Performance Estimation of Karnaugh Map through UML

Dr. Vipin Saxena[†], Manish Shrivastava[††] and Dr. Deepak Arora[†††]

Department of Computer Science, B.B. Ambedkar University (A Central University), Vidya Vihar Rae Bareilly Road, Lucknow, U.P. 226025, India

Summary

The digital circuits are mostly constructed through digital gates. The minimization of the number of digital gates is an important activity in designing the digital circuits. This minimization reduces the size and cost of these systems and the performance can be improved. There are some well established methods for doing these simplifications. One of the famous methods is known as Karnaugh map (K-map) method. The digital circuits can be represented and analyzed using the boolean functions. K-map is in fact a visual diagram of representing all possible ways a boolean function may be expressed.

In the present wok, a well known modeling language, the Unified Modeling Language (UML) is used for designing an Object-oriented model for Karnaugh map with the help of digital gates. An Object-oriented algorithm is also proposed for simplification of boolean functions through K-map. The UML stereotypes and class diagrams are presented and performance of UML model is analyzed through a case study.

Key words:
Boolean functions, Digital Circuit, Karnaugh map, Minterm, UML Class diagram, Object-oriented Model.

1. Related Work

The digital gates (Logic gates) are basic electronic components of any digital circuit. A logic gate performs a logical operation based on one or more inputs and produces a single output voltage value (i.e. voltage levels high and low). Logically these voltage values can be referred to as 1s and 0s and are used in designing and analyzing the operations of logic gates. A logic gate represents a boolean function. A boolean function is an algebraic expression formed with boolean variables (having values true or 1 and false or 0) and the logical operators (i.e. OR, AND, and NOT). There may be a large number of boolean algebraic expressions that specify a given boolean function. It is therefore important to find the simplest one. A boolean function can be represented by a truth table. A truth table is a tabular arrangement of representing all the input-output relationships of a digital circuit. It displays all possible input values combinations with their respective output values. There is much literature available on the concepts of digital circuits design. The basic concepts of digital circuit design are available in Mano [1], Rajaraman & Radhakrishnan [2], Tanenbaum [3], Leach et al. [4], Wakerly [5], Crenshaw [6] and Kuphaldt [7].

Normally a Boolean expression can be given using two forms:

1. Sum-of-Products (SOP): This is the more common form of Boolean expressions. The expressions are implemented as AND gates (products) feeding a single OR gate (sum).
2. Product-of-Sums (POS): This is less commonly used form of Boolean expressions. The expressions are implemented as OR gates (sums) feeding into a single AND gate (product).

SOP Boolean expressions may be generated from truth tables quite easily by forming an OR of the ANDs of all input variables (standard producst or minterms) for which the output is 1. POS expressions are based on the 0s, in a truth table and generated oppositely as SOP by taking an AND of the ORs of all input variables (standard sums or maxterms).

The Unified Modeling Language (UML) was created as a result of unification of different Object-oriented design methodologies. The standards and recent developments of UML are available on [8]. The good descriptions of UML diagrams and notations are available in [9] and [10]. Originally it is defined and has been successfully applied in software systems design, but can also be applied in the design of hardware systems as well. The Object-oriented design using UML diagrams in hardware system modeling and designing have been proposed in some research papers, but there is very less work available on the applications of UML in digital logic minimization. The use of UML in real-time and embedded systems specification and design has been explored by Gomaa [11] and Schattkowsky [12]. Recently Saxena et al. [13] proposed the UML model for the Multiplex system for the processes which are executing in distributed environment. Damasevicius and Stuikys [14]

Manuscript received June 5, 2009
Manuscript revised June 20, 2009

presented a design process model for adopting Object-oriented design concepts in hardware design processes.

Al-Rababah [15] introduced a novel approach for the synthesis of reconfigurable hardware from UML models. The presented approach enables the synthesis of Object-oriented specifications into hardware circuits. Kohut et al. [16] suggested a new approach for modeling of boolean neural networks on field programmable gate arrays (FPGAs) using UML. Sun et al. [17] outlined a design flow to develop clocked hardware circuits using UML notations. The UML Class, Statechart and Component diagrams are used to model system specifications.

2. Background

2.1 Simplification of Logic Designs

The simplification or minimization of any digital circuit is an important activity in digital circuit design. To simplify the circuit, the designer tries to find another circuit that produces the same output as the original one but with less number of gates. The main objective of this process is to keep the number of digital gates as minimum as possible and thus get a minimal cost solution. There are various methods of simplification such as boolean algebra, Karnaugh maps, Tabulation method, Computer Aided Design etc. All these methods use the simplification of boolean function that represents the digital logic.

2.2 The Karnaugh map

Maurice Karnaugh developed the Karnaugh map in 1953 during designing of digital circuits for telephone switching circuits. This technique is quite easy and fast in comparison with boolean algebra. Karnaugh maps work well for up to six input variables. A Karnaugh map consists of an array of rectangles or boxes arranged in rows and columns. The size of the Karnaugh map with **n** boolean variables is equal to 2^n. The size for maps of 2 variables is a 2×2 map (four boxes), for 3 variables it is a 2×4 map, and for 4 variables it is a 4×4 map and so on. The boolean variables are arranged in an order according to the principles of gray code where only one variable changes in adjacent squares. Each square represents a minterm (sometimes a maxterm) corresponding to the truth table. A minterm is a boolean expression consisting of a product term of those variables (or their complimented form). The minterms are identified by associating numbers to them like m0, m1, ….m_n etc.

For simplifying an input expression, the adjacent minterms are identified and a group of 2 (Pair), 4 (Quad) or 8 (Octet) adjacent minterms are formed. The minterms can only form a group if they are adjacent horizontally and vertically and not diagonally. The groups should be as large as possible and overlapping of any minterm on two or more groups is allowed. Similarly the wrap around of minterms is also allowed for forming a group. If a term and its compliment both appear in a group, delete both from the resultant product term. Finally write the boolean expression of the remaining terms. For example, A K-map of three input variables can be expressed as E = A'B'C'+A'BC+AB'C (=m0+m3+m5) is represented in the figure 1 below:

BC		00	01	11	10
A	0	m0 000 A'B'C'	m1 001 A'B'C	m2 011 ABC	m3 010 A'BC'
	1	m4 100 AB'C'	m5 101 AB'C	m6 111 ABC	m7 110 ABC'

Fig. 1 An Example of a K-map for Three variables

3. UML Modeling and Minimization of K-map

3.1 UML Representation of Digital Gates

The UML provides the facility of defining profiles and stereotypes that can be used to define a relevant domain-specific model element. The UML modelling of digital gates is shown in figure 2. In this design, a stereotype "Digital_Gate" is defined.

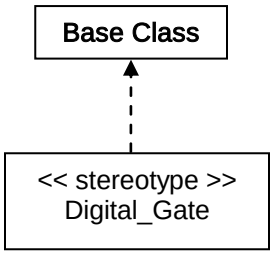

Fig. 2 Stereotype of Digital Gate

Figure 3 shows the class definition of all the digital gates. Here the class "Gate" is defined, which is derived from the stereotype "Digital_Gate". The three basic gates are defined as the subclasses of this class. The classes namely "AND", "NOT" and "OR" are inherited from the class "Gate". The other gates are defined as the classes which are the composition of these basic gates. The class "NAND" is a composition of the classes "AND' and "NOT". The class "NOR" is a composition of the classes

"OR" and "NOT" where as the class "XOR" is a composition of the classes "AND", "OR" and "NOT".

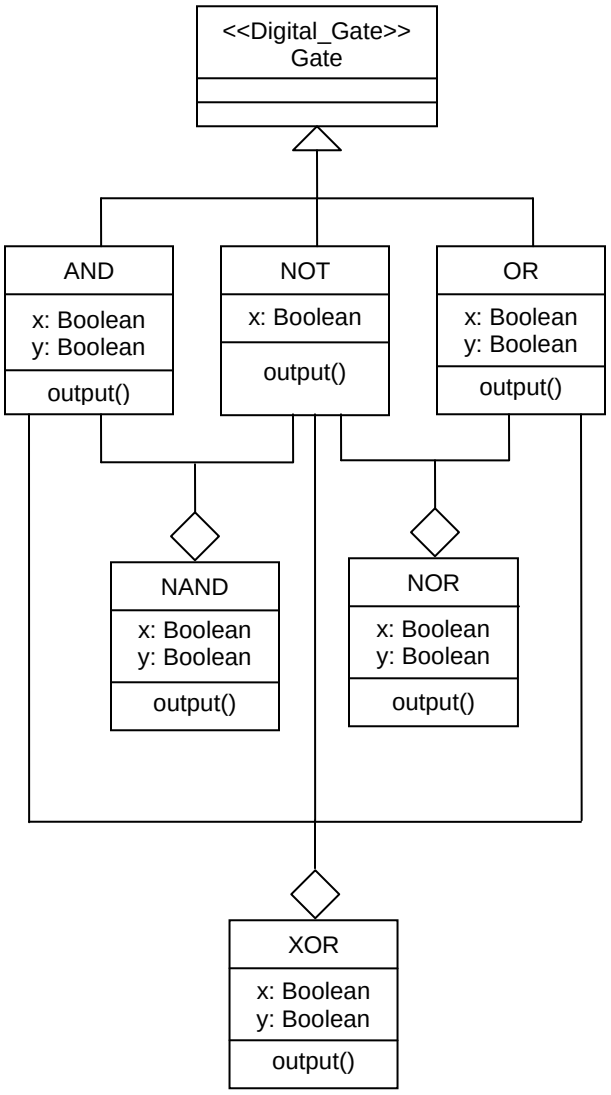

Fig. 3 UML Class Definition of Digital Gates

3.2 UML Representation and Minimization of Karnaugh Map

The Karnaugh map is also defined using the UML stereotype mechanism. The stereotype "Map" is defined and is shown in figure 4. The class definition of Karnaugh map is shown in figure 5. In this design, a class "K-map" is defined which is derived from the stereotype "Map". This class contains multiple instances (1 to 2^n) of a class "minterms" which is also defined as a stereotype. In the definition of a Karnaugh map, an attribute "n" is more significant. This attribute identifies the number of input variables. Based on the number of inputs, the minterms are generated. The generalized instance and class instances diagram of "minterms" for 3 variables are shown in figure 6(a) and 6(b) respectively. Each minterm represents a term which is a standard product of the values of input variables. According to the K-map simplification process, these minterms are arranged in an order and they form a logical adjacency to each other. The adjacency is initialized as an attribute "adjacency" vector. The input string contains only those minterms for which the output is 1. According to the minterms specified in the input string, the objects of the class "minterms" are generated and all the attributes are initialized. For example, for input expression E=x'y'z'+x'yz+xy'z'+xy'z, the minterms inputs will be equal to m0+m3+m4+m5. The corresponding digit values will be 000+011+100+101. This arrangement is shown in figure 6. The annotation mechanism of the UML is used to indicate the participating Octet, Quad or Pair. The simplification is obtained by using an algorithm described below and it is based upon the Object-oriented methodology.

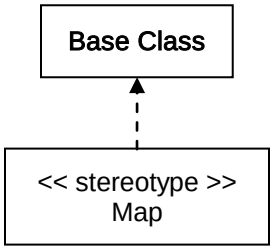

Fig. 4 Stereotype of Map

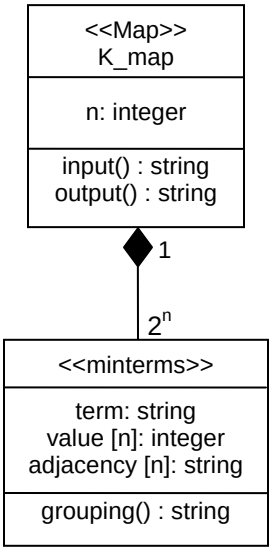

Fig. 5 UML Class Definition of Karnaugh map

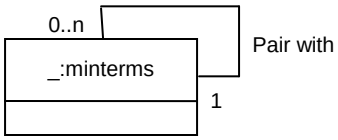

Fig. 6 (a) UML Instance Diagram for minterms

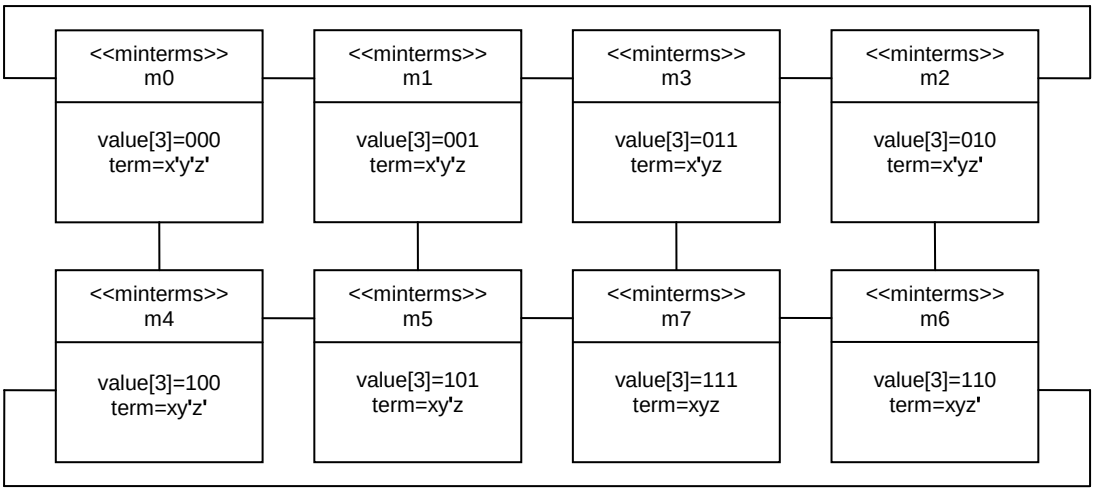

Fig. 6 (b) UML Class Instances of Minterms for n=3 Variables

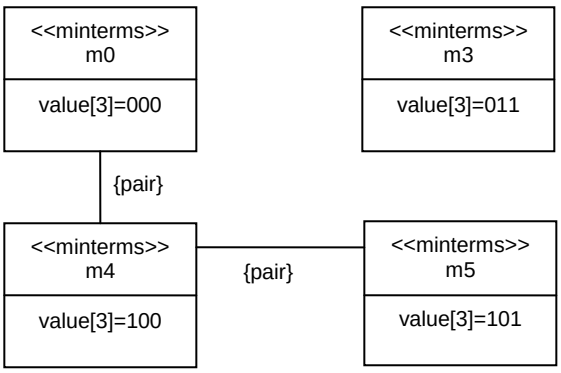

Fig. 7 UML Class Instances for the Expression E=m0+m3+m4+m5

3.3. Minimization of K-map

1. Enter the input string containing the minterms for which the output is 1 (e.g. E=m0+m3+m4+m5).
2. The maximum number of minterms = 2^n where n= number of inputs (x,y,z etc.).
3. If there is only one input minterm, then the output will be equal to that minterm only.
4. Create the objects of those minterm classes which are present in the input string and initialize the value vector correspondingly with the values 0 and 1. Also initialize the adjacency matrix for indicating the adjacency minterms to that minterm. The maximum number of adjacent minterms to any minterm will be equal to n only.
5. Check if the adjacency minterm exists or not. If there is no adjacency minterm exists then that minterm will be marked as isolated and its value will be a part of the output.
6. If adjacency minterms exist then check for the possibilities of Octet, Quad or Pair. This can be done using a method "grouping()".
7. Select first minterms which are the part of an Octet and apply minimization and get the corresponding values as a part of the output. Repeat this for Quad and then for Pair.
8. Check overlapping of any minterm with the Octet, Quad and Pair and apply minimization.
9. The final output will be the combination of all minimized values that can be mapped or converted in the form of sum of products in terms of variables x,y,z etc.

4. Results and Discussions

For a Pair: take the value vector of both the minterms. Find out the common bits positions (unchanged bits) and neglect (or mark as ☒) those which are changing from 0 to 1 or 1 to 0. Minimizing pairs will eliminate only one variable at a time. The resultant values will be converted in terms of variables x,y,z etc.

For a Quad and Octet: For a Quad and Octet, take 4 or 8 minterms and find the common bits positions among all participating minterms. The output will be the corresponding bit positions of all common bits (unchanged). In case of a Quad only two variables will be reduced at a time and in case of an Octet, three variables will be reduced at a time.

4.1 A Case Study

As a case study we consider an input string (sum of product) as E=x'y'z'+x'yz+xy'z'+xy'z this will be equal to minterms m0+m3+m4+m5
The corresponding digit values are 000+011+100+101

As indicated in the figure 7 above, the four objects are created and there adjacency relationship is formed. Here the minterms m0 and m4 forms a pair and similarly the minterms m4 and m5. The minterm m3 is not adjacent to any of these minterms so it will be treated as isolated and can be further minimized. The value for this will be x'yz.

Take first pair containing (m0, m4)
The value vector m0.value [3] = 000
The value vector m4.value [3] = 100
The most significant bits are changing from 0 to 1, the last two bits are common. By taking these common bits the value= ⊠00, which corresponds to y'z'

Take first pair containing (m4, m5)
The value vector m0.value [3] = 100
The value vector m4.value [3] = 101
The least significant bits are changing from 0 to 1, the first two bits are common. By taking these common bits the value= 10⊠, which corresponds to xy'

The simplified expression will be = x'yz+y'z'+xy'

This reduces the number of AND, OR and NOT gates and the number objects required to instantiate for implementing this digital circuit according to the object oriented design.

The equivalent Karnaugh map can be drawn as follows:

m0 000	m1 001	m3 011	m2 010
1	0	**1**	0
1	**1**	0	0
m4 100	m5 101	m7 111	m6 110

5. Concluding Remarks

The minimization of digital gates in any digital circuit design is an important aspect for performance improvement. In the present paper, an Object-oriented design procedure of the well known Karnaugh map minimization is presented. The UML modeling is done for the digital gates and the K-map. The UML stereotypes and class diagrams are presented. The relationship between the digital gates and the minterms are described. Simplification of boolean expression through UML model is also done through a case study.

Acknowledgements

The authors are very thankful to Prof. B. Hanumaiah, Vice-Chancellor, Babasaheb Bhimrao Ambedkar University (A Central University), Vidya Vihar, Rae Bareilly Road, Lucknow, India, for providing excellent computation facilities in the University campus. Thanks are also due to the University Grant Commission, India, for providing financial assistance to the Central University for research work.

References

[1] Mano, Morris M. (1991), Digital Design, Second Edition, Prentice-Hall of India Private Limited.
[2] Rajaraman, V., Radhakrishnan, T. (2001), An Introduction to Digital Computer Design, Fourth Edition, Prentice-Hall of India Private Limited.
[3] Tanenbaum, A. S. (2004), Structured Computer Organization, Fourth Edition, Prentice-Hall of India Private Limited.
[4] Leach, D. P, Malvino, A. P. and Saha, G. (2006), Digital Principles and Applications, Second Edition, Tata McGraw-Hill.
[5] Wakerly, John F. (2005), Digital Design, Principles and Practices, Third Edition Updated, Ninth Indian Reprint, Pearson Education Inc.
[6] Crenshaw, Jack W. (2003), A primer on Karnaugh maps, Embedded Systems Design, Retrieved from http://www.embedded.com/columns/programmers toolbox/16100908?_requestid=264392
[7] Kuphaldt, Tony R. (2007), Lessons in Electric Circuits, Volume IV – Digital, Fourth Edition, Available as part of the Open Book Project collection retrieved from: www.ibiblio.org/obp/electricCircuits

[8] OMG, UML Superstructure Specification, v2.0, Retrieve from http://www.omg.org/cgi-bindoc?formal/05-07-04.
[9] Booch, G., Rumbaugh, J., Jacobson, I. (2004), The Unified Modeling Language User Guide, Twelfth Indian Reprint, Pearson Education.
[10] Roff, T. (2006), UML: A Beginner's Guide, Tata McGraw-Hill Edition, Fifth Reprint.
[11] Gomaa, H. (2001), Designing Concurrent, Distributed, and Real-Time Applications with UML, Proceedings of the 23rd International Conference on Software Engineering (ICSE'01), IEEE Computer Society.
[12] Schattkowsky, Tim (2005), UML 2.0 - Overview and Perspectives in SoC Design, IEEE.
[13] Saxena, V., Arora D. and Ahmad S. (2007), Object Oriented Distributed Architecture System through UML, IEEE International Conference on Advanced in Computer Vision and Information Technology, ACVIT-07, Nov. 28-30, ISBN 978-81-89866-74-7, pp. 305-310.
[14] Kohut, R., Steinbach, B., and Fröhlich, D. (), FPGA Implementation of Boolean Neural Networks using UML.
[15] Damasevicius, R., Stuikys, V. (2004), Application of UML for Hardware Design Based on Design Process Model, IEEE.
[16] Al-Rababah Ahmad, A. (2009), UML - Models Implementations in Software Engineering System Equipments Representations, International Journal of Soft Computing Applications, Issue 4 , pp.25-34, Euro Journals Publishing, Inc., Retrieved from: http://www.eurojournals.com/IJSCA.htm
[17] Sun, Zhenxin, Wong, Weng-Fai, Zhu, Yongxin and Pilakkat, Santhosh Kumar (2005), Design of Clocked Circuits Using UML, IEEE ASP-DAC 2005 (901-904).

Dr. Vipin Saxena: He is a Reader, Founder and Ex-Head, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Application in 1992 & Ph.D. Degree work on Scientific Computing from University of Roorkee (renamed as Indian Institute of Technology, India) in 1997. He has more than 13 years and 08 months of teaching experience and 17 years research experience in the field of Scientific Computing & Software Engineering. Currently he is proposing software designs by the use of Unified Modeling Language for the various research problems related to the Software and Hardware Domains. He has published more than 75 International and National publications.

Manish Shrivastava: He is a Research Scholar, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Applications in 1992. He has more than 12 years of teaching experience. Currently he is actively engaged in the research work on the Unified Modeling Language. He has produced several outstanding research publications.

Dr. Deepak Arora received his Masters in Computer Applications in 2003 and M.Phil. in Computer Science in 2006. He obtained his Ph.D. degree in Computer Science from Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow in 2009. His research mainly focuses on Parallel and Distributed Computing Applications. His current research interests include Mobile and Wireless Networking Architectures, Data Mining and Data Warehousing. For his research efforts, Dr. Deepak has published several peer-reviewed publications in various journals and conferences. He is a senior member of IACSIT and CSI.