New Orthogonalizing Boolean Equation using in the Calculation of Test Patterns for Combinatorial Circuits

Yavuz Can[1], Georg Fischer[2]

[1,2] Institute for Electronics Engineering
Friedrich-Alexander-University Erlangen-Nuremberg, Germany
{yavuz.can, georg.fischer}@fau.de

Abstract

In this paper a new Boolean equation for the orthogonalization of Boolean functions respectively of Ternary-Vector-Lists of disjunctive normal form is presented. It provides the mathematical solution of orthogonalization. The new equation is based on the new method of orthogonalizing OR-ing $\bigtriangledown$ which enables the building the union of two product terms respectively of two Ternary-Vectors whereby the result is orthogonal. The algorithm based on the new equation has a faster computation time in contrast to other methods. Further advantage is the smaller number of the product terms respectively of the Ternary-Vectors in the orthogonalized result which reduces the number of further calculation steps. Furthermore, the new equation can be used as a part in the calculation procedure of getting suitable test patterns for combinatorial circuits for verifying feasible logical faults.

1. Introduction

Combinatorial circuits, which are described mathematically by corresponding Boolean function, consist of several gates which are built of transistors. Faults caused by defective transistors of a gate or broken respectively shorted wires between gates affects the output of the combinatorial circuit. All changing behavior of the circuit can be described by the Boolean Differential Calculus by which all possible test patterns can also be determined which are necessary to discover possible occuring faults in the logic circuit. However, this calculation method can be performed easier in TVL-arithmetic by the method in [11] and thereby the advantages in terms of computation time and memory request can be utilized computationally very well (Fig. 1).

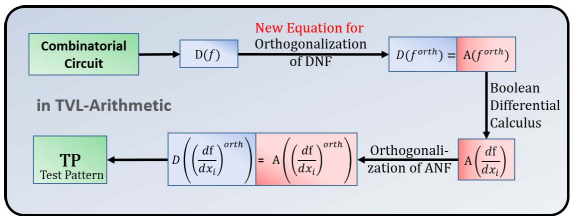

Figure 1. Calculation steps for test patterns in TVL-arithmetic

In this way, the Boolean function is converted in a corresponding disjunctive normal form $D(f)$ which is orthogonalized and subsequently transformed into the orthogonal antivalence normal form $A(f^{orth})$. For $A(f^{orth})$ the Boolean Differential Calculus is performed on the TVL-level. After another orthogonalization of the result of the differential $A(\frac{df}{dx_i})$ it is transformed into the differential $D((\frac{df}{dx_i}^{\text{orth}}))$ as the orthogonal disjunctive normal form. This differential provides all possible test patterns. A new equation for the orthogonalization of disjunctive normal form $D(f)$ based on the new method orthogonalizing OR-ing $\bigtriangledown$ which is used as a part of the whole calculations for determining test patterns is presented in this work.

2. Theoretical Foundations

2.1. Ternary-Vector-List (TVL)

Ternary-Vector-Lists (TVLs) are representations of Boolean functions and can be treated computationally easier. Using TVL has many advantages in terms of computation time and memory request [1], [11], [12]. TVL consists of one or more Ternary-Vectors (TVs) which constitute the terms or the clauses contained in the Boolean function. The length of a TV called dimension which can take one of three possible values for each point, $t \in \{0,1,-\}$. A non-negated variable is characterized by '1', a negated by '0' and not included by '-'. A TVL consists of m-rows and n-columns (number of independent variables) and is characterized with $m,n \in \mathbb{N}^+$:

$$\text{TVL} = \begin{bmatrix} TV_1 \\ TV_2 \\ \vdots \\ TV_m \end{bmatrix} := \begin{array}{c} \begin{matrix} x_n & \dots & x_2 & x_1 \end{matrix} \\ \begin{bmatrix} t_{1n} & \dots & t_{12} & t_{11} \\ t_{2n} & \dots & t_{22} & t_{21} \\ . & \dots & . & . \\ t_{mn} & \dots & t_{m2} & t_{m1} \end{bmatrix} \end{array} \tag{1}$$

Any form of a function, that means disjunctive normal form (DNF), conjunctive normal form (CNF), equivalence normal form (ENF) and antivalence normal form (ANF), can be represented as a TVL. As no operators $(\vee, \wedge, \odot, \oplus)$ in the TVL presentation are given, the designations of the matrix by $D(f), K(f), E(f)$ and $A(f)$ show the type of the TVL [10], [11], [12], [17]. The rule for AND-ing $(\wedge)$ of two TVs $(TV_{i,j})$ of the type $D(f)$, which represent an average TV (TV_k), is defined by Table 1:

$\wedge$	0	1	–
0	0	×	0
1	×	1	1
–	0	1	–

if $\times \geq 1$
$\quad \text{TV}_k = \times$ (empty TV)
else
$\quad \text{TV}_k = \text{TV}_i \wedge \text{TV}_j$

Table 1. Rule for AND-ing of two TVs

The rule for OR-ing $(\vee)$ of two TVs $(TV_{i,j})$ of the type $D(f)$, which represent the union of both TVs, is defined by Eq. (2):

$$TV_i \vee TV_j = \begin{bmatrix} TV_i \\ TV_j \end{bmatrix} \tag{2}$$

2.2. Characteristic of Orthogonality

A Boolean function or TVL is orthogonal if its product terms respectively its TVs are disjoint to one another in pairs. Consequently, these product terms ($m_{i,j}$) or these TVs ($TV_{i,j}$) then have no common covering after their logical conjunction, $m_i(\underline{x}) \wedge m_j(\underline{x}) = 0$ or $TV_i \wedge TV_j = \times$. An orthogonal disjunctive normal form is equivalent to the orthogonal antivalence normal form including the same product terms, $D(f)^{orth} = A(f)^{orth}$. That means the orthogonalization of a disjunctive normal form facilitates the transformation into an orthogonal antivalence normal form. This characteristic simplifies the handling for further calculations as the Boolean Differential Calculus especially in TVL-arithmetic.

3. Method of orthogonalizing OR-ing $\veebar$

3.1. $\veebar$ of two product terms or of two TVs

Orthogonalizing OR-ing $\veebar$ is a variant of building the disjunction of two product terms ($m_{S_1}(\underline{x}), m_{S_2}(\underline{x})$) whereby the result is orthogonal. Orthogonalizing OR-ing is explained by an example in a K-map with 4 variables (Fig. 2). Two product terms (a group of 8 and a group of 4) are orthogonalizing OR-ed and a result consisting of several blocks appears which are pairwise orthogonal to each other.

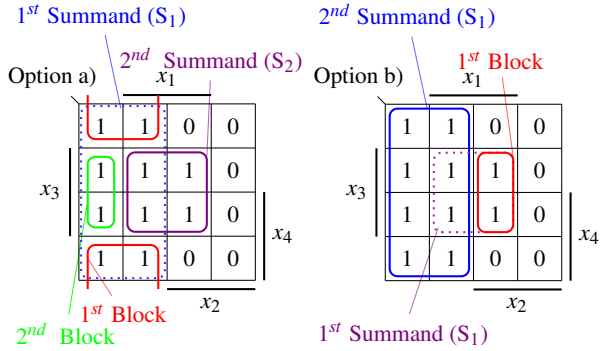

Figure 2. Example of $\veebar$ in a K-map

Option a): $\bar{x}_2 \veebar x_3x_1 = \underbrace{\bar{x}_3\bar{x}_2 \vee x_3\bar{x}_2\bar{x}_1 \vee x_3x_1}_{\text{Sum of Blocks}}$

Option b): $x_3x_1 \veebar \bar{x}_2 = \underbrace{x_3x_2x_1 \vee \bar{x}_2}_{\text{Sum of Blocks}}$

The method, "the orthogonalizing OR-ing $\veebar$" , is based on the orthogonalizing difference-building [6], [5], which is the basis for setting up an equation (Eq. (3)) in the following. The method $\veebar$ corresponds to the removal of the intersection which is formed between the first summand $m_{S_1}(\underline{x})$ and the second summand $m_{S_2}(\underline{x})$, from the first summand $m_{S_1}(\underline{x})$, which means $m_{S_1}(\underline{x}) \setminus (m_{S_1}(\underline{x}) \wedge m_{S_2}(\underline{x}))$ and the disjunction of the second summand $m_{S_2}(\underline{x})$ at last; the result is orthogonal:

$$m_{S_1}(\underline{x}) \veebar m_{S_2}(\underline{x}) = \left(\bigwedge_{s_1=0}^{n-1} x_{n-s_1}\right) \veebar \left(\bigwedge_{s_2=0}^{n-1} x_{n-s_2}\right) :=$$

$$= \left[\left(\bigwedge_{s_1=0}^{n-1} x_{n-s_1}\right) \wedge \left(\bigvee_{s_2=0}^{(n-1)_j} \bar{x}_{(n-s_2)_j}\right)\right] \vee \left(\bigwedge_{s_2=0}^{n-1} x_{n-s_2}\right) = \quad (3)$$

$$= [\left(x_nx_{n-1}..x_1\right)_{S_1} \wedge \left(\bar{x}_{n_j} \vee x_{n_j}\bar{x}_{(n-1)_j} \vee .. \vee x_{n_j}x_{(n-1)_j}..\bar{x}_{1_j}\right)_{S_2}] \vee$$
$$\vee \left(x_nx_{n-1}..x_1\right)_{S_2}$$

In this case, the formula $(\bigvee_{i=1}^{n_j} \bar{x}_{i_j} = \bar{x}_{1_j} \vee x_{1_j}\bar{x}_{2_j} \vee .. \vee x_{1_j}x_{2_j}..\bar{x}_{n_j})$ from [7] is used in rearranged form to describe the orthogonalizing OR-ing in a mathematically easier way, where $x_{1_j}, x_{2_j}, .., x_{n_j}$ are literals of the second summand product term. Depending on the starting literal the result may differ. There are many equivalent options which only differ in the form of coverage. If both product terms are disjoint (orthogonal) to each other, $m_{S_1}(\underline{x}) \perp m_{S_2}(\underline{x})$, the result corresponds to the disjunction of both product terms:

$$m_{S_1}(\underline{x}) \veebar m_{S_2}(\underline{x}) = m_{S_1}(\underline{x}) \vee m_{S_2}(\underline{x}) \quad (4)$$

By swapping the positions of the two summands, the result changes. However both solutions are equivalent because the same set is covered. They only differ in the form of coverage which can also be seen in Figure 2, in which both possible solutions are presented. The following Equation (5) is used for the orthogonalizing OR-ing of two Ternary-Vectors (TV_{S_1}, TV_{S_2}):

$$TV_{S_1} \veebar TV_{S_2} := [t_n,..,t_2,t_1]_{S_1} \veebar [t_n,..,t_2,t_1]_{S_2} = \quad (5)$$

$$= \left[[t_n,..,t_2,t_1]_{S_1} \wedge \begin{bmatrix} \overline{t_n} & .. & - & - \\ : & : & : & : \\ t_n & .. & \overline{t_2} & - \\ t_n & .. & t_2 & \overline{t_1} \end{bmatrix}_{S_2} \right] \vee [t_n,..,t_2,t_1]_{S_2}$$

3.2. $\veebar$ of Function and Product Term or TVL and TV

By Equation (6) based on the orthogonalizing OR-ing it is possible to calculate an orthogonal union of an orthogonal function $f_{S_1}(\underline{x})^{orth}$ and a product term $m_S(\underline{x})$. Equation (6) and (7) are used in the new function for orthogonalization which is described in the next chapter. At the same time the orthogonal function $f_{S_1}(\underline{x})^{orth}$ in (6) is calculated by Equation (3):

$$f_{S_1}(\underline{x})^{orth} \veebar m_{S_2}(\underline{x}) = \left(\bigvee_{i=1}^{l_{s_1}} \bigwedge_{s_1=0}^{n-1} x_{i,(n-s_1)}\right) \veebar \left(\bigwedge_{s_2=0}^{n-1} x_{(n-s_2)}\right) := \quad (6)$$

$$= \bigvee_{i=1}^{l_{s_1}} \left[\left(\bigwedge_{s_1=0}^{n-1} x_{i,(n-s_1)}\right) \wedge \left(\bigvee_{s_2=0}^{(n-1)_j} \bar{x}_{(n-s_2)_j}\right)\right] \vee \left(\bigwedge_{s_2=0}^{n-1} x_{(n-s_2)}\right)$$

In this case it is important to stress that any outcome of each individual $\veebar$-linking has to be considered. Corresponding to the TVL-arithmetic the orthogonalizing OR-ing between a TVL ($D(f_{S_1})$) and a TV (TV_{S_2}) can be determined by the Equation (7):

$$D(f_{S_1}^{orth}) \veebar TV_{S_2} = \begin{bmatrix} TV_{1S_1} \\ TV_{2S_1} \\ : \\ TV_{mS_1} \end{bmatrix} \veebar [TV_{S_2}] :=$$

$$= \begin{bmatrix} TV_{1S_1} \ominus TV_{S_2} \\ TV_{2S_1} \ominus TV_{S_2} \\ : \\ TV_{mS_1} \ominus TV_{S_2} \end{bmatrix} \vee [TV_{S_2}] \quad (7)$$

4. Orthogonalization based on $\veebar$

4.1. Boolean Equation

A new Equation (8) based on the new method of orthogonalizing OR-ing $\veebar$ (ov) is formed for the orthogonalization of Boolean functions of DNF. With $n \in \mathbb{N}^+$ as the number of product terms follows:

$$f(\underline{x})_{ov}^{orth} := \bigvee_{i=1}^{n} m_i(\underline{x}) = \tag{8}$$
$$= m_1(\underline{x}) \,\text{Ⓥ}\, m_2(\underline{x}) \,\text{Ⓥ}..\text{Ⓥ}\, m_{n-1}(\underline{x}) \,\text{Ⓥ}\, m_n(\underline{x})$$

In the following the explanation for Equation (8) is provided:

- The orthogonalizing OR-ing is realized between the first and the second product term, $m_1(\underline{x})$ Ⓥ $m_2(\underline{x})$, by Eq. (3). After that, the orthogonalizing OR-ing is realized between the result of $\left(m_1(\underline{x}) \,\text{Ⓥ}\, m_2(\underline{x})\right)$ and the third product term, $\left(m_1(\underline{x}) \,\text{Ⓥ}\, m_2(\underline{x})\right)$ Ⓥ $m_3(\underline{x})$, by Eq. (6). The procedure is continued until the last product term. The method ends with the last product term.

The orthogonal result may differ depending on the order of the product terms due to the commutativity. All solutions are equivalent. However, the resorting of the product terms has the advantage to obtain better results, that means an orthogonal function with a smaller number of product terms in the result. The re-ordering starts with the smallest product term and ends with the largest product term. The general validity is proved by mathematical induction:

Proof.

(1) Basis: $n = 1$

$$\bigvee_{i=1}^{1} m_i(\underline{x}) = m_1(\underline{x})$$
$$m_1(\underline{x}) = m_1(\underline{x}) \qquad \text{(Statement is true)}$$

(2) Inductive step: $n = n+1$

$$\bigvee_{i=1}^{n+1} m_i(\underline{x}) = m_1(\underline{x}) \,\text{Ⓥ}\, m_2(\underline{x}) \,\text{Ⓥ}..\text{Ⓥ}\, m_{(n+1)-1}(\underline{x}) \,\text{Ⓥ}\, m_{n+1}(\underline{x})$$

$$\left(\bigvee_{i=1}^{n} m_i(\underline{x})\right) \,\text{Ⓥ}\, m_{n+1}(\underline{x}) = \underbrace{m_1(\underline{x}) \,\text{Ⓥ}..\text{Ⓥ}\, m_n(\underline{x})}_{\bigvee_{i=1}^{n} m_i(\underline{x})} \,\text{Ⓥ}\, m_{n+1}(\underline{x})$$

□

Example: Function $f_5(\underline{x}) = \bar{x}_2 \vee x_1 \vee x_3 \equiv \begin{bmatrix} - & 0 & - \\ - & - & 1 \\ 1 & - & - \end{bmatrix}$ is orthogonalized by Eq. (8). Both functions are illustrated in K-maps (Fig. 3)

$$f_5(\underline{x})^{orth} = \bar{x}_2 \,\text{Ⓥ}\, x_1 \,\text{Ⓥ}\, x_3 = (\bar{x}_2\bar{x}_1 \vee x_1)\text{Ⓥ}x_3 =$$
$$= \bar{x}_3\bar{x}_2\bar{x}_1 \vee \bar{x}_3 x_1 \vee x_3 \equiv \begin{bmatrix} 0 & 0 & 0 \\ 0 & - & 1 \\ 1 & - & - \end{bmatrix}$$

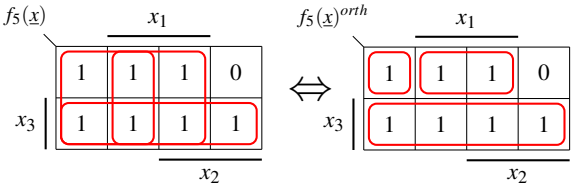

Figure 3. $f_5(\underline{x})$ and $f_5(\underline{x})^{orth}$ in K-maps

The corresponding Equation (9) is derived for the orthogonalization in the TVL-representation. The orthogonalization of a TVL of disjunctive normal form $D(f)$ or the conjunctive normal form $K(f)$ is determined by the same equation due to the duality. The interpretation of the forms take place after the orthogonalization.

$$D(f)_{ov}^{orth} := \bigvee_{i=1}^{n} TV_i \qquad \text{with} \quad n \in \mathbb{N}^+ \tag{9}$$

4.2. Algorithm $ORTH[\text{Ⓥ}]$

In Figure 4 the flow chart for the algorithm for the orthogonalization of a TVL of the disjunctive normal form $D(f)$ based on the new Eq. (9) is illustrated. Two sub-functions *Absorb*() and *Sort*() are additionally used to reach a better result, which means the number of TVs in the orthogonal result is smaller in contrast to the methods in [14] and [17].

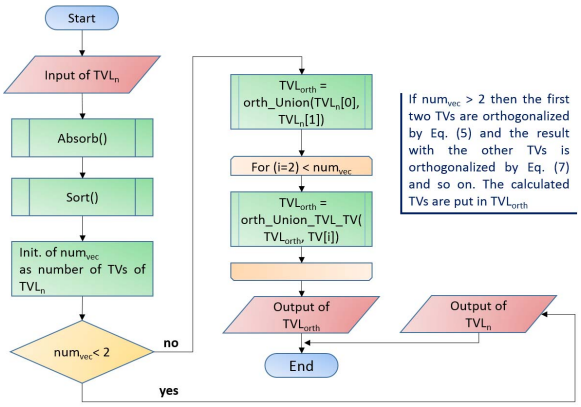

Figure 4. Flow Chart of $ORTH[\text{Ⓥ}]$

The function *Absorb*() ensures that TVs which constitute smaller product terms can be absorbed by TVs which constitute larger product terms if those are already covered by the larger ones. Thereby, the number of TVs is reduced by absorption. Thus, duplicated TVs are reduced to a single TV. Consequently, the number of TVs that have to be treated decreases. By the function *Sort*() optionally follows the resorting of the TVs from small to large, which means from TVs with higher number of $\{0,1\}$-values to TVs with lower number of $\{0,1\}$-values.After proceeding these two sub-functions the process of orthogonalization according to the new Eq. (9) is performed. If the order of the two sub-functions is interchanged, the result may differ but all several results are equivalent to each other.

5. Measurements and Results

5.1. Comparison of Computation Times

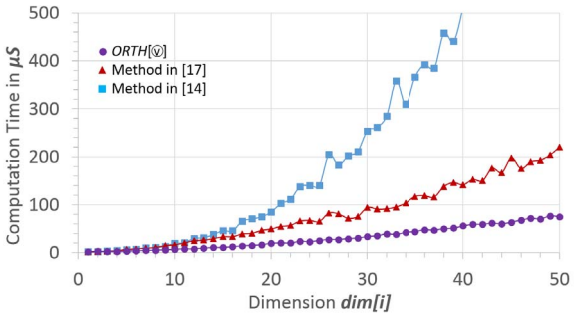

Figure 5. Comparison of computation times (TVL_{in} with 5 TVs)

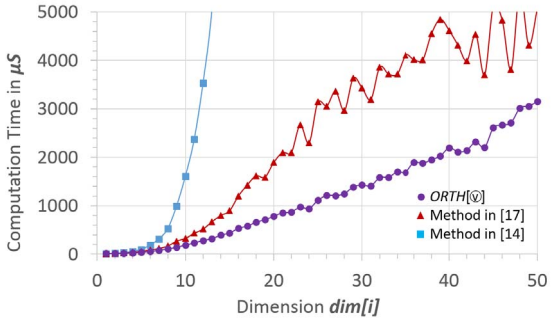

Figure 6. Comparison of computation times (TVL_{in} with 25 TVs)

In diagrams 5 and 6 the methods for orthogonalization in [14] (blue) and [17] (red) are compared with the new method $ORTH[Ⓥ]$ (violet) in respect to the dimension $dim[i]$. Dimension corresponds to the number of literals of the product terms. One literal corresponds to one input of the combinatorial circuit. As in most applications the number of inputs of a circuit does not rise above 50 the measurements are limited to the dimension of $dim[50]$. The new method has faster computation time in comparison to the other two methods in [14] (blue) and [17] (red). The computation time of the method in [14] increases polynomial by increasing dimension. The value of the polynome rises with increasing number of TVs in the Input-TVL. The curve for the new method runs similarly dynamic as the curve for the method in [17]. However, the computation time of the new method is faster in comparison. The distinction is that the new method calculates the orthogonalizing OR-ing Ⓥ consistently no matter if two TVs are orthogonal.

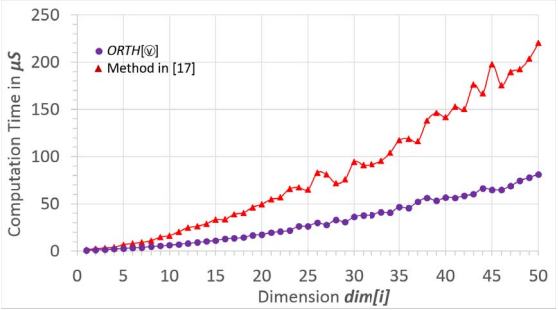

Figure 7. Comparison of the method in [17] and the new method excluding *Absorb*() and *Sort*() (TVL_{in} with 5 TVs)

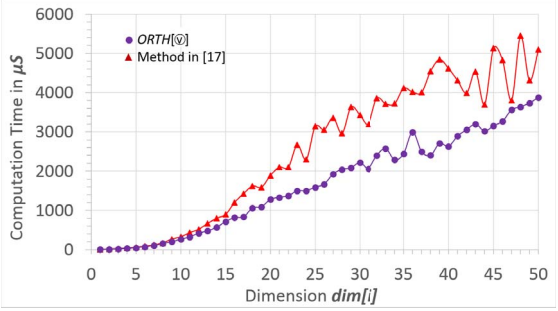

Figure 8. Comparison of the method in [17] and the new method excluding *Absorb*() and *Sort*() (TVL_{in} with 25 TVs)

In the method in [17] the investigation if two TVs are orthogonal to each other takes place in form of another sub-function; this might be a reason for the deterioration of the computation time. In summary, it has to be clarified here that the new method has a faster computation time for TVLs including the number of TVs between 2 and 25 than the other methods. Even if the two sub-functions *Absorb*() and *Sort*() are excluded, the new method (violet) has faster computation time in comparison to the method in [17] (red), as shown in Figure 7 and 8.

5.2. Comparison of the Number of TVs in the Result-TVL

In this section the number of TVs in the result-TVL is analyzed in comparison to the methods in [14] and [17]. In this case, the average number of TVs in respect to the dimension $dim[i]$ in the orthogonal result is compared. The average value is formed of 100 calculated tasks for each dimension. That comparison with an input-TVL (TVL_n) including 5 TVs is shown in Fig. 9 and the comparison with an input-TVL (TVL_n) including 25 TVs is illustrated in Fig. 10. Both charts illustrate that the new method $ORTH[Ⓥ]$ (violet) depending on the dimension $dim[i]$ offers better results than the other two methods in [14] (blue) and [17] (red) with increasing number of TVs in TVL_n. This attribute is important because a further calculation of orthogonalized TVL needs fewer operations and is carried out more quickly. In this case, a further calculation of a TVL such as the Boolean Differential Calculus is performed with a fewer number of TVs and thus reduces both the memory request and the number of calculation steps which certainly affects the computation time. The corresponding average values depending on the dimension $dim[i]$ are partially shown in Table 2.

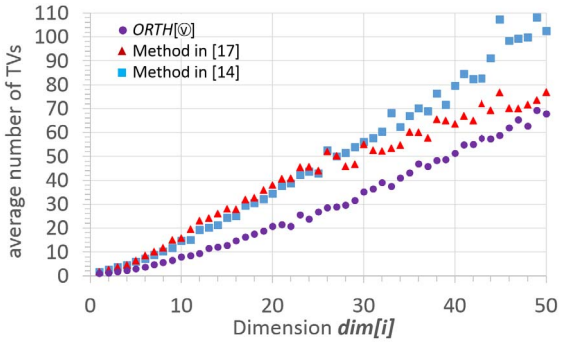

Figure 9. Average number of TVs in the result-TVL (TVL_{in} with 5 TVs)

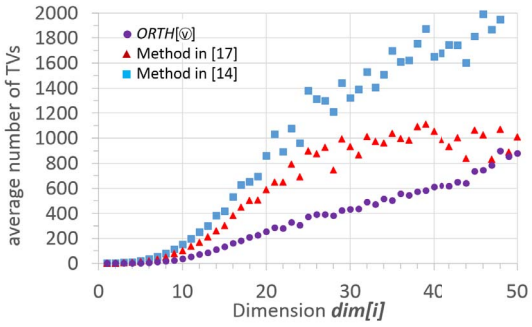

Figure 10. Average number of TVs in the result-TVL (TVL_{in} with 25 TVs)

	TVL_{in} with 5 TVs			TVL_{in} with 25 TVs		
	Methods			Methods		
dim[i]	[14]	[17]	$ORTH[\bigtriangledown]$	[14]	[17]	$ORTH[\bigtriangledown]$
1	1.76	1.67	1.00	2	1.66	1.00
2	2.69	2.61	1.29	3.96	2.94	1.00
3	3.76	3.71	1.73	7.39	5.23	1.26
:	:	:	:	:	:	:
25	43.00	44.10	26.63	1381.58	897.47	373.82
:	:	:	:	:	:	:
48	99,86	71,60	62,77	1949.19	1069.61	897.69
49	108.20	73.59	69.25	2010.11	888.01	854.23
50	102.57	76.98	67.78	2102.15	1011.66	878.78

Table 2. Average values of TVs in result-TVL

6. Conclusion

In this paper, a new Boolean equation for the orthogonalization of Boolean functions respectively of Ternary-Vector-Lists of disjunctive normal form is presented which is based on the new combination method of orthogonalizing OR-ing $\bigtriangledown$ and has general validity. By the application of the new equation the processing step for orthogonalization is also supported in the TVL-arithmetic. Boolean equation for the problem of orthogonalization of disjunctive normal form is shown here. Additionally, the simpler computational recharge of orthogonal Ternary-Vector-Lists has advantages in terms of memory request and computation time. The application of the Boolean Differential Calculus on orthogonal TVLs is significantly facilitated. The new method $ORTH[\bigtriangledown]$ has faster computation time in respect to increasing dimension in contrast to the methods in [14] and [17]. Furthermore, the new method has the property to determine better results, which means the average value of the number of product terms respectively of Ternary-Vectors in the orthogonal result is lower in contrast to the other methods. Thus, further processing of a result-TVL with a smaller number of TVs has the advantages in terms of memory request and computation time because the number of additional operation steps is reduced. The new function can used as a part of a calculation for determining of test patterns for verifying possible faults in combinational circuit.

7. References

[1] D. Bochmann. *Binäre Systeme - Ein Boolean Buch*. LiLoLe-Verlag, Hagen, Germany, 2006.

[2] D. Bochmann and C. Posthoff. *Binäre dynamische Systeme*. Akademie-Verlag, Berlin, DDR, 1981.

[3] D. Bochmann, A. Zakrevskij, and C. Posthoff. *Boolesche Gleichungen. Theorie - Anwendungen - Algorithmen*. VEB Verlag Technik, Berlin, DDR, 1984.

[4] I. Bronstein, G. Musiol, H. Mühlig, and K. Semendjajew. *Taschenbuch der Mathematik*. Harri Deutsch Verlag, Frankfurt am Main, Thun, Germany, 1999.

[5] Y. Can and G. Fischer. Boolean Orthogonalizing Combination Methods. In *Fifth International Conference on Computational Science, Engineering and Information Technology (CCSEIT 2015)*, Vienna, Austria, 23-24 May, 2015.

[6] Y. Can and G. Fischer. Orthogonalizing Boolean Subtraction of Minterms or Ternary-Vectors. In *International Conference on Computational and Experimental Science and Engineering (ICCESEN 2014)*, Antalya,Turkey, 24-29 October, 2014.

[7] Y. Crama and P. Hammer. *Boolean Functions. Theory, Algorithms, and Applications*. Cambridge University Press, New York, USA, 2011.

[8] D. Hoffmann. *Theoretische Informatik*. Carl Hanser Verlag GmbH & Co, Munich, Germany, 2009.

[9] H. Kassim, Y. Can, and M. Sattler. *Untersuchung eines neuen Algorithmus zur Berechnung orthogonalisierter Differenz*. Bachelor-thesis, Erlangen, Germany, 2014.

[10] G. Kempe. *Tupel von TVL als Datenstruktur für Boolesche Funktionen*. Dissertation, Freiberg, Germany, 2003.

[11] M. Kühnrich. *Ternärvektorlisten und deren Anwendung auf binäre Schaltnetzwerke*. Dissertation, Karl-Marx-Stadt (Chemnitz), DDR, 1979.

[12] W. Matthes. *Spezielle Hardware zur Verarbeitung von Ternärvektorlisten*. Dissertation, Karl-Marx-Stadt (Chemnitz), DDR, 1987.

[13] W. Matthes. *Datenzugriffsprinzipien in objektorientierten Rechnerarchitekturen*. Preprint, Karl-Marx-Stadt (Chemnitz), DDR, 1989.

[14] C. Posthoff. *Der Boolesche Differentialkalkül und seine Anwendung auf die Untersuchung dynamischer Erscheinungen in binären Systemen*. Dissertation, Karl-Marx-Stadt (Chemnitz), DDR, 1979.

[15] C. Posthoff, D. Bochmann, and K. Haubold. *Diskrete Mathematik*. BSB Teubner, Leipzig, DDR, 1986.

[16] C. Posthoff and B. Steinbach. *Binäre Gleichungen - Algorithmen und Programme.* wissenschaftliche Schriftreihen, Karl-Marx-Stadt (Chemnitz), DDR, 1979.

[17] C. Posthoff and B. Steinbach. *Logikentwurf mit XBOOLE. Algorithmen und Programme.* Verlag Technik GmbH, Berlin, Germany, 1991.

[18] J. Whitesitt. *Boolesche Algebra und Ihre Anwendungen. Band 3*. Friedr. Vieweg + Sohn GmbH, Braunschweig, Germany, 1969.

[19] H. Zander. *Logischer Entwurf binärer Systeme*. Verlag Technik, Berlin, DDR, 1989.