

A Novel Method for Minimization of Boolean Functions using Gray Code and development of a Parallel Algorithm

Shrish Verma and K. D. Permar

Government Engineering College, Raipur 492 010 CG, India

email: yshrish9@yahoo.com and kdpermar@yahoo.com

Abstract

The present paper proposes a new method for two level minimization of Boolean functions, which is not only a versatile and elegant paper-and-pencil method but also the one for which a parallel minimization algorithm can be readily developed. A novel minterm numbering scheme based on Gray code is proposed exploiting the Gray code's unit distance and reflection properties for obtaining adjacency among the given minterms in paper-and-pencil method as well as to develop a parallel algorithm for generating prime implicants and subsequent minimization of the given Boolean function. The proposed paper-and-pencil method and the parallel algorithm are implemented on a few examples having don't care terms.

Presentation Lay-out

- ▶ Introduction
- ▶ Properties of Gray code & Minterm numbering
- ▶ Development of the proposed method
- ▶ The Adjacency Rules
- ▶ The Rules of Minimization of the Boolean Function
- ▶ An Example
- ▶ Development of the Parallel Minimization Algorithm
- ▶ An Illustration
- ▶ Conclusion

Introduction

Minimization of Boolean function is an important step in digital design.

Present methods

- Karnaugh Map Method (1953)
- Quine-McCluskey's tabular Method (1956)

CAD methods available for two level minimization

- ESPRESSO (1984-)
- BOOM (2001)

Need of an effective paper and pencil minimization method

- Break down of a CAD Tool
- To check correctness of a CAD tool

Properties of Gray Code

Following two properties of the Gray Code are exploited in the proposed method

1. UNIT DISTANCE PROPERTY

☞ Two consecutive code terms differ in only one bit.

2. REFLECTION PROPERTY

☞ In a reflective Gray code of 'n' bits, there will be a total of $2^n = m$ (say) terms.

☞ Out of these m terms, most significant bit (MSB) of lower m/2 terms will be complement of MSB of upper m/2 terms and remaining (n-1) bits of lower m/2 terms will be the mirror image of (n-1) bits of upper m/2 terms.

☞ This makes each term in the upper half at unit distance from its reflected term in the lower half. Further, a plane (mirror) can be assumed to exist between the two halves around which the reflection is taking place. This plane is henceforth referred to as a reflection plane.

☞ This property of reflection is as well true for remaining (n-1) bits and subsets thereof such that a term in the first quarter of m terms is at unit distance from its reflected term in the second quarter and so on.

Properties of Gray Code and Minterm Numbering

Use of Gray Code for numbering the minterms

Variables (arranged in Gray code)			Product term	Minterms m_j^g in the proposed scheme	Minterms ' m_j ' in 8-4-2-1 code
A	B	C			
0	0	0	$\bar{A}\bar{B}\bar{C}$	m_0^g	m_0
0	0	1	$\bar{A}\bar{B}C$	m_1^g	m_1
0	1	1	$\bar{A}BC$	m_2^g	m_3
0	1	0	$\bar{A}B\bar{C}$	m_3^g	m_2
1	1	0	$AB\bar{C}$	m_4^g	m_6
1	1	1	ABC	m_5^g	m_7
1	0	1	$A\bar{B}C$	m_6^g	m_5
1	0	0	$A\bar{B}\bar{C}$	m_7^g	m_4

Reflection planes

☞ Gray Code is used to number the minterms

☞ Properties of Gray code, viz.

▶ Unit Distance Property

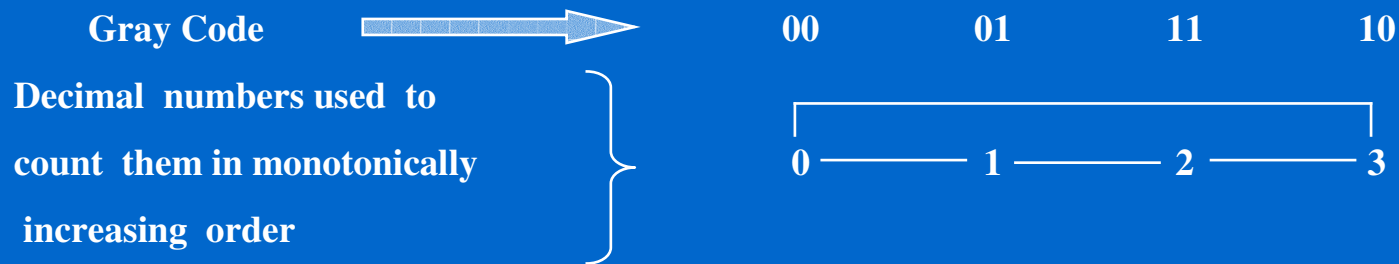
▶ Reflection Property

are utilized to find adjacency among such minterms

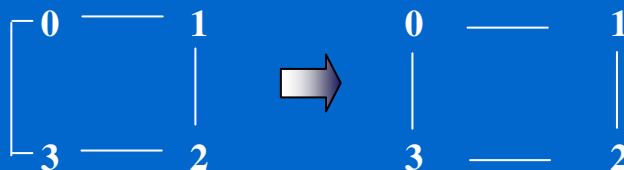
Development of the Proposed method

- ❁ Adjacency between Consecutive minterms is depicted by hyphen (—)
- ❁ Adjacency due to reflection is shown by $\overline{\quad}$ and $\overline{\quad}$
- ❁ More Flexible way of writing minterms and depicting adjacencies
- ❁ Minterms are depicted without drawing cells or squares

Adjacency among four minterms of a 2-variable Boolean function using the above two notations can now be depicted as





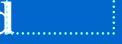
Rotating the same minterms at a reflection plane we get the alternate depiction of the adjacency as



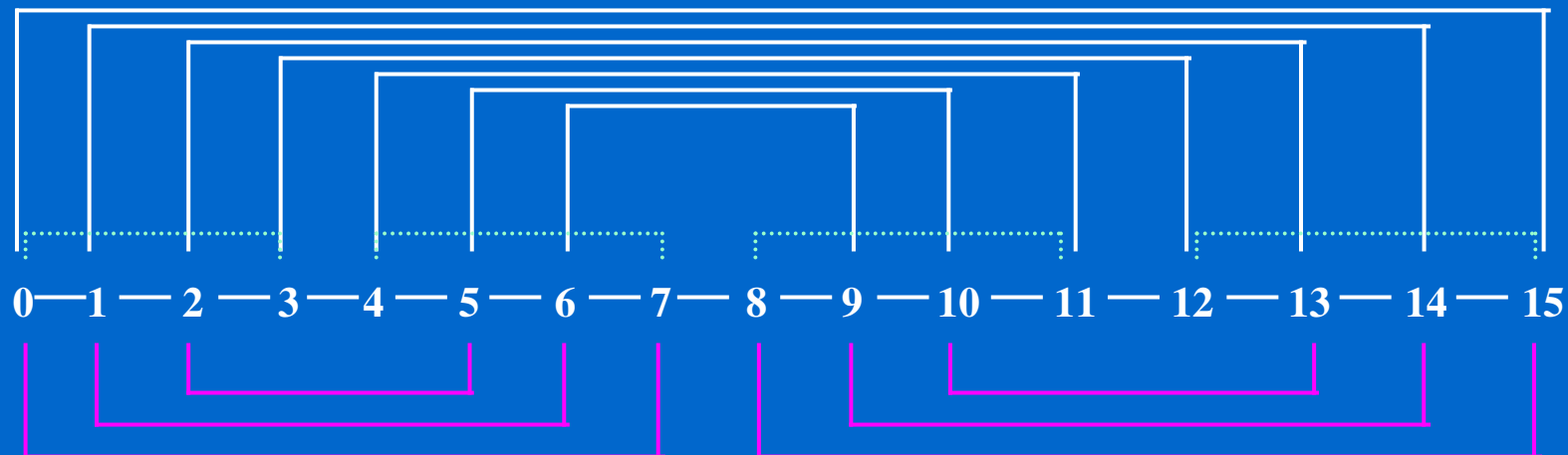
Alternate Depictions of adjacency among four minterms of a 2-variable Boolean function

Development of the Proposed method

For a Boolean function of four variables

- ⊗ Every minterm will have four adjacent minterms
- ⊗ A hyphen (—) in between the two consecutive minterms shows the adjacency due to “**Unit Distance Property**” of the Gray code.
- ⊗ Lines like ,  and  show the adjacent minterms of a given minterm due to “**Reflection Property**” of the Gray code

Adjacencies amongst the minterms of a four variable function can now be shown as



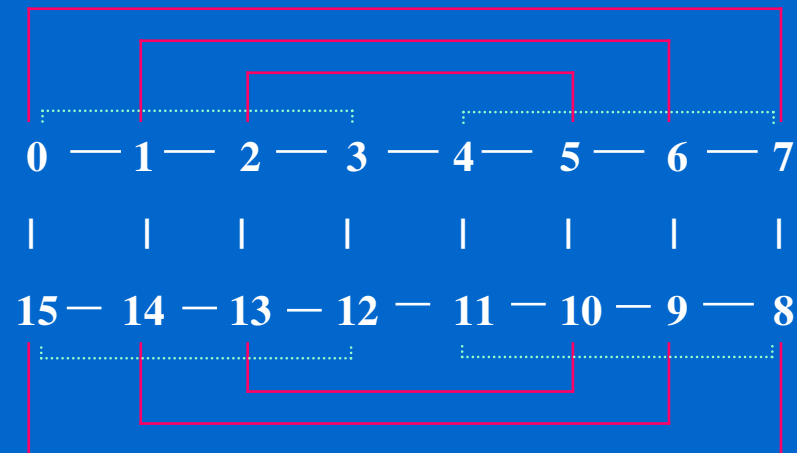
Adjacencies amongst the minterms of a four variable function when all the minterms are written along one horizontal line

Development of the Proposed method

The proposed method offers more flexibility to visualize the adjacencies among the minterms of a Boolean function by writing minterms along many horizontal lines by rotating the minterms after a plane of reflection.


The minterms of a four variable function can be written in two horizontal lines by rotating the minterms around a plane of reflection occurring after the minterm number 7.

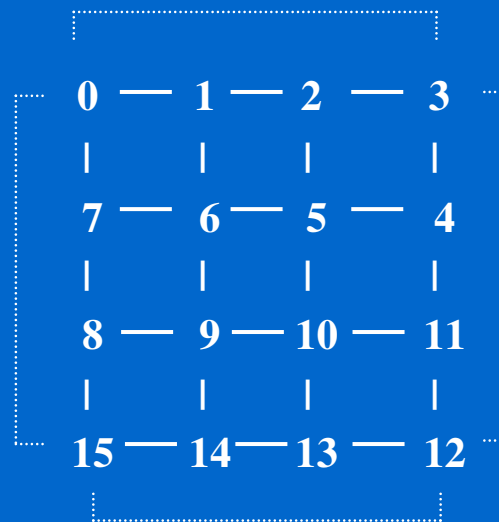
The adjacency of a minterm thus falling below another minterm after rotation is carried out is shown by a small vertical line (|).



Depiction of adjacencies amongst the minterms of a four variable function when rotated after minterm number 7

Development of the Proposed method

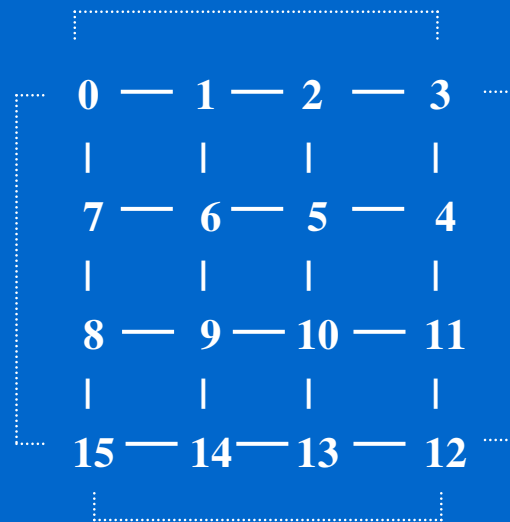
 The minterms of a four variable function can be written in four horizontal lines by further rotating the minterms around planes of reflection occurring after the minterm number 3 and 11.



Depiction of adjacencies amongst the minterms of a four variable function in four horizontal lines when rotation is carried out after minterm number 7, 3 and 11.

Development of the Proposed method

The Karnaugh Map is a special case of the proposed method



CD	00	01	11	10
AB				
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Figure-1 The proposed method

Figure-2 The Karnaugh map

☞ If the minterms of the proposed method are converted into their equivalent 8-4-2-1 weighted code and put in the square cells then the depiction turns to be a Karnaugh map of four variable case as shown in the Figure-1 and the Figure-2

The Adjacency Rules

- The consecutive minterms along a horizontal line are adjacent to each other and this adjacency is depicted by “—”.
- A minterm is adjacent to a term appearing above and below it, if the minterms are written along more than one line after rotating the minterms around a reflection plane. This adjacency is shown by a vertical line “|”.
- A minterm is adjacent to all its reflected minterms, where the reflection takes place after every $2^1, 2^2, 2^3, \dots, 2^i$ terms. These adjacencies are shown by —| and —|— lines.

Rules For Minimization of the Boolean Expression

Step 1: Number the given minterms in increasing order in their Gray code and write the function as sum of these minterms.

Step 2: Arrange all possible minterms along a line or along many horizontal lines (where number of horizontal lines $l=2^i$) and encircle those minterms which are appearing in the given logic function.

Step 3: Draw all possible adjacencies in horizontal and vertical directions by applying the adjacency rules mentioned earlier.

Step 4: If from an encircled minterm no adjacency line is leading to another encircled minterm then that minterm will appear as it is in the minimized expression and is an essential prime implicant, “e”.

Rules For Minimization of the Boolean FunctionContd.

Step 5: Starting from the first encircled minterm move along the adjacency lines to all possible adjacent minterms and form the largest group of traversed minterms such that from the last traversed minterm it should always be possible to come back following a single adjacency line to the minterm from where the current traversing started. In this manner form octets, quads, pairs as the largest possible groups of minterm. Write the minterms of a group separating them by “—” (hyphen), naming them p, q, o, h for pairs, quads, octets and hexatets, respectively.

Step 6: From the result obtained in Step 5 above, the minimized terms for octets, quads, pairs and individual essential prime implicant are written by successively combining the two adjacent terms by putting “x” where they differ by one literal, as it is done in Quine-McCluskey’s method and the corresponding final minimized Boolean function is written.

EXAMPLE 1

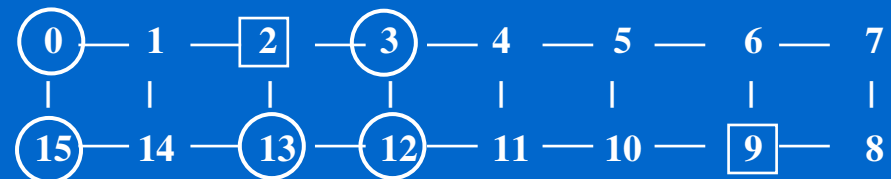
Minimize the Boolean function

$$f(A, B, C, D) = \sum m(0, 2, 8, 10, 11) + \sum d(3, 13)$$

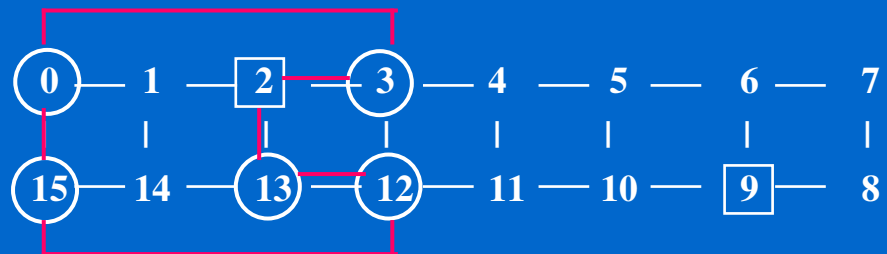
Step 1: Converting to Gray code, we get

$$f(A, B, C, D) = \sum m_g(0, 3, 12, 13, 15) + \sum d_g(2, 9)$$

Step 2: Writing all possible minterms in two horizontal lines and encircling the minterms and putting a square around don't care terms appearing in the given Boolean function, we get



Step 3: Marking all possible adjacencies from an encircled minterm to another encircled minterm or to a don't care term, we get



EXAMPLE 1

contd..

Step 4: No essential prime implicant is present.

Step 5: For carrying out the minimization, now, we start from minterm 0 and traversing through minterms 3, 12, 15 and back to 0, we find that a quad $Q_1 = 0-3-12-15$ is obtained. Similarly, traversing through minterms 3, 12, 13 we find that by including don't care term 2 another quad $Q_2 = 3-12-13-2$ can be obtained. The don't care term 9 is not included in minimization as it does not help in making larger group.

Step 6: Carrying out the successive combination of the terms changing in one literal only as in Quine-McCluskey's method, we get,

$$Q_1 = 0-3-12-15 = 0000-0010-1010-1000$$

$$= \underbrace{00x0} - \underbrace{10x0} = x0x0 = \overline{\overline{B}}\overline{\overline{D}}$$

$$Q_2 = 12-13-2-3 = 1010-1011-0011-0010$$

$$= \underbrace{101x} - \underbrace{001x} = x01x = \overline{\overline{B}}C$$

The minimized expression, therefore, is

$$f_{\min} = \overline{\overline{B}}\overline{\overline{D}} + \overline{\overline{B}}C$$

Algorithm for Distribution of minterms on nodes

M_D_A (Given GM[b]) // Minterm Distribution Algo.

// A ring /Torus of dimension = N nodes

// Number of Boolean variables = v

// Total number of minterms $n = 2^v$

//Number of terms to be distributed to

//each node $k = 2^v / N$ ($v > N$)

begin

DM[N-1,K-1] // of Gray Codes:

a=0;b=0 // (Gray variables)

for(j=0toK-1) do

begin

if ((j==0)OR(jmod2==0)) then

begin

for (I=0to N-1) do

begin

if (a== GM[b]) then

begin

DM[I,j]=GM[b];

b=b+1;

end

else

begin

DM[I,j]=-1;

a=a+1;

end // if

end //for

else

begin

for(I=N-1 to 0) do

begin

if (a = GM[b]) then

begin

DM[I,j]=GM[b];

b=b+1;

end

else

begin

DM[I,j]=-1;

a=a+1;

end;

end // if

end // for

end // for

end // end of M_D_A

Parallel Minimization Algorithm

- Step 1:** Distribute the given minterms on 'N' nodes of the chosen topology, i.e., a ring, torus or K-ary n-cube.
- Step 2:** If at each node the number of minterms distributed is more than one then apply the Minterm Distribution Algorithm recursively till all the prime implicants are generated (following the adjacency rules given in section 4) from each node.
- Step 3:** Form the largest possible groups at each node by making the larger groups from smaller groups (prime implicants) if they are existing at the same position.
- Step 4:** Form the bigger groups by combining largest groups obtained at each node in step 3 following the adjacency rules applicable at the chosen topology of step 1.
- Step 5:** Solve covering problem to optimize the number of groups obtained in step 4 to finally occur in the minimized Boolean expression.

Example (2)

Minimize the following Boolean function, assuming that a **Ring topology** of processors with **four number** of nodes is available

$$f(A, B, C, D) = \sum m_g(5, 6, 7, 8, 9, 15)$$

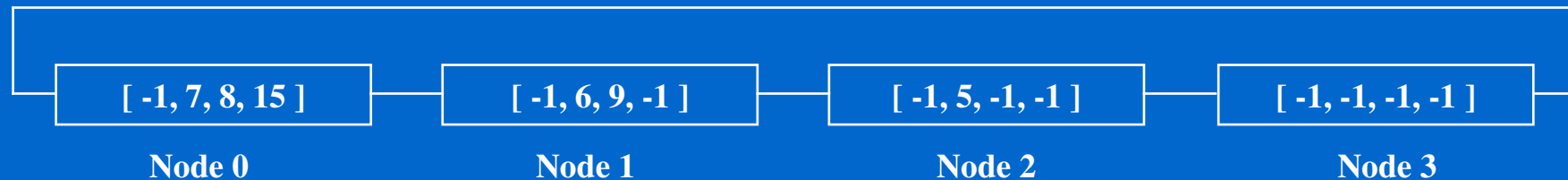
Step 1: Distributing the minterms at each node of the given ring as shown in the Figure(3), we get,

At node '0': $DM[0] = [-1, 7, 8, 15]$

At node '1': $DM[1] = [-1, 6, 9, -1]$

At node '2': $DM[2] = [-1, 5, -1, -1]$

At node '3': $DM[3] = [-1, -1, -1, -1]$



Figure(3) Minterms of the given Boolean function distributed at each node of the ring with number of nodes = 4

Example (2) contd.

Step 2 Since the number of minterms distributed to each node is greater than one, the Minterm Distribution Algorithm is applied recursively at each node. Applying Minterm Distribution Algorithm at node '0', we find that the resulting four nodes of the ring constructed at node '0' contain one minterm each. Step 2 is carried out in parallel at each node to get the same result. The prime implicants generated at each node following

the adjacency rule of section 4 are

At node '0' : pairs (7,8) and (8,15)

At node '1' : pair (6,9)

At node '2' : essential prime implicant 5, and

At node '3' : No prime implicant.

Step 3 It is not required in this example.

Step 4 Following the adjacency rule for prime implicants existing at the same position, we get the quad (7,8,6,9) from node '0' and '1', the pair (5,6) from node '1' and '2'.

Step 5 Optimizing the number of groups to occur in the final expression, we get the quad (7,8,6,9), the pairs (8, 15) and (5, 6) which yield the minimum expression as

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

Conclusion

& A simpler and much more versatile method of minimization of Boolean functions has been presented here.

& A Gray code based minterm numbering scheme is developed. Various adjacency rules, grouping rules and the minimization steps were formulated and implemented upon an example for the paper-and-pencil method.

& The proposed method has been shown to be equally suitable for parallel implementation.

& An algorithm for its parallel implementation has been developed for various topologies of multiprocessor and the same has been demonstrated on another example.

& It has been shown that the proposed method is so versatile that the Karnaugh map can be taken as its special case.

References

- [1] R. L. Rudell, and A. Sangiovanni-Vincentelli: Multiple-valued minimization for PLA optimization, in: 1987, IEEE Trans. CAD, Vol.6(5), pp. 725-750.
- [2] J. Hlavicka and P. Fisr: BOOM- A heuristic Boolean minimizer, Proceedings of the International Conference on Computer Aided Design ICCAD, 2001, San Jose, California(USA), pp. 439-442.
- [3] P. McGeer et al.: ESPRESSO-SIGNATURE: A new exact minimizer for logic functions, Proceeding of Design automation Conference, 1993.
- [4] M. Karnaugh: The map method of synthesis of combinational logic circuits, Transactions of AIEE, 1953, Vol.72(I), pp. 593-598.
- [5] E. J. McCluskey: Minimization of Boolean functions, Bell Systems Technical Journal, 1956, Vol.35, pp. 1417-1444.
- [6] A. E. Barbour: A Hypercube minimization algorithm for Boolean functions, Proceedings of Int. Conf. On Parallel and Distributed Processing Techniques and Applications PDPTA 1997, Las Vegas, Nevada(USA), pp. 811-815.
- [7] E. M. Reingold, J. Nievergelt, and N. Deo: Combinatorial Algorithms, Prentice Hall, Englewood Cliffs, NJ, 1977.
- [8] Y. Saad, and M. Shultz: Topological properties of Hypercubes, IEEE Transactions on Computers, 1988, Vol.37(7), pp. 867-872.
- [9] Kai Hwang: Advanced Computer Architecture, McGraw-Hill International, New York, 1993.

-
-
-

A Novel Method for Minimization of Boolean Functions using Gray Code and development of a Parallel Algorithm

Thank you

6th International Workshop on Boolean Problems,

September 23-24, 2004

Freiberg