Three-Dimensional Lattice Logic Circuits, Part I: Fundamentals

Anas N. Al-Rabadi

Abstract: Fundamentals of regular three-dimensional (3D) lattice circuits are introduced. Lattice circuits represent an important class of regular circuits that allow for local interconnections, predictable timing, fault localization, and self-repair. In addition, three-dimensional lattice circuits can be potentially well suited for future 3D technologies, such as nanotechnologies, where the intrinsic physical delay of the irregular and lengthy interconnections limits the device performance. Although the current technology does not offer a menu for the immediate physical implementation of the proposed three-dimensional circuits, this paper deals with three-dimensional logic circuit design from a fundamental and foundational level for a rather new possible future directions in designing digital logic circuits.

Keywords: Electronic circuits, Galois logic, lattice circuits, regularity, symmetry by variable repetition, three-dimensional logic circuits.

1 Introduction

With future logic realization in technologies that are scaled down rapidly in size, the emphasis will be increasingly on the mutually linked issues of regularity, predictable timing, high testability, fast fault localization, and self-repair [1, 2, 3]. For the current leading technologies with the active-device count reaching the hundreds of millions, and more than 80% of circuit areas are occupied by local and global interconnects, the delay of interconnects is responsible for about 40-50% or more of the total delay associated with a circuit [1, 4].

In future technologies, interconnects will take even higher percent of area and delay which creates interest in regular cellular circuits, especially for deep sub-micron technologies. For example, Figure 1 illustrates trends for electrical signal

Manuscript received October 1, 2004.

The author is with Portland State University, Post Office Box 85, Portland, Oregon 97207-0085 (e-mail: alrabadi@yahoo.com). The author is also associated with the Computer Engineering Department at the University of Jordan
delays for global interconnects with repeaters and without repeaters versus the local interconnects [4].

Lattice circuits [1] generalize the ideas from the well-known regular circuits: Fat trees, Generalized PLAs, Maitra cascades, and Akers arrays [5], into a more systematic framework which is closely related to the symmetry of functions [1, 6], and symmetric networks [6].

Realization of logic circuits in three-dimensional space can be very important for future technologies, as it shows that the best way is to place combinational logic functions in a three-dimensional space where all local interconnections are of the same length and global interconnections are only inputs on parallel oblique planes (cf. Section 4). Moreover, three-dimensional cubical lattice circuits have special importance since three-dimensional Crystal Lattices exist where inter-related atoms, that lay in a potential field, are spaced on the corners of three-dimensional cubes [7, 8, 9], and thus the potential of the physical implementation of 3D lattices using 3D Crystal Lattices.

The remainder of this paper is organized as follows: Fundamentals are presented in Section 2. Two-dimensional lattice circuits are presented in Section 3. Three-dimensional lattice circuits are introduced in Section 4. Conclusions are presented in Section 5.

2 Fundamentals

The binary hierarchy of families of canonical forms [1, 10, 11, 12] and the corresponding decision diagrams [1, 13, 12, 14] are based on three basic functional
expansions: Shannon, positive Davio, and negative Davio expansions, which are
given below respectively [12]:

\[
\begin{align*}
 f(x_1, x_2, \ldots, x_n) &= x_1^1 \cdot f_0(x_1, x_2, \ldots, x_n) \oplus x_1 \cdot f_1(x_1, x_2, \ldots, x_n) \quad (1) \\
 f(x_1, x_2, \ldots, x_n) &= 1 \cdot f_0(x_1, x_2, \ldots, x_n) \oplus x_1 \cdot f_2(x_1, x_2, \ldots, x_n) \quad (2) \\
 f(x_1, x_2, \ldots, x_n) &= 1 \cdot f_1(x_1, x_2, \ldots, x_n) \oplus x_1^0 \cdot f_2(x_1, x_2, \ldots, x_n) \quad (3)
\end{align*}
\]

where \( f_0(x_1, x_2, \ldots, x_n) = f(0, x_2, \ldots, x_n) = f_0 \) is the negative cofactor of variable \( x_1 \),
\( f_1(x_1, x_2, \ldots, x_n) = f(1, x_2, \ldots, x_n) = f_1 \) is the positive cofactor of variable \( x_1 \),
and \( f_2(x_1, x_2, \ldots, x_n) = f(0, x_2, \ldots, x_n) \oplus f(1, x_2, \ldots, x_n) = f_0 \oplus f_1 \). All operations
in Equations (1) - (3) are performed using Boolean algebra, i.e., \( \oplus \) is Boolean XOR,
and \( \cdot \) is Boolean multiplication.

Multiple-valued spectral methods are used in many applications in synthesis,
analysis, testing, classification, and verification of logic circuits and systems
[1, 2, 6, 12]. Because Galois field proved to possess desired properties in many ap-
plications such as test, communications, and signal processing, the developments
of three-dimensional logic circuits, in this work, will be conducted on the corre-
sponding Galois field (GF) algebraic structures. Radix three Galois field (GF(3))
addition and multiplication are defined in Tables 1(a) and 1(b), respectively.

### Table 1. (a) GF(3) addition, and (b) GF(3) multiplication.

<table>
<thead>
<tr>
<th>+</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

(A)

<table>
<thead>
<tr>
<th>*</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

(B)

A literal is a function of a single variable. 1-Reduced Post Literal (1-RPL) [1]
is defined as:

\[ i \cdot x = 1 \iff x = i \quad \text{else} \quad i \cdot x = 0 \]  \( (4) \)

For example \( 0 \cdot x, 1 \cdot x, 2 \cdot x \) are the zero, first, and second polarities of the 1-Reduced
Post literal, respectively. Also, the ternary shifts of variable \( x \) are defined as: \( x \) with
no shift, \( x' \) with one shift, and \( x'' \) with two shifts (i.e., \( x = x + 0, \ x' = x + 1, \)
and \( x'' = x + 2 \), respectively), and \( x \) can take any value in the set \( \{0, 1, 2\} \). Ternary
1-Reduced Post Literals will be used in Section 4 to construct 3D lattice circuits by
controlling the propagation of signals (sub-functions) in three-dimensions.

### 3 Two-Dimensional Lattice Circuits

The concept of lattice circuits for switching functions [1] involves three compo-
nents: (1) expansion of a function, that corresponds to the initial node (root) in
the lattice, which creates several successor nodes of the expanded node, (2) joining (collapsing) of several nodes of a decision tree level to a single node, which is the reverse operation of the expansion process, and (3) regular geometry to which the nodes are mapped that guides which nodes of the level are to be joined.

While the realization of Boolean non-symmetric functions in Akers arrays requires an exponential growth of repetition of variables in the worst case [5], the realization of Boolean non-symmetric functions in lattice circuits requires a linear growth of repetition of variables, and consequently one need not to repeat the variables of non-symmetric functions too many times to realize such functions in lattice circuits for most practical benchmarks [1].

Figure 2 illustrates, as an example, the geometry of a 4-neighbor lattice circuit and joining operations on the nodes where each cell has two inputs and two outputs (i.e., four neighbors). The construction of the lattice circuit in Figure 2 implements the following one possible convention: (1) top-to-bottom expansion, and (2) left-to-right joining (i.e., left-to-right propagation of the corresponding correction functions in Figures 2(c) and 2(d), respectively).

![Diagram](image-url)

Fig. 2. (a) A two-dimensional 4-neighbor lattice circuit, (b) joining rules for: binary Shannon lattice circuit, (c) binary positive Davio lattice circuit, and (d) binary negative Davio lattice circuit, where + is Boolean addition, ∗ is Boolean multiplication, and ⊕ is Boolean XOR.
**Definition 1.** The function that is generated by joining two nodes (subfunctions) in a lattice circuit is called the *joined function*. The function that is generated in nodes other than the joining nodes, to preserve the functionality in the lattice circuit, is called the correction function.

Note that the lattices shown in Figure 2 preserve the functionality of the corresponding sub-functions $f$ and $g$. This can be observed, for instance, in Figure 2(b) as the negated variable $\{d\}$ will cancel the un-complemented variable $\{a\}$, when propagating the cofactors from the lower levels to the upper levels or vice versa, without the need for any correction functions to preserve the output functionality of the corresponding lattice circuit. This simple observation cannot be seen directly in Figures 2(c) and 2(d), as the correction functions are involved to cancel the effect of the new joining nodes for the preservation of the functionality of the new lattice circuits (these correction functions are shown in the extreme right leaves of the second level in Figures 2(c) and 2(d), respectively).

It is shown in [5] that every function that is not symmetric can be symmetrized by repeating variables, and that a totally symmetric function can be obtained from an arbitrary non-symmetric function by the repetition of variables. Consequently, lattice circuits and the symmetry of functions are very much related to each other. Example 1 will illustrate such close relationship.

**Example 1.** For the following non-symmetric three-variable Boolean function: $F = a \cdot b + d \cdot c$, by utilizing the joining rule that was presented in Figure 2(b) for two-dimensional lattice circuit with binary Shannon nodes, one obtains the following lattice circuit.

![Shannon lattice circuit](image)

Fig. 3. Shannon lattice circuit for the non-symmetric function $F = a \cdot b + d \cdot c$. 
One can note that without the repetition of variable(s) (e.g., variable $b$ in Figure 3) $F$ cannot be produced by any lattice circuit. All internal nodes in Figure 3 are 2-to-1 multiplexers (i.e., selectors). In Figure 3, if one multiplies each leaf value, from left to right, with all possible bottom-up paths (from the leaves to the root $F$) and add them over Boolean algebra then one obtains the function $F$ (i.e., the root) as follows:

$$F = (0 \cdot c' \cdot b' \cdot d') + (1 \cdot c \cdot b' \cdot d') + (0 \cdot c \cdot b \cdot d') + (0 \cdot b' \cdot c \cdot b' \cdot a) + (1 \cdot b \cdot c \cdot b \cdot a') + (1 \cdot b \cdot c \cdot b' \cdot a) + (1 \cdot c \cdot b \cdot a) = (1 \cdot c \cdot b' \cdot d') + (1 \cdot b \cdot c \cdot b' \cdot a') + (1 \cdot c \cdot b \cdot a) + (1 \cdot c \cdot b \cdot a) = d' \cdot c + a \cdot b.$$

One can observe that in order to represent the non-symmetric function in Example 1 in the two-dimensional lattice circuit, variable $b$ is repeated, where the nodes in Figure 3 are Shannon nodes, which are merely two-input one-output multiplexers, whose output goes in two directions, with the set of variables $\{a,b,c\}$ operate as control signals [1].

The results from this Section will be generalized to ternary logic in Section 4, and thus from two-dimensional space to three-dimensional space. It is important to prove that the repetition of variables will have an end in the process of the symmetrization of the non-symmetric functions. An intuitive proof is as follows: For totally symmetric functions the number of variables are equal to the number of levels of the lattice circuit, as there is no need to repeat variables, and as it is known that by the repetition of variables every non-symmetric function is symmetrized [5], then this must result in definite number of levels in the corresponding lattice circuit and as a consequence in certain number of total variables, repeated and non-repeated, that will result in the termination of the process of symmetrization.

One method to define the symmetry of a Boolean function is by using symmetry indices $S^i$ [6]. A symmetry index ($S^i$) has superscript $i$ equals to the count of the number of “1” values in the states of variables in the corresponding cell in a Karnaugh map [6]. Example 2 illustrates the concept of symmetry indices for a two-variable Boolean function, and shows the close relationship between two-dimensional lattice circuits and symmetry indices.

**Example 2.** For the binary non-symmetric implication function: $F = d + b$, Figure 4 illustrates the relationship between (1) the $k$-map with non-conflicting symmetry indices $S^i$ and (2) the two-dimensional lattice circuit with non-conflicting leaves, which is achieved by repeating variable $\{a\}$ twice in the two-dimensional lattice circuit.

All internal nodes in Figure 4 are 2-to-1 multiplexers (i.e., selectors); if one multiplies each leaf value, from left to right, with all possible bottom-up paths (from the leaves to the root $F$) and add them over Boolean algebra then one obtains the function $F$ in the root.
Three-Dimensional Lattice Logic Circuits, Part I: Fundamentals

4 Three-Dimensional Lattice Circuits

The concept of two-dimensional lattice circuits that was presented in Section 3 can be generalized to include the case of three-dimensional lattice circuits. Since the most natural way to think about a binary lattice circuit is the two-dimensional 4-neighbor lattice circuit that was shown in Figure 2(a), one can extend the same idea to utilize the full three-dimensional space in the case of ternary lattices. Such lattices represent three-dimensional 6-neighbor lattice circuits (See Figures 6 and 7, for instance).

Although regular lattices can be realized in the three-dimensional space for the third Galois radix while maintaining their full regularity, they are unrealizable for radices higher than three (i.e., 4, 5, etc). Higher dimensionality lattices can be implemented in 3D space but on the expense of loosing the full regularity. This is because the circuit realization for the ternary case produces a regular circuit in three dimensions that is fully regular in terms of interconnections; all interconnec-
tions are of the same length. Realizing the higher dimensionality lattices in lower
dimensionality space is possible but with the expense of regularity; the lattices will
not be fully regular due to the uneven length of the interconnections between nodes.

As a topological concept, and as stated previously, lattice circuits can be created
for two, three, four, and any higher Galois radix. However, because our physical
space is three-dimensional, lattice circuits, as a geometrical concept, can be realized
in solid material, with all the interconnections between the cells are of the same
length, only for Galois radix two (2D space) or Galois radix three (3D space). It is
thus interesting to observe that the characteristic geometric regularity of the lattice
circuits realization which is observed for binary and ternary functions will be no
longer observable for quaternary and higher radix functions. Thus, the ternary
lattice circuits have a unique position as structures that can make the best use of
three-dimensional space.

Because three-dimensional lattice circuits exist in a three-dimensional space, a
geometrical reference of coordinate systems is needed in order to be systematic in
the realizations of the corresponding logic circuits. Consequently, the right-hand
rule of the Cartesian coordinate system is adopted (cf. Example 3).

As will be noted in Example 3, each dimension in a three-dimensional lattice
circuit corresponds to a value of the corresponding control variable: value zero of
the control variable propagates along the x-axis, value one of the control variable
propagates along the y-axis, and value two of the control variable propagates along
the z-axis.

**Example 3.** This example shows the realization of ternary 3-digit full adder
using three-dimensional lattice circuits. Multi-valued addition is performed utilizing
the modsum operator. The following table is the modsum addition for ternary
inputs.

<table>
<thead>
<tr>
<th>+</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Utilizing Table 2, 3-digit two ternary inputs are added as follows:

<table>
<thead>
<tr>
<th>A₂</th>
<th>A₁</th>
<th>A₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>B₂</td>
<td>B₁</td>
<td>B₀</td>
</tr>
<tr>
<td>C_out</td>
<td>S₂</td>
<td>S₁</td>
</tr>
</tbody>
</table>

The following is the logic circuit of the ternary 3-digit full adder:
The following maps are the Sum ($S$) and the output Carry ($C_{out}$) functions that appear in the logic circuit in Figure 5 using 1-RPLs for variables $a$, $b$, and $c$, and using Galois field addition and multiplication.

Table 3. Ternary maps for (a) ternary Sum: $F_1 = a^2b^1c + a^1b^0c + a^1b^1c + a^2b^0c + a^0b^1c + a^0b^0c + a^1b^1c + a^2b^0c + 2a^1b^1c + 2a^2b^1c + 2a^2b^0c + 2a^1b^0c + 2a^0b^0c + 2a^0b^1c + 2a^1b^0c + 2a^1b^1c + 2a^2b^0c + 2a^2b^1c$, and (b) ternary Carry: $F_2 = a^2b^1c + a^1b^1c + a^0b^1c + a^1b^1c + a^1b^0c + 2a^1b^1c + a^1b^0c + 2a^1b^1c + a^2b^0c + a^2b^1c + a^2b^0c + a^2b^1c + a^2b^0c + a^2b^1c$. 

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>(a)</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>(b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figures 6 and 7 are the 3D lattice realizations of the functions in Table 3.

As observed in Example 3, Figures 6 and 7 represent a fully regular lattice circuit in three-dimensions. Each dimension corresponds to a value of the corresponding control variable: value zero of the control variable propagates along the $x$-axis, value one of the control variable propagates along the $y$-axis, and value two of the control variable propagates along the $z$-axis. Since the ternary function in
Example 3 is fully symmetric, no variables are needed to be repeated in the corresponding lattice circuit. In three-dimensional space, each control variable spreads in a plane to control the corresponding nodes (these parallel planes are represented using the dotted triangles in Figures 6 and 7), in contrast to the binary case where each control variable spreads in a line to control the corresponding nodes (these control signals are represented in the solid horizontal lines in Figure 2(a)). Each node in Figures 6 and 7 represents a three-input one-output multiplexer, whose output goes in three directions.

All internal nodes in Figures 6 and 7 are 3-to-1 multiplexers (i.e., selectors), where if one multiplies each leaf value, going counter clockwise (CCW), with all possible out-to-in paths (i.e., from the leaves to the root) and add them over Galois field (Table 1) then one obtains the maps in Table 3(a) ($F_1$) and Table 3(b) ($F_2$), respectively. Also, one notes that internal nodes in Figures 6 and 7 lay on the corners of three-dimensional cubes, in contrast to the binary case (e.g., Figure 3) where nodes lay on the corners of 2D squares.

From Figures 6 and 7, one observes that the Sum and Carry functions are both symmetric, since there is no conflict in leaf values, and consequently there is no need to repeat variables to make the ternary functions realizable in the 3D lattice circuits. In ternary non-symmetric functions, at least one leaf has conflict values,

![Fig. 6. The Sum function $F_1$ of the ternary 3-digit full adder, where $-$ is ternary don’t care (i.e., 0,1 or 2).](image-url)
Three-Dimensional Lattice Logic Circuits, Part I: Fundamentals

Fig. 7. The Carry function $F_2$ of the ternary 3-digit full adder, where -- is ternary don’t care (i.e., 0,1 or 2).

and one needs to repeat variables to symmetrize the corresponding non-symmetric functions, in order to realize such functions in the corresponding 3D lattice circuits. Example 4 shows the realization of a ternary non-symmetric function in a 3D lattice through the repetition of variables.

**Example 4.** For the non-symmetric two-variable ternary-input ternary-output function: $F = ab + d' b'$, and by adopting the right-hand rule of the Cartesian coordinate system, Figure 8 illustrates the three-dimensional logic circuit to implement such non-symmetric function. In Figures 8(c) and 8(d), if one multiplies each leaf value, going counter clock wise (CCW), with all possible out-to-in paths (i.e., from the leaves to the root) and add them over Galois field (Table 1) then one obtains the maps in Figures 8(a) and 8(b), respectively, where $\{0, 1, 2\}$ are the zero, first, and second polarities of the 1-Reduced Post literal of variable $a$, $\{0, 1, 2\}$ are the zero, first, and second polarities of the 1-Reduced Post literal of variable $b$, and variables $a$ and $b$ can take any value in the set $\{0, 1, 2\}$.

5 Conclusions

The concept of three-dimensional lattice circuits is introduced. Three-dimensional lattice circuits generalize the concept of two-dimensional 4-neighbor lattice circuits...
into three-dimensional 6-neighbor lattice circuits.

Lattice circuits possess a very important property of high regularity, which is useful in many applications including fault-related issues: (1) fault diagnosis (testing), (2) fault localization, and (3) fault self-repair. Other advantages of the new three-dimensional logic circuits include: (a) no need for 3D layout routing and placement, analogously to the two-dimensional case, and (b) regularity lead to comparable ease of manufacturability. The 3D lattice circuits can be especially well suited for future 3D based technologies, where the intrinsic physical delay of the irregular and lengthy interconnections limits the device performance in the form of high power consumption and high delay in the interconnects especially at high frequencies.

Although the current technological advances do not yet offer practical solutions for the direct physical realization of the proposed three-dimensional lattice circuits,
this paper deals with three-dimensional logic circuit synthesis from a fundamental level for possible new future directions in designing circuits in three-dimensions.

References


