Int. J. Appl. Math. Comput. Sci., 2002, Vol.12, No.4, 571–579

EFFICIENT CALCULATION OF THE REED-MULLER FORM BY MEANS OF THE WALSH TRANSFORM

PIOTR PORWIK*

* Institute of Computer Science, Silesian University
ul. Będzińska 39, 41–200 Sosnowiec, Poland
e-mail: porwik@us.edu.pl

The paper describes a spectral method for combinational logic synthesis using the Walsh transform and the Reed-Muller form. A new algorithm is presented that allows us to obtain the mixed polarity Reed-Muller expansion of Boolean functions. The most popular minimisation (sub-minimisation) criterion of the Reed-Muller form is obtained by the exhaustive search of all the polarity vectors. This paper presents a non-exhaustive method for Reed-Muller expansions. The new method allows us to build the Reed-Muller form based on the analysis of Walsh-Hadamard coefficients. The presented method has much less complexity than the procedures which have been applied until now. Both the transforms and the presented Walsh-Hadamard spectral characterization of the Reed-Muller expansion are compared. An analysis of the properties of the spectra obtained from these transforms is made.

Keywords: Reed-Muller coefficients, Walsh coefficients, coefficient distribution, Boolean function, synthesis of Boolean functions

1. Introduction

Manipulations and calculations of discrete functions are an important task in many areas of computer science. An example is the exhaustive answer to questions about the equivalence and classifications of Boolean functions, which have applications in various problems of CAD, such as synthesis, verification or testing.

A logic function can be implemented as many different circuit designs. This function can also be implemented as the multi-level tree of XOR gates. In two-valued systems, the testing of the multi-level tree of XOR gates is easy because in a fanout free linear circuit any single fault propagates to the output independently of the applied input vector. This property allows us to minimize the number of tests required for fault detection. Such an implementation can also offer significant benefits by employing fewer transistors, connections and tracks. The testability will not be considered here in view of the fact that the analysis of the circuit testability based on the Reed-Muller representation has been discussed in many works so far (Falkowski and Chang, 1999; Karpovsky, 1985; Sasao, 1993; 1995).

The classical approach to the analysis, synthesis or testing of digital circuits is based on the description by the Boolean algebra operators. Over many years an alternative description based on the operations of modulo-2 arithmetic has been developed (Damarla and Karpovsky, 1989; Falkowski and Chang, 1995; Sasao, 1995; Yanushkevich, 1998). The modulo-2 algebra is the simplest case of the Galois field algebra. Any Boolean function can be represented in the modulo-2 algebra. In practical applications, operations of modulo 2 can be realized by means of exclusive OR (EXOR) gates. The modulo-2 sum-of-products expression is known as the Reed-Muller expansion. Nowadays we can observe that the role of EXOR gates in the design process is very important. The new method allows us to build an optimal Reed-Muller form based on the analysis of Walsh-Hadamard coefficients.

Unfortunately, relatively few functions are fully amenable to the Reed-Muller implementation. However, a lot of well-known functions can be partially realized in the XOR gates technique, and hence we have several canonical families of AND/XOR forms. Many authors have studied these forms, because they offer an interesting compromise between the testability, number of terms, area and speed (Falkowski and Chang, 2000; Karpovsky, 1976; Sasao, 1993). Examples of such canonical binary forms are Shannon, Positive Davio and Negative Davio representations (Sasao, 1995). Any combination of the above types of trees can be used to create canonical trees: Kronecker, Pseudo Kronecker, Reed-Muller or Pseudo Reed-Muller trees. Several types of representations have already been introduced, investigated and implemented in CAD (Falkowski and Chang, 2000; Sasao, 1995), but most of them still remain to be defined and experimentally evaluated.

Any Boolean function $f(x_1, x_2, \ldots, x_n)$ of n variables can be represented by three forms of the Reed-Muller expansion (Falkowski and Chang, 1995; Sasao, 1995). The positive polarity Reed-Muller form is an EXOR sum of products, where each variable is not complemented:

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus \cdots \oplus a_{2^n-1} x_1 x_2 \cdots x_n, \quad (1)$$

with $a_i \in \{0, 1\}$. In the fixed polarity form (or the generalised Reed-Muller expansion, GRM) each variable may appear as either complemented or not complemented:

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1 \dot{x}_1 \oplus a_2 \dot{x}_2 \oplus \cdots \oplus a_{2^n-1} \dot{x}_1 \dot{x}_2 \cdots \dot{x}_n, \quad (2)$$

where $a_i \in \{0, 1\}$ and $\dot{x} \in \{x, \bar{x}\}$. The GRM can be obtained from the positive polarity form using the identity $\bar{x} = 1 \oplus x$. The mixed polarity form (MPRM) allows a variable to appear in both polarities in the same equation. The mixed polarity expansion is more compact than the positive and fixed polarity expansions, because there are no restrictions on the polarity of the input variables. The forms of the Reed-Muller tree can be used directly in designing AND-EXOR PLAs (Perkowski, 1996; Sasao, 1993).

Table 1 shows the relations between the trees described above. This table unifies most of the known EXOR-based representations and presents how many different expressions can be generated for particular types of trees for a given n-variate Boolean function.

Many authors use the term GRM to denote the FPRM (Sasao, 1993). Some of them use another terminology for GRMs. The Kronecker form is also called the mixed polarity Reed-Muller expansion (MPRM). Thus, the terminology is not uniform. Let us observe that there are many more GRMs than FPRMs for a Boolean function, and thus the minimal GRM is not worse (it is usually much better) than the minimal FPRM of the same function. Though the GRM cannot be found for each variable separately, it is more difficult to find a good GRM which must be found for all variables together.

The above-mentioned Reed-Muller expansion is an alternative description of a Boolean function. It employs the modulo-2 arithmetic, which is unique and canonical for a given Boolean function. The application of EXOR and AND gates has only some advantages over other implementations. We know (Perkowski, 1996) that if a circuit is presented as a Reed-Muller expansion, it is easily testable. This implies that the Reed-Muller or Kronecker methods are often very efficient. Because the number of different expansions for each type of a tree is large or very large (see Table 1), how to find the minimal or subminimal expansion is very important. The correlation between the SOP (Sum-of-Product) and ESOP (Exclusive-Sum-of-Product) representations of switching functions is intensively investigated. It has been found that for some specific classes of functions the AND-OR representation may be less economical in use than the AND-EXOR design.

To illustrate this problem, consider the Boolean function f, which has the exact SOP form $f = \bar{x}_1 x_3 + \bar{x}_1 x_2 x_4 + x_1 \bar{x}_2 \bar{x}_3 + x_1 \bar{x}_3 \bar{x}_4$, but minimization as the ESOP

Table 1. Relation between the trees and AND-EXOR expressions.

Type of a tree	Expression generated from the tree	Number of different expressions
Positive Davio	Positive polarity Reed-Muller expression (PPRM)	1
Reed-Muller	Fixed polarity Reed-Muller expression (FPRM)	2^n
	Pseudo Reed-Muller expression (PSDRM)	2^{2^n-1}
No corresponding tree	Generalised Reed-Muller expression (GRM)*	$2^{n2^{n-1}}$
Kronecker	Kronecker expression (KRO), or Mixed Polarity Reed-Muller expression (MPRM)	3^n
Pseudo Kronecker	Pseudo Kronecker expression (PSDKRO)	3^{2^n-1}

* Definition proposed in (Sasao, 1993)

gives a better solution $f = x_1 \oplus x_3 \oplus x_2\bar{x}_3x_4$. Unfortunately, some forms of the Reed-Muller representation can be very inefficient. For example, the PPRM representation of the Boolean function $f = \bar{x}_1\bar{x}_2\bar{x}_3$ gives $2^n = 8$ product terms: $f_{PPRM} = 1 \oplus x_3 \oplus x_2 \oplus x_2x_3 \oplus x_1 \oplus x_1x_3 \oplus x_1x_2 \oplus x_1x_2x_3$. Therefore, in practical solutions partial Reed-Muller implementations are very often used.

On the other hand, the Reed-Muller form of logic implementation allows for a much greater number of possible representations of the Boolean functions. The techniques developed for the optimization of circuit complexity (in particular, the number of gates required to implement the function) in the Boolean domain cannot be applied to the Reed-Muller form. Consequently, recently there has been a lively interest in developing techniques for the optimization of gate requirements in the Reed-Muller domain or for the exact minimization of Reed-Muller expansions. Hence some authors investigated FPRM expansions and presented results which allowed them to generate optimal fixed polarity Reed-Muller expansions of Boolean functions (Falkowski and Chang, 2000; 1995; 1999; Perkowski, 1996). Unfortunately, the methods described by (Falkowski and Chang, 1999) and others are very difficult to use when a Boolean function has a lot of variables. Many authors find the minimal Reed-Muller representation of a function by means of an exhaustive search of all possible representations and the choice of the best one. These methods are impractical for functions with large numbers of input variables (Falkowski and Chang, 1995).

This paper presents a method which allows us to calculate the mixed Reed-Muller polarity expansion directly from Walsh coefficients. The presented algorithm is characterized by a low complexity and can be applied to all n-variate Boolean functions. The total number of arithmetic operations required to compute all Walsh-Hadamard transform coefficients is approximately only $O(2^n \log_2 2^n) = O(2^n n)$, similarly as in the classical Fourier transform.

2. Calculation of Spectral Coefficients

Some solutions in the Boolean domain inform us precisely about the behaviour of the function at a single point but say nothing about its behaviour referring to any other points. That becomes possible when using an alternative representation of a function where the information is much more global in nature. This alternative representation is the spectral domain (Giani *et al.*, 2001; Porwik, 1996; Porwik and Falkowski, 1999). The spectral data are used for many applications in the digital logic design. Some of them include classification of Boolean functions (Hurst *et al.*, 1985; Porwik and Falkowski, 1999), fault synthesis, signal processing (Karpovsky, 1985; Porwik and Falkowski, 1999; Sasao, 1993), etc. At first, spectral data are generated, and then they are manipulated in accordance with the application.

A Boolean function $f(x_1, x_2, \ldots, x_n)$ can be transformed from the domain $\{0, 1\}$ into the spectral domain by a linear transformation $\boldsymbol{T} \cdot \boldsymbol{Y} = \boldsymbol{S}$, where $\boldsymbol{T}$ is a $2^n \times 2^n$ orthogonal transform matrix, $\boldsymbol{Y} = [y_0, y_1, \ldots, y_{2^n-1}]^T$ is the two-valued truth table vector of $f(x_1, x_2, \ldots, x_n)$ and $\boldsymbol{S} = [s_0, s_1, \ldots, s_{2^n-1}]^T$ is the vector of spectral coefficients. The inverse transformation comes back from the spectral domain $\boldsymbol{S}$ to the Boolean function domain by the application of the transform $\boldsymbol{T}^{-1} \cdot \boldsymbol{S} = \boldsymbol{Y}$. The most popular transforms used in the design of logic networks are Walsh, Reed-Muller, (to a lesser degree Haar) and arithmetic transforms (Falkowski and Chang, 2000; Hurst *et al.*, 1985). We can observe that most research works and applications of spectral techniques in logic design were done for either Walsh or Reed-Muller transforms.

By definition, the spectrum of a Boolean function is obtained by multiplying a transformation matrix by the function's output vector. The result of the vector-matrix product is called the spectral vector and is composed of elements that are referred to as spectral coefficients. The type of information that is obtained from the spectral coefficients depends on the transformation matrix. One of several ways of interpreting the meaning of each spectral coefficient is to view it as a measure of the correlation between two binary functions (vectors). Hence the first function is a Boolean function represented by the two-valued truth table vector $\boldsymbol{Y}$ and the second Boolean function is one from the collection of constituent functions of the transformation matrix $\boldsymbol{T}$. For a given Boolean function the Walsh spectrum has only one representation, unlike the Reed-Muller spectrum, where we have a lot of different expansions (see Table 1).

Besides the matrix method outlined in the present paper, data flow graph methods and parallel calculations similar to the Fast Fourier Transform have also been used.

2.1. Walsh Spectral Coefficients

Walsh functions can be generated in a recursive way by using the Hadamard matrix (Hurst *et al.*, 1985). The Hadamard matrix of any dimension is generated as follows:

$$\boldsymbol{T}_w(n) = \begin{bmatrix} \boldsymbol{T}_w(n-1) & \boldsymbol{T}_w(n-1) \\ \boldsymbol{T}_w(n-1) & -\boldsymbol{T}_w(n-1) \end{bmatrix}, \qquad \boldsymbol{T}_w(0) = [1], \quad \boldsymbol{T}_w^{-1}(n) = \frac{1}{2^n}\boldsymbol{T}_w(n). \tag{3}$$

Each row of the matrix $\boldsymbol{T}_w(n)$ thus created includes a discrete Walsh sequence $wal(w,t)$ (in other words, a discrete Walsh function). In this notation, w identifies the number of the Walsh function, and t stands for the discrete point of the function determination interval.

The set of all Walsh sequences included in the matrix $\boldsymbol{T}_w(n)$ constitutes the orthogonal space basis $l^2_{2^m}$. Each Boolean function f can be distributed in a finite Walsh series by multiplying the Hadamard matrix by the truth vector of the Boolean function. In such a case the function f possesses an alternative representation in the form of the ordered set $\boldsymbol{S}$ of Walsh spectral coefficients resulting from multiplication.

We have two forms of Walsh-Hadamard spectra of the Boolean function f known in the literature as the $\boldsymbol{R}$ and $\boldsymbol{S}$ spectra (Falkowski and Porwik, 1999; Hurst *et al.*, 1985; Porwik, 1996). The spectral coefficients of $\boldsymbol{R}$ have exactly the same information content as the coefficients of $\boldsymbol{S}$, but will not have the same magnitudes. Only the $\boldsymbol{S}$ spectrum is used in this paper. In the $\boldsymbol{S}$ coding, the $\{0,1\}$ values corresponding to true and false minterms are respectively replaced by the $\{1,-1\}$ values. Each spectrum coefficient $s_i \in \boldsymbol{S}$ is described by its order. The order is equal to the number of variables describing the linear function, which corresponds to the row in the matrix $\boldsymbol{T}_w(n)$ for a given spectral coefficient. The s_i elements of vector $\boldsymbol{S}$ are ordered according to a straight binary code of literals describing the minterms of the original truth vector $\boldsymbol{Y}$. The relationship between the Walsh coefficients and the variables of a Boolean function can be described as follows.

Definition 1. Any Boolean function $f(x_1, x_2, \ldots, x_n)$ of n variables can be expressed by means of Walsh-Hadamard coefficients as the arithmetical polynomial

$$\begin{aligned} f(x_1,x_2,\ldots,x_n) &= \frac{1}{2^{n+1}}\Big[2^n - s_0 - s_1(-1)^{x_n} - s_2(-1)^{x_{n-1}} \\ &\quad - \cdots - s_{2^n-1}(-1)^{x_1 \oplus x_2 \cdots \oplus x_n}\Big], \end{aligned} \quad (4)$$

where $\oplus$ stands for the modulo-2 addition, and $s_0, s_1, \ldots, s_{2^n-1} \in \boldsymbol{S}$ are spectral coefficients.

The knowledge of the spectral coefficients of Boolean functions raises the possibility of evaluating properties of both a single Boolean function and a selected group of functions. In (Porwik, 1996) it was shown that separate spectral coefficients can be treated as a correlation measure between the basic standard trivial functions corresponding to the coefficients and the Boolean function. Spectral coefficients have many properties which are already well known (Hurst *et al.*, 1985; Karpovsky, 1976). The analysis of the properties of the Hadamard matrix reveals the following important dependence between the spectrum elements s_j and a given Boolean function f.

Property 1. *If a coefficient s_j has a large (resp. small) positive value, then f is strongly (resp. weakly) dependent on the linear combination of variables*

$$\sum_{j=0}^{2^n-1} x_1^{(j_1)} \oplus \cdots \oplus x_n^{(j_n)} \quad \textit{over} \quad \textit{GF(2)}.$$

If a coefficient s_j has a large (resp. small) negative value, then f is strongly (resp. weakly) dependent on the linear combination of variables

$$\sum_{j=0}^{2^n-1} \overline{x_1^{(j_1)} \oplus \cdots \oplus x_n^{(j_n)}} \quad \textit{over} \quad \textit{GF(2)}\,,$$

where

$$x_i^{(j_i)} = \begin{cases} 0 & \textit{if} \quad j_i = 0, \\ x_i & \textit{if} \quad j_i = 1, \end{cases}$$

$j_1, j_2, \ldots, j_n$ constitute the binary representation of a number j, j_1 is the MSB, and j_n is the LSB.

2.2. Mixed Reed-Muller Spectral Coefficients

The mixed polarity Reed-Muller canonical forms represent expansions where each Boolean variable x_i may occur with mixed polarity $c_i = 0, 1, 2$. Here 2 means that this variable can be represented as complemented or not complemented. Therefore we obtain the following family of 3^n mixed polarity Reed-Muller expansions (Sasao, 1995; Yanushkevich, 1998):

$$\begin{aligned} \big(\boldsymbol{T}_{RM}(n)\big)_c &= (\boldsymbol{T}_{RM})_{c_1} \otimes (\boldsymbol{T}_{RM})_{c_2} \otimes \cdots \otimes (\boldsymbol{T}_{RM})_{c_n}, \\ \big(\boldsymbol{T}^{-1}_{RM}(n)\big)_c &= (\boldsymbol{T}^{-1}_{RM})_{c_1} \otimes (\boldsymbol{T}^{-1}_{RM})_{c_2} \otimes \cdots \otimes (\boldsymbol{T}^{-1}_{RM})_{c_n}, \\ (\boldsymbol{T}_{RM})_0 = (\boldsymbol{T}^{-1}_{RM})_1 &= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \\ (\boldsymbol{T}_{RM})_1 &= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \\ (\boldsymbol{T}^{-1}_{RM})_1 &= \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \\ (\boldsymbol{T}_{RM})_2 = (\boldsymbol{T}^{-1}_{RM})_2 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \end{aligned} \quad (5)$$

where $\otimes$ signifies the tensor product, $c \in \{0, 1, \ldots, 3^n - 1\}$ denotes the polarity, and $c_1, c_2, \ldots, c_n$ constitute the ternary representation of c.

The relationship between the mixed polarity Reed-Muller coefficients and the variables of the Boolean function can be expressed as follows:

$$f_c(x_1, x_2, \ldots, x_n) = \sum_{j=0}^{2^n-1} s_j (x_1 \oplus c_1)^{j_1} \cdots (x_n \oplus c_n)^{j_n} \quad \text{over GF(2)}, \quad (6)$$

where

$$(x_i \oplus c_i)^{j_i} = \begin{cases} 1 & \text{if } c_i = 0,1 \text{ and } j_i = 0, \\ x_i & \text{if } c_i = 0 \quad \text{and } j_i = 1 \text{ or } c_i = 2 \text{ and } j_i = 1, \\ \overline{x_i} & \text{if } c_i = 1 \quad \text{and } j_i = 1 \text{ or } c_i = 2 \text{ and } j_i = 0, \end{cases}$$

$j_1, j_2, \ldots, j_n$ constitute the binary representation of j, j_1 is the MSB, j_n is the LSB and $s_0, s_1, \ldots, s_{2^n-1}$ are the spectral coefficients.

Example 1. Determine Walsh-Hadamard and mixed polarity Reed-Muller spectra for the Boolean function $f(x_1, x_2, x_3) = \bar{x}_1\bar{x}_2\bar{x}_3 + \bar{x}_1 x_2 \bar{x}_3 + x_1 x_2 \bar{x}_3$. Using (4)–(6) we can form Table 2.

Table 2. Walsh spectrum of a Boolean function f.

$x_1x_2x_3$	$f(x_1, x_2, x_3)$	$\boldsymbol{S}$	Type of correlation (*)
0 0 0	1	2	0
0 0 1	0	−6	$\bar{x}_3$
0 1 0	1	2	x_2
0 1 1	0	2	$x_2 \oplus x_3$
1 0 0	0	−2	$\bar{x}_1$
1 0 1	0	−2	$\overline{x_1 \oplus x_3}$
1 1 0	1	−2	$\overline{x_1 \oplus x_2}$
1 1 1	0	−2	$\overline{x_1 \oplus x_2 \oplus x_3}$

(*) Each spectral coefficient from $\boldsymbol{S}$ represents a correlation measure between the truth column vector $\boldsymbol{Y}$ of function f and the corresponding Walsh-Hadamard row function.

For Walsh-Hadamard coefficients we obtain from (4) another expansion of f:

$$f(x_1, x_2, x_3) = \frac{1}{16}\Big[6 + 6(-1)^{x_3} - 2(-1)^{x_2} - 2(-1)^{x_2 \oplus x_3} + 2(-1)^{x_1} + 2(-1)^{x_1 \oplus x_3} + 2(-1)^{x_1 \oplus x_2} + 2(-1)^{x_1 \oplus x_2 \oplus x_3}\Big].$$

From (5) and (6) we can calculate all the possible polarities of the mixed Reed-Muller expansions (see Table 3). For different mixed polarity coefficients, from (6) we get the following Reed-Muller expansions of f:

for $\boldsymbol{S}^0$: $1 \oplus x_3 \oplus x_1 \oplus x_1x_3 \oplus x_1x_2 \oplus x_1x_2x_3$,
for $\boldsymbol{S}^1$ and $\boldsymbol{S}^2$: $\overline{x}_3 \oplus x_1\overline{x}_3 \oplus x_1x_2\overline{x}_3$,
for $\boldsymbol{S}^3$: $1 \oplus x_3 \oplus x_1\overline{x}_2 \oplus x_1\overline{x}_2x_3$,
for $\boldsymbol{S}^4$ and $\boldsymbol{S}^5$: $\overline{x}_3 \oplus x_1\overline{x}_2\overline{x}_3$,
for $\boldsymbol{S}^6$: $\overline{x}_2 \oplus \overline{x}_2x_3 \oplus x_2 \oplus x_2x_3 \oplus x_1\overline{x}_2 \oplus x_1\overline{x}_2x_3$,
for $\boldsymbol{S}^7$: $\overline{x}_2\overline{x}_3 \oplus x_2\overline{x}_3 \oplus x_1\overline{x}_2\overline{x}_3$,
⋮ ⋮
for $\boldsymbol{S}^{25}$ and $\boldsymbol{S}^{26}$: $\overline{x}_1\overline{x}_2\overline{x}_3 \oplus \overline{x}_1x_2\overline{x}_3 \oplus x_1x_2\overline{x}_3$.

♦

Property 2. *The minimal complexity of a given Boolean function f is equal to the minimal number of EXOR gates required in the realization of f.*

Property 3. *For a given Boolean function f, more than one Reed-Muller polarity vector $\boldsymbol{S}^i$ can have the same form.*

Corollary 1. (Porwik and Falkowski, 1999) *A Boolean function $f(x_1, x_2, \ldots, x_n)$ includes a redundant variable x_i if for each linear combination in which the variable x_i occurs the value of the corresponding Walsh-Hadamard spectral coefficient is equal to zero.*

The above proves that Walsh-Hadamard representations of Boolean functions are less complicated and have simpler implementations than mixed Reed-Muller representations. This is the reason why an efficient algorithm minimizing and determining the Reed-Muller form by means of the Walsh-Hadamard spectrum could be developed. The described algorithm allows us to calculate the optimal mixed Reed-Muller form directly from the Walsh spectrum, without finding all mixed Reed-Muller expansions as in the classical method.

3. Algorithm

Input Data: The input required for the analysis method consists of only the two-valued truth table vector $\boldsymbol{Y}$ of a Boolean function $f(x_1, x_2, \ldots, x_n) = f(x)$.

Output Data: A Boolean function in the Reed-Muller form.

1. Compute the Wash-Hadamard transformation matrix $\boldsymbol{T}_w(n)$.
2. Convert the truth table vector $\boldsymbol{Y} = [y_0, y_1, \ldots, y_{2^n-1}]^T$, $y_i \in \{0, 1\}$ according to the formula $g : \{0, 1\} \to \{1, -1\}$, and determine the vector of spectral coefficients $\boldsymbol{S}$.

Table 3. Reed-Muller spectrum $\boldsymbol{S}^i$ of a Boolean function f for different polarities, $i = 0, 1, \ldots, 3^n - 1$.

$x_1x_2x_3$	$f(x_1, x_2, x_3)$	Mixed polarity Reed-Muller spectrum										
		$\boldsymbol{S}^0$	$\boldsymbol{S}^1$	$\boldsymbol{S}^2$	$\boldsymbol{S}^3$	$\boldsymbol{S}^4$	$\boldsymbol{S}^5$	$\boldsymbol{S}^6$	$\boldsymbol{S}^7$	$\cdots$	$\boldsymbol{S}^{25}$	$\boldsymbol{S}^{26}$
0 0 0	1	1	0	1	1	0	1	1	0	$\cdots$	0	1
0 0 1	0	1	1	0	1	1	0	1	1	$\cdots$	1	0
0 1 0	1	0	0	0	0	0	0	1	0	$\cdots$	0	1
0 1 1	0	0	0	0	0	0	0	1	1	$\cdots$	1	0
1 0 0	0	1	0	1	0	0	0	1	0	$\cdots$	0	0
1 0 1	0	1	1	0	0	0	0	1	1	$\cdots$	0	0
1 1 0	1	1	0	1	1	0	1	0	0	$\cdots$	0	1
1 1 1	0	1	1	0	1	1	0	0	0	$\cdots$	1	0

3. Using Corollary 1 check whether f is redundant. If it is, then modify the function and go to Step 2.
4. Choose the largest (in magnitude) spectral coefficient $s_i = |s_i|$, $i = 0, 1, \ldots, 2^n - 1$. The spectral coefficient(s) with the largest magnitude(s) indicate(s) the most important input conditions that control the function output. If more than one spectral coefficient have equal magnitudes, the choice is arbitrary.
5. Realize the new function $f_s(x)$ that corresponds to the coefficient chosen in Step 4.
6. Compute the function $d(x) = f(x) \oplus f_s(x)$. The function $d(x)$ indicates the number of agreements (disagreements) between $f(x)$ and $f_s(x)$ at point x:

$$d(x) = \begin{cases} 1 & \text{for} \quad f(x) \neq f_s(x), \\ 0 & \text{for} \quad f(x) = f_s(x). \end{cases}$$

7. Combine all the intermediate realizations at points x where $d(x) = 1$ using the sum-modulo-two operator $(\oplus)$.
8. If the Hamming distance is $W[d(x)] = 1$ or $W[d(x)] = 0$, then the obtained result is optimal (minimal number of products and literals). Otherwise, we obtain the Reed-Muller form, which can be additionally minimized by means of other methods (Yanushkevich, 1998).

End of Algorithm.

Example 2. Let f be a Boolean function introduced in Example 1. At the first stage, spectral Walsh-Hadamard coefficients are calculated: $\boldsymbol{Y} = [2, -6, 2, 2, -2, -2, -2, -2]^T$. From Corollary 1 we know that the function f is not redundant. Next we find coefficients with the largest magnitudes. In our example this condition is satisfied for the coefficient $s_1 = -6$. Hence $f_s(x) = f(\overline{x}_3)$. In the next step we determine the function $d(x)$ (see Table 4). We can observe that in this function only one disagreement point appears. The terminal condition is satisfied and the remaining term can be realized directly.

The appropriate elements of Table 4, which were used for creation of the new form of the function, are marked with bold symbols. The white frames indicate the correlation measure between the function f and the appropriate row of the Hadamard matrix. The bold symbols indicate the essential values of the function $d(x)$.

Table 4. Truth table of the Boolean function $f(x)$, its spectrum, correlation measure and function $d(x)$.

$x_1x_2x_3$	x	$f(x)$	$\boldsymbol{S}$	$f_s(\bar{x}_3)$	$d(x) = f(x) \oplus f_s(x)$	Type of correlation
0 0 0	0	1	2	1	0	0
0 0 1	1	0	**−6**	0	0	$\boldsymbol{\bar{x}_3}$
0 1 0	2	1	2	1	0	x_2
0 1 1	3	0	2	0	0	$x_2 \oplus x_3$
1 0 0	4	0	−2	1	**1**	$\bar{x}_1$
1 0 1	5	0	−2	0	0	$\overline{x_1 \oplus x_3}$
1 1 0	6	1	−2	1	0	$\overline{x_1 \oplus x_2}$
1 1 1	7	0	−2	0	0	$\overline{x_1 \oplus x_2 \oplus x_3}$

Hence we construct the equation $f(x_1, x_2, x_3) = \bar{x}_3 \oplus x_1\bar{x}_2\bar{x}_3$. Note that the obtained result is consistent with the $\boldsymbol{S}^4$ Reed-Muller form. This form had already been calculated in Example 1 but now the final result was obtained faster. Unlike in the classical Reed-Muller method, this result was calculated by one pass of the algorithm. ♦

Example 3. Let f be the Boolean function described in Table 5. As in Example 2, we construct the equation $f(x_1, x_2, x_3, x_4) = x_2 \oplus x_3 \oplus \bar{x}_1\bar{x}_2x_3\bar{x}_4$. ♦

The first three cases represent one-output functions. The last four benchmarks were formed from multi-output functions by choosing the 2nd, 7th, 12th or 61st output. The experiments show that the algorithm of Section 3 is more effective than the well-known Espresso algorithm.

The differences between the Reed-Muller implementation and standard Boolean logic can be explained by means of a practical example. Table 7 shows a comparison of a 4-bit synchronous binary counter which was implemented using standard Boolean logic and the Reed-Muller form. In both realizations we used the 0.9 μm ASIC library and CMOS technology.

Table 5. Truth table of a Boolean function $f(x)$, its spectrum, correlation measure and error function $d(x)$.

$x_1x_2x_3x_4$	x	$f(x)$	$\boldsymbol{S}$	$f_s(x_2 \oplus x_3)$	$d(x) = f(x) \oplus f_s(x_2 \oplus x_3)$	Type of correlation
0 0 0 0	0	0	2	0	0	0
0 0 0 1	1	0	2	0	0	x_4
0 0 1 0	2	0	-2	1	**1**	$\bar{x}_3$
0 0 1 1	3	1	-2	1	0	$\overline{x_3 \oplus x_4}$
0 1 0 0	4	1	2	1	0	x_2
0 1 0 1	5	1	2	1	0	$x_2 \oplus x_4$
0 1 1 0	6	0	**14**	0	0	$\boldsymbol{x_2 \oplus x_3}$
0 1 1 1	7	0	-2	0	0	$\overline{x_2 \oplus x_3 \oplus x_4}$
1 0 0 0	8	0	2	0	0	x_1
1 0 0 1	9	0	2	0	0	$x_1 \oplus x_4$
1 0 1 0	10	1	-2	1	0	$\overline{x_1 \oplus x_3}$
1 0 1 1	11	1	-2	1	0	$\overline{x_1 \oplus x_3 \oplus x_4}$
1 1 0 0	12	1	2	1	0	$x_1 \oplus x_2$
1 1 0 1	13	1	2	1	0	$x_1 \oplus x_2 \oplus x_4$
1 1 1 0	14	0	-2	0	0	$\overline{x_1 \oplus x_2 \oplus x_3}$
1 1 1 1	15	0	-2	0	0	$\overline{x_1 \oplus x_2 \oplus x_3 \oplus x_4}$

4. Experimental Results and Final Remarks

The proposed approach allows us to solve the problem of the minimal (in terms of the number of products) canonical representation of a completely specified Boolean function. The results of the realization of the presented algorithm were compared with the U.C. Berkeley product Esperesso (Micheli, 1994). The Espresso package takes as the input a two-level representation of a two-valued Boolean function, and produces an equivalent minimal representation. This algorithm gives the optimal solution in heuristic Boolean minimization. It is necessary to note that the outputs of the Espresso are sums of products (SOP) and the outputs of the algorithm are exclusive sums of products (ESOP). The number of products (P) as well as the number of literals (L) (complemented or non-complemented variables) appearing in all the products P in the obtained expression are given in Table 6.

Table 6. Number of products and terms (literals) after function minimization for the presented examples and MCNC* benchmarks.

Boolean function or circuits			Espresso algorithm		Presented method	
	IN	OUT	L	P	L	P
Example 2	3	1	4	2	4	2
Example 3	4	1	8	3	6	3
xor5	5	1	80	15	5	5
rd53	5	3 (2nd)	4	4	4	2
dc1	4	7 (7th)	6	3	6	3
bw	5	28 (12th)	16	4	17	4
ex5	8	63 (61st)	19	9	18	4

* MCNC – Microelectronics Center of North Carolina

Table 7. Standard Boolean and partial Reed-Muller (R-M) implementations*.

	Optimized Boolean	Optimized partial R-M	Comparation (%)
Maximum delay (nsec)	1.713	1.758	2.6 increase
Transistors**	60	40	33 reduction
Tracks	20	12	40 reduction
Track segments	34	15	55 reduction

* From a report by C. Maxfield, EDN Magazine, USA, March 1996 (with the author's permission).

** Omitted transistors which occur in both implementations.

The XOR gates in this library are buffered pass-transistor implementations, each requiring only 10 transistors (as opposed to building the XOR gates from the AND/OR/NOT gates, which would require more transistors and would be slower). From Table 7 we can observe that the Reed-Muller implementation is only slightly slower than its Boolean counterpart but offers significant benefits in reducing the area utilization, because the implementation requires substantially fewer transistors, tracks and connections.

Additionally, as was already mentioned, the circuits built in the Reed-Muller form are very susceptible to testing. Any Reed-Muller circuit can be tested for all single stuck-at-faults with a maximum of $3n+4$ input vectors, where n is the number of primary inputs (Breuer and Friedman, 1976). The problem of test generation for digital circuits was not considered in this paper.

Besides the matrix method presented in this paper, data flow graph methods and parallel calculations similar to the Fast Fourier Transform were also used to calculate Walsh-Hadamard and Reed-Muller transforms. These methods reduce the number of necessary calculations. The classical methods of spectrum calculation (for both Reed-Muller and Walsh matrix transforms) can be described using the complexity criterion:

$$C(\boldsymbol{S},\boldsymbol{T}) = k^{2n} + k^n(k^n-1),$$

where k is the valence of f (for a two-valued Boolean function we have $k=2$), n denotes the number of arguments of the Boolean function, k^{2n} stands for the number of multiplications, and $k^n(k^n-1)$ denotes the number of additions.

Hence the complexity criterion for the Reed-Muller expansions of all mixed polarities is $3^nC(\boldsymbol{S},\boldsymbol{T})$. Next we must choose the shortest expansion. For the Walsh spectrum we get only the complexity $C(\boldsymbol{S},\boldsymbol{T})$. The solution is computed immediately and very often has the shortest representation.

Notice that the data flow graph methods and parallel calculations have also been used to calculate the Walsh-Hadamard and Reed-Muller transforms. These methods reduce the number of necessary calculations. Nowadays the computational process in either the Reed-Muller or the Walsh domain is based on the well-known butterfly signal flow graph configuration. In this case computational complexity can be described by $C_{\min}(\boldsymbol{S},\boldsymbol{T}) = nk^n$. This means that we execute addition operations only. Clearly, in this situation for all mixed Reed-Muller expansions we obtain the complexity $3^nC_{\min}(\boldsymbol{S},\boldsymbol{T})$.

References

Breuer M.A. and Friedman A.D. (1976): *Diagnosis and Reliable Design of Digital Systems*. — Potomac, Maryland: Computer Science Press.

Damarla T.R. and Karpovsky M. (1989): *Fault detection in combinational networks by Reed-Muller transforms*. — IEEE Trans. Comp., Vol. 38, No. 6, pp. 788–797.

Falkowski B.J. and Chang C.H. (2000): *Minimisation of k-variable mixed-polarity Reed-Muller expansions*. — VLSI Design, Vol. 11, No. 4, pp. 311–320.

Falkowski B.J. and Chang C.H. (1995): *An exact minimizer of fixed polarity Reed-Muller expansion*. — Int. J. Electron., Vol. 79, No. 3, pp. 389–409.

Falkowski B.J. and Chang C.H. (1999): *Hadamard-Walsh spectral characterization of Reed-Muller expansions*. — Comp. Electr. Eng., No. 25, pp. 111–134.

Falkowski B.J. and Porwik P. (1999): *Evaluation of nonlinearity in Boolean functions by extended Walsh-Hadamard transform*. — Proc. 2nd Int. Conf. *Information Communications and Signal Processing, ICISC'99*, Singapore, paper 2B2.2, pp. 1–4.

Giani A., Sheng S., Hsiao M.S. and Agraval V.D. (2001): *Efficient spectral techniques for sequential ATPG*. — Proc. Int. Conf. IEEE *Design Automation & Test in Europe*, Munich, Germany, pp. 204–208.

Hurst S.L., Miller D.M. and Muzio J.C. (1985): *Spectral Techniques in Digital Logic*. — London: Academic Press.

Karpovsky M.G. (1976): *Finite Orthogonal Series in the Design of Digital Devices*. — New York: Wiley.

Karpovsky M.G. (1985): *Spectral Techniques and Fault Detection*. — London: Academic Press.

Micheli De G. (1994): *Synthesis and Optimization of Digital Circuits*. — Boston: McGraw-Hill.

Perkowski M. (1996): *A unified approach to exor-based representation of Boolean functions*. — Proc. XIX Nat. Conf. *Circuit Theory and Electronics Circuits*, Krynica, Poland, Vol. 1, pp. 27–41.

Porwik P. (1996): *Fault path detection using a spectral method*. — Proc. Int. *Baltic Electronic Conference, BEC'96*, Tallin, Estonia, pp. 315–318.

Porwik P. and Falkowski B.J. (1999): *Informatics properties of the Walsh transform*. — Proc. 2-nd Int. Conf. *Information Communications and Signal Processing, ICISC'99*, Singapore, paper 2B2.4, pp. 1–5.

Sasao T. (1993): *Logic Synthesis and Optimalization*. — Dordrecht: Kluwer.

Sasao T. (1995): *Representation of logic functions using EXOR operators*. — Proc. Workshop *Applications of the Read-Muller Expansion in Circuit Design*, Makuhari, Japan, pp. 308–313.

Yanushkevich S. (1998): *Logic Differential Calculus in Multi-Valued Logic Design*. — Szczecin, Poland: Technical University Press.

Received: 6 December 2001
Revised: 16 May 2002