Journal of ELECTRICAL ENGINEERING, VOL. 56, NO. 7-8, 2005, 209–212

COMMUNICATIONS

POCKET–PC BOOLEAN FUNCTION SIMPLIFICATION

Ledion Bitincka — George E. Antoniou [*]

In this paper a useful educational tool is presented for minimizing low order Boolean expressions. The algorithm follows the Karnaugh map looping approach. For the implementation, which provides optimal results, C++ coding was used on the Embedded Visual C++ 3.0 Pocket PC using Windows CE Operating System environment. In order to make the overall implementation efficient, the object oriented approach was used. Two examples are presented to illustrate the efficiency of the proposed algorithm. The proposed application is a useful tool for students and professors in the fields of electrical and computer engineering and computing sciences.

K e y w o r d s: Logic design, Circuit simplification, Pocket PC, Windows CE application, Educational tool

1 INTRODUCTION

It is well known that the Karnaugh-map (K-map) technique is an elegant teaching resource for academics and a systematic and powerful tool for a digital designer in minimizing low order Boolean functions.

Why is the minimization of the Boolean expression needed?

By simplifying the logic function, we can reduce the original number of digital components (gates) required to implement digital circuits. Therefore, by reducing the number of gates, the chip size and the cost will be reduced and the computing speed will be increased.

The K-map technique was proposed by M. Karnaugh [1]. Later Quine and McCluskey reported tabular algorithmic techniques for the optimal Boolean function minimization [2], [3]. Almost all techniques have been embedded into many computer aided design packages and in all the logic design university textbooks [4]-[11]. K-map is a graphical representation of a truth table using Gray code order. It is suitable for elimination by grouping redundant terms in a Boolean expression. By optimizing the algorithm it is possible to simplify entirely a given Boolean expression. Unfortunately almost all the techniques along with the Espresso technique [12] do not always guarantee optimal solutions.

In this paper a Pocket PC (Windows CE) -based implementation is proposed for simplifying four-variable Boolean functions, using the K-map looping technique. The implementation is found to have excellent results.

The proposed Pocket PC application is a useful tool for students and professors in the fields of electrical and computer engineering and computer science. It provides a fast and portable way to check and solve problems in digital logic, discrete mathematics and computer architecture courses. Also, the proposed algorithm can be a valuable utility for the computer chip design industry.

2 ALGORITHM

The proposed algorithm is based on the looping of redundant terms. Therefore in order to take a closer look at how to loop two, four or eight 1's to get the least possible number of groups in a K-map table setting, consider the following simple example (lower case letter represents the complement value, for example "a" means complement of "A"):

$$F = aBcd + ABcd + ABcD + AbcD + AbCD + AbCd$$

	cd	cD	CD	Cd
ab	0	0	0	0
aB	1	0	0	0
AB	1	1	0	0
Ab	0	1	1	1

Analyzing the above K-map table, the following looping observation can be made for each 1 present in the Table:

Cell

- $aBcd$ – has one possibility to be paired, with $ABcd$
- $ABcd$ – has two possibilities to be paired, with $aBcd$ and $ABcD$
- $ABcD$ – has two possibilities to be paired, with $ABcd$ and $AbcD$
- $AbcD$ – has two possibilities to be paired, with $ABcD$ and $AbCD$
- $AbCD$ – has two possibilities to be paired, with $AbcD$ and $AbCd$
- $AbCd$ – has one possibility to be paired, with $AbCD$

It is obvious that there are two cells that have one possibility to be paired, namely $aBcd$ and $AbCd$, so they get the highest priority. These two cells get paired first, $aBcd$ gets paired with $ABcd$ and $AbCd$ gets paired with $AbCD$. After these pairings the pairing possibilities

* Image Processing and Systems Laboratory, Department of Computing Sciences, Montclair State University, Montclair, N.J. 0704, USA, E-mail: ledion@bitincka.com, george.antoniou@montclair.edu

of $ABcD$ and $AbcD$ are decremented by one, leaving both of them with one possibility to be paired. They get paired together finishing this way the optimization of the Boolean function, which results in three pairs.

$$F = Bcd + AbC + AcD\,.$$

The observation reveals the presence of a consistency or rule that lies beneath this logic. Extending the described idea the following algorithm is derived for the optimal looping of 1's in a K-map table. Sharing in the following algorithm means a cell included in more than one looping.

```
Step-1: Find and loop a possible octet.
Step-2: Find and loop cells that have one
          possibility to pair.
Step-3: Find and loop cells that have one
          possibility to quad.
    Step-3a: Repeat Step-2 for new cells with one
               possibility to get paired.
    Step-3b: Repeat Step-3 for new cells with one
               possibility to get quaded.
Step-4: Find and loop cells that have two
          possibilities to get quaded without sharing.
    Step-4a: If a quad is found then repeat Step-4
               until it fails.
Step-5: Find and loop cells that have two
          possibilities to get quaded with sharing.
    (always choosing a quad with minimal sharing).
Step-6: Find and loop cells that have two
          possibilities to be paired without sharing
    Step-6a: If a pair is found then repeat Step-2
               until it fails.
    Step-6b: Repeat Step-6 until it fails.
Step-7: Find and loop cells that have two
          possibilities to be paired with sharing
    Step-7a: If a pair is found then repeat Step-2
               until it fails.
    Step-7b: Repeat Step-7 until it fails.
Step-8: If there are cells that have a value of one and
          are not quaded or paired, then the cells either:
          a) Can not be paired/quaded, or
          b) Have more than 2 possibilities
               to get paired/quaded.
    Step-8a: If a) is valid then go to Step-9
    Step-8b: If b) is valid then change the possibilities
               of one of these cells to 2 and go to Step-3
               to repeat the procedure.
Step-9: Form a minimum Boolean Expression
          using the created loops.
```

It is noted that each successful looping is followed by the steps:

a) All the cells involved in the looping are denoted as "done" and are not considered in the algorithm as cells that can trigger a looping.
b) The looping possibilities of cells that could be looped with the just-looped cells are reduced by one if their looping possibilities are more than one.

2.1 Design

The program was developed using the "Embedded Visual C++ 3.0" for Pocket PC, which supports C and C++. The implementation of the program was done in C++ using its object oriented features. The program is divided into two major components: a parent class (Kmap), and a child class (KmapElement). Each of these classes represents a logical division of the K-map table. The Kmap object controls everything related to the K-map table as a whole, such as initializing and simplifying. In order to apply these functions, the Kmap object creates 16 "children" objects representing each cell. Each child object (KmapElement) learns its own properties and can only perform functions on itself.

Breaking the program down in this way is advantageous because the amount of complete, detailed, organized and correct information about the K-map is maximized. The parent object holds 16 children of the class KmapElement and administers the way the simplification methods are called. The child class KmapElement represents one element of the Kmap; some of its properties are: pairing and quading possibilities, the value of the cell, the current status and a pointer to the parent object. These objects need to know the properties of other neighboring objects. KmapElement uses the parent object to access information about its neighbors.

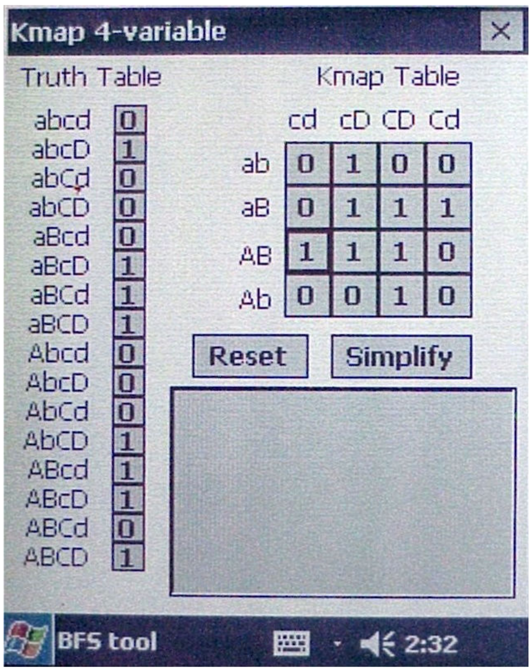
2.2 Program Flow

When input is presented, a Kmap object is created. As a result, 16 children of class KmapElement are created and initialized. During the initialization phase each KmapElement gathers data about itself and its possibilities to be paired, quaded or octeted. The instance variables of this object are updated when a pair, quad or octet is formed.

After initialization, the Kmap object defines the way that the simplification is going to take place according to the presented algorithm. The actual pairing, quading, octeting and updating of the instance variables is done by the functions of the KmapElement class. The order in which these functions are executed is determined by the Kmap class because it knows the algorithm. The simplification happens only once, which means that there are no trials or secondary simplifications. All the functions in the KmapElement have options of choosing sharing and priority in any combination of the two. Each of the functions is executed only on objects that meet the requirements. In the case where the function completes its main task, it updates the instance variables of the neighboring KmapElement objects that need to know the occurred looping.

3 EXAMPLES

Two salient examples, simple yet illustrative of the theoretical concepts presented in this work, follow below:


Kmap 4-variable

Truth Table

abcd	0
abcD	1
abCd	0
abCD	0
aBcd	0
aBcD	1
aBCd	1
aBCD	1
Abcd	0
AbcD	0
AbCd	0
AbCD	1
ABcd	1
ABcD	1
ABCd	0
ABCD	1

Kmap Table

	cd	cD	CD	Cd
ab	0	1	0	0
aB	0	1	1	1
AB	1	1	1	0
Ab	0	0	1	0

Fig. 1. K-map setting of (3)

3.1 Example 1

Consider the following Boolean Expression:

$$F = abCD + abCd + aBcD + aBCD + ABcD + ABCD + Abcd + AbcD \quad (1)$$

The following K-map table is generated.

	cd	cD	CD	Cd
ab	0	0	1	1
aB	0	1	1	0
AB	0	1	1	0
Ab	1	1	0	0

For each one on the table the following data can be collected:

- $abCD$ has two possibilities to be paired.
- $abCd$ has one possibility to be paired.
- $aBcD$ has two possibilities to be paired, and one to quad.
- $aBCD$ has three possibilities to be paired and one to quad.
- $ABcD$ has three possibilities to be paired and one to quad.
- $ABCD$ has two possibilities to be paired and one to quad.
- $Abcd$ has one possibility to be paired.
- $AbcD$ has two possibilities to be paired.

Following the presented algorithm we have

1. There are no octets in the K-map table.
2. Terms $abCd$ and $Abcd$ are paired with $abCD$ and $AbcD$ respectively. $ABcD$ and $aBCD$ pair possibilities are decremented by one. Therefore $abCD$, $abCd$, $Abcd$ and $AbcD$ are marked as looped.
3. The term $aBcD$ is looped in a quad with, $ABcD$, $aBCD$ and $ABCD$. All these cells are marked as looped.
4. No other cells of value of one are marked as looped. All the cells have been included into pairs and quads. Therefore

$$F = abC + Abc + BD\,. \quad (2)$$

The above simplified Boolean expression, with three terms, is the optimal solution for the given Boolean expression (1).

3.2 Example 2

Consider the following Boolean expression:

$$F = abcD + aBcD + aBCD + aBCd + ABcd + ABcD + ABCD + AbCD\,. \quad (3)$$

Using the Compaq iPAQ H3870 Pocket PC the cells, shown in Fig. 1, are selected according to each term of the given Boolean expression (3).

In this case, a quad is possible to be looped, but according to the algorithm any quad of any type will not be looped until all the 1's that have one possibility to be paired are looped. Therefore,

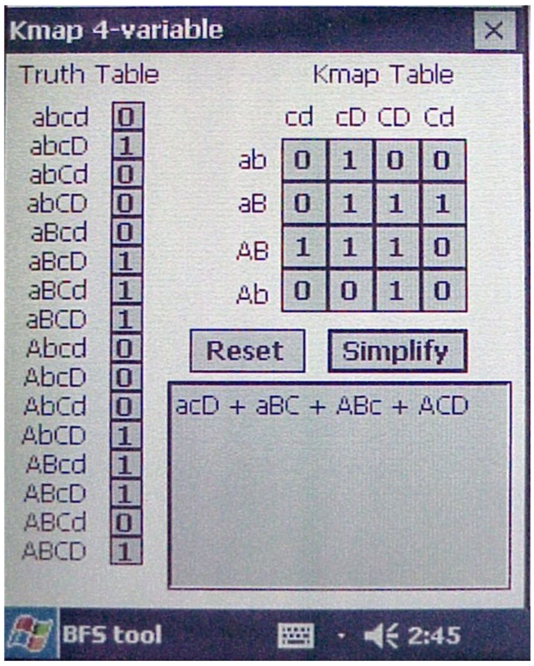
- $abcD$ has one possibility to be paired.
- $aBcD$ has three possibilities to be paired and one possibility to be quaded.
- $aBCD$ has three possibilities to be paired and one possibility to be quaded.
- $aBCd$ has one possibility to be paired.
- $ABcd$ has one possibility to be paired.
- $ABcD$ has three possibilities to be paired and one possibility to be quaded.
- $ABCD$ has three possibilities to be paired and one possibility to be quaded.
- $AbCD$ has one possibility to be paired.

According to the algorithm the following looping combinations can be obtained:

- $abcD$ is paired with $aBcD$ since $abcD$ has one possibility to be paired.
- $aBCd$ is paired with $aBCD$ since $aBCd$ has one possibility to be paired.
- $ABcd$ is paired with $ABcD$ since $ABcd$ has one possibility to be paired.
- $AbCD$ is paired with $ABCD$ since $AbCD$ has one possibility to be paired.

Finally the following result is obtained,

$$F = acD + aBC + ABc + ACD\,. \quad (4)$$



Truth Table

abcd	0
abcD	1
abCd	0
abCD	0
aBcd	0
aBcD	1
aBCd	1
aBCD	1
Abcd	0
AbcD	0
AbCd	0
AbCD	1
ABcd	1
ABcD	1
ABCd	0
ABCD	1

Kmap Table

	cd	cD	CD	Cd
ab	0	1	0	0
aB	0	1	1	1
AB	1	1	1	0
Ab	0	0	1	0

acD + aBC + ABc + ACD

Fig. 2. The result as in (4)

Using the Compaq iPAQ and pressing the "Simplify" button the above derived result (4) is displayed in the result area, depicted in Fig . 2.

The simplified Boolean expression (4) is the optimal solution for the given Boolean expression (3).

4 NOTE

It is noted that the proposed low order simplification algorithm was tested correctly for all the 65535, using the four variables, combinations in conjunction with the K-map Internet based minimization algorithm [13].

5 CONCLUSION

In this paper, an algorithm was presented to minimize a Boolean expression on a Pocket PC. For the implementation, C++ coding was used on the Embedded Visual C++ 3.0 Pocket PC environment. The .exe file, which is executable on the COMPAQ iPAQ 3870 Pocket PC, is 54K and is available from the authors. The presented application is a very useful tool for college students and instructors in the digital design courses.

Acknowledgment

The authors would like to thank the anonymous reviewers of the paper for their valuable comments.

References

[1] KARNAUGH, M.: The Map Method for Synthesis of Combinatorial Logic Circuits, Trans. AIEE, Communications and Electronics **72** (1953), 593-598.

[2] QUINE, W. V.: The Problem of Simplifying Truth Tables, Am. Math. Monthly **59** No. 8 (1952), 521–531.

[3] McCLUSKEY, E. J.: Minimization of Boolean functions, Bell System Tech. Journal **35** No. 5 (1956), 1417–1444.

[4] GAJSKI, D. D.: Principles of Digital Design, Prentice-Hall, 1997.

[5] WAKERLY, J. F.: Digital Design, Prentice-Hall, New York, 2000.

[6] NELSON, V. P.—NAGLE, H. T.—CARROLL, B. D.—IRWIN, D.: Digital Logic Circuit Analysis and Design, Prentice-Hall, New Jersey, 1995.

[7] KATZ, R..: Contemporary Logic Design, Benjamin/Cummings Publ, Redwood City, CA, 1994.

[8] MANO, M.—KIME, C. R.: Logic Computer Design Fundamentals, Prentice Hall, New York, 2000.

[9] BROWN, S.—VRANESIC, Z.: Fundamentals of Digital Logic with VHDL, McGraw-Hill, New York, 2003.

[10] HAYES, J. P.: Digital Logic Design, New York, 1993.

[11] CHIRLIAN, P. M.: Digital Circuits with Microprocessor Applications, Matrix Publishers, Oregon, 1982.

[12] HILL, F. J,—PETERSON, G. R.: Computer Aided Logical Design with Emphasis on VLSI, Wiley, New York, 1993.

[13] TOMASZEWSKI, S. P.—ILGAZ, I. U.—ANTONIOU, G. E.: WWW-Based Boolean Function Simplification, International Journal of Mathematics and Computer Science **13** No. 4 (2003), 577–583.

Received 18 October 2004

Ledion Bitincka (BSCS) was awarded the Bachelor of Science in Computer Science degree (Magna Cum Laude) in 2003 from Montclair State University, Montclair, NJ, USA. He is currently with the Department of Biochemistry and Biophysics of the University of California San Francisco, San Francisco, CA, USA. He is a student member of the IEEE.

George E. Antoniou (PhD) was awarded his Doctor of Electrical Engineering (Informatics) degree in 1998 from National Technical University of Athens, Athens, Greece. He is currently a professor in the department of Computer Science at Montclair State University, Montclair, New Jersey, USA. His present research interests include system theory, multidimensional digital signal processing and FPGA/VHDL applications. He is a senior member of the IEEE.