A Numerical Method for Reed-Muller Circuit Synthesis

M. A. Thornton, V.S.S. Nair
Department of Computer Science and Engineering
Southern Methodist University
Dallas, Texas, 75275

Abstract

A numerical computation based circuit synthesis technique is presented for the Reed-Muller canonical form. The synthesis methodology is reduced to the problem of finding a solution to linear system of equations in the real-field. The mathematical formulation for this technique is developed and it is shown that a unique solution exists. A synthesis example is presented along with a discussion on implementation issues.

1 Introduction

As the complexity of VLSI circuitry increases, testability concerns tend to grow proportionally. The well-known properties of the Reed-Muller canonical form make it an attractive candidate for testable circuit implementations [1]. Previously, the automated synthesis of Reed-Muller logic circuits was accomplished via symbolic Boolean algebra manipulations [2] [3], or the use of the Reed-Muller transforms [9] [10] [13]. In this paper, we present an approach similar to the Reed-Muller transform technique, however, we develop a more general mathematical framework such that the calculations may be performed in the field of real numbers. Specifically, it is shown that the Reed-Muller synthesis methodology can be reduced to solving a system of linear equations over the real-number field.

By translating the Reed-Muller logic circuit realization problem into a numerical calculation, we are able to take advantage of the vast amount of results available in the areas of numerical analysis and linear system theory (such as LINPACK [6]). Numerical techniques using spectral information have been proposed for general logic circuit synthesis in the past [7] [8] [11]. These techniques have suffered from the exponentially increasing sizes of the transformation matrices. Recent work has resulted in more efficient means for the computation of spectral coefficients [12] which has renewed interest in spectral techniques and numerical methods in general. The approach presented here differs from the Reed-Muller transform in that we formulate the problem as the determination of a solution of a general linear system in the field of real numbers rather than using spectral, or correlation, computations.

The organization of this paper is as follows. First, the mathematical foundations of this technique are developed and all of the necessary intermediate results are derived. Following the mathematical formulation, a synthesis example is given. The details of the synthesis example, including the specific solution vector and resulting circuit, are provided. Further implementation concerns are discussed including the Input/Output and efficiency issues. Finally, we a discuss the advantages and motivation for the use of this technique. In section 4 we conclude the results of this research.

2 Mathematical Formulation

The development of the mathematical foundations of this technique relies upon the concepts of rings and morphisms.

The Boolean ring satisfies all of the properties of a general ring along with the idempotence property [4]. The familiar Boolean algebra is equivalent to the Boolean ring, $\Re$, defined as follows:

$$\Re : \{0, 1, \oplus, \cdot\}$$

Where 0 and 1 denote the logic values of zero and one, the addition operator, $\oplus$, denotes the binary XOR operation, and the multiplicative operator, $\cdot$, denotes the binary AND function. Clearly, the set, $\Re$, forms a Boolean ring since all of the properties of a

general ring are satisfied [5] along with the idempotence property.

Next, consider an alternative set, $\Re'$. This set contains the following elements:

$$\Re' : \{\emptyset, I, +, \times\}$$

Where:

$$\emptyset \equiv \{0, \pm 2, \pm 4, \pm 6, \ldots, \pm 2j, \ldots\}$$

and,

$$I \equiv \{\pm 1, \pm 3, \pm 5, \ldots, \pm(2i+1), \ldots\}$$

Thus, $\emptyset$ and I are the sets of all even integers (including 0) and all odd integers (excluding 0), respectively. The two operators in the set, $\Re'$, are the additive operator, $+$, and the multiplicative operator, $\times$. These operators perform real addition and real multiplication, respectively. $\Re'$ also satisfies the properties of a Boolean ring [4].

The following lemma establishes the relationship between the two rings $\Re$ and $\Re'$:

Lemma 1 : *The function, $f(x) = x(\mathrm{mod}2)$, forms a ring morphism from $\Re'$ to $\Re$.*

Proof: To prove this lemma, we demonstrate that the following relationships exist:

$$f(a+b) = f(a) \oplus f(b) \tag{1}$$

$$f(a \times b) = f(a) \cdot f(b) \tag{2}$$

for the following three exhaustive cases:

Case 1 : $(a, b) \in \emptyset$

Case 2 : $(a, b) \in I$

Case 3 : $a \in \emptyset, b \in I$

Case 1:

Let $a = 2n$ and $b = 2m$:

$$\begin{aligned} f(a+b) &= f(2n+2m) \\ &= [2(n+m)](\mathrm{mod}2) \\ &= 0 \\ f(a) \oplus f(b) &= [2n(\mathrm{mod}2)] \oplus [2m(\mathrm{mod}2)] \\ &= 0 \oplus 0 \\ &= 0 \end{aligned}$$

$\therefore f(a+b) = f(a) \oplus f(b)$

$$\begin{aligned} f(a \times b) &= f(2n \times 2m) \\ &= 4nm(\mathrm{mod}2) \\ &= 0 \\ f(a) \cdot f(b) &= [2n(\mathrm{mod}2)] \cdot [2m(\mathrm{mod}2)] \\ &= 0 \cdot 0 \\ &= 0 \end{aligned}$$

$\therefore f(a \times b) = f(a) \cdot f(b)$

Case 2:

Let $a = 2n + 1$ and $b = 2m + 1$:

$$\begin{aligned} f(a+b) &= f(2n+1+2m+1) \\ &= [2(n+m+1)](\mathrm{mod}2) \\ &= 0 \\ f(a) \oplus f(b) &= (2n+1)(\mathrm{mod}2) \oplus (2m+1)(\mathrm{mod}2) \\ &= 1 \oplus 1 \\ &= 0 \end{aligned}$$

$\therefore f(a+b) = f(a) \oplus f(b)$

$$\begin{aligned} f(a \times b) &= f[(2n+1) \times (2m+1)] \\ &= [2(2nm+n+m)+1](\mathrm{mod}2) \\ &= 1 \\ f(a) \cdot f(b) &= (2n+1)(\mathrm{mod}2) \cdot (2m+1)(\mathrm{mod}2) \\ &= 1 \cdot 1 \\ &= 1 \end{aligned}$$

$\therefore f(a \times b) = f(a) \cdot f(b)$

Case 3:

Let $a = 2n$ and $b = 2m + 1$:

$$\begin{aligned} f(a+b) &= f(2n+2m+1) \\ &= [2(n+m)+1](\mathrm{mod}2) \\ &= 1 \\ f(a) \oplus f(b) &= 2n(\mathrm{mod}2) \oplus (2m+1)(\mathrm{mod}2) \\ &= 0 \oplus 1 \\ &= 1 \end{aligned}$$

$\therefore f(a+b) = f(a) \oplus f(b)$

$$\begin{aligned} f(a \times b) &= f[2n \times (2m+1)] \\ &= [2(2nm+n)](\mathrm{mod}2) \\ &= 0 \\ f(a) \cdot f(b) &= 2n(\mathrm{mod}2) \cdot (2m+1)(\mathrm{mod}2) \\ &= 0 \cdot 1 \\ &= 0 \end{aligned}$$

$\therefore f(a \times b) = f(a) \cdot f(b)$

Since equations 1 and 2 hold for all 3 cases, $f(x) = x \bmod 2$ forms a ring morphism between $\Re'$ and $\Re$. □

Next, the Reed-Muller form is presented with a 3-variable example as motivation for the following

mathematical results. The complement-free ring sum formulation of the canonical Reed-Muller form for a three-variable function may be expressed as:

$$F(x) = r_0 \oplus r_1\dot{x_1} \oplus r_2\dot{x_2} \oplus r_3\dot{x_3} \oplus r_4\dot{x_1}\dot{x_2} \oplus$$

$$r_5\dot{x_1}\dot{x_3} \oplus r_6\dot{x_2}\dot{x_3} \oplus r_7\dot{x_1}\dot{x_2}\dot{x_3} \quad (3)$$

Where the dotted variables represent function inputs that are either all complemented, or, none are complemented. The r_i terms are Boolean constants that have value "1" or "0". Equation 3 contains operations and elements from the Boolean ring, $\Re$ (the omission of a binary operator between two consecutive variables in 3 implies that the operation denoted by $\cdot$ occurs). Equation 3 is rewritten in the following form:

$$F(x) = r_0g_0 \oplus r_1g_1 \oplus r_2g_2 \oplus r_3g_3 \oplus r_4g_4 \oplus$$

$$r_5g_5 \oplus r_6g_6 \oplus r_7g_7 \quad (4)$$

Where the r_i are Boolean constants described above, and each g_i is the AND (product) of a subset of the function's literals. Each particular realization of the complement-free ring sum of the Reed-Muller form is then simply a specified set of r_i values that will be referred to as the vector, $\underline{R}$. The set of functions, g_i, will be referred to as the "constituent functions" of $f(x)$. It is convenient to rewrite equation 4 in vector matrix form as:

$$G\underline{R} = \underline{F} \quad (5)$$

Where the elements of the function vector, $\underline{F}$, and the constituent function matrix, G, are known (or are easily computed), and the elements of the $\underline{R}$ vector specify the complement-free ring sum formulation of the Reed-Muller canonical form. The column vectors formed from all possible outputs of the constituent functions, g_i, are concatenated to form the coefficient matrix, G, and the corresponding function outputs are concatenated to form the function vector, $\underline{F}$.

The two forms for the complement-free ring sum expression were given in equation 3. This implies that two different coefficient matrices exist for 5, denoted by G and G'. The matrix, G, is used when all $\dot{x_i} = x_i$ and G' is used when all $\dot{x_i} = x_i'$ ($'$ indicates the application of the complement function to a Boolean variable). In the following, we define the coefficient matrix, G formally.

Definition 1 : *The matrix, G, is a concatenation of the output column vectors of the constituent functions, g_i, with no inverted inputs.* □

Consider the smallest possible G matrix, denoted as G_2, where the subscript indicates the number of function inputs.

$$G_2 = \begin{array}{c} \begin{matrix} g_0 & g_1 & g_2 & g_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Where the constituent functions are defined as:

$$\begin{aligned} g_0 &= 1 \\ g_1 &= x_0 \\ g_2 &= x_1 \\ g_3 &= x_0 \cdot x_1 \end{aligned}$$

Likewise, the matrix G_2' is defined as:

$$G_2' = \begin{array}{c} \begin{matrix} g_3' & g_2' & g_1' & g_0 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Where the constituent functions are defined as:

$$\begin{aligned} g_0 &= 1 \\ g_1' &= x_0' \\ g_2' &= x_1' \\ g_3' &= x_0' \cdot x_1' \end{aligned}$$

Lemma 2 : *The coefficient matrices, G and G' are triangular matrices with all $g_{ii} = g_{ii}' = 1$.*

Proof: The trivial matrix,

$$G_1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

is triangular.

Also, from definition 1, G_2 is also seen to be triangular. Higher ordered matrices, G_n, may be recursively defined using G_1 as a kernel since all functions, g_i, are AND functions or literal values themselves. The following definition holds:

$$G_n = \begin{bmatrix} G_{n-1} & 0 \\ G_{n-1} & G_{n-1} \end{bmatrix}$$

Also, G'_1 and G'_2 are triangular and G'_2 may also be used to recursively define G'_n as follows:

$$G'_n = \begin{bmatrix} G'_{n-1} & G'_{n-1} \\ 0 & G'_{n-1} \end{bmatrix}$$

Where,

$$G'_1 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Since G_1 and G_2 are triangular and G_n may derived in terms of G_{n-1}, by induction, G_n is also triangular since it is formed with sub-matrices, G_{n-1}, along its diagonal with the all zero sub-matrix in one quadrant. □

Next, we will show that the solution vector, $\underline{R}$, always contains components that are members from the field of positive and negative integers, Z. Where Z is defined as:

$$Z = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$$

Lemma 3 : *The solution vector, $\underline{R}$, in the equation, $G\underline{R} = \underline{F}$, contains only integer components.*

Proof: As mentioned previously, the coefficient matrices, G_n and G'_n, contain only 1 and 0 elements. Also, these matrices are triangular with diagonals consisting of all 1's. Hence:

$$det(G_n) = det(G'_n) = 1$$

It is desired to solve the equation:

$$G^*\underline{R} = \underline{F}$$

Where G^* represents either G_n, or G'_n. The solution to this equation is well known and may be expressed in terms of the cofactors of G^* as:

$$\begin{aligned} \underline{R} &= G^{*-1}\underline{F} \\ &= [det(G^*)]^{-1}[cofactor(G^*)]\underline{F} \\ &= [1][cofactor(G^*)]\underline{F} \end{aligned}$$

Since the cofactor matrices are formed with no division operations, the matrix, $[cofactor(G^*)]$, contains only integers. Since the matrix, G^*, always has a determinant of 1, the solution vector, $\underline{R}$, is formed as a vector-matrix multiplication of the integer matrix, $[cofactor(G^*)]$, and the integer vector, $\underline{F}$. Hence the solution vector always contains integer components.

□

This lemma establishes the fact that the solution of equation 5 is performed over the Boolean ring, $\Re'$. This result along with the previous definitions and lemmas allow the following theorem to be stated and proved:

Theorem 1 : *Given any Boolean function, F, and a binary vector, $\underline{B} = [b_1, b_2, b_3, \ldots, b_n]^T$ such that $F = b_1g_1 \oplus b_2g_2 \oplus \ldots \oplus b_ng_n$, then there exists a vector, $\underline{R}$ with each $r_i \in Z$, such that $G\underline{R} = \underline{F}$ and $\underline{R}(\text{mod } 2) = \underline{B}$. Where $\underline{F}$ is the binary vector formed from the output column of the truth table of the function, F.*

Proof: From lemma 3, $\underline{R}$ has the property that each $r_i \in Z$. From lemma 1, the function, $f(x) = x(\text{mod}2)$ forms a ring morphism between $\Re'$ and $\Re$. Hence, the application of $f(x)$ to each component in the vector, $\underline{R}$, isomorphically maps $\underline{R}$ to $\underline{B}$. □

Corollary 1 : *Every Boolean function has a unique complement-free ring sum form of the Reed-Muller expansion.*

Proof: From lemma 2 it was proven that all G_n and G'_n matrices are triangular. From definition 1 it was noted that all $g_{ii} = g'_{ii} = 1$. Hence the determinant of the coefficient matrices is 1, guaranteeing a unique matrix inverse exists. Since $\underline{R}$ and $\underline{B}$ from theorem 1 are isomorphic and $\underline{R}$ is unique due to the existence of a unique inverse of G and G', then $\underline{B}$ also is unique. □

This section has presented the mathematical justification for the synthesis technique for the complement-free Reed-Muller forms. These results may easily be extended to any of the generalized Reed-Muller (GRM) expansions with any arbitrary polarity number. The following section will provide a numerical example and a discussion of implementation issues.

3 Example

Consider the Boolean function specified by the following truth table:

x_2	x_1	x_0	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Figure 1: Truth Table Contents of the Function to be Synthesized

The resulting matrix equation is given as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \underline{R} \quad = \quad \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Solving this system in the real field yields the following solution vector:

$$\underline{R^T} \quad = \quad \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 1 & -1 & 0 \end{bmatrix}$$

Mapping this vector, from $\Re'$ to $\Re$ using $f(x)$, we obtain:

$$\underline{\tilde{R}^T} \quad = \quad \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Each element of the solution vector corresponds to a specific r_i resulting in a realization of the Reed-Muller form as given in 3. The resulting function is:

$$f(x) = 1 \oplus x_1 \oplus x_2 x_1 \oplus x_2 x_0 \oplus x_1 x_0 \qquad (6)$$

Currently, we have implemented an experimental version of this code using the C language. It is apparent, however, that due to the well-defined nature of the G_n and G'_n matrices, it is not necessary to compute a linear equation solution for each synthesis. In fact, a more efficient method would consist of accessing a database of the inverses of G_n or G'_n and simply performing a vector-matrix multiplication for each synthesis task similar to computing the Reed-Muller transform in the real field.

The input to the experimental algorithm is the output vector of the function to be synthesized. However, it would be very easy to incorporate a Boolean expression parser at the front end to produce the output vector automatically, resulting in defining the input in terms of an equation.

The output of our program is of the form of a a structural Hardware Description Language (HDL) net-list. By choosing this form for output, the corresponding circuit is easily verified and may be piped into any automated synthesis tool that supports structural HDL input. Specifically, the output of this program is in the *Verilog*© HDL.

4 Conclusions

A numerical technique for Reed-Muller circuit synthesis has been developed and presented. It has been shown that this technique is equivalent to determining the solution to a set of linear equations in the real-number field. An example of this technique was provided with a brief description of implementation issues.

Further investigations into numerical techniques for logic synthesis are ongoing. In particular, we are investigating the use of heuristics for the reduction of the synthesis algorithm complexity (both space and time) and the use of generalized linear transformation techniques. The result presented in this paper is significant in our ongoing research since it develops the necessary mathematics for performing Reed-Muller circuit synthesis using real-number arithmetic and providing a unique solution.

References

[1] Reddy, S.M. Easily Testable Realizations for Logic Functions, *IEEE Trans. Comp.* **vol. C-21, no. 11**, pp.1183-1188, 1972.

[2] Muller, D.E. Application of Boolean Algebra to Switching Circuit Design and to Error Detection, *IRE Trans. Elec. Comp.* **vol. EC-3**, pp.6-12, Sept. 1954.

[3] Mukhopadhyay, A. and Schmitz, G. Minimization of EXCLUSIVE OR and LOGICAL EQUIVALENCE Switching Circuits, *IEEE Trans. Comp.* **vol. C-19, no. 2**, pp.132-140, 1970.

[4] Bartee, T.C., Lebow, I.L., and Reed, I.S., **Theory and Design of Digital Machines**, Mc-Graw-Hill, New York, 1962.

[5] Larney, V.H., **Abstract Algebra A First Course**, Prindle, Weber, and Schmidt, Boston, MA., 1975.

[6] Dongarra, J., Bunch, J.R., Moler, C.B., and Stewart, G.W. , **LINPACK Users Guide**, SIAM Publications, Philadelphia, PA., 1978.

[7] Thornton, M.A., Nair, V.S.S. An Iterative Combinational Logic Synthesis Technique Using Spectral Information, *To Appear in Proceedings of EURO-DAC '93*

[8] Edwards, C.E., The design of easily tested circuits using mapping and spectral techniques, *Radio and Electronic Engineer* **vol. 47, no. 7**, pp.321-342, 1977.

[9] Green, D., **Modern Logic Design**, Addison-Wesley, Reading, Massachusetts, 1986.

[10] Davio, M., Deschamps, J.-P., and Thayse, A., **Discrete and Switching Functions**, Mc-Graw-Hill, New York, 1978.

[11] Hurst, S.L., Miller, D.M., and Muzio, J.C., **Spectral Techniques in Digital Logic**, Academic Press, Orlando, Florida, 1985.

[12] Clarke, E.M., McMillan, K.L., Zhao, X., Fujita, M., and Yang, J., Spectral Transforms for Large Boolean Functions with Applications to Technology Mapping, *30-th ACM/IEEE Design Automation Conference*, Dallas, Texas, 1993.

[13] Varma, D. and Trachtenberg, E.A. Design Automation Tools for Efficient Implementation of Logic Functions by Decomposition, *IEEE Trans. on CAD* **vol. 8 no. 8**, August, pp. 901-916, 1989.