# Fast Reed-Muller Spectrum Computation Using Output Probabilities *

M. A. Thornton

Department of Computer Systems Engineering
University of Arkansas
Fayetteville, AR 72701
Tel: 501-575-6036
Fax: 501-575-5339
e-mail: mitch@seas.smu.edu

V. S. S. Nair

Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75275
Tel: 214-768-2856
Fax: 214-768-3085
e-mail: nair@seas.smu.edu

**Abstract— This paper presents a new methodology for the computation of the Reed-Muller spectral coefficients of a function of any fixed polarity using its OBDD representation. By using past results that allow the computations to be performed using real arithmetic, an efficient technique may be developed in the real domain with the resulting coefficients obtained by using a simple mapping relation to the $GF(2)$ domain. These results mathematically justify the OBDD based approach developed in this work. This result is novel since it relies on the use of OBDDs and the concept of a Boolean function output probability to compute the coefficients. This approach is also very general in that it allows other types of coefficients (such as the Walsh) to be computed as well as the Reed-Muller forms with a single OBDD operation. A small example of this technique is given to illustrate the approach followed by some experimental results.**

## I. Introduction

The Reed-Muller (RM) spectrum can be used to provide information regarding the realization of a Boolean function. However, when the spectrum is computed as a vector-matrix product, difficulties arise for functions of even moderate size due to large memory requirements. We present a new approach that overcomes this difficulty by allowing the function to be represented using a binary decision diagram thus reducing the storage requirements. In addition, the computational requirements are also reduced since an efficient method for computing the spectral coefficients is employed.

By exploiting the relationship between output probabilities of logic functions and their corresponding spectral values, we have developed an efficient method for computing the RM spectrum of a function of any fixed polarity. The spectrum computation problem is first formulated in terms of operations using real valued arithmetic to mathematically justify the approach. Next, it is shown that the spectral calculation problem may be translated to the computation of the probability that the output of a function is a logic "1" given all possible input values are equally likely to occur. With this approach, we are essentially computing the number of minterms using the concepts of output probabilities as they were first defined by Parker and McCluskey in [1].

The method presented here is a viable alternative to other efficient spectral computation techniques recently developed by other researchers [2] [3] [4] [5]. These techniques use MTBDDs that are formed by performing BDD operations that represent the computation of a vector matrix product. In contrast, we use OBDDs that represent the logic function being transformed and a function representing a specific transform matrix row. The method presented here has the desirable property of allowing each spectral coefficient to be computed separately so that memory resources for the entire spectral vector are not required.

An approach very similar to the one presented here is given in [6]. In this similar approach, the BDD representing the function to be transformed is used. For a given RM coefficient, the BDD is examined for paths terminating in a logic "1" node that represent a specific set of minterms. Although our method can be viewed as a technique for for counting minterms also, it has the desirable feature of allowing other forms of spectral coefficients (such as the Walsh forms) to be computed as well as the RM types for a single OBDD operation.

Other approaches have been developed to compute spectral coefficients effectively using disjoint sets of cubes to represent the function to be transformed [7] [8]. The method we present can be advantageous to these approaches since, for large functions, the corresponding OBDDs may be very compact while the determination of a corresponding set of disjoint cubes can result in a large list. Further, methods to compute OBDD representations directly from multi-level circuit diagrams may be

*This project was supported in part by an NSF grant under contract MIP-9410822.

employed in our method without resorting to "flattening" a multi-level circuit in order to compute a corresponding set of product terms.

One other important approach to the minimization of the fixed polarity forms, and more general Exclusive-OR Product of Sums (ESOP) forms is given in [9]. This method uses ternary decision diagrams (TDDs) and accompanying algebraic subexpressions. The TDD representation requires the nonterminal nodes to have three children, two are similar to those found in standard OBDDs with the third representing an XOR relation with preceding node variables. Although this approach is more general in that it handles ESOP forms other than the fixed polarity class, it can require more memory during the computations.

Recently we developed an efficient technique for the computation of the Walsh coefficients using output probability expressions (OPEs) [10]. This methodology was not directly applicable to the RM spectrum since it relied upon computations performed in the real field. In this work, we show that the RM spectral coefficients are mathematically related to values that may be computed efficiently using real arithmetic. Once these computations are complete, the actual RM spectral values are obtained by using the isomorphic relation developed in the work described in [11]. This result allows the recently developed efficient method for the computation of the Walsh spectrum to be applied to the RM spectrum as well.

This work was motivated by the fact that other methods for the efficient computation of the fixed polarity RM coefficients required the generation of a list of disjoint cubes representing the function to be transformed, the use of BDD techniques to compute a vector matrix product, or, the evaluation of a BDD for paths terminating at a logic "1" node. By developing this technique, we were able to verify results of our experimentation with estimation of fixed-polarity numbers for minimal RM forms using the same method as that for computing the Walsh coefficients [10]. Since this method allows the coefficients to be computed separately, it may also prove to be useful for the application of RM coefficients to the Boolean matching problem such as that recently proposed in [12]. In addition, the usual motivations for utilizing the RM forms also apply such as a reduced product set for certain classes of functions [13] [14], and the desirable testability properties [15].

The remainder of the paper is organized as follows. First, an overview of the pertinent results in [11] are presented which describe why the computations may be performed using real arithmetic. Next, the relationship between the output probabilities and the spectral coefficients are presented along with an efficient method for their computation. Following the formulation of the computation technique, a small example is given to illustrate the method. Finally, a section on experimental results is given followed by the conclusions.

## II. Overview of the Relation Between $GF(2)$ and the Real Field

This section provides a brief description of the pertinent results given in [11]. Although these results may be used to directly compute the RM spectral coefficients using real arithmetic, no savings in computation requirements will result. The purpose of this section is to provide the theoretical background needed to explain the computationally efficient approach in the next section. Specifically, the results of this section show the relationship between an integer valued linear transformation and the various fixed polarity RM spectra. In particular, these results describe an isomorphic mapping function between the Galois field containing two numbers, $GF(2)$, and the real field. Further, it is shown that the isomorphic function maps into equivalence classes containing single elements and hence may be used to determine a unique value. Since the RM transform is defined over $GF(2)$, the existence of an isomorphism that provides a unique mapping is crucial because it will allow the transform calculations to be performed in the real field where the efficient computation method can be exploited. In order to formally develop the isomorphic relation, the concept of an algebraic ring and hence, a ring morphism is utilized.

A Boolean ring satisfies all of the properties of a general ring along with the idempotence property [16]. The Boolean algebra can be described with respect to the Boolean ring, $\Re$, defined as follows:

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

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. The set, $\Re$, forms a Boolean ring since all of the properties of a general ring are satisfied [17] along with the idempotence property.

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

$$\Re' : \{\emptyset, I, +, \times\} \tag{2}$$

Where:

$$\emptyset \equiv \{0, \pm 2, \pm 4, \pm 6, \ldots, \pm 2j, \ldots\} \tag{3}$$

and,

$$I \equiv \{\pm 1, \pm 3, \pm 5, \ldots, \pm(2i+1), \ldots\} \tag{4}$$

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 [16].

The following lemma establishes the relationship between the two rings $\Re$ and $\Re'$. The proof of this lemma appears in [11].

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

Although Lemma 1 gives the isomorphic relationship between the rings, $\Re$ and $\Re'$, it is also necessary to show that the mapping is one-to-one so that a particular real value, $r_i'$, does not map into an equivalence class of values in $\Re$ that contain more than one value. This fact may be shown to be true based upon the uniqueness of the solution of a linear set of equations. In particular, the RM transformation matrix is used to form a set of equations whose solutions lie in the real field. The following is a definition of the polarity-0 RM transformation matrix, however **all of the results hold for matrices of any arbitrary polarity** since they are all triangular.

**Definition 1** *The matrix, $G$, is a concatenation of all possible product terms with each row forming the output vector of the constituent functions, $g_i$.*

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

$$G_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} g_0 \\ g_1 \\ g_2 \\ g_3 \end{matrix} \quad (5)$$

The constituent functions for $G_2$ are defined as:

$$\begin{aligned} g_0 &= \overline{x}_1 \cdot \overline{x}_2 \\ g_1 &= \overline{x}_1 \\ g_2 &= \overline{x}_2 \\ g_3 &= 1 \end{aligned}$$

As described in [18] [19], the RM spectrum, $\underline{R}$, may be computed as given in Equation 6 where all arithmetic operations are performed using those defined in $\Re$.

$$\underline{R} = G\underline{F} \quad (6)$$

As defined above, matrix, $G$, is formed as the concatenation of column vectors corresponding to the output vectors of all possible product terms. Since the RM form is a linear combination of the product vectors, $\underline{R}$ must also satisfy Equations 7 and 8.

$$G\underline{R} = \underline{F} \quad (7)$$

$$\underline{R} = G^{-1}\underline{F} \quad (8)$$

The reason Equation 6 holds is because $G = G^{-1}$ in the ring $\Re$. However $G \neq G^{-1}$ in the ring $\Re'$, therefore the solution of $G\underline{R} = \underline{F}$ is required when real valued arithmetic is used.

The following Lemmas establish that the solution of the linear system given in Equation 7 is unique, thereby allowing the morphism to always map to a single value in $GF(2)$. The proofs of these Lemmas appear in [11].

**Lemma 2** *The coefficient matrix, $G$, is triangular with all $g_{ii} = 1$.*

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

Lemma 3 establishes the fact that the solution of the matrix equation performed over the Boolean ring, $\Re'$ is always unique. This result along with the previous definitions and lemmas allow Theorem 1 to be stated. The proof for this theorem is given in [11].

**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 $F = b_1 g_1 \oplus b_2 g_2 \oplus \ldots \oplus b_n g_n$, then there exists a vector, $\underline{R}$ with each $r_i \in Z$, such that $G\underline{R} = \underline{F}$ and $\underline{R}(\mathrm{mod}2) = \underline{B}$. Where $\underline{F}$ is the binary vector formed from the output column of the truth table of the function, $f$.*

This subsection has presented the mathematical justification for computing the RM spectrum using real arithmetic and then using the mapping relation to obtain the coefficients in $GF(2)$. Since the computations may be performed in the ring, $\Re'$, an efficient method may be formulated and used as described in the following section.

## III. Relationship of Output Probabilities and the RM Spectra

The output probability of a circuit is the probability that the output of the circuit is a logic "1" given the probabilities that each of the inputs are at a logic "1" value. In past work, OPEs were used to evaluate the effectiveness of random testing [1]. Two methods were developed for computing the OPE in algebraic form in [1]. One method used the Boolean logic equation and the other used logic diagrams. In [10] we noted that for all possible inputs, each input is equally likely to be "0" or "1", and thus if $\frac{1}{2}$ is used as the probability value for all inputs, the resulting evaluation of the OPE will yield the overall percentage of times that the function output is at logic "1".

Following this observation, we presented a methodology for computing this quantity directly from an OBDD representation of a logic function without resorting to using symbolic algebraic manipulations to form the actual OPE. This technique is called the probability assignment algorithm (PAA), and is summarized as follows.

### Probability Assignment Algorithm

1: Assign probability = 1 for the initial node.

2: If the probability of node $j = P_j$, assign a probability of $\frac{1}{2}P_j$ to each of the outgoing arcs from $j$.

3: The probability, $P_k$, of node $k$ is the sum of the probabilities of the incoming arcs.

In the development given in the preceding subsection, it was shown that a RM spectral coefficient may be obtained by first computing a value in $\Re'$ (denoted as $r'$) and then mapping it to $\Re$. Unfortunately the formation and solution of the matrix equation to compute the values in $\Re'$ has a complexity similar to computing the coefficient directly using $GF(2)$ arithmetic. However, the $r'$ value can be computed without resorting to solving a matrix equation by exploiting its relation to the output probability of a circuit. Before this result is derived, the following definition is helpful.

**Definition 2** *A Boolean function, $f \cdot g_c$, which is composed as the AND of a function to be transformed, $f$, and a constituent function, $g_c$, is termed the composition function and is denoted by $f_{comp}$.*

Definition 2 allows Lemma 4 to be easily stated and proven.

**Lemma 4** *The real value, $r'$, is directly proportional to the output probability of the Boolean function, $f_{comp}$.*

**Proof:** The value, $r'$, can be viewed as the inner product of the output vector of the function to be transformed, $f$, and some constituent function, $g_c$. Since each element in these two output vectors is either 1 or 0, each partial product of the inner product is also 1 or 0. In fact, the partial product value of 1 only occurs when both $f$ and $g_c$ produce an output of 1 for a common given set of input values. Thus, $r'$ is the total number of times the composition function, $f_{comp} = f \cdot g_c$, produces an output of 1. If the output probability of $f_{comp}$ is known (denoted by $\wp\{f_{comp}\}$) then $r'$ is easily computed as shown in Equation 9. Note that $\wp\{f_{comp}\}$ is evaluated with all function variables, $x_i$, assumed to have a corresponding $\wp\{x_i\} = \frac{1}{2}$.

$$r' = 2^n \times \wp\{f_{comp}\} \tag{9}$$

Therefore $r'$ is directly proportional to the output probability of the composition function with a constant of proportionality of $2^n$ where $n$ is the number of primary inputs. □

It is now clear that the value of $r'$ can be obtained without computing an inner product if the output probability is known. This leads to the main result of this paper and is given in Theorem 2.

**Theorem 2** *A RM spectral coefficient, $r$, may be directly computed using the value of the output probability of $f_{comp}$.*

**Proof:**

From Lemma 1 it was shown that a particular RM coefficient can be computed from a real value as:

$$r = r'(\text{mod}2) \tag{10}$$

From Lemma 4, the value, $r'$ is related to the output probability, $\wp\{f_{comp}\}$. Substituting Equation 9 into Equation 10 yields the final result as given in Equation 11.

$$r = (2^n \wp\{f \cdot g_c\})(\text{mod}2) \tag{11}$$

□

Theorem 2 provides the necessary theoretical relation for the implementation of a method for computing a RM spectral coefficient without requiring storage resources proportional to the size of a functions' truth table.

## IV. Example and Experimental Results

A program for the efficient computation of the RM spectral coefficients was implemented using the $C$ programming language. The BDD package developed by David Long was used for all BDD processing. The program receives the BDD description of the circuit to be transformed as input. Based upon the desired polarity number, a set of constituent functions corresponding to the rows of the RM transformation matrix are specified. For each computation of a spectral value, an OBDD representation of each composition function is formed. After the formation of the OBDD representing $g_c$, the $APPLY$ algorithm [20] is invoked thus forming the OBDD representing the composition function, $f_{comp}$. Next, the PAA is invoked resulting in the output probability which is converted to the RM spectral coefficient according to Equation 11.

Many logic functions are represented in a very compact form when they are expressed as BDDs [21] in contrast to the exponentially large size (in terms of number of inputs) required by a truth table, and hence an output vector. This fact allows the methodology implemented here to compute a RM coefficient with complexity $O(\|E_{comp}\|)$ where $\|E_{comp}\|$ is the number of edges in the BDD representing $f_{comp}$. However, there are some functions that have a number of BDD nodes that are comparable to the number of truth table values [22] [23]. For these cases, our methodology degenerates to having a complexity equivalent to performing the matrix calculations. Fortunately, many functions of practical importance are described with far fewer BDD nodes than truth table entries.

The overall time complexity of our approach is $O(\|E_f\| \times \|E_{g_c}\|)$ for each RM coefficient since each application of the $APPLY$ algorithm visits each node in the BDD representing $f$ a number of times equal to the size of the OBDD representing $g_c$. Since the constituent functions for the RM transform are extremely simple,

the OBDD representing $f_{comp}$ is generally comparatively small with respect to the OBDD for $f$. Most of the time $\|E_{comp}\| < \|E_f\| \times \|E_{g_c}\|$ so the PAA algorithm requires relatively little computation time. The storage requirement also has a complexity of $O(\|E_{comp}\|$ since only the storage of the composition function BDD is required.

Instead of computing and storing a probability value (which lies in the interval $[0,1]$), the numerator of the probability value is stored (which lies in the interval, $[0,2^n]$). This modification alleviates possible underflow errors, however the numerator value can overflow for large values of $n$. The potential overflow problem was addressed by storing a mantissa and exponent value at each node. The exponent value is a power of 2 and is always as large as possible. Therefore, a node at the $k^{th}$ level in the OBDD always has an exponent value that is at least $n-k$. This observation leads to an interesting fact as given in Lemma 5.

**Lemma 5** *A necessary condition for a RM spectral coefficient, $r_i$, to have a value of 1 is that the OBDD representing the composition function, $f \cdot g_c$, must contain a path consisting of n non-terminal nodes and a logic-1 valued terminal node.*

**Proof:** Since the initial node is labeled with an exponent value of $n$ and the probability value of a node is halved to obtain the value of subsequent nodes, each node at the $k^{th}$ level of the OBDD will have an exponent that is at least $n-k$ in value. Since the coefficient, $r_i$ is computed as the modulo-2 value of the product of the probability value and $2^n$ as given in Equation 11, the exponent of the terminal "1" node must be equal to zero in order for $r_i = 1$. An exponent value can never reach zero unless $n$ nodes in a single path have been traversed, allowing the initial nodes' exponent value, $n$, to be decremented $n$ times. □

As an example of the computation, consider the Boolean function described by Equation 12.

$$f(x) = \overline{x}_1\overline{x}_3 + x_1\overline{x}_2x_3 + \overline{x}_1x_2 + x_2\overline{x}_3 \qquad (12)$$

The constituent function corresponding to the $7^{th}$ RM spectral coefficient for a polarity-0 transform is given by Equation 13.

$$g_6(x) = \overline{x}_3 \qquad (13)$$

The output vector of Equation 13 is identical to the $7^{th}$ row of the polarity-0 RM transformation matrix for a 3 variable function. The corresponding OBDD representations for $f(x)$ and $f(x) \cdot g_6(x)$ are given in Figure IV.

The PAA algorithm is applied to the composition function, $f(x) \cdot g_6(x)$, and the probabilities assigned to each node are shown in Figure IV. The required quantity is the sum of all probabilities at each terminal logic "1" node and is given as $\wp\{f(x) \cdot g_6(x)\} = 0.625$. Substituting the probability value into Equation 11 yields the RM spectral coefficient as shown in Equation 14.

$$r_6 = [2^3 \times 0.625](mod2) = 1 \qquad (14)$$

This implementation was tested using some of the $ISCAS85$ benchmark circuits as input. The following tables contain some results of the computations. For ease of description, we will refer to the subset of RM coefficients corresponding to constituent functions containing $m$ literals as the $m^{th}$-order subset of RM coefficients. The particular polarity of each literal is specified by the polarity number of the corresponding RM transformation matrix that defines the set of constituent functions.

Table I contains the results of the $0^{th}$-order ($g_c = 1$) RM coefficients for several different $ISCAS85$ circuits. The label assigned to the particular output used is given followed by the number of inputs, the size of the OBDD in terms of vertices, and, the CPU time on a DECstation 500 that the computations required. The CPU time also includes the time required to convert the $ISCAS$ netlist into an OBDD representation. Since the OBDD for $f_{comp}$ is exactly the same as the OBDD for $f$ in the case of the $0^{th}$-order RM coefficient, only $\|f\|$ is given in Table I.

Table II contains some of the $1^{st}$-order RM coefficients for the circuit, $c432$ output $329gat$ which has 27 inputs and 2792 vertices in its OBDD representation. The entire amount of CPU time required for the computation of Table II was 1.0 seconds including the time required to parse the input netlist and construct the OBDD for $c432$.

TABLE I
$O^{th}$ ORDER RM COEFFICIENTS FOR VARIOUS NETLISTS

| Netlist | Output | # Inputs | $\|f\|$ | CPU Time (sec) |
|---|---|---|---|---|
| $c432$ | $223gat$ | 18 | 2791 | 0.2 |
| $c880$ | $850gat$ | 29 | 196 | $< 0.1$ |
| $c2670$ | 188 | 14 | 14 | 0.1 |
| $c3540$ | 399 | 26 | 185 | 0.1 |
| $c6288$ | $4946gat$ | 24 | 17057 | 7.6 |
| $c7552$ | 376 | 28 | 52 | 0.3 |

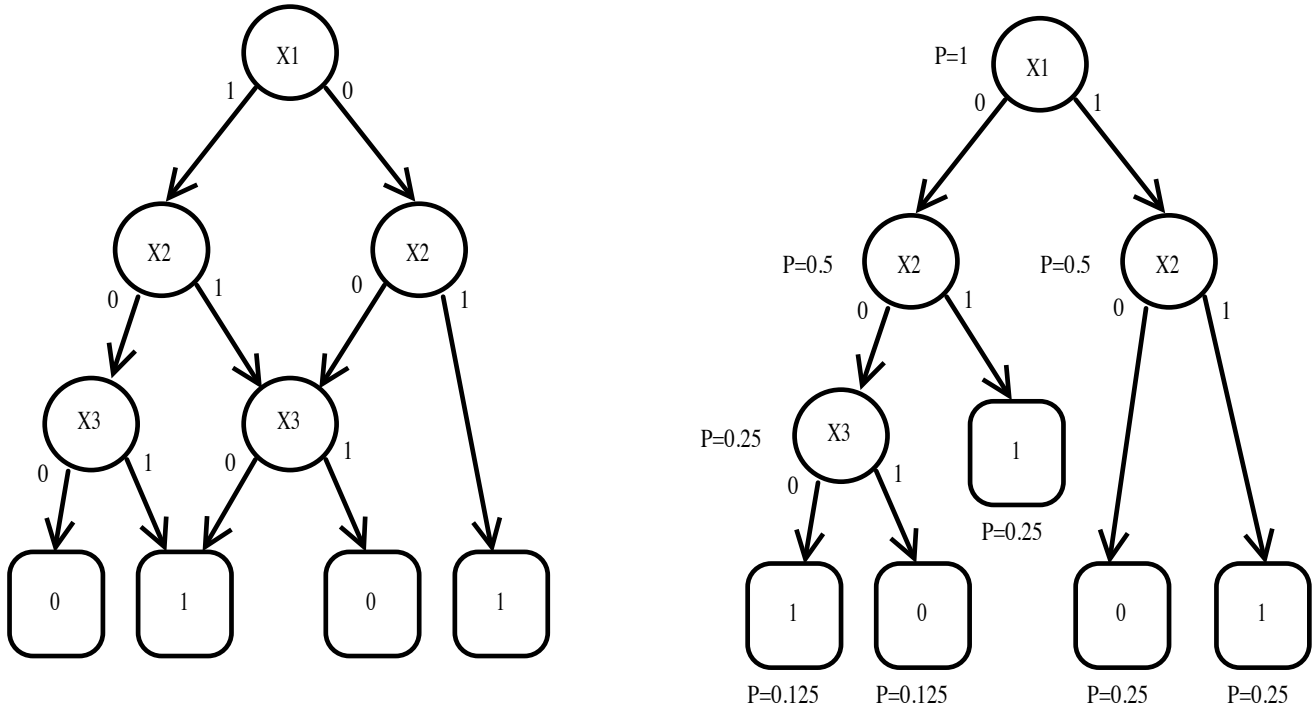


Fig. 1. Binary decision diagrams of example function and composition function.

TABLE II
$1^{st}$ Order RM Coefficients for Output $329gat$ of $c432$

| Constituent Function | $\|\|f \cdot g_c\|\|$ | Spectral Coefficient |
|---|---|---|
| $\overline{1gat}$ | 1387 | 0 |
| $\overline{4gat}$ | 2156 | 1 |
| $\overline{11gat}$ | 1389 | 0 |
| $\overline{17gat}$ | 2160 | 1 |
| $\overline{24gat}$ | 1393 | 0 |
| $\overline{30gat}$ | 2168 | 1 |
| $\overline{37gat}$ | 1401 | 0 |
| $\overline{43gat}$ | 2184 | 1 |
| $\overline{50gat}$ | 1417 | 0 |
| $\overline{60gat}$ | 1488 | 1 |
| $\overline{73gat}$ | 1558 | 1 |
| $\overline{86gat}$ | 1712 | 1 |
| $\overline{99gat}$ | 2028 | 1 |
| $\overline{112gat}$ | 2027 | 1 |

## V. Conclusions

An efficient means for the computation of the RM spectral coefficients for a Boolean function has been presented. The development of the method was described and the relation between the spectral coefficients and the function output probabilities was given. The experimental results illustrate the viability of this approach.

This technique is particularly applicable for large functions where it would be impractical to compute and store the output vector. Since most functions may be represented in a compact manner using an OBDD representation, significant savings in computational resources result. This method is comparable to efficient spectral computation techniques recently proposed by other researchers since it uses BDD structures for the computations. In some cases this approach may be advantageous since it does not require the use of lists of disjoint cubes can become very large and also since the spectral coefficients can be computed one at a time. This technique also allows spectral coefficients of the Walsh type to be computed as well as the RM form with a single application of the probability assignment algorithm.

In the future, we plan to extend the efficient spectral computation methodology to use other forms of input representation. Specifically, the use of shared BDDs [24] would allow multi-output functions to be used as input, and, the use of IBDDs [25] would reduce the size of the required input for some classes of functions.

## References

[1] K. P. Parker and E. J. McCluskey. Probabilistic Treatment of General Combinational Networks. *IEEE Transactions on Computers*, vol. c-24:668–670, June 1975.

[2] E. M. Clarke, K. L. McMillan, X. Zhao, M. Fujita, and J. Yang. Spectral Transformations for Large Boolean Functions with Applications to Technology Mapping. *Proceedings of ACM/IEEE Design Automation Conference*, pages 54–60, 1993.

[3] M. Fujita, J. Yang, E. M. Clarke, X. Zhao, and P. McGeer. Fast Spectrum Computation for Logic Functions using Binary Decision Diagrams. *Proceedings of the International Conference on Circuits and Systems*, 1994.

[4] E. M. Clarke, X. Zhao, M. Fujita, Y. Matsunaga, R. McGeer, and J. Yan. Fast Walsh Transform Computation with Binary Decision Diagram. *Proceedings of the IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 82–85, September 1993.

[5] E. M. Clarke, K. L. McMillan, X. Zhao, and M. Fujita. Spectral Transforms for Extremely Large Boolean Functions. *Proceedings of the IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 86–90, September 1993.

[6] S. Purwar. An Efficient Method of Computing Generalized Reed-Muller Expansions from Binary Decision Diagram. *IEEE Transactions on Computers*, vol. 40, no. 11:1298–1301, November 1991.

[7] B. J. Falkowski, I. Schafer, and M. A. Perkowski. Calculation of the Rademacher-Walsh Spectrum from a Reduced Representation of Boolean Functions. *Proceedings of the European Design Automation Conference*, pages 181–186, September 1992.

[8] A. Sarabi and M. Perkowski. Fast Exact Quasi-Minimal Minimization of Highly Testable Fixed-Polarity AND/XOR Canonical Networks. *Proceedings of the IEEE Design Automation Conference*, pages 30–35, 1992.

[9] T. Sasao (editor). *Logic Synthesis and Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1993.

[10] M. A. Thornton and V. S. S. Nair. Efficient Calculation of Spectral Coefficients and Their Applications. *IEEE Trans. on CAD*, to appear.

[11] M. A. Thornton and V. S. S. Nair. A Numerical Method for Reed-Muller Circuit Synthesis. *Proceedings of the IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 69–74, September 1993.

[12] C. Tsai and M. Marek-Sadowska. Boolean Matching Using Generalized Reed-Muller Forms. *Proceedings of ACM/IEEE Design Automation Conference*, pages 339–344, June 1994.

[13] T. Sasao and P. Besslich. On the Complexity of Mod-2 Sum PLA's. *IEEE Transactions on Computers*, vol. C-39, no. 2:262–266, February 1990.

[14] T. Sasao. A Design Method for AND-OR-EXOR Three-Level Networks. *Proceedings of the 1995 ACM/IEEE International Workshop on Logic Synthesis*, pages 8–11 – 8–20, May 1995.

[15] S. M. Reddy. Easily Testable Realizations for Logic Functions. *IEEE Transactions on Computers*, vol. C-21, no. 11:1183–1188, November 1972.

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

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

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

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

[20] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, vol. C-35, no. 8:677–691, August 1986.

[21] S. B. Akers. Binary Decision Diagrams. *IEEE Transactions on Computers*, vol. C-27, no. 6:509–516, June 1978.

[22] S. Devadas. Comparing Two-Level and Ordered Binary Decision Diagram Representations of Logic Functions. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 5:722–723, May 1993.

[23] R. Bryant. On the Complexity of VLSI Implementations and Graph Representations of Boolean Functions with Applications to Integer Multiplication. *IEEE Transactions on Computers*, vol. 40, no. 2:205–213, February 1991.

[24] S. Minato, N. Ishiura, and S. Yajima. Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean Function Manipulation. *Proceedings of the ACM/IEEE Design Automation Conference*, pages 52–57, 1990.

[25] J. Jain, M. Abadir, J. Bitner, D. S. Fussell, and J. A. Abraham. IBDDs: An Efficient Functional Representation or Digital Circuits. *Proceedings of the European Design Automation Conference*, March 1992.