DECISION DIAGRAM BASED COMPUTATION OF LINEARLY INDEPENDENT TERNARY ARITHMETIC TRANSFORM SPECTRA

Cicilia C. Lozano and Bogdan J. Falkowski

School of Electrical and Electronic Engineering
Nanyang Technological University
Block S1, 50 Nanyang Avenue, Singapore 639798
email: cicilia@ntu.edu.sg; efalkowski@ntu.edu.sg

Tadeusz Łuba

Institute of Telecommunications
Warsaw University of Technology
Nowowiejska 15/19, 00-665, Warsaw, Poland
luba@tele.pw.edu.pl

ABSTRACT

Classes of fastest linearly independent ternary arithmetic (FLITA) expansions have been proposed recently. They operate in arithmetic domain and have been shown to be useful for optimization of ternary functions representation. All FLITA transforms possess fast forward and inverse transforms and therefore can be calculated by fast transform method. However, it has been shown that for manipulation of large functions it is more advantageous to start from decision diagrams rather than truth vector. Hence in this paper new algorithm to obtain FLITA spectrum from ternary decision diagram is presented. It is developed based on the new notations for spectrum of an FLITA transform introduced here. The algorithm derives each spectral coefficient independently from each other, allowing the coefficients to be calculated in parallel manner. By starting from decision diagram, the algorithm enables the FLITA expansion to be computed for large functions for which the fast transform based algorithm may fail.

1. INTRODUCTION

Many problems in practice, such as transformation and optimization of microprograms, routing, and scheduling, can be formulated more naturally in terms of functions of multiple-valued variables [1], [2]. Multiple-valued logic also plays an important role in reversible logic synthesis for quantum computing [2], [3]. The simplest case of the multiple-valued logic is the three-valued or ternary logic. Besides being used to deal with ternary circuits, such as ternary multiplexer or ternary ROM, ternary logic has also been applied to solve problems related to binary circuits by formulating the problems in terms of ternary functions. Examples of such problems include the detection of hazard in binary logic circuits, evaluation of logic functions in the presence of unknown inputs, as well as AND-EXOR circuit minimization.

Fastest linearly independent ternary arithmetic (FLITA) transforms are one of the spectral arithmetic expansions that have been proposed to represent and manipulate ternary functions. The transforms were first introduced in [4] and have since been extended in [5]. FLITA transforms belong to the broader class of LITA transform, which encompasses all ternary arithmetic polynomial representations whose basis functions are linearly independent over Galois field (3) (GF(3)) [2]. The word 'fastest' in FLITA transforms was used to denote the fact that the transforms have low computational cost, which is lower than that of the arithmetic counterpart of the more well known fixed polarity Reed-Muller transform over GF(3) [2]. In addition to [4] and [5], other FLITA transforms have also been presented in [6]. Compared to the transforms in [5] the FLITA transforms introduced in [6] have slightly higher computational costs but more regular structures. It has been shown that for some ternary functions the expansions based on the transforms in [6] have smaller number of nonzero terms compared to those based on the transforms in [4] and [5]. It should be noted that for spectral expansions it is desirable to have a small number of nonzero terms as this often corresponds to simpler hardware implementation or smaller storage requirement. It also generally leads to smaller complexity for algorithms that take the spectral coefficients as their starting point. All FLITA transforms have fast forward and inverse transforms. Similar to other arithmetic expansions, FLITA transforms are useful for parallel processing of multi-output ternary functions.

Decision diagrams are modern data structures for representation and manipulation of large functions. In digital design they have been used in many applications including circuit optimization, testing, fault simulation, time-driven analysis, estimation of power dissipation, decomposition, and verification [2]. Many variants of decision diagrams have been introduced to better suit specific applications or functions with specific properties [2], [7]. Decision diagrams have also been introduced for multiple-valued functions. The simplest among them is the multiple-valued decision diagram (MDD) [1], [2], [8] or also known as multiple-place decision diagram. MDD is similar to binary decision diagram (BDD) [2], [7], except that each node may represent multiple-valued function (the outgoing edges may assume the values of 0, 1, …, $p-1$ for p-valued function) and that the terminal values are not restricted to only 0 or 1. For some functions, the compactness of the MDD can be improved by introducing edge values as well as cylic negations and complement edge operations [8], which correspond to edge values and edge negations in BDD. MDD has been applied for sequential logic synthesis [1] whereas edge-valued MDD has been shown to be efficient for representing one and two-variable elementary functions [9].

Applicability of a spectral transform depends to some extent on the availability of efficient ways to calculate its spectrum. In [4]–[6] the fast calculation of FLITA spectrum from the truth vector has been presented. However, for large functions it is impractical to work on the truth vector due to the large memory space required to store the information. Hence, in this paper an algorithm to obtain FLITA spectrum from MDD representing ternary functions is presented. The algorithm is based on the new notations for the FLITA spectral coefficients introduced here and it employs a matching process similar to that used in [10].

2. FLITA TRANSFORM AND SPECTRA

In this section several basic definitions for ternary functions and general LITA transforms are first given in Definitions 1–5. New definitions and properties used in the proposed algorithm are subsequently presented. The new definitions and properties are derived based on a particular FLITA transform [4], [6] whose definition is given in Definition 6. However, they can be easily modified for other FLITA transforms.

Definition 1. An n-variable ternary function is a mapping $f(\vec{X}_n)=\{0,1,2\}^n \rightarrow \{0,1,2,-\}$ where $\vec{X}_n=[X_n,X_{n-1},\dots,X_1]$ and

'–' denotes a don't care value. When the outputs of the function always take values from the set {0, 1, 2} for all possible input values, the ternary function is a completely specified function. Otherwise, it is an incompletely specified function.

Definition 2. The ternary variable X_i $(1 \le i \le n)$ can assume a value of 0, 1, or 2. In polynomial expansions the variable X_i is represented by its literal ${X_i}^{S_i}$, which is defined as

$$ {X_i}^{S_i} = \begin{cases} 1 \text{ if } X_i \in S_i \\ 0 \text{ if } X_i \notin S_i . \end{cases} \tag{1}$$

The set S_i is called the true set of the literal X_i, where $S_i \in \{0,1,2\}$. When S_i contains only one element *j*, the literal $X_i^{\{j\}}$ is simply written as X_i^j.

Definition 3. A logical product of literals $X_n^{S_n} X_{n-1}^{S_{n-1}} \ldots X_1^{S_1}$ is a product term. When all S_i $(1 \le i \le n)$ contains only one element the product term is a minterm. Otherwise it is a cube. Note that the literal ${X_i}^{\{0,1,2\}}$ is always equal to 1 and therefore can be dropped from the product term.

Definition 4. Let T_n be a $3^n \times 3^n$ matrix with rows corresponding to minterms and columns corresponding to the truth vectors of 3^n *n*-variable ternary switching functions. If the set of ternary switching functions are linearly independent over GF(3), then T_n has only one inverse in GF(3) and is said to be a LITA matrix. A LITA matrix T_n is a nonsingular square matrix with respect to arithmetic additions and multiplications and has a unique arithmetic inverse ${T_n}^{-1}$.

Definition 5. Let $\vec{F} = \left[f_0, f_1, \ldots, f_{3^n-1}\right]^T$ be the truth column vector of a ternary function $f\left(\vec{X}_n\right)$ in natural ternary ordering and $\vec{C} = \left[c_0, c_1, \ldots, c_{3^n-1}\right]^T$ be the coefficient column vector (spectrum) of $f\left(\vec{X}_n\right)$ for a LITA transform T_n, where T represents matrix transpose operator. Then, for any LITA transform T_n

$$\vec{F} = T_n \cdot \vec{C} \tag{2}$$

and

$$\vec{C} = {T_n}^{-1} \cdot \vec{F} . \tag{3}$$

Definition 6. The forward and inverse matrices for the FLITA transform discussed in this paper are given by the recursive equations

$$T_n = \begin{bmatrix} T_{n-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & T_{n-1} & O_{n-1} \\ Y_{n-1} & O_{n-1} & T_{n-1} \end{bmatrix} \tag{4}$$

and

$$T_n^{-1} = \begin{bmatrix} T_{n-1}^{-1} & O_{n-1} & O_{n-1} \\ O_{n-1} & T_{n-1}^{-1} & O_{n-1} \\ -Y_{n-1} & O_{n-1} & T_{n-1}^{-1} \end{bmatrix} \tag{5}$$

where O_{n-1} denotes a zero matrix of size $3^{n-1} \times 3^{n-1}$ and Y_{n-1} denotes a $3^{n-1} \times 3^{n-1}$ matrix with all zero elements except for one element at the bottom left corner of the matrix which is equal to 1.

In order to derive the algorithm to calculate the FLITA spectral coefficients from decision diagrams, it is useful to first understand the relation between the individual spectral coefficient and the function being represented. In the following several definitions and properties are given to show the relation more clearly. They are developed based on the observation that the rows in T_n^{-1} defined in (5) can be divided into (*n* + 1) groups according to the number of nonzero elements inside them and that rows with the same number of nonzero elements have the same pattern of 0, 1, and -1, which in turn corresponds to the function represented by the corresponding spectral coefficients. The number of nonzero elements in the rows ranges from 1 to $(n+1)$.

Definition 7. Each row in T_n^{-1} represents an arithmetic function on a subset of the truth vector of the ternary function being operated upon. Let the functions represented by the rows of T_n^{-1} be denoted by $C_k^{(l)}$ and the spectral coefficients c_i $(0 \le i \le 3^n - 1)$ be mapped to an alternative representation $c_k^{(l)}$ such that

$$c_k^{(l)} = C_k^{(l)} \cdot \vec{F} . \tag{6}$$

The parameters *k* and *l* are called the degree and order of the function $C_k^{(l)}$, respectively, where $0 \le k \le n$, $0 \le l \le 2 \cdot 3^{n-k-1} - 1$ for $k \ne n$, and $l = 0$ for $k = n$.

Property 1. The mapping from the notation c_i $(0 \le i \le 3^n - 1)$ to $c_k^{(l)}$ can be performed according to the following rules: A spectral coefficient c_i is mapped to $c_k^{(l)}$ with $k = j$ iff $i \bmod 3^j = 3^j - 1$ and $i \bmod 3^{j+1} \ne 3^{j+1} - 1$. Once *k* is obtained, *l* can be calculated as

$$l = \operatorname{int}(i/3^k) - \operatorname{int}(i/3^{k+1}) . \tag{7}$$

Conversely, $c_k^{(l)}$ can be mapped back to c_i by

$$i = \left(l + \operatorname{int}(l/2)\right) \cdot 3^k + 3^k - 1 . \tag{8}$$

Definition 8. Let $\left(T_n^{-1}\right)'$ be defined as T_n^{-1} that has been reordered such that the rows are arranged in ascending values of *k* and *l*.

Property 2. According to Definition 8, the function $C_k^{(l)}$ is located at the *i*-th $(0 \le i \le 3^n - 1)$ row of $\left(T_n^{-1}\right)'$ where

$$i = 3^{n-k} \cdot (3^k - 1) + l . \tag{9}$$

Property 3. The matrix $\left(T_n^{-1}\right)'$ has a regular structure which allows it to be represented by a layered Kronecker matrix equation as follows:

$$\left(T_n^{-1}\right)' = \left[\begin{array}{c} I_{n-1} \otimes \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \otimes \vec{t}_0 \\ \hdashline \vdots \\ \hdashline I_{n-j} \otimes \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \otimes \vec{t}_{j-1} \\ \hdashline \vdots \\ \hdashline I_0 \otimes \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \otimes \vec{t}_{n-1} \\ \hdashline I_0 \otimes \quad [1] \quad \otimes \vec{t}_n \end{array}\right] \tag{10}$$

where I_j $(0 \le j \le n-1)$ denotes an identity matrix of size $3^j \times 3^j$, $\otimes$ denotes Kronecker product [2], [7], and $\vec{t}_j$ $(0 \le j \le n-1)$ represents a row vector with 3^j elements. The vectors $\vec{t}_j$ are recursively defined by

$$\vec{t}_j = \left([0\ 0\ 1] \otimes \vec{t}_{j-1}\right) + \vec{s}_j \quad (11)$$

where $\vec{t}_0 = [1]$, $\vec{s}_0 = [-1]$, and $\vec{s}_j = [1\ 0\ 0] \otimes \vec{s}_{j-1}$.

Example 1. For $n = 2$, the matrices T_n^{-1} and $\left(T_n^{-1}\right)'$ are given by

$$T_2^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix} \quad \left(T_2^{-1}\right)' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}.$$

Let $\vec{C}' = \left[c'_0, c'_1, \ldots, c'_{3^n-1}\right]$ be the spectral coefficient vector obtained as the result of $\left(T_n^{-1}\right)' \cdot \vec{F}$. Then, by comparing the matrices T_2^{-1} with $\left(T_2^{-1}\right)'$, the relations between the spectral coefficients c_i, c'_i, and $c_k^{(l)}$ can be obtained as shown in Table 1 where the three notations in the same row represent the same spectral coefficient. Note that the same relations can also be derived from (7)–(9).

Definition 9. An $(n-k)$ variable standard trivial function u and a function v_k are defined for each function $C_k^{(l)}$ where u is a single ternary cube of the form

$$u(X_n, X_{n-1}, \ldots, X_{k+1}) = \begin{cases} X_n^{l_n} X_{n-1}^{l_{n-1}} \cdots X_{k+1}^{l_{k+1}} & \text{for } k \neq n \\ 1 & \text{for } k = n \end{cases} \quad (12)$$

and v_k is a function on the decomposition of the transformed function $f\left(\vec{X}_n\right)$ with respect to the ternary variables X_i, $1 \le i \le k$, as follows:

$$v_k\left(f\left(\vec{X}_n\right)\right) = \begin{cases} f_{X_k^2 X_{k-1}^2 \ldots X_1^2} - \sum\limits_{L \in \left\{3^k - 3^j \mid 1 \le j \le k\right\}} f_{X_k^{l_k} X_{k-1}^{l_{k-1}} \ldots X_1^{l_1}} & \text{for } k \neq 0 \\ 1 & \text{for } k = 0. \end{cases} \quad (13)$$

In the above, $f_{X_k^{l_k} X_{k-1}^{l_{k-1}} \ldots X_1^{l_1}}$ denotes the cofactor of f with respect to the cube $X_k^{l_k} X_{k-1}^{l_{k-1}} \ldots X_1^{l_1} = f\left(X_k = l_k, X_{k-1} = l_{k-1}, \ldots, X_1 = l_1\right)$ and the digits $l_n, l_{n-1}, \ldots, l_1$ are obtained by $< l_n, l_{n-1}, \ldots, l_{k+2} > = \langle \text{int}(l/2) \rangle_3$, $l_{k+1} = l \bmod 2$, and $< l_k, l_{k-1}, \ldots, l_1 > = \langle L \rangle_3$ where the symbol $< i >_3$ represents the ternary number representation of i.

Table 1 – Conversions between c_i, c'_i, and $c_k^{(l)}$

c'_i	$c_k^{(l)}$	c_i
c'_0	$c_0^{(0)}$	c_0
c'_1	$c_0^{(1)}$	c_1
c'_2	$c_0^{(2)}$	c_3
c'_3	$c_0^{(3)}$	c_4
c'_4	$c_0^{(4)}$	c_6
c'_5	$c_0^{(5)}$	c_7
c'_6	$c_1^{(0)}$	c_2
c'_7	$c_1^{(1)}$	c_5
c'_8	$c_2^{(0)}$	c_8

Property 4. All $C_k^{(l)}$ functions with the same k value have the same associated function v_k. On the other hand, the function u for $C_k^{(l)}$ functions with the same value of l varies, where the size of the cube represented by u (the number of minterms it covers) increases with k but does not change with n. Furthermore, from (10) and Definition 9, it can be deduced that v_k is closely related to the vector $\vec{t}_k$ whereas the function u is related to the location of the vector $\vec{t}_k$. Thus we can say that the value of the spectral coefficient $c_k^{(l)}$ is the value of the function v_k evaluated over the window function u of $C_k^{(l)}$.

Example 2. For $k = 2$ $v_k\left(f\left(\vec{X}_n\right)\right) = f_{X_2^2 X_1^2} - f_{X_2^2 X_1^0} - f_{X_2^0 X_1^0}$ whereas $v_k\left(f\left(\vec{X}_n\right)\right) = f_{X_3^2 X_2^2 X_1^2} - f_{X_3^2 X_2^2 X_1^0} - f_{X_3^2 X_2^0 X_1^0} - f_{X_3^0 X_2^0 X_1^0}$ for $k = 3$. Also, for $k = 1$ and $l = 5$ the function u of $C_k^{(l)}$ is equal to $X_3^2 X_2^1$ if $n = 3$ and is equal to $X_4^0 X_3^2 X_2^1$ if $n = 4$.

3. TERNARY DECISION DIAGRAMS

An MDD is a rooted, acylic directed graph representation of multiple-valued functions. It is an extension of the well-known BDD for which the variables and the terminal values are allowed to have more than two possible values. When the MDD represents a completely specified ternary function as given in Definition 1, such MDD is said to be a ternary decision diagram (TDD). Some terms and notations for TDDs used in this paper are reviewed below [1], [2], [8].

Definition 10. A TDD is a graphical representation of a ternary function which consists of a set of nodes *V* connected by directed edges *E*. Each node *V* in TDD represents a particular ternary function $f(V)$. The nodes pointed by the outgoing edges of a particular node *V* are said to be the children of *V*. TDD is obtained by recursively deriving the children of each node *V* through application of the multi-valued Shannon decomposition [1], [2] until a terminal value is reached.

Definition 11. The nodes in TDD can be classified into nonterminal nodes and terminal nodes. A nonterminal node *V* is labeled by a ternary variable *var*(*V*) = X_i ($1 \le i \le n$) where *i* is said to be the index of node *V* (*index*(*V*)). Each terminal node has three outgoing edges labeled by 0, 1, and 2 where the node pointed to by the edge with label j ($j \in \{0,1,2\}$) represents the cofactor of the function $f(V)$ with respect to X_i^j, $f_{X_i^j}(V)$. Unlike nonterminal node, a terminal node *V* is labeled by a constant value *value*(*V*) = *j* ($j \in \{0,1,2\}$). It represents a constant function and it does not have any outgoing edges.

Definition 12. The topmost node in TDD is called the *root node*. The incoming edge of the root node is labeled by the function represented by TDD ($f\left(\vec{X}_n\right)$).

Definition 13. Each set of directed edges which connect the root node to a terminal node forms a path. Each path represents a product term where the value of the terminal node reached by the path is the value of $f\left(\vec{X}_n\right)$ for the input variables assignment given by the product term. If V_p is the set of the nodes encountered along the path and *label* ($E(W)$) denotes the label of the edge emanating from node $W \in V_p$ along the path, then the product term represented by the path is given by $\prod\limits_{W \in V_p} X_{index(W)}^{label(E(W))}$.

Definition 14. A reduced decision diagram is a decision diagram in which the redundant nodes (nodes whose outgoing edges all point to the same node) are deleted and the isomorphic subgraphs are merged. A decision diagram is ordered if each variable is encountered at most once in every path and the variables are encountered in the same order in every path when traversing from the root node to the terminal node. A TDD that is both reduced and ordered is called the reduced ordered TDD (ROTDD).

4. ALGORITHM TO CALCULATE FLITA SPECTRAL COEFFICIENTS FROM ROTDDS

Based on the definitions for FLITA transform and ROTDD presented in Sections 2 and 3, an algorithm to calculate the FLITA spectral coefficients from ROTDDs has been derived. The algorithm works by examining every path in the ROTDD and determining whether the function output value for the product term represented by the path has an influence on the value of the spectral coefficient being considered. The determination is performed through a matching process, where the literals of the ternary variables along the path are matched with the corresponding literals in the functions u and v_k of the spectral coefficient. Two variables, *index1* and *index2*, are utilized in the matching process where *index1* (*index2*) is used to keep track of the largest (smallest) index of the variables in v_k with 0 (2) value. A path is said to match a spectral coefficient if the value of the spectral coefficient is dependent on the terminal value reached by the path, i.e., if the product term of the path covers at least one minterm which falls within the window function u and has cube $X_k^{l_k} X_{k-1}^{l_{k-1}} \dots X_1^{l_1}$ that matches one of the cubes in v_k. The number of such minterms within a matching path can be determined from *index1* and *index2*. Every time a matching path is found, the algorithm calculates the contribution of the path's terminal value to the spectral coefficient and accumulates them in the temporary variable *temp*. At the end of the algorithm, the value contained in *temp* is the value of the spectral coefficient.

Algorithm to calculate the spectral coefficient $c_k^{(l)}$

```
Step 1: Initialize temp to zero.
Step 2: Traverse all paths from the root node down to the terminal nodes. For each path carry out the matching process shown in Step 3.
Step 3: Check whether the path being considered matches $c_k^{(l)}$.
Initialize index1 and index2 with 0 and $k+1$, respectively;
For each directed edge from node $V_1$ to $V_2$ labeled by $j$ in the path
{   If ($var(V_1)$ appears in the function $u$ of $C_k^{(l)}$ )
    {   If (the literal of $var(V_1)$ in $u$ = $X_{index(V_1)}^{j}$ )
            Go to match_ok;
        Else
            Go to match_not_ok;}
    Else
    {   If ($j$ =1)
            Go to match_not_ok;
        Else
        {   If ($j$ = 0)
            {   If (max(index1, $index(V_1)$) = $index(V_1)$)
                    index1 = $index(V_1)$;}
            Else // ($j$ = 2)
            {   If (min(index2, $index(V_1)$) = $index(V_1)$)
                    index2 = $index(V_1)$;}
            If (index1 > index2)
                Go to match_not_ok;}}
match_ok:
    If $V_2$ is a terminal node
        The path matches $c_k^{(l)}$. Go to Step 4;
    Else
        Continue with the next edge;}
match_not_ok:
    The path does not match $c_k^{(l)}$. Continue to the next path;
Step 4: Determine the contribution of the matching path to the value of $c_k^{(l)}$.
If (index1 = 0)
    temp = temp − ((index2 − 2) × value($V_2$));
Else
    temp = temp − ((index2 − index1) × value($V_2$));
If (the current path is not the last path)
    Continue to the next path;
Else
    $c_k^{(l)}$ = temp;
```

Example 3. Let $f(\vec{X}_n)$ be a three-variable ternary function with truth vector $\vec{F}$ = [0,2,1,2,2,2,1,1,1,2,0,2,2,1,2,0,0,0,2,0,2,2,0,1, 1,2,0]. By (10), it can be easily obtained that $\vec{C}'$= [0,2,2,2,1,1,2,0,2,1,0,0,2,0,2,0,1,2,1,0,0,0,0,−1,0,−2,−3]. The function $f(\vec{X}_n)$ can also be represented by the ROTDD shown in Fig. 1, where the ordering is $X_2 < X_3 < X_1$ and there are a total of 21 paths. Given the ROTDD, the elements of $\vec{C}'$ for $f(\vec{X}_n)$ can also be calculated using the algorithm presented in Section 4. Table 2 lists the paths that contribute to the value of all $c_k^{(l)}$, $k > 0$, their contributions, as well as the final spectral coefficient values. Note that the listed paths are those paths that reach *match_ok* in the algorithm. The rest of the paths reach *match_not_ok* and therefore do not affect the value of *temp*. Comparing the spectral coefficient values in the table with the corresponding elements of $\vec{C}'$ obtained by (10), it can be verified that they are the same.

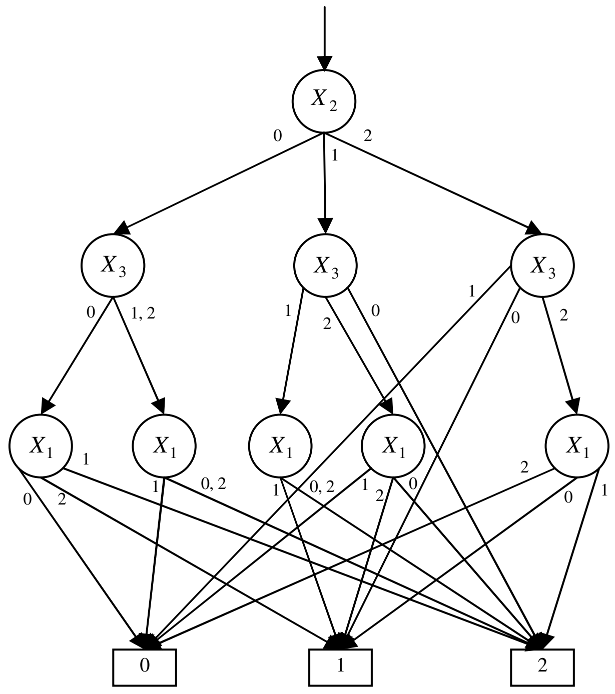

Figure 1 – ROTDD for $f(\vec{X}_n)$.

5. CONCLUSIONS

New equation for FLITA spectral coefficients has been presented which reveals more clearly the relations between the individual spectral coefficient with the function being transformed. A layered Kronecker matrix equation for the transform has also been given. As layered Kronecker matrix structure is very closely related to the pseudo-Kronecker type decision diagrams, the given equation can be helpful in obtaining spectral transform decision diagrams for FLITA expansions. An algorithm to calculate FLITA spectral coefficients from ROTDDs has also been derived based on the new definitions. By leveraging on the compactness of the ROTDD representation, the given algorithm allows FLITA spectral coefficients to be computed with less memory requirement and computational cost compared to the fast transform method [4], [6], especially for large functions. The algorithm presented here can also be implemented for MDD with edge value and/or cyclic negation and complement edge operations. The difference is only in the determination of the terminal value associated with a path. Similar to the concept of polarity used in optimization of transforms such as Reed-Muller, Arithmetic, and Haar, the FLITA expansions with permutation [6] are simply the equivalent FLITA expansion when the input variables are modified through unary cyclic addition and complement operators. Hence, the concept introduced here can also be directly extended for them.

REFERENCES

[1] T. Kam, T. Villa, R. Brayton, and A. Sangiovanni-Vincentelli, *Synthesis of Finite State Machines: Functional Optimization*. Boston: Kluwer Academic Publishers, 1997.

[2] S. N. Yanushkevich, D. M. Miller, V. P. Shmerko, and R. S. Stankovic, *Decision Diagram Techniques for Micro- and Nanoelectronic Design Handbook*. Boca Raton: CRC Press, 2006.

[3] A. N. Al-Rabadi, *Reversible Logic Synthesis: From Fundamentals to Quantum Computing*. New York: Springer-Verlag, 2004.

[4] B. J. Falkowski and C. Fu, "Properties and experimental results of fastest linearly independent ternary arithmetic transforms," *IEEE Trans. Circuits Syst. I*, vol. 53, pp. 858–866, April 2006.

[5] B. J. Falkowski, C. C. Lozano, and T. Łuba, "Properties and relations of new fastest linearly independent arithmetic transforms for ternary functions," in *Proc. 15th IEEE European Signal Processing Conf.*, Poznan, Poland, September 2007, pp. 2129–2133.

[6] C. C. Lozano, B. J. Falkowski, and T. Łuba, "Ternary polynomial expansions based on generalized fastest linearly independent arithmetic transforms," in *Proc. 15th IEEE Int. Conf. Mixed Design of Integrated Circuits and Systems*, Poznan, Poland, June 2008, pp. 297–302.

[7] R. S. Stankovic and J. T. Astola, *Spectral Interpretation of Decision Diagrams*. New York: Springer-Verlag, 2003.

[8] D. M. Miller and R. Drechsler, "On the construction of multiple-valued decision diagrams," in *Proc. 32nd IEEE Int. Symp. Multiple-Valued Logic*, Boston, Massachusets, USA, May 2002, pp. 245–253.

[9] S. Nagayama and T. Sasao, "Representations of two-variable elementary functions using EVMDDs and their applications to function generators," in *Proc. 38th IEEE Int. Symp. Multiple-Valued Logic*, Dallas, Texas, USA, May 2008, pp. 50–56.

[10] B. J. Falkowski and C. H. Chang, "Efficient algorithms for forward and inverse transformations between Haar spectrum and binary decision diagram representations of Boolean functions," in *Proc. 13th IEEE Int. Phoenix Conf. Computers and Communications*, Phoenix, Arizona, USA, April 1994, pp. 497–503.

Table 2 – Spectral coefficients computation for $f(\vec{X}_n)$

Spectral coefficient	u	v_k	Contributing paths	*index1*	*index2*	Contribution to *temp*	Final value
$c_1^{(0)}$	$X_3^0X_2^0$	$f_{X_1^2} - f_{X_1^0}$	$X_2 = 0, X_3 = 0, X_1 = 0$	1	2	$-((2-1)\times 0)) = 0$	1
			$X_2 = 0, X_3 = 0, X_1 = 2$	0	1	$-((1-2)\times 1)) = 1$	
$c_1^{(1)}$	$X_3^0X_2^1$		$X_2 = 1, X_3 = 0$	0	2	$-((2-2)\times 2)) = 0$	0
$c_1^{(2)}$	$X_3^1X_2^0$		$X_2 = 0, X_3 = 1, X_1 = 0$	1	2	$-((2-1)\times 2)) = -2$	0
			$X_2 = 0, X_3 = 1, X_1 = 2$	0	1	$-((1-2)\times 2)) = 2$	
$c_1^{(3)}$	$X_3^1X_2^1$		$X_2 = 1, X_3 = 1, X_1 = 0$	1	2	$-((2-1)\times 2)) = -2$	0
			$X_2 = 1, X_3 = 1, X_1 = 2$	0	1	$-((1-2)\times 2)) = 2$	
$c_1^{(4)}$	$X_3^2X_2^0$		$X_2 = 0, X_3 = 2, X_1 = 0$	1	2	$-((2-1)\times 2)) = -2$	0
			$X_2 = 0, X_3 = 2, X_1 = 2$	0	1	$-((1-2)\times 2)) = 2$	
$c_1^{(5)}$	$X_3^2X_2^1$		$X_2 = 1, X_3 = 2, X_1 = 0$	1	2	$-((2-1)\times 2)) = -2$	-1
			$X_2 = 1, X_3 = 2, X_1 = 2$	0	1	$-((1-2)\times 1)) = 1$	
$c_2^{(0)}$	X_3^0	$f_{X_2^2X_1^2} - f_{X_2^2X_1^0} - f_{X_2^0X_1^0}$	$X_2 = 0, X_3 = 0, X_1 = 0$	2	3	$-((3-2)\times 0)) = 0$	0
			$X_2 = 2, X_3 = 0$	0	2	$-((2-2)\times 1)) = 0$	
$c_2^{(1)}$	X_3^1		$X_2 = 0, X_3 = 1, X_1 = 0$	2	3	$-((3-2)\times 2)) = -2$	-2
			$X_2 = 2, X_3 = 1$	0	2	$-((2-2)\times 0)) = 0$	
$c_3^{(0)}$	1	$f_{X_3^2X_2^2X_1^2} - f_{X_3^2X_2^2X_1^0} - f_{X_3^2X_2^0X_1^0} - f_{X_3^0X_2^0X_1^0}$	$X_2 = 0, X_3 = 0, X_1 = 0$	3	4	$-((4-3)\times 0)) = 0$	-3
			$X_2 = 0, X_3 = 2, X_1 = 0$	2	3	$-((3-2)\times 2)) = -2$	
			$X_2 = 2, X_3 = 2, X_1 = 0$	1	2	$-((2-1)\times 1)) = -1$	
			$X_2 = 2, X_3 = 2, X_1 = 2$	0	1	$-((1-2)\times 0)) = 0$	