Modified Karnaugh Map for Quantum Boolean Circuits Construction

Shiou-An Wang, Chin-Yung Lu, I-Ming Tsai, and Sy-Yen Kuo
Department of Electrical Engineering and Graduate Institute of Electronic Engineering
National Taiwan University
Taipei, Taiwan
sykuo@cc.ee.ntu.edu.tw

***Abstract*—Karnaugh map is an efficient method of minimization for conventional logic design. Unfortunately, it is usually used for 3 or 4 variables, at most 6 variables. In this paper, we modify the Karnaugh map and propose a set of reduction rules for quantum Boolean circuit optimization. By applying these rules, we can efficiently simplify a quantum Boolean circuit that has an arbitrary number of input variables. In terms of the space consumption, we use only one auxiliary qubit as the output qubit, and keep all the input qubits unchanged.**

Keywords-quantum Boolean circuits; Karnaugh map; logic minimization

I. INTRODUCTION

Quantum computing is one of the most rapidly expanding research fields recently. Over the last few years, several quantum algorithms, such as Shor's quantum factoring [8] and Grover's fast database search algorithm [7] have emerged. They are much faster than their best classical counterparts. To implement a quantum computer, we need to construct quantum Boolean circuits which consist of quantum gates. Unlike conventional AND-OR-NOT-based circuits, quantum Boolean circuits are based on NOT, CN, and CCN gates. Although with different building blocks, they can still be synthesized by the classical AND, XOR, and NOT functions.

Up to date, the number of quantum bits that can be used to implement a quantum computer is still small. It is not feasible now to build a large circuit for the general application. Therefore, constructing a quantum circuit is always limited by the number of quantum bits, and so quantum logic circuits need to be reduced into a simpler format. Several simplification methods have been proposed [2-6]. However, some methods are complex and others can not be applicable to large circuits easily. Developing a simple and easy simplification method is very important for the implementation of quantum circuits.

Karnaugh map is an efficient method of minimization for conventional logic design. However, using a Karnaugh map to simplify quantum logic circuits is not easy because the XOR functions are substituted for the OR functions in a quantum logic design. It makes a little bit difference between the classical and the quantum logic circuits by using the Karnaugh map to simplify logic functions. Moreover, Karnaugh map can be used manually only for a few variables. It is not a good method when there are a large number of variables, e.g. over 6 variables. In this paper, we modify the form of a Karnaugh map, and we propose an algorithm to simplify logic functions with any number of variables by using the modified Karnaugh map.

II. MODIFIED KARNAUGH MAP

In classical computation, Karnaugh map is used as a method for minimizing a Boolean expression. It is usually aided by a rectangular map of the values of the expression for all possible input combinations. Input values are arranged in a Gray code, which is an ordering of 2^n binary numbers such that only one bit changes from one entry to the next. Maximal rectangular groups that cover the inputs where the expression is true give a minimum implementation. Karnaugh map can be used to find the most efficient quantum logic circuit [4], but it is useful only for a circuit of binary function with at most 3 or 4 inputs. It is difficult to use the Karnaugh map when more input variables are involved. In the following sections, it is explained how a modified Karnaugh map can be used to simplify a quantum Boolean circuit.

Definition 1. Let M be a map with m input variables with size $2^{m/2}\times 2^{m/2}$ (m is even) or $2^{(m-1)/2}\times 2^{(m+1)/2}$ (m is odd). We define $E(M)$ as the Boolean expression of M with variables v_1, v_2, ... , v_m. v_i represents the i-th variable of M.

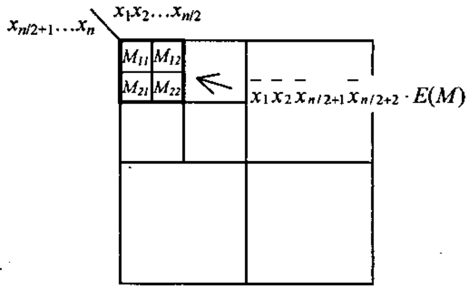


Figure 1. Map partitioning.

Without loss of generality, we assume that the number of input variables n is an even number. We can construct a $2^{n/2}\times 2^{n/2}$ modified Karnaugh map from the truth table. Unlike the original design of Karnaugh map, we label each row and column a binary number from 0 to $2^{n/2}$-1, and fill in the blanks with 0, 1, or X (don't care). In our algorithm, a modified Karnaugh map can be equally divided into 4 square parts (a map with an odd number of input variables can be divided into 2 square parts) and each part can further be divided recursively, as shown in Fig.1. Assume we divide a map M with size $2^{m/2}\times 2^{m/2}$ into 4 equal square parts M_{11}, M_{12}, M_{21} and M_{22}, where M_{11} is the upper left part; M_{12} is the upper right part, etc. (Fig. 1), then

$$E(M)=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})+v_1\bar{v}_{m/2+1}E(M_{12})+\bar{v}_1v_{m/2+1}E(M_{21})+v_1v_{m/2+1}E(M_{22}) \quad (1)$$

where "+" denotes OR. Because M_{11}, M_{12}, M_{21}, M_{22} are disjoint with each other, so we can rewrite Eq. (1) as

$$E(M)=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})\oplus v_1\bar{v}_{m/2+1}E(M_{12})\oplus\bar{v}_1v_{m/2+1}E(M_{21})\oplus v_1v_{m/2+1}E(M_{22}) \quad (2)$$

where "$\oplus$" denotes XOR. For example, M is the square with bold border as shown in Fig. 1. Then the Boolean expression of the square can be written as $\bar{x}_1\bar{x}_2\bar{x}_{n/2+1}\bar{x}_{n/2+2}\cdot E(M)$, where $E(M)$ is the Boolean expression of M with variables $x_3, \ldots, x_{n/2}, x_{n/2+3}, \ldots, x_n$.

Theorem 1. If there is a modified Karnaugh map with n input variables $x_1, x_2, \ldots, x_n$, and M is a square part of it. The size of M is $2^{m/2}\times 2^{m/2}$. Let $v_1, v_2, \ldots, v_m$ be the variables of M, where v_i represents the i-th variable of M. Then

$$v_i=\begin{cases}x_{(n-m)/2+i} & ,1\le i\le m/2\\ x_{n-m+i} & ,m/2+1\le i\le m\end{cases}$$

Proof: Let a modified Karnaugh map with n input variables be divided into 4 square parts. Each part of them has the same x_1 value and the same $x_{n/2+1}$ value. For example, the values of x_1 and $x_{n/2+1}$ in the upper left part are all 0. So we can take out the variables x_1 and $x_{n/2+1}$, and leave $x_2, \ldots, x_{n/2}, x_{n/2+2}, \ldots, x_n$. After the k-th division, the number of variables m becomes n-$2k$, i.e. $k=(n-m)/2$. The first $(n-m)/2$ variables are fixed and can be excluded, therefore $v_1 = x_{(n-m)/2+1}$, $v_2 = x_{(n-m)/2+2}$, ... , and $v_i = x_{(n-m)/2+i}$, where $1 \le i \le m/2$. Similarly, $x_{n/2+1}, \ldots, x_{n/2+(n-m)/2}$ can be excluded, $v_{m/2+1} = x_{n-m/2+1}$, $v_{m/2+2} = x_{n-m/2+2}$, ... , and $v_i = x_{n-m+i}$, where $m/2+1 \le i \le m$.

III. Reduction Rules

In this section, we introduce a set of reduction rules. By applying these rules, we can transform the modified Karnaugh map into a simplified quantum Boolean circuit.

Definition 2. Let M be a modified Karnaugh map which consists of $2^{m/2}\times 2^{m/2}$ squares. V_{ij} represents the Boolean value of the square on the i-th row and j-th column of M. We call M' a submap of M and M'' the remainder of M if M' and M'' have the same size as M and $V_{ij} = V'_{ij} \oplus V''_{ij}$.

Rule 1. If M_{11}, M_{12}, M_{21}, M_{22} have common submap M', then they are reduced to $E(M')$.

Proof: By Eq. (2),

$$\begin{aligned}E(M)&=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})\oplus v_1\bar{v}_{m/2+1}E(M_{12})\oplus\bar{v}_1v_{m/2+1}E(M_{21})\oplus v_1v_{m/2+1}E(M_{22})\\&=\bar{v}_1\bar{v}_{m/2+1}E(M')\oplus v_1\bar{v}_{m/2+1}E(M')\oplus\bar{v}_1v_{m/2+1}E(M')\oplus v_1v_{m/2+1}E(M')\oplus E(M'')\\&=E(M')\oplus E(M'')\end{aligned}$$

where M'' is the remainder of M.

Rule 2. If M_{ij} and M_{kl} (where i=k or j=l) have common submap M', that is, two square parts on the same row or on the same column have common submap M', then they are reduced to $X\cdot E(M')$ ("$\cdot$" denotes AND), where

$$X=\begin{cases}\bar{x}_{n-m/2+1} & ,i=k=1\\ x_{n-m/2+1} & ,i=k=2\\ \bar{x}_{(n-m)/2+1} & ,j=l=1\\ x_{(n-m)/2+1} & ,j=l=2\end{cases}.$$

Proof: In the case of $i = k = 1$. The two upper square parts M_{11} and M_{12} have common submap M'. Then

$$\begin{aligned}E(M)&=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})\oplus v_1\bar{v}_{m/2+1}E(M_{12})\oplus\bar{v}_1v_{m/2+1}E(M_{21})\oplus v_1v_{m/2+1}E(M_{22})\\&=\bar{v}_1\bar{v}_{m/2+1}E(M')\oplus v_1\bar{v}_{m/2+1}E(M')\oplus E(M'')\\&=\bar{v}_{m/2+1}E(M')\oplus E(M'')\end{aligned}$$

By ***Theorem 1***, $v_{m/2+1} = x_{n-m/2+1}$,

$$E(M)=\bar{x}_{n-m/2+1}E(M')\oplus E(M'')$$

For other cases, they can be shown similarly.

Rule 3. If M_{ij} and M_{kl} (where i=k, j<l or i<k, j=l) are complement to each other, then they are reduced to $X\cdot(Y\oplus E(M_{ij}))$, where

$$Y=\begin{cases}x_{(n-m)/2+1} & ,i=k\\ x_{n-m/2+1} & ,j=l\end{cases}$$

Proof: In the case of $i = k = 1$. The two upper square parts M_{11} and M_{12} are complement to each other. Then

$$\begin{aligned}E(M)&=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})\oplus v_1\bar{v}_{m/2+1}E(M_{12})\oplus\bar{v}_1v_{m/2+1}E(M_{21})\oplus v_1v_{m/2+1}E(M_{22})\\&=\bar{v}_1\bar{v}_{m/2+1}E(M_{11})\oplus v_1\bar{v}_{m/2+1}\overline{E(M_{11})}\oplus E(M'')\\&=\bar{v}_{m/2+1}(\bar{v}_1E(M_{11})\oplus v_1\overline{E(M_{11})})\oplus E(M'')\\&=\bar{v}_{m/2+1}\cdot(v_1\oplus E(M_{11}))\oplus E(M'')\\&=\bar{x}_{n-m/2+1}\cdot(x_{(n-m)/2+1}\oplus E(M_{11}))\oplus E(M'')\end{aligned}$$

For other cases, they can be shown similarly.

Rule 4. If M_{ij} and M_{kl} (where i≠k and j≠l) have common submap M', then they are reduced to $Z\cdot E(M')$, where

$$Z = \begin{cases} x_{(n-m)/2+1} \oplus x_{n-m/2+1} & ,i \neq j \\ \overline{x}_{(n-m)/2+1} \oplus x_{n-m/2+1} & ,i = j \end{cases}$$

Proof : Suppose M_{12} and M_{21} have common submap M', then

$$\begin{aligned} E(M) &= \bar{v}_1\bar{v}_{m/2+1}E(M_{11}) \oplus v_1\bar{v}_{m/2+1}E(M_{12}) \oplus \bar{v}_1 v_{m/2+1}E(M_{21}) \oplus v_1 v_{m/2+1}E(M_{22}) \\ &= v_1\bar{v}_{m/2+1}E(M') \oplus \bar{v}_1 v_{m/2+1}E(M') \oplus E(M'') \\ &= (v_1\bar{v}_{m/2+1} \oplus \bar{v}_1 v_{m/2+1})E(M') \oplus E(M'') \\ &= ((v_1 \oplus v_{m/2+1}) \cdot E(M')) \oplus E(M'') \\ &= ((x_{(n-m)/2+1} \oplus x_{n-m/2+1}) \cdot E(M')) \oplus E(M'') \end{aligned}$$

Similarly, if M_{11} and M_{22} have common submap M', then we can show

$$E(M) = ((\bar{x}_{(n-m)/2+1} \oplus x_{n-m/2+1}) \cdot E(M')) \oplus E(M'')$$

Rule 5. $(x_p \ldots \bar{x}_i \ldots x_j \ldots x_q) \oplus (x_p \ldots x_i \ldots \bar{x}_j \ldots x_q) = x_p \ldots (x_i \oplus x_j) \ldots x_q$

Proof : $(x_p \ldots \bar{x}_i \ldots x_j \ldots x_q) \oplus (x_p \ldots x_i \ldots \bar{x}_j \ldots x_q)$

$$\begin{aligned} &= x_p \ldots (\bar{x}_i x_j \oplus x_i \bar{x}_j) \ldots x_q \\ &= x_p \ldots (x_i \oplus x_j) \ldots x_q \end{aligned}$$

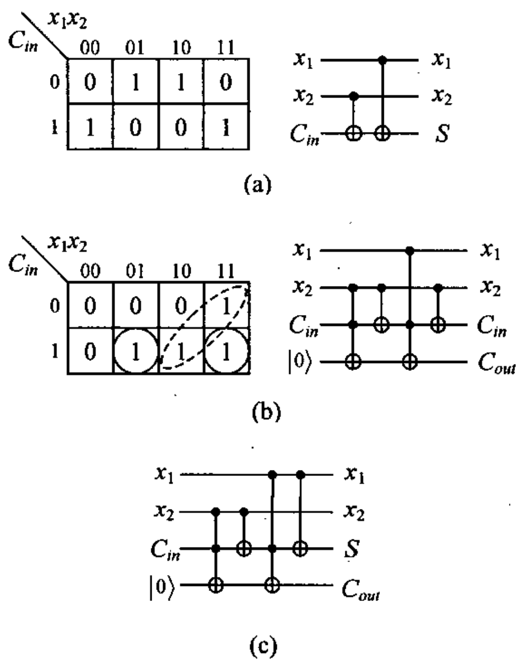
Now we give a concrete example using the truth table of a full adder. The truth table of a full adder is shown in Fig. 2. It has 3 input variables x_1, x_2 ,C_{in}, and 2 output bits S, C_{out}. The modified Karnaugh map of the sum bit is shown in Fig. 3(a). We can divide the map M into two equal square parts, left part M_1 and right part M_2. Then we can find that M_1 and M_2 are complement to each other. By applying Rule 3, we obtain $E(M)$=$x_1 \oplus E(M_1)$. In this example, the variable X in Rule 3 is useless because the number of input variables is an odd number. $E(M_1)$ can be simplified to $x_2 \oplus C_{in}$ by Rule 4. If it is not necessary to keep C_{in} unchanged, we don't even need an auxiliary qubit as the output qubit. The quantum circuit is also shown in Fig. 3(a). The modified Karnaugh map of the carry out bit is shown in Fig. 3(b). It can be divided into two square parts too. The left part M_1 and the right part M_2 have common submap circled by solid line. By applying Rule 2, they can be reduced to x_2C_{in}. In addition, the submap circled by the dotted line can be simplified to $x_1 \cdot (x_2 \oplus C_{in})$ by Rule 4. Thus the carry out bit can be represented as $x_2C_{in} \oplus (x_1 \cdot (x_2 \oplus C_{in}))$. The last CN gate in the circuit (Fig. 3(b)) is used to restore the value of C_{in}. Fig. 3(c) shows the final circuit of a full adder.

IV. ALGORITHM

In this section, we propose an algorithm that shows how to apply the reduction rules we presented in the previous section to get an optimal circuit. Based on the rules, we can construct a series of steps to find the best solution. The more squares can be grouped, the simpler the logic function will be. Unlike general Karnaugh map, the XOR functions is substituted for the OR functions, so few 0's can be grouped with 1's if it is helpful. After applying the reduction rule, we change 1's into 0's and 0's into 1's, and repeat the steps of simplification until all 1's on the map disappear.

x_1	x_2	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 2. The truth table of a full adder



x_1x_2
C_{in}
00 01 10 11
0 0 1 1 0
1 1 0 0 1
x_1 x_2 C_{in} x_1 x_2 S
(a)
x_1x_2
C_{in}
00 01 10 11
0 0 0 0 1
1 0 1 1 1
x_1 x_2 C_{in} $|0\rangle$ x_1 x_2 C_{in} C_{out}
(b)
x_1 x_2 C_{in} $|0\rangle$ x_1 x_2 S C_{out}
(c)

Figure 3. The modified Karnaugh map and the circuit of a full adder

It is possible that more than one reduction rules can be applied at the same time, so we need a mechanism to estimate the cost of each selection. Our cost function is based on [1]. The arbitrary 2-bit gate is taken as the basic operation for convenience. Cost (k) means the number of basic operations used to produce a Toffoli gate with k control bits. The cost function is defined as below:

TABLE I. COST FUNCTION

k	1	2	3	4	5	≥ 6
Cost(k)	1	5	13	29	61	48k-108

Suppose that the original truth table have n variables. The algorithm is described as follows:

Step 1. Construct a modified Karnaugh map with size $2^{\lfloor n/2 \rfloor} \times 2^{\lceil n/2 \rceil}$ from the truth table. Partition the modified Karnaugh map recursively.

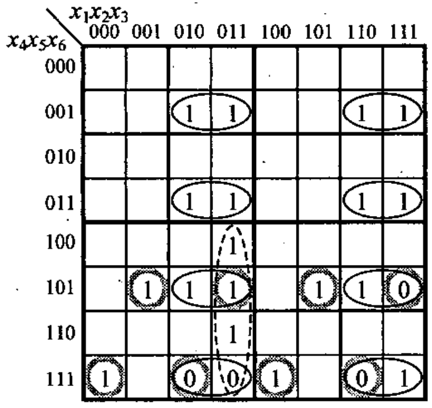
Step 2. Find the possible groups that can be simplified by applying reduction rules. A group shall include as many 1's as possible.

Step 3. Select a group that is the largest in size.

Step 4. Apply the reduction rules to the selected group and simplify it recursively. Then change 1's into 0's and 0's into 1's in the selected group. If there is any 1 on the map, go to step 2.

Step 5. Calculate the total cost of the expression. If there are other possible solutions, restore the modified Karnaugh map and go to step 2.

Step 6. Select the expression with minimum cost as our final solution.



$x_4x_5x_6 \backslash x_1x_2x_3$	000	001	010	011	100	101	110	111
000								
001			1	1			1	1
010								
011			1	1			1	1
100				1				
101		1	1	1		1	1	0
110				1				
111	1		0	0	1		0	1

Figure 4. A modified Karnaugh map with 6 variables

As an example, a modified Karnaugh map is shown in Fig. 4. From the truth table, the original expression can be implemented using 18 7-bit Toffoli gates if no reduction is applied. It is too costly to realize this circuit due to the number of qubits involved in a single quantum gate. Our algorithm can be used to simplify the original circuits. There are two possible solutions after going through our algorithm. The first logic expression is $(x_2x_6)\oplus((x_3\oplus x_5)x_4x_6)\oplus(\bar{x}_1x_2x_3x_4)$, and the cost of this expression is 49. The second logic expression is $(x_2\bar{x}_4x_6)\oplus((x_2\oplus x_3\oplus x_5)x_4x_6)\oplus(\bar{x}_1x_2x_3x_4)$ with cost 59. So we select the former logic expression as our final result. With this expression, the final circuit can be derived easily. It consists of only 2 CN gates and 3 Toffoli gates. The circuit is shown in Fig 5.

V. Conclusion

In this paper, we have proposed an algorithm that transforms an original truth table into a quantum Boolean circuit. With our algorithm, we can reduce not only the number of quantum gates but also the basic operations of the circuit. Barence et al. [1] have shown that any n-qubit gate with n-1 control bits can be simulated in terms of $\Theta(n^2)$ basic operations. However, an n-qubit gate with n-2 control bits can be simulated by only $\Theta(n)$ basic operations for $n \geq 7$. In our circuit, there is at most one quantum gate with n-1 control bits. Thus, we can reduce the total gate count and the number of basic operations effectively with our method. In addition, we use only one qubit as the output qubit, and keep all the input variables unchanged. No other auxiliary qubits or intermediate storage are needed. Thus, it is efficient in terms of both space and time.

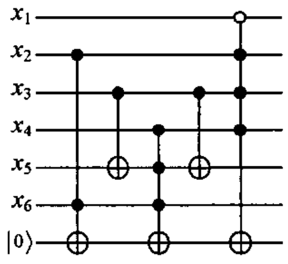


Figure 5. The final circuit

References

[1] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," Physical Review A, 52(5): pp. 3457-3467, Nov. 1995.

[2] I-Ming Tsai and Sy-Yen Kuo, "Quantum Boolean Circuit Construction and Layout under Locality Constraint," in Proc. Of the 1st IEEE Conference on Nanotechnology, 2001, pp. 111-116.

[3] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes, "Reversible logic circuit synthesis," Proc. IEEE/ACM Intl. Conf. on Computer Aided Design, pp. 353-360, November, 2002.

[4] J. -S. Lee, Y. Chung, J. Kim, and S. Lee, "A Practical Method of Constructing Quantum Combinational Logic Circuits," http://arXiv.org/quant-ph/9911053, 1999.

[5] K. Iwama, Y. Kambayashi, and S. Yamashita, "Transformation Rules for Designing CNOT-based Quantum Circuits," Proc. DAC, pp.419-425, June, 2002.

[6] K. N. Patel, I. L. Markov, and J. P. Hayes, "Efficient Synthesis of Linear Reversible Circuits," http://arXiv.org/quant-ph/0302002, 2003.

[7] L. Grover, "A fast quantum mechanical algorithm for database search," in Proc. of the 28th Annual ACM symposium on the Theory of Computing, pp. 212-219, 1996.

[8] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in Proc. of the 35th Annual IEEE Symposium on the Foundations of Computer Science, pp. 124-134, 1994