A Framework for Design of Multivalued Logic Functions and Its Application Using CMOS Ternary Switches

Mudhafar A. Hassan Ali, *Student Member, IEEE*, Falah J. Hassan-Alshiroofi, and Hemant G. Rotithor, *Senior Member, IEEE*

***Abstract*— Two important problems in implementing multivalued logic (MVL) as compared to binary logic (BL) are the lack of an efficient logic minimization technique and larger chip area and power consumption for realizing an MVL function. In this paper, a new theory for the implementation of MVL is proposed in an attempt to address these problems. In the proposed theory, an MVL function is decomposed into a set of subfunctions that can be efficiently realized using *switches* that are multivalued-in-nature. Each switch consists of a group of more elementary switches, called *subswitches.* A complete set of algebraic operators and relations are presented to facilitate the construction of the switches from a set of subswitches. The proposed algebra can be used to minimize the number of subswitches required for each switch in a function realization. Application of the proposed theory to implementing a ternary logic (TL) truth table is illustrated which is expected to encourage further investigation into exploring the possibility of using TL as a competitor to BL. The realization of subswitches is done using CMOS transistors that has potential for a VLSI implementation which can have a small chip area and consume low power.**

I. INTRODUCTION

RECENTLY, MVL has been a subject of considerable study [1]–[13]. In particular, a significant body of the research has addressed the problem of MVL realization [4]–[13] with the goal of competing with the firmly established BL for design of digital systems. BL techniques have been mainly adopted for solving MVL implementation problems [9]–[11]. All such binary techniques have used gates built from binary-in-nature devices as building blocks. However, an efficient implementation of MVL can exist only when devices used in MVL gate design are multivalued-in-nature by including more operational states in each device in order to reduce the hardware required for realization [2]. In other MVL system designs, a cascade of MVL decoder-BL-MVL encoder has been used to emulate an MVL function [9]–[11], [13]. Since this cascade has a binary component, the implementation is not fully MVL, and is thus not an efficient method. The circuit realization of MVL gates using 1^2L, charge-coupled devices (CCD), voltage-mode CMOS (VMCL), current-mode CMOS (CMCL), and dynamic CMOS technologies was reported in the literature [4]–[8]. In most of these cases, however, the MVL circuits require CMOS transistors with a different geometry; i.e., width: length (W : L) ratio, which makes circuit fabrication complex [4]. Unlike the binary CMOS gates that virtually have zero static current, these MVL circuits have a static current which dissipates power [7].

Apart from the gate design dilemma, MVL implementation still lacks efficient minimization algorithms because of its algebraic complexity [2]. Therefore, utilizing general purpose gates as reported in [1]–[4] that collectively are: ***{min, mar, cyclic operator,*** and ***window literal)*** might not yield an efficient design in terms of the hardware requirements.

In order to address these problems, we need an implementation using MVL devices and a technique for logic minimization which can make the resulting design easy to implement and efficient in terms of the chip area and power consumption. The objective of this paper is to present such an approach.

First, the concept of MVL ***switches*** is introduced in order to form a basis for implementing any MVL function by replacing MVL gates by switches. It is shown that a MVL function of radix R can be decomposed into no more than R subfunctions. Each subfunction can be realized by a specific switch which is shown to consist of a group of more elementary switches, called ***subswitches.*** Each subswitch is synthesized from multivalued devices that are interconnected. An algebra that governs the interconnections of the subswitches is proposed. Logic minimization techniques using this algebra are also described.

TL is presented as a case study to demonstrate potential application of the theory. TL was selected because it was theoretically shown in [2] to be an optimum radix in terms of the hardware requirement. CMOS transistor switches are designed to realize a ternary function (TF) based on the proposed theory. All CMOS transistors in this design have identical geometry, unlike other techniques [4]–[8], [12], where a different geometry of the transistors was required. Using the proposed algebra, a technique for implementing a design using a minimum number of transistors is shown. Further, since CMOS requires virtually zero-static current, low power dissipation can be attained.

In Section II, the switch theory is stated. The algebra of the subswitches and the types of the interconnection between them are described in Section III. Section IV discusses the imple-

Manuscript received January 24, 1995; revised July 18, 1995. This paper was recommended by Associate Editor D. Haigh.
M. A. Hassan Ali and H. G. Rotithor are with the Electrical and Computer Engineering Department, Worcester Polytechnic Institute, Worcester, MA 10609 USA.
F. J. Hassan-Alshiroofi was with the Department of Electronics Engineering, University of Seventh of April, Sabrata, Libya.
Publisher Item Identifier S 1057-7122(96)02254-4.

mentation of the Ternary Logic using only CMOS transistors. Our conclusions are presented in Section V.

II. Switches in MVL

In a MVL of any Radix R (where $R \geq 2$), the set of MVL elements is $S = \{0, 1, 2, \ldots, R-1\}$. The following definition shows how a MVL function can be expressed algebraically.

Definition 1: Let $f(x_1, x_2, \ldots, x_n)$ be an MVL function where $(x_1, x_2, \ldots, x_n)$ (designated as X) is the variable vector; and $f(x_1, x_2, \cdots, x_n)$ and $x, \in S$.

$f(x_1, x_2, \ldots, x_n)$ can be expressed as

$$f(x_1, x_2, \cdots, x_n) = \sum_{i=0}^{R-1} i * \hat{F}_i(x_1, x_2, \cdots, x_n). \quad (1)$$

Where $\sum$ is the logical OR {*max* operation), $*$ is the logical AND {*min* operation), and

$$\hat{F}_i(x_1, x_2, \ldots, x_n) = \begin{cases} & \text{if } (x_1, x_2, \cdots, x_n) \in \mathbf{T}_i \\ & \text{otherwise} \end{cases} \quad (2)$$

$\hat{F}_i$ acts as a *threshold* operator similar to the notion of literals as defined in [14]. Its operation is to activate; i.e., to give the highest value R, only when the variable vector $X = (x_1, x_2, \cdots, x_n)$ is an element in the threshold set $\mathbf{T}_i$. Here, T, is a set of variable vectors at which the function $f(x_1, x_2, \cdots, x_n) = i$; i.e., $\mathbf{T}_i = \{\mathbf{X}'_{i,1}, \mathbf{X}'_{i,2}, \cdots, \mathbf{X}'_{i,L_i}\}$ where $\mathbf{X}'_{i,l_i} = (x'_{i,l_i,1}, x'_{i,l_i,2}, \cdots, x'_{i,l_i,n})$ for $l_i = 1, 2, \ldots, L_i$, and $\sum_{i=0}^{R-1} L_i = R^n$. Equation *(1)* decomposes the function $f(x_1, x_2, \ldots, x_n)$ into R mutually exclusive subfunctions $\hat{F}_i$. For example: Table I shows a ternary function $f(x_1, x_2)$ which can be decomposed according to (1) as follows:

$$\begin{aligned} f(x_1, x_2) &= \sum_{i=0} i * \hat{F}_i(x_1, x_2) \\ &= 0 * \hat{F}_0(x_1, x_2) + 1 * \hat{F}_1(x_1, x_2) \\ &\quad + 2 * \hat{F}_2(x_1, x_2) \end{aligned}$$

where

$$\hat{F}_0(x_1, x_2) = \begin{cases} 3 & \text{if } (x_1, x_2) \in \mathbf{T}_0 = \{(0, 1), (1, 1), (2, 2)\} \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{F}_1(x_1, x_2) = \begin{cases} 3 & \text{if } (x_1, x_2) \in \mathbf{T}_1 = \{(2, 0), (0, 2)\} \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{F}_2(x_1, x_2) = \begin{cases} 3 & \text{if } (x_1, x_2) \in \mathbf{T}_2 = \{(0, 0), (1, 0), (2, 1), (1, 2)\} \\ 0 & \text{otherwise.} \end{cases}$$

Hence, $L_0 = 3$, $L_1 = 2$, and $L_2 = 4$ and $\mathbf{T}_1 = \mathbf{X}'_{1,1}, \mathbf{X}'_{1,2}$ where $\mathbf{X}'_{1,1} = (x'_{1,1,1}, x'_{1,1,2}) = (2, 0)$ and $\mathbf{X}'_{1,2} = (x'_{1,2,1}, x'_{1,2,2}) = (0, 2)$. Similarly, values of other threshold subsets T_0 and T_2 can be determined.

Alternatively, the decomposition of $f(x_1, x_2, \ldots, x_n)$ can be achieved by changing the threshold operators in (1) into switch subfunctions as defined below

TABLE I
A Ternary Function

x_1	x_2	$f(x_1, x_2)$
0	0	2
1	0	2
2	0	1
0	1	0
1	1	0
2	1	2
0	2	1
1	2	2
2	2	0

$$f(X) = \sum_{i=0}^{R-1} i \circ \tilde{F}_i(X). \quad (3)$$

In (3), the logical OR represented by $\sum$ is realized by a wired-OR circuit connection. The switch subfunction $\tilde{F}_i(X)$ is defined as follows:

$$\tilde{F}_i(X) = \begin{cases} \text{ON} & \text{if } X \in T_i \\ \text{OFF} & \text{otherwise} \end{cases} \quad (4)$$

$\tilde{F}_i(X)$ is the ith MVL switch subfunction, and $\circ$ is the transmission operator; i.e.,

$$i \circ \tilde{F}_i(X) = F_i(X) = \begin{cases} i & \text{if } X \in \mathbf{T}, \Rightarrow \tilde{F}_i(\mathbf{X}) \text{ is ON} \\ \text{OFF} & \text{if } X \notin \mathbf{T}, \Rightarrow \tilde{F}_i(\mathbf{X}) \text{ is OFF} \end{cases} \quad (5)$$

where $F_i(\mathbf{X})$ denotes the ith MVL switch.

Equation (5) gives a formal definition of an MVL switch that has three terminals: the control variable X, the desired value i, and the output $F_i(\mathbf{X})$. Each MVL switch represents one element of S in $f(X)$. Fig. *1* shows how (1) can be realized with MVL switches.

The internal structure of each switch F_i is decomposed into more elementary switches called subswitches. A subswitch is a switch with a single control input. One terminal holds a certain value $j \in S$ that is to be switched to the output terminal. The switching occurs only when the control input x is an element of the threshold set $t \subseteq S$. A formal definition of a subswitch is as follows:

Definition 2: A subswitch is characterized as

$$x^t(j) = \begin{cases} j & \text{if } x \in t \\ \text{OFF} & \text{otherwise} \end{cases} \quad \text{where } j \in S \text{ and } t \subseteq S. \quad \textbf{(6)}$$

where x represents the control input (single MVL variable), and j is the desired value to be switched to the output terminal when x is an element of the threshold set t. Fig. 2 schematically shows this definition. When t is a single element set, then, the subswitch is called a *unary* subswitch.

The following example illustrates the idea of a subswitch.

Example: Let $R = 4$, and x is a 4-valued logic variable; i.e., $x \in \{0, 1, 2, 3)$. The output of subswitch $x^{(0,3)}(2)$ is 2 when $x \in \{0, 3)$ otherwise it is OFF.

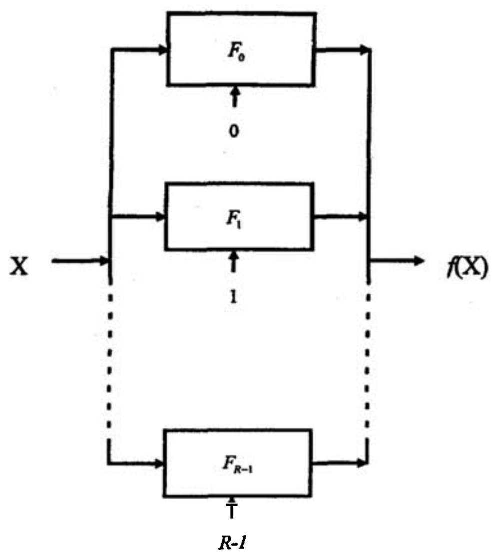


Fig. 1. Multivalued logic function realization.

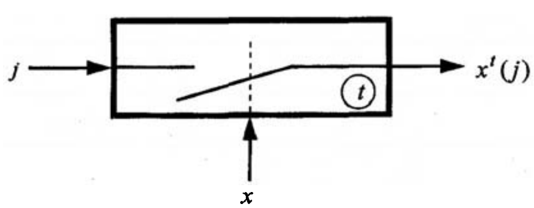


Fig. 2. Subswitch.

It should be noted that x, t, and j are variables each of which can be a function of other variables.

In the following section, more properties of the subswitches are shown through a set of algebraic relations.

III. Properties of Subswitches

In this section, interconnections between different subswitches are characterized. Since their threshold set t is single element, the unary subswitches are considered as the fundamental building blocks. First, let us answer the following question: how many different unary subswitches can be generated?

Theorem 1: There are R^2 different unary subswitches for the radix-R MVL.

Proof: Since t is a single element, it can take R different values. Similarly, x takes R values. Therefore, the total number of combinations obtainable is R^2.

A consequent question that may rise from Theorem 1 is are the R^2 **subswitches** enough to build any subswitch (where t is a multielement set)? An answer to this question is provided by different subswitch connection patterns described in Theorems 2–4.

Theorem 2. [Series Interconnection]: If x is an MVL input variable, and z is the output from the series interconnection of the subswitches $x^{t_1}(j)$ and $x^{t_2}(j)$, then,

$$z = x^{t_1}(j) \bullet x^{t_2}(j), \\ = x^t(j) \quad \text{where} \quad t = t_1 \cap t_2. \tag{7}$$

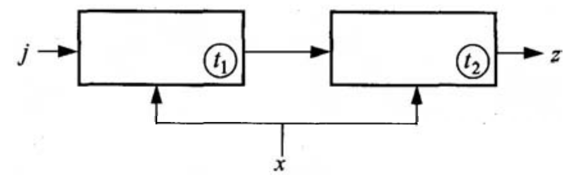


Fig. 3. A series interconnection.

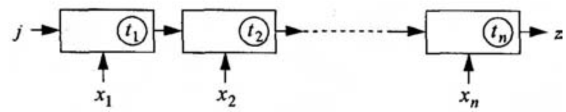


Fig. 4. Series **multivariable** interconnection

Here $\bullet$ denotes a series interconnection. Series interconnection is a commutative operation. The interconnection is depicted in Fig. 3.

Proof: By using Definition 2, the output x is equal to j only when both subswitch 1 and 2 are ON at the same time. Otherwise, the output is OFF. This is equivalent to having one subswitch with threshold set that includes the elements common in t_1 and t_2. Due to the intersection of the two threshold sets t_1 and t_2, this type of interconnection is used when the threshold set t is required to be of a reduced size.

Since $t_1 \cap t_2 = t_2 \cap t_1$, then $z = x^{t_1}(j) \bullet x^{t_2}(j) = x^{t_2}(j) \bullet x^{t_1}(j)$ which is a commutative operation.

Corollary 1: Theorem *1* can be extended to the case of multivariable function; **i.e.,**

$$z = x_1^{t_1}(j) \bullet x_2^{t_2}(j) \bullet \cdots \bullet x_n^{t_n}(j) \\ = \prod_{i=1}^{n} x_i^{t_i}(j)$$

$$z = \begin{cases} j & \text{if } x_1 \in t_1,\ x_2 \in t_2, \cdots, x_n \in t_n \\ \text{OFF} & \text{otherwise.} \end{cases} \tag{8}$$

Proof: Using Definition *2*, the output is equal to j only when all subswitches $x_1^{t_1}(j), x_2^{t_2}(j), \cdots$, and $x_n^{t_n}(j)$ are all ON simultaneously. This can happen only when $x_1 \in t_1$, $x_2 \in t_2, \ldots$, and $x_n \in t_n$ simultaneously. This corollary is depicted in Fig. 4.

When $x_1 = x_2 = \ldots = x_n$, (8) can be rewritten as follows:

$$z = \prod_{i=1}^{n} x_i^{t_i}(j) \\ = x^t(j) \quad \text{where} \quad t = t_1 \cap t_2 \cap \cdots \cap t_n. \tag{9}$$

Theorem 3 [Parallel Interconnection]: If x is an MVL input variable, and z is the output of the parallel interconnection of the subswitches $x^{t_1}(j)$ and $x^{t_2}(j)$, then

$$z = x^{t_1}(j) + x^{t_2}(j), \\ = x^t(j) \quad \text{where} \quad t = t_1 \cup t_2. \tag{10}$$

The $+$ sign denotes a parallel interconnection. This operation is also commutative. Fig. 5 shows this configuration schematically.

Proof: Since the two subswitches are connected in parallel and are receiving the same input x, the output z has j when one or both of the two subswitches are ON. This happens when either $x \in t_1$ or $\in t_2$, that is, $x \in t_1 \cup t_2$. Due to the union of the two threshold sets t_1 and t_2, this type of interconnection

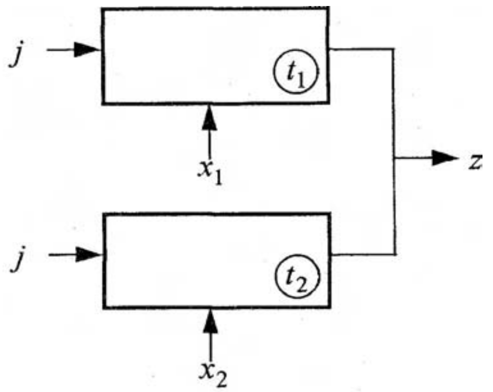


Fig. 5. A parallel interconnection.

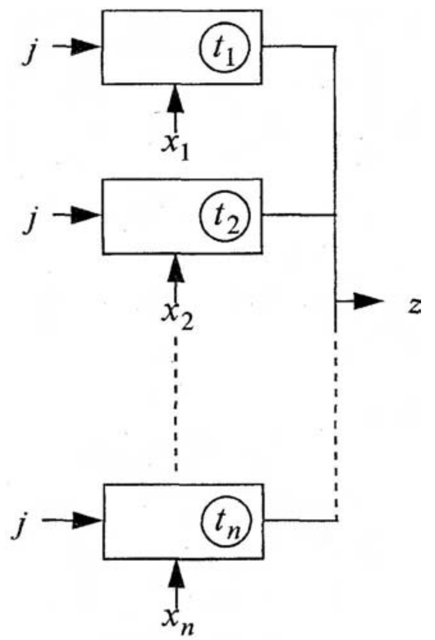


Fig. **6.** A parallel multivariable interconnection

is used when the threshold set t is required to be extended.

Since $t_1 \cup t_2 = t_2 \cup t_1$, then $z = x^{t_1}(j) + x^{t_2}(j) = x^{t_2}(j) + x^{t_1}(j)$ which is a commutative operation.

Corollary 2: Theorem **3** can be extended to the case of n subswitches; i.e.,

$$\begin{aligned} z &= x^{t_1}(j) + x^{t_2}(j) + \ldots + x^{t_n}(j) \\ &= \sum_{i=1}^{n} x^{t_i}(j) \\ &= x^{t}(j), \quad \text{where} \quad t = t_1 \cup t_2 \cup \ldots \cup t_n. \end{aligned} \tag{11}$$

Proof: A recursive application of Theorem 3 will result in the above corollary and is shown in Fig. 6.

It is evident from Theorem **3** as well as from Corollary 2 that all subswitches have the same desired value j. However, when the threshold sets t_i are mutually exclusive, a parallel interconnection of n subswitches is permissible even if their desired values j_i are different. The following corollary formulates this interconnection which acts as a MVL *multiplexer* (as seen in Fig. 7).

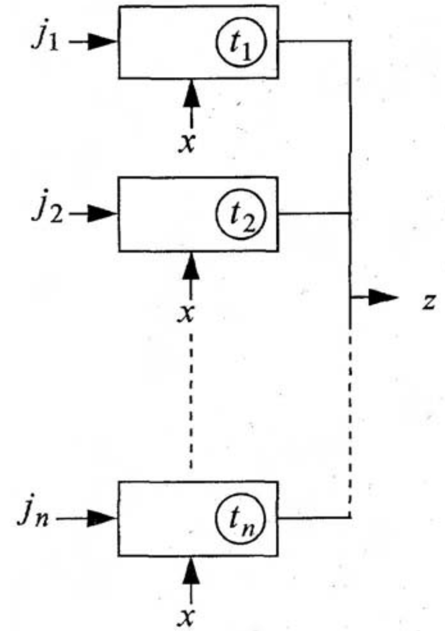


Fig. 7. **A** multiplexing interconnection.

Corollary 4 [Multiplexing Interconnection]: A parallel interconnection of n different subswitches $x^{t_i}(j_i)$ is permissible only when $t_{i_1} \cap t_{i_2} = 0$ where $i_1 \# i_2$, then

$$\begin{aligned} z &= \sum_{i=1} x^{t_i}(j_i) \\ &= \begin{cases} j_i & \text{if } x \in t_i \\ \text{OFF} & \text{otherwise} \end{cases} \\ &= x(j_1^{t_1}, j_2^{t_2}, \cdots, j_n^{t_n}). \end{aligned} \tag{12}$$

Proof: Since t, are mutually exclusive sets, one and only one subswitch will switch its desired value j_i to the output terminal z at a particular input x.

As a special case, a multiplexing interconnection of two subswitches $x^t(j)$ and $x^{\bar{t}}(\hat{j})$, where $\bar{t}$ is the complement set of t; i.e., $t \cup \bar{t} = S$ and $t \cap \bar{t} = \emptyset$. The *complementary* interconnection is depicted in Fig. 8 and expressed by

$$\begin{aligned} x^t(j, \hat{j}) &= x^t(j) + x^{\bar{t}}(\hat{j}) \\ &= \begin{cases} j & \text{if } x \in t \\ \hat{j} & \text{otherwise.} \end{cases} \end{aligned} \tag{13}$$

Since parallel interconnection is commutative, $x^t(j, \hat{j}) = x^{\bar{t}}(\hat{j}, j)$.

Theorem 4 [Cascade Interconnection]: If x is an MVL input variable, and z is the output from a cascade interconnection of the subswitches $x^{t_1}(j_1)$ (the input subswitch) and $x^{t_2}(j_2)$ (the output subswitch), then

$$\begin{aligned} z &= [x^{t_1}(j_1)]^{t_2}(j_2) \\ &= x^{t_1}(j_2) \quad \text{if} \quad j_1 \in t_2. \end{aligned} \tag{14}$$

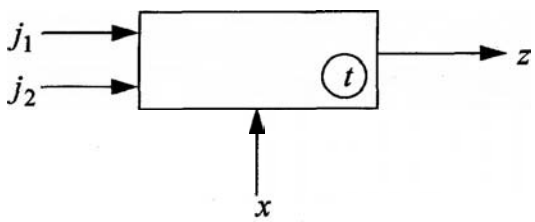


Fig. 8. **A** complementary subswitch.

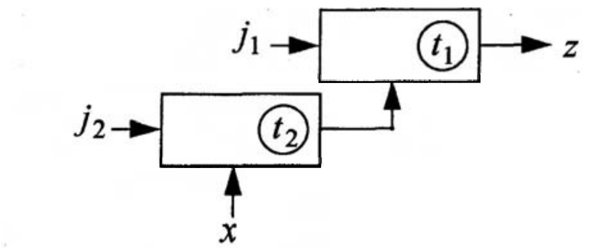


Fig. 9. **A** cascade interconnection.

Symbolically the above equation can be written as

$$\begin{aligned} z &= x^{t_1}(j_1) \# x^{t_2}(j_2) \\ &= x^{t_1}(j_2) \quad \text{if} \quad j_1 \in t_2 \end{aligned} \tag{15}$$

where # denotes the cascade interconnection. Fig. 9 shows this type of interconnection.

Proof: Let $\tilde{z} = x^{t_1}(j_1)$, and $z = \tilde{z}^{t_2}(j_2)$. According to Definition 2

$$\begin{aligned} z &= \tilde{z}^{t_2}(j_2) \\ &= \begin{cases} j_2 & \text{if } \tilde{z} \in t_2 \\ \text{OFF} & \text{otherwise} \end{cases} \end{aligned} \tag{16}$$

where

$$\tilde{z} = \begin{cases} j_1 & \text{if } x \in t_1 \\ \text{OFF} & \text{otherwise} \end{cases}$$

which, by transitivity, means

$$z = \begin{cases} j_2 & \text{if } x \in t_1 \text{ and } j_1 \in t_2 \\ \text{OFF} & \text{otherwise.} \end{cases}$$

It can be proved that the cascade interconnection is not commutative simply by exchanging the places of the two subswitches. The result will be as follows:

$$\begin{aligned} z &= x^{t_2}(j_2) \# x^{t_1}(j_1), \\ &= x^{t_2}(j_1) \quad \text{if} \quad j_2 \in t_1 . \\ &\neq x^{t_1}(j_2) \end{aligned}$$

It is seen that both the series and the parallel interconnections are commutative, whereas the cascade interconnection is non-commutative. In the following Theorem, the distributive law of interconnection is introduced.

Theorem 5 [Distributive Law]:

1) Series and Parallel Interconnections:

$$\begin{aligned} z &= x_1^{t_1}(j) \bullet x_2^{t_2}(j) + x_3^{t_3}(j) \bullet x_2^{t_2}(j) \\ &= [x_1^{t_1}(j) + x_3^{t_3}(j)] \bullet x_2^{t_2}(j) \end{aligned} \tag{17}$$

as special case when $x_1 = x_3$, then,

$$\begin{aligned} z &= x_1^{t_1}(j) \bullet x_2^{t_2}(j) + x_1^{t_3}(j) \bullet x_2^{t_2}(j) \\ &= [x_1^{t_1}(j) + x_1^{t_3}(j)] \bullet x_2^{t_2}(j) \\ &- x_1^{t_1 \cup t_3} \bullet x_2^{t_2}. \end{aligned} \tag{18}$$

2) Cascade and Parallel Interconnections:

$$\begin{aligned} z &= x^{t_1}(j_1) \# x^{t_2}(j_2) + x^{t_3}(j_1) \# x^{t_4}(j_2) \\ &= x^{t_1}(j_2) + x^{t_3}(j_2) \quad \text{if} \quad j_1 \in t_2,\ j_1 \in t_4 \\ &= x^{t_1 \cup t_3}(j_2). \end{aligned} \tag{19}$$

Proof: A straightforward application of Theorems 2 and 3 will prove this Theorem.

The types of interconnections and their algebraic properties can now be used to construct any switch from a certain group of subswitches. To generalize this idea, the following Theorem states that any switch can be decomposed into a combination of subswitches that are interconnected together.

Theorem 6: If $F_i(\mathbf{X})$ is an ***MVL*** switch, where $X = (x_1, x_2, \cdots, x_n)$ is the input variable vector, then

$$F_i(\mathbf{X}) = \sum_{l_i=1}^{L_i} \prod_{r=1}^{n} x_r^{x'_{l_i, i, r}}(i) \tag{20}$$

where $\mathbf{T}_i = \{X i,, \mathbf{X}'_{i,2}, \ldots, \mathbf{X}'_{i,L_i}\}$.

Proof: The operation of the switch $F_i(\mathbf{X})$ can be decomposed into L_i **ORed** branches, where L_i is the number of elements of $\mathbf{T}_i$. Let us take one input vector $\mathbf{X}' \in \mathbf{T}_i$, according to (5), the realization of this branch through a combination of subswitches is as follows:

$$\begin{aligned} F_i(\mathbf{X}') &= \begin{cases} & \text{if } \mathbf{X}' \in \mathbf{T}_i \\ \text{OFF} & \text{otherwise} \end{cases} \\ &= x_1^{x'_1}(i) \bullet x_2^{x'_2}(i) \bullet \cdots \bullet x_n^{x'_n}(i) \\ &= \prod_{r=1}^{n} x_r^{x'_r}(i). \end{aligned} \tag{21}$$

Since the OR connection, according to Theorem 3, is a parallel connection denoted by the operator $\sum$, (20) follows.

The construction of any subswitch with multielement threshold set from the unary subswitches can be inferred from Theorem 6, the following Theorem reveals this result explicitly.

Theorem 7: The R^2 unary **subswitches** for the MVL **radix-*R*** consist of a complete set through which any subswitch can be composed.

Proof: Let $x^t(j)$ be any subswitch, and $t = (e_1, e_2, \ldots, e_m)$, where $e_l \in S$, then

$$\begin{aligned} x^t(j) &= x^{(e_1, e_2, \cdots, e_m)}(j) \\ &= x^{e_l}(j) + x^{e_2}(j) + \cdots + x^{e_m}(j) \\ &= \sum_{l=1}^{m} x^{e_l}(j) \end{aligned} \tag{22}$$

which means that **any** subswitch can bc constructed from only unary subswitches $[x^{e_l}(j)]$. Since Theorem 1 assures that the number of the unary subswitches is R^2, the Theorem 7 follows.

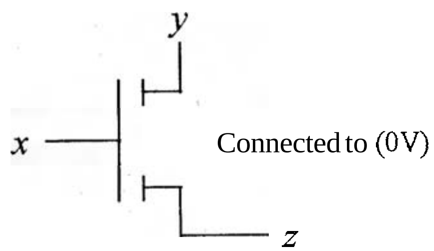


Fig. 10. n-EMOS transistor.

TABLE II
n-EMOS CHARACTERISTICS

x \ y	0	1	2
0	F	F	F
1	0	F	F
2	0	1	U

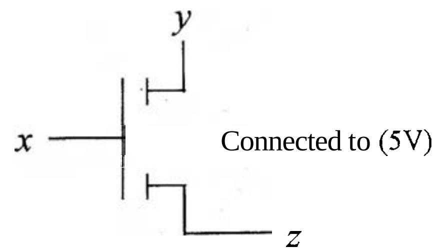


Fig. 11. p-EMOS transistor.

TABLE III
p-EMOS CHARACTERISTICS

x \ y	0	1	2
0	U	1	2
1	F	F	2
2	F	F	F

IV. TERNARY ALL CMOS REALIZATION

In this section, we will illustrate the application of the theory proposed in Sections II and III to ternary logic (TL). TL is better known as the competitor of the BL [3].

The CMOS transistors, both the n- and p-Enhancement (n-EMOS and p-EMOS) types, are employed here to realize the design of all the TL unary subswitches. At the end of this section, an example is presented to illustrates the procedure of designing the three switches from the unary subswitches.

A. CMOS Transistors

The TL values (levels) are used to introduce the properties of the CMOS transistors. Using 0–5 V power supply, the three levels of TL are defined as: 0 denotes (0 V); 1 denotes (2.5 V); and 2 denotes (5 V). The Connector-Switch-Attenuator(CSA) model [15] is adopted to characterize switching properties of the CMOS transistors as follows:

1) *n-EMOS Transistor:* It is assumed that the threshold voltage < logic 1 (2.5 V) and the substrate of the transistor is connected to the lowest voltage level 0 V. Fig. 10 shows the configuration of such a transistor. The Gate (x) and the Drain (y) are connected to independent inputs, whereas the Source (z) is the output. Here, z is assumed to be connected to an active load, such as a switch or a subswitch.

   In Table II, the properties of this configuration are revealed using a truth table where x and y are TL variables; i.e., x, y $\in \{0, 1, 2\}$, F designates a Float state (OFF, in [15] denoted by z), whereas U designates an Undefined state.

2) *p-EMOS Transistor:* The same argument mentioned in the n-EMOS case is applicable except for the threshold voltage >-2.5 V and the substrate has to be connected to the highest voltage level 5 V. Fig. 11 and Table III show the configuration and the properties of this transistor.

B. Subswitch and Switch Design

Using Tables II and 111, we can specify switching boundaries for both n- and p-EMOS transistors. Subswitches and combinations of them can be extracted from these tables using the theory presented in Sections II and III. The subswitches that can be identified from these tables are referred to as the primary subswitches and designated by p.

1) n-EMOS Transistor:

i) when $y = 0 \Rightarrow p_1$
$$= x^{1,2}(0) = \begin{cases} 0 & \text{if } x \in \{1, 2\} \\ F & \text{if } x = 0 \end{cases}$$

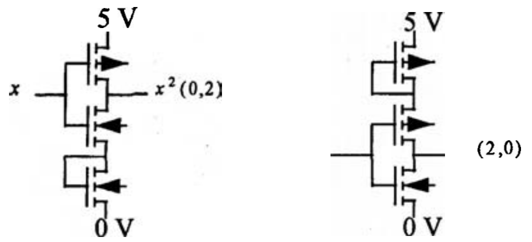
ii) when $y = 1 \Rightarrow p_2$
$$= x^{2}(1) = \begin{cases} 1 & \text{if } x = 2 \\ F & \text{if } x \in \{0, 1\} \end{cases}$$

iii) when $x = 1 \Rightarrow p_3$
$$= y^{0}(0) = \begin{cases} 0 & \text{if } y = 0 \\ F & \text{if } y \in \{1, 2). \end{cases}$$

2) p-EMOS Transistor:

i) when $y = 1 \Rightarrow p_4$
$$= x^{0}(1) = \begin{cases} 1 & \text{if } x = 0 \\ F & \text{if } x \in \{1, 2\} \end{cases}$$

ii) when $y = 2 \Rightarrow p_5$
$$= x^{0,1}(2) = \begin{cases} 2 & \text{if } x \in \{0, 1) \\ F & \text{if } x = 2 \end{cases}$$



(a) (b)

Fig. 12. The complementary CMOS subswitches.

$$\text{iii) when } x = 1 \Rightarrow p_6 = y^2(2) = \begin{cases} 2 & \text{if } y = 2 \\ F & \text{if } y \in \{0, 1\}. \end{cases}$$

The rest of the possible *primary* subswitches as well as $y^0(0)$ and $y^2(2)$ are less desirable for implementation. For $y^0(0)$ and $y^2(2)$, this is due to the loading effects; i.e., the control input is not applied at the gate and thus can cause signal degradation. In the case of the other unusable primary subswitches, this is due to the existence of the undefined state (U). This means only four feasible subswitches can be extracted from the two CMOS transistors. Are they enough for TL function realization?

Generally, CMOS transistors are nonideal switches [15], for instance when a gate of a transistor receives an F (OFF) state then that transistor state is U (Undefined). Therefore, we need to add constraints on the application of the algebra described in Section III. In particular, the input subswitch of a cascade interconnection, $x^{t_1}(j_1)$ in (14), should not include F state in its output. To circumvent this problem, complementary subswitches have to be used when a cascade interconnection is carried out. If a complementary subswitch $x^{t_1}(j_1, \hat{j}_1)$ is connected in cascade to $x^{t_2}(j_2)$, then (14) is rewritten as follows:

$$\begin{aligned} z &= [x^{t_1}(j_1, \hat{j}_1)]^{t_2}(j_2) \\ &= x^{t_1}(j_2) \quad \text{if} \quad j_1 \in t \text{ and } \hat{j}_1 \notin t_2. \end{aligned} \tag{23}$$

Symbolically the above equation can be written as

$$\begin{aligned} z &= x^{t_1}(j_1, \hat{j}_1) \# x^{t_2}(j_2) \\ &= x^{t_1}(j_2) \quad \text{if} \quad j_1 \in t_2 \quad \text{and } \hat{j}_1 \notin t_2. \end{aligned} \tag{24}$$

The construction of complementary switches for the primary subswitches is shown as follows:

1) $x^2(1,2) = x^{0,1}(2, 1) = x^2(1) + x^{0,1}(2) = p_2 + p_5$;
2) $x^0(1,0) = x^{1,2}(0, 1) = x^0(1) + x^{1,2}(0) = p_4 + p_1$;
3) $x^2(0, 2) = x^{0,1}(2, 0) = x^2(0) + x^{0,1}(2) = x^2(0) + p_5$;
4) $x^0(2,0) = x^{1,2}(0, 2) = x^0(2) + x^{1,2}(0) = x^0(2) + p_1$.

Both $x^2(0)$ and $x^0(2)$ are difficult to realize from the primary subswitches only. However, $x^2(0,2)$ and $x^0(2,0)$ do exist in the literature and they are denoted by ${}^2x^2$ and ${}^0x^0$, respectively, [9] as shown in Fig. 12(a) and (b). We implemented $x^2(0,2)$ and $x^0(2,0)$ using a 4007 CMOS invertor *IC* and verify their operation.

The following Theorem assures the existence of the complete set of subswitches required to realize any TL function.

Theorem 8: n-EMOS and p-EMOS transistors are enough to realize any TL function.

Proof: To prove this Theorem, we need to realize the $3^2 = 9$ subswitches (complete set), according to Theorem 6 and 7. The design of the complete set of Ternary subswitches is as shown below

1. $x^0(0) = x^0(1, 0) \# x^{1,2}(0) = (p_4 + p_1) \# p_1$ [see Fig. 13(a)]
2. $x^1(0) = x^{0,1}(0) \bullet x^{1,2}(0) = [x^{0,1}(2, 0) \# p_1] \bullet p_1$ [see Fig. 13(b)]
3. $x^2(0) = [x^{0,1}(0) + x^2(1)] \# x^{1,2}(0) = \{[x^{0,1}(2, 0) \# p_1] + p_2\} \# p_1$ [see Fig. 13(c)]
4. $x^0(1) = p_4$ (single p-MOS transistor)
5. $x^1(1) = \{[x^{1,2}(0) + x^0(1)] \# x^0(1)\} \bullet \{[x^2(1) + x^{0,1}(2)] \# x^2(1)\} = [(p_1 + p_4) \# p_4] \bullet [(p_2 + p_5) \# p_2]$ [see Fig. 13(d)]
6. $x^2(1) = p_2$ (single n-MOS transistor)
7. $x^0(2) = [x^0(1) + x^{1,2}(2)] \# x^{0,1}(2) = \{p_4 + [x^{1,2}(0, 2) \# p_5]\} \# p_5$ [see Fig. 13(e)]
8. $x^1(2) = x^{1,2}(2) \bullet x^{0,1}(2) = \{[x^{1,2}(0, 2) \# p_5] \bullet p_5$ [see Fig. 13(f)]
9. $x^2(2) = x^2(1, 2) \# x^{0,1}(2) = (p_2 + p_5) \# p_5$ [see Fig. 13(g)]

The set above is not a unique design. Thus, it is possible to obtain different designs using algebraic manipulations. However, the set described above attempts to use a minimum number of transistors.

C. *Ternary Function Realization*

After a complete set of subswitches is specified, the TL Function implementation will be shown below. The implementation depends on the block diagram in Fig. 14 which depicts a general solution for TL function implementation. F_0, F_1, and F_2 represent the subswitches for 0, 1, and 2 of the TL levels, respectively.

The procedure for implementation is illustrated with an example. Table I shows a TL function that is implemented. The Procedure is explained through the following steps.

1) Partition the truth-table into three separate columns, each corresponding to a switch function (F_0, F_1, or F_2). Table IV shows such a partition.
2) Apply Theorem 6 together with Theorems 2–5 and 7 to synthesize the basic switch design for the function.

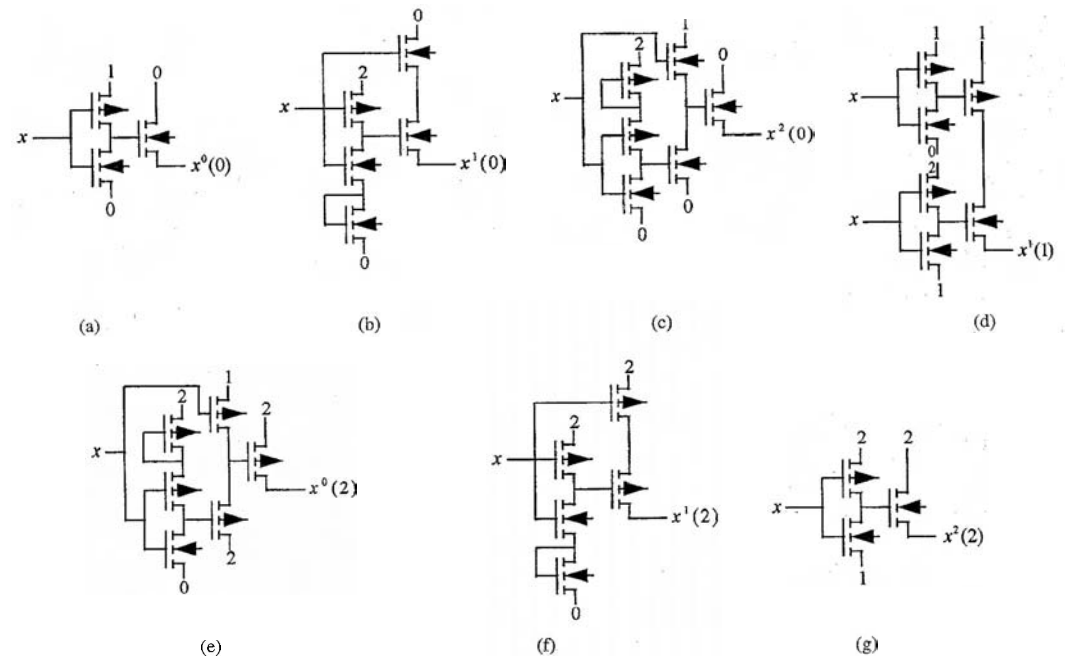


Fig. 13. Multitransistor unary subswitches

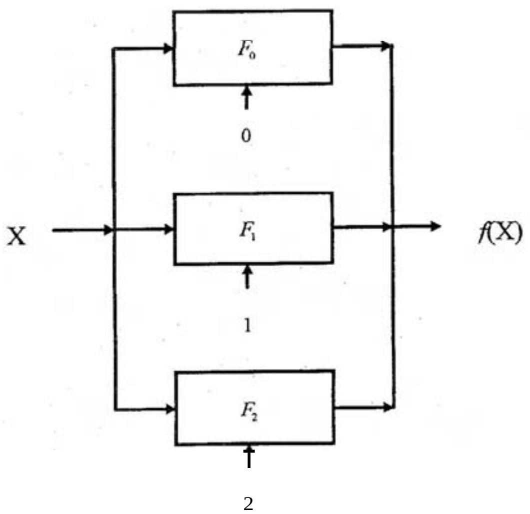


Fig. **14.** Ternary circuit by switch realization.

a)

$$F_0(x_1, x_2) = \sum_{(x'_1, x'_2) \in T_0} \prod_{r=1}^{2} x_r^{x'_r}(0)$$

where $\mathbf{T}_0 = \{(0, 1), (1, 1), (2, 2)\}$. (25)

This switch will be ON when the input vector is an element of $\mathbf{T}_0$, and thus, the output is 0. Otherwise it holds a Float state **F**. It can be generated through the following combination of subswitches:

$$F_0(x_1, x_2) = x_1^0(0) \bullet x_2^1(0) + x_1^1(0) \bullet x_2^1(0) + x_1^2(0) \bullet x_2^2(0). \quad (26)$$

TABLE IV
SWITCH DECOMPOSITION OF A TERNARY FUNCTION

x_1	x_2	$f(x_1,x_2)$	F_0	F_1	F_2
0	0	2	F	F	2
1	0	2	F	F	2
2	0	1	F	1	F
0	1	0	0	F	F
1	1	0	0	F	F
2	1	2	F	F	2
0	2	1	F	1	F
1	2	2	F	F	2
2	2	0	0	F	F

b) By the same argument in -i-, we have

$$F_1(x_1, x_2) = x_1^2(1) \bullet x_2^0(1) + x_1^0(1) \bullet x_2^2(1)$$

where $\mathbf{T}_1 = \{(2, 0), (0, 2)\}$. (27)

c) Similarly,

$$F_2(x_1, x_2) = x_1^0(2) \bullet x_2^0(2) + x_1^1(2) \bullet x_2^0(2) + x_1^2(2) \bullet x_2^1(2) + x_1^1(2) \bullet x_2^2(2)$$

where $\mathbf{T}_2 = \{(0, 0), (1, 0), (2, 1), (1, 2)\}$. (28)

D. Minimization of the Switch Design

It is evident that the above expressions for the three switches are in terms of the unary subswitches, and can be further minimized. The following definition can be used in minimizing the design of each switch.

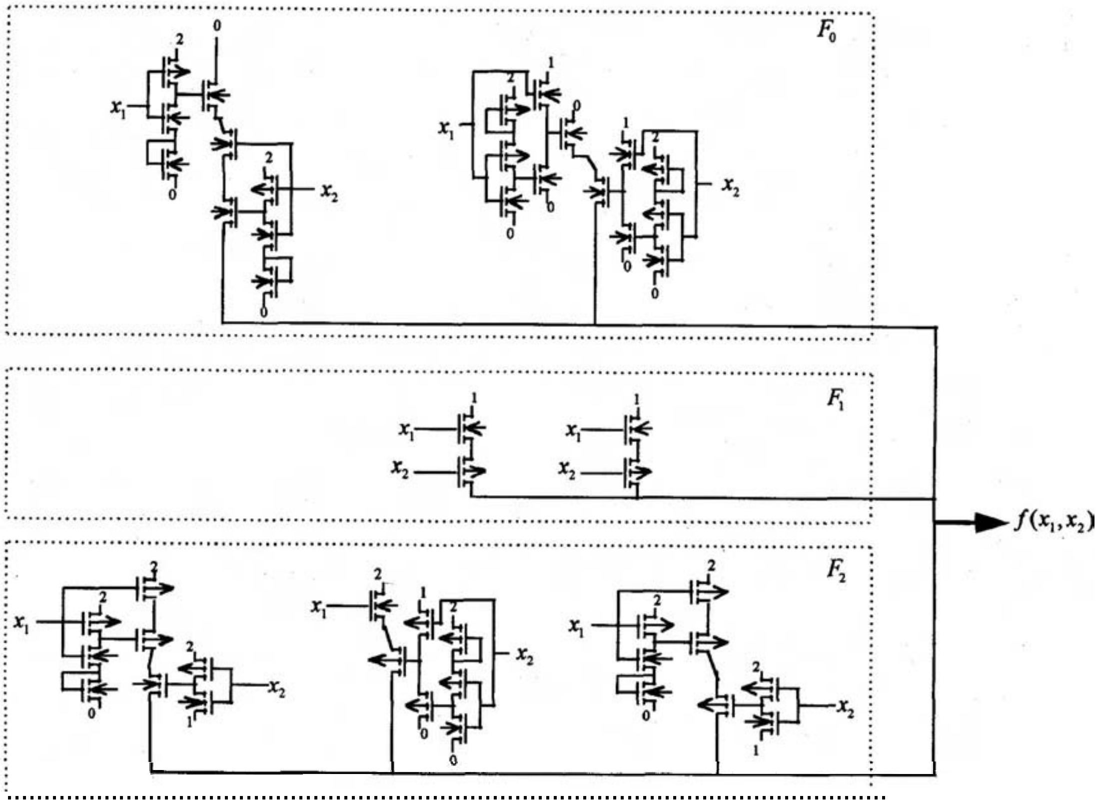


Fig. 15. CMOS ternary logic circuit for the function in Table III.

Definition 3: According to Section IV-B and Theorem 8, the subswitches in both the complete set and the primary set are divided into the following categories according to the number of transistors used in their structure:

Category 1. (One Transistor):

$$p_4 = x^0(1)$$
$$p_2 = x^2(1)$$
$$p_1 = x^{1,2}(0)$$

and

$$p_5 = x^{0,1}(2).$$

Category 2. (Three Transistors):

$$x^0(0)$$

and

$$x^2(2).$$

Category 3. (Five Transistors):

$$x^1(0)$$

and

$$x^2(1).$$

Category 4. (Six Transistors):

$$x^2(0)$$
$$x^1(1)$$
$$x^0(2).$$

This categorization can be similarly performed for a higher radix after describing devices that are used in subswitch design.

To obtain a minimized design (lowest number of transistors) in realizing a TL function, one can utilize the algebra in Sections **II** and **III** along with Definition **3.** One of the most useful properties that can be utilized in the minimization is the distributive law. For example, the minimization procedure will be applied on the three switches designed in the previous section.

1)

$$\begin{aligned} &F_0(x_1, x_2) \\ &= x_1^0(0) \bullet x_2^1(0) + x_1^1(0) \bullet x_2^1(0) + x_1^2(0) \bullet x_2^2(0) \\ &= [x_1^0(0) + x_1^1(0)] \bullet x_2^1(0) + x_1^2(0) \bullet x_2^2(0) \\ &= x_1^{0,1}(0) \bullet x_2^1(0) + x_1^2(0) \bullet x_2^2(0) \\ &= [x_1^{0,1}(2, 0) \# x_1^{1,2}(0)] \bullet x_2^1(0) + x_1^2(0) \bullet x_2^2(0). \end{aligned}$$

The last design has fewer transistors than the basic design. This can be verified by counting the transistors using Definition **3,** in the basic design expressed in (26) there are 30, whereas the final design there are 21.

2)

$$F_1(x_1, x_2) = x_1^1(1) \bullet x_2^0(1) + x_1^0(1) \bullet x_2^2(1).$$

According to Definition **3,** this design has the lowest number of transistors; **i.e.,** 4. We can conclude that each subswitch implemented by one transistor *(primary)* is minimal.

3)

$$
\begin{aligned}
F_2(x_1, x_2) &= x_1^0(2) \bullet x_2^0(2) + x_1^1(2) \bullet x_2^0(2) \\
&\quad + x_1^2(2) \bullet x_2^1(2) + x_1^1(2) \bullet x_2^2(2) \\
&= [x_1^0(2) + x_1^1(2)] \bullet x_2^0(2) \\
&\quad + x_1^2(2) \bullet x_2^1(2) + x_1^1(2) \bullet x_2^2(2) \\
&= x_1^{0,1}(2) \bullet x_2^0(2) + x_1^2(2) \\
&\quad \bullet x_2^1(2) + x_1^1(2) \bullet x_2^2(2).
\end{aligned}
$$

In this design we need 23 transistors, whereas in the basic design expressed in (28) we need 39 transistors.

The complete realization is shown in Fig. 15.

V. Conclusion

A new theory concerning the implementation of Radix R MVE functions is proposed in this paper. In this theory, the standard gates *{min, max, cyclic operator,* and *window literal)* used in MVL design are replaced by switches. Each switch implements one of the R mutually exclusive subfunctions into which a function is decomposed. Thus the implementation is divided into smaller modules that are easily synthesized. Each switch consists of a combination of more elementary switches called subswitches. Three types of interconnections of subswitches were proposed with an operational algebra. The proposed theory is applied to TL as an example. Ordinary CMOS transistors were used to realize an all CMOS TL function. Using CMOS technology in conjunction with the proposed logic minimization technique in implementing an MVL function is expected to result in a smaller chip area, low power consumption, and simple fabrication. The procedure of the proposed TL design was described in three systematic steps. First, a complete set of subswitches were defined from the properties of the CMOS transistors (n- and p-EMOS transistors). Second, the complete set of unary subswitches was utilized to design the three switches necessary to realize a TL function. Finally, a minimization method was described and used to reduce the number of the transistors required for implementation. Some of the points that need further investigation are: 1) A systematic algorithm to simplify the minimization techniques; 2) If OFF state is considered as one of the values for the MVL, then, the implementation can be achieved using $R-1$ switches instead of R; and 3) Using other types of MOS transistors in the design of primary subswitches, such as Depletion MOS type [16], and multiple threshold voltage CMOS type [9].

Acknowledgment

The authors wish to thank the reviewers for their valuable and constructive comments. We also wish to thank the Electrical and Computer Engineering Department of the Worcester Polytechnic Institute for their continuous support.

References

[1] D. C. Rine, Ed., *Computer Science and Multiple-Valued Logic Theory and Application.* New York: North-Holland, 1979 and 1984.
[2] S. L. Hurst, "Multiple-valued logic—Its status and its future," *IEEE Trans. Cornput.,* vol. C-33, pp. 1160–1179, Dec. 1984.
[3] K. Y. Fang and A. S. Wojcik, "Modular decomposition of combination multiple-valued circuits," *IEEE Trans. Comput.,* vol. 37, pp. 1293–1301, Oct. 1988.
[4] A. Jain, R. Bolton, and M. Abd-El-Barr, "CMOS multiple-valued logic design Part I: Circuit implementation," *IEEE Trans. Circuits Syst.,* vol. 40, pp. 503–514, Aug. 1993.
[5] , "CMOS multiple-valued logic design, Part II: Function realization," *IEEE Trans. Circuits Syst.,* vol. 40, pp. 515–522, Aug. 1993.
[6] Y.-J. Chang and C. L. Lee, "Synthesis of multi-variable MVL functions using hybrid mode CMOS logic," in *Proc. 24th ISMVL,* May 1994, pp. 35–41.
[7] K. W. Current, "Multiple valued logic: Current-mode CMOS circuits," in *Proc. 23th ISMVL,* May 1993, pp. 176–181.
[8] C. Y. Wu and H. Y. Huang, "Design and application of pipelined dynamic CMOS ternary logic and simple ternary differential logic," *IEEE J. Solid-State Circuits,* vol. 28, pp. 895–906, Aug. 1993.
[9] X. W. Wu and F. Prosser, "CMOS ternary logic circuits," *IEE Proc.,* vol. *137,* pt. G, pp. 21–27, Feb. 1990.
[10] F. Prosser, X. Wu, and X. Chen, "CMOS temary flip flops and their applications," *IEE Proc.,* vol. 135, pt. E, pp. 266–272, Sept. 1988.
[11] S. A. H. Shalash, M. H. Ali, J. Abdul-Rahim, M. A. Abdul-Karim, and S. Adham, "CMOS J-K triflop," *Int. J. Electron.,* vol. 58.2, pp. 355–363, 1985.
[12] W. S. Chu and W. Current, "Quaternary multiplier circuit," in *Proc. 24th ISMVL,* May 1994, pp. 15–18.
[13] A. Kazeminejad, K. Navi, and D. Etiemble, "CML current mode full adders for 2.5-V power supply," in *Proc. 24th ISMVL,* May 1994, pp. 10–14.
[14] C. M. Allen and D. D. Givone, "A minimization technique for multiple-valued logic systems," *IEEE Trans. Comput.,* vol. C-17, pp. 182–184, Feb. 1968.
[15] J. P. Hayes, "Fault modeling for digital MOS integrated circuits," *IEEE Trans. Computer-Aided Design,* vol. CAD-3, pp. 200–207, July 1984.
[16] A. Heung and H. T. Mouftah, "Depletion/enhancement CMOS for a low power family of three-valued logic circuits," *IEEE J. Solid-State Circuits,* vol. SC-20, pp. 609–616, Apr. 1985.

Mudhafar A. Hassan Ali (S'94) was born in Baquba, Iraq on April 13, 1962 He received the B Sc and M Sc degrees in electrical engineering from the University of Baghdad (with highest honors), Baghdad, Iraq, in 1984 and 1989, respectively He is pursuing the Ph.D degree from Worcester Polytechnic Institute, Worcester, MA

From 1984 to 1987, he was a teaching assistant with the University of Baghdad From 1989 to 1991, he was an instructor with the University of Baghdad From 1992 to 1993. he was the Head of the Electronics Engineering Department, College of Engineering, University of Seventh of April, Sabrata, Libya. His research interests include multivalued logic, communication theory, signal processing, VLSI architecture, neural networks, and coding and cryptography.

Mr. Ali is a member of Eta Kappa Nu and Sigma Xi.

Falah J. Hassan-Alshiroofi received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Baghdad, Baghdad, Iraq, in 1978 and 1983, respectively.

From 1983 to 1991 he was an assistant inshuctor at The Technical Institute of Najaf, Najaf, Iraq. From 1991 to 1994 he was an instructor at the College of Engineering, University of Seventh of April, Sabrata, Libya, Sabarata, Libya.

Hemant G. Rotithor (M'88–SM'93) received the B.S., M.S., and Ph.D. degrees in electrical engineering in 1979, 1981, and 1989, respectively.

From 1982 to 1985, he worked as a development engineer in the research and development division of Philips India. From 1990 to 1995, he was with the Electrical and Computer Engineering Department, Worcester Polytechnic Institute, MA, working as an assistant professor. His research interests include logic design, distributed computing, parallel algorithms and architectures. microprocessor based instrumentation and process control, decision making under uncertainty, and hardware and software support for performance evaluation. He has published approximately 25 papers in refereed journals and conferences related to his research.

Dr. Rotithor was a member of the steering committee for the Third IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC'92) held in Boston, MA and his biographical profile has been listed in Marquis ***Who's Who in Science and Engineering*** and ***Who's Who in the World.*** He is a member of Eta Kappa Nu, Tau Beta Pi, and Sigma Xi.