Electronics **2015**, *4*, 150-172; doi:10.3390/electronics4010150

OPEN ACCESS

electronics

ISSN 2079-9292

www.mdpi.com/journal/electronics

Article

Optimally Fortifying Logic Reliability through Criticality Ranking

Yu Bai, Mohammed Alawad, Ronald F. DeMara and Mingjie Lin *

Department of Electrical Engineering and Computer Sciences, University of Central Florida, Orlando, FL 32816, USA; E-Mails: yubai.2003@gmail.com (Y.B.); malawad86@knights.ucf.edu (M.A.); ronald.demara@ucf.edu (R.F.D.)

* Author to whom correspondence should be addressed; E-Mail: mingjie@eecs.ucf.edu; Tel.: +407-882-2298.

Academic Editor: Mostafa Bassiouni

Received: 27 October 2014 / Accepted: 3 February 2015 / Published: 13 February 2015

Abstract: With CMOS technology aggressively scaling towards the 22-nm node, modern FPGA devices face tremendous aging-induced reliability challenges due to bias temperature instability (BTI) and hot carrier injection (HCI). This paper presents a novel anti-aging technique at the logic level that is both scalable and applicable for VLSI digital circuits implemented with FPGA devices. The key idea is to prolong the lifetime of FPGA-mapped designs by strategically elevating the V_{DD} values of some LUTs based on their modular criticality values. Although the idea of scaling V_{DD} in order to improve either energy efficiency or circuit reliability has been explored extensively, our study distinguishes itself by approaching this challenge through an analytical procedure, therefore being able to maximize the overall reliability of the target FPGA design by rigorously modeling the BTI-induced device reliability and optimally solving the V_{DD} assignment problem. Specifically, we first develop a systematic framework to analytically model the reliability of an FPGA LUT (look-up table), which consists of both RAM memory bits and associated switching circuit. We also, for the first time, establish the relationship between signal transition density and a LUT's reliability in an analytical way. This key observation further motivates us to define the modular criticality as the product of signal transition density and the logic observability of each LUT. Finally, we analytically prove, for the first time, that the optimal way to improve the overall reliability of a whole FPGA device is to fortify individual LUTs according to their modular criticality. To the best of our knowledge, this work is the first to draw such a conclusion.

Keywords: criticality analysis; VLSI; logic circuit; discriminative; voltage scaling

1. Introduction

As electronic device technology aggressively scales towards the 22-nm node, especially with the recent introduction of high-k material to avoid the gate tunneling effect, the aging-induced reliability issue will be exacerbated greatly [1,2]. As such, structural degradation in modern Complementary Metal Oxide Semiconductor (CMOS) devices can potentially accelerate, therefore resulting in hard faults at a much faster pace [3]. Because these hard faults cannot be rectified to make an ICchip more reliable to use, it is imperative to develop effective anti-aging techniques at the circuit, logic and architecture levels, especially for the applications that require high field reliability. Such applications include automobiles, aircraft, medical equipments or power plants, whereby the performance degradation and circuit failure can potentially be life-threatening.

Major aging mechanisms of CMOS technology include bias temperature instability (BTI), hot carrier injection (HCI), electro-migration (EM), stress migration (SM) and time-dependent dielectric breakdown (TDDB) [4]. All of these mechanisms are responsible for the gradual oxide wear-out or the interconnect failure that causes circuit performance degradation and transistor failure. Furthermore, all of these mechanisms can be worsened by the high switching rate of a circuit, excess supply voltage or high operational temperature. Among all of these transistor aging mechanisms, the most prominent ones are the negative bias-temperature instability (NBTI), which affects PMOS transistors, and the positive one (PBTI), which affects NMOStransistors [1,2,5,6]. The major effect of the NBTI and PBTI is that they increase the magnitude of the transistor's threshold voltage and reduce the effective carrier mobility over time, therefore leading to a reduction in the operational reliability of the CMOS transistor. Ultimately, such aging mechanisms will shorten the lifetime of CMOS devices. In the past, the effect of PBTI was negligible in comparison to NBTI. However, since the introduction of the high-k/metal gate materials, its effect becomes comparable.

Historically, Field Programmable Gate Array (FPGA) technology has been always at the forefront to exploit the latest advancements in CMOS technology. This is because FPGA devices typically have regular and highly-scalable structures, as well as stringent demands on high performance and energy efficiency. For example, FPGAs that use a 22-nm high-k/metal gate process technology and operate with frequencies up to 1.5 GHz have been announced [7]. Unfortunately, CMOS technology scaling also poses several technical challenges to FPGA device's reliability. Specifically, these issues include manufacturing variability, sub-threshold leakage, power dissipation, increased circuit noise sensitivity and reliability concerns, due to transient (e.g., radiation-induced soft errors) and permanent (e.g., transistor aging) failures [8,9]. In this paper, we present a novel technique at the logic level, specifically designed to mitigate the aging effect of FPGA devices. Our proposed method is both scalable and applicable for Very Large Scale Integration (VLSI) digital circuits implemented with modern FPGA devices.

1.1. Research Objective and Key Contribution

Fundamentally, there are two approaches to mitigate the reliability issues in FPGA devices. The first approach takes a bottom-up strategy, which involves analyzing failure mechanisms at the level of device physics, therefore improving the overall reliability of FPGA devices through transistor engineering or circuit optimization. The second approach attempts to improve FPGA device reliability in a top-down direction, *i.e.*, establishing the relationship between the reliability of individual circuit logic components and the reliability of the whole device. In other words, the second approach formulates the FPGA device reliability problem as a system engineering problem and solves it at the logic and architecture design levels.

This paper focuses on mitigating the negative impacts due to FPGA transistor aging at the logic level. Specifically, we aim at developing a systematic approach for discriminatively scaling V_{DD}s within an FPGA device in order to optimally improve its overall reliability. As will be shown later, our proposed criticality-based approach is totally independent of the specific ways to enhance the reliability of individual FPGA components. Besides elevating the V_{DD}s of LUTs, we can also use device engineering or even modular redundancy. As such, we first develop a systematic framework to analytically model the reliability of an FPGA LUT (look-up table), which consists of both Static Random-Access Memory (SRAM) bits and associated switching circuits. While the majority of all existing work focused on studying the timing degradation due to BTI effects, we concentrate on investigating the BTI-induced switching degradation in FPGA. We also, for the first time, establish the relationship between signal transition density and a LUT's reliability in an analytical way. This key observation further motivates us to define the modular criticality as the product of signal transition density and the logic observability of each LUT. Finally, we analytically prove that the optimal way to improve the overall reliability of a whole FPGA device is to fortify individual LUTs according to their modular criticality. To the best of our knowledge, this work is the first to draw such a conclusion.

The rest of the paper is organized as follows. Section 2 states the existing study results on CMOS technology aging. We then delve into more detailed descriptions of the analysis procedure for FPGA aging due to BTI in Section 4. In Sections 5 and 6, we outline our modeling strategy of FPGA reliability, our proposed strategy to maximize its overall reliability and the optimality proof of our proposed approach, respectively. Subsequently, Section 7 describes the reliability improvement results that we obtained using benchmarks from the Altera benchmark suite of the Quartus University Interface Program (QUIP). In these results, we aim to illustrate both the effectiveness and the computational efficiency of our proposed approach. Afterwards, we present and analyze the usefulness of modular criticality values by applying discriminative logic fortification to several circuits. As we will show, the knowledge of modular criticality values for a given circuit can significantly increase the cost-effectiveness of hardware redundancy. Finally, Section 9 concludes the paper.

2. Modeling BTI-Induced CMOS Device Aging

Several predictive models for BTI have been developed based on reaction-diffusion (R-D) models [10,11]. In particular, several studies analyzed the BTI effect on threshold voltage changes. Traditionally, although BTI can be categorized into two different effects on the

transistor model, the NBTI, which affects PMOS transistors, is far more important than the PBTI, which affects NMOS transistors. However, with the better understanding of high κ/metal gate transistors in sub-45-nm technology, the PBTI effect becomes more important and comparable to NBTI. In this paper, we adopt the most recent results and combine both the NBTI and PBTI effects when estimating BTI's impact on the transistor threshold voltage $V_{\rm th}$. For brevity, we omit the detailed description of the physical mechanism for both the BTI and PBTI. Instead, we refer interested readers to many existing studies [11–13] for further information.

Fundamentally, there are two types of BTI effects: static BTI and dynamic BTI. The static NBTI/PBTI corresponds to the case when the PMOS/NMOS transistor is under constant stress. In this case, $\Delta V_{\rm th}$ due to NBTI/PBTI at time t can be expressed, according to [14], as:

$$\Delta V_{\rm th} = A\left((1+\delta)t_{\rm ox} + \sqrt{C(t-t_0)}\right)^{2n},$$

where n is the time exponent and $n = 1/6$ to $1/4$ depends on the diffusion type used in the physics modeling. A is another constant depending on the hole density, temperature T and the electrical field $E_{\rm ox} = (V_{\rm GS} - V_{\rm th})/t_{\rm ox}$. Specifically,

$$A = \left(\frac{qt_{\rm ox}}{\epsilon_{\rm ox}}\right)\left[K^2 C_{\rm ox}(V_{\rm GS} - V_{\rm th})\left(\exp\left(\frac{E_{\rm ox}}{E_0}\right)\right)\right]^{\frac{2}{3}},$$

where q is the electron charge, k is the Boltzmann constant and $C_{\rm ox}$ is the oxide capacitance per unit area.

Dynamic BTI corresponds to the case where the PMOS/NMOS transistor undergoes alternate stress ($V_{\rm gs} = (-/+)V_{\rm DD}$) and recovery ($V_{\rm gs} = (+/-)V_{\rm DD}$) periods. Fundamentally, both NBTI and PBTI have two phases: (1) the stress phase, at which the gate-source voltage is reversely (positively) biased ($V_{GS} = -(+)V_{DD}$); and (2) the relaxation phase ($V_{GS} = 0$). As shown in Figure 1, at the stress phase, the interface of channel and gate oxide creates some interface traps. The created interface traps make the magnitude of threshold voltage V_{th} increase; on the other hand, some of the interface traps may be removed, and as a result, the V_{th} of the transistor decrease, due to the widely different diffusivity of H_2 in the oxide and poly-Si. The recovery becomes a two-step process, with fast recovery driven by the H_2 in oxide followed by slow recovery of H_2 by back diffusion from poly-Si. Thus, $\Delta V_{\rm th}$ can be separately expressed in stress and recovery periods.

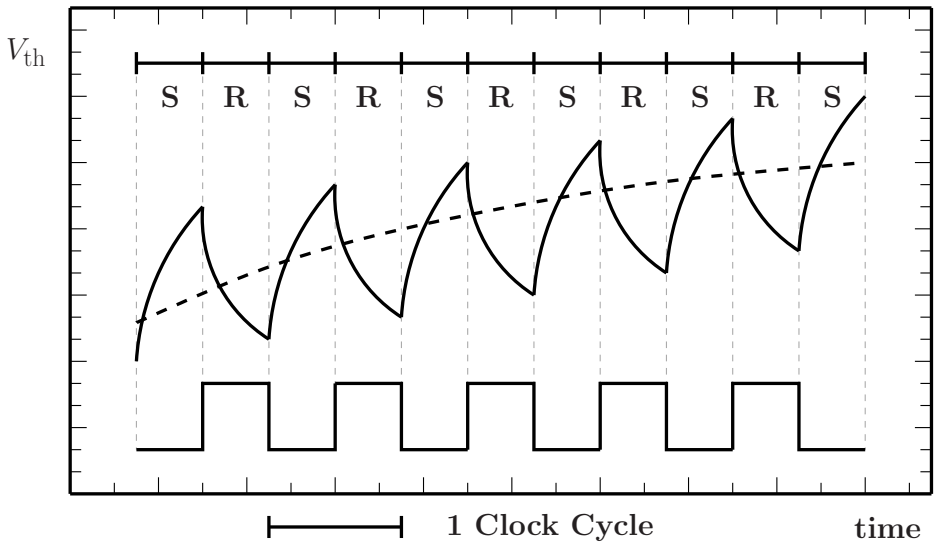
$$\Delta V_{\rm th} = \left[K_{\rm v}(t-t_0)^{\frac{1}{2}} + (\Delta V_{\rm th0})^{\frac{1}{2n}}\right]^{2n},$$

and

$$\Delta V_{\rm th} = V_{\rm th0}\left[1 - \frac{2\xi_1 t_e + \sqrt{\xi_2 C(t-t_0)}}{2t_{\rm ox} + \sqrt{Ct}}\right],$$

where t_e either equals $t_{\rm ox}$ or the diffusion distance of hydrogen in the initial stage of recovery. This parameter captures the fast drop of $V_{\rm th}$ at the beginning of the recovery phase. This effect is verified with estimated measurement data from [11]. This model also accurately captures the dependence of the fractional recovery on $t_{\rm ox}$. Thus, thicker dielectrics have higher fractional recovery.

Figure 1. Illustration of Dynamic BTI. Each clock cycle consists of two phases: (1) Stress (D) and (2) Recovery (R). The dashed line represents the overall aging process, *i.e.*, the increasing trend of V_{th}.



In order to predict the long-term threshold voltage degradation (ΔV_{th}) due to NBTI at a time t, the stress and recovery cycles can be simulated for $m = \frac{t}{T_{\text{clk}}}$ cycles to obtain the long-term degradation. However, for high performance circuits, m can be very large, even for t = 1 month. Thus, it becomes impractical to perform simulation in order to predict ΔV_{th}. However, various recent studies have shown that it is possible to obtain a closed form for the upper bound on the ΔV_{th} as a function of the duty cycle α, T_{clk} and t [15]. In fact, the models of PBTI and NBTI are similar to each other. The BTI effect on V_{th} can be calculated as follows [15],

$$\Delta V_{\text{th}} = AY^n t^n, \tag{1}$$

where A is a function-dependent factor of the temperature, n is a constant depending on the fabrication process ($n = 1/6$ or $n = 1/4$ based on the diffusion type [14]), Y is the duty cycle and t is the total time (transistor age) [15]. In this paper, we define the duty cycle of a transistor as the ratio between the stress time to the total time, which also can be defined by signal probability (SP). To further verify the accuracy of this model, we have compared the results of our modeling and the experimental data collected by [15] for the TSMC45-nm technology node. Both data have shown very good matching.

The R-D based V_{th} model discussed above assumes nominal degradation without considering the statistical variation in the underlying degradation process. In reality, due to the finite number of Si-H bonds in the channel, breaking and re-passivation of these bonds experience stochastic fluctuations [16]. This phenomenon is similar to the random V_{th} variation induced by the number and the placement of dopant atoms in the channel, known as the random dopant fluctuation (RDF) effect. The general framework of BTI variation has been proposed by Stewart in [17], where the number of broken bonds N_{IT} in the channel has been modeled as a Poisson random variable. Under this assumption, N_{IT} satisfies the following:

$$\sigma^2(N_{\text{IT}}) = \mu(N_{\text{IT}}) = \frac{C_{\text{ox}}\mu(\Delta V_{\text{th}})}{q} = \frac{\epsilon_{\text{ox}} A_{\text{G}} \mu(\Delta V_{\text{th}})}{q t_{\text{ox}}},$$

where $\sigma(N_{\mathrm{IT}})$ and $\mu(N_{\mathrm{IT}})$ represent the mean and the standard deviation (SD) of N_{IT}. $\mu(\Delta V_{\mathrm{th}})$ is the nominal (mean) V_{th} degradation due to the BTI. A_{G} is the effective channel area. We can further derive the SD of V_{th} as:

$$\sigma^2(\Delta V_{\mathrm{th}}) = \sigma^2(N_{\mathrm{IT}}) \left(\frac{q}{C_{\mathrm{ox}}}\right)^2 = \frac{q t_{\mathrm{ox}} \mu(\Delta V_{\mathrm{th}})}{\epsilon_{\mathrm{ox}} A_{\mathrm{G}}}. \quad (2)$$

This equation shows that since the nominal V_{th} degradation follows a fractional power law, the $\mu(\Delta V_{\mathrm{th}})$ also maintains a power relationship with respect to time with a fixed exponent of 1/12. Note that unlike the nominal V_{th} degradation, the BTI-induced V_{th} SD depends on the transistor dimension A_{G} with a reverse square relationship.

3. Aging-Induced Error Probability in CMOS

In Section 2, we analyzed the temporal degradation of V_{th} in CMOS transistors due to BTI. In this section, we show that knowing the threshold voltage degradation of a single transistor due to BTI, one can predict the degradation of CMOS transistor switching performance and SRAM read/write performance with a high degree of accuracy.

3.1. BTI-Induced Error Probability in CMOS Switches

As V_{th} increases due to transistor aging, the voltage drive ($V_{\mathrm{DD}} - V_{\mathrm{th}}$) decreases, thus gradually degrading the digital switching behavior of a CMOS transistor. However, quantifying such a negative impact on digital switching is very challenging for two fundamental reasons [18]. First, the switching failure of a given logic circuit is almost always caused by a group of several transistors that gradually experience increases in V_{th}; therefore, it is very difficult to attribute the overall logic failure of a circuit to a single gate. Second, it is very challenging to define a clean cut-off point of the voltage drive ($V_{\mathrm{DD}} - V_{\mathrm{th}}$), beyond which the transistor switching will stop functioning correctly. In fact, for a given logic gate, as long as its output voltage level can be correctly interpreted by its receiving circuit, any input voltage level is theoretically acceptable. To overcome these issues, we use a modeling approach based on the voltage transfer curve (VTC) analysis proposed in [19] and further developed in [18]. In this method, the amount of headroom to a switching failure is measured with the worst-case static noise margin (SNM) present in the gate pair, which can be determined using the DC noise source configuration shown in Figure 2a,b, or equivalently from a butterfly plot shown in Figure 2c. Here, the VTC of the first gate is plotted combined with the inverse VTC of the second one. Positive SNM corresponds to the existence of two areas entirely enclosed by the VTCs, corresponding to the two stable states of the gate pair. Intuitively, the larger the values of W_1 and W_2, the better the switching performance is. As in both [18] and [19], the exact threshold value of W_1 and W_2, as well as their corresponding threshold values of V_{th}^* can be determined empirically. In this paper, we define that the switching failure happens when the VTC asymmetry ratio γ exceeds 0.1, where γ is defined as $\frac{|W_1 - W_2|}{\max\{W_1, W_2\}}$. In many aspects, this quantifying method based on SNM is conceptually very similar to the well-known "eye diagram" used in analog circuit analysis.

Figure 2. (**a**) Using a looped gate chain with feedback configuration to model a gate chain with infinite length in (**b**); (**c**) the voltage transfer curves (VTCs) of the gate pair are used in butterfly plots to determine the static noise margins (SNMs) [18].

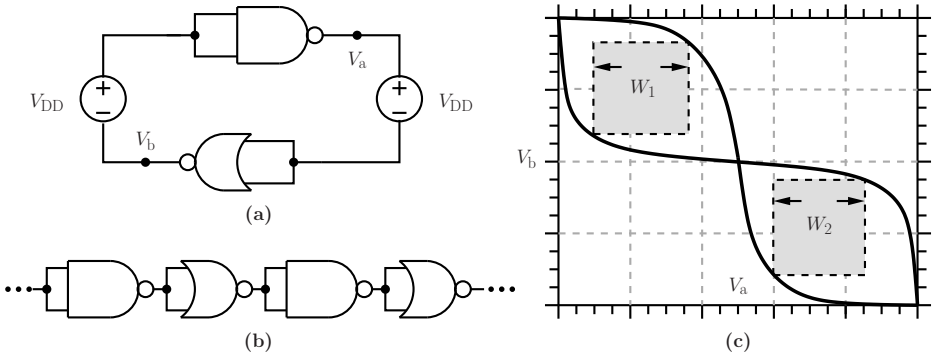
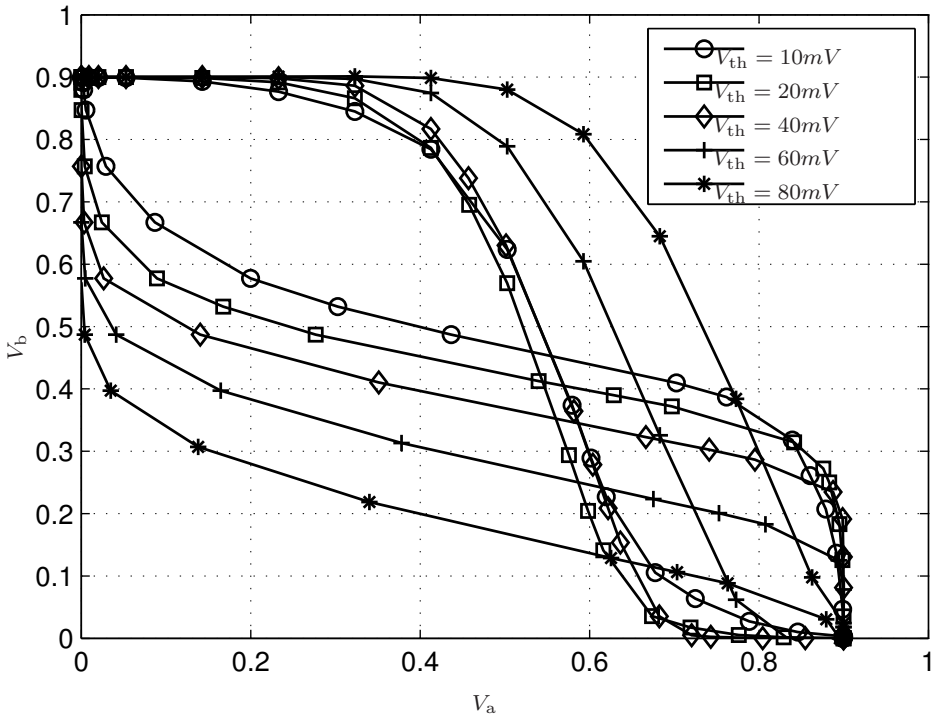


Figure 3. VTC curves of the 22-nm predictive CMOS device model [20] for five different V_{th} values.



To further validate the SNM-based method, we have used the 22-nm predictive CMOS device model [20], and our SPICE simulation results are presented in Figure 3. We define the SNM values to be the side lengths of the largest squares, which can be inscribed into the areas (Figure 2c). For gates with more than one input, multiple possible VTCs exist depending on the input configuration. As suggested in [18], we solve this problem by considering only gate combinations that are expected to be critical, due to their topology. For example, given the common logic gates, such as NAND2 and NOR2, an obvious choice for the critical VTC will be the input combination of a weak low and a weak high values due to the stacked transistors in the corresponding output path. In Figure 3, we have plotted the VTC curves of the 22-nm predictive CMOS device model [20] for five different V_{th} values. It shows that as the V_{th}

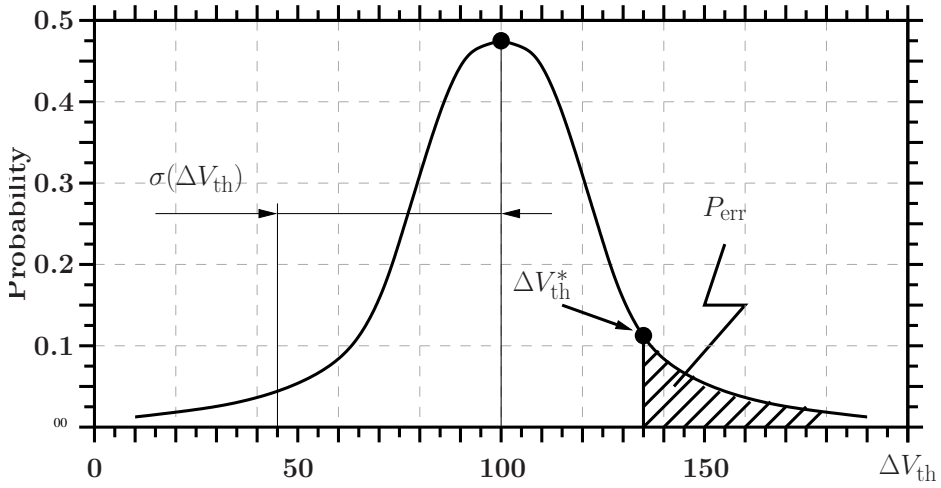
value increases from 20 mV to 100 mV, the VTC asymmetry ratio γ defined above increases from 0.0 to 0.5. Clearly, when $V_{\rm th}$ reaches 50 mV, the VTC asymmetry ratio $\gamma \geq 0.1$, thus signaling the switching failure.

As discussed in Section 2, both analytical modeling and empirical study have shown that $\Delta V_{\rm th}$ follows a Gaussian distribution. Furthermore, its mean $\mu(\Delta V_{\rm th})$ and variance $\sigma(\Delta V_{\rm th})$ can be obtained by Equations (1) and (2). For any given CMOS device, when $\Delta V_{\rm th} > \Delta V_{\rm th}^*$, digital switching fails, where the threshold value $\Delta V_{\rm th}^*$ can be obtained through SPICE simulations. Therefore, the error probability of digital switching for a particular gate can be computed as:

$$P_{\rm err} = \int_{V=V_{\rm th}^*}^{+\infty} f_{V_{\rm th}}(V). \tag{3}$$

The relationship between $P_{\rm err}$ and the normal distribution of $\Delta V_{\rm th}$ can be depicted as in Figure 4.

Figure 4. Probabilistic density function of $\Delta V_{\rm th}$ (mV). $\Delta V_{\rm th}^*$ denotes the cut-off point, beyond which the transistor stops switching correctly. The shaded area represents the total error probability that the transistor malfunctions.



3.2. BTI-Induced Error Probability in SRAM Cells

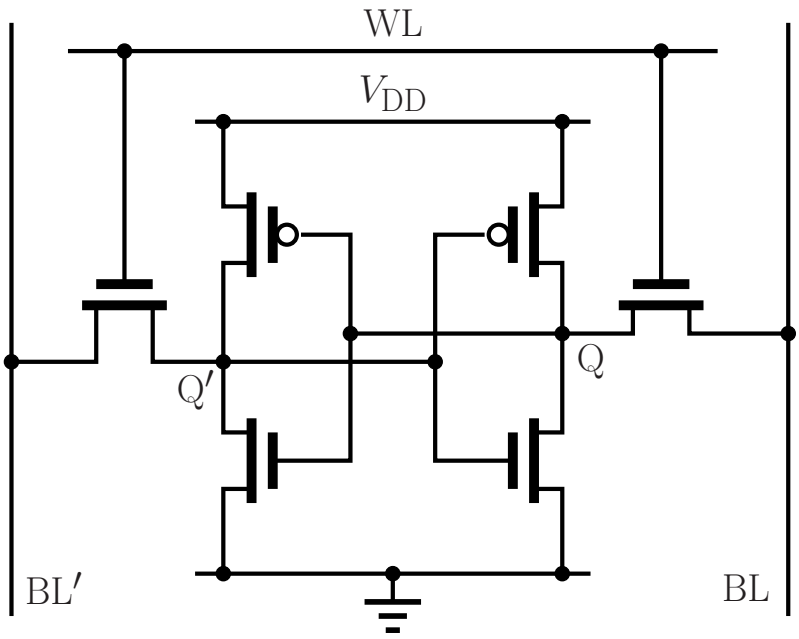
The majority of FPGA devices are SRAM-based, *i.e.*, they store logic cell configuration data in the static memory organized as an array of latches. Figure 5 illustrates a standard logic design for a SRAM cell consisting of six transistors. Unfortunately, In a static random-access memory (SRAM) cell, a mismatch in the strength between the neighboring transistors, caused by BTI-induced aging, can result in the failure of the cell [21], therefore causing the FPGA logic to malfunction. Specifically, there are mainly three causes of SRAM failure.

Read failure: An increase in the cell access time that exceeds the delay requirements can cause SRAM cell read failure. In SRAM cell concepts, the cell access time is defined as time of generating a difference of pre-specified voltage between two bit-lines. The threshold voltage V_{th} of access transistor AX_R and the pull-down NMOS N_R may significantly increase the access time.

Write failure: The inability of writing data into SRAM cells is called write failure. For example, suppose the SRAM cell currently stores the value "1", when writing "0" into this cell; the node $V_{\rm L}$ gets

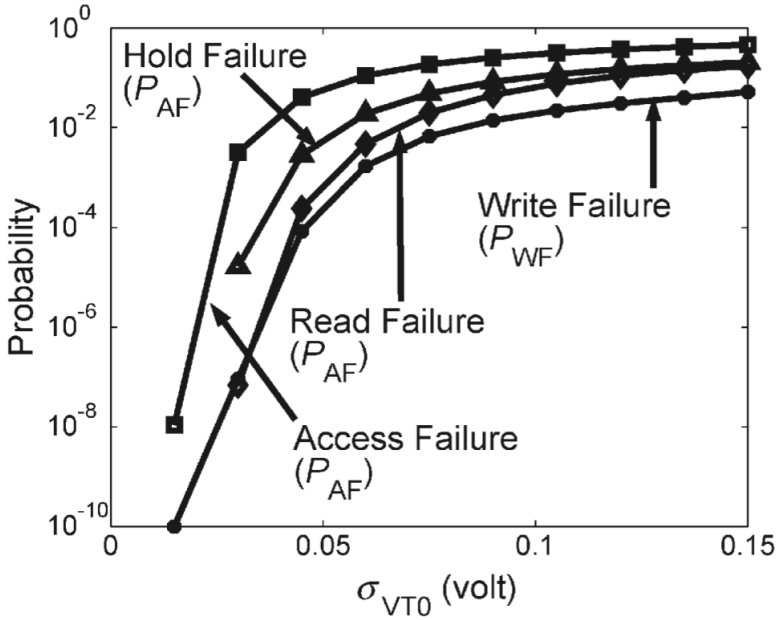
discharged through the bit line BL in Fig. 5 to the low value V_{WR} determined by the voltage division between the PMOS P_{L} and the access transistor AX_{L} [21]. If V_{L} cannot be reduced within time below the trip point of inverter $P_{\mathrm{R}} - N_{\mathrm{R}}(V_{\mathrm{TRIPWR}})$, the write failure occurs.

Figure 5. Transistor network of a standard SRAM cell.



Hold Failure: Hold failure happens when the content of a SRAM cell cannot be preserved due to the application of lower power voltage V_{DD}, which aims at saving leakage power consumption. For example, in Figure 5, if the voltage of node L is lower than the trip point of inverter ($P_{\mathrm{R}} - N_{\mathrm{R}}$), hold failure occurs. Additionally, flipping of the cell data with the application of a supply voltage lower than the nominal one can also cause the failure of data holding in a SRAM cell at the standby mode.

Figure 6. Variation of failure probability with $\sigma(V_{\mathrm{th}})$ [21].

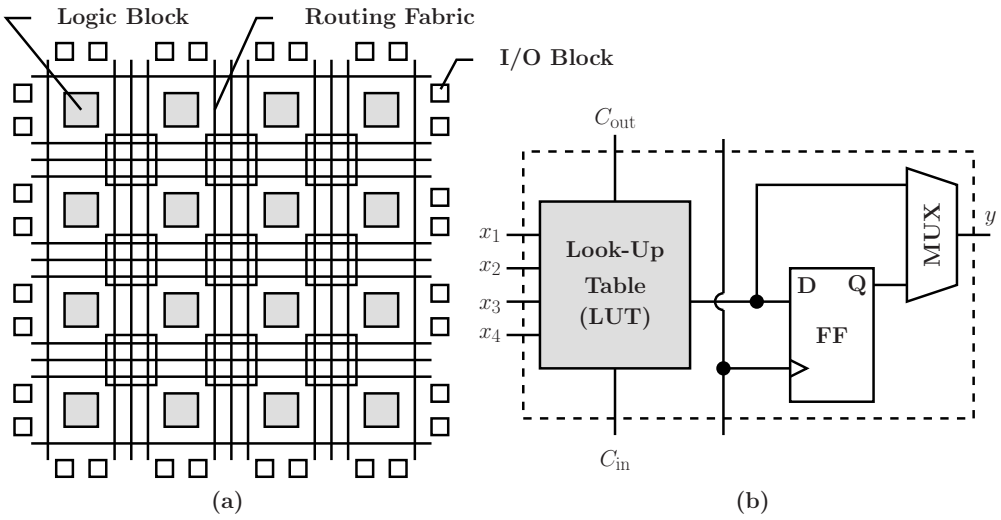


All of these failure modes can be caused by the BTI-induced V_{th} changes in CMOS transistors. In this paper, we adopt the probabilistic SRAM failure model first proposed in [21] in order to analyze and quantify the failure probabilities (access-time failure, read/write failure and hold failure) of synchronous random-access memory (SRAM) cells. Unfortunately, there is no close-form solution for the overall error probability for a given SRAM cell. Instead, we rely on a numerical method to obtain the error probability solutions. In Figure 6, we present such error probability results. Note that it is the $\sigma(V_{\text{th}})$, not the V_{th} itself, that determines the combined error probability of a SRAM cell. Later, in Section 4.1, we will use these results to compute the aggregated error probability of FPGA LUTs due to BTI effects.

4. Modeling FPGA Device Aging

An FPGA is a logic device that contains a two-dimensional array of generic logic elements (LEs) and programmable switches, as shown in Figure 7a. A logic element depicted in Figure 7b can be configured (*i.e.*, programmed) to perform a simple function, and a programmable switch can be customized to provide interconnections among the logic elements. A custom design can be implemented by specifying the function of each logic element, selectively setting the connection of each logic element and selectively setting the connection of each programmable switch. A logic element usually contains a programmable look-up table (LUT), programmable interconnects and flip-flops (FF). An *n*-input look-up table is typically implemented by a static random access memory (SRAM) and is used to implement any n-input combinational function. The flip-flops can be selectively used to implement sequential circuits. Most FPGA devices also embed certain macro cells, such as block RAMs, dedicated multipliers, clock managers and I/O interface circuits. Logic elements are usually grouped into logic array blocks (LABs). Since the LUT is the basic logic element to implement the logic function, in this work, we analytically quantify the aging-induced effect on the transistor for the FPGA reliability issue.

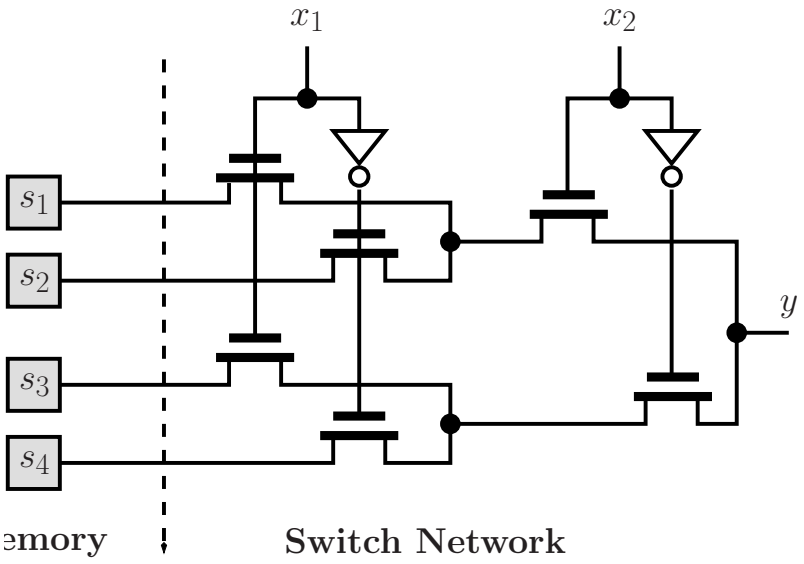
Figure 7. (**a**) Sketch of the FPGA architecture; (**b**) diagram of a simple logic block. FF, flip-flop.



In FPGAs, LUTs are considered the basic blocks for mapping Boolean functions. Modern FPGAs allow modifications to the mapped function of LUTs through reconfiguration, partial or full, online

or offline. The logic structure of a typical FPGA logic block consists of SRAM configuration bits and switching network. Figure 8 depicts a small two-input LUT, whereas modern FPGA devices typically use six- or eight-input LUTs. In the following, we derive the error probability of a LUT analytically based on the error probability results of the SRAM cell and switching transistors developed in Sections 3.1 and 3.2. Ideally, we should also incorporate flip-flops into our analytical framework. There are two reasons why we did not do that. First, this paper mainly focuses on the logic correctness of a placed and routed circuit implemented with an FPGA device. Flip-flops are clocked circuits whose outputs may change on an active edge of the clock signal based on its input. Flip-flops normally would not change the output upon input change, even when the clock signal is asserted. Therefore, the logic correctness of an FF mostly depends on the timing violations due to device aging, which can be more effectively addressed by reducing the clock rate or allowing more generous timing margins at the design stage. Secondly, for all combinational circuits, FF does not exist. Even for the sequential circuits, the number of FFs is much smaller than the number of switches in a modern FPGA. Finally, although we did not include FFs in our theoretical analysis, we include every gate in our experiments of extracting modular criticality through simulations.

Figure 8. Logic diagram of a two-input LUT.



4.1. Aging-Induced Effect on LUT-SRAM

As discussed in Section 3, the device aging effect can cause read, write and hold failure in SRAM cells. In fact, the error probability of a SRAM cell is determined by $\sigma(V_{\text{th}})$, which can be described as a function of device duration t, technology node G_{device} and signal probability α. Therefore, the error probability of a SRAM cell $e_{\text{SRAM}} = f(t, G_{\text{device}}, \alpha)$.

After configuring an FPGA device, the SRAM cells in each of the activated LUTs store different logic values, "0" and "1", that determine the functionality of each LUT, hence the overall functionality of the complete implemented logic design. During the operation of an FPGA device, for any given LUT used, different combinations of input signals will switch on different transistor paths. The switched-on path will establish the connection to a specific SRAM cell, whose stored logic bit becomes the output.

Assume that the LUT has N inputs; the total number of bits in the N-LUT will be 2^N. Furthermore, assume the access probability and the error probability of each SRAM cell to be P_i and e_i, respectively. Because the error probability of a memory cell also depends on its content [21], we denote the error probability of a SRAM cell that stores "0" and "1" as $e_{i,0}$ and $e_{i,1}$, respectively. Finally, we suppose that the error probability of each memory cell is totally independent. Therefore, the error probability of a 2^N configuration memory block in a N-LUT can be written as $P_{\text{SRAM}} = \sum_{i=1}^{2^N} P_i \cdot e_i = \sum_{i \in S_0} P_i \cdot e_{i,0} + \sum_{i \in S_1} P_i \cdot e_{i,1}$. Now, we assume that all SRAM cells are designed and manufactured with the same error characteristics; therefore, $\forall i$, $e_{i,0} = e_0$ and $e_{i,1} = e_1$. As a result, $P_{\text{SRAM}} = \left(\sum_{i \in S_0} P_i\right) e_0 + \left(\sum_{i \in S_1} P_i\right) e_1$, where S_0 and S_1 denote the set of all memory cells that store "0" and "1", respectively. Because the signal probability α equals $\sum_{i \in S_1} P_i$, the final result:

$$P_{\text{SRAM}} = (1 - \alpha)e_0 + \alpha e_1. \tag{4}$$

As discussed in the previous section, the "1" cell is much more critical than the "0" cell, and the increasing of the signal probability of the LUT output for the "1" cell error probability will definitely increase the total probability of the SRAM.

4.2. Modeling Error Probability of Switching Network

As illustrated in Figure 8, the LUT of modern FPGA devices typically uses an NMOS pass transistor as the switching elements. Different input signal combinations can turn on some of these switching transistors and route the bit content of one of LUT memory cell to the output y. Therefore, the error probability for the switching network can be written as:

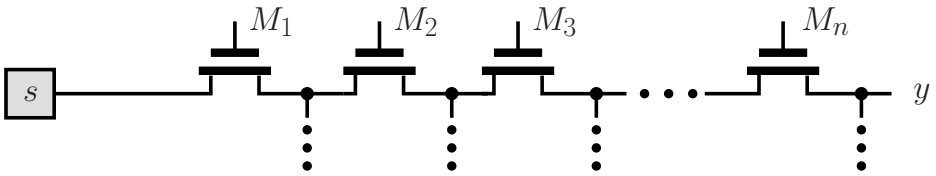
$$P_{\text{SN}} = \sum_{i=1}^{2^N} P_i \cdot e_{\text{path},i}, \tag{5}$$

where P_i and $e_{\text{path},i}$ denote the probability of taking switching path i and the probability that the path i malfunctions.

A faulty switching path can be caused by the switching degradation of the NMOS transistor, as discussed in Section 3.1. Fundamentally, the long stress time on the NMOS transistor may lead to increasing of threshold voltage V_{th}, which, in turn, results in degraded switching strength. Figure 9 illustrates a switching path consisting of N NMOS transistor switches. Assuming the aging-induced error probability of an individual NMOS transistor M_i to be e_{M_i}, the probability for the switching path to be faulty can be written as $e_{\text{path}} = 1 - \prod_{i=1}^{N}(1 - e_{M_i}) = 1 - (1 - e_{\text{M}})^N$, when $\forall i$, $e_{M_i} = e_{\text{M}}$. Substituting this into Equation (5) results in:

$$P_{\text{SN}} = \sum_{i=1}^{2^N} P_i \cdot e_{\text{M}} = 1 - (1 - e_{\text{M}})^N. \tag{6}$$

Figure 9. One switched path containing N NMOSswitches.



4.3. Analytically Modeling Error Probability in LUT

Each LUT in an FPGA device consists of two parts: the SRAM cell array and the multiplexer switching network. Assuming that these two components malfunction independently, the error probability of a LUT can be formalized as follows:

$$\begin{aligned} P_{\text{err,LUT}} &= 1-(1-P_{\text{SRAM}})\cdot(1-P_{\text{SN}}) \\ &= 1-(1-((1-\alpha)e_0+\alpha e_1))(1-P_{\text{SN}}) \text{ (Substitute with Equation (4))} \\ &= 1-(1-e_0-\alpha(e_1-e_0))(1-(1-e_{\text{M}})^N) \text{ (Substitute with Equation (6))} \\ &= 1-(1-e_{\text{M}})^N(1-e_0)+\alpha(e_1-e_0)(1-e_{\text{M}})^N, \end{aligned} \tag{7}$$

where e_1, e_0 and e_{M} denote the error probability of a SRAM cell that stores "1" and "0" and the aging-induced error probability of an individual NMOS transistor (M), respectively. Furthermore,

$$\frac{dP_{\text{err,LUT}}}{d\alpha} = (1-e_{\text{M}})^N\cdot(e_1-e_0) \tag{8}$$

Because $e_1 > e_0$, as discussed in Section 4.1, obviously $\frac{dP_{\text{err,LUT}}}{d\alpha} > 0$, which means that the overall error probability of a LUT increases monotonically with the increase of the output signal probability α. It will become clear that this result is critical in our optimal solution of improving the overall reliability of a placed and routed logic design with an FPGA device.

5. Analyzing FPGA Device Reliability

In this section, an intuitive approach to reliability analysis is described. It is based on the observation that a failure at a gate close to the primary output has a greater probability of propagating to the primary output than a gate several levels of logic away from the primary outputs. This is because a failure that has to propagate through several levels of logic has a higher probability of being logically masked. This can be quantified by applying the concept of observability, which has historically found use in the testing and logic synthesis domains.

In reliability analysis, the logic observability of any logic node can be defined as the probability that a logic value upset error ($0 \to 1$ or $1 \to 0$) at the logic node under consideration will change the circuit outputs. As stated in [22], logic observability can be computed with Boolean differences, symbolic techniques based on binary decision diagrams (BDDs) or simulation. In this study, we will attempt to derive a closed-form expression for the logic reliability, $P_{\text{correct}}(\mathbf{e})$, where $\mathbf{e}$ denotes the error probability of each logic gate.

To the best of our knowledge, there has not been any systematic study on accurately measuring modular criticality values within a large-scale VLSI digital circuit. The most related works to this paper are several recent studies that explored various analytical ways of computing the overall logic reliability of VLSI logic circuits [23–26]. Reliability analysis of logic circuits refers to the problem of evaluating the effects of errors due to noise at individual transistors, gates or logic blocks on the outputs of the circuit. The models for noise range from the highly specific decomposition of the sources, e.g., single-event upsets, to highly abstract models that combine the effects of different failure mechanisms [27,28]. For example, in [22], the authors developed an observability-based approach that can compute a closed-form expression for circuit reliability as a function of the failure probabilities and observability of the gates. Unfortunately, all of these analytical studies, although mathematically concise, have to make some key assumptions, therefore seriously limiting their applicability and accuracy. For example, the method in [22] needs to approximate the multi-gate correlations in order to handle reconvergent fan-out. In addition, it is not clear how the existing analytical approaches can handle some unspecified probabilistic input vector distributions or more complicated correlation patterns within a VLSI logic circuit.

6. Optimally Improving Reliability via Discriminative V_{DD} Scaling

In a typical FPGA CAD flow, after logic synthesis and technology mapping, any given logic circuit will be converted into a network of LUTs ($\mathcal{G}$). Without loss of generality, we assume that the circuit under consideration consists of N LUTs, and each LUT has k inputs and one output. Furthermore, we assume that $\mathcal{G}$ has M signal nets, each of which connects the output port of exactly one LUT to the input ports of a number of LUTs. Furthermore, we define the signal probability and error observability of signal net i as α_i and β_i, respectively. Finally, in this study, we define the product of α_i and β_i as the logic criticality γ_i of LUT i.

$\mathcal{G}$'s output reliability $R(\mathcal{G}, \{e_i\}_{i=1}^{N})$ is its probability of being correct in all its output ports when a large ensemble of identically and independently distributed (i.i.d.) random inputs are applied. Here, $\{e_i\}_{i=1}^{N}$ denotes the vector of error probability of all N gates.

Intuitively, the larger the γ_i is, the more critical the LUT i is to the correctness of the whole circuit $\mathcal{G}$. Note that the input vector distribution need not to be uniform i.i.d. Instead, it can be any general form. In other words, the larger the logic criticality γ_i is, the more sensitive the overall output reliability is towards LUT i's error.

The intuitive explanation of our definition of $\gamma_i = \alpha_i \times \beta_i$ is straightforward. First, for any LUT i, α_i represents the frequency of its output switching, which is directly related to the transistor aging and shows how likely a switching error will occur. Second, β_i shows how sensitive the final output of $\mathcal{G}$ will be to the output error of LUT i. Essentially, γ_i reflects the combined effect of both α_i and β_i towards $\mathcal{G}$'s overall correctness. In the following, we will show that our definition is not only intuitive, but also optimal in the sense that, using the ranking of logic criticality γ_i as the guidance, we can optimally maximize the overall reliability improvement of $\mathcal{G}$ given a fixed amount of extra resources, such as additional chip area or extra power budget.

Thus, observability-based reliability analysis makes two simplifying assumptions for estimating the effect of multiple gate failures.

1. The effect of LUT failures at the primary output is decoupled from each other, *i.e.*, a failure at each LUT i is assumed to affect the output with a probability β_i regardless of other LUT failures. This assumption allows the joint observability to be replaced by simultaneous observability, which is computationally less demanding, to compute the effect of multiple gate failures at the output.
2. The observability of the LUTs are assumed to be independent of each other. Using this assumption, the computation of the simultaneous observability of two LUTs can be simplified to the product of the individual LUT observabilities. For instance, the probability that LUT 1 is observable and LUT 2 is not observable is given by $\beta_1(1-\beta_2)$, and the probability that LUT 1 and LUT 2 are both not observable is given by $(1-\beta_1)(1-\beta_2)$.

With this background, we shall derive the expression for the probability of error at the output for a general circuit network $\mathcal{G}$ with N LUTs. Without loss of generality, we assume that the circuit has a single output y. Denote the error probability and logic observability of the ith LUT by e_i and β_i, respectively. Using the first assumption, the output y will be in error when an odd number of faulty LUTs in $\mathcal{G}$ are simultaneously observable. Using the second assumption, the simultaneous observability of a set of LUTs can be computed by simply multiplying the individual observabilities of the LUTs.

In general, the probability that only the LUTs in F are observable is given by $A = \prod_{i\notin F}(1-\beta_i)\prod_{i\in F}\beta_i$. The expression $B = \prod_{i\notin F}(1-\beta_i)\prod_{i\in F} -\beta_i$ has the same magnitude as A and the same sign as A when F has an even number of LUTs and the opposite sign as A when F has an odd number of LUTs. Thus, when F has an odd number of LUTs, the expression $1/2(A-B)$ gives the probability that the LUTs in F are observable, and when F has an even number of LUTs, $1/2(A-B)$ is equal to zero. Thus, the probability that an odd number of LUTs in $\mathcal{G}$ is observable is given by:

$$\sum_{F\in 2^{\mathcal{G}}} \frac{1}{2}\left(\prod_{i\notin F}(1-\beta_i)\prod_{j\in F}\beta_j - \prod_{i\notin F}(1-\beta_i)\prod_{j\in F}-\beta_j\right).$$

By the first simplifying assumption, the probability of error at the output y given that the LUTs in $\mathcal{G}$ have failed is also given. Thus, $\Pr(y_{\text{err}})|\mathcal{G} = \frac{1}{2}\left(\sum_{F\in 2^{\mathcal{G}}}\prod_{i\notin F}(1-\beta_i)\prod_{j\in F}\beta_j - \sum_{F\in 2^{\mathcal{G}}}\prod_{i\notin F}(1-\beta_i)\prod_{j\in F}-\beta_j\right) = \frac{1}{2}\left(\prod_{j\in\mathcal{G}}(\beta_j+(1-\beta_j)) - \prod_{j\in\mathcal{G}}((1-\beta_j)-\beta_j)\right) = \frac{1}{2}\left(1-\prod_{j\in\mathcal{G}}(1-2\beta_j)\right)$.

The probability that the LUTs in $\mathcal{G}$ are in error and the LUTs in $\mathcal{G}^c$ are error-free is given by $\prod_{i\in\mathcal{G}} e_i \prod_{j\in\mathcal{G}^c}(1-e_j)$. Thus, the probability of error at the output y is given by $\Pr(y_{\text{err}}) = \left(\sum_{G\in S}\prod_{i\in\mathcal{G}} e_i\prod_{j\in\mathcal{G}^c}(1-e_j)\right)\frac{1}{2}\left(1-\prod_{j\in\mathcal{G}}(1-2\beta_j)\right)$. Finally,

$$\Pr(y_{\text{err}}) = \frac{1}{2}\left(1-\prod_{i\in\mathcal{G}}(1-2e_i\beta_i)\right). \tag{9}$$

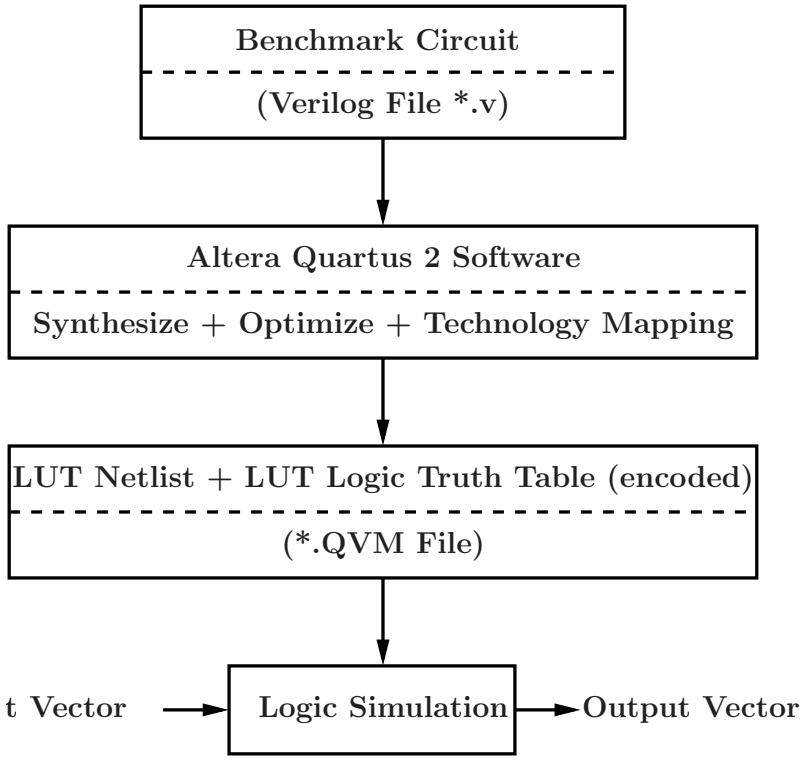
This result clearly shows that, in order to minimize the overall error probability $\Pr(y_{\text{err}})$, we should always choose the largest $e_i\beta_i$ terms to remove. Therefore, given N LUTs in an FPGA design, if only K of them can be fortified, in order to maximize the overall design reliability, we should always choose the K LUTs with the largest criticality values γ_i, where $\gamma_i = \alpha_i \times \beta_i$.

7. Results and Analysis

To validate our error probability model and our discriminative assignment strategy, we have chosen 10 circuits from the Altera benchmark suite of the Quartus University Interface Program (QUIP). The overall procedure of our experiments is depicted in Figure 10. All of our test circuits are in the form of Verilog source files. We rely on the commercial Altera Quartus 2 software to perform all FPGA logic synthesis, logic optimization and technology mapping. Finally, the resulting .QVM files from the Quartus contain both the LUT netlist and the encoded logic truth table for each LUT. We then use our in-house logic simulation tools to read in the .QVM file, and perform logic simulation. As in many other studies, we use extensive Monte Carlo logic simulation to obtain the error probability of any given circuit design. For each benchmark circuit, we cover all possible input combinations. For each input combination, we run many simulation iterations in order to obtain accurate output reliability. Of course, the number of simulation iterations for any given input vector will highly depend on the specific topology and complexity of the targeted logic circuit. We continue logic simulations until the out error probability saturates. Our results have shown that typically 5,000 logic simulations for each input vector are often sufficient. As for logic observability, we use a similar approach, the only difference being that we only invert the logic value at the logic node under consideration, while keeping all logic values at all other nodes unchanged. The observability will be measured by counting the probability for any output to change its value. Obviously, these measurement results also take the dependency of logic observability on internal logic values into considerations. To deal with the intensive computations required for the above logic simulations, we employ the STOKEScomputing cluster at UCF (University of Central Florida), which consists of 3,450 compute cores (Intel Xeon 64-bit processors) and over 7.5 TB of RAM. The total simulation took about one week to complete.

We use the 45-nm predictive technology model (PTM) to model all CMOS devices (http://ptm.asu.edu). At the nominal $V_{\mathrm{DD}} = 1V$ and $V_{\mathrm{th}} = 0.18V$, we assume the error probability of all transistors to be zero. We then set the on-time to be $C = 3$ years and obtain the duty cycle values Y from our logic simulations. Next, using Equation (1), we compute the ΔV_{th}, which can then be used to obtain $\sigma^2(\Delta V_{\mathrm{th}})$ using Equation (2). Using Equation (3), we then obtain the error probability of a single transistor P_{err}. Finally, we can calculate the error probability of any single LUT by the method discussed in Section 4. Note that the above methodology of computing error probability caused by device aging is only applicable to pass-transistor switches. Because the SRAM elements store constant values, we use a different approach to evaluate the aging effect on the error probability. Specifically, as discussed in Section 4.1, after obtaining $\sigma^2(\Delta V_{\mathrm{th}})$ using Equation (2), we can utilize the empirically-measured data, as shown in Figure 6, to read out three main components of the error probability of a SRAM memory cell [21], which can be readily combined to obtain the total memory error probability due to device aging. Our results have shown that for the 45-nm CMOS technology, after three years of switch-on time, the ΔV_{th} is 0.063 V, which induces about 1.34×10^{-4} in LUT error probability. This error probability can be completely eliminated by elevating the V_{DD} to be about 1.1 V.

Figure 10. CAD flow of our circuit design experiments.



All results in Table 1 have been obtained under the above assumptions. For each of these ten benchmarks, we conduct four sets of experiments denoted by U, A, B and C. Type U experiments serve as the baseline when no circuit fortification is done. In Type A experiments, we use the optimal fortification strategy that we developed in Section 6, *i.e.*, we chose K LUTs with the largest criticality values to fortify. In Type B experiments, we randomly pick K LUTs to fortify, while in Type C experiments, we do the opposite to our optimal fortification strategy: we chose K LUTs with the smallest criticality values to fortify. Finally, we have tried three different K values, which are 10%, 20% and 30% of N.

Table 1. Results of the overall error probability P_{err} for all 10 Quartus University Interface Program (QUIP) benchmarks.

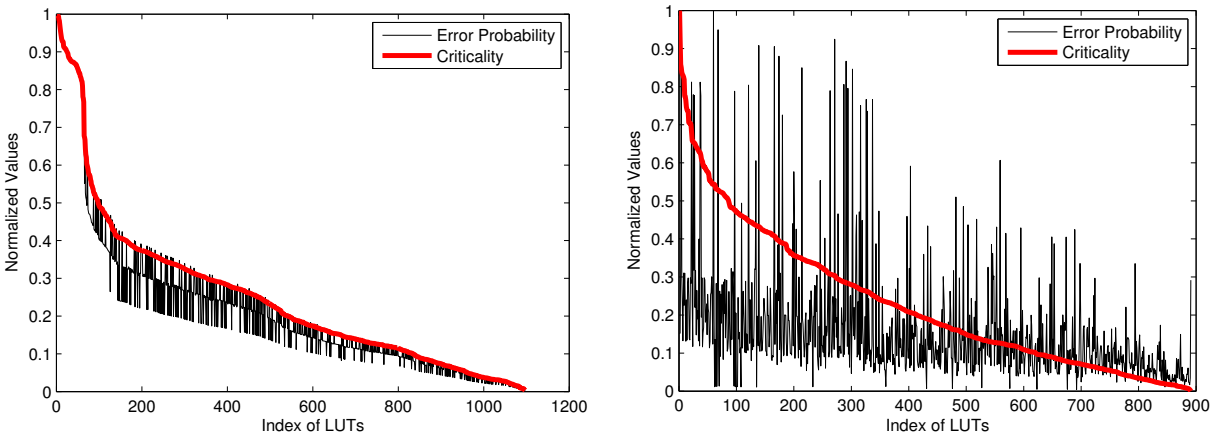
	LE#	PIN#	U	top 10%			top 20%			top 30%		
				A	B	C	A	B	C	A	B	C
OC_SRAM	243	153	0.0387	0.01046	0.0265	0.02636	0.00331	0.00887	0.008812	0.00331	0.00887	0.008812
B04	189	21	0.03698	0.01581	0.0276	0.0281	0.00404	0.00843	0.008139	0.0016	0.002811	0.002711
Barrel16	135	38	0.0314	0.01314	0.0192	0.0198	0.00361	0.00714	0.007108	0.00105	0.00295	0.003071
Barrel64	897	136	0.1765	0.03023	0.1274	0.1211	0.00841	0.0231	0.02341	0.000817	0.00714	0.007131
EX1010	871	20	0.145	0.0912	0.1287	0.1233	0.045	0.0854	0.0844	0.01226	0.0368	0.03067
FLIP_RISK8	1490	113	0.282	0.0876	0.235	0.2345	0.0484	0.085	0.082	0.0106	0.021	0.024
MUX8_64	835	76	0.161	0.054	0.112	0.116	0.0114	0.0877	0.0881	0.006325	0.00969	0.00971
OC_AES_CORE	5005	388	0.312	0.082	0.278	0.277	0.01087	0.051	0.0526	0.0081	0.0314	0.0322
OC_FPU	6972	110	0.351	0.0754	0.2588	0.257	0.012745	0.0621	0.0652	0.006124	0.00881	0.00892
OC_DES3PERF	20188	298	0.413	0.0959	0.381	0.3815	0.010745	0.07821	0.07813	0.00196	0.009181	0.00912

As shown in Table 1, for all benchmark circuits, our optimal discriminative voltage scaling method has significantly improved its overall logic circuit reliability. The improvement ranges from approximately

three-times to five-times. Not surprisingly, the opposite voltage scaling (Type C) has performed poorly with reliability improvements ranging from merely 10% to 30% for $K = 10\%N$. Also intuitively true, when K values increases from 10% to 30%, for any benchmark circuit and any voltage scaling method, the improvement in overall circuit reliability steadily increases. Somewhat surprisingly, when comparing Type B with Type C experiments, very few differences can be found. This essentially shows that, without utilizing the LUT criticality values as the guidance for discriminative voltage scaling, the reliability improvement is almost as poor as the worst scenario. This finding clearly shows the significant advantage of our proposed discriminative voltage scaling scheme based on the LUT criticality ranking.

When examining the results in Table 1 more carefully, one can find that the effectiveness of our discriminative voltage scaling method varies widely. For example, after fortification, the reliability of `FLIP_RISKS` has been improved by almost 3.22-times, while the reliability of `EX1010` has only been improved by 1.58-times, although both circuits are of almost the same size. To better understand this phenomenon, in Figure 11, we have plotted the value profile of LUT criticality and LUT error probability values for both circuits. In each circuit, we first sort all of the LUTs according to the decreasing order of criticality. We then plot the LUT error probability values according to this sorted order. Comparing Figure 11a,b, one can easily observe that for the circuit `FLIP_RISKS`, the sorting order of LUT criticality and error probability match quite closely. In contrast, for `Ex1010`, these two orderings differ greatly. In other words, for `FLIP_RISKS`, the most critical LUT often is the one with the highest error probability, while for `Ex1010`, the opposite is true. Therefore, in the case of `Ex1010`, we may have fortified many LUTs with very low error probability, hence the relatively low effectiveness of our discriminative voltage scaling.

Figure 11. Profile comparison between LUT criticality and LUT error probability values. (a) Results of circuit `FLIP_RISKS`. (b) Results of circuit `Ex1010`.



8. Related Work

Criticality analysis has been extensively studied in software [29], but is quite rare in error-resilient computing device research. Only recently, the general area of criticality analysis (CA), which provides relative measures of significance for the effects of individual components on the overall correctness of system operation, has been investigated in digital circuit design. For example, in [30], a novel approach

to optimize digital integrated circuit yield with regards to speed and area/power for aggressive scaling technologies is presented. The technique is intended to reduce the effects of intra-die variations using redundancy applied only on critical parts of the circuit. In [31], the researchers have explored the idea of discriminatively fortifying a large H.264 circuit design with FPGA fabric. They recognize that: (1) different system components contribute differently to the overall correctness of a target application and therefore should be treated distinctively; and (2) abundant error resilience exists inherently in many practical algorithms, such as signal processing, visual perception and artificial learning. Such error resilience can be significantly improved with effective hardware support. However, in [31], the authors used Monte Carlo-based fault injection, and therefore, the resulting algorithm cannot be efficiently applied to large-scale circuits. Furthermore, their definition of modular criticality was quite *ad hoc*, therefore lacking analytical justification.

More relevant to our study, [32] introduced a logic-level soft error mitigation methodology for combinational circuits. Their key idea is to exploit the existence of logic implications in a design and to selectively add pertinent functionally redundant wires to the circuit. They have demonstrated that the addition of functionally redundant wires reduces the probability that a single-event transient (SET) error will reach a primary output and, by extension, the soft error rate (SER) of the circuit. Obviously, the proposed circuit techniques can be readily applied using our proposed criticality estimation method, especially in a large-scale circuit case. However, more importantly, the method used in [32] to determine circuit criticality is mostly done by assessing the SET sensitization probability reduction achieved by candidate functionally-redundant wires and selects an appropriate subset that, when added to the design, minimizes its SER. Consequently, their overall method of criticality analysis is rather heuristic and utilizes largely "local" information. In addition, it is not very clear how this method can scale with very large-scale circuits.

Samudrala *et al.* [33] also targeted hardening combinational circuits, but focused on mapping digital designs onto Xilinx Virtex FPGAs against single-event upsets (SEUs). They do not perform detailed criticality analysis. Instead, their method uses the signal probabilities of the lines to detect SEU-sensitive sub-circuits of a given combinational circuit. Afterwards, the circuit components deemed to be sensitive are hardened against SEUs by selectively applying triple modular redundancy (STMR) to these sensitive sub-circuits. More recently, in [34], a new methodology to insert selective TMR automatically for SEU mitigation has been presented. Again, the criticality was determined based on empirical data. Because the overall method is cast as a multi-variable optimization problem, it is not clear how this method can scale with circuit size, and few insights will be provided as to which part of the circuit is more critical than others, and by how much.

Finally, another related study [6] also studied the transistor aging mostly due to NBTI and PBTI for FPGA technology. However, they only investigated the effect of transistor aging, due to NBTI and PBTI, in LUTs, by considering different implementations through detailed SPICE simulations. In contrast, our study involves both analytical and empirical studies. More importantly, we study how to improve the overall logic reliability for logic circuits implemented with an FPGA device, without modifying any logic structure in FPGA circuit implementations. Our main approach is to strategically elevate V_{DD} at various critical components to maximize its reliability benefits.

9. Conclusions

There are two fundamental contributions in this work. First, to the best of our knowledge, this study is the first one to reveal the analytical relationship between the BTI-induced device aging and its device reliability through a probabilistic argument. Building upon this finding, we were able to derive analytical models to model the circuit reliability of LUTs in an FPGA device. Second, for the first time, we show that, given a fixed amount of extra resources, the optimal way to allocate them, so that the overall reliability of circuit design can be maximized, is to use the criticality to prioritize the resource allocation. This solution is quite general in its applicability. Moreover, the extra resource considered can take many forms. In this work, we chose to use elevated V_{DD}, but this can also be replaced with hardware redundancy, transistor device engineering or transistor sizing.

Acknowledgments

This work was supported in part by an ARODURIPgrant, N16-22-6056, an NSF BRIGEgrant, ECCS-1342225, and an NSF CCF grant, 1319884.

Author Contributions

Both Y. Bai and M. Lin developed all analytical methods presented in the paper; M. Lin, Y. Bai, and R.F. DeMara wrote the paper; Illustrations were generated by Y. Bai, M. Alawad, and M. Lin; Simulated and real experimental data was recorded by Y. Bai and M. Alawad; Analysis of experiments was performed by Y. Bai, M. Lin, M. Alawad and R.F. DeMara.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. Pae, S.; Maiz, J.; Prasad, C.; Woolery, B. Effect of BTI Degradation on Transistor Variability in Advanced Semiconductor Technologies. *IEEE Trans. Device Mater. Reliab.* **2008**, *8*, 519–525.
2. Kumar, S.; Kim, C.; Sapatnekar, S. Adaptive Techniques for Overcoming Performance Degradation Due to Aging in CMOS Circuits. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2011**, *19*, 603–614.
3. Oates, A. Reliability challenges for the continued scaling of IC technologies. In Proceedings of the 2012 IEEE Custom Integrated Circuits Conference (CICC), San Jose, CA, USA, 9–12 September 2012; pp. 1–4.
4. Ghosh, S.; Roy, K. Parameter Variation Tolerance and Error Resiliency: New Design Paradigm for the Nanoscale Era. *Proc. IEEE* **2010**, *98*, 1718–1751.

5. Amouri, A.; Kiamehr, S.; Tahoori, M. Investigation of aging effects in different implementations and structures of programmable routing resources of FPGAs. In Proceedings of the 2012 International Conference on Field-Programmable Technology (FPT), Seoul, Korea, 10–12 December 2012; pp. 215–219.9
6. Kiamehr, S.; Amouri, A.; Tahoori, M. Investigation of NBTI and PBTI induced aging in different LUT implementations. In Proceedings of the 2011 International Conference on Field-Programmable Technology (FPT), New Delhi, India, 12–14 December 2011; pp. 1–8.
7. Naito, T.; Ishida, T.; Onoduka1, T.; Nishigoori, M.; Nakayama, T.; Ueno, Y.; Ishimoto, Y.; Suzuki, A.; Chung, W.; Madurawe, R.; *et al.* World's first monolithic 3D-FPGA with TFT SRAM over 90 nm 9 layer Cu CMOS. *I*n Proceedings of the 2010 Symposium on VLSI Technology, Digest of Technical Papers, Honolulu, HI, USA, 15–17 June 2010.
8. Srinivasan, S.; Mangalagiri, P.; Xie, Y.; Vijaykrishnan, N.; Sarpatwari, K. FLAW: FPGA lifetime awareness. In Proceedings of the 43rd Annual Design Automation Conference (DAC), San Francisco, CA, USA; ACM: New York, NY, USA, 2006; pp. 630–635.
9. Xilinx 7 Series FPGAs Overview. Xilinx Specifications; 2012.
10. Stott, E.; Sedcole, P.; Cheung, P.Y.K. Fault tolerant methods for reliability in FPGAs. In Proceedings of the International Conference on Field Programmable Logic and Applications, FPL 2008, Heidelberg, Germany, 8–10 September 2008; pp. 415–420.
11. Krishnan, A.; Chancellor, C.; Chakravarthi, S.; Nicollian, P.; Reddy, V.; Varghese, A.; Khamankar, R.; Krishnan, S. Material dependence of hydrogen diffusion: implications for NBTI degradation. In Proceedings of the IEEE International Electron Devices Meeting, IEDM Technical Digest, 5 December 2005, Washington, DC, USA; pp. 4–691.
12. Maricau, E.; Gielen, G. Efficient Variability-Aware NBTI and Hot Carrier Circuit Reliability Analysis. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2010**, *29*, 1884–1893.
13. Mahapatra, S.; Goel, N.; Desai, S.; Gupta, S.; Jose, B.; Mukhopadhyay, S.; Joshi, K.; Jain, A.; Islam, A.; Alam, M.; *et al.* A Comparative Study of Different Physics-Based NBTI Models. *IEEE Trans. Electron Devices* **2013**, *60*, 901–916.
14. Bhardwaj, S.; Wang, W.; Vattikonda, R.; Cao, Y.; Vrudhula, S. Predictive Modeling of the NBTI Effect for Reliable Design. In Proceedings of the IEEE Custom Integrated Circuits Conference, CICC '06, 10–13 September 2006, San Jose, CA , USA; pp. 189–192.
15. Wang, W.; Yang, S.; Bhardwaj, S.; Vrudhula, S.; Liu, T.; Cao, Y. The Impact of NBTI Effect on Combinational Circuit: Modeling, Simulation, and Analysis. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2010**, *18*, 173–183.
16. Kang, K.; Park, S.P.; Roy, K.; Alam, M. Estimation of statistical variation in temporal NBTI degradation and its impact on lifetime circuit performance. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, ICCAD 2007, 4–8 November 2007, San Jose, CA, USA; pp. 730–734.
17. Velamala, J.; Sutaria, K.; Sato, T.; Cao, Y. Aging statistics based on trapping/detrapping: Silicon evidence, modeling and long-term prediction. In Proceedings of the 2012 IEEE International Reliability Physics Symposium (IRPS), Anaheim, CA, USA, 15–19 April 2012; pp. 2F.2.1–2F.2.5.

18. Lotze, N.; Manoli, Y. A 62 mV 0.13 m CMOS Standard-Cell-Based Design Technique Using Schmitt-Trigger Logic. *IEEE J. Solid-State Circuits* **2012**, *47*, 47–60.
19. Kwong, J.; Ramadass, Y.; Verma, N.; Koesler, M.; Huber, K.; Moormann, H.; Chandrakasan, A. A 65 nm Sub-Vt Microcontroller with Integrated SRAM and Switched-Capacitor DC-DC Converter. In Proceedings of the IEEE International Solid-State Circuits Conference, ISSCC 2008, Digest of Technical Papers, San Francisco, CA, USA, 3–7 Feburary 2008; pp. 318–616.
20. Sinha, S.; Yeric, G.; Chandra, V.; Cline, B.; Cao, Y. Exploring sub-20 nm FinFET design with predictive technology models. In Proceedings of the 49th Annual Design Automation Conference (DAC); ACM: New York, NY, USA, 2012; pp. 283–288.
21. Mukhopadhyay, S.; Mahmoodi, H.; Roy, K. Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2005**, *24*, 1859–1880.
22. Choudhury, M.; Mohanram, K. Reliability Analysis of Logic Circuits. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2009**, *28*, 392 –405.
23. Najm, F.N.; Burch, R.; Yang, P.; Hajj, I.N. Probabilistic simulation for reliability analysis of CMOS VLSI circuits. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2006**, *9*, 439–450.
24. Verovka, O.; Parasyuk, I. Probability propagation in fuzzy Bayesian belief networks with nondeterministic states. *Cybern. Syst. Anal.* **2008**, *44*, 925–940.
25. Bogdan, P.; Marculescu, R. Hitting Time Analysis for Fault-Tolerant Communication at Nanoscale in Future Multiprocessor Platforms. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2011**, *30*, 1197–1210.
26. Ibrahim, W.; Beiu, V.; Beg, A. GREDA: A Fast and More Accurate Gate Reliability EDA Tool. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2012**, *31*, 509–521.
27. Miskov-Zivanov, N.; Marculescu, D. Multiple Transient Faults in Combinational and Sequential Circuits: A Systematic Approach. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2010**, *29*, 1614–1627.
28. Wang, D.; Huang, N. Reliability Analysis of Component-Based Software Based on Rewrite Logic. In Proceedings of the 12th IEEE International Workshop on Future Trends of Distributed Computing Systems, FTDCS '08, 21–23 October 2008, Kunming, China; pp. 126–132.
29. Bishop, P.; Bloomfield, R.; Clement, T.; Guerra, S. Software Criticality Analysis of COTS/SOUP. In *Computer Safety, Reliability and Security*; Lecture Notes in Computer Science; Anderson, S., Felici, M., Bologna, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2434, pp. 198–211.
30. Stanisavljevic, M.; Schmid, A.; Leblebici, Y. Selective redundancy-based design techniques for the minimization of local delay variations. In Proceedings of the 2010 IEEE International Symposium on Circuits and Systems (ISCAS), Paris, France, 30 May–2 June 2010; pp. 2486–2489.
31. Lin, M.; Bai, Y.; Wawrzynek, J. Discriminatively Fortified Computing with Reconfigurable Digital Fabric. In Proceedings of the 2011 IEEE 13th International Symposium on High-Assurance Systems Engineering (HASE), Boca Raton, FL, USA, 10–12 November 2011 ; IEEE Computer Society: Washington, DC, USA, 2011; pp. 112–119.

32. Almukhaizim, S.; Makris, Y. Soft Error Mitigation Through Selective Addition of Functionally Redundant Wires. *IEEE Trans. Reliab.* **2008**, *57*, 23–31.
33. Samudrala, P.; Ramos, J.; Katkoori, S. Selective triple Modular redundancy (STMR) based single-event upset (SEU) tolerant synthesis for FPGAs. *IEEE Trans. Nucl. Sci.* **2004**, *51*, 2957–2969.
34. Ruano, O.; Maestro, J.; Reviriego, P. A Methodology for Automatic Insertion of Selective TMR in Digital Circuits Affected by SEUs. *IEEE Trans. Nucl. Sci.* **2009**, *56*, 2091–2102.