

# The Word-Level Models for Efficient Computation of Multiple-Valued Functions. PART 1: LAR Based Model

Svetlana N. Yanushkevich

*Department of Electrical and Computer Engineering,  
University of Calgary, CANADA, yanush@enel.ucalgary.ca*

Piotr Dziurzynski\*

*Faculty of Computer Science,  
Technical University of Szczecin,  
POLAND, pdziurzynski@wi.ps.pl*

Vlad P. Shmerko

*Department of Electrical and Computer  
Engineering, University of Calgary,  
CANADA, shmerko@enel.ucalgary.ca*

## Abstract

*A new model of a multi-level combinational Multiple-Valued Logic (MVL) circuit with no feedback and no learning is introduced. This model includes Neuron-Like Gates (NLGs), each represents a level of the MVL circuit, so that the number of NLGs in the corresponding Neural-Like Network (NLN) is equal to the number of levels in the circuit. The formal description of an NLG is a Linear Arithmetic Expression (LAR) that is directly mapped to the Linear word-level Decision Diagram (LDD) planar by its nature. Thus, an  $l$ -level MVL circuit is described by a set of  $l$  LDDs. The experiments on simulation of large MVL circuits show that the LDD format of an MVL circuit consumes 5-20 times less memory than EDIF and ISCAS formats. The proposed technique allows to simulate an arbitrary MVL circuit by an NLN and corresponding set of LDDs. In particular, we successfully simulated an NLN with about 250 NLGs corresponding to an MVL circuit with more than 8000 ternary gates that has been impossible by any recently reported threshold gate-based network.*

## 1 Introduction

For many years, logic circuit design based on Threshold Gates (TGs) - **feedforward networks with TGs and no learning** - has been considered as an alternative to the traditional logic gate design

---

\*Supported by PhD grant from State Committee for Scientific Research (KBN), Poland

procedure [6]. Recently, the interest in study of TG-based computing of logic functions has resurged, since the advances in VLSI technology have made possible the implementation of a massively interconnected network of the TGs [11]. Formally, a TG is described by a threshold decision (linearly separable) functions. This principle is general itself, so the simple logic gates, such as AND and OR gates, are special cases of the TG. The power of the TG design style lies in the intrinsic complex functions implemented by such the gates, which allow implementations with less TGs or gate levels than design with standard logic gates. In particular, multiple-addition, multiplication, division or sorting can be implemented by polynomial-size TG circuits of small depth.

It was reported in many papers that **it is possible to realize TGs in silicon having area and time performances comparable with classical logic gates**, and foresee the VLSI implementation of larger NNs [8, 11]. This is one of the motivation of our interest. We refer the previously obtained results on MVL NNs as follows. A Multiple-Valued TG (MV-TG) and MV NN have been introduced in [9] and MVL NN to represent logic knowledge has been reported in [10]. In [2] an MV NN based minimizer of MVL functions has been proposed. Unfortunately, these implementations of the TG and MV-TG networks are limited in the size of modeled logic functions and circuits (about a dozen inputs). This is the second motivation to develop new models of MVL networks.

While NN based computation of logic functions is the focus of many researches, the decision diagram representation of TG networks are far from being understood. Our idea is to apply word-level description (ex-

pressions and decision diagrams) of logic function to represent NLGs whose parameters (weights, threshold) are word-level. We found that **the boundary case, LDDs**, meets the number of requirements to model the circuit implemented on deep sub-micron technology where the wiring delay, communication complexity, crosstalk may be critical. Note, that LDDs are planar by their nature, that (planarity) of DDs [7] is an important factor for the VLSI technology.

## 2 Basic definitions

Our study has been stimulated by the ideas of computation of logic functions on MV-TG [2, 9, 10].

**Definition 1. [9]:** *An MV-TG is a unit with  $n$  MVL inputs (variables)  $x_i \in \{0, 1, 2, \dots, m-1\}$ ,  $i = 1, 2, \dots, n$ , and the output*

$$f = \begin{cases} L & \text{if } L > 0 \\ 0 & \text{if } L \leq 0 \end{cases}, \quad L = \sum_{i=1}^n w_i x_i - \theta, \quad (1)$$

$w_1, w_2, \dots, w_n$  (weights) and  $\theta$  (threshold) are real numbers.

Below we specify our model of computation of an MVL function.

**Definition 2.** *An NLG is an  $(n+1)$ -input single-output processing element that implements the function*

$$f = \Xi^\xi\{D\}, \quad D = \sum_{i=1}^n d_i x_i + d_0, \quad (2)$$

where  $d_1, \dots, d_n$  are integer numbers (weights),  $d_0$  is the bias, and  $\Xi^\xi$  is a masking operator (defined below).

We call expression (2) a **word-level model of NLG**. The expression of the  $D$  in (2) is a Linear Arithmetic expression (LAR), i.e. an arithmetical expression that does not include any product of variables.

In our model, a threshold of the word-level notation be a masking operator.

**Definition 3. The masking operator** is an operator to extract the  $\xi$ -th digit,  $\xi \in \{0, \dots, k\}$ , of an  $m$ -valued representation of the number  $A = m^k a_k + m^{k-1} a_{k-1} + \dots + m^\xi a_\xi + \dots + m^0 a_0$ ,

$$\Xi^\xi\{A\} = \left\lfloor \frac{A}{m^\xi} \right\rfloor \bmod m = a_\xi. \quad (3)$$

For instance, given  $A = 7$ ,  $\xi = 2$  and  $m = 2$ , we obtain  $a_2 = \left\lfloor \frac{7}{2^2} \right\rfloor \bmod 2 = 1$ .

**Definition 4.** *The NLN is a model of a multi-level combinational MVL network with **no feedback and no learning**.*

We will show below that the initial MVL circuit is mapped into the NLN directly, level by level. In traditional methods, every gate of a circuit is mapped to a MV TG or to a set of MV TGs.

## 3 Task formulation and the strategy to solve the problem

**Given** an arbitrary combinational MVL circuit,  
**Find** an alternative to MV-TG representation of combinational MVL circuit.

**Strategy. At the first phase** we represented every MVL gate (from the library of gates) by a LAR. This LAR is a formal description of the NLG. Then, every level of the circuit is described by a LAR that is the weighted sum of the LAR of the gates. This is a more powerful gate that includes a number of simple NLGs. In this way we represent the circuit by a set of LARs, as every LAR corresponds to an NLG. **At the second phase** we mapped every LAR (NLG) into an LDD, so the circuit is represented by a set of LDDs. We utilize the state-of-the-art DD technique, word-level DDs and Zero-Suppressed DDs. The LARs and LDDs are the equivalent word-level models of the NLG.

**Comments.** There are two main difficulties to realize this strategy: (i) the well-known fact that the values of weights in LAR are of  $O(m^n)$ , i.e. for real circuits these values are astronomical; (ii) a problem of connection of a set of LDDs representing a given multi-level circuit.

## 4 NLG design

We utilize the beneficial idea by [1] and its further development [12] to build LAR model of NLG. Note, in [4] there were introduced fundamental results on linearization of ARs of Boolean functions.

Let us partition the truth column vector  $\mathbf{X}$  of an  $n$ -variable  $m$ -valued function  $f$  into  $\tau = \lceil m^n / (n+1) \rceil$  parts, where  $\lceil a \rceil$  is the nearest integer number greater than  $a$ . The order of partition is fixed (with respect to assignments of variables). Let us call  $\mu$ -th part of  $\mathbf{X}$  the **sub vector (function)  $\mathbf{X}_\mu$** ,  $\mu = 0, 1, \dots, \tau - 1$ . The index  $\mu$  of the  $\mathbf{X}_\mu$  that contains a given  $i$ -th element of the truth table  $\mathbf{X}$  is equal to  $\mu = \lfloor i / (n+1) \rfloor$ , where  $\lfloor a \rfloor$  is the nearest integer number lower than  $a$ . In particular,  $\mathbf{X}$  for an arbitrary 3-input gate includes  $3^3 = 27$  elements and can be partitioned into

$\tau = \lceil 3^3/(3+1) \rceil = \lceil 27/4 \rceil = \lceil 6.7 \rceil = 7$  sub-vectors  $\mathbf{X}_0, \mathbf{X}_1, \dots, \mathbf{X}_6$ . The  $i$ -th element ( $i = 20$ ) of the  $\mathbf{X}$  is located in the sub-vector  $\mathbf{X}_5$ , since  $\mu = \lfloor 20/(3+1) \rfloor = 5$ .

**Table 1. LAR model to implement the ternary MAX( $x_1, x_2$ ) gate**

Function MAX			LAR model		
$x_1x_2$	$\mathbf{X}$	$\mathbf{X}_\mu$	$x_1^\circ x_2^\circ$	$\mathbf{P}_\mu$	LAR
00	$\begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}$	$\mathbf{X}_0 = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}$	00	$\mathbf{P}_0 = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} = x_2^\circ + 2x_1^\circ$	
01			01		
02			10		
10	$\begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$	$\mathbf{X}_1 = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$	00	$\mathbf{P}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = 1 + x_1^\circ$	
11			01		
12			10		
20	$\begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$	$\mathbf{X}_2 = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$	00	$\mathbf{P}_2 = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} = 2$	
21			01		
22			10		

**Example 1.** Let us find a LAR to describe the ternary MAX( $x_1, x_2$ ) gate with truth vector  $\mathbf{X}$  (Table 1). After partitioning of a column-truth vector into three sub-vectors  $\mathbf{X}_\mu$ ,  $\mu = 0, 1, 2$ , where  $\tau = \lceil 3^2/(2+1) \rceil = 3$ , we calculate a set of LARs. The so-called truth-matrix is derived from  $\mathbf{X}_\mu$  as follows

$$\mathbf{X} = [\mathbf{X}_2 | \mathbf{X}_1 | \mathbf{X}_0] = \begin{bmatrix} 2 & 1 & 0 \\ 2 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix}. \quad (4)$$

Then, the matrix can be transformed into the truth vector  $\mathbf{X}^\circ$  by multiplying it by a weight vector  $\mathbf{W} = [3^2 \ 3^1 \ 3^0]^T$ , i.e.

$$\mathbf{X}^\circ = \mathbf{X}\mathbf{W} = \begin{bmatrix} 2 & 1 & 0 \\ 2 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix} \begin{bmatrix} 3^2 \\ 3^1 \\ 3^0 \end{bmatrix} = \begin{bmatrix} 21 \\ 22 \\ 26 \end{bmatrix}. \quad (5)$$

**Lemma 1.** The sub-function  $\mathbf{X}_\mu$ ,  $\mu = 0, 1, \dots, \tau - 1$ , corresponds to a LAR  $\mathbf{P}_\mu$ , of  $n$  binary variables  $x_1^\circ, x_2^\circ, \dots, x_n^\circ$  (called **pseudo variables**), where at most one variable takes the value 1.

Table 1 shows the encoding and decoding rules that follow from the above Lemma 1. The relationship between sub-vector  $\mathbf{X}_\mu$  and the LAR coefficient vector  $\mathbf{P}_\mu$  is specified by the pair of arithmetic transforms

$$\mathbf{P}_\mu = \mathbf{K} \cdot \mathbf{X}_\mu, \quad \mathbf{X}_\mu = \mathbf{K}^{-1} \cdot \mathbf{P}_\mu, \quad (6)$$

where the  $3 \times 3$  forward  $\mathbf{K}$  and inverse  $\mathbf{K}^{-1}$  truncated matrices are defined as

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad \mathbf{K}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

**Example 2.** Let us find an LAR expression to describe the two-input MAX gate (Example 1). In Table 1, we give the partitioned column truth vector as a set of sub-vectors  $\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2$ , and the corresponding LAR coefficient vectors  $\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2$  for the assignments  $\{x_1^\circ, x_2^\circ\} = \{00, 01, 10\}$ . By (6),

$$\mathbf{P}_0 = \mathbf{K} \cdot \mathbf{X}_0 = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix},$$

i.e.  $P_0 = x_2^\circ + 2x_1^\circ$ . Also,  $P_1 = 1 + x_1^\circ$  and  $P_2 = 2$ . The weighted sum of these LAR expressions produces the expression for MAX( $x_1, x_2$ ) in symbolic and matrix form (5) respectively

$$P = 3^0 P_0 + 3^1 P_1 + 3^2 P_2 = 21 + 5x_1^\circ + x_2^\circ,$$

$$\mathbf{P} = \mathbf{K} \cdot \mathbf{X}^\circ = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 21 \\ 22 \\ 26 \end{bmatrix} = \begin{bmatrix} 21 \\ 1 \\ 5 \end{bmatrix}.$$

The below Karnaugh maps illustrate the issue:

$x_1$		0	1	2
$x_2$	0	0	1	2
	1	1	1	2
	2	2	2	2

$x_1^\circ$		0	1
$x_2^\circ$	0	21	26
	1	22	-

$\mu$		0	1	2
$x_1^\circ x_2^\circ$	00	0	1	2
	01	1	1	2
	10	2	2	2

The next example illustrates the restoration procedure.

**Example 3.** Given the LAR for NLG,  $P = 21 + 5x_1^\circ + x_2^\circ$ , that represents MAX( $x_1, x_2$ ) gate, calculate MAX( $x_1, x_2$ ) = MAX(2, 1). Based on the encoding rule given in Table 1,  $x_1x_2 = 21$  is equivalent to  $x_1^\circ x_2^\circ = 01$ . In order to find the sub-vector  $\mathbf{X}_\mu$ , let us calculate  $\mu$ : since the variable assignments  $x_1x_2 = 21$  correspond to the 7-th element of the truth vector  $\mathbf{X}$ ,  $\mu = \lfloor 7/3 \rfloor = 2$ .

The LAR  $P$  satisfies the model (2) of an NLG, i.e.  $P = D$ . So, taken  $\xi = \mu$  in (2) and the masking operator (3), we are able to define an arbitrary 3-valued 2-input NLG.

**Definition 5.** A 3-valued ( $m = 3$ ) two-input ( $n = 2$ ) NLG is an element with the output

$$f = \Xi^\mu \{P\} = \left\lfloor \frac{P}{3^\mu} \right\rfloor \bmod 3, \quad (7)$$

where  $P = d_1x_1^\circ + d_2x_2^\circ + d_0$  (2).

The below example illustrates how to calculate any value of the MVL function via the NLG model (7) of the MVL circuit.

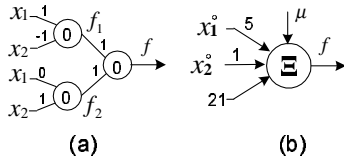
**Example 4.** Calculation of  $MAX(2, 1)$  is equivalent to the computation of  $f = \Xi^\mu\{P\}$ . Since  $x_1^\circ x_2^\circ = 01$ , then  $\Xi^2\{21 + 5x_1^\circ + x_2^\circ\} = \Xi^2\{22\} = \lfloor \frac{22}{3^2} \rfloor \bmod 3 = 2$ .

Table 2 contains the LAR-based representations of NLG of the ternary gates from a typical library of gates. The new models can be formed using these basic ones. For example,  $\overline{x_1 + x_2} \pmod 3 = \Xi^\mu\{3^0(2 - (x_2^\circ + 2x_1^\circ)) + 3^1(2 - (1 + x_2^\circ - x_1^\circ)) + 3^2(2 - (2 - 2x_2^\circ - x_1^\circ))\} = \Xi^\mu\{5 + 14x_2^\circ + 10x_1^\circ\}$ .

**Table 2. LAR-based models of elementary ternary gates**

$\bar{x}$	$=$	$2 - x$
$x_1 \cdot x_2 \pmod 3$	$=$	$\Xi^\mu\{15x_1^\circ + 21x_2^\circ\}$
$MIN(x_1, x_2)$	$=$	$\Xi^\mu\{21x_1^\circ + 12x_2^\circ\}$
$TSUM(x_1, x_2)$	$=$	$\Xi^\mu\{21 + 5x_1^\circ + 4x_2^\circ\}$
$MAX(x_1, x_2)$	$=$	$\Xi^\mu\{21 + 5x_1^\circ + x_2^\circ\}$
$TPROD(x_1, x_2)$	$=$	$\Xi^\mu\{21x_1^\circ + 9x_2^\circ\}$
$(x_1 + x_2) \pmod 3$	$=$	$\Xi^\mu\{21 - 10x_1^\circ - 14x_2^\circ\}$
$x_1   x_2$	$=$	$\Xi^\mu\{1 - x_1^\circ + 5x_2^\circ\}$

Let us compare the MV-TG [9] and the proposed NLG, for the ternary MAX gate. Meanwhile, Tan's model includes three MV-TGs (Figure 1a). The proposed NLG with pseudo-variables as inputs is given in Figure 1b. Table 3 explains the computations in both cases.



**Figure 1. MAX gate implementation**

**Remark.** The inputs of NLG are pseudo-variables. It means that in all cases we need encode the multiple-valued assignments by the binary ones. Although the process of computation pseudo-variables and parameter  $\mu$  is complicated in general case, for a 2-input ternary gate it is trivial, as values of  $x_1^\circ$  and  $x_2^\circ$  depend only on the variable  $x_2$  and  $\mu$  is equal to a value of the variable  $x_1$  (see Example 2).

**Table 3. Truth table of MV-TG and NLG**

$x_1 x_2$	MV-TG [9]: Equation (1)						NLG: $f = \Xi^\mu\{D\}$			
	$L_1$	$L_2$	$L$	$f_1$	$f_2$	$f$	$D$	$\mu$	$0$	
0 0	0	0	0	0	0	0	21	2	1	0
0 1	-1	1	1	0	1	1	22	2	1	1
0 2	-2	2	2	0	2	2	-	-	-	-
1 0	1	0	1	1	0	1	26	2	2	2
1 1	0	1	1	0	1	1	-	-	-	-
1 2	-1	2	2	0	2	2	-	-	-	-
2 0	2	0	2	2	0	2	-	-	-	-
2 1	1	1	2	1	1	2	-	-	-	-
2 2	0	2	2	0	2	2	-	-	-	-

## 5 Representation of a set of ternary gates by a single NLG

Let us treat a level in an MVL circuit as an  $n$ -input  $r$ -output function.

**Definition 6.** The LAR expression of an  $n$ -input  $r$ -output level of a ternary circuit ( $r$ -gate level) is the weighed sum of LARs of each output  $f_j$ ,  $j \in \{1, \dots, r\}$ ,

$$L = \sum_{j=1}^r (3^{3(j-1)} D_j). \quad (8)$$

The below Lemma follows from Definition 8.

**Lemma 2.** An  $n$ -input  $r$ -output level of an MVL circuit corresponds to an NLG whose formal model is expressed by equation (8).

Thus, a particular case, ternary NLG, is described as follows.

**Definition 7.** A 3-valued ( $m = 3$ )  $n$ -input  $r$ -output NLG is an NLG which  $j$ -th output,  $j \in \{1, \dots, r\}$ , is derived from (3) taken  $\xi = 3(j-1) + \mu$

$$f_j = \Xi^{3(j-1)+\mu}\{L\} = \left\lfloor \frac{L}{3^{3(j-1)+\mu}} \right\rfloor \bmod 3. \quad (9)$$

**Example 5.** Find a word-level expression to represent a circuit level consisting of three MAX gates. Based on the word-level description of MAX gate (Example 1):  $P = 3^0(21 + 5x_1^\circ + x_2^\circ) + 3^3(21 + 5x_3^\circ + x_4^\circ) + 3^6(21 + 5x_5^\circ + x_6^\circ) = 15897 + 5x_1^\circ + x_2^\circ + 135x_3^\circ + 27x_4^\circ + 3645x_5^\circ + 729x_6^\circ$ .

The correspondence of the assignments of variables and pseudo-variables is given below:  $x_1 x_2 x_3 x_4 x_5 x_6 = 20 \ 11 \ 12 \implies x_1^\circ x_2^\circ x_3^\circ x_4^\circ x_5^\circ x_6^\circ = 00 \ 01 \ 10$  and  $\mu_1 = 2$ ,  $\mu_2 = 1$ ,  $\mu_3 = 1$ . For this assignment,  $P = 15897 + 5 \cdot 0 + 1 \cdot 0 + 135 \cdot 0 + 27 \cdot 1 + 3645 \cdot 1 + 729 \cdot 0 = 19569$ . Then, the outputs  $f_j$ ,  $j \in \{1, 2, 3\}$ , of the level are calculated as follows

$$\begin{aligned}
f_1 &= \Xi^{3 \cdot 0 + 2} \{19569\} = \left\lfloor \frac{19569}{3^2} \right\rfloor \bmod 3 = 2, \\
f_2 &= \Xi^{3 \cdot 1 + 1} \{19569\} = \left\lfloor \frac{19569}{3^4} \right\rfloor \bmod 3 = 1, \\
f_3 &= \Xi^{3 \cdot 2 + 1} \{19569\} = \left\lfloor \frac{19569}{3^7} \right\rfloor \bmod 3 = 2.
\end{aligned}$$

In particular, let us compare different implementations of the 4-variable parity function: the AND/OR network, TG network and NLG network given in Figure 2a, b and c accordingly. Notice, the NLG network has same architecture for any  $m$ .

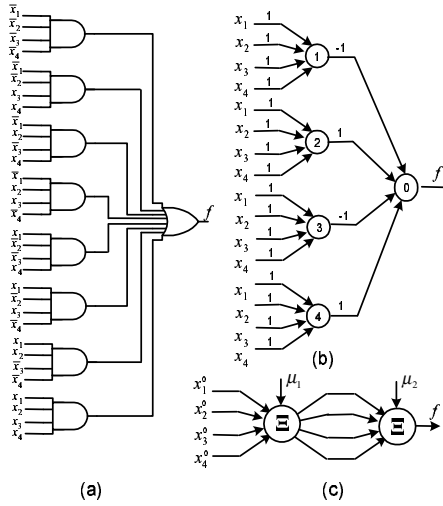


Figure 2. Parity function network architecture

## 6 LDD representation of the NLN

The second stage of the NLG design by an LDD with arithmetic analog of the Positive Davio  $pD_A$  expansion in the nodes [12]. Given  $LAR = 21 + 5x_1^\circ + x_2^\circ$  for the ternary MAX gate, its LDD is shown in Figure 3.

**Lemma 3.** *An NLN model of an  $l$ -level MVL circuit consists of  $l$  NLGs, each described by a LAR and a corresponding LDD.*

The proof follows from Lemma 2 and equivalence of both the representations of an MVL circuit: the LAR and the LDD derived from the LAR. Consequently, a circuit of  $l$  levels is described by  $l$  LDDs.

Figure 3 shows the ternary NLGs derived from the LAR and LDD-based models.

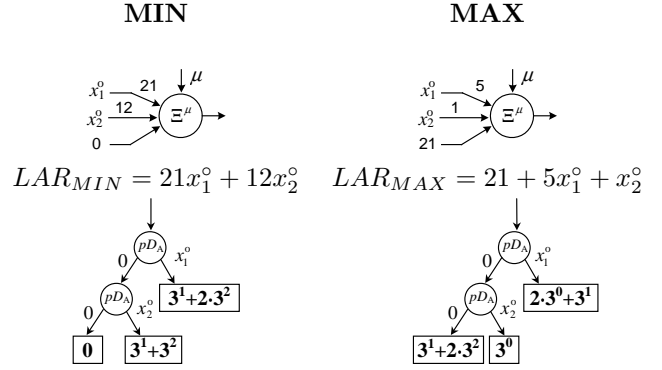


Figure 3. NLGs for MIN and MAX gates

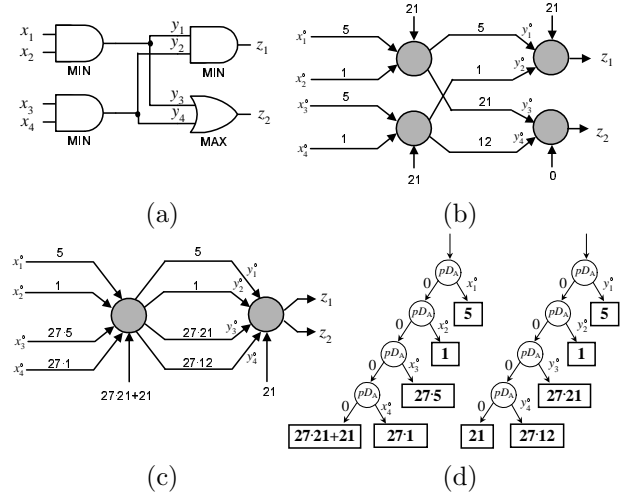


Figure 4. Models of a 2-level ternary circuit

Now let us consider the two level ternary network given in Figure 4a. Following the approach of [2, 9, 10], we can represent this circuit by the MV-TG network (Figure 4b) where each node includes three MV-TGs. By our technique, we design an NLN with two NLGs that are mapped into two LDDs (Figure 4c and 4d accordingly). Notice that the resulting NLN has no learning capability.

We introduce our main result by the theorem.

**Theorem 4.** *An arbitrary  $l$ -level MVL circuit over a standard library of cells can be represented by an NLN with  $l$  NLGs; every NLG is formally described by a LAR that is mapped into an LDD, so that the resulting NLN is mapped into  $l$  LDDs.*

The proof follows from Lemma 3.

Thus, the number of NLGs in an NLN is equal to the number of levels in the corresponding circuit, that is the initial MVL circuit is mapped into the NLN level by level in opposite to the MV-TG approach.

## 7 Algorithm

The algorithm to derive an LDD-based model from a circuit description is given in Figure 5. One of the painful problems of the word-level representation, including linear forms, is the exponential values of terminal nodes. To calculate it, we utilize the Zero-Suppressed BDD-like trees [5]. We developed a special encoding scheme in order to obtain reasonable memory requirements. In a case of LDD built for ternary gates, each terminal node is a sum of three integer exponential numbers, where mantissa takes a value from a set  $\{-2, -1, \dots, 2\}$  and the exponentials are three subsequent integers. Moreover, as we use circuits with 2 inputs only, two adjoining terminal nodes in the  $2i$ -th and  $2i + 1$ -th LDD levels have the same components,  $3^{3i}$ ,  $3^{3i+1}$  and  $3^{3i+2}$ . Consequently, there is no need of keeping the huge exponential numbers, as they can be simply calculated from the given tree level and, thus, each terminal node is fully described by a set of three mantissas. In this way, we replace the calculation by *manipulating* the encoded bits of numbers (for more details, see [12]). We also utilized ideas of links between the cells in linear systolic arrays to processing the information by a set of LDDs [3].

```

procedure buildLDD(circuit):
for( $\forall$  level i in the circuit)
{
  Design LDD for the level i;
  for( $\forall$  gate g in the level i)
    for( $\forall$  input in of the gate g)
      if(no input in in the LDD for the level i) then
        add a new node with a pseudo-variable
        corresponding to the given input;
      else
        add a new terminal value to the existing node
        with the pseudo-variable corresponding to the given input;
  return a set of LDDs;
}
procedure buildNLN(circuit):
Execute buildLDD(circuit) and keep the result in a variable trees;
for( $\forall$  tree i in trees)
{
  Form the i-th NLG;
  for( $\forall$  node j in the tree i)
    add to the i-th NLG an input with
    · a splitting variables from the j-th node,
    · a weight equal to the j-th terminal node;
  add values from free terminal node of the tree i to the bias of the i-th NLG;
}

```

**Figure 5. The algorithm to generate the NLN**

## 8 Experiments

We simulated the NLNs of the benchmark circuits (with about 5000 gates) from LGSynth 93<sup>1</sup> base. Every binary gate in a circuit is interpreted as ternary gate, i.e. AND gate is replaced with MIN gate, and OR gate is replaced with MAX gate. Then, we represent each MVL gate by a LAR model of an NLG. Finally, we mapped this model to the NLN. Run time was measured in seconds for PC Pentium III 450 MHz processor with 128 MB RAM. Our program *NetDesign* was written in C++.<sup>2</sup> The experimental results are given in Tables 4 - 6.

The columns 1-5 in Table 4 contain the name of testing circuit, the number of inputs/outputs (**I/O**), the number of gates (**#G**), the number of circuit levels (**#L**) and the total number of nodes (**#N**) in LDDs.  $N_{max}$  denotes the number of nodes of the largest LDD. The parameter  $W_{max}$  that is the maximal number of digits in a terminal node value. The last column contain run time (**CPU**) in miliseconds to simulate a circuit via a set of LDDs. Some observations follow from Table 4.

(i) The number of nodes (**#N**) in LDDs is comparable with the number of gates (**#G**) in the circuit. For example, 207-input 109-output circuit C7552 that includes 4094 gates is represented by 66 LDDs (**#L**=66) with **#N**=5935 nodes. This is because of complexity  $O(\#G)$  of our algorithm.

(ii) Run time (**CPU**) to transform circuit from EDIF format to a system of LDDs is acceptable for practice.

This proves the efficiency of our algorithm to generate NLN models, which manipulates, for instance, a model of the 32-digit multiplier (C6288), LDDs with the terminal values that are the numbers of about 800 ternary digits.

In Table 5 we give comparison of average CPU time to calculate outputs of the circuits in EDIF format (**CPU**) and LDD format **CPU1** for 10 000 randomly generated input assignments. We observe that **CPU**  $\approx$  **CPU1**.

In Table 6 we give the memory requirements for the EDIF and ISCAS formats (**MEM**) compared to the proposed LDD representation (**MEM1**): **MEM1** is about 5-20 times less than **MEM**.

### Concluding remarks

We have proposed the fast, simple, and accurate model for the large MVL networks (in our experiment,

<sup>1</sup>[http://zodiac.cbl.ncsu.edu/CBL\\_Docs/lgs93.html](http://zodiac.cbl.ncsu.edu/CBL_Docs/lgs93.html)

<sup>2</sup>The package is available from the authors

**Table 4. Characteristics of LDD-based NLN**

Name	T E S T			LDD-based network		
	I/O	#G	#L	#N( $N_{max}$ )	$W_{max}$	CPU [s]
C17	5/2	12	6	23(5)	6	0.00
C432	36/54	319	28	503(46)	63	0.04
C499	41/32	524	25	880(105)	200	0.15
C880	60/32	619	39	860(59)	148	0.10
C1355	41/32	908	43	1354(129)	192	0.25
C1908	33/25	765	54	1140(71)	101	0.12
C2670	233/108	1234	45	1715(228)	372	0.91
C3540	50/24	1670	65	2397(202)	311	1.14
C5315	178/163	2910	78	3977(356)	646	2.06
C6288	32/32	4768	245	6917(256)	768	1.43
C7552	207/109	4094	66	5935(407)	561	3.69

**Table 5. Comparison of RUN time**

Test	IN	#L	CPU1 [ms]	CPU2 [ms]
C17	5	6	0.002	0.004
C432	36	28	0.109	0.150
C499	41	25	0.148	0.168
C880	60	39	0.188	0.339
C1355	41	43	0.241	0.288
C1908	33	54	0.247	0.252
C2670	233	45	0.379	0.491
C3540	50	65	0.447	0.593
C5315	178	78	1.240	1.580
C6288	32	245	1.100	2.194
C7552	207	66	2.011	2.340

**Table 6. Comparison of memory requirements**

TEST	EDIF format			ISCAS format		
	#G	MEM	MEM1	#G	MEM	MEM1
C17	12	5478	258	6	1303	201
C432	433	102257	6314	160	18991	4601
C499	516	167343	15301	202	21989	5427
C880	619	190729	8869	383	37844	8306
C1355	1204	273018	16133	546	59573	16850
C1908	2134	230507	13098	880	78574	22491
C2670	2603	400707	32395	1193	110025	36305
C3540	3901	503457	34744	1669	145359	54857
C5315	6018	903899	88965	2307	220596	107392
C6288	4847	1387230	71476	2416	255406	68572
C7552	8067	1236066	146057	3512	311939	271970

more than 200 inputs and 8000 ternary gates). The crucial idea was to utilize the advantages of word-level description of logic functions. We have proved that the proposed model (with no learning capability) is more powerful compared to [2, 9, 10]. Moreover, we have shown that the boundary case of the word-level expressions, LARs allow an arbitrary MVL circuit to be represented by a set of the LDDs planar by their nature. In particular, to represent the ternary 32-digit

multiplier, we derive 245 LDDs with 6917 nodes for 1.43 seconds. We believe that NLN, in general, might be used to alleviate some relevant problems of planar MVL network design.

**Acknowledgment.** The authors would like to thank *Prof. G. W. Dueck* (University of New Brunswick, Canada) for his help and critical remarks at all stages of this research, and also the anonymous reviewers for their helpful comments.

## References

- [1] V. Antonenko, A. Ivanov, V. Shmerko, Linear Arithmetical Forms of  $k$ -valued Logic Functions and their Realization on Systolic Arrays, *Automation and Remote Control (USA)*, vol. 56, no. 3, Pt. 2, 1995, pp. 419 - 432
- [2] T. Hozumi, N. Kamiura, Y. Hata, K. Yamato, On Minimization of Multiple-Valued Sum-of-Products Expression with Multiple-Valued TRSUM, *Int. Journal on Multiple-Valued Logic*, vol. 2, 1997, pp. 141 - 158
- [3] G. Kukharev, A. Tropchenko, V. Shmerko, *Systolic Signal Processors*, Publishing House "BELARUS", Minsk, Belarus, 1988 (in Russian)
- [4] V. Malyugin, Realization of Boolean Function'S Corteges by Means of Linear Arithmetical Polynomial, *Automation and Remote Control (USA)*, vol.45, no. 2, Pt. 1, 1984, pp. 239-245
- [5] S. Minato, *Binary Decision Diagrams and Applications for VLSI CAD*. Kluwer Academic Publishers, 1996
- [6] S. Muroga, *Threshold Logic and its Applications*. Wiley-Interscience, NY, 1971
- [7] T. Sasao, J. T. Butler, Planar Decision Diagrams for Multiple-valued Functions, *Int. Journal on Multiple-Valued Logic*, vol. 1, 1996, pp. 39 - 64
- [8] K-Y. Siu, V. P. Roychowdhury, T.Kailath, Depth-Size Tradeoffs for Neural Computation *IEEE Trans. on Computers*, vol. C-40. no 12, 1991, pp. 1402 - 1411
- [9] Z. Tang, Q. Cao, O. Ishizuka, A Learning Multiple-Valued Logic Network: Algebra, Algorithm and Applications, *IEEE Trans. on Computers*, vol. 47, no. 2, 1998, pp. 247 - 251
- [10] G. Wang, H. Shi, TMLNN: Triple-Valued or Multiple-Valued Logic Neural Network, *IEEE Trans. on Neural Networkes*, vol. 9, no. 6, 1998, pp. 1099 - 1117
- [11] J. Webster (Ed.), *Encyclopedia of Electrical and Electronics Engineering. Threshold logic*. John Wiley & Sons, vol. 22, 1999, pp. 178 - 190
- [12] S. Yanushkevich, V. Maluygin, P. Dziurzanski, V. Shmerko, Linear Models of Multiple-Valued Networks, *Automation and Remote Control (USA)*, 2002, accepted