The CSI Journal on Computer Science & Engineering

The CSI Journal on
Computer Science and Engineering
Vol. 6, No. 2 & 4 (b), 2008
Pages 46-57
**Regular Paper**

# Using Error Compensation for Defect Tolerance of Nano-Systems*

Masoud Hashempour Zahra Mashreghian Arani Fabrizio Lombardi

Department of Electrical and Computer Engineering, Northeastern University, Boston, Massachusetts, USA

## Abstract

As still in infancy, emerging technologies are exhibiting high defect rates; this is mostly due to the stochastic nature of the bottom-up chemical and physical self-assembly processes that are commonly employed in manufacturing. As relationships between components are very complex, a system-level solution is commonly pursued. This challenge is commonly identified with the generic problem of building reliable systems out of unreliable components. In this paper, a novel model which exploits the universality of some circuits (such as a multiplexer), is proposed. In this model a feature that is employed in the presence of errors due to faulty gates, is given by the capability of a restorative stage to have a controlled functional relationship between inputs and the output.
Using bifurcation and its geometric representation, the gate error probability is analyzed in detail. Differently from a single gate, the 4-to-1 multiplexer based implementation is considered as instance of a multi-level universal (MLU) circuit. It is proved that in the presence of multiple faulty gates, compensation takes place among them, such that a correct output is still generated at higher threshold failure probability (for a region bounded within the values of 0.471 and 0.523) as compared with previous schemes. Due to its MLU nature, the proposed model operates in a mode that allows generality and flexibility in implementation for threshold analysis.

**Keywords:** Fault-Tolerance, Nanotechnology, Bifurcation, Multiplexing, Circuit Model.

## 1. Introduction

The fabrication and manufacturing of submicron and nanometer-scale devices are challenging processes; sensitivity to external factors (such as cosmic radiation, electromagnetic migration, interference, and thermal fluctuations) results in the almost unavoidable occurrence of permanent faults (due to defects during the manufacturing process) and transient faults (during the operational life of the system). *Tolerance* to these faults is required at all levels of design and integration. For emerging technologies (such as Quantum-dot Cellular Automata (QCA) [1]), defects have been reported at manufacturing due to the unique features of the employed processes (such as for biological and molecular implementations) [2] [3].

While device- and circuit-level solutions are sought in the initial stage of technology development (to assess process monitoring and scaling), it is anticipated that *system-level solutions* will be ultimately required to cope with the complexity of these systems and the difficulty to properly model interactions at lower levels. The seed manuscript of [4] has proposed the so-called *NAND multiplexing* technique as a possible solution to achieve a reliable system assembly from unreliable components; this scheme was originally developed to circumvent the poor reliability of tubes for the assembly of digital computers in the early 1950s. For fault tolerance, [4] utilizes a high degree of redundancy through the use of two types of circuits (majority voters and NAND gates for restoration) under the assumption that circuits are made of not fully reliable components. [4] has proved that if the failure probability of the gates in the components is sufficiently small and statistically independence is assumed for failures, then computation can be reliably performed with high probability. A similar scenario is also applicable to nanotechnology [3]. High defect rates have been experienced for components at submicron and nano ranges, especially in molecular implementations [5]. Moreover as a high

probability of failure of components is encountered, new fault tolerant techniques must employ redundancy for assembling systems which are expected to have an extremely large number of components.

Nanotechnology components are *inherently unreliable* [3] and corrective actions are required because due to the high density and large number of defective components [2], it is not possible to replace or isolate them. As a *restorative stage* for fault tolerance, NAND multiplexing has been analyzed in the literature [6] [7]; bounds for the maximum fault probability of each device and the reliability of NAND multiplexing have been treated extensively [7]. In 1977, [8] presented a rigorous proof to improve von Neumann's result by showing that logarithmic redundancy is sufficient for any boolean function. The work of [8] was extended in [9] to a necessary condition for at least some boolean functions; these findings have been confirmed in [10]. [11] has shown that for so-called "noisy" NAND gates, the maximum probability of failure (also known as threshold) for each component is $\frac{(3-\sqrt{7})}{4} \approx 0.08856$.

Recently, [12] has established through a reliability analysis of NAND multiplexing that the use of additional restorative stages improves performance. [13] has considered the use of restorative stages to improve system reliability; at small fault rates, an increase in the number of restorative stages improves reliability, while at high fault rates the increase in restorative stages may result in a degradation of reliability [13]. Also [13] has shown that the analysis of [12] is only partially correct. By considering a different model for the so-called "U" random permutation, it has been shown that the results of [12] are not always the upper or lower reliability bounds.

The objective of this paper is to propose a new implementation for the restorative stage in assemblying nano-systems made of highly unreliable components. This arrangement is based on a novel model by which multi-level universal (MLU) operation occurs in the circuit. Dynamic operation refers to the capability of *selectively adjusting* the functional relation between the inputs and the output, such that through a *multi-level implementation*, multiple failures can be tolerated in the gates, while still providing on a probabilistic basis the correct value at the output.

This effect is referred to as *compensation* and in this paper, it will be proved that in a MLU circuit, the probability of compensation occurrence depends *only* on the number of gates, not on the function of the model and the input values. A multiplexer (in a 4-to-1 arrangement) is proposed for *implementation*. The compensation provided by this multiplexer implementation is extremely close to the theoretical limit, thus providing *near optimality* for application to nano-systems.

The paper is organized as follows. In Section 2, the basic principles of bifurcation analysis are provided for completeness using a geometric interpretation. In Section 3, a Generic Dynamic Model is proposed; as implementation, a 4-to-1 multiplexer is treated in detail. Section 4 presents a detailed analysis of error compensation. Section 5 extends the bifurcation analysis to the multiplexer implementation. The last Section presents some very important observations as result of the presented analysis and concludes this paper.

# 2. Bifurcation Analysis

This section presents in detail the bifurcation analysis of two-input NAND/NOR gates as in a restorative stage; this analysis is included for completeness and to address issues related to multiplexing using these gates. In particular, probabilistic logic and the associated concept of reliable computation are introduced.

## 2.1. NAND Gates

Consider an individual two-input NAND gate in NAND multiplexing. As for probabilistic logic, only the probability of an input or output being in the conventional logic state of "1" is relevant, then all signals in a circuit made of noisy gates are also analyzed as probabilities [4] [14] [11]. Let $X$ and $Y$ denote the probabilities of the two inputs being "1" and "0" Under the assumption that the two inputs are independent, the probability $Z$ of the output being "1" is:

$$z = (1-\epsilon)(1-XY) + \epsilon XY = (1-\epsilon) + (2\epsilon - 1)XY \qquad (1)$$

As analyzed in [6], the worst case scenario in terms of computation reliability occurs for the equality $X=Y$ in Eq.1. For this case as in previous papers [6], a full binary tree of faulty NAND gates (as shown in Figure 1) must be analyzed for assessing reliable computation on a probabilistic basis.

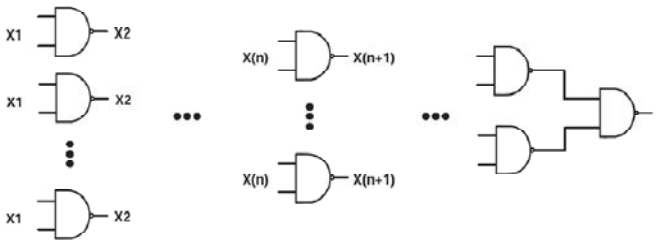


Figure 1. NAND Binary Tree

In the analysis, it is assumed that this is a discrete time system; the leaves and root of the tree correspond to the start and end points of the computation, respectively [6]. Under the assumption that all inputs to the leaf NAND gates are independent and have equal probabilities of being "1" (denoted by $X_1$), then this structure guarantees that the inputs to all gates at an arbitrary stage (given by $n$) are also independent and have equal probabilities of being "1" (denoted by $X_n$). For this circuit, Eq.1 reduces to the following iterative equation (or commonly referred to as the *map*).

$$X_{n+1} = f(X_n) = (1-\epsilon) + (2\epsilon - 1)X_n^2 \qquad (2)$$

A bifurcation of Eq.2 is employed to establish the dependence of the signal propagation in this circuit on the *Gate Error Probability* (GEP) denoted by $\epsilon$. For each $\epsilon \in (0,1)$, an arbitrary initial condition (given by $X_1$) is chosen. A sequence $X_i$, $i= 2,...,n,...$ is generated by iterating Eq.2 until convergence to the attractors is observed. The attractors for the $X_i$ sequence are then plotted in Figure 2.

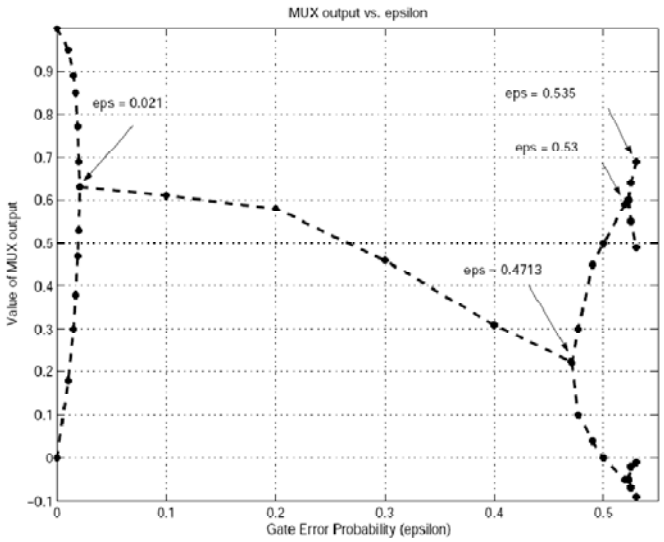


Figure 2. Bifurcation Map Diagram for a NAND

This diagram reveals that a period-doubling bifurcation occurs a $\epsilon_* = \frac{(3-\sqrt{7})}{4} \approx 0.08856$ t. An understanding of this mathematical formulation can be attained as follows. Initially the fixed point solution for the map $X_{n+1}=f(X_n)$ is found. By solving:

$$X_f = (1-\epsilon) + (2\epsilon - 1)X_f^2 \tag{3}$$

the following equation is obtained.

$$X_f = \frac{-1+\sqrt{4(1-\epsilon)(1-2\epsilon)+1}}{2(1-2\epsilon)} \tag{4}$$

These fixed points are plotted in Figure 3.

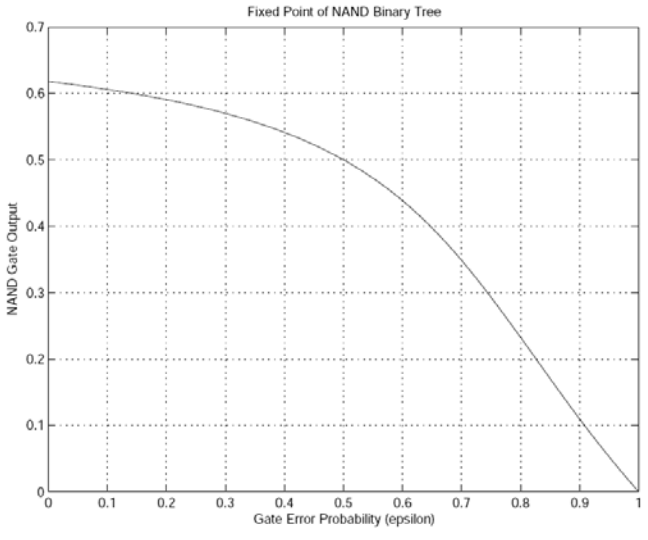


Figure 3. Fixed Points for a NAND Binary Tree

Consider next the stability of a general map,

$$X_{n+1} = g(X_n) \tag{5}$$

Let $X_*$ denote a fixed point solution of the map that satisfies, the condition:

$$g(X_*) = X_* \tag{6}$$

Let $X_n=X_*+\delta_n$, where $\delta_n$ is the deviation of $X_n$ from $X_*$. Assume as initial step $X_1=X_*+\delta_1$, where $\delta_1$ is a small deviation from $X_*$. From Eq.5,

$$X_2 = g(X_* + \delta_1) \tag{7}$$

Using a Taylor's series expansion for $X_*$,

$$X_2 = g(X_* + \delta_1) = g(X_*) + g'(X_*)\delta_1 + O(\delta_1^2) \tag{8}$$

By Eq.6 and neglecting the term $O(\delta_1^2)$, the following equation is obtained.

$$\delta_2 = g'(X_*)\delta_1 \tag{9}$$

By repeating this process, the derivative is found as:

$$\delta_{n+1} = \left(g'(X_*)\right)^n \delta_1 \tag{10}$$

If $|g'(X_*)| < 1$, then $\delta_n \to \infty$ as $n \to \infty$ and the fixed point $X_*$ is locally stable. Conversely, if $|g'(X_*)| > 1$, then the fixed point is unstable. For the marginal case when $|g'(X_*)| = 1$, the stability of $X_*$depends on $O(\delta_1^2)$. So, starting from a point in the neighborhood of $X_*$, the orbit is either attracted to or repelled from $X_*$in the long run depending on whether $|g'(X_*)| < 1$ or $|g'(X_*)| > 1$. For the stable case, the smaller $|g'(X_*)|$is, the faster the convergence. $\epsilon_*$ is found such that $|g'(X_*)| = 1$.

Applying the above analysis to Eq.2, it is possible to understand the system behavior in the two regions. So,

$$f'(X_*) = (4\epsilon - 2)X_* = 1 - \sqrt{8\epsilon^2 - 12\epsilon + 5} \tag{11}$$

Eq.11 is plotted in Figure 4.

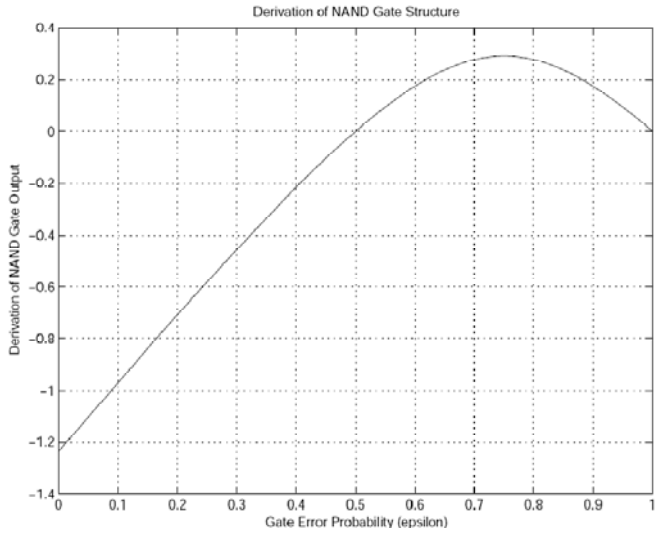


Figure 4. Output Function Derivative for a NAND Binary Tree

For $\epsilon_*$,

$$|f'(X_*)| = 1 \tag{12}$$

Then,

$$\sqrt{8\epsilon^2 - 12\epsilon + 5} = 0 \text{ or } 2 \quad (13)$$

As it is known that:

$$\forall\epsilon: \quad \sqrt{8\epsilon^2 - 12\epsilon + 5} \geq \frac{\sqrt{2}}{2} \quad (14)$$

therefore,

$$\sqrt{8\epsilon^2 - 12\epsilon + 5} = 2 \quad (15)$$

By the above equation, it is found that $\epsilon_* = \frac{(3-\sqrt{7})}{4} \approx 0.08856$.

## 2.2. NOR Gates

If NAND gates are replaced by NOR gates the analysis is initially different, So,

$$z = (1-\epsilon) - (2\epsilon - 1)XY + (2\epsilon - 1)(X + Y) \quad (16)$$

By applying the same conditions for *X* and *Y*,

$$f(X_n) = X_{n+1} = (1-\epsilon) - (2\epsilon - 1)X_n^2 + 2(2\epsilon - 1)X_n \quad (17)$$

For finding the fixed points,

$$X_f = f(X_f)$$
$$X_f = (1-2\epsilon)X_n^2 + (4\epsilon - 2)X_f + (1-\epsilon)$$
$$X_f = \frac{(3-4\epsilon) - \sqrt{(4\epsilon-3)^2 - 4(1-2\epsilon)(1-\epsilon)}}{2(1-2\epsilon)}$$
$$X_f = \frac{(3-4\epsilon) - \sqrt{8\epsilon^2 - 12\epsilon + 5}}{2(1-2\epsilon)} \quad (18)$$

Eq.18 can be written as:

$$X_f = 1 - \frac{-1 + \sqrt{4(1-\epsilon)(1-2\epsilon)+1}}{2(1-2\epsilon)} \quad (19)$$

Therefore for the fixed points Eq.18= 1 - Eq.4; this is depicted in Figure 5.

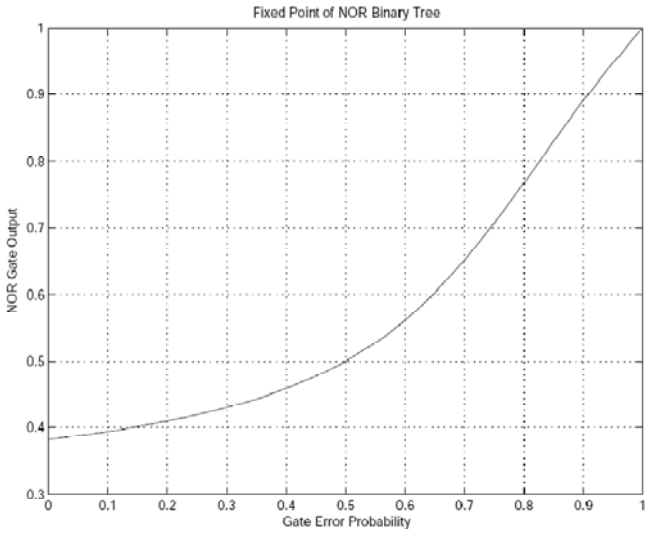


Figure 5. Fixed Points for a NOR Binary Tree

the derivative of Eq.17 is:

$$f'(X) = (2-4\epsilon)X + 2(2\epsilon - 1) = (3-4\epsilon) - \sqrt{8\epsilon^2 - 12\epsilon + 5} + 2(2\epsilon - 1) = 1 - \sqrt{8\epsilon^2 - 12\epsilon + 5} \quad (20)$$

Eq.20 is the same as Eq.11; so, the same $\epsilon_*$is found. Also as shown in Figure 6, the NOR bifurcation map diagram is symmetrical with the NAND bifurcation plot at *X=0.5*.

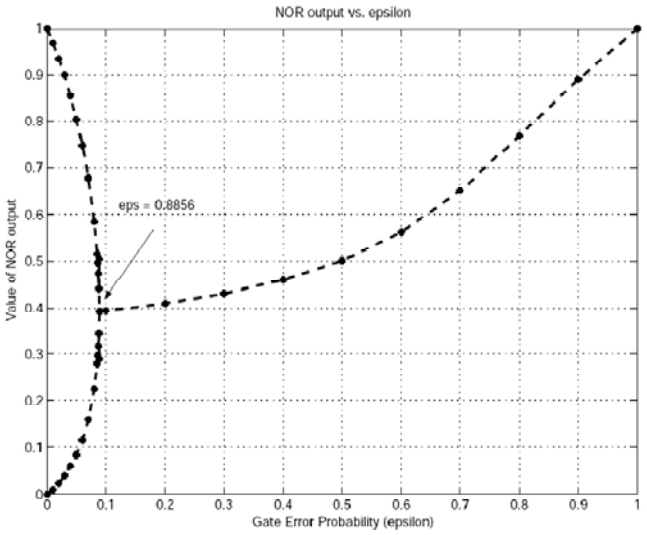


Figure 6. Bifurcation map for a NOR Binary Tree

Using bifurcation [15] [6], the error threshold value for a two-input NAND gate of $\epsilon_* \approx 0.08856$ (commonly referred to as the threshold gate failure probability) was reported in [11] [6]. This threshold value reveals two parametric intervals of $\epsilon$ for reliable computation [6]. Above analysis shows that for restoration a NAND/NOR structure can be used as an effective restorative stage provided that the GEP is less than $\epsilon_* = \frac{(3-\sqrt{7})}{4} \approx 0.08856$. Also note that for avoiding confusion between the "1" and "0" states, these structures must be applied to systems with GEP very close, albeit different from *0*. Nanotechnology exhibits high fault rates in devices and components [3]; a GEP of 0.088 is very limiting at this stage of technology development. Therefore, different structures are urgently needed for assembly nano-based systems [3]. These structures should preferably have a very high GEP, so that the large density that is expected in these systems, can be exploited for enhancing the reliable system assembly. In the next section, a new generic model is proposed; using this model and its multi-level implementation in the restorative stage, it is possible to attain a $\epsilon_*$ that achieves a higher GEP.

# 3. Generic Dynamic Model

In this section, a new generic model for assemblying reliable systems from unreliable components is proposed. This model utilizes as implementation the universal nature of a multiplexer and its flexibility in generating different circuits inclusive of NAND. In the proposed model, the so-called *MLU feature* of a circuit is utilized. A circuit is said to be MLU provided if it has the feature of changing its output using control signals (as selectors). Figure 7 shows a MLU circuit model that satisfies the previous definition. This circuit has four input functions and single output; few control lines (as selectors) are also provided. As an example of a

nanotechnology, a QCA implementation of this circuit would require fixed polarity cells for the *Fi* signals, so only the two input selector lines (as in a NAND gate) would be required [1]. Without loss of generality and correctness, the model can be extended to *m* input lines with $log_2 m$ control lines per output. Examples of circuit that exhibit MLU behavior in implementation, are memories (such as look-up-tables) and multiplexers.

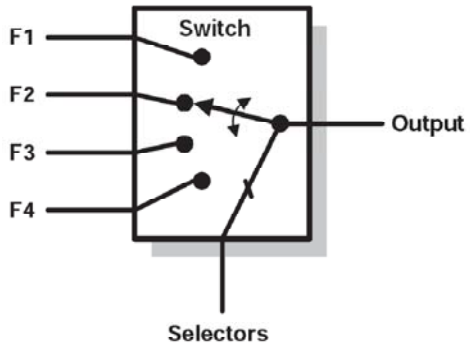


Figure 7. Generic Dynamic Circuit Model

As in previous methods found in the literature [6], the behavior of the proposed model is analyzed in terms of its functional features for fault tolerance in assemblying unreliable components at system level. In particular, the MLU behavior of the multiplexer together with its logic universality (i.e. the capability to generate any combinational function at its output) is characterized with respect to its ability to probabilistically handle multiple component failures (as defined through a gate error probability) in a MLU circuit. This is made possible by the control lines that selectively handle the functional connection between one of the fixed value inputs and the output. Moreover due to its multi-level implementation, a MLU circuit can result in a behavior that handles faults in a non a-priori fixed arrangement among its gate-level structure.

As an example, assume only a single NAND gate (as in previous papers [6]) is utilized as a building block of a probabilistic structure for restoration. As a single gate, a NAND is static because no additional controllability can be exercised over the inputs. This is not applicable to a MLU circuit; under the proposed MLU model, the following parameters can be defined for controllability:

- The probability of the two input signals as selectors is given by.

$$P^1_{In_1} = X, \quad P^0_{In_1} = 1 - X \tag{21}$$

$$P^1_{In_2} = Y, \quad P^0_{In_2} = 1 - Y \tag{22}$$

- The probability of gate error (for both cases of faulty and error free, also referred to as perfect) is given as follows.

$$P_{page}(faulty) = \epsilon \tag{23}$$

$$P_{page}(error\ free\ or\ perfect) = 1 - \epsilon \tag{24}$$

Next, consider a two-input NAND gate implemented as a MLU circuit under the proposed model. The multiplexer is a universal circuit that can implement any combinational function, so it is also possible to implement the functionality of the NAND gate through the proposed MLU circuit of Figure 7 as a 4-to-1 multiplexer (Figure 8).

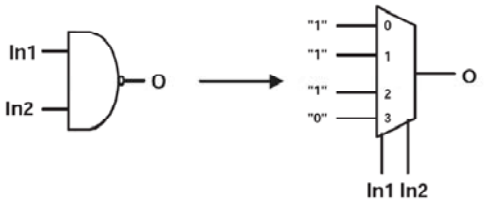


Figure 8. Multiplexer Implementation of NAND under the Generic Dynamic Model

As shown in Figure 9, the multiplexer implementation under this model has three gate levels:

I. Output OR gate.
II. Four AND gates.
III. Two Inverters.

By comparing the multiplexer implementation to the generic.

MLU circuit of Figure 7, the following conditions are applicable (the gate-level structure of the proposed multiplexer implementation is depicted in Figure 9):

$$F_1 = F_2 = F_3 = \text{"1"} \tag{25}$$

$$F_4 = \text{"0"} \tag{26}$$

$$Selectors = In_1, In_2 \tag{27}$$

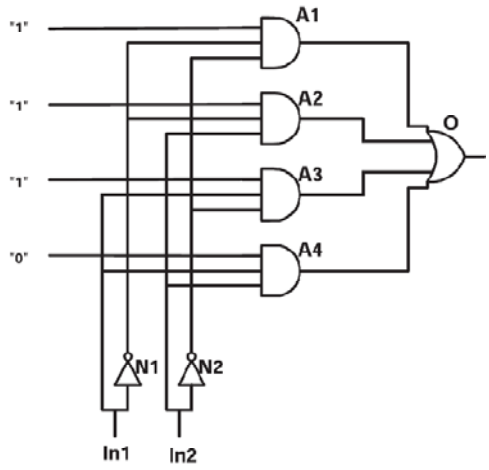


Figure 9. Gate-Level Implementation of the Multiplexer based NAND under the Generic Dynamic Model

Assume that each faulty gate causes a bit flip in its output and $In_1$=$In_2$=”0”; in the error free circuit, the output is “1” (Figure 10). As shown in Figure 11, if $A_1$ is faulty, then the output will be faulty too (“0”); however if both $A_1$ and $N_2$ are faulty, then the output is error free (Figure 12), i.e. the combination of two faulty gates results in an error free output. This effect is referred in this paper as *compensation*, i.e. for the above example, a fault in $N_2$ compensates the fault in $A_1$. This is a feature of MLU circuits that is caused by the multi-level implementation in the proposed model (assuming independence in the gates' failures). This behavior however, does not occur in a NAND gate as a single device, hence

static structures (such as a single NAND gate) have no compensation. Such feature can be analyzed on a probabilistic basis; for a static circuit the error compensation probability is *0*, while its implementation as a MLU circuit results in a non zero probability. As a large number of faulty gates is present in nano scale systems [3], then it is realistic to expect that compensation in MLU circuits can be used for designing more reliable circuits. This aspect will be rigorously analyzed next.

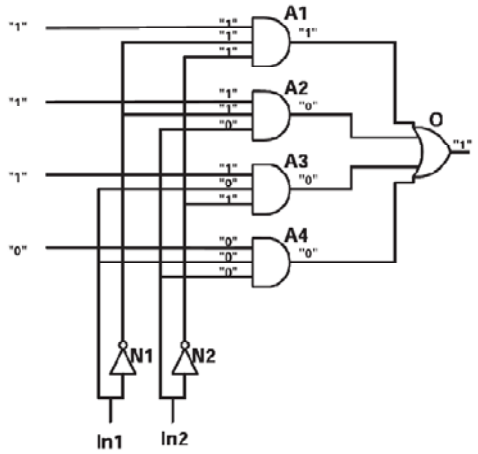


Figure 10. Fault-Free Multiplexer-based NAND Structure

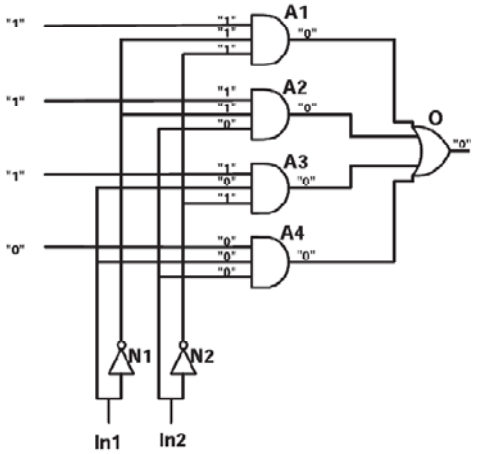


Figure 11. A1 is Faulty

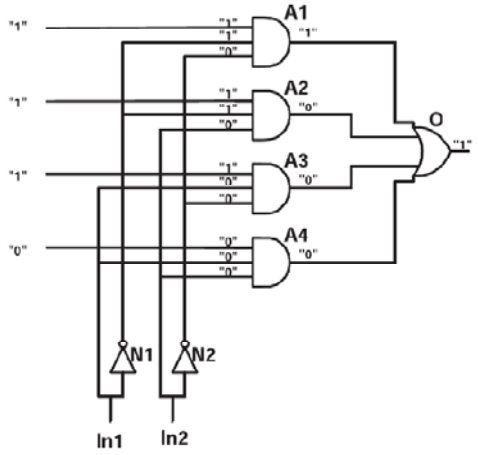


Figure 12. A1 and N2 are Faulty

# 4. Error Compensation

With no redundancy, it will be shown that the error compensation of a MLU circuit depends on the number of gates. The circuit is defined by the following parameters:

- *n* denotes the number of gates in the circuit.
- *i* is the number of faulty gates in the circuit.
- *Err(i)* is the number of states for which no compensation occurs in the presence of *i* faulty gates (or errors).
- *Comp(i)* is the number of states for which compensation occurs in the presence of *i* faulty gates (or compensations).

Therefore for each *i*,

$$Err(i) + Comp(i) = \binom{n}{i} \tag{28}$$

For all states,

$$\sum_{i=1}^{n} Err(i) + \sum_{i=1}^{n} Comp(i) = \sum_{i=1}^{n} \binom{n}{i} = 2^n - 1 \tag{29}$$

Also in the proposed model and implementation, four distinct states are possible in the system; these states are given as follows:

**i. All gates are fault free (perfect):** no compensation and no error are present.

**ii. All gates are perfect except the output gate (OR gate):** an error will be present at the output of the circuit.

**iii. All other states in which the output gate is perfect:** $\frac{2^n-2}{2}$ states exist. In this case, assume that the total number of errors is equal to *K* and the total number of compensations is given by *L*, so.

$$\sum_{i=1}^{n} Err(i) = K \tag{30}$$

$$\sum_{i=1}^{n} Comp(i) = L \tag{31}$$

$$K + L = \frac{2^n-2}{2} \tag{32}$$

**iv. All other states in which the output gate is faulty:** again $\frac{2^n-2}{2}$ states are possible. In this case the same scenario as in iii above is applicable, but there is an inversion at the output (because the output gate is faulty). Therefore, the total number of errors is equal to the total number of compensations in the previous state *L* and the total number of compensations is equal to the total number of errors in the previous state *K*.

So, the total numbers of errors and compensations for all states are as follows:

$$\sum_{i=1}^{n} Err(i) = K + L + 1 = 2^{n-1} \tag{33}$$

$$\sum_{i=1}^{n} Comp(i) = K + L = 2^{n-1} - 1 \tag{34}$$

In percentage, compensation of the proposed implementation of the MLU model is given by,

$$Compensate\% = \frac{\sum_{i=1}^{n} Comp(i)}{\sum_{i=1}^{n} Err(i) + \sum_{i=1}^{n} Comp(i)} = \frac{2^{n-1}-1}{2^n-1} \tag{35}$$

The above analysis shows that compensation depends only to the number of gates, not on the function of the model or the inputs value.

For a very large number of gates,

$$\lim_{n\to\infty}\left(\frac{2^{n-1}-1}{2^n-1}\right)=\frac{1}{2}=50\% \quad (36)$$

In the proposed 4-to-1 multiplexer based implementation of a NAND, *n=7*, so its compensation is given by,

$$Compensation=\frac{2^6-1}{2^7-1}=\frac{63}{127}=49.606299\% \quad (37)$$

For *n*=1 (a single NAND gate), the compensation is 0, thus proving the limitations of existing NAND multiplexing arrangements. The above analysis proves also that for the proposed NAND MLU implementation (using a 4-to-1 multiplexer as an instance of a circuit), compensation is very close to the maximum theoretical limit.

Table 1 shows the implementations of the 4-to-1 multiplexer using two-input NAND and NOR gates as of a MLU circuits. Each table relates to a input value for the two control lines (in all cases *n*=7 as number of gates in the implementation). The "# of Faulty Gates" is the number of injected faults (or faulty gates) in the multiplexer implementation. The "# of Err (Comp)" denotes the number of cases for which we have an error (compensation occurs) at the output for each case of possible fault(s). Note that the sum of "# of Err" and "# of Comp" is equal to the "# of Faulty Gates". "% comp" is the percentage of the ratio of ("# of Comp") and ("# of Comp" + "# of Err"). Error compensation is depend on the inputs and the number of faulty gates but the total compensation are equal (49.606299%) for different input set. Considering the most frequent input set which a circuit may use and the error compensation of different gates, it is possible to select the best gate for implementing the proposed multiplexer used in a Nand-Maultiplexing system.

# 5. Bifurcation of Multiplexer-based NAND

As proved in previous sections, compensation is a powerful feature to consider when assemblying nano-systems using unreliable components. The multiplexer implementation as instance of a MLU circuit has a compensation very near to its theoretical limit; this feature will be further analyzed in the binary tree structure by replacing the NAND gates with the multiplexer implementation of a MLU circuit; bifurcation is used to generate its map.

Table 1: Multiplexer error compensation using NAND and NOR gates

| **NAND (In : 00)** | | | | **NAND (In : 11)** | | | |
|---|---|---|---|---|---|---|---|
| # Faulty Gates | # Err | # Comp | % Comp | # Faulty Gates | # Err | # Comp | % Comp |
| 1 | 4 | 3 | 42.857143 | 1 | 7 | 0 | 0 |
| 2 | 4 | 17 | 80.952377 | 2 | 13 | 8 | 38.095238 |
| 3 | 14 | 21 | 60 | 3 | 22 | 13 | 37.142857 |
| 4 | 20 | 15 | 42.857143 | 4 | 15 | 20 | 57.142857 |
| 5 | 15 | 6 | 28.571428 | 5 | 5 | 16 | 76.190475 |
| 6 | 6 | 1 | 14.285714 | 6 | 2 | 5 | 71.428574 |
| 7 | 1 | 0 | 0 | 7 | 0 | 1 | 100 |
| **Total Compensation = 49.606299** | | | | **Total Compensation = 49.606299** | | | |
| **NAND (In : 01)** | | | | **NAND (In : 10)** | | | |
| 1 | 3 | 4 | 57.142857 | 1 | 3 | 4 | 57.142857 |
| 2 | 5 | 16 | 76.190475 | 2 | 5 | 16 | 76.190475 |
| 3 | 15 | 20 | 57.142857 | 3 | 15 | 20 | 57.142857 |
| 4 | 19 | 16 | 45.714287 | 4 | 19 | 16 | 45.714287 |
| 5 | 15 | 6 | 28.571428 | 5 | 15 | 6 | 28.571428 |
| 6 | 6 | 1 | 14.285714 | 6 | 6 | 1 | 14.285714 |
| 7 | 1 | 0 | 0 | 7 | 1 | 0 | 0 |
| **Total Compensation = 49.606299** | | | | **Total Compensation = 49.606299** | | | |
| **NOR (In : 00)** | | | | **NOR (In : 11)** | | | |
| # Faulty Gates | # Err | # Comp | % Comp | # Faulty Gates | # Err | # Comp | % Comp |
| 1 | 4 | 3 | 42.857143 | 1 | 5 | 2 | 28.571428 |
| 2 | 4 | 17 | 80.952377 | 2 | 17 | 4 | 19.047619 |
| 3 | 14 | 21 | 60 | 3 | 19 | 16 | 45.714287 |
| 4 | 20 | 15 | 42.857143 | 4 | 16 | 19 | 54.285713 |
| 5 | 15 | 6 | 28.571428 | 5 | 6 | 15 | 71.428574 |
| 6 | 6 | 1 | 14.285714 | 6 | 1 | 6 | 85.714287 |
| 7 | 1 | 0 | 0 | 7 | 0 | 1 | 100 |
| **Total Compensation = 49.606299** | | | | **Total Compensation = 49.606299** | | | |
| **NOR (In : 01)** | | | | **NOR (In : 10)** | | | |
| 1 | 6 | 1 | 14.285714 | 1 | 6 | 1 | 14.285714 |
| 2 | 14 | 7 | 33.333332 | 2 | 14 | 7 | 33.333332 |
| 3 | 22 | 13 | 37.142857 | 3 | 22 | 13 | 37.142857 |
| 4 | 15 | 20 | 57.142857 | 4 | 15 | 20 | 57.142857 |
| 5 | 6 | 15 | 71.428574 | 5 | 6 | 15 | 71.428574 |
| 6 | 1 | 6 | 85.714287 | 6 | 1 | 6 | 85.714287 |
| 7 | 0 | 1 | 100 | 7 | 0 | 1 | 100 |
| **Total Compensation = 49.606299** | | | | **Total Compensation = 49.606299** | | | |

In the multiplexer structure, it is assumed (as commonly found in existing literature [6]) that all gates have the same GEP ($\epsilon$) and all connections are fault free (perfect).

In the analysis, $N_i=0$ denotes that "Inverter $i$ is error free" while $N_i=1$ denotes that "Inverter $i$ is faulty".

Similar definitions apply also to the AND gates ($A_i$) and OR gate ($O$). These are binary variables, so $\sum_{i=1}^{4} A_i \geq 1$ means that "at least one of the AND gates is faulty".

For calculating the probability of having "1" at the output, all cases must be considered. Four states are possible for the values of the inputs of the two control lines under the exhaustive combination of the status of the two inverters.

## 5.1. $In_1$=$In_2$=1

When the two control input lines are in the "1" state, the probability of having "1" at the output can be calculated as follow:

### 5.1.1. $N_1$ and $N_2$ Both Perfect

When the inverters are perfect, a "0" value will be present at the inputs of all AND gates (for $In_1=In_2=$"1"). So for an error free circuit, "0" is the correct value at the output; however, if a fault is present in one of the gates at the AND or Output levels, the output will change to "1". Therefore, the probability of having a "1" at the output is given as follows:

$$\underbrace{(1-\epsilon)}_{N_1=0}\underbrace{(1-\epsilon)}_{N_2=0}\left[\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}+\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\underbrace{\epsilon}_{O=1}\right] \tag{38}$$

### 5.1.2. $N_1$ Perfect and $N_2$ Faulty

In this case, an AND gate with all inputs equal to "1" ($A_3$) is present; therefore, a "1" will appear at the output of the multiplexer for the states given below.

i. $A_3$ and at least one of the other AND gates are faulty and the output gate is fault free.
ii. $A_3$ and the output gate are fault free.
iii. $A_3$ and the output gate are faulty and the other AND gate is fault free.

So for this state, the following is applicable:

$$\underbrace{(1-\epsilon)}_{N_1=0}\underbrace{\epsilon}_{N_2=1}\left[\left(\underbrace{\epsilon}_{A_3=1}\underbrace{(1-(1-\epsilon)^3)}_{\sum_{i=1,2,4}A_i\geq 1}\underbrace{+(1-\epsilon)}_{A_3=0}\right)\times\underbrace{(1-\epsilon)}_{O=0}+\underbrace{(1-\epsilon)^3}_{\sum_{i=1,2,4}A_i=0}\underbrace{\epsilon}_{A_3=1}\underbrace{\epsilon}_{O=1}\right] \tag{39}$$

### 5.1.3. $N_1$ Faulty and $N_2$ Perfect

Also in this case, an AND gate with all inputs equal to "1" ($A_2$), is present. Hence, a "1" will appear at the output of the multiplexer for the following states:

i. $A_2$ and at least one of the other AND gates are faulty and the output gate is fault free.
ii. $A_2$ and the output gate are fault free.
iii. $A_2$ and the output gate are faulty, while the other AND gates is fault free.

So for this state,

$$\underbrace{\epsilon}_{N_1=1}\underbrace{(1-\epsilon)}_{N_2=0}\left[\left(\underbrace{\epsilon}_{A_2=1}\underbrace{(1-(1-\epsilon)^3)}_{\sum_{i=1,3,4}A_i\geq 1}\underbrace{+(1-\epsilon)}_{A_2=0}\right)\times\underbrace{(1-\epsilon)}_{O=0}+\underbrace{(1-\epsilon)^3}_{\sum_{i=1,3,4}A_i=0}\underbrace{\epsilon}_{A_2=1}\underbrace{\epsilon}_{O=1}\right] \tag{40}$$

### 5.1.4. $N_1$ and $N_2$ Both Faulty

Three AND gates have all inputs equal to "1" ($A_1$,$A_2$,$A_3$). So, the probability of having "1" at the output is.

$$\underbrace{\epsilon}_{N_1=1}\underbrace{\epsilon}_{N_2=1}\times\left[1-\left(\underbrace{\epsilon^3(1-\epsilon)\overbrace{(1-\epsilon)}^{O=0}+\left(1-\epsilon^3(1-\epsilon)\overbrace{\epsilon}^{O=1}\right)}_{A_2=1}\right)\right] \tag{41}$$

By adding Eq.38, Eq.39, Eq.40, Eq.41 the following equation holds for $In_1=In_2=1$.

$$P^1_{In_1=In_2=1}=4\epsilon^7-28\epsilon^6+67\epsilon^5-83\epsilon^4+62\epsilon^3-29\epsilon^2+7\epsilon \tag{42}$$

## 5.2. $In_1$=$In_2$=0

### 5.2.1. $N_1$ and $N_2$ Both Perfect

An AND gate has all inputs equal to "1" ($A_1$). So, the probability of having "1" at the output is.

$$\underbrace{(1-\epsilon)}_{N_1=0}\underbrace{(1-\epsilon)}_{N_2=0}\left[\left(\underbrace{(1-\epsilon)}_{A_1=0}+\underbrace{\epsilon}_{A_1=1}\underbrace{(1-(1-\epsilon)^3)}_{\sum_{i=2,3,4}A_i\geq 1}\right)\times\underbrace{(1-\epsilon)}_{O=0}+\underbrace{\epsilon}_{A_1=1}(1-\epsilon)^3\underbrace{\epsilon}_{O=1}\right] \tag{43}$$

### 5.2.2. $N_1$ Perfect and $N_2$ Faulty

In this case all AND gates have at least one input equal to "0", so the probability of having "1" at the output is.

$$\underbrace{(1-\epsilon)}_{N_1=0}\underbrace{\epsilon}_{N_2=1}\left[\left(\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\underbrace{\epsilon}_{O=1}+\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}(1-\epsilon)\right)\right] \tag{44}$$

### 5.2.3. $N_1$ Faulty and $N_2$ Perfect

Again, all AND gates have at least one input equal to "0" and the probability of having a "1" at the output is the same as Eq.44.

### 5.2.4. $N_1$ and $N_2$ Both Faulty

All AND gates have at least two inputs equal to "0", so the following equation is applicable.

$$\underbrace{\epsilon}_{N_1=1}\ \underbrace{\epsilon}_{N_2=1}\left[\left(\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\ \underbrace{\epsilon}_{O=1}+\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}\right)\right] \quad (45)$$

By adding these equations, the equation for total probability of the output for the case of $In_1=In_2=0$, is given by

$$P^1_{In_1=In_2=0} = -4\epsilon^7+24\epsilon^6-59\epsilon^5+76\epsilon^4-54\epsilon^3+20\epsilon^2-4\epsilon+1 \quad (46)$$

## 5.3. $In_1=0$, $In_2=1$ or $In_1=1$, $In_2=0$

### 5.3.1. $N_1$ and $N_2$ Both Perfect

For the $In_1$=0, $In_2$=1 case, an AND gate with all inputs equal to "1" ($A_2$) is present, while for the $In_1$=1, $In_2$=0 case, the same scenario of gate $A_3$ is applicable. Therefore, the output probability is

$$\underbrace{(1-\epsilon)}_{N_1=0}\underbrace{(1-\epsilon)}_{N_2=0}\left[\left(\underbrace{(1-\epsilon)}_{A_3=0}+\underbrace{\epsilon}_{A_3=1}\underbrace{(1-(1-\epsilon)^3)}_{\sum_{i=1,2,4}A_i\geq 1}\right)\times\underbrace{(1-\epsilon)}_{O=0}+\underbrace{\epsilon}_{A_3=1}(1-\epsilon)^3\underbrace{\epsilon}_{O=1}\right] \quad (47)$$

### 5.3.2. $N_1$ Perfect and $N_2$ Faulty

For the $In_1$=0, $In_2$=1 case, two AND gates with all inputs equal to "1" ($A_1$,$A_2$) are present, so,

$$\underbrace{(1-\epsilon)}_{N_1=0}\ \underbrace{\epsilon}_{N_2=1}\left[\underbrace{(1-\epsilon^2)}_{A_1+A_2\leq 1}+\epsilon^2\underbrace{(1-(1-\epsilon)^2)}_{\sum_{i=3,4}A_i\geq 1}+\epsilon^2(1-\epsilon)^2\ \underbrace{\epsilon}_{O=1}\right] \quad (48)$$

For the $In_1=1$, $In_2=0$ case, all AND gates have at least one input equal to "0", i.e.

$$\underbrace{(1-\epsilon)}_{N_1=0}\ \underbrace{\epsilon}_{N_2=1}\left[\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\ \underbrace{\epsilon}_{O=1}+\epsilon^2\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}\right] \quad (49)$$

### 5.3.3. $N_1$ Faulty and $N_2$ Perfect

For the $In_1=0$, $In_2=1$ case, all AND gates have at least one input equal to "0" and,

$$\underbrace{\epsilon}_{N_1=1}\ \underbrace{(1-\epsilon)}_{N_2=0}\left[\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\ \underbrace{\epsilon}_{O=1}+\epsilon^2\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}\right] \quad (50)$$

For the $In_1=1$, $In_2=0$ case, two AND gates with all inputs equal to "1" ($A_1$,$A_3$) are present, therefore.

$$\underbrace{\epsilon}_{N_1=1}\ \underbrace{(1-\epsilon)}_{N_2=0}\left[\underbrace{(1-\epsilon^2)}_{A_1+A_3\leq 1}\underbrace{(1-\epsilon)}_{O=0}+\epsilon^2\underbrace{(1-(1-\epsilon)^2)}_{\sum_{i=2,4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}+\epsilon^2(1-\epsilon)^2\ \underbrace{\epsilon}_{O=1}\right] \quad (51)$$

### 5.3.4. $N_1$ and $N_2$ Both Faulty

For both input states ($In_1=0$, $In_2=1$ and $In_1=1$, $In_2=0$) all AND gates have at least one input equal to "0", hence.

$$\underbrace{\epsilon}_{N_1=1}\ \underbrace{\epsilon}_{N_2=1}\left[\underbrace{(1-\epsilon)^4}_{\sum_{i=1}^{4}A_i=0}\ \underbrace{\epsilon}_{O=1}+\epsilon^2\underbrace{(1-(1-\epsilon)^4)}_{\sum_{i=1}^{4}A_i\geq 1}\underbrace{(1-\epsilon)}_{O=0}\right] \quad (52)$$

By adding the above equations the following equation is obtained for $In_1=0$, $In_2=1$ or $In_1=1$, $In_2=0$:

$$P^1_{In_1=1,In_2=0} = P^1_{In_1=0,In_2=1} = -4\epsilon^7+20\epsilon^6-43\epsilon^5+51\epsilon^4-35\epsilon^3+13\epsilon^2-3\epsilon+1$$

Therefore, it is now possible to establish the probability of having "1" at the output under the new proposed model for the 4-to-1 implementation as.

$$Z=f(X,Y)=XP^1_{In_1=In_2=1}+(1-X)(1-Y)P^1_{In_1=In_2=0}+(1-X)YP^1_{In_1=0,In_2=1}++X(1-Y)P^1_{In_1=1,In_2=0} \quad (53)$$

With the same probability for having "1" at the gate inputs ($X$=$Y$) and also $P^{I}{}_{In1=1,\ In2=0} = P^{I}{}_{In1=0,\ In2=1}$, then.

$$Z=X_{n+1}=f(X_n)=\left(P^1_{In_1=In_2=0}+P^1_{In_1=In_2=1}-2P^1_{In_1=1,In_2=0}\right)X_n^2+2\left(P^1_{In_1=1,In_2=0}-P^1_{In_1=In_2=0}\right)X_n+P^1_{In_1=In_2=0} \quad (54)$$

The derivative of this function is:

$$f'(X_{n+1})=2\left(P^1_{In_1=In_2=0}+P^1_{In_1=In_2=1}-2P^1_{In_1=1,In_2=0}\right)X_n+2\left(P^1_{In_1=1,In_2=0}-P^1_{In_1=In_2=0}\right) \quad (55)$$

$f'(x)$, the fixed points and the bifurcation map of the multiplexer implementation of the proposed MLU model are similar to the single NAND gate but the most interesting feature of this model is the fact that the above formulation of this model is really close to a fractal function.

Precisely if we had $9\epsilon^2$ instead of $13\epsilon^2$ in the $In_1=0$, $In_2=1$ (or $In_1=1$, $In_2=0$) case, the outcome would be a fractal function whose bifurcation map is plotted in Figure 13, Figure 14 shows the enlarged region of the fractal function to allow a reliable system assembly using unreliable components (prior to the fractal/chaotic region occurring at values higher than 0.523 for $\epsilon_*$).

The bifurcation map of this fractal function has four regions in the range of 0 to 1 for $\epsilon_*$.

1) The first region is periodic, but its error probability ($\epsilon$) is very close to zero, so it has very limited usefulness.

2) The second region is not periodic (fixed points).

3) The third region is also a periodic region centered at $\epsilon_* = 0.5$, and can be used for operation in assemblying reliable systems out of unreliable components. This region is bounded by 0.471 and 0.523 as the value for $\epsilon_*$ prior to the fractal region.

4) The fourth region is the so-called chaos region and therefore, it cannot be used.

So using a similar model for molecular electronic devices with a probabilistic output, it is possible to achieve a MLU with fractal behavior. This makes the proposed implementation very attractive for nanotechnology. Current

research is being pursued to establish together with DNA-based self-assembly an appropriate molecule for device implementation; the findings of investigation will be reported in future papers. Consider the model of [6]. A binary tree made of NAND/NOR gates is a discrete MLU system in which each element (NAND/NOR) on the tree has a static behavior in the presence of an error in the gate. The bifurcation map for these two binary trees shows that a periodic region occurs only near $\epsilon_* = 0$ while a long non-periodic region (fixed point) occurs over the remaining range (up to 1) of $\epsilon_*$. If the static gate element of such binary trees is changed into the multiplexer-based MLU implementation of the NAND/NOR gate, error compensation occurs with beneficial consequences as reflected in a new bifurcation map.

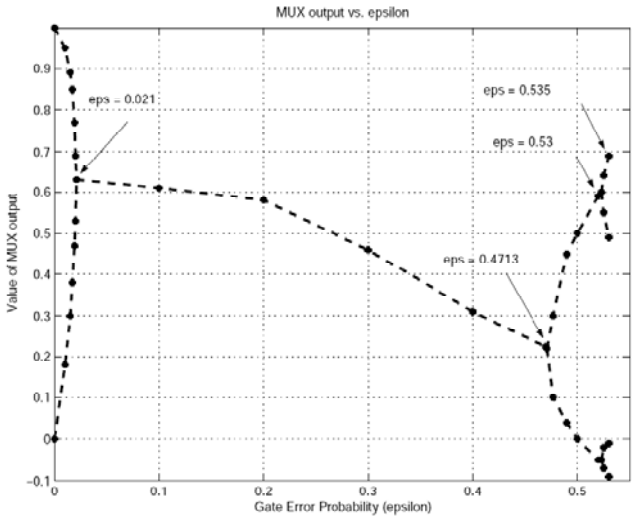


Figure 13. Bifurcation Map Diagram for Fractal Function

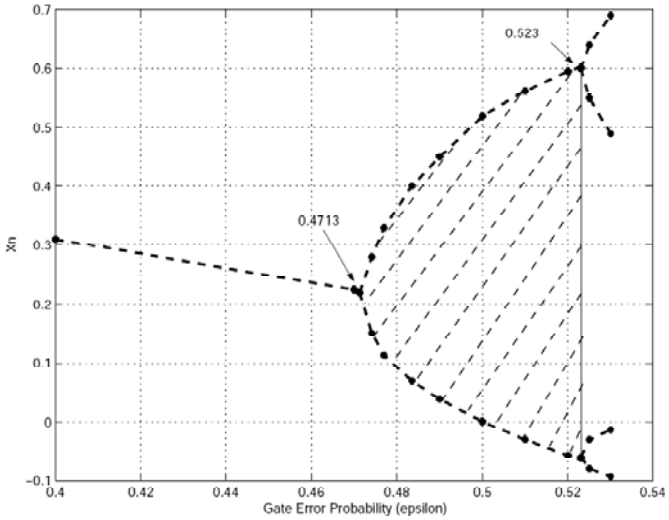


Figure 14. Enlarged Periodic Region for Fractal Function

## 6. Discussion and Conclusion

The model and implementation proposed in this paper can be compared with a multiplexing scheme using single NANDs based on different features as figures of merit.

- Figure 15 shows the plots for the derivative of a single NAND and the ideal fractal-based implementation (under the proposed model) for restoration. The derivative of the fractal-based implementation has two points for which $|f'(X)| = 1$, so more than a single area have the feature of a periodic function compared with the single NAND scheme for NAND multiplexing.
- Figure 16 shows the comparison of the fixed points in the two implementations and related models.

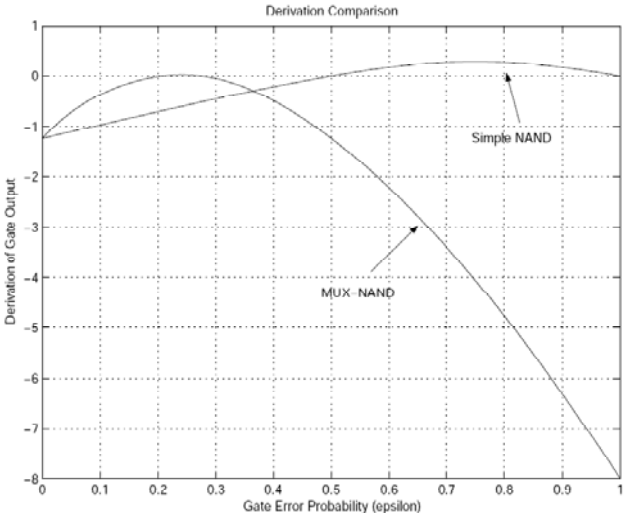


Figure 15. The Derivative Comparison of Simple-NAND and Fractal-based Implementation

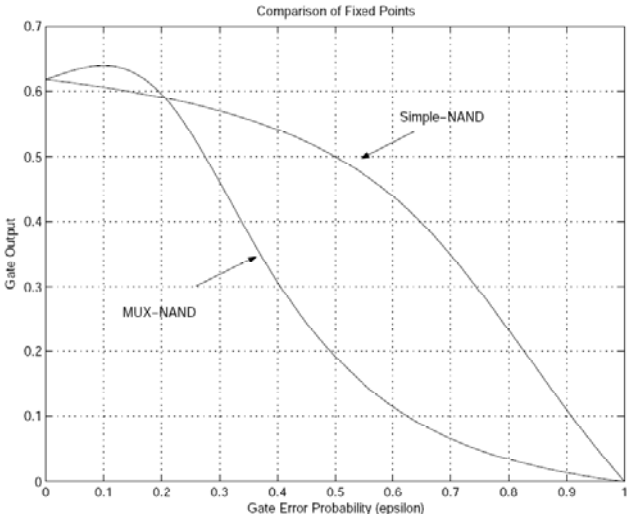


Figure 16. Fixed Point Comparison of Simple-NAND and Fractal-Based Implementation

- Figure 17 shows the bifurcation map for the two implementations. It is significant to point out that for $\epsilon_* = 0.4713$, (up to $\epsilon_* = 0.523$) reliable system assembly is now possible using the proposed implementation.

These features basically prove that the multiplexer-implementation of the NAND as an instance of a MLU circuit under the proposed model results in a significant improvement over previous arrangements (mostly single NAND based) for the restorative stage. As a multi-level implementation of a MLU circuit (such as the multiplexer) result in a higher overhead, the substantial increase in threshold failure probability (due to compensation in the extremely huge number of components) is a very positive feature for assembly nano-based systems using emerging technologies [3].

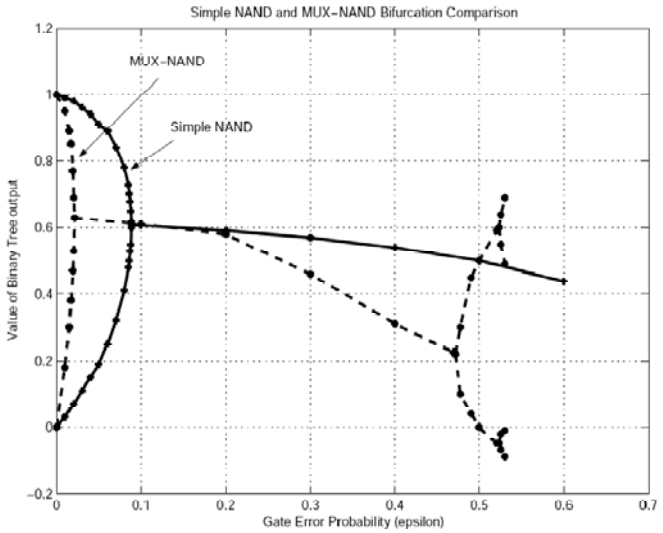


Figure 17. The Bifurcation Comparison of the Simple-NAND and Fractal-Based Implementation

The analysis reported in this paper has shown that if each gate fails independently and unbiased in a bitable fashion (such as by tossing a coin in probability theory), then over a long run (i.e. for large values of $n$) the assembly of such a system will be probabilistically possible using restorative stages implemented by multiplexer-based NANDs. Moreover due to the large density of emerging technologies, the increase in complexity of the restorative stage can be easily accommodated (albeit a two-level implementation introduces a longer delay). The 4-to-1 multiplexer implementation presented in this paper is very close to the theoretical limit by which compensation due to multiple faults can probabilistically result in a correct output by a restorative stage. The use of the proposed implementation for QCA in molecular implementations is currently being evaluated and efforts for finding a molecular implementation with fractal behavior is under investigation.

# References

[1] C. S. Lent, and P. D. Tougaw, "A device architecture for computing with quantum dots," *Proc. of the IEEE,* pp. 541-557, 1997.

[2] J. Huang, M. Momenzadeh, M. B. Tahoori, and F. Lombardi, "Defect Characterization for Scaling of QCA Devices," *Proc. of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems,* pp. 30-38, 2004.

[3] J. Han, E. Taylor, J. Gao, and J. Fortes, "Reliability Modeling of Nanoelectronic Circuits," *Proc. of the IEEE Conference on Nanotechnology,* pp. 104-107, 2005.

[4] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components." *Automata Studies*, Shannon C.E.& McCarthy J., eds., pp. 43-98, 1955.

[5] M. B. Tahoori, M. Momenzadeh, J. Huang, and F. Lombardi, "Defects and Faults in Quantum Cellular Automata," *Proc. of the 22nd VLSI Test Symposium*, pp. 291-296, 2004.

[6] Y. Qi, J. Gao, and J. A. B. Fortes, "Probabilistic Computation: A General Framework for Fault Tolerant Nanoelectronic Systems," University of Florida, Gainesville, Florida, Tech. Rep. TR-ACIS-03-002, 2003.

[7] J. B. Gao, Y. Qi, and J. Fortes, "Bifurcations and Fundamental Error Bounds for Fault-Tolerant Computations," *IEEE Transactions On Nanotechnology,* vol. 4, no. 4, pp. 395-402, 2005.

[8] R. L. Dobrushin, and S. I. Ortyukov, "Upper Bound on the Redundancy of Self-Correcting Arrangements of Unreliable Functional Elements," *Problems of Information Transmission*, vol. 13, no. 3, pp. 203-218, 1977.

[9] N. Pippenger, G. D. Stamoulis, and J. N. Tsitsiklis, "On a Lower Bound for the Redundancy of Reliable Networks with Noisy Gates," *IEEE Transactions on Information Theory*, vol. 37, no. 3, pp. 639-643, 1991.

[10] W. Evans, and N. Pippenger, "On the Maximum Tolerable Noise for Reliable Computation by Formulas," *IEEE Transactions on Information Theory*, 44, no. 3, pp. 1299-1305, 1998.

[11] J. Han, and P. Jonker, "A System Architecture Solution for Unreliable Nonoelectronic Devices," *IEEE Transactions on Nanotechnology*, vol. 1, no. 4, pp. 201-208, 2002.

[12] G. Norman, D. Parker, M. Kwiatowska, and S. K. Shukla, "Evaluating the Reliability of Defect-Tolerant Architectures for Nanotechnology with Probabilistic Model Checking," *Proc. of the International Conference on VLSI Design*, pp. 907-912, 2004.

[13] N. Pippenger, "Reliable Computation by Formulas in the Presence of Noise," *IEEE Transactions on Information Theory*, vol. 34, no. 2, pp. 194-197, 1998.

[14] S. H. Strogatz, *Nonlinear Dynamics and Chaos-With Applications to Physics, Biology, Chemistry, and Engineering*, Westview Press, 2001.

**Masoud Hashempour** received the B.S. degree in computer engineering from Isfahan University of Technology, Isfahan, Iran, in 1998 and the M.S. degree in computer engineering from Sharif University of Technology, Tehran, Iran, in 2000. In 2000, he was with Emad Semicon Co. as a designer of digital systems. In 2002, he was with Sharif AICTC Co. as a computer engineer and the manager of iTV group. In 2003, he was with Basamad Co. as the DSP group's manager. Since September 2004, he started working toward the Ph.D. degree in electrical and computer engineering and got graduated on Fall 2008 at Northeastern University, Boston, MA, USA. He has worked for EMC Co. since 2007 to 2009. His research interests include yield and reliability modeling, fault-tolerant architectures, test and testable design and design of nanoscale circuits and systems.

**E-mail:** masoud@ece.neu.edu

**Zahra Mashreghian Arani** received the B.S. degree in computer engineering from Iran University of Science and Technology, Tehran, Iran. In 2001, she was with I.R.I.B Institute as a computer engineer. In 2002, she was with Sharif AICTC Co. as an application designer in iTV group. In January 2007, she started her Ph.D. degree in electrical and computer engineering at Northeastern University, Boston, MA, USA. Her research interests include robotics, machine vision, pattern recognition and design of nanoscale circuits and systems.

**E-mail:** zahra@ece.neu.edu

**Fabrizio Lombardi** graduated in 1977 from the University of Essex (UK) with a B.Sc. (Hons.) in Electronic Engineering. In 1977 he joined the Microwave Research Unit at University College London, where he received the Master in Microwaves and Modern Optics (1978), the Diploma in Microwave Engineering (1978) and the Ph. D. from the University of London (1982).

He is currently the holder of the International Test Conference (ITC) Endowed Chair Professorship at Northeastern University, Boston. At the same Institution during the period 1998-2004 he served as Chair of the Department of Electrical and Computer Engineering. Prior to Northeastern University he was a faculty member at Texas Tech University, the University of Colorado-Boulder and Texas A&M University.

Dr. Lombardi has received many professional awards: the Visiting Fellowship at the British Columbia Advanced System Institute, University of Victoria, Canada (1988), twice the Texas Experimental Engineering Station Research Fellowship (1991-1992, 1997-1998) the Halliburton Professorship (1995), the Outstanding Engineering Research Award at Northeastern University (2004) and an International Research Award from the Ministry of Science and Education of Japan (1993-1999). Dr. Lombardi was the recipient of the 1985/86 Research Initiation Award from the IEEE/Engineering Foundation and a Silver Quill Award from Motorola-Austin (1996).

Since 2000, Dr. Lombardi is an Associate Editor of the IEEE Design and Test Magazine. He also serves as the Chair of the Committee on "Nanotechnology Devices and Systems" of the Test Technology Technical Council of the IEEE (2003 - ). In the past, Dr. Lombardi was an Associate Editor (1996-2000) and the Associate Editor-in-Chief (2000-2006) of IEEE Transactions on Computers and twice a Distinguished Visitor of the IEEE-CS (1990-1993 and 2001-2004). Since January 1, 2007 he is the Editor-In-Chief of the IEEE Transactions on Computers.

Dr. Lombardi has been involved in organizing many international symposia, conferences and workshops sponsored by professional organizations as well as guest editor of Special Issues in archival journals and magazines such as IEEE Transactions on Computers, IEEE Transactions on Instrumentation and Measurement, the IEEE Micro Magazine and the IEEE Design & Test Magazine. He is the Founding General Chair of the IEEE Symposium on Network Computing and Applications.

His research interests are testing and design of digital systems, bio and nano computing, emerging technologies, defect tolerance and CAD VLSI. He has extensively published in these areas and coauthored/edited seven books.

**E-mail:** lombardi@ece.neu.edu

**Paper Handling Data:**

Submitted: 03.09.2008
Received in revised form: 08.23.2009
Accepted: 10.16.2009
Corresponding author: Masoud Hashempour,
Department of Electrical and Computer Engineering,
Northeastern University, Boston, Massachusetts, USA.

[*] This manuscript is an extended version of a paper presented at the International CSI Computer Conference (CSICC'08), March, 2008.