A Hybrid Method for Spectral Translation Equivalent Boolean Functions

Mathias Soeken, Eleonora Testa
École Polytechnic Fédérale de Lausanne, Switzerland
Email: mathias.soeken,elonora.testa@epfl.ch

D. Michael Miller
University of Victoria, Canada
Email: mmiller@uvic.ca

***Abstract*—The equivalence of Boolean functions with respect to five invariance (aka translation) operations has been well considered with respect to the Rademacher-Walsh spectral domain. In this paper, we introduce a hybrid approach that uses both the Reed-Muller and the Rademacher-Walsh spectra. A novel hybrid algorithm that maps a Boolean function to a representative function for the equivalence class containing the original function is presented. The algorithm can be used to determine a sequence of translations that maps one function to an equivalent function. We present experimental results that show the hybrid algorithm can determine the equivalence classes for 5 variables much more efficiently than before. We also show that for 6 variables where there are 150,357 equivalence classes, 8 are very difficult, a further 58 are difficult and the remainder are straightforward in terms of the CPU time required by the hybrid algorithm.**

I. Introduction

The equivalence and classification of Boolean functions has been a topic of interest for some time. Two Boolean functions are equivalent with respect to a particular class of transformations if there is a sequence of translations that maps one function to the other. Function equivalence partitions the Boolean functions for a particular number of variables into equivalence classes. Such classes are of interest since, for example, if one has an inexpensive implementation for one function f in the class, a sequence of translations to map g to f provides a potentially efficient implementation for g. Finding the least costly translation sequence affects the overall cost.

NPN equivalence which allows for negation of inputs, permutation of inputs and negation of the function, was considered in 1963 by Harrison [1]. NPN equivalence has been applied in technology mapping [2] and a variety of other applications in logic design.

A Boolean function can be transformed to the Rademacher-Walsh (RW) spectral domain [2]–[7]. Unlike the functional domain where the individual 2^n function values provide local information, each of the 2^n integer-valued spectral coefficients provide global information about the function. Two XOR-based spectral translations added to the NPN operations allow for the spectral classification of Boolean functions where the number of equivalence classes is far smaller than for NPN equivalence [5]–[8]. This classification was termed *restricted affine equivalence* by R. J. Lechner in [9].

As an alternative to working in the RW domain, a Boolean function can be transformed to the Reed-Muller spectral domain [10] where different information is highlighted in comparison to the functional and RW domains. In this paper, we present a novel hybrid spectral algorithm which works in both the RM and RW domains to transform a Boolean function f to the unique representative function for the equivalence class that contains f. We validate the algorithm by generating representative functions for the spectral equivalence classes for $1 \leq n \leq 5$ variables using function neighbourhood searching as introduced by Fuller [11]. The results show the hybrid approach is much more efficient than the RW approach presented by two of the authors in [12].

Determining function equivalence and enumerating the spectral equivalence classes for more than 5 variables are very difficult computational tasks. There are 2^{64} Boolean functions of 6 variables which are known to partition into 150,357 equivalence classes [13] . By way of further verifying the hybrid algorithm we have applied it to 150,357 functions, one per equivalence class, extracted from the data base available at [14]. The results show the efficiency of the hybrid approach except for a small number of cases.

The rest of the paper is organized as follows. Section II provides the necessary background for the paper particularly with respect to the RM and RW spectra. The new hybrid algorithm is described in Section III and its application with respect to spectral equivalence classes is examined in Section IV. Experimental results are shown in Section V and Section VI provides observations and suggestions for ongoing research. A description of the RW TRANSFORM algorithm employed by the new hybrid algorithm can be found in [12].

II. Background

A. Spectra of Boolean Functions

Definition 1: An n-input *Boolean function* is a mapping $f : \mathbb{B}^n \rightarrow \mathbb{B}$ where $\mathbb{B} = \{0, 1\}$. The input variables are $x_1, \ldots, x_n$ and the function is denoted $f(x_1, x_2, \ldots, x_n)$.

A Boolean function f can be represented by a "truth" (column) vector with 2^n entries. In this work, we use both $\{0, 1\}$ coding, denoted F, and the so-called $\{1, -1\}$ coding [7], denoted F^*, where 1 denotes logic-0 and -1 denotes logic-1. For example, the AND function for 3 variables has

$$F = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^t \tag{1}$$

which in $\{1, -1\}$ coding becomes

$$F^* = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \end{bmatrix}^t \tag{2}$$

Note, that column vectors are written as transposed row vectors for space considerations.

Definition 2: The *Reed-Muller (RM) spectrum* of a Boolean function is given by

$$R = M^n F \tag{3}$$

where

$$M^n = \begin{bmatrix} M^{n-1} & 0 \\ M^{n-1} & M^{n-1} \end{bmatrix},\ M^0 = \begin{bmatrix} 1 \end{bmatrix} \tag{4}$$

The arithmetic is over $GF(2)$.

For example, for F given in (1), the RM spectrum is

$$R = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^t \tag{5}$$

A Boolean function $f(X), X = \{x_1, x_2, \ldots, x_n\}$ can be written in the form

$$f(X) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus a_3 x_1 x_2 \oplus \cdots \oplus a_{2^n-1} x_2 x_2 \ldots x_n \tag{6}$$

where $\forall a_\alpha \in \{0, 1\}$. This is the algebraic normal form of the function and corresponds directly to the positive polarity Reed-Muller expansion of the function, The a_α in (6) are precisely the Reed-Muller spectral coefficients denoted

$$R = \begin{bmatrix} r_0, r_1, r_2, r_{12}, r_3, \ldots, r_{1,2,\ldots,n} \end{bmatrix}^t \tag{7}$$

Definition 3: The *Rademacher-Walsh (RW) spectrum*, in Hadamard order, of a Boolean function is given by

$$S = T^n F^* \tag{8}$$

where

$$T^n = \begin{bmatrix} T^{n-1} & T^{n-1} \\ T^{n-1} & -T^{n-1} \end{bmatrix},\ T^0 = \begin{bmatrix} 1 \end{bmatrix} \tag{9}$$

Given $[1, -1]$ coding, each row of the RW transform matrix represents a function which is the XOR of a subset of the variables $x_1, x_2, \ldots, x_n$. Note that the top row denotes the constant 0 function, *i.e.* the XOR of no variables. The elements of S are for this reason, identified by the variables involved in the corresponding XOR function where, as is standard notation, 0 denotes the empty set *i.e.* no variables. For example for $n = 3$,

$$S = \begin{bmatrix} s_0\, s_1\, s_2\, s_{12}\, s_3\, s_{13}\, s_{23}\, s_{123} \end{bmatrix}^t. \tag{10}$$

For example, for F^* given in (2), the RW spectrum is

$$S = \begin{bmatrix} 6 & 2 & 2 & -2 & 2 & -2 & -2 & 2 \end{bmatrix}^t \tag{11}$$

Each RW spectral coefficient can be seen to measure the correlation of f and the XOR function corresponding to the coefficient. A value of 2^n indicates perfect correlation i.e. f is the XOR function, whereas a value of -2^n indicates perfect correlation to the inverse of the XOR function.

Note that for clarity we are using R for the RM spectrum and S for the RW spectrum. Readers are cautioned that in many references R has a different interpretation as it is used for a RW spectrum computed using a function vector in $\{0, 1\}$ coding.

It is readily verified that $(M^n)^{-1} = M^n$ over $GF(2)$ and that $(T^n)^{-1} = \frac{1}{2^n} T^n$. As a consequence, we observe that the RM and RW spectra of a Boolean function are unique.

We have presented the calculation of spectra as a matrix multiplication for clarity. In practice, fast transform techniques [7] are used and the computational complexity is $O(n2^n)$.

B. Coefficient Ordering

The coefficients in equations (7,10) are in 'natural' order. In this work we use an alternative ordering [6], which we term *index ordered*, which groups the coefficients by the number of variables associated with the coefficient. For example, for $n = 3$, this order is

$$R = \begin{bmatrix} r_0 | r_1\, r_2\, r_3 | r_{12}\, r_{13}\, r_{23} | r_{123} \end{bmatrix}^t \tag{12}$$

and similarly for S. Unless otherwise specified, spectra will be considered to be index ordered which in general is

$$\begin{aligned} R = [r_0 | r_1\, r_2\, r_3\ \ldots\ r_n | r_{12}\, r_{13}\, r_{23}\, r_{14}\, r_{24}\, r_{34}\ \ldots\ r_{(n-1)n} | \\ r_{123}\, r_{124}\, r_{134}\, r_{234}\ \ldots\ r_{(n-2)(n-1)n} | \ldots | r_{12\ldots n}]^t \end{aligned} \tag{13}$$

and similarly for S.

As can be seen, the coefficients are ordered with the 0-order coefficient first, followed by the first-order, second-order, third–order coefficients and so on where the order means the number of indices with 0 denoting the empty set. Note that within each group the coefficients are arranged by increasing values of their indices.

C. Spectral Translations

Five translations of Boolean function spectra are of interest in this work. Given a Boolean function $f(x_1, x_2, \ldots, x_n)$ with RM spectrum R and RW spectrum S, the translations are defined as shown in Table I. The RW translations shown have been well documented [6]–[8]. Below, we detail the RM translations.

Type 1: Interchanging two variables essentially repositions them in the spectrum. The resulting operation on the spectral coefficients is the same in the RM domain as it is in the RW domain.

Type 2: The negation of x_i can be expressed as $1 \oplus x_i$. Hence any term $x_i P$, where P is a product of variables excluding x_i is replaced by $x_i P \oplus P$. In terms of the RM coefficients, let α denote the variables indices for P, then if $r_{i\alpha} = 1$, r_α is inverted i.e. the corresponding term is either added, or removed from the function if it was already present. This follows from the fact $x \oplus x = 0$ for any expression x.

Type 3: It is clear from (6) that inverting r_0 complements the function.

Type 4: The operation $x_i \leftarrow x_i \oplus x_j$ is quite similar to variable negation. The difference is that $r_{i\beta} = 1$, the generated term corresponds to $r_{j\beta}$ and again that term is added/removed depending on whether it was not/was already present.

Type 5: The operation $f(X) \leftarrow f(X) \oplus x_i$ clearly inverts r_i since if the term x_i was not present it is added and if it was present it is removed since $x_i \oplus x_i = 0$.

Applying the above translations leads to the following definition of Boolean function equivalence.

Definition 4: Two Boolean functions $f(x_1, x_2, \ldots, x_n)$ and $g(x_1, x_2, \ldots, x_n)$ are *equivalent* with respect to the 5 spectral

TABLE I
SPECTRAL TRANSLATIONS

Type	Functional Operation	RM Domain	RW Domain
1	Interchange x_i and x_j	$r_{i\alpha} \leftrightarrow r_{j\alpha} \forall \alpha$	$s_{i\alpha} \leftrightarrow s_{j\alpha} \forall \alpha$
2	Negation of the input variables x_i	invert $r_\beta \forall r_{i\beta} = 1$	negate all $r_{i\beta} \forall \beta$
3	Negation of $f(X)$	invert r_0	negate all RW coefficients
4	Replacement of x_i by $x_i \oplus x_j$	invert all $r_{j\beta} \forall r_{i\beta} = 1$	$s_{i\alpha} \leftrightarrow s_{ij\alpha} \forall \alpha$
5	Replacement of $f(X)$ by $f(X) \oplus x_i$	invert r_i	$s_{i\alpha} \leftrightarrow s_\alpha \forall \alpha$

α denotes any set of variable indices excluding i and j including the empty set
β denotes any set of variable indices excluding i including the empty set; β may include j

translations if f can be transformed into g by the application of some sequence of those translations. Note that the translations are all self-inverse, so the reverse sequence of translations will transform g to f. Also as shown in the definitions of the translations they can be directly carried out in either spectral domain, *i.e.* it is straightforward to transform between the spectra of f and g.

It is clear from the above that the 5 translations partition the set of Boolean functions into equivalence classes. The following definitions apply.

Definition 5: Given two Boolean functions $f(x_1, x_2, \ldots, x_n)$ and $g(x_1, x_2, \ldots, x_n)$ with spectra S_f and S_g respectively, we say f *precedes* g, denoted $f \prec g$ if for the first coefficient position (in index order) for which the coefficients from S_f and S_g differ, the coefficient from S_f has larger magnitude, or if the two coefficients have the same magnitude, the coefficient from S_f is positive. Note that for convenience we will also write $S_f \prec S_g$.

Definition 6: Clearly, a function equivalence class must contain a function f^R that precedes every other function in the class. We term f^R the *representative function* for the class.

In some applications the cost of a sequence of translations will be important. A straightforward cost model is:

- Translation 1 interchanges 2 variables and requires a swap gate which can be implemented using 3 XOR gates. We thus assume a cost of 3.
- Translations 2 and 3 each require a single NOT gate to implement an inversion and we use a cost of 1.
- Translations 4 and 5 each require a single XOR gate and again we use a cost of 1.

Note that alternative cost models including a model where the cost of a translation varies by context can be used. For reversible or quantum circuits the XOR gates mentioned above are implemented as controlled-NOT (CNOT) gates [15].

III. TRANSFORMING A BOOLEAN FUNCTION TO THE REPRESENTATIVE FUNCTION

In this section, we present the new hybrid spectral algorithm to map a Boolean function f to the representative function f^R for the equivalence class that contains f. In doing so, the algorithm identifies a sequence of translations to map f into f^R.

In [12], two of the current authors presented an RW spectrum based algorithm for this mapping problem. The work here uses procedures TRANSFORM and CLOSER from that work. The main contribution of this paper is to show how by first considering the RM spectrum of a function, we can significantly improve upon the efficiency of the approach described in [12].

The following definition which applies to RM spectra is a key concept in the hybrid algorithm.

Definition 7: An RM spectrum is termed *disjoint* if the index sets for all coefficients equal to 1 are disjoint, i.e. each variable index appears in at most one coefficient with value 1. An RM spectrum with 0 or 1 nonzero coefficient is clearly disjoint.

The new hybrid algorithm is shown below as Algorithm 1. The basic idea of the RM component of the algorithm is to remove coefficients with value 1 using a type 3 translation for r_0, a type 5 translation for first order coefficients and type 2 or 4 translations for other coefficients. The ordering prefers type 2 to type 4 translations.

The functions RMtrans1 to RMtrans5 perform the appropriate spectral translation in the RM domain. The sort in line 26 of the hybrid algorithm is a simple selection sort of the variables $x_1, x_2, \ldots, x_n$ so that the sorted spectrum S is such that for each $i < j$, there is no α such that $|s_{\alpha,i}| < |s_{\alpha,j}|$ or, $|s_{\alpha,i}| = |s_{\alpha,j}|$ and $s_{\alpha,i} < 0$. In other words, there is no reordering of the variables that will map S to a spectrum that precedes S. The sort applies type 1 translations when two variables have to be reordered.

IV. FINDING SPECTRAL EQUIVALENCE CLASSES

Given an algorithm such as the one above to transform f to f^R, it is possible to find the equivalence classes for n variables, at least for small n, by applying it to all 2^{2^n} functions and keeping track of the unique f^R encountered. However, that approach becomes prohibitive even for $n = 5$. Here we use an alternate approach based on Algorithm 4.2.1 from J. E. Fuller's PhD thesis [11]. We will show this approach is quite efficient. Its limitation is that it does not directly yield information on the size of each equivalence class.

Fuller's algorithm is a search based on the concept of the 1-neighbourhood of a Boolean function defined as follows:

Definition 8: A function g is in the *1*-neighbourhood of function f, if g differs from f for precisely one input assignment.

Our implementation of Fuller's method is given in Algorithm 2. The result of executing this procedure is the set of representative functions in $list$.

V. EXPERIMENTAL RESULTS

Our program is written in C and was compiled using gcc with full execution speed optimization (-O3). Our experiments were run on a computer with an Intel i5 650 processor @ 3.2GHz and 3 GB of memory.

Algorithm 1:

procedure HYBRID(R, n)
  for each $r_\gamma = 1 \in \{r_{12}, r_{13}, \ldots, r_{12\ldots n}\}$ **do**
    find the lowest i such that $r_{\gamma i} = 1$
    if such an i exists **then**
      apply $RMtrans2(R, n, i)$
    else
      find the first $r_\delta == 1$ following r_γ
        in index order such that $|\gamma \cap \delta| = 1$
      if such an r_δ exists **then**
        set $i = \gamma - \gamma \cap \delta$
        set j to the lowest value in $\delta - \gamma \cap \delta$
        $RMtrans4(R, n, i, j)$
      end if
    end if
  end for
  if $r_0 = 1$ **then**
    $RMtrans3(R, n)$
  end if
  for $i = 1, 2, \ldots, n$ **do**
    if $r_i = 1$ **then**
      $RMtrans5(R, n, i)$
    end if
  end for
  determine an RW spectrum S from R
  if R is disjoint **then**
    $sort(S, n)$
  else
    $transform(S, n, 0)$
  end if
  convert S back to an RM spectrum R
end procedure

Algorithm 2:

procedure FULLER_SEARCH(n)
  $f \leftarrow$ constant 0 function of n variables
  $list \leftarrow hybrid(f)$
  $p \leftarrow 0$
  while $p < length(list)$ **do**
    for each g in the 1-neighbourhood of $list[p]$ **do**
      $h \leftarrow hybrid(g)$
      if h not in $list$ **then**
        add h to the end of $list$
      end if
    end for
    $p \leftarrow p + 1$
  end while
end procedure

For our first experiment we used the new hybrid algorithm and a Fuller search to find spectral equivalence classes. The representative functions determined for $n = 1, 2, 3, 4$ are shown in Table III and for $n = 5$ are shown in Table V. Each representative functions f^R is coded as a decimal number in the conventional way where the least significant bit of the equivalent binary number is the function value when all variables are assigned 0.

Table II shows statistics for these tests including CPU utilization which was determined using the clock function from the library time.h. The results clearly show the advantage of the new hybrid RM-RW approach as applying the program with the RM techniques excluded, i.e. just using TRANSFORM from [12], requires 0.10147 CPU sec. for $n = 4$ and 110.25 CPU sec. for the $n = 5$ case. The speedups are approximately 20 and 19 for $n = 4$ and 5, respectively. Note that the timings for $n < 4$ are too short for a meaningful comparison due to the resolution of the clock function.

TABLE II
SPECTRAL EQUIVALENCE CLASS EXPERIMENT

n	Functions Considered	Number of Classes	CPU sec.
1	3	1	0.000
2	9	2	0.001
3	25	3	0.003
4	129	8	0.005
5	1537	48	5.413

We next consider $n = 6$. It is known that there are 150,357 spectral equivalence classes. A Fuller search would require consideration of $150,357 \times 2^6 + 1 = 9,622,849$ functions. If a function could on average be handled in about 0.025 CPU sec., this would take on the order of 2.8 CPU days, and as the results below show 0.025 is an overly optimistic figure.

By way of evaluating the hybrid algorithm for $n = 6$ we have applied it to 150,357 functions, one per equivalence class, extracted from the data base available at [14]. Note that these functions are not the representative functions for the classes as used in this work. Except for 3 cases the hybrid algorithm correctly maps each of these functions to a unique representative function. Those 3 cases timed out at 2 CPU hours per case.

Definition 9: [16] A Boolean function is *bent* if it is as distant as possible from all affine functions. As a result, the Rademacher-Walsh spectral coefficients of an n-variable bent function all have the same magnitude of $2^{n/2}$. Note that a Boolean bent function must have an even number of variables.

The three cases noted above that timed out are bent functions and thus have flat spectra which is the worst possible situation as TRANSFORM undertakes a fully exhaustive search.

Five further functions were found to require more than 10 CPU minutes. In 4 of these cases, the RW spectrum sent to TRANSFORM has 2 coefficient magnitudes and in the fifth it has 3. Again, these situations require extensive searching by the TRANSFORM procedure. The 8 very difficult cases are enumerated in Table IV.

A further 58 functions were found to each take more than 2 CPU minutes. Of the remaining 150,291, only 9 functions required more than 1 CPU minute. For the remaining 150,282 functions that require less than 1 minute of CPU time, the average CPU use is 0.026 seconds.

VI. CONCLUSION

A novel hybrid Reed-Muller and Rademacher-Walsh spectra-based algorithm hat maps a Boolean function to a

TABLE III
SPECTRAL EQUIVALENCE CLASSES $n = 1, 2, 3, 4$

class $n=1$	f^R	spectrum 0	1														
1	0	2	0														
$n=2$		0	1	2	12												
1	0	4	0	0	0												
2	8	2	2	2	-2												
$n=3$		0	1	2	3	12	13	23	123								
1	0	8	0	0	0	0	0	0	0								
2	128	6	2	2	2	-2	-2	-2	2								
3	136	4	4	4	0	-4	0	0	0								
$n=4$		0	1	2	3	4	12	13	23	14	24	34	123	124	134	234	1234
1	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	27328	4	4	4	4	4	4	4	-4	-4	4	4	-4	-4	-4	-4	4
3	32768	14	2	2	2	2	-2	-2	-2	-2	-2	-2	2	2	2	2	-2
4	32896	12	4	4	4	0	-4	-4	-4	0	0	0	4	0	0	0	0
5	34944	10	6	6	2	2	-6	-2	-2	-2	-2	2	2	2	-2	-2	2
6	34952	8	8	8	0	0	-8	0	0	0	0	0	0	0	0	0	0
7	43136	8	8	4	4	4	-4	-4	0	-4	0	0	0	0	0	-4	4
8	59520	6	6	6	6	6	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	6

TABLE IV
VERY DIFFICULT FUNCTIONS FOR $n = 6$

Function from [14]	Representative Function	CPU sec.
4ed236aae4789c00		timed out
16e6da2abc4c7080		timed out
b2e42e78ca9c5600		timed out
10092a00853640c0	544ac820c2800220	1774.3
ced236aae4789c00	1eee72287448e880	1421.4
96e6da2abc4c7080	166e7aa87cc8e000	1414.3
16665aaa3cccf000	566a6aa9fcc0c002	1412.9
32e42e78ca9c5600	3e6ef20854c868a0	1409.6

The time out was set at 7200 CPU sec. (2 hours).

representative function for the spectral translation equivalence class containing the original function has been presented. Experimental results show the efficiency of the algorithm for generating the representative functions for $n \leq 5$ and, in particular, that it is much more efficient than using the RW spectrum alone. For $n = 6$, our results show the effectiveness of the approach except for a relatively small number of cases. Our ongoing work will analyze those cases to see if they have particular properties in the functional, RM or RW domains that will make it possible to handle them more efficiently.

We note that the autocorrelation coefficients [5] are the same for all three cases, so do not provide any insight. Also, the number of positive and negative RW coefficients is the same in each case, 38 and 28 respectively, and each has 16 nonzero RM coefficients. So simple counting arguments that most often distinguish one spectral class from another do not work in this case. It is the distribution of the values that must be considered.

In terms of applications, we plan to consider incorporating the hybrid algorithm into the rewriting system in [17].

Our experiments finding representative functions for the equivalence classes were designed to help verify the validity of the hybrid approach. We used Fuller's neighbourhood search and demonstrated the hybrid approach introduced here is quite efficient for $n \leq 5$. Since that search does not examine all functions of n variables, it does not determine the size of the equivalence classes. That is not a concern in our ongoing work as we are primarily concerned with function equivalence applied in problems such as synthesis where the main question is whether two functions are equivalent and mapping of a given function to a standard implementation for its equivalence class.

The work here has concentrated on spectral translation equivalence using all 5 translations. As shown in [12], it is possible to consider NPN, linear or affine function classification by restricting the translations considered.

REFERENCES

[1] M. A. Harrison, *Introduction to Switching and Automata Theory*. New York, USA: McGraw Hill, 1963.

[2] M. A. Thornton, R. Drechsler, and D. M. Miller, *Spectral Techniques in VLSI CAD*. Boston, USA: Kluwer Academic Publishers, 2001.

[3] H. Rademacher, "Einige Sätze über Reihen von allgemeinen Orthogonalfunktionen," *Math. Annen*, vol. 87, pp. 112–138, 1922.

[4] J. L. Walsh, "A closed set of orthogonal functions," *Am. J. Math*, vol. 45, pp. 5–24, 1923.

[5] M. G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*. New York, USA: John Wiley & Sons, 1976.

[6] S. L. Hurst, *The Logical Processing of Digital Signals*. London, UK: Arnold, 1978.

[7] S. L. Hurst, D. M. Miller, and J. C. Muzio, *Spectral Techniques in Digital Logic*. London, UK: Academic Press, 1985.

[8] C. R. Edwards, "The application of the Rademacher-Walsh transform to Boolean function classification and threshold logic synthesis," *IEEE Trans. on Computers*, vol. 24, no. 1, pp. 48–62, 1975. [Online]. Available: http://dx.doi.org/10.1109/T-C.1975.224082

[9] R. J. Lechner, "Harmonic analysis of switching functions," in *Recent Development in Switching Theory*, A. Mukhopadhyay, Ed. Academic Press, 1971.

[10] D. H. Green, *Modern Logic Design*. Addison Wesley, 1986.

[11] J. E. Fuller, "Analysis of affine equivalent Boolean functions for cryptography," Ph.D. dissertation, Queensland University of Technology, 2003.

[12] D. M. Miller and M. Soeken, "An algorithm for linear, affine and spectral classification of Boolean functions," in *Advanced Boolean Techniques*, R. Drechsler and M. Soeken, Eds. Springer, 2019.

[13] J. A. Maiorana, "A classification of the cosets of the Reed-Muller code $r(1, 6)$," *Mathematics of Computation*, vol. 57(195), pp. 403–414, 1991.

[14] "https://github.com/usnistgov/circuits/," 2018.

[15] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[16] O. Rothaus, "On "bent" functions," *J. of Combinatorial THeory(A)*, vol. 20, pp. 300–305, 1976.

[17] E. Testa, M. Soeken, L. Amaru, and G. De Micheli, "Reducing the multiplicative complexity in logic networks for cryptography and security applications," in *DAC*, 2019.

TABLE V
SPECTRAL EQUIVALENCE CLASSES $n = 5$

class	f^R	spectrum 0	1	2	3	4	5	12	13	23	14	24	34	15	25	35	45	123	124	134	234	125	135	235	145	245	345	1234	1235	1245	1345	2345	12345
1	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	680042496	20	12	4	4	4	4	-4	-4	4	-4	4	-4	-4	-4	4	4	-4	-4	4	-4	4	-4	-4	-4	-4	-4	4	4	4	4	4	-4
3	1790995136	8	8	8	8	8	0	8	8	-8	-8	8	8	0	0	0	0	-8	-8	-8	-8	0	0	0	0	0	0	8	0	0	0	0	0
4	1791549440	12	8	8	8	8	8	4	4	-4	-4	4	-4	-4	-4	4	4	-8	-8	0	0	0	-8	0	0	-8	-8	4	4	4	4	4	0
5	1791550016	8	8	8	8	8	4	8	8	-8	-8	8	0	-4	-4	4	4	-8	-8	0	0	4	-4	-4	-4	-4	-4	0	4	4	4	4	-4
6	2022213768	12	12	12	4	4	4	-12	4	4	4	4	-4	4	4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	4	4	4	4	4	4	-4
7	2147483648	30	2	2	2	2	2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	2	2	2	2	2	2	2	2	2	2	-2	-2	-2	-2	-2	2
8	2147516416	28	4	4	4	4	0	-4	-4	-4	-4	-4	-4	0	0	0	0	4	4	4	4	0	0	0	0	0	0	-4	0	0	0	0	0
9	2155905024	26	6	6	6	2	2	-6	-6	-6	-2	-2	-2	-2	-2	-2	2	6	2	2	2	2	2	2	-2	-2	-2	-2	-2	2	2	2	-2
10	2155905152	24	8	8	8	0	0	-8	-8	-8	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	2290122752	24	8	8	4	4	4	-8	-4	-4	-4	-4	0	-4	-4	0	0	4	4	0	0	4	0	0	0	0	-4	0	0	0	4	4	-4
12	2290122880	22	10	10	6	2	2	-10	-6	-6	-2	-2	2	-2	-2	2	-2	6	2	-2	-2	2	-2	-2	2	2	-2	2	2	-2	2	2	-2
13	2290124928	20	12	12	4	4	0	-12	-4	-4	-4	-4	4	0	0	0	0	4	4	-4	-4	0	0	0	0	0	0	4	0	0	0	0	0
14	2290649216	18	14	14	2	2	2	-14	-2	-2	-2	-2	2	-2	-2	2	2	2	2	-2	-2	2	-2	-2	-2	-2	2	2	2	2	-2	-2	2
15	2290649224	16	16	16	0	0	0	-16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	2826993664	22	10	6	6	6	6	-6	-6	-2	-6	-2	-2	-6	-2	-2	-2	2	2	2	-2	2	2	-2	2	-2	-2	2	2	2	2	6	-6
17	2826993792	20	12	8	8	4	4	-8	-8	-4	-4	0	0	-4	0	0	-4	4	0	0	-4	0	0	-4	4	0	0	4	4	0	0	4	-4
18	2826995840	18	14	10	6	6	2	-10	-6	-2	-6	-2	2	-2	2	-2	-2	2	2	-2	-6	-2	2	-2	2	-2	2	6	2	2	-2	2	-2
19	2827004032	16	16	8	8	8	0	-8	-8	0	-8	0	0	0	0	0	0	0	0	0	-8	0	0	0	0	0	0	8	0	0	0	0	0
20	2827520128	16	16	12	4	4	4	-12	-4	0	-4	0	0	-4	0	0	0	0	0	0	-4	0	0	-4	0	-4	4	4	4	4	-4	0	0
21	2827526272	16	16	8	8	4	4	-8	-8	0	-4	4	-4	-4	-4	4	0	0	-4	4	-4	4	-4	-4	0	0	0	4	4	0	0	0	0
22	2833303552	14	10	10	10	6	6	-2	-2	-2	-6	-6	2	-6	2	-6	6	-6	6	-2	-2	-2	6	-2	2	-6	-6	2	2	-2	-2	6	2
23	2860548224	18	14	6	6	6	6	-6	-6	-6	-6	2	2	-6	2	2	-6	6	-2	-2	-2	-2	-2	-2	6	-2	-2	2	2	2	2	2	-2
24	3095701632	12	12	8	8	8	4	-8	-8	4	-8	4	-4	4	0	0	0	4	4	-4	-8	0	0	-4	0	-4	4	8	-4	-4	4	0	0
25	3163070592	12	12	8	8	8	8	-8	-8	4	0	4	-4	0	-4	4	-4	4	-4	4	-8	4	-4	-8	-4	0	0	0	0	0	0	4	4
26	3769157760	12	12	12	12	4	4	-4	-4	-4	4	-4	-4	-4	4	-4	4	-4	-4	-4	4	-4	4	-4	-4	-4	4	4	4	4	-4	-4	4
27	3783297152	16	8	8	8	8	4	-8	-8	-8	0	0	0	4	4	-4	-4	0	0	0	0	-4	4	4	-4	-4	4	8	4	4	-4	-4	-4
28	3900735488	20	8	8	8	8	8	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	0	0	0	0	0	0	0	0	0	0	4	4	4	4	4	-8
29	3900735490	18	10	6	6	6	6	-2	-2	-6	-2	-6	-6	-2	-6	-6	-6	2	2	2	-2	2	2	-2	2	-2	-2	6	6	6	6	2	-6
30	3900735616	18	10	10	10	6	6	-6	-6	-6	-2	-2	-2	-2	-2	-2	-6	2	-2	-2	-2	-2	-2	-2	2	2	2	6	6	2	2	2	-6
31	3900737664	16	12	12	8	8	4	-8	-4	-4	-4	-4	0	0	0	-4	-4	0	0	-4	-4	-4	0	0	0	0	4	8	4	4	0	0	-4
32	3900745856	14	14	10	10	10	2	-6	-6	-2	-6	-2	-2	2	-2	-2	-2	-2	-2	-2	-6	-2	-2	2	-2	2	2	10	2	2	2	-2	-2
33	3900762240	12	12	12	12	12	0	-4	-4	-4	-4	-4	-4	0	0	0	0	-4	-4	-4	-4	0	0	0	0	0	0	12	0	0	0	0	0
34	3900803200	14	10	10	6	6	6	-10	-6	-6	-6	-6	-2	2	2	-2	-2	-2	-2	-6	-6	-2	2	2	2	2	6	6	6	6	2	2	-2
35	3900827776	10	10	10	10	10	2	-6	-6	-6	-6	-6	-6	2	2	2	2	-6	-6	-6	-6	2	2	2	2	2	2	10	2	2	2	2	2
36	3901261952	14	14	14	6	6	6	-10	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	6	6	6	6	-2	-2	-2
37	3902834816	14	14	10	10	6	6	-6	-6	-2	-2	-6	2	-2	2	-6	-2	-2	2	-6	-2	-6	2	-2	-2	2	2	6	6	2	2	-2	-2
38	3904940160	12	12	12	12	8	4	-4	-4	-4	-8	0	0	4	-4	-4	0	-4	0	0	-8	-4	-4	4	0	0	0	8	4	0	0	0	0
39	3917512832	16	8	8	8	8	8	-8	-8	-8	0	0	0	0	0	0	-8	0	0	0	0	0	0	0	0	0	0	8	8	0	0	0	-8
40	3919626376	10	10	10	10	6	6	-6	-6	-6	6	-2	-2	-2	6	-2	-6	-6	-2	-2	6	-2	6	-2	2	2	-6	6	6	-6	2	2	-6
41	3921717376	10	10	10	10	10	6	-6	-6	-6	-6	2	2	6	-2	-2	-2	-6	2	2	-6	-2	-2	6	-2	-2	-2	10	6	-2	-2	-2	-2
42	3934290048	16	12	8	8	8	8	-4	-4	-8	-4	0	0	-4	0	0	-8	4	-4	-4	0	-4	-4	0	4	0	0	4	4	4	4	0	-4
43	3939000896	8	8	8	8	8	8	8	8	-8	-8	8	0	-8	-8	0	0	-8	-8	0	0	8	0	0	0	0	0	0	0	0	0	0	0
44	3969943680	12	12	12	8	8	8	-4	-8	0	0	-8	4	0	0	-4	-4	0	0	-4	-4	-8	4	-4	-4	4	0	4	4	4	0	0	0
45	4001669216	8	8	8	8	8	8	8	-8	0	-8	0	0	0	-8	8	-8	0	0	0	0	0	0	0	0	0	0	0	-8	8	8	0	-8
46	4169173120	14	10	10	10	10	6	-10	-2	-2	-2	-2	-2	2	2	-6	-6	2	2	-6	-6	-2	-2	-2	-2	-2	6	6	2	2	2	2	-2
47	4169697408	12	12	12	8	8	8	-12	0	0	0	0	-4	0	0	-4	-4	0	0	-4	-4	0	-4	-4	-4	-4	8	4	4	4	0	0	0
48	4238379136	10	10	10	10	10	10	-6	-6	2	2	-6	2	2	2	-6	-6	2	2	-6	-6	-6	2	-6	-6	2	2	2	2	2	2	2	2