Recycled FPGA Detection Using Exhaustive LUT Path Delay Characterization

Md Mahbub Alam, Mark Tehranipoor and Domenic Forte
Department of Electrical and Computer Engineering, University of Florida, Gainesville, Florida
Email: mahbub.alam@ufl.edu, {tehranipoor, dforte}@ece.ufl.edu.

***Abstract*—Field programmable gate arrays (FPGAs) have been extensively used because of their low non-recurring engineering and design costs, instant availability and reduced visibility of failure, high performance and power benefits. Reports indicate that a great amount of counterfeit FPGAs is infiltrating the IC supply chain, most of which are recycled (previously used). Counterfeit components pose a significant threat to the government and industrial sectors of the economy because they undermine the security and reliability of critical systems and networks. Recycled FPGA detection procedures include parametric tests, functional tests, and burn-in tests that requires golden data and/or parts specifications from original component manufacturers. In this work, a sophisticated ring oscillator design method is used to measure the delay of both unused and aged FGPAs. The design uses XNOR and XOR based mapping to exploit all the possible paths in lookup tables (LUTs). A recycled FPGA is likely to have fully used, partially used, and unused LUTs. The proposed mapping targets all the paths of LUTs and forms a frequency array. A support vector machine is trained with frequency array from unused FPGAs, which differentiates between unused and aged FPGAs. An unsupervised method based on k means clustering is also proposed to classify recycled components without golden information. Simulation and silicon data demonstrate high rates of success using the proposed methods.**

***Keywords*—FPGA Aging, LUT path delay, ring oscillators in FGPAs, recycled FPGAs.**

I. Introduction

The problem of counterfeit electronic components continues to grow given the increased complexity of the supply chain and the lack of appropriate countermeasures and detection schemes. Counterfeit components do not possess the exact same specifications as genuine parts, and can impose significant vulnerabilities and threats to the systems in which they are placed. If such parts end up in critical applications (defense, medical, aerospace, transportation, etc.), catastrophic scenarios that result in mission failures, health and safety hazards, and jeopardize national security can occur. Thus, it has become imperative that manufacturers, distributors, and users of electronic components inspect all electronic components for authenticity. In addition, counterfeit parts have a negative impact on corporate identity and reputation which can trigger massive revenue losses.

The counterfeit types include recycled, remarked, overproduced, out-of-spec/defective, cloned, forged documentation, and tampered [1]. Among all of the counterfeit types today, recycled counterfeit components are the most common ones. In 2013, SMT Corp. estimated that recycled ICs comprised 80 to 90% of all counterfeits in circulation worldwide [2]. The recycled electronic components are reclaimed or recovered from a system, and then modified to be misrepresented as a new component of an original component manufacturer (OCM). Since the recycled parts are used in an unknown condition and have generally been recycled in harsh conditions such as high temperature and high humidity, they likely have reliability issues. It should be pointed out that, given the huge volume of containers recycled each year, it is perfectly possible for the same component to be recycled multiple times.

Since their introduction, field programmable gate arrays (FPGAs) have been widely used because of their low non-recurring engineering (NRE) and design costs, instant availability and reduced visibility of a failure, high performance and power benefits [3], [4]. It is forecasted that the global FPGA market will reach approximately $\$9,883$ million by 2020 [4]. Reports from 2012 show that programmable logic is in the top 5 counterfeited electronic components with the percentage of 8.3% of reported counterfeit incidents [5]. With the increased volume of usage, FPGAs will likely become an even better target for counterfeiting and thus its reliability becomes a great concern to government and industry. In today's complex electronic component supply chain, it is very challenging to prevent the infiltration of recycled FPGAs.

Physical and Electrical test methods have been developed to distinguish counterfeit from authentic components [6]. Physical tests are performed to examine the physical and chemical/material properties of the component. The electrical inspection phase of the tests includes AC/DC parametric tests, functional tests, and burn-in tests. Although physical inspection methods are effective in detecting visual defects of recycled parts, methods are time consuming, costly and sometime destructive. Sophisticated recycled components having visual properties as good as original components can pass physical test methods. Electrical test methods overcome these disadvantages and meets the functionality, quality, authenticity and reliability requirements.

There has been few works aimed at recycled IC detection by using electrical tests and on-chip sensors [7]–[10]. Authors in [11] proposed ring oscillators (RO-based) and antifuse (AF-based) on-chip lightweight sensors to identify recycled ICs by measuring circuit usage time. Aging monitoring sensors inserted on critical paths have been used by [12], [13] to detect aged paths. In [10] authors suggested light-weight on-chip sensors based on ring oscillators (RO-CDIR), anti-fuses (AF-CDIR) and fuses (F-CDIR) for detecting IC usage. Sensors

based recycled IC detection incur hardware overhead and do not work for existing ICs in the market. They also dont directly apply to FPGAs.

Zhang et. al. [9] proposed a path-delay based method to detect recycled ICs that does not add area overhead. Huang et. al. [8] proposed statistical methods for detecting recycled ICs through the use of one-class classifiers and degradation curve sensitivity analysis. These techniques rely on drifts of parametric profile of an IC due to silicon aging and use parametric behavior of new devices as a reference point to detect recycled ICs. Most of the prior works for recycled IC detection neglect FPGAs and focus on ASICs. The only exception is the work by Dogan et. al. [7], where are a two phase detection approach targeting FPGAs was proposed. Phase I detects recycled FPGAs by comparing the frequencies of ring oscillators (ROs) distributed on the FPGAs against a golden model. This phase fails to detect FPGAs at fast corners and those with lesser prior usage. To overcome this, phase II was proposed which applies an accelerating aging step and exploits the aging rate (lower in case of prior usage) of FPGAs to the suspect components from phase I. This method implements inverter based RO that covers few logic paths in FPGA. Thus, it does not represent the actual delay characteristics of the FPGA. In addition, unused FPGAs having accelerated aging in phase II suffer from early life degradation.

In this work, we attempt to reduce the recycled detection failure of fast corner FPGAs while avoiding an accelerated aging phase entirely. We made the following contributions

- We analyze FPGA aging to understand the nature of recycled components. A recycled FPGA is likely to have fully used, partially used, and unused lookup tables (LUTs). Each of them exhibits different behavior in terms of path delay variation which is useful for distinguishing new and used/recycled FPGAs.
- We construct an exhaustive path delay characterization scheme by implementing a sophisticated ring oscillator design method. The design uses XNOR and XOR based mapping to *all the possible paths in lookup tables (LUTs)*.
- We propose a supervised learning technique based on support vector machine which is trained on the frequency data (obtained above) from unused golden FPGAs. This creates a boundary between unused and aged FPGAs.
- We also propose an unsupervised machine learning method based on k-means clustering and silhouette analysis, which does not demand reference data from golden FPGAs.
- We provide simulation and silicon results to demonstrate the effectiveness of the proposed detection methods.

The rest of the paper is organized as follows: In Section II, we discuss the background on look-up table (LUT) structures, ring oscillator formation in FPGAs, and aging mechanisms. LUT path characteristics, path configurations and proposed approaches are described in Section III, IV, and V. Section VI presents the experimental results and analysis. We conclude the paper in Section VII.

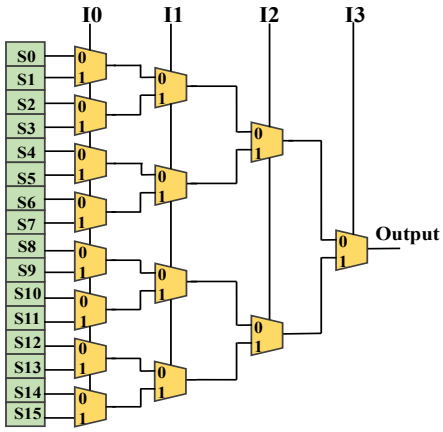


Figure 1. Structure of a 4 input Look-up Table.

II. Background

In this section, we provide background on look-up table (LUT) structures, ring oscillator formation in FPGAs, aging mechanisms, and impact of aging on FPGAs.

A. Look-up Table (LUTs) Structures

FPGAs implement logic functions through configurable logic blocks (CLBs). Look-up tables (LUTs) acts as function generators in CLBs and can be considered as a basic building blocks of FPGA configurations. Modern FPGAs allow modification to the mapped function of LUTs through reconfiguration. Thus, LUTs collectively implement billions of logic functions. It is thereby important to understand behavior of LUTs to study the aging degradation for recycled FPGA detection. An LUT is typically built out of SRAM bits to hold mapped values and a set of multiplexers to select the bit that is drives the LUT output. An example of an LUT architecture is illustrated in Fig. 1. In this example, a 4 input LUT is shown which consists of 16 SRAM cells and a 16:1 multiplexer. A tree of 2:1 multiplexers has been used to build the 16:1 multiplexer. Any logic function of 4 inputs can be realized by setting the appropriate value in the SRAM cells and 4 level hierarchical selectors (I0, I1, I2, I3).

The circuit level design of LUTs is not revealed by the vendor. However, a systematic examination of LUT characteristics claims a good knowledge of internal circuit design. Thus, researches has studied different architecture of LUTs to support their observations in few prior works [14]–[16]. Pass transistor based 2 input LUTs has been used to measure and model the degradation of FPGAs in [15]. Saman et. al [16] investigated bias temperature instability (BTI) in pass transistors, transmission gates, and logic gates based on 2 input LUTs. Since this work focuses on all the possible paths on LUTs, we examine larger 4 input LUTs. Here we briefly describe three different LUT implementations that we have studied in this work. For simplicity, only 2 input LUTs are described in detail. Note that this discussion is important for describing how different types of FPGAs can experience aging in later sections.

A transmission gate based LUT is depicted in Fig. 2. The structure consists of four SRAM cells (S0-S3) to store

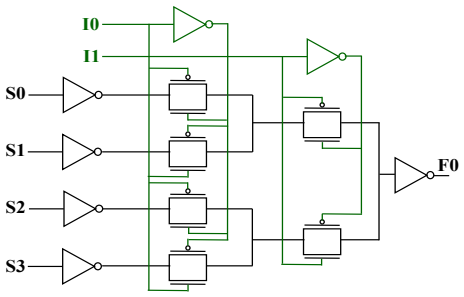


Figure 2. Transmission gate based implementation of a 2 input LUT.

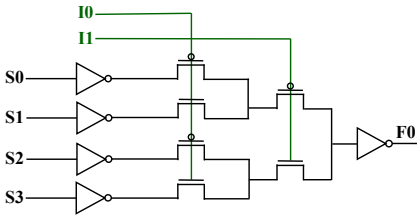


Figure 3. Pass transistor based implementation of a 2 input LUT.

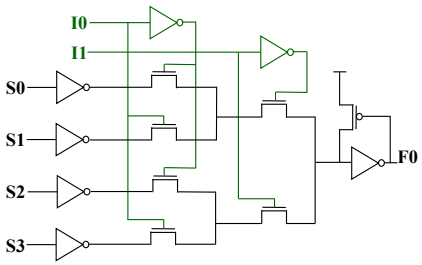


Figure 4. NMOS pass transistor based implementation of a 2 input LUT.

the mapped values and two inputs (I0 and I1) to select the corresponding transmission gates. Complementary input pins are used to form 4:1 MUX logic. Six pairs of NMOS and PMOS transistors build six transmission gates to form the multiplexer. Since the transmission gates can pass both strong logic zero and logic one, this implementation optimizes delay. However, this comes at a price of increased area.

The circuit diagram for the pass transistor based 2 input LUT is given in Fig. 3. It contains either NMOS or PMOS transistors to transmit logic. Three NMOS and three PMOS pass transistors form the multiplexers and propagate weak logic value ('0' or '1'). Thus, a level restoring stage is used to provide a strong output. Downscaling of technology leads to a small difference between supply voltage and threshold voltage of transistors, and hence this structure faces logic restoring issues in newer technologies. Mapped values in SRAM cells propagate through a series of pass transistors, and suffer from weak logic transmission. Thus, it has more propagation delay than transmission gate based structure.

LUT structure with only NMOS pass transistor as selection circuit is presented in Fig. 4. Since, NMOS transistor pass weak logic '1', a half latch keeper circuit is needed. This structure also suffers from logic restoring issues due to downscaling of technology, but it provides an advantage over transmission gate architecture by minimizing the area.

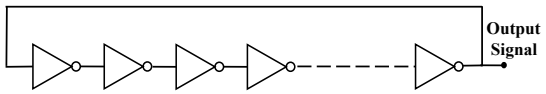


Figure 5. Ring Oscillator with odd number of inverters.

B. Ring Oscillators (ROs) Configuration in FPGAs

The use of ring oscillators (ROs) is a standard technique for measuring delay variation in ICs [17]. In this work, ROs will be created to compare the performance of FPGAs. An RO is a circuit that consists of an odd number of inverting delay stages connected in series to form a closed loop chain. An example where each delay stage consists of an inverter is shown in Fig. 5. The oscillation period is twice the sum of the delays of all elements that compose the loop. Thus, the oscillation frequency of n stages of inverters can be expressed as:

$$f_0 = \frac{1}{2\sum_{i=1}^{n} \tau_{d,i}} \tag{1}$$

where n is the number of stages and $\tau_{d,i}$ is the propagation delay of ith stage.

ROs can be mapped on FPGAs using LUTs that implement an inverting stage. Propagation delay of each stage is the sum of the delay contributed by SRAM cells, selector transistors, and interconnect delay.

C. Aging Mechanism

An operational FPGA slows down over the course of its lifetime. The degradation mechanisms includes bias temperature instability (BTI), hot Ccrrier injection (HCI), iime dependent dielectric breakdown (TDDB) and electromigration [16], [18]–[20]. This work focuses on BTI and HCI due to their significant impact on the switching speed of transistors. This impact is measurable and can be used for recycled FPGA detection.

1) Bias Temperature Instability: BTI is one of the major reliability concerns in MOS technology that affects the threshold voltage of transistors. Negative BTI (NBTI) and positive BTI (PBTI) increase the threshold voltage of pMOS and nMOS respectively. PMOS transistors suffer from NBTI during prolonged times of negative bias stress. Prolonged stress creates interface traps at interface of gate oxide and channel that increases the threshold voltage, which, in turn, decreases the switching speed. This effect is dominant at high temperature and voltages. Removal of the stress conditions allows some release of trapped charges. NBTI is dominant compared to PBTI beyond 65nm technology nodes. However, the introduction of high-k gate dielectrics and metal gates transistors elevates the effect of PBTI [16] that creates positive charge defects in NMOS transistor. Since modern FPGAs are scaling beyond 65nm [3], they are prone to PBTI effects.

2) Hot Carrier Injection: Similar to BTI, HCI effect leads to increased threshold voltage and degrades the switching speed of the transistor. This phenomenon happens when electrons or holes in the substrate attaining higher energies above the average due to very high electric eld in the drain region and get trapped in gate oxide layer. Over the course of time, charge defects build up electric charge within the dielectric layer, increase the threshold voltage, and decrease the mobility. It irreversibly slows down the switching activity of the transistor. Degradation due to HCI becomes worse in lower technology nodes as effective channel length gets smaller. Since HCI is

driven by the flow of energetic carriers, it happens during dynamic switching activity.

D. Impact of Aging on FPGAs

BTI and HCI degrade the performances of FPGAs over time by affecting threshold voltage of the circuits. As propagation delay of the BTI and HCI induced transistors increases, selector circuits of LUTs slows down. NBTI threshold degradation slows down PMOS transistors and significantly reduces the static noise margin of SRAM cells. Electromigration due to gradual movement of ions causes wire faults and TDDB caused by prolonged exposure to low electric field degrades performance as well. All degradation mechanisms are highly dependent on temperature with high temperature accelerating the aging. A significant aging degradation has been reported by [15], [21] at high temperature and voltage.

In this paper, these aging phenomena are important for two reasons: (i) it motivates the need for recycled FPGA detection. Used FPGAs will be slower and could have unacceptably premature failures if employed in critical applications; (ii) the fact that recycled FPGAs have measureable performance degradation in some LUTs implies that it is possible to detect them through intelligent, low-cost electrical tests.

III. LUT Path Delay Analysis

As stated in Section II-A, LUTs are the basic blocks for mapping logic circuits. Depending on the configuration and input signal, different paths are used to propagate logic stored in SRAM cell to the output. Note that it is highly unlikely that all the paths in a LUT will be used. For instance, 3 input 'OR' gate implementation in a LUT requires eight SRAM cells with corresponding selector circuits. As shown in Fig. 6 S0-S7 cells can implement 'OR' gate logic using I0, I1, I2 with I3=0, where S0=0, S1=1, S2=1, S3=1, S4=1, S5=1, S6=1, and S7=1. Propagation paths are depicted in red in Fig. 6. This example uses 8 paths from 16 possible paths provided by 4 input LUTs. This mapping leaves unused portions (lower half of the LUTs). The amount of unused portions depends on the spare input. An LUT with one spare input leaves half of LUT unused, two spare inputs uses only quarter of its resources. Modern architecture provides 4-, 5- or 6-input LUTs. Since logic in most cases does not use all the inputs, many LUTs are partially filled. This leads to a number of partially aged LUTs.

All the available logic resources are rarely used in practical FPGA applications. Hence, spare LUTs are frequently found in most designs. Aging degradation of these unused LUTs are less than used LUTs that go through different ac/dc stress [22].

BTI and HCI induced degradation depends on the history of LUT configurations, input signal probabilities [16], and switching activities. LUTs with different configurations and different input signal probabilities have different aging-induced path delay. Thus delay variations can be observed even in completely filled LUTs.

Delay characterization using ring oscillators (ROs) have been studied in earlier works [7], [17]. In [7] ROs are created

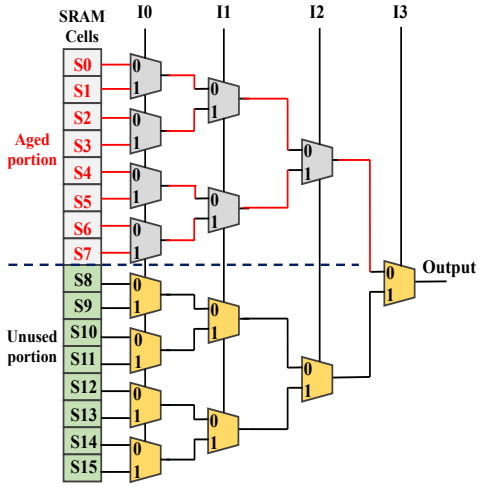


Figure 6. An example of aged and unused portion of Look-up Table (red line shows the aged path).

by implementing inverter logic that spreads over a single path. The ROs implemented using one set of input pins *does not cover all the paths of the LUTs under test*. When a certain area of the FPGA is suffering from the aging effect, a method which only characterizes the delay of one path or some path is not comprehensive enough to give a complete delay characterization. For instance, if covered path contains spare LUTs or unused paths of partially filled LUTs, path delay will not represent the actual aging characterization. Therefore, a sophisticated design of test configuration is required to overcome this concern.

This work focuses on characterizing delay variations of *all the possible paths* regardless of unused, partially used and fully used LUTs that helps to better differentiate between unused and recycled FPGAs. In order to achieve this we form ring oscillator that has the following properties:

- All the MUX circuits of the LUT should be used in the RO loop.
- All the SRAM values should be read out during test configurations.

An RO with above attributes provides all the possible paths for exhaustive delay characterization.

IV. Implementation of Path Configurations

We form ring oscillators using odd number of LUT stages and all the LUT input pins. Figure 7 illustrates a possible formation of ring oscillator with N number of stages using four input LUTs. In this example, 16 SRAM cells are located on the left side in each LUTs and their outputs are selected by 4-level hierarchical selectors (4 input MUX). Output of each LUT is connected to selection bits of next LUTs and a closed loop chain of inverters is formed. A proper mapping of SRAM cell data and MUX selection bits are required to obtain the oscillation behavior from LUTs. Since one of the inputs is required to form the chain, the rest of the input combinations can create different paths. For k input LUTs, 2^{k-1} number of paths can be obtained [23].

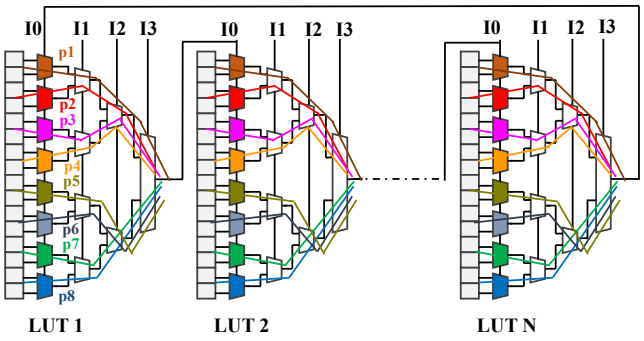


Figure 7. Ring oscillator in 4 input LUTs with eight possible path (p1-p8) shown in different color.

Let $p = 2^{k-1}$ represent the number of paths. We denote $P_1, P_2, P_3, ..., P_p$ as all p paths and the frequency of each path as $f_1, f_2, f_3, ..., f_p$ respectively. We collect all the ring oscillator frequencies in a frequency array

$$f_{array} = [f_1, f_2, f_3, f_4, ..., f_p] \quad (2)$$

All the frequencies of f_{array} will vary a little due to process variation [17]. State of the art design optimizes timing behavior of the LUTs and thus frequency variation is expected to be minimized in unused FPGAs.

In used FPGAs, aging degradation will affect each path differently. Simulated delay characteristics of a seven stage 4 input LUT in unused and used conditions (details in VI-A) is shown in table I. 4 input LUTs provide eight possible path. Path P1-P4 shows the greater degradation than others and provides favorable circumstance for classification.

Implementation of ROs in FPGAs faces challenges from design and synthesis tools. Generally, designers do not have enough control over the synthesis tool that optimizes boolean functions, place and routing configuration, etc. This causes problems in implementing low level design like RO from accessing all SRAM cells and selector paths. Hence, a Boolean function that implements inverter logic and selects all the entries of the truth table is needed. XNOR and XOR functions can satisfy the requirement. An example of XNOR and XOR based mapping for 4 input LUT is illustrated in Fig 8. Here, eight path has been configured for inverter logic. A set of {I1, I2, I3} input combinations determine the path P1-P8 and I0

Table I
SIMULATION RESULT OF UNUSED AND AGED PATH DELAY VARIATION. NOTE P1-P8 CORRESPOND TO THE PATHS SHOWN IN FIGURE 7.

Path	Unused		Aged	
	Mean (MHz)	Standard deviation (MHz)	Mean (MHz)	Standard deviation (MHz)
P1	68.7817	0.4806	65.0157	0.4751
P2	69.6022	0.4317	65.7677	0.4354
P3	68.7407	0.5307	66.8057	0.4996
P4	69.5668	0.4298	63.4969	0.4339
P5	70.0030	0.4334	68.242	0.4269
P6	69.5760	0.4302	68.2103	0.4499
P7	68.5781	0.4392	66.4915	0.4179
P8	69.5581	0.4407	68.8625	0.4264

I3	I2	I1	I0	XNOR	XOR	Path
0	0	0	0	1	0	Path 1
0	0	0	1	0	1	
0	0	1	0	0	1	Path 2
0	0	1	1	1	0	
0	1	0	0	0	1	Path 3
0	1	0	1	1	0	
0	1	1	0	1	0	Path 4
0	1	1	1	0	1	
1	0	0	0	0	1	Path 5
1	0	0	1	1	0	
1	0	1	0	1	0	Path 6
1	0	1	1	0	1	
1	1	0	0	1	0	Path 7
1	1	0	1	0	1	
1	1	1	0	0	1	Path 8
1	1	1	1	1	0	

Figure 8. Formation of paths in 4 input LUTs using XNOR logic (light orange) and XOR logic (gray) function.

acts as the inverter input logic. Using input pins except I0 will also exhibit the same oscillating behavior, but with different paths used. XOR and XNOR functions both share half of the paths and cover all the possible configurations the RO.

V. RECYCLED FPGA CLASSIFICATION

We described in previous sections that XNOR and XOR based mapping can exploit path delay o of all paths in the circuit and therefore used to generate a frequency array. Here, we discuss two approaches to detect recycled FPGAs utilizing this path delay/frequency information: (i) with golden data available and (ii) without golden data. For the former we can measure the above frequency vector for ROs that cover all paths of unused FPGAs, and then form a database of golden data. The same frequency vector can be measured for FPGAs under test and support vector machine (SVM) can classify them. As described earlier, aging induced degradation affects the propagation delay of each path differently, to clearly differentiate between aged and unused FGPAs.. For the latter, we will propose an unsupervised learning method that can distinguish well between used and aged FPGAs.

A. Classification with One Class Support Vector Machine

We construct the frequency array of golden FPGAs using XNOR-XOR based mapping. We use this data to train a one-class Support Vector Machines (SVM) that creates a decision boundary. FPGAs under test go through the same measurement process as the golden FPGAs and frequency array for test FPGA is genrated. The trained one class classifier examines test data using decision boundary and classifies test components as recycled or unused. The proposed approach is illustrated in Fig. 9. One class SVM training and detection process will be discussed below.

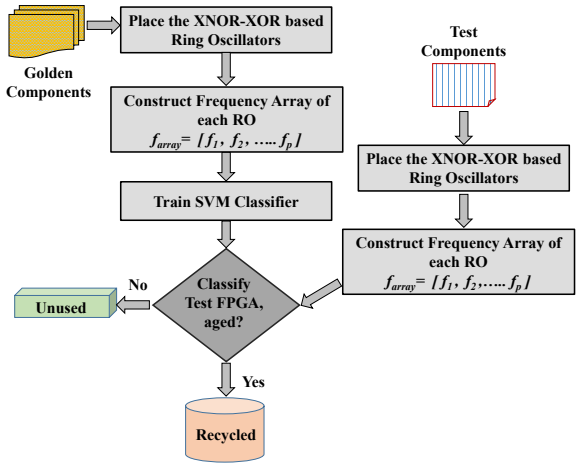


Figure 9. Block diagram of proposed approach.

Support Vector Machines (SVM) is a distinctive classifying technique that has been generally used for data classification. This technique separates the data into training and testing sets. The training set contains the class labels and several attributes (i.e. the features or observed variables). The features in our training set are the frequencies of all paths collected from a golden (unused) FPGA. Based on the training data set, the SVM creates a decision model which predicts the label of the test data given test data has the same attributes as training data. Since we do not have the prior knowledge of the usage time and environment condition of recycled components, we consider one class (or single class) support vector machine that only requires training data from single class (i.e., the unused or golden FGPA frequency data). By just providing the training data from unused FPGAs, an algorithm creates a decision model of this data. If test data is different, decision model labels it as out-of-class (i.e., recycled in this paper).

Tax and Duin [24] and Schölkopf et al. [25] independently introduced one class classification as an extension of the support vector methodology. One class classifier maps training data into a high dimensional feature (H) space via a kernel function and finds the maximal margin hyperplane from the origin iteratively [25]. This method can be imagined as a regular two-class SVM where first class contains all the training data, and the origin is taken as the only member of the second class.

Before a classifier can assign labels to test components, the parameters of the classifier's model should be determined. To do so, we obtain data from M golden FPGAs. Each FPGA contains N number of ROs. The training data set is as follows:

$F_{training} = [S_1, S_2, S_3, ..., S_M]$

for M golden samples.

$S_m = [f_{array,1}, f_{array,2}, f_{array,3}, ..., f_{array,N}]$

for N ROs in each sample.

$F_{training}$ is the total training set. $S_1, S_2, S_3, ..., S_M$ denotes the feature vector from M golden samples. Frequency vector of each sample is $f_{array,1}, f_{array,2}, f_{array,3}, ..., f_{array,N}$ with N number of ROs. Each element of frequency vector is constructed following the definition of frequency array in Equation (2).

Let ϕ be the feature map ($F_{training} \rightarrow H$) that uses kernel function to transform the training points into high dimensional Hilbert space [26]. Generally, the hyperplane in feature space induces a nonlinear surface in the input space. Thus, the following quadratic programming problem needs a solution to separate the data points from the origin

$$\min_{w,\varepsilon,\rho} \frac{1}{2} \parallel w \parallel^2 + \frac{1}{\nu M} \sum_{i=1}^{M} \varepsilon_i - \rho \tag{3}$$

subject to $\langle \mathbf{w}, \phi(\mathbf{S_i}) \rangle \geq \rho - \varepsilon_i$ where $i = 1, 2, ..., M$ and $\varepsilon_i \geq 0$. Here ε_i is the slack variable that allows for some points to be within the margin in the scenario of a nonexistent separating hyperplane. The free parameter ν is used to characterize the solution by setting an upper bound on the training samples which are classified as outlier and setting a lower bound on the number of support vectors. Optimal normal vector w defines the hyperplane. The distance from the hyperplane to the origin is given by the margin $\frac{\rho}{\|w\|}$. Decision function $D(\mathbf{S_x})$ for test data $\mathbf{S_x}$ is calculated by using the Lagrange multiplier (α_i) and a kernel function.

$$D(\mathbf{S_x}) = sgn(\langle \mathbf{w}, \phi(\mathbf{S_i}) \rangle - \rho) = sgn(\sum_{i=1}^{M} \alpha_i K(\mathbf{S_x}, \mathbf{S_i}) - \rho) \tag{4}$$

Classification label is created based on the value of $D(S_x)$. Any anomaly on test data leads to $D(S_x) < 0$ and labeled as out of class, otherwise it is within the training class.

Commonly used kernel functions include linear, polynomial, sigmoid, and radial basis function (RBF). In general, the RBF kernel is used as it nonlinearly maps samples into a higher dimensional space and can handle the case when the relation between class labels and attributes is nonlinear. It also offers fewer numerical difficulties [27]. This kernel deals well with data containing Gaussian distribution. Since the RO frequency data follows the Gaussian distribution, RBF kernel is the one we choose for our classification problem. RBF kernel function can be expressed as

$$K(\mathbf{S_x}, \mathbf{S_i}) = exp(-\gamma \mid \mathbf{S_x} - \mathbf{S_i} \mid^2), \gamma > 0 \tag{5}$$

Here, γ is kernel parameter that determine how far a single training example can have effect. Large γ indicates a few number of training samples in kernel. In contrary, small γ kernel has wide boundary containing more training examples.

To classify any test component using decision function $D(S_x)$ we place XNOR-XOR based mapping on it and measure the frequency array for each RO. Let S_t represent the test data,

$S_t = [f_{array,1}, f_{array,2}, f_{array,3}, ..., f_{array,N}]$

for N number of ROs.

Using equation 4 value $D(S_t)$ is calculated and they are labeled as unused (if $D(S_t) > 0$) or recycled (if $D(S_t) < 0$) as shown in Fig. 9.

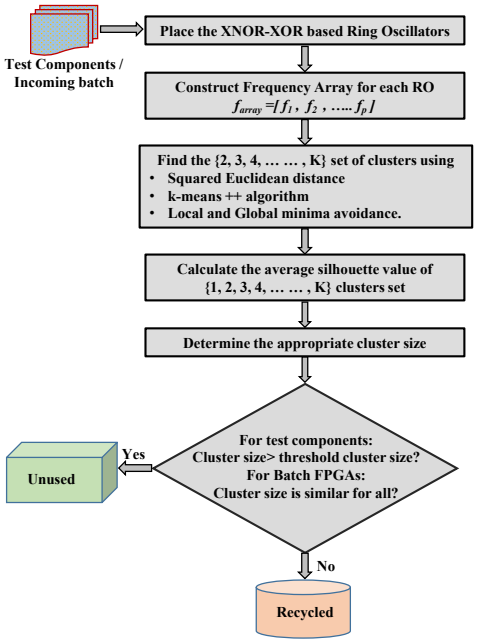


Figure 10. Proposed approach using unsupervised machine learning.

B. Classification Using Unsupervised Method

Sometimes it is difficult to find golden FPGAs. For example, legacy FPGAs are no longer being manufactured, but are still in use in older systems. Thus, the necessity arises for a method that can detect recycled FPGAs without prior knowledge. Here, we described an unsupervised classification method for a single FPGA component and batch FPGAs as shown in Fig. 10. Note that this method also utilizes frequency characteristics obtained from XNOR-XOR based ROs. However, instead of SVM, we perform clustering of the obtained frequency using k means algorithm. Then, for classification, we calculate the average silhouette value [28] of each cluster. The silhouette value for each observations point indicates similarity of that point with other points in its own cluster compared to points in other clusters. We use this value to determine the appropriate cluster number of the test FPGA. Cluster number is essentially used as decision label (unused or recycled) for test FPGAs. Details of the detection process is described below.

As described earlier, for a fresh FPGA, path characteristics are similar as only process variation impacts the delay. In used FPGA component, partially filled, completely filled and spared LUTs age differently and increase the delay variation. Thus, frequency distribution of aged and unused FPGAs are expected to be different. We presented a HSPICE simulation result of an unused and aged FPGA in Fig. 11. The frequency distribution shows that frequencies are clustered in unused FPGA. In contrast, the frequency distribution of a used FPGA can be multi-modal and and thus can be divided into more clusters. Similar phenomena is also reported in [7].

We use k-means method [29] also known as Lloyd's algorithm [30] to characterize this phenomena. The method is a widely used clustering technique as it is fast, robust and efficient. The basic idea of k-means method is to partition

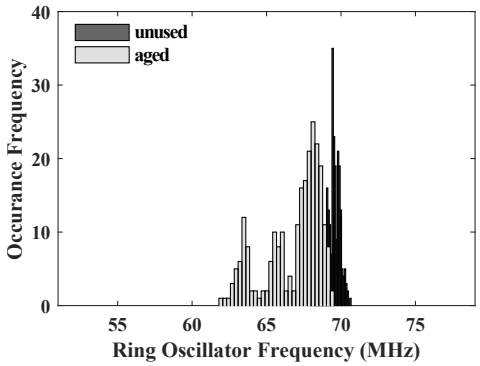


Figure 11. Simulated frequency distribution of unused and used FPGAs.

frequencies into mutually exclusive clusters fixed a priori and find the index of the cluster assigned to each frequency. Frequencies within each cluster are as close to each other as possible, and as far from frequencies in other clusters as possible.

Let frequency data obtained from a test FPGA be represented as $S_t = [f_{array,1}, f_{array,2}, f_{array,3}, ..., f_{array,N}]$ for N number of ROs as defined in earlier sections. Each frequency array is formed following the equation (2). Thus, total number of frequency used in this algorithm is RO number (N) multiplied by number of possible path (p). The k-means algorithm used to cluster this data into k parts is as follows:

1) Initialize the center $C = c_1, c_2, c_3, ..., c_k$ of the k clusters using k-means++ initialization algorithm.
2) Compute the squared Euclidean distance between data points and center of clusters of all frequencies to each center. Squared Euclidean distance from frequency f_i to cluster center c_j is

$$d(f_i, c_j) = \|f_i - c_k\|^2 . i \in \{1, 2, ..., (N \times p)\}.$$

3) Assign the each frequency points f_i of the frequency array to the cluster center c_j $j \in \{1, 2, ..., k\}$ whose distance $d(f_i, c_j)$ from the cluster center is minimum among all the cluster i.e. $min(d(f_i, c_j))$.
4) Recalculate the k new center $c_{new,k}$ by computing mean of all data points belonging to that cluster:

$$c_{new,k} = \frac{1}{|c_k|} \sum_{f_i \in c_k} f_i$$

5) Recalculate the Euclidean distance of each frequency using new obtained cluster centers.
6) Repeat step 3, 4, and 5 until cluster centers move no more.

Our method uses k-means++ algorithm [31] for initializing the cluster centers as it improves the quality of the final solution and running time of Lloyd's algorithm. This procedure incorporates batch and online algorithm [32] to solve the convergence problem by avoiding the global and local minima.

We calculate cluster indices of each frequency for every cluster number that indicates the cluster assignment of the corresponding frequency. Since our objective is to find the appropriate cluster number for a given FPGA data, a set of

clusters $R = [2,3,...,K]$ is formed using k-means algorithm (i.e. 2 cluster from all data, etc.). Optimal choice of cluster number leads to balance between matching the frequency within its own cluster and difference with the neighboring clusters. The silhouette value [28] refers to this balance and helps to find the proper cluster number.

To calculate the silhouette value we find average Euclidian distance of a frequency point to the other frequency of its own cluster and to frequency points to the neighboring cluster. Let average Euclidian distance of the *i*th frequency point within its cluster be OC_i, and to neighboring cluster is NC_i(minimized over clusters). The silhouette value SV_i for *i*th frequency is expressed as

$$SV_i = \frac{OC_i - NC_i}{max(OC_i, NC_i)}, -1 < SV_i < 1$$

A high silhouette value suggests that *i*th frequency fits well to its own cluster, and poorly-matches to neighboring clusters. The clustering solution is appropriate if most frequency points have a high silhouette value.

Silhouette value for each frequency points is calculated for every cluster number. Next, we find average silhouette values for each cluster number $avg(SV_2), avg(SV_3), ..avg(SV_K)$. Appropriate cluster number (ACN) for the test components is cluster number that has maximum average silhouette value. For recycled FPGA detection we compare appropriate cluster number (ACN) with threshold cluster number (TCN). The decision function can be expressed as:

$$D(S_t) = \begin{cases} Recycled; & \text{ACN of } S_t > TCN \\ Unused; & otherwise \end{cases}$$

The appropriate cluster number (ACN) is lower for unused FPGAs, while it is higher for used FPGAs. Thus we can set a threshold value that is close to the expected cluster number of unused FPGAs. We can extend this method for a batch of FPGAs by comparing the similarities of cluster number among themselves.

Note that one potential attack on FPGAs would be to perform the same measurements on path delay and intentionally age the rest of the LUT paths. However, this requires far too much time and work for a counterfeiter. Thus, it is unrealistic and not worth worrying about.

VI. Experimental Results and Discussion

A. Experimental Setup

We discuss the simulation procedure, experimental setup, and the devices that were used in our experiments in following subsections.

Pass transistor, transmission gate, and nmos pass transistor based 4 input LUTs are created in HSPICE [33] in 90nm technology node using the Monte Carlo simulation. We implemented 30 FPGA chips for each of three LUT structure and each of them contain 50 ROs. We placed 7 stage ROs that provides 8 paths (2^{4-1}) for 4 input LUTs. Transistor model paramters for 90nm technology has been obtained from [34]. Process variation for the chips includes 10% inter-die, 5% intra-die, and 3 sigma variations for channel length L, channel width W, threshold voltage V_{th}, and gate oxide thickness Tox. Minimum size transistors (Wn = 0:12u and Wp = 2:5 Wn) are used to ensure similar rise and fall time. We measure period of oscillation at 50% signal transition point. Bias temperature instability (both NBTI and PBTI) and HCI have been accounted in aging analysis using HSPICE MOSRA [33]. MOSRA uses pre-stress and post stress for aging simulation. We did aging aging simulation for five different time span (2 hour, 1 day, 1 week, 1 month, 6 month). After each time span we measure the frequency array. Simulation result has been applied to both one class classifier and unsupervised method.

In this work, we also implemented XNOR-XOR based RO in FPGAs. We used Spartan 3E FPGAs (90 nm technology node) and implemented 265 ROs. Each RO is 7 stage and placed in a CLB that contains 8 LUTs. Hardmacro has been used for the RO design, so every RO has the same internal routing and structure. Thus, we minimize the frequency variations induced by the routing differences. Measurement is done in room temperature and with the on-chip clock. To avoid the measurement noise each frequency is measured 10 times and average value is taken. We used the same conditions and same approaches for 7 FPGAs (5 unused and 2 aged). Since sample number is not large here, we can not consider them as golden data and classify using one class classifier method. Hence, Silicon data from FPGA is only used for unsupervised detection method.

B. Results and Discussions

1) Simulation Results for One Class Classifier: As described earlier we placed 50 ROs in each of 30 chips. For training purposes, we considered 15 of them as golden FPGA chips and measured the frequency array of each ROs. A decision boundary for one class classifier was created following the method in V-A using this golden data. This process is followed for all three structures of LUTs: transmission gate, pass transistor and NMOS based. We used one class classifier to detect the remaining 15 unused FPGA chips. Although classifier could detect all the unused FPGAs for transmission based and NMOS based structures, it failed to detect two of the unused FPGAs in pass transistor based structures. This can be explained considering the dominance of either NMOS or PMOS transistors, as they are connected in series to propagate a logic value.

In the next step, some of the paths of LUTs are aged to incorporate the path delay variation. This is done as in practical applications, i.e., all paths are not equally aged. We started with the case of two-hour maximum aging path. The trained classifier predicts inaccurately most of aged FPGA chips for all the structure. The path delay is affected by process variation and a small degradation is not enough to reflect in delay characterization. This detection process is continued for 1 day, 1 week, 1 month, 6 month aged chips. The trained one class classifier can detect chips with these aged path as aged/recycled with high confidence for all the structures.

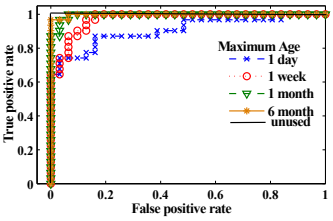


Figure 12. ROC curve for transmission gate based LUT structure.

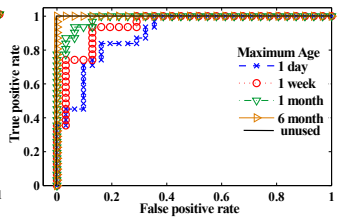
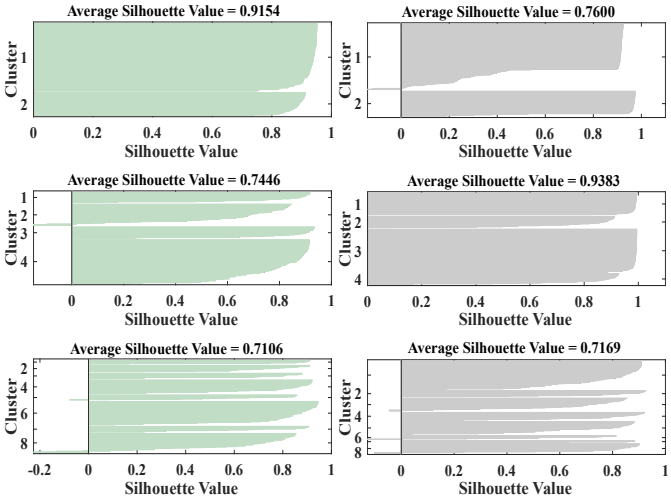


Figure 13. ROC curve for NMOS based LUT structure.

Detection performance of the classifier for transmission gate based structure is illustrated in a receiver operating characteristic (ROC) curve (Fig. 12). ROC curves are useful technique to visualize and compare the prediction accuracy of the classifier while varying discriminating threshold. The figure shows that with the increase of usage time from 1 day to 6 month, the ROC curve shifts to left and top axis, and area under the curve increases. Thus, Detection success rate of the classifier is increasing. This corresponds to better performance, i.e., lower false positive and higher true positive rates. From the area under the curve it is found that classifier can detect 1 day and 1 week aged/recycled chip with 89% and 95% probability. It shows a probability of more than 98% for 1 month and 6 month maximum aged chips. Figure 13 depicts the ROC curve for classifier created for NMOS based LUT structure. This ROC curve shows similar behavior as previous classifier built for transmission based structure but area under the curve is slightly lower than before. Thus its detection probability is lower. A detection probability of 86%, 93%, 95% is observed for 1 day, 1 week, 1-month maximum usage path. The classifier exhibits about 100% probability for 6-month maximum usage path. Since NMOS suffers less from aging degradation, path delay changes slowly. The classifier for pass transistor based structure also manifests the similar behavior.

Based on the simulation results it is observed that, the proposed method based on exhaustive path delay characterization detects recycled chips regardless of LUT structure with high probability, particularly for chips aged more than a week. This method uses golden data and a greater number of golden chips available for training will likely improve the performance of the classifier.

2) Simulation Results for Unsupervised Machine Learning Algorithm: The recycled FPGA detection approaches proposed until now requires data from golden FPGAs. Since golden data is not available in all cases, we proposed a unsupervised detection method in V-B. The frequency array has been constructed for 30 chips with three LUT structure as described in earlier sections. Using the measured frequency, k-means cluster set has been created and average silhouette value is calculated. Figure 14 shows an example average silhouette value calculation process for transmission gate based LUTs. Silhouette value of each frequency is in x axis and cluster number is shown in y axis. Maximum average silhouette value for unused chips (Fig. 14(a)) is 0.9154 at cluster number 2. Maximum frequency has high silhouette value at cluster



(a) Silhouette value for unused FPGA (b) Silhouette value for aged FPGA

Figure 14. Silhouette value for each frequency at 2 (top), 4, 8 (bottom) cluster.

number 2 (top), thus they are well matched within their clusters and appropriate cluster number (ACN) is 2. Following the approach, ACN for aged chip is 4 with maximum average silhouette value of 0.9383.

As described earlier, ACN for unused FPGA is expected to be lower than for an aged one. It is observed from the simulation result of all the chips and LUT structure that ACN is 2-3 for unused chips. It is expected since only process variation impacts the path delay and variation is minimum for an optimized architecture. Silicon results also manifest similar behavior (discussion forthcoming for Table III).

Considering a threshold cluster number as 3, detection approach is applied on all the chips at unused and aged conditions. Classification results are for the simulation are shown in Table II. Detection rate is less than one class supervised method but is still very effective in higher usage FPGAs.

3) Silicon Results for Unsupervised Machine Learning Algorithm: For silicon results, we constructed a frequency array for 265 ROs implemented in Spartan 3E FPGAs. Cluster number and status of the FPGAs are demonstrated in Table III. Using the same threshold value as simulations (3), the unused and aged FPGAS are all correctly classified. The aged FPGAs have appropriate clusters numbers of 6 and 8 ($>>$ 3) while the unused FPGAs have appropriate cluster values $\leq$ 3). Although the number of sample chips is limited, this unsupervised method seems very effective for detecting recycled FPGAs.

Table II
UNSUPERVISED METHOD DETECTION RATE(%).

Maximum Age of path	Mux based LUT	Pass transistor based LUT	NMOS based LUT
unused	97%	90%	97%
1 day	60%	63%	53%
1 week	83%	86%	73%
1 month	93%	94%	90%
6 month	100%	100%	100%

Table III
APPROPRIATE CLUSTER NUMBER DETECTION OF ROS IMPLEMENTED IN SPARTAN 3E FPGA.

FPGA	Maximum Average Silhoutte value	Appropriate Cluster Number	Status
1	0.7021	3	unused
2	0.7214	2	unused
3	0.7617	2	unused
4	0.7795	8	aged
5	0.7568	3	unused
6	0.7705	6	aged
7	0.7316	2	unused

VII. CONCLUSION

In this work, we discussed a supervised unsupervised methods (no golden data required) for recycled FPGA detection. A partially used, fully used, and spare LUTs in FPGA exhibits different path delay characteristics. Both methods characterize this variation with the help of an advanced ring oscillator design that covers all possible LUT paths. Simulation results shows that supervised method detects recycled FPGAs with high confidence after a certain usage time. The unsupervised detection approach is evaluated using simulation and silicon data from Spartan 3E FPGAs. Both methods perform quite well. Since it is based on machine learning and data labels, the supervised method is more appropriate when enough golden samples are available. The unsupervised method is better when the amount of golden samples is either very limited or non-existent. In future work, we plan on improving both approaches and testing on more FPGAs at different technology nodes.

REFERENCES

[1] M. M. Tehranipoor, U. Guin, and D. Forte, *Counterfeit Integrated Circuits: Detection and Avoidance*. Springer, 2015.

[2] M. Tehranipoor and J. Villasenor. The hidden dangers of chop-shop electronics. Available: http://spectrum.ieee.org/semiconductors/processors/the-hidden-dangers-of-chopshop-electronics.

[3] S. M. Trimberger, "Three ages of fpgas: A retrospective on the first thirty years of fpga technology," *Proceedings of the IEEE*, vol. 103, no. 3, pp. 318–331, 2015.

[4] Consumer electronics drive FPGA growth. Available: http://electronicspurchasingstrategies.com.

[5] Press room. ihs isuppli, april 2012. Available: http://press.ihs.com/

[6] U. Guin, D. DiMase, and M. Tehranipoor, "Counterfeit integrated circuits: detection, avoidance, and the challenges ahead," *Journal of Electronic Testing*, vol. 30, no. 1, pp. 9–23, 2014.

[7] H. Dogan, D. Forte, and M. M. Tehranipoor, "Aging analysis for recycled fpga detection," in *Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2014 IEEE International Symposium on*. IEEE, 2014, pp. 171–176.

[8] K. Huang, Y. Liu, N. Korolija, J. M. Carulli, and Y. Makris, "Recycled ic detection based on statistical methods," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 34, no. 6, pp. 947–960, 2015.

[9] X. Zhang, K. Xiao, and M. Tehranipoor, "Path-delay fingerprinting for identification of recovered ics," in *Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 13–18.

[10] U. Guin, X. Zhang, D. Forte, and M. Tehranipoor, "Low-cost on-chip structures for combating die and ic recycling," in *Proceedings of the 51st Annual Design Automation Conference*. ACM, 2014, pp. 1–6.

[11] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ics," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 22, no. 5, pp. 1016–1029, 2014.

[12] A. Amouri and M. Tahoori, "A low-cost sensor for aging and late transitions detection in modern fpgas," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*. IEEE, 2011, pp. 329–335.

[13] C. Leong, J. Semião, I. C. Teixeira, M. B. Santos, J. P. Teixeira, M. Valdes, J. Freijedo, J. J. Rodriguez-Andina, and F. Vargas, "Aging monitoring with local sensors in fpga-based designs," in *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*. IEEE, 2013, pp. 1–4.

[14] Y. Sato, M. Monden, Y. Miyake, and S. Kajihara, "Reduction of nbti-induced degradation on ring oscillators in fpga," in *Dependable Computing (PRDC), 2014 IEEE 20th Pacific Rim International Symposium on*. IEEE, 2014, pp. 59–67.

[15] E. A. Stott, J. S. Wong, P. Sedcole, and P. Y. Cheung, "Degradation in fpgas: measurement and modelling," in *Proceedings of the 18th annual ACM/SIGDA international symposium on Field programmable gate arrays*. ACM, 2010, pp. 229–238.

[16] S. Kiamehr, A. Amouri, and M. B. Tahoori, "Investigation of nbti and pbti induced aging in different lut implementations," in *Field-Programmable Technology (FPT), 2011 International Conference on*. IEEE, 2011, pp. 1–8.

[17] P. Sedcole and P. Y. Cheung, "Within-die delay variability in 90nm fpgas and beyond," in *Field Programmable Technology, 2006. FPT 2006. IEEE International Conference on*. IEEE, 2006, pp. 97–104.

[18] D. Lorenz, G. Georgakos, and U. Schlichtmann, "Aging analysis of circuit timing considering nbti and hci," in *On-Line Testing Symposium, 2009. IOLTS 2009. 15th IEEE International*. IEEE, 2009, pp. 3–8.

[19] S. Khan, S. Hamdioui, H. Kukner, P. Raghavan, and F. Catthoor, "Incorporating parameter variations in bti impact on nano-scale logical gates analysis," in *Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 158–163.

[20] A. Lesea and A. Percey, "Negative-bias temperature instability (nbti) effects in 90 nm pmos," *White Paper, 224*, 2005.

[21] A. Maiti, L. McDougall, and P. Schaumont, "The impact of aging on an fpga-based physical unclonable function," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*. IEEE, 2011, pp. 151–156.

[22] E. Stott, J. S. Wong, and P. Y. Cheung, "Degradation analysis and mitigation in fpgas," in *Field Programmable Logic and Applications (FPL), 2010 International Conference on*. IEEE, 2010, pp. 428–433.

[23] M. Majzoobi and F. Koushanfar, "Time-bounded authentication of FPGAs," vol. 6, no. 3, pp. 1123–1135.

[24] D. M. Tax and R. P. Duin, "Support vector domain description," *Pattern recognition letters*, vol. 20, no. 11, pp. 1191–1199, 1999.

[25] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt *et al.*, "Support vector method for novelty detection." in *NIPS*, vol. 12. Citeseer, 1999, pp. 582–588.

[26] G. Lee and C. D. Scott, "The one class support vector machine solution path," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2. IEEE, 2007, pp. II–521.

[27] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," 2003.

[28] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[29] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA., 1967, pp. 281–297.

[30] S. P. Lloyd, "Least squares quantization in pcm," *Information Theory, IEEE Transactions on*, vol. 28, no. 2, pp. 129–137, 1982.

[31] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[32] O. Mazhelis, "One-class classifiers: a review and analysis of suitability in the context of mobile-masquerader detection." *South African Computer Journal*, vol. 36, pp. 29–48, 2006.

[33] Synopsys. [Online]. Available: http://www.synopsys.com

[34] Y. Cao, T. Sato, D. Sylvester, M. Orshansky, and C. Hu, "Predictive technology model," *Internet: http://ptm. asu. edu*.