
0 0 0 0 0 0 0 1 3 0 0 0 0 0 0 2 3
0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 3

----Essential Prime Implicants----
A'C'D'E'
A'BCE
BDE
BCD</td>
</tr>
<tr>
<td>F(a, b, c, d, e) = ∑ m(0, 1, 2, 3, 4, 8, 8, 10, 11, 19, 21,
22, 23, 27, 28, 29, 30)</td>
<td>F(a, b, c, d, e) = ∑ m(1, 3, 4, 5, 9, 11, 12, 13, 18, 22,
23, 26, 30, 31)</td>
</tr>
<tr>
<td>----QM operation----
A B C D E g u f
----
0 0 0 0 0 0 4 1 ||| m(0)
0 0 0 0 1 1 3 1 ||| m(1)
0 0 0 1 0 1 3 1 ||| m(2)
0 0 0 1 1 2 4 1 ||| m(3)
0 0 1 0 0 1 1 1 ||| m(4)
0 1 0 0 0 1 3 1 ||| m(8)
0 1 0 0 1 2 3 1 ||| m(9)
0 1 0 1 0 2 3 1 ||| m(10)
0 1 0 1 1 3 4 1 ||| m(11)
1 0 0 1 1 3 3 1 ||| m(19)</td>
<td>----QM operation----
A B C D E g u f
----
0 0 0 0 1 1 3 1 ||| m(1)
0 0 0 1 1 2 2 1 ||| m(3)
0 0 1 0 0 1 2 1 ||| m(4)
0 0 1 0 1 2 3 1 ||| m(5)
0 1 0 0 1 2 3 1 ||| m(9)
0 1 0 1 1 3 2 1 ||| m(11)
0 1 1 0 0 2 2 1 ||| m(12)
0 1 1 0 1 3 3 1 ||| m(13)
1 0 0 1 0 2 2 1 ||| m(18)
1 0 1 1 0 3 3 1 ||| m(22)</td>
</tr>
</table>

1 0 1 0 1 3 2 1 ||| m(21)
1 0 1 1 0 3 2 1 ||| m(22)
1 0 1 1 1 4 3 1 ||| m(23)
1 1 0 1 1 4 2 1 ||| m(27)
1 1 1 0 0 3 2 1 ||| m(28)
1 1 1 0 1 4 2 1 ||| m(29)
1 1 1 1 0 4 2 1 ||| m(30)

----QM operation----
A B C D E g u f
----
0 0 0 0 -1 0 2 1 ||| m(0 1)
0 0 0 -1 0 0 2 1 ||| m(0 2)
0 0 -1 0 0 0 0 1 ||| m(0 4)
0 -1 0 0 0 0 2 1 ||| m(0 8)
0 0 0 -1 1 1 2 1 ||| m(1 3)
0 -1 0 0 1 1 2 1 ||| m(1 9)
0 0 0 1 -1 1 2 1 ||| m(2 3)
0 -1 0 1 0 1 2 1 ||| m(2 10)
0 1 0 0 -1 1 2 1 ||| m(8 9)
0 1 0 -1 0 1 2 1 ||| m(8 10)
0 -1 0 1 1 2 3 1 ||| m(3 11)
-1 0 0 1 1 2 1 1 ||| m(3 19)
0 1 0 -1 1 2 2 1 ||| m(9 11)
0 1 0 1 -1 2 2 1 ||| m(10 11)
-1 1 0 1 1 3 1 1 ||| m(11 27)
1 0 -1 1 1 3 0 1 ||| m(19 23)
1 -1 0 1 1 3 1 1 ||| m(19 27)
1 0 1 -1 1 3 0 1 ||| m(21 23)
1 -1 1 0 1 3 0 1 ||| m(21 29)
1 0 1 1 -1 3 0 1 ||| m(22 23)
1 -1 1 1 0 3 0 1 ||| m(22 30)
1 1 1 0 -1 3 0 1 ||| m(28 29)
1 1 1 -1 0 3 0 1 ||| m(28 30)

----QM operation----
A B C D E g u f
----
0 0 0 -1 -1 0 1 1 ||| m(0 1 2 3)
0 -1 0 0 -1 0 1 1 ||| m(0 1 8 9)
0 -1 0 -1 0 0 1 1 ||| m(0 2 8 10)
0 -1 0 -1 1 1 1 1 ||| m(1 3 9 11)
0 -1 0 1 -1 1 1 1 ||| m(2 3 10 11)
0 1 0 -1 -1 1 1 1 ||| m(8 9 10 11)
-1 -1 0 1 1 2 0 1 ||| m(3 11 19 27)

----QM operation----
A B C D E g u f
----
0 -1 0 -1 -1 0 0 1 ||| m(0 1 2 3 8 9 10 11)

----List of Prime Implicants----
m(0 4)
m(19 23)
m(21 23)
m(21 29)
m(22 23)
m(22 30)
m(28 29)

1 0 1 1 1 4 2 1 ||| m(23)
1 1 0 1 0 3 2 1 ||| m(26)
1 1 1 1 0 4 3 1 ||| m(30)
1 1 1 1 1 5 2 1 ||| m(31)

----QM operation----
A B C D E g u f
----
0 0 0 -1 1 1 1 1 ||| m(1 3)
0 0 -1 0 1 1 1 1 ||| m(1 5)
0 -1 0 0 1 1 2 1 ||| m(1 9)
0 0 1 0 -1 1 1 1 ||| m(4 5)
0 -1 1 0 0 1 1 1 ||| m(4 12)
0 -1 0 1 1 2 1 1 ||| m(3 11)
0 -1 1 0 1 2 2 1 ||| m(5 13)
0 1 0 -1 1 2 1 1 ||| m(9 11)
0 1 -1 0 1 2 1 1 ||| m(9 13)
0 1 1 0 -1 2 1 1 ||| m(12 13)
1 0 -1 1 0 2 1 1 ||| m(18 22)
1 -1 0 1 0 2 1 1 ||| m(18 26)
1 0 1 1 -1 3 1 1 ||| m(22 23)
1 -1 1 1 0 3 2 1 ||| m(22 30)
1 1 -1 1 0 3 1 1 ||| m(26 30)
1 -1 1 1 1 4 1 1 ||| m(23 31)
1 1 1 1 -1 4 1 1 ||| m(30 31)

----QM operation----
A B C D E g u f
----
0 -1 0 -1 1 1 0 1 ||| m(1 3 9 11)
0 -1 -1 0 1 1 0 1 ||| m(1 5 9 13)
0 -1 1 0 -1 1 0 1 ||| m(4 5 12 13)
1 -1 -1 1 0 2 0 1 ||| m(18 22 26 30)
1 -1 1 1 -1 3 0 1 ||| m(22 23 30 31)

----List of Prime Implicants----
m(1 3 9 11)
m(1 5 9 13)
m(4 5 12 13)
m(18 22 26 30)
m(22 23 30 31)

----Prime Implicants Chart----
1 3 4 5 9 11 12 13 18 22 23 26 30 31
----
2 1 0 0 2 1 0 0 0 0 0 0 0 0
2 0 0 2 2 0 0 2 0 0 0 0 0 0
0 0 1 2 0 0 1 2 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 2 0 1 2 0
0 0 0 0 0 0 0 0 0 2 1 0 2 1

----Essential Prime Implicants----
A'C'E
A'CD'
ADE'
ACD

m(28 30) m(3 11 19 27) m(0 1 2 3 8 9 10 11) ----Prime Implicants Chart---- 0 1 2 3 4 8 9 10 11 19 21 22 23 27 28 29 30 ------------------------------ 2 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 0 0 0 2 0 0 0 0 2 2 0 0 0 1 0 0 0 2 1 1 2 0 1 1 1 2 0 0 0 0 0 0 0 0 ----Essential Prime Implicants---- A'B'D'E' C'DE A'C'	

Ⅴ. Code manual

<Input>

Step 1 : Enter number of variables. If input does not satisfy the condition, the system prints a warning and repeat Step 1.

Step 2 : Enter number of min-terms. Don't-care terms are ignored. If input does not satisfy the condition, the system prints a warning and repeat Step 1.

Step 3 : Enter list of min-terms

<Output>

1. Each stage of Quine-McCluskey algorithm.
2. List of prime implicants
3. Prime Implicant Chart
4. Essential Prime Implicants

Ⅵ. Full text of the code

Full text of the code is attached to the report for anti-plagiarism test.

```
/*
=================================================================
"Computational Simulation of Quine-McCluskey Algorithm using C/C++"
```

```
        - Digital Logic Circuits, Yonsei university-
==================================================================
Date        | Author
11/30/2022 | Taewan Ham
==================================================================
*/

#include <iostream>
#include <math.h>
#include <vector>
#include <set>
#include <string>
#include <algorithm>

using namespace std;

//variables
int num_of_var = 0;  // number of variables (4 or 5)
int num_of_min = 0;  // number of min-terms
bool bool_tmp = true;  // temporary variable for while loop in step 1
bool flag = true;  // temporary variable for while loop in step 2
int min_index = 0;  // index for vector minterm_list
int size_imp = 1;  // size of implicants in tmp_table
int l = 0;  // index number for qm_table and tmp_table
int u = 0;  // used
int g = 0;  // group
int f = 0;  // f in init_var4,5_table

// arrays
// size of the column of each table is fixed to 8 to use as parameter in functions
int init_table[32][8];
int init_4var_table[16][8] = {  // initial table of four variable minterms
  // A, B, C, D, g, u, f - g: group number, f: 0 or 1, u: used(1) not used(0)
         {0, 0, 0, 0, 0, 0, 0, 0},  //m0
         {0, 0, 0, 1, 1, 0, 0, 0},  //m1
         {0, 0, 1, 0, 1, 0, 0, 0},  //m2
         {0, 0, 1, 1, 2, 0, 0, 0},  //m3
         {0, 1, 0, 0, 1, 0, 0, 0},  //m4
         {0, 1, 0, 1, 2, 0, 0, 0},  //m5
         {0, 1, 1, 0, 2, 0, 0, 0},  //m6
         {0, 1, 1, 1, 3, 0, 0, 0},  //m7
         {1, 0, 0, 0, 1, 0, 0, 0},  //m8
         {1, 0, 0, 1, 2, 0, 0, 0},  //m9
         {1, 0, 1, 0, 2, 0, 0, 0},  //m10
         {1, 0, 1, 1, 3, 0, 0, 0},  //m11
         {1, 1, 0, 0, 2, 0, 0, 0},  //m12
         {1, 1, 0, 1, 3, 0, 0, 0},  //m13
         {1, 1, 1, 0, 3, 0, 0, 0},  //m14
         {1, 1, 1, 1, 4, 0, 0, 0},  //m15
};
int init_5var_table[32][8] = {  // initial table of five variable minterms
  // A, B, C, D, E, g, u, f - g: group number, f: 0 or 1, u: used(1) not used(0)
         {0, 0, 0, 0, 0, 0, 0, 0},  //m0
         {0, 0, 0, 0, 1, 1, 0, 0},  //m1
         {0, 0, 0, 1, 0, 1, 0, 0},  //m2
         {0, 0, 0, 1, 1, 2, 0, 0},  //m3
         {0, 0, 1, 0, 0, 1, 0, 0},  //m4
         {0, 0, 1, 0, 1, 2, 0, 0},  //m5
         {0, 0, 1, 1, 0, 2, 0, 0},  //m6
         {0, 0, 1, 1, 1, 3, 0, 0},  //m7
         {0, 1, 0, 0, 0, 1, 0, 0},  //m8
         {0, 1, 0, 0, 1, 2, 0, 0},  //m9
         {0, 1, 0, 1, 0, 2, 0, 0},  //m10
         {0, 1, 0, 1, 1, 3, 0, 0},  //m11
```

```
        {0, 1, 1, 0, 0, 2, 0, 0},  //m12
        {0, 1, 1, 0, 1, 3, 0, 0},  //m13
        {0, 1, 1, 1, 0, 3, 0, 0},  //m14
        {0, 1, 1, 1, 1, 4, 0, 0},  //m15
        {1, 0, 0, 0, 0, 1, 0, 0},  //m16
        {1, 0, 0, 0, 1, 2, 0, 0},  //m17
        {1, 0, 0, 1, 0, 2, 0, 0},  //m18
        {1, 0, 0, 1, 1, 3, 0, 0},  //m19
        {1, 0, 1, 0, 0, 2, 0, 0},  //m20
        {1, 0, 1, 0, 1, 3, 0, 0},  //m21
        {1, 0, 1, 1, 0, 3, 0, 0},  //m22
        {1, 0, 1, 1, 1, 4, 0, 0},  //m23
        {1, 1, 0, 0, 0, 2, 0, 0},  //m24
        {1, 1, 0, 0, 1, 3, 0, 0},  //m25
        {1, 1, 0, 1, 0, 3, 0, 0},  //m26
        {1, 1, 0, 1, 1, 4, 0, 0},  //m27
        {1, 1, 1, 0, 0, 3, 0, 0},  //m28
        {1, 1, 1, 0, 1, 4, 0, 0},  //m29
        {1, 1, 1, 1, 0, 4, 0, 0},  //m30
        {1, 1, 1, 1, 1, 5, 0, 0},  //m31
};
int tmp_table[32][8];  // table used in step 2
int qm_table[32][8];  // table used in step 2 (quine-mccluskey table)
int num_list[32] = { };  // number list (index : 0~31, initial value : 0)

// vectors
vector <int> vector_tmp;  // temporary vector
vector <int> epi;  // list of essential prime implicant
vector <vector <int>> minterm_list;  // list of minterms used in step 2
vector <vector <int>> tmp_minterm_list;  // list of minterms used in step 2
vector <vector <int>> final_minterm_list;  // final list of minterms (prime implicants)
vector <vector <int>> final_implicants;  // final list of implicants (prime implicants)

// functions
bool compare_digits(int arr[][8], int n, int m, int var_num);  // compare each two digits
number (n & m)
void initialize_table(int arr[][8]);  // initialize elements of arr in to 0
void print_QM_op(int var_num);  // prints column info of qm operation in step 2
string binary_to_literal(vector <int> binary, int var_num);  // changes binary notation in
to literal notation


int main(void) {
        // ---- Step 1 : getting input from user ----
        while (bool_tmp) {  // getting number of variables
                printf("Please enter number of variables : ");
                scanf_s("%d", &num_of_var);  // input of number of variables (ex: 4, 5)
                if (num_of_var == 4 || num_of_var == 5) bool_tmp = false;
                else printf("Only 4 or 5 variables operation is available...\n");  //
repeat until it gets the correct input
        }
        if (num_of_var == 4) {  // if variable is 4
                g = 4; u = 5; f = 6;
                std::copy(&init_4var_table[0][0], &init_4var_table[0][0] + 16 * 8,
&init_table[0][0]);
        }
        else {  // if variable is 5
                g = 5; u = 6; f = 7;
                std::copy(&init_5var_table[0][0], &init_5var_table[0][0] + 32 * 8,
&init_table[0][0]);
        }

        bool_tmp = true;
        printf("\n");
```

```
	while (bool_tmp) {
		printf("Please enter number of min-terms : ");
		scanf_s("%d", &num_of_min);  // input of number of minterms (maximum: 
2^(num_of_var)-1)
		if (num_of_var == 4 && num_of_min < 16) bool_tmp = 0;
		else if (num_of_var == 5 && num_of_min < 32) bool_tmp = 0;
		else printf("Please check your input again...\n");  // repeat until it 
gets the correct input
	}

	printf("\n");
	printf("Please enter minterms : ");
	for (int i = 0; i < num_of_min; i++) {  // getting minterms as input
		int tmp = 0;
		scanf_s("%d", &tmp);
		init_table[tmp][f] = 1;  // update initial table
	}

	// ---- Step 2 : Finding Prime implicants ----
	int cnt = 0;

	for (int i = 0; i < int(pow(2, num_of_var)); i++) {  //initializing qm_table with 
init_table
		if (init_table[i][f] == 1) {
			for (int j = 0; j < 8; j++) qm_table[l][j] = init_table[i][j];
			minterm_list.push_back(std::vector <int>());
			minterm_list[l].push_back(i);
			l++;
		}
	}

	while (flag) {
		int group = 0;
		int s = 0; // size of tmp_table
		flag = false;  // initialize flag
		size_imp *= 2;
		while (group < num_of_var) {  // operation (qm_table -> tmp_table)
			for (int i = 0; i < l; i++) {
				if (qm_table[i][f] == 1 && qm_table[i][g] == group) {
					for (int j = 0; j < l; j++) {
						if (qm_table[j][f] == 1 && 
qm_table[j][g] == group + 1) {
							if 
(compare_digits(qm_table, i, j, num_of_var)) {  // if two number satisfies the condition
								qm_table[i][u]++; 
// check 'used'
								qm_table[j][u]++;
								bool fflag = 
false;

//tmp_minterm_list.push_back(std::vector <int>());
								for (int n = 0; n 
< minterm_list[i].size(); n++) vector_tmp.push_back(minterm_list[i][n]); // record minterms
								for (int n = 0; n 
< minterm_list[j].size(); n++) vector_tmp.push_back(minterm_list[j][n]);

sort(vector_tmp.begin(), vector_tmp.end());
								for (int n = 0; n 
< tmp_minterm_list.size(); n++) {

std::set<vector<int>> s;

s.insert(tmp_minterm_list[n]);
```

```
auto
rep = s.insert(vector_tmp);
if
(!rep.second) {  // if it is repeated
        fflag = true;
        break;
}
}
if (!fflag) { //
if it is not a repeated operation
tmp_minterm_list.push_back(vector_tmp);
sort(tmp_minterm_list[s].begin(), tmp_minterm_list[s].end());
for
(int k = 0; k < num_of_var; k++) {
        if (qm_table[i][k] == qm_table[j][k]) tmp_table[s][k] = qm_table[i][k];
        else tmp_table[s][k] = -1;
}
tmp_table[s][f] = 1;
tmp_table[s][g] = group;
s++;
flag =
true;  // flag is false if there is no available combining
}
vector_tmp.clear();
                                        }
                                }
                        }
                }
        }
        group++;
}
for (int i = 0; i < l; i++) {  // find prime implicants
        if (qm_table[i][f] == 1 && qm_table[i][u] == 0) {
                final_minterm_list.push_back(minterm_list[i]);
                for (int j = 0; j < num_of_var; j++) {
                        vector_tmp.push_back(qm_table[i][j]);
                }
                final_implicants.push_back(vector_tmp);
        }
        vector_tmp.clear();
}

// print each stage of operation
printf("\n----QM operation----\n");
print_QM_op(num_of_var);
for (int q = 0; q < l; q++) {
        for (int w = 0; w < 8; w++) printf("%d ", qm_table[q][w]);
        printf("||| m( ");
        for (int e = 0; e < minterm_list[q].size(); e++)  printf("%d ",
minterm_list[q][e]);
        printf(")\n");
}
```

```
            // qm reset, qm = tmp, tmp reset
            initialize_table(qm_table);
            std::copy(&tmp_table[0][0], &tmp_table[0][0] + s * 8, &qm_table[0][0]);
            initialize_table(tmp_table);

            // minterm_list reset, min = tmp, tmp reset
            minterm_list.assign(tmp_minterm_list.size(),
std::vector<int>(tmp_minterm_list.size()));
            std::copy(tmp_minterm_list.begin(), tmp_minterm_list.end(),
minterm_list.begin());
            tmp_minterm_list.assign(0, std::vector<int>(0));

            // initialize index values
            l = s;
            s = 0;  // initializing size of qm_table
        }

        // ---- Step 3: Prime Implicant Chart ----
        int pi_num = final_minterm_list.size();  // number of rows of the prime implicant
chart
        std::vector <int> col_chart;  // column of the prime implicant chart

        // printing prime implicants
        printf("\n----List of Prime Implicants----\n");
        for (int i = 0; i < pi_num; i++) {
            printf("m( ");
            for (int j = 0; j < final_minterm_list[i].size(); j++) {
                printf("%d ", final_minterm_list[i][j]);
                num_list[final_minterm_list[i][j]]++;  // count how many time
the min-term is included in prime implicants
            }
            printf(")\n");
        }

        //  make a vector that contains min-term list
        for (int i = 0; i < 32; i++) {
            if (num_list[i] != 0) {
                col_chart.push_back(i);
            }
        }


        int col_size = col_chart.size();
        std::vector< vector <int>> pi_chart;  // prime implicant chart

        // creating prime implicant chart (pi_chart)
        for (int i = 0; i < pi_num; i++) {
            bool fflag = true;  // flag for epi
            for (int j = 0; j < col_size; j++) {
                bool fflag_ = true;  // flag for pushing back 0 in pi_chart
                for (int k = 0; k < final_minterm_list[i].size(); k++) {
                    if (final_minterm_list[i][k] == col_chart[j]) {

vector_tmp.push_back(num_list[col_chart[j]]);  // num_list[col_chart[j]] is equal to the
frequency of min-term
                        if (num_list[col_chart[j]] == 1 && fflag ==

true) {
                            epi.push_back(i);  // essential

prime implicant
                            fflag = false;  // prevent repeat

of same epi value
                        }
                        fflag_ = false;
                    }
```

```
                        }
                        if(fflag_) vector_tmp.push_back(0);  // push 0 if there is a
number in col_chart[] but not in final_minterm_list[][]
                }
                pi_chart.push_back(vector_tmp);
                vector_tmp.clear();
        }

        printf("\n----Prime Implicants Chart----\n");
        for (int i = 0; i < col_size; i++) printf("%d ", col_chart[i]);  // print column
info (which is list of min-terms
        printf("\n------------------------------\n");
        for (int i = 0; i < pi_num; i++) {  // print prime implicant chart
                for (int j = 0; j < col_size; j++) {
                        printf("%d ", pi_chart[i][j]);
                }
                printf("\n");
        }

        printf("\n----Essential Prime Implicants----\n");
        for (int i = 0; i < epi.size(); i++) {  // print essential prime implicants
                string str_tmp = binary_to_literal(final_implicants[epi[i]],
num_of_var);
                for(int j = 0; j < str_tmp.size(); j++) printf("%c", str_tmp[j]); //
print literals
                printf("\n");
        }

        return 0;
}

bool compare_digits(int arr[][8], int n, int m, int var_num) {
        int result = 0;
        for (int i = 0; i < var_num; i++) {
                if (arr[n][i] != arr[m][i]) result++;
        }
        if (result == 1) return true;  // if n, m have only one digit in common return
true
        else return false;
}

void initialize_table(int arr[][8]) {
        for (int i = 0; i < 32; i++) {
                for (int j = 0; j < 8; j++) arr[i][j] = 0;
        }
}

void print_QM_op(int var_num) {
        for (int i = 0; i < var_num; i++) printf("%c ", 65 + i);
        printf("g u f");  // group, used, f
        printf("\n-------------------\n");
}


string binary_to_literal(vector <int> binary, int var_num) {
        char l_tmp;  // literal
        char c_tmp;  // asterisk
        string str_tmp;  //  output variable
        for (int i = 0; i < var_num; i++) {
                if (binary[i] == 1) {
                        l_tmp = char(i + 65);  // change it to literal
                        str_tmp += l_tmp;
                }
                else if (binary[i] == 0) {
```

```
                        l_tmp = char(i + 65);  // change it to literal
                        str_tmp += l_tmp;
                        str_tmp.push_back(char(39));  // add asterisk (using ascii
code)
                }
        }
        return str_tmp;
}
```

Ⅶ. Screenshots of the result

```
----QM operation----
A B C D g u f
--------------------
0 0 1 0 1 2 1 0 ||| m( 2 )
0 1 0 0 1 3 1 0 ||| m( 4 )
0 1 0 1 2 2 1 0 ||| m( 5 )
0 1 1 0 2 2 1 0 ||| m( 6 )
1 0 0 1 2 2 1 0 ||| m( 9 )
1 0 1 0 2 2 1 0 ||| m( 10 )
1 0 1 1 3 3 1 0 ||| m( 11 )
1 1 0 0 2 2 1 0 ||| m( 12 )
1 1 0 1 3 4 1 0 ||| m( 13 )
1 1 1 1 4 2 1 0 ||| m( 15 )

----QM operation----
A B C D g u f
--------------------
0 -1 1 0 1 0 1 0 ||| m( 2 6 )
-1 0 1 0 1 0 1 0 ||| m( 2 10 )
0 1 0 -1 1 1 1 0 ||| m( 4 5 )
0 1 -1 0 1 0 1 0 ||| m( 4 6 )
-1 1 0 0 1 1 1 0 ||| m( 4 12 )
-1 1 0 1 2 1 1 0 ||| m( 5 13 )
1 0 -1 1 2 1 1 0 ||| m( 9 11 )
1 -1 0 1 2 1 1 0 ||| m( 9 13 )
1 0 1 -1 2 0 1 0 ||| m( 10 11 )
1 1 0 -1 2 1 1 0 ||| m( 12 13 )
1 -1 1 1 3 1 1 0 ||| m( 11 15 )
1 1 -1 1 3 1 1 0 ||| m( 13 15 )

----QM operation----
A B C D g u f
--------------------
-1 1 0 -1 1 0 1 0 ||| m( 4 5 12 13 )
1 -1 -1 1 2 0 1 0 ||| m( 9 11 13 15 )
----List of Prime Implicants----
m( 2 6 )
m( 2 10 )
m( 4 6 )
m( 10 11 )
m( 4 5 12 13 )
m( 9 11 13 15 )

----Prime Implicants Chart----
2 4 5 6 9 10 11 12 13 15
------------------------------
2 0 0 2 0 0 0 0 0 0
2 0 0 0 0 2 0 0 0 0
0 2 0 2 0 0 0 0 0 0
0 0 0 0 0 2 2 0 0 0
0 2 1 0 0 0 0 1 2 0
0 0 0 0 1 0 2 0 2 1

----Essential Prime Implicants----
BC'
AD
```

Figure 1. (1)

```
----QM operation----
A B C D g u f
--------------------
0 0 0 0 0 1 1 0 ||| m( 0 )
0 0 1 1 2 2 1 0 ||| m( 3 )
0 1 0 0 1 2 1 0 ||| m( 4 )
0 1 0 1 2 3 1 0 ||| m( 5 )
0 1 1 1 3 2 1 0 ||| m( 7 )
1 0 0 1 2 2 1 0 ||| m( 9 )
1 0 1 1 3 2 1 0 ||| m( 11 )
1 1 0 1 3 2 1 0 ||| m( 13 )

----QM operation----
A B C D g u f
--------------------
0 -1 0 0 0 0 1 0 ||| m( 0 4 )
0 1 0 -1 1 0 1 0 ||| m( 4 5 )
0 -1 1 1 2 0 1 0 ||| m( 3 7 )
-1 0 1 1 2 0 1 0 ||| m( 3 11 )
0 1 -1 1 2 0 1 0 ||| m( 5 7 )
-1 1 0 1 2 0 1 0 ||| m( 5 13 )
1 0 -1 1 2 0 1 0 ||| m( 9 11 )
1 -1 0 1 2 0 1 0 ||| m( 9 13 )

----List of Prime Implicants----
m( 0 4 )
m( 4 5 )
m( 3 7 )
m( 3 11 )
m( 5 7 )
m( 3 7 )
m( 3 11 )
m( 5 7 )
m( 5 13 )
m( 9 11 )
m( 9 13 )

----Prime Implicants Chart----
0 3 4 5 7 9 11 13
------------------------------
1 0 2 0 0 0 0 0
0 0 2 3 0 0 0 0
0 2 0 0 2 0 0 0
0 2 0 0 0 0 2 0
0 0 0 3 2 0 0 0
0 0 0 3 0 0 0 2
0 0 0 0 0 2 2 0
0 0 0 0 0 2 0 2

----Essential Prime Implicants----
A'C'D'
```

Figure 2. (2)

```
----QM operation----
A B C D g u f
--------------------
0 0 0 0 0 3 1 0 ||| m( 0 )
0 0 0 1 1 3 1 0 ||| m( 1 )
0 0 1 0 1 3 1 0 ||| m( 2 )
0 0 1 1 2 3 1 0 ||| m( 3 )
0 1 0 1 2 3 1 0 ||| m( 5 )
0 1 1 1 3 3 1 0 ||| m( 7 )
1 0 0 0 1 3 1 0 ||| m( 8 )
1 0 1 0 2 2 1 0 ||| m( 10 )
1 1 0 0 2 2 1 0 ||| m( 12 )
1 1 0 1 3 3 1 0 ||| m( 13 )
1 1 1 1 4 2 1 0 ||| m( 15 )

----QM operation----
A B C D g u f
--------------------
0 0 0 -1 0 1 1 0 ||| m( 0 1 )
0 0 -1 0 0 2 1 0 ||| m( 0 2 )
-1 0 0 0 0 1 1 0 ||| m( 0 8 )
0 0 -1 1 1 2 1 0 ||| m( 1 3 )
0 -1 0 1 1 1 1 0 ||| m( 1 5 )
0 0 1 -1 1 1 1 0 ||| m( 2 3 )
-1 0 1 0 1 1 1 0 ||| m( 2 10 )
1 0 -1 0 1 1 1 0 ||| m( 8 10 )
1 -1 0 0 1 0 1 0 ||| m( 8 12 )
0 -1 1 1 2 1 1 0 ||| m( 3 7 )
0 1 -1 1 2 2 1 0 ||| m( 5 7 )
-1 1 0 1 2 1 1 0 ||| m( 5 13 )
1 1 0 -1 2 0 1 0 ||| m( 12 13 )
-1 1 1 1 3 1 1 0 ||| m( 7 15 )
1 1 -1 1 3 1 1 0 ||| m( 13 15 )

----QM operation----
A B C D g u f
--------------------
0 0 -1 -1 0 0 1 0 ||| m( 0 1 2 3 )
-1 0 -1 0 0 0 1 0 ||| m( 0 2 8 10 )
0 -1 -1 1 1 0 1 0 ||| m( 1 3 5 7 )
-1 1 -1 1 2 0 1 0 ||| m( 5 7 13 15 )

----List of Prime Implicants----
m( 8 12 )
m( 12 13 )
m( 0 1 2 3 )
m( 0 2 8 10 )
m( 1 3 5 7 )
m( 5 7 13 15 )

----Prime Implicants Chart----
0 1 2 3 5 7 8 10 12 13 15
-----------------------------
0 0 0 0 0 0 2 0 2 0 0
0 0 0 0 0 0 0 0 2 2 0
2 2 2 2 0 0 0 0 0 0 0
2 0 2 0 0 0 2 1 0 0 0
0 2 0 2 2 2 0 0 0 0 0
0 0 0 0 2 2 0 0 0 2 1

----Essential Prime Implicants----
B'D'
BD
```

Figure 3. (3)

```
----QM operation----
A B C D E g u f
--------------------
0 0 0 0 0 0 4 1 ||| m( 0 )
0 0 0 0 1 1 2 1 ||| m( 1 )
0 0 0 1 0 1 2 1 ||| m( 2 )
0 0 1 0 0 1 2 1 ||| m( 4 )
0 1 0 0 0 1 1 1 ||| m( 8 )
0 1 0 1 1 3 2 1 ||| m( 11 )
0 1 1 0 1 3 1 1 ||| m( 13 )
0 1 1 1 0 3 2 1 ||| m( 14 )
0 1 1 1 1 4 4 1 ||| m( 15 )
1 0 0 0 1 2 2 1 ||| m( 17 )
1 0 0 1 0 2 2 1 ||| m( 18 )
1 0 1 0 0 2 2 1 ||| m( 20 )
1 0 1 0 1 3 2 1 ||| m( 21 )
1 1 0 1 0 3 3 1 ||| m( 26 )
1 1 0 1 1 4 3 1 ||| m( 27 )
1 1 1 1 0 4 3 1 ||| m( 30 )
1 1 1 1 1 5 3 1 ||| m( 31 )

----QM operation----
A B C D E g u f
--------------------
0 0 0 0 -1 0 0 1 ||| m( 0 1 )
0 0 0 -1 0 0 0 1 ||| m( 0 2 )
0 0 -1 0 0 0 0 1 ||| m( 0 4 )
0 -1 0 0 0 0 0 1 ||| m( 0 8 )
-1 0 0 0 1 1 0 1 ||| m( 1 17 )
-1 0 0 1 0 1 0 1 ||| m( 2 18 )
0 1 1 -1 1 3 0 1 ||| m( 13 15 )
0 1 1 1 -1 3 1 1 ||| m( 14 15 )
-1 1 1 1 0 3 1 1 ||| m( 14 30 )
1 1 0 1 -1 3 1 1 ||| m( 26 27 )
1 1 -1 1 0 3 1 1 ||| m( 26 30 )
-1 1 1 1 1 4 2 1 ||| m( 15 31 )
1 1 -1 1 1 4 2 1 ||| m( 27 31 )
1 1 1 1 -1 4 2 1 ||| m( 30 31 )

----QM operation----
A B C D E g u f
--------------------
-1 1 -1 1 1 3 0 1 ||| m( 11 15 27 31 )
-1 1 1 1 -1 3 0 1 ||| m( 14 15 30 31 )
1 1 -1 1 -1 3 0 1 ||| m( 26 27 30 31 )

----List of Prime Implicants----
m( 0 1 )
m( 0 2 )
m( 0 4 )
m( 0 8 )
m( 1 17 )
m( 2 18 )
m( 4 20 )
m( 17 21 )
m( 18 26 )
m( 20 21 )
m( 13 15 )
m( 11 15 27 31 )
m( 14 15 30 31 )
m( 11 15 27 31 )
m( 14 15 30 31 )
m( 26 27 30 31 )

----Prime Implicants Chart----
0 1 2 4 8 11 13 14 15 17 18 20 21 26 27 30 31
-----------------------------------
4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0
4 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0
0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0 0
0 0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0
0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0
0 0 0 0 0 0 1 0 3 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 3 0 0 0 0 0 2 0 3
0 0 0 0 0 0 0 1 3 0 0 0 0 0 0 2 3
0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 3

----Essential Prime Implicants----
A'C'D'E'
A'BCE
BDE
BCD
```

Figure 4. (4)

```
----QM operation----
A B C D E g u f
--------------------
0 0 0 0 0 0 4 1 ||| m( 0 )
0 0 0 0 1 1 3 1 ||| m( 1 )
0 0 0 1 0 1 3 1 ||| m( 2 )
0 0 0 1 1 2 4 1 ||| m( 3 )
0 0 1 0 0 1 1 1 ||| m( 4 )
0 1 0 0 0 1 3 1 ||| m( 8 )
0 1 0 0 1 2 3 1 ||| m( 9 )
0 1 0 1 0 2 3 1 ||| m( 10 )
0 1 0 1 1 3 4 1 ||| m( 11 )
1 0 0 1 1 3 3 1 ||| m( 19 )
1 0 1 0 1 3 2 1 ||| m( 21 )
1 0 1 1 0 3 2 1 ||| m( 22 )
1 0 1 1 1 4 3 1 ||| m( 23 )
1 1 0 1 1 4 2 1 ||| m( 27 )
1 1 1 0 0 3 2 1 ||| m( 28 )
1 1 1 0 1 4 2 1 ||| m( 29 )
1 1 1 1 0 4 2 1 ||| m( 30 )

----QM operation----
A B C D E g u f
--------------------
0 0 0 0 -1 0 2 1 ||| m( 0 1 )
0 0 0 -1 0 0 2 1 ||| m( 0 2 )
0 0 -1 0 0 0 0 1 ||| m( 0 4 )
0 -1 0 0 0 0 2 1 ||| m( 0 8 )
0 0 0 -1 1 1 2 1 ||| m( 1 3 )
0 -1 0 0 1 1 2 1 ||| m( 1 9 )
0 0 0 1 -1 1 2 1 ||| m( 2 3 )
0 -1 0 1 0 1 2 1 ||| m( 2 10 )
0 1 0 0 -1 1 2 1 ||| m( 8 9 )
0 1 0 -1 0 1 2 1 ||| m( 8 10 )
0 1 0 -1 0 1 2 1 ||| m( 8 10 )
0 -1 0 1 1 2 3 1 ||| m( 3 11 )
-1 0 0 1 1 2 1 1 ||| m( 3 19 )
0 1 0 -1 1 2 2 1 ||| m( 9 11 )
0 1 0 1 -1 2 2 1 ||| m( 10 11 )
-1 1 0 1 1 3 1 1 ||| m( 11 27 )
1 0 -1 1 1 3 0 1 ||| m( 19 23 )
1 -1 0 1 1 3 1 1 ||| m( 19 27 )
1 0 1 -1 1 3 0 1 ||| m( 21 23 )
1 -1 1 0 1 3 0 1 ||| m( 21 29 )
1 0 1 1 -1 3 0 1 ||| m( 22 23 )
1 -1 1 1 0 3 0 1 ||| m( 22 30 )
1 1 1 0 -1 3 0 1 ||| m( 28 29 )
1 1 1 -1 0 3 0 1 ||| m( 28 30 )

----QM operation----
A B C D E g u f
--------------------
0 0 0 -1 -1 0 1 1 ||| m( 0 1 2 3 )
0 -1 0 0 -1 0 1 1 ||| m( 0 1 8 9 )
0 -1 0 -1 0 0 1 1 ||| m( 0 2 8 10 )
0 -1 0 -1 1 1 1 1 ||| m( 1 3 9 11 )
0 -1 0 1 -1 1 1 1 ||| m( 2 3 10 11 )
0 1 0 -1 -1 1 1 1 ||| m( 8 9 10 11 )
-1 -1 0 1 1 2 0 1 ||| m( 3 11 19 27 )

----QM operation----
A B C D E g u f
--------------------
0 -1 0 -1 -1 0 0 1 ||| m( 0 1 2 3 8 9 10 11 )
m( 19 23 )
m( 21 23 )
m( 21 29 )
m( 22 23 )
m( 22 30 )
m( 28 29 )
m( 28 30 )
m( 3 11 19 27 )
m( 0 1 2 3 8 9 10 11 )

----Prime Implicants Chart----
0 1 2 3 4 8 9 10 11 19 21 22 23 27 28 29 30
--------------------------------
2 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 2 0 0 3 0 0 0 0
0 0 0 0 0 0 0 0 0 0 2 0 3 0 0 0 0
0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 2 0
0 0 0 0 0 0 0 0 0 0 0 2 3 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 2
0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2
0 0 0 2 0 0 0 0 2 2 0 0 0 1 0 0 0
2 1 1 2 0 1 1 1 2 0 0 0 0 0 0 0 0

----Essential Prime Implicants----
A'B'D'E'
C'DE
A'C'
```

Figure 5. (5)

```
----QM operation----
A B C D E g u f
--------------------
0 0 0 0 1 1 3 1 ||| m( 1 )
0 0 0 1 1 2 2 1 ||| m( 3 )
0 0 1 0 0 1 2 1 ||| m( 4 )
0 0 1 0 1 2 3 1 ||| m( 5 )
0 1 0 0 1 2 3 1 ||| m( 9 )
0 1 0 1 1 3 2 1 ||| m( 11 )
0 1 1 0 0 2 2 1 ||| m( 12 )
0 1 1 0 1 3 3 1 ||| m( 13 )
1 0 0 1 0 2 2 1 ||| m( 18 )
1 0 1 1 0 3 3 1 ||| m( 22 )
1 0 1 1 1 4 2 1 ||| m( 23 )
1 1 0 1 0 3 2 1 ||| m( 26 )
1 1 1 1 0 4 3 1 ||| m( 30 )
1 1 1 1 1 5 2 1 ||| m( 31 )

----QM operation----
A B C D E g u f
--------------------
0 0 0 -1 1 1 1 1 ||| m( 1 3 )
0 0 -1 0 1 1 1 1 ||| m( 1 5 )
0 -1 0 0 1 1 2 1 ||| m( 1 9 )
0 0 1 0 -1 1 1 1 ||| m( 4 5 )
0 -1 1 0 0 1 1 1 ||| m( 4 12 )
0 -1 0 1 1 2 1 1 ||| m( 3 11 )
0 -1 1 0 1 2 2 1 ||| m( 5 13 )
0 1 0 -1 1 2 1 1 ||| m( 9 11 )
0 1 -1 0 1 2 1 1 ||| m( 9 13 )
0 1 1 0 -1 2 1 1 ||| m( 12 13 )
1 0 -1 1 0 2 1 1 ||| m( 18 22 )
1 -1 0 1 0 2 1 1 ||| m( 18 26 )
1 0 1 1 -1 3 1 1 ||| m( 22 23 )
1 -1 1 1 0 3 2 1 ||| m( 22 30 )
1 1 -1 1 0 3 1 1 ||| m( 26 30 )
1 -1 1 1 1 4 1 1 ||| m( 23 31 )
1 1 1 1 -1 4 1 1 ||| m( 30 31 )

----QM operation----
A B C D E g u f
--------------------
0 -1 0 -1 1 1 0 1 ||| m( 1 3 9 11 )
0 -1 -1 0 1 1 0 1 ||| m( 1 5 9 13 )
0 -1 1 0 -1 1 0 1 ||| m( 4 5 12 13 )
1 -1 -1 1 0 2 0 1 ||| m( 18 22 26 30 )
1 -1 1 1 -1 3 0 1 ||| m( 22 23 30 31 )

----List of Prime Implicants----
m( 1 3 9 11 )
m( 1 5 9 13 )
m( 4 5 12 13 )
m( 18 22 26 30 )
m( 22 23 30 31 )
m( 4 5 12 13 )
m( 18 22 26 30 )
m( 22 23 30 31 )

----Prime Implicants Chart----
1 3 4 5 9 11 12 13 18 22 23 26 30 31
--------------------------------
2 1 0 0 2 1 0 0 0 0 0 0 0 0
2 0 0 2 2 0 0 2 0 0 0 0 0 0
0 0 1 2 0 0 1 2 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 2 0 1 2 0
0 0 0 0 0 0 0 0 0 2 1 0 2 1

----Essential Prime Implicants----
A'C'E
A'CD'
ADE'
ACD
```

Figure 6. (6)