Hierarchical Fault Simulation for Mixed-Signal Circuits Using Template Based Fault Response Modeling

by

Nikhil Sagar Modala

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2024 by the
Graduate Supervisory Committee:

Sule Ozev, Chair
Krishnendu Chakrabarty
Seth Abraham

ARIZONA STATE UNIVERSITY

May 2024

ABSTRACT

The objective of fault simulation is to estimate the fault coverage of a given test input. Established fault models in the analog domain are based on detailed transistor-level netlists. Existing fault simulation tools inject and analyze fault responses at this level of detail. However, extending fault simulation to large circuits, especially when digital signals and/or frequency translation is involved, can be difficult due to the nature of simulations. Designers work with models at higher abstraction levels where simulations are more efficient. The goal of this paper is to bridge the gap between available transistor-level fault simulation tools, where fault simulation can be accurate, and behavioral abstraction levels, where simulation time can be shorter. This work aims to achieve this by judiciously adding various functional enhancements to individual functional blocks from a list of templates into their behavioral model until the responses at the two abstraction levels match. Transistor-level simulations are only limited to smaller functional blocks, where they are feasible, and individual fault responses are captured for behavioral simulations. Experimental results on the flash ADC (Analog-to-Digital Converter), show that accurate simulations can be achieved at a fraction of the simulation time.

ACKNOWLEDGMENTS

Throughout my journey completing my MS thesis, Professor Sule Ozev has been an exceptional advisor, providing unwavering direction and support that kept me motivated, even during the most challenging moments. Her consistent encouragement to think creatively during our frequent technical discussions, coupled with his insightful views, not only enhanced my research skills but also bolstered my confidence. Her mentorship has significantly contributed to my growth as a researcher, and I am grateful for his assistance throughout my MS journey. I would like to thank Tolga Aksoy, it was great working with you and learning more about data and machine learning and continuing the research ahead. I also extend my gratitude to Professor Bertan Bakkaloglu, under whom I have gained essential knowledge in analog electronics fundamentals, shaping my research interests and academic pursuits. Additionally, I am thankful to Professor Krishnendu Chakrabarty and Professor Seth Abhram for agreeing to be a part of my graduate committee and for their participation in my MS Thesis defense. Furthermore, I am thankful to Rubin Parekhji and Lakshmanan Balasubramanian from Semiconductor Research Corporation and Texas Instruments, India, for their invaluable insights and suggestions which leads to improvements to the overall project. I am deeply appreciative of my research group colleagues for their unwavering guidance and availability. Their support has been instrumental in navigating through intricate decision-making processes. I am grateful to my family and my friends for their support, which has been crucial in shaping my daily routine and keeping me motivated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Background

Analog circuits, which have been typically manufactured with mature technology nodes that are one or two generations older than the leading node, have caught up to their digital counterparts, leading to higher-performance analog devices and better integration. However, this leapfrogging also brought about higher defect rates for analog devices. Built-in self-test (BIST) and design-for-test (DFT) techniques are also becoming more prominent in the analog domain to tackle the challenge or low accessibility. The confluence of higher defect density and the trends of relying on internal measurements with additional uncertainty results in the need to quantify the test quality. Furthermore, stringent limits on defect levels from automotive and other safety-critical domains dictate that test quality be quantified. As a result, analog test patterns, regardless of whether they are specification-based, internally, or automatically generated, need to be evaluated in terms of their defect coverage. To achieve this goal, observed defects are modeled as faults in a way that can be simulated using existing circuit simulators. Fault models in the analog domain include hard faults (resistive opens and shorts) as well as soft faults (out-of-tolerance variations in process parameters) [1]. In most cases, this boundary is fuzzy since a good fraction of hard faults do not result in catastrophic failures but shift the performance of the device to an undesirable level.

A defect is an unwanted physical change in a circuit element or connection between circuit elements that are not within fabrication specifications for the circuit element or connection. A fault, on the other hand, is a way to model defects at a certain level of abstraction. Faults are majorly classified as hard and soft faults. Hard faults are all those faults that cause the circuit to fail catastrophically or cause large variations from the actual parameters. Soft faults, also called parametric faults, cause one or more performance parameters to be degraded.

A consensus has been reached in hard fault models, where an open circuit is modeled with a large resistor (1GΩ - 100kΩ) and a short circuit is modeled with a small resistor (1Ω - 100Ω) [2]. Analog fault simulation involves injecting faults and comparing the faulty circuit's response with the fault-free circuit's response. Most commercial electronic design automation tools include analog fault injection and simulation capabilities. The resistive hard fault model generates five potential faults per transistor (D open, S open, G open, DG short, SG short), and two faults per passive circuit component. This results in a rather large number of faults generated even for a small-scale analog circuit for evaluation. Conducting fault simulations for each possible fault in large circuits with thousands of transistors becomes computationally infeasible, limiting fault simulations to smaller circuits. Techniques, such as fault dropping and collapsing, have been introduced to reduce the number of faults to be simulated based on physical observation (e.g., D open and S open faults behave almost identically) [3], [4]. However, fault simulations for large circuits, such as a PLL, have remained challenging even after collapsing.

The typical analog circuit design process involves starting with a behavioral description using a high-level language like MATLAB or Verilog-A. This description is used to define system-level parameters as well as translate the system-level specifications to the specifications of individual design blocks [5]. Simulations at the behavioral level are much faster and suitable for running many fault simulations. However, behavioral models are often not conducive to fault simulations as changing behavioral parameters does not correspond directly to fault behaviors. Furthermore, faults can alter the behavior of circuit blocks beyond what is included in the behavioral models.

Hierarchical fault simulation plays a crucial role in the efficient and effective analysis of analog circuit designs by introducing a structured method of fault diagnosis that addresses the complexity of modern circuits. his approach provides a structured method of fault diagnosis that accounts for the complexity of modern circuits. By breaking down the simulation into levels of abstraction, from individual transistors to the entire system, engineers can precisely identify faults and understand their impact on circuit performance. The first stage of transistor fault simulation focuses on identifying potential faults at the most granular level, that is, in individual transistors. This meticulous scrutiny ensures that faults are not overlooked in the early stages of design verification, laying a solid foundation for subsequent analyses.

As ascending the hierarchy, fault response modeling acts as a critical link between identifying potential faults and comprehending their impact on the circuit's functionality. This important step entails crafting mathematical models or simulations that forecast how faults at the transistor level will impact the behavior of larger circuit blocks. These models play a pivotal role in gauging the severity of identified faults and determining whether they necessitate further investigation or can be disregarded based on their impact on the circuit's operation. By prioritizing faults with substantial functional implications, this phase optimizes design and verification resource allocation and streamlines the fault diagnosis process.

The hierarchical fault simulation process culminates in fault grouping, clustering, and pruning, followed by system-level simulation. Engineers can reduce the complexity and number of simulations needed to diagnose and address issues by grouping faults with similar characteristics or effects. This speeds up the fault identification process and enhances simulation efficiency by focusing on the most critical faults.

System-level simulation integrates all findings from lower levels of abstraction, offering a comprehensive view of the circuit's behavior under various fault conditions. This holistic approach ensures that the circuit design is robust and capable of meeting its performance requirements, even in the presence of faults, thereby enhancing the reliability and functionality of the final product.

Analog fault simulation is a method to predict circuit behavior under various faulty conditions. There are different techniques for performing fault simulation. Some of them are Inductive fault simulation, Monte Carlo simulation, Fault simulation, fault grouping/ collapsing, and Fault simulation using high-level models. The literature has combined and used these techniques to speed up the simulation time to simulate all the faults in an analog circuit.

This work aims to bridge the gap between transistor-level fault simulations where the fault models are more accurate and behavioral simulations that are faster and feasible for larger circuits. The circuit is partitioned into its building blocks, as guided by its behavioral model. Transistor-level fault simulations are conducted on the smaller building blocks. For each building block, fault behavior is captured through additional circuit components and functional transformations that are added to the behavioral model. Once the behavioral model is enhanced with the fault response models, system level simulations can be conducted using this enhanced behavioral model. To systemically extract the fault response model, I propose a small set of library templates that can capture a wide range of fault behaviors. I propose a two-tier algorithm to match the fault response obtained from SPICE simulations to that obtained with the enhanced behavioral model. I conduct experiments on two sample circuits. A flash analog to- digital architecture is chosen as a straightforward open-loop application and a phased locked loop is chosen as a closed loop application that typically takes a long time to simulate. I compare the results of transistor-level fault simulations to those of behavioral fault simulations. These experiments show that hierarchical fault simulations can produce accurate results at a fraction of the time that it would take to conduct the entire simulation at the transistor level.

1.2 Prior Work

Analog fault simulations are time-consuming due to the large number of faults that need to be evaluated and the lack of accepted simple fault models at the behavioral level [6]. While there is a large body of work on analog fault simulation, this research focuses specifically on facilitating high-level fault simulations using behavioral models captured from transistor-level fault simulations in this section.

Several basis functions have been explored in the literature to model the fault response of analog circuits. Chebyshev and Newton (CN) interpolating polynomials have been used to approximate nonlinear circuit behaviors using a single, state-space CN polynomial model [7]. The work mainly theorized to implement a single model that could eliminate the need for multi model fault modelling. They implement it in two parts: model generation and model evaluation. The paper explains how to obtain CN polynomials for one-dimensional (1D) and two-dimensional (2D) nonlinear functions using Chebyshev nodes and Newton divided difference (NDD) methods. They also describe how to employ CN polynomials in ss to obtain AMG-CNIP macro models, and how to tune the polynomial order using an automatic polynomial tuner (APT) algorithm. The ground-up non-linear model extraction suffers from a very large search space and cannot model fault effects that result in delays and phase changes.

In [8], the authors propose to link fault responses directly to the functional specifications of the circuit block, which then can be included in the behavioral model. The primary objective of this methodology is to calculate crucial test metrics, such as the probability of failure, fault coverage, and/or yield coverage for a given measurement under process variations. To achieve this, the authors propose generating models for both faulty and fault-free circuits. Once these models are in place, Monte-Carlo sampling can be utilized to simulate the conditions without the need for exhaustive Monte-Carlo simulations, which are typically more resource-intensive. However, this approach will not be able to capture faults that change the functional response of the circuit in addition to its performance.

A complex-field-based fault modeling approach is explored in [9] for linear time-invariant circuits. To enhance fault detection against measurement errors and parametric tolerance, the authors further introduce an improved fault model in a three-dimensional (3D) complex space, which achieves a higher fault detection ratio (FDR). Addressing the issue of fault masking observed in both 2D and 3D fault models, the authors propose a Design for Testability (DFT) method that involves adding redundant bypassing components in the Circuit Under Test (CUT). This method effectively isolates ambiguity groups, improving the fault isolation ratio (FIR).

Similarly, in [10], the authors explore the application of system identification techniques to fault diagnosis where fault responses are captured into the transfer function of the linear time-invariant circuit. The authors categorize traditional fault diagnosis techniques into two main approaches: Simulation Before Test (SBT) and Simulation After Test (SAT), with a focus on fault dictionary methods under SBT and noting their limitations in diagnosing soft faults caused by parameter deviation. The researchers introduced a novel fault parameter diagnosis method that utilizes system identification tools to construct nonlinear diagnostic equations based on multi-frequency testing. The method employs a Genetic Algorithm (GA) to search for global optimal parameters, aiming to diagnose module-level parameter faults. A fault module, as defined by the authors, comprises several components, and the method allows for the diagnosis of module faults by comparing estimated system parameters against their normal values.

In [11], the authors capture the changes to the impulse response of the circuit by various faults. It introduces a functional fault model where a faulty module is represented as a fault-free module in series or parallel with a fault module. They employ an iterative deconvolution technique to extract the impulse response of the fault module from the faulty response, significantly improving diagnostic resolution by separating the fault from the system function. This approach also enables the application of single-module fault tables in multi-module system diagnosis, simplifying the diagnostic process.

The authors [12] propose to capture fault responses of linear time-invariant circuits in terms of three basic steps, namely enhancements to Kirchoff's current and voltage equations, extraction of differential equations based on a signal flow graph model, and symbolically solving the system of equations . In [13], the authors propose to model the fault response effect on the output signal (for a given input signal) directly using the Hilbert transform, which captures both the magnitude and phase shifts but cannot capture nonlinear fault effects.

In reference [14], a defect simulator was established, incorporating defect models and assigning weights to each defect type model through sampling-based defect simulation. The approach addresses common inquiries within the analog design community regarding the selection of sample sizes, verification of sampling algorithm accuracy, and potential savings achieved through the application of sampling technologies. The methodology presented is grounded in a "trust but verify" approach, utilizing a commercial analog defect simulator, PrimeSim Custom Fault, across three industrial-sized analog circuits: a Phase-Locked Loop (PLL), a Successive Approximation Register Analog-to-Digital Converter (SARADC), and a Physical Layer (PHY) circuit, showcasing transistor counts ranging approximately from 3,000 to 30,000.

In [15], the authors employ a nonlinear autoregressive exogenous (NLARX) automatic model generation technique to perform high-level fault modeling and fault propagation. The extracted MATLAB models are then automatically transferred into VHDL-AMS language format. This approach can handle non-linear behavior and time delays. However, without any guidance from the existing circuit model, the search space is extremely large, making fault response extraction complex.

In [16], the authors aim at capturing fault behavior into behavioral models (VHDL-AMS) using DC analysis sweep and macro models of passive (resistors, capacitors, diodes) devices in addition to ideal voltage and current sources. The limited nature of circuit enhancements makes it difficult to capture a wide range of faulty behaviors. Similarly, in [17], the aim is to extract fault responses at the block level and propagate this information to the system level using behavioral simulations. The extraction of this model is manual, relying on the designer for this task. In this work, need to build a fault response model that aim to capture a wide range fault behavior and to define a two-tier algorithm to extract fault response models based on the templates and reduce the simulation time.

1.3 Research Goals/Scope

- To bridge the gap between transistor-level fault simulations where the fault models are more accurate and behavioral simulations that are faster and feasible for larger circuits.
- To enhance fault response model templates to capture a wide range of fault behaviors for analog building blocks and reduce the simulation time taken circuit level fault simulations.
- To provide a two-tiered algorithm to extract fault response models based on the templates, namely global search for a coarser level of model fitting and local search or fine tune modelling.
- To demonstrate the accuracy of the fault simulation on circuit response for a Analog to Digital converter circuit.
- To implement a unsupervised learning model that could further reduce the simulation time at the system level

Chapter 2

METHODOLOGY

The overview of the proposed methodology is shown in Figure 1. The process starts by exploring the existing behavioral model of the mixed-signal circuit that has already been partitioned into circuit blocks (or modules). Our goal is to simulate the transistor-level faults for each block separately and capture their effect into a complete fault response model to enable system-level simulations. For each of the blocks identified in the behavioral model, I modify the netlist to inject predefined faults.

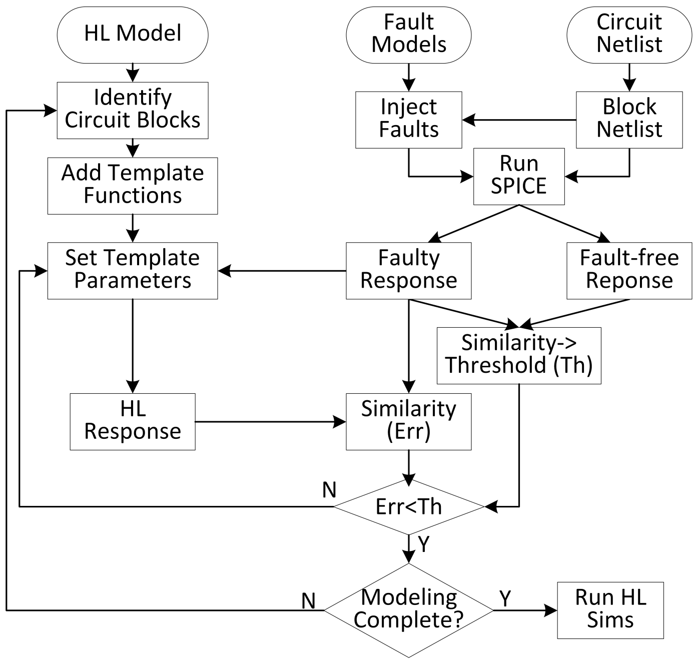

Figure 2.1: Overview of the Proposed Hierarchical Fault Simulation Approach

This step is identical to the offerings of existing commercial tools and in compliance with the majority of the prior work. Perform SPICE level simulation on the fault-free circuit to gather required results. Utilizing the same simulation testbench, then generate responses from a circuit with introduced faults. These faulty responses serve as the foundational ground truth for the High-level model. In the high-level model, identify the circuit block based on established behavioral model.

After modifying the netlist, conduct SPICE level simulations on both the pristine and fault-injected circuit configurations, using the same simulation testbench to ensure consistent results. I meticulously record the responses elicited from the circuit under fault conditions, which serve as the empirical basis for calibrating the high-level model. Then employ a fault response template library that seamlessly integrates with the circuit's behavioral model. By strategically inserting fault characteristic templates such as non-linearity, delay, or hysteresis at key points in the circuit block, our methodology facilitates an unparalleled level of fault response accuracy and model fidelity.

To refine the high-level model, utilize a dual-level parameter search approach guided by the objective of aligning the model's response with that observed in SPICE simulations. Our meticulous alignment process relies on two key similarity metrics - Dynamic Time Warping (DTW) and Euclidean Distance (ED) - each chosen for its ability to capture the essence of the behavioral discrepancy between fault-free and fault-induced circuit operations. By adopting these metrics and a strategically determined threshold reflective of the fault's severity, can precisely tune our search for optimal model fitting parameters. Once tailored the model for each fault and circuit block, can integrate it into the behavioral simulator to transition to system-level fault simulations. This approach offers a holistic view of the circuit's performance under a spectrum of fault conditions, promising enhanced reliability and fidelity in the design and assessment of mixed-signal circuits. Our methodology's strength lies in its rigorous and systematic approach to fault simulation.

2.1 Fault injection

Faults in a transistor can broadly classified into hard faults, which are opens and shorts between each terminal of the transistor and parametric/global faults are those which shift in one or more parameters such as length, width and threshold voltage for all of the circuit. To inject both faults, extract the circuit level netlist from the circuit designed. Now to inject shorts insert employ a 10Ω resistance while more several faults (1 mΩ - 20 kΩ) have also been explored in the literature. Drain and source-open defects are modeled with the insertion of a 1 MΩ resistance although more severe faults (1 MΩ - 1 GΩ) have been explored in the literature [4], [19].

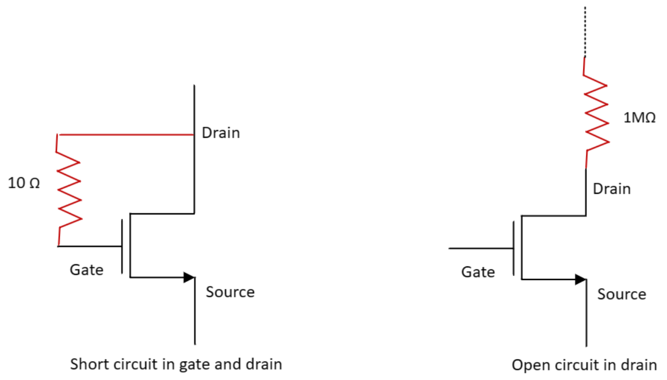

Figure 2.2: Fault Model for Open and Short Circuit Faults

The gate-open defect is treated as special since it does not draw any DC current. For gate-open faults, terminate the gate connection (to ground, negative supply, or positive supply, depending on the transistor type) through a 1 MΩ resistor. This ensures the transistor remains in the off state. The global threshold and length faults are modeled using the values indicated in the process design kit (PDK). In the experiments, use the 65nm kit. 50% deviation in threshold voltage and length are simulated as global parametric faults. Multiple fault instances have been explored in the literature [18]. However, injecting single faults in our experiments since this does not change the hierarchical fault simulation approach. These opens and shorts are injected to the netlist one by one to the fault free netlist. Multiple fault instances have been explored in the literature

Manually implementing fault models is often time-consuming and impractical. Our team has developed a Python script that simplifies this process by systematically manipulating the original netlist. This script can insert both shorts and opens, and simulate global faults with precision and speed. To mimic a short circuit condition, use a resistor to bridge two terminals. Conversely, replicate open faults with a large resistor, typically 1M Ohm, between a transistor terminal and its corresponding original net. The approach address global faults by adjusting critical parameters such as the transistor's length and threshold voltage, reducing them to 50% of their original values. All modifications are meticulously cataloged in a revised netlist, which is then ready for comprehensive SPICE level fault simulation.

This foundational work paves the way for fault simulations to be executed automatically, a task that has traditionally been time-consuming when performed manually. The introduction of an OCEAN script specifically tailored to the Cadence environment represents a significant breakthrough in this process. This script encapsulates all necessary Analog Design Environment (ADE) simulation parameters, such as temperature settings, simulation duration, time steps, and the nature of the simulation (i.e. DC or transient). It also specifies the output requirements, including the data to be saved and any relevant output equations. The OCEAN script serves as a conduit that seamlessly translates the ADE-defined simulation parameters into an automated and repeatable process, eliminating the need for manual intervention and greatly accelerating the fault simulation workflow.

The culmination of this process is the generation of output data in a CSV format, meticulously organized for ease of analysis. This not only facilitates a straightforward comparison between the simulated responses of fault-free and fault-induced circuits but also enhances the accessibility of the data for subsequent high-level modeling efforts. By automating the fault injection and simulation processes, this not only expedite the evaluation of potential circuit failures but also elevate the precision of our fault analysis, enabling a more robust and reliable design process for mixed-signal circuits. This methodological innovation represents a leap forward in the field of circuit design and analysis, promising significant improvements in both the efficiency and efficacy of fault simulation procedures.

2.2 Fault model templates

Our approach to modeling fault responses exists at the boundary of system-level description and circuit-level description. Fault responses are obtained by running transistor-level simulations in the SPICE environment. This library, designed for quick fitting while capturing a wide range of fault effects from non-linear phenomena to temporal variations like frequency shifts and hysteresis, enables us to replicate the intricate behaviors of circuit failures. Through a detailed mapping process, translate these failures into quantifiable model adjustments, allowing for precise identification and targeted model modification to mirror fault perturbations. There can also be potential in further enhancing our methodology through advanced techniques such as fault collapsing and ordering, aimed at streamlining simulations and improving efficiency. However, these are areas for future exploration.

Polynomial Hash Function:
If the analog or mixed-signal circuit has nominally non-linear behavior, this will already be captured in the fault-free behavioral model. For instance, a comparator is a mixed-signal device with a highly non-linear large signal behavior. The ideal comparator model captures this non-linear behavior. Thus, for our modeling approach, captures additional non-linearity that is not present in the existing fault-free model. This relaxes the modeling need, and can limit our modeling approach to polynomials up to 13th order (following most non-linear distortion models [20]) although most non-linear effects can be modeled with a third order polynomial [21]. The polynomial hash function can be applied to the input or the output of any functional block.

$$Y = c_0 + c_1X + c_2X^2 + \dots \tag{2.1}$$

Baseline Delay:

An important fault effect to model is the delay that it may induce in the response of the circuit. While it is difficult to capture the delay with a ground-up mathematical model, it is fairly straightforward to add it to the behavioral model as a block with a variable delay amount.

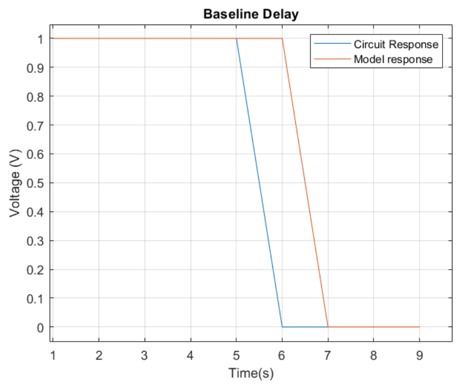

Figure 2.3: Baseline Delay Function

Noise: Some of the fault effects increase the noise level in the circuit. Noise can be added as a Gaussian random variable into the signals at the input or output of the functional block. In some cases, noise may need to be filtered and/or upconverted to higher frequencies to model band-limited noise effects.

RC Delay, Ringing, and Oscillatory Behavior:
Some faults cause changes in the rise time or fall time of the circuit or limitations to the slew rate of the signals. These effects can be captured with an RC circuit at the input or the output of the functional block. Ringing behavior can be modeled using a second-order RC circuit and oscillatory behavior can be modeled with a positive feedback circuit.

Hysteresis:
Faults may cause hysteresis in circuit behavior where the response is dependent on whether the input (or output) signal is rising or falling. This effect can be modeled using a derivative block that controls the output circuit parameters (such as delay,

RC values, etc.). In effect, the circuit is transformed into two distinct embodiments where the correct version is selected based on the derivative of the input signal (falling or rising).

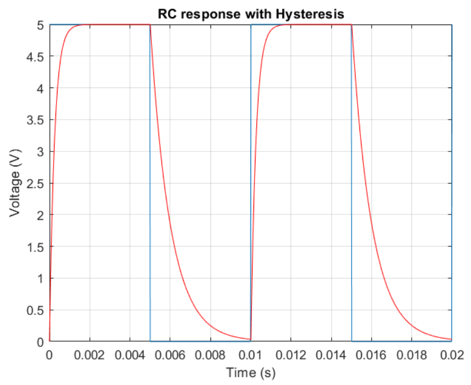

Figure 2.4: RC and Hysteresis Function

Frequency Shift:

For a sub-class of circuits, such as oscillators and phase-locked loops, faults may alter the operating frequency of the circuit. This can be captured by a shift in the frequency domain or ideal mixing and filtering in the behavioral model.

2.3 Dynamic Accuracy Setting

In order to enhance the model's flexibility and responsiveness, it is crucial to implement a dynamic threshold setting technique that can refine the accuracy of our simulated fault models. By leveraging the natural variability of faults, ranging from those with significant impacts to those with minimal effects, this innovative approach enables the model to intelligently adjust its precision based on the observed severity of the fault response. At the core of our adaptable methodology lies the crucial notion of similarity error. This metric is derived from a thorough comparison of the flawed circuit's response, obtained through rigorous transistor-level fault simulations, and the unblemished, error-free response. By employing Dynamic Time Warping (DTW) or Euclidean distance measurements, this comparison generates a measurable gauge of the variance between the anticipated and real-world circuit behaviors. The similarity error holds great significance as it acts as a reliable indicator of the model's fidelity. It helps to adjust the accuracy threshold based on the magnitude of the error. In cases where the similarity error exceeds predefined limits, signaling a significant divergence that can be deemed catastrophic, the model takes a pragmatic approach of reducing its demand for precision. This acknowledgement of catastrophic failures as beyond the scope of detailed accuracy highlights a strategic decision to allocate computational resources more efficiently. The focus is on the nuances of faults that subtly alter the circuit's performance, to ensure optimal utilization of resources.

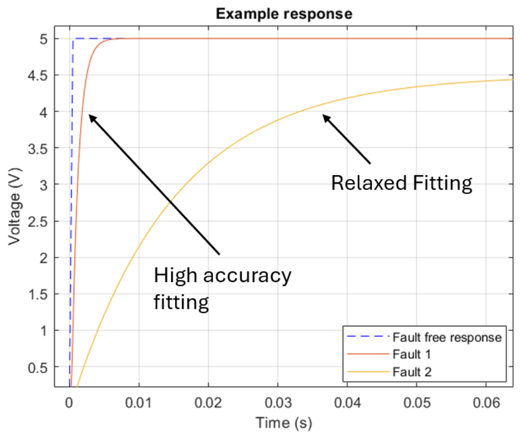

Figure 2.5: Ideal Response and Two Faults Response

An excellent way to demonstrate this concept is by comparing different responses to faults. The response of a fault-free scenario is typically step-like and serves as a reference point for evaluating the effects of faults. Fault 1, which settles within a tolerable timeframe but with noticeable behavior, is an example of a fault that is significant but does not prevent the circuit from meeting its specifications. On the other hand, Fault 2, which has a prolonged and pronounced settling period, causes the response to deviate far beyond acceptable limits, making it a catastrophic fault that severely impacts the circuit's functionality. The strategic use of model accuracy selection, based on the severity of similarity errors, demonstrates a sophisticated approach to simulation. This technique optimizes time and computational resources by focusing efforts on analyzing subtle yet critical faults that impact circuit performance. By adjusting the model's precision dynamically in response to each fault's unique characteristics, this methodology enhances simulation efficiency while also generating more insightful and practical fault analysis. Ultimately, this paves the way for more nuanced and effective electronic circuit design.

2.4 Fault Response Model Fitting

The process of developing model templates for fault simulations in analog circuits involves a particular approach to parameter selection and fitting, where the complexity of the task can worsen significantly due to the expansive search space for these parameters. For instance, when modeling the rise time or fall time of a signal and selecting the value of capacitor the parameters involved span several orders of magnitude. This vast range complicates the optimization process, as finding the optimal set of parameters that accurately reflect the circuit's real-world behavior under fault conditions requires navigating through a significantly enormous search space. The fact that these elements are interdependent adds to the complexity of this process. Unlike variables that can be optimized in isolation, these fitting parameters influence the circuit model's behavior in a collective manner.

This interlinked optimization problem is computationally demanding, especially when applied to all the possible fault scenarios that an analog circuit could experience. The essence of hierarchical fault simulations is to streamline the simulation process, allowing for efficient analysis of faults at various levels of the circuit design. However, the high computational complexity involved in functionally fitting multiple interleaved parameters threatens to undermine this efficiency. The hierarchical approach's anticipated benefits may be undermined by the time and processing resources required for the thorough search necessary for each fault model. Strategies such as employing machine learning algorithms for intelligent parameter estimation, simplifying the model complexity without sacrificing accuracy, or developing more efficient search algorithms could prove pivotal in preserving the hierarchical fault simulation's viability as a tool for robust and efficient circuit design and analysis.

In order to ensure accurate and efficient detection and characterization of circuit faults, a commonly used two-phase technique in machine learning is employed to construct the algorithm for fault response modeling. The first metric used is Dynamic Time Warping (DTW)[22], which assesses the similarity between two time series data by allowing for the alignment of sequences that may vary in time or speed. DTW's ability to precisely record the subtle differences and similarities between a circuit's fault-free and faulty responses is due to its capacity to adjust to these variances by "warping" the time axis until an ideal match between the two sequences is established. This method directly tackles the challenge of time-based discrepancies, ensuring that the analysis remains robust even when direct comparisons might be complicated by temporal shifts in the signal responses. The figure shows the difference between ED and DTW distance for the same responses

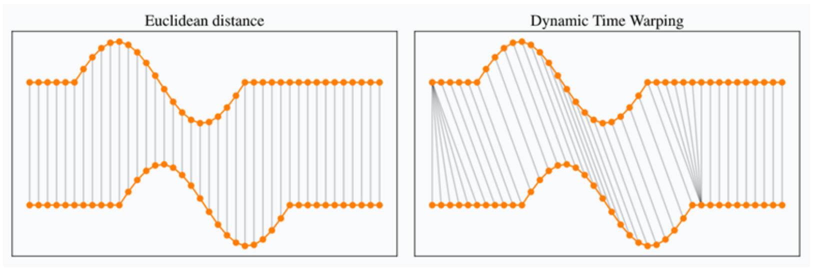

Figure 2.6: Comparison between DTW and ED Metrics

The Euclidean Distance (ED) is a valuable metric that provides a different view of the data by measuring the distance between two points in multi-dimensional space. Specifically, it measures the straight-line distance between corresponding time series data points. Unlike DTW, ED is most effective when time series are aligned, and the primary objective is to measure the degree of divergence between them. It does not factor in temporal shifts. By leveraging both DTW and ED, the algorithm gains a comprehensive understanding of fault impacts, incorporating insights from both time-warped and direct comparisons.

To search for the model parameters efficiently, developed a two-phase algorithm as shown in Figure 2.

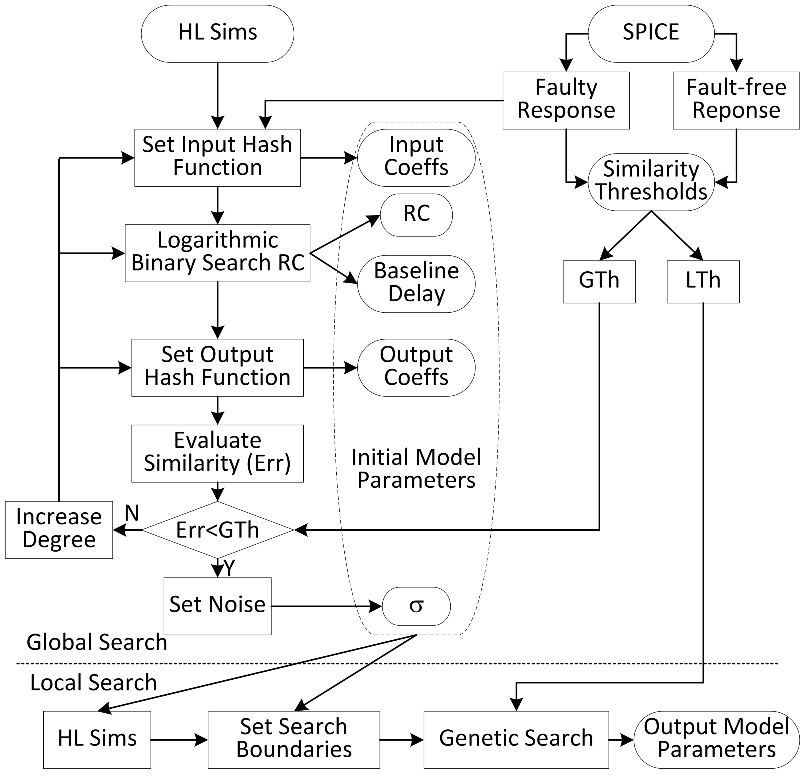

Figure 2.7: Search Algorithm

Realizing that adjusting template functions individually within a fault model can create a cumbersome search space that hinders efficiency, a new approach is necessary. As a result, a two-tiered search algorithm has been developed that combines both global and local search strategies to tackle the complexities involved in optimizing template function values. This advanced algorithm is engineered to effectively navigate through extensive parameter space, identifying the best configurations that accurately reflect the circuit's faulty behaviors. The global search begins by casting a wide net to identify promising regions within the vast search landscape, aiming to pinpoint potential areas where optimal solutions can be found. After conducting a thorough global search, the local search phase delves deeper into the identified areas, employing a focused and precise technique for parameter optimization. This phase is essential for perfecting the template functions to closely match the observed faulty behaviors of the circuit, as determined by rigorous fault simulations at the transistor level. The meticulous refinement during the local search phase is guided by the principle of attaining high fidelity in the model's response, dynamically adjusting accuracy in response to the severity of faults as indicated by the similarity error criteria.

Our algorithm simplifies the search process, minimizing the intricacies of the search space. It utilizes a systematic methodology for fault modeling, blending extensive, exploratory phases with precise, detail-oriented ones. This strategy guarantees that the model precisely captures the intricate behaviors of defective circuits, amplifying the usefulness and importance of the simulations. Our algorithm furnishes a sturdy framework for identifying and comprehending circuit faults with a versatile and adjustable accuracy threshold.

In this phase, our global threshold (GTh) for model fitting is set to be 10% of the baseline similarity error, a metric obtained from a comparison between the faulty and the fault-free responses of the circuit. This crucial step guarantees that the model's precision is perfectly tailored to the severity of the fault-induced deviations, creating a clear benchmark for accuracy. The algorithm then systematically explores the template function values, beginning with the polynomial function coefficients at the input. This initial stage is essential for capturing the circuit's nuanced pre-comparator behavior, laying the groundwork for a comprehensive modeling of the fault impact.

Next, utilized a logarithmic function of the RC (resistance-capacitance) and delay function variables, taking into account their interconnection and expansive operational scope. Employing this logarithmic tactic enables swift exploration of the search space, from the leisurely milliseconds of gradual dynamics to the swift picosecond transitions, guaranteeing a thorough analysis for the most desirable values. Subsequent to the RC and delay parameterization, our algorithm turns its attention to the output, ascertaining the coefficients for the output polynomial hash function to accurately mirror the comparator's reaction under fault conditions.

When the derived template function values reach the required accuracy, the global search phase is considered successful, and the values are finalized. However, if the accuracy benchmark is not met, an iterative refinement process follows, including an increase in the order of coefficients and a renewed search. This iterative refinement shows the algorithm's ability to adapt and ensure precision. It helps capture the circuit's faulty behavior and meets accuracy requirements. This diligent process forms a solid foundation for a fault model that combines high fidelity with practical insight into circuit behavior.

During the second phase of our algorithm, narrow down the search area based on what was learned in the first phase. Use 50% range of values found in the initial phase to focus our search. This helps us to fine-tune the parameters more efficiently. Using a genetic algorithm, can fine-tune the values for the template function in this precise search area.

The genetic algorithm (GA) is a powerful optimization and search technique inspired by the principles of genetics and natural selection. It mimics the process of biological evolution by starting with a population of potential solutions to a given problem, each represented by a set of "genes" or parameters. These solutions then evolve over successive generations, with the algorithm applying natural operations such as selection, crossover, and mutation. Fit individuals are favored through selection, allowing them to pass their genes to the next generation. Crossover mixes the genes of parent solutions to produce offspring, introducing new solution variants. Mutation introduces random changes to individual genes, which prevents the algorithm from becoming stuck in local optima and encourages exploration of the solution space. Through these iterative processes, the GA aims to evolve solutions towards increasing levels of fitness, ultimately converging on an optimal or near-optimal solution to the problem at hand.

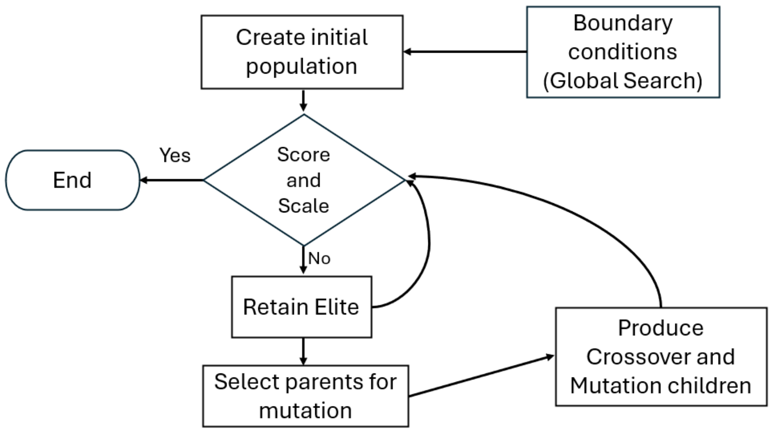
Create initial population
Boundary conditions (Global Search)
Yes
End
Score and Scale
No
Retain Elite
Produce Crossover and Mutation children
Select parents for mutation

Figure 2.8: Genetic Algorithm

Genetic algorithms (GAs) excel at solving intricate optimization problems that stymie traditional search and optimization techniques. These problems may be challenging due to their size, non-linearity, multi-modality, or other characteristics. GAs search a population of solutions instead of a single point, making them adept at finding global optima and resistant to the pitfalls of local optima traps. The algorithm's flexibility and adaptability make it a potent tool for a wide range of applications, including engineering design, scheduling, machine learning, and artificial intelligence. GAs' evolutionary approach allows them to navigate vast and complex search spaces efficiently, making them invaluable for tackling problems where the search space is poorly understood or difficult to navigate using conventional methods.

This genetic algorithm uses an iterative and adaptable approach to find optimal configurations that meet the model's accuracy standards. The local search phase demonstrates the effectiveness of evolutionary principles in fault modeling. The resulting fault model is precise and informative, meeting strict accuracy criteria and providing valuable insights into the subtle impacts of marginal faults on circuit performance. This process combines a targeted, limited search space with the adaptive, iterative capabilities of the genetic algorithm, resulting in a efficient and powerful model.

2.5 Fault Clustering

In our pursuit of refining the model's efficiency and achieving a reduction in simulation time, have strategically proposed the adoption of a fault clustering algorithm. This approach aims to improve our current process by adding a complex data grouping/clustering layer that depends on the template parameters values taken out of our fault response model. Our goal is to streamline the analysis process by classifying the collected data according to these core factors using the k-means method. The incorporation of k-means into our fault analysis model takes advantage of the algorithm's ability to handle large, complicated data sets with resilience to guarantee a more precise, accurate, and speedy simulation process.

The K-means clustering algorithm functions by first identifying the desired number of clusters, K, utilizing methods such as the Elbow Curve. The algorithm then selects initial centroids either randomly or through more informed techniques like k-means++. Data points are subsequently assigned to the nearest centroid based on Euclidean distance, effectively grouping them into clusters. After assignment, the centroids are updated as the mean of all points within their cluster, refining their position to better represent the cluster center. This process of point assignment and centroid update repeats until convergence is achieved. Ultimately, K-means efficiently organizes data into distinct clusters, with the final centroids accurately embodying the centers of their respective clusters, thereby enabling clear partitioning of data based on similarity.

Implementing fault clustering can significantly streamline simulation time by grouping similar faults, reducing the number of individual simulations required. While some steps of the algorithm are undoubtedly essential and cannot be replaced or reduced, would like to focus on reducing the more mundane simulation time-consuming parts.

Chapter 3

IMPLEMENTATION

To validate the effectiveness and precision of our hierarchical fault modeling methodology, selected an analog-to-digital converter (ADC), focusing specifically on a flash ADC due to its widespread use and simplicity. By applying our hierarchical fault modeling approach to showcase the method's accuracy in modeling faults. The evaluation process involves a detailed comparison of the fault model's accuracy at the system level, utilizing MATLAB for visualization, analysis and simulation. This aims to show notable improvements in fault detection accuracy and simulation process efficiency by comparing the results of our hierarchical modeling approach with conventional fault simulation techniques. Our dedication to improving fault analysis techniques is demonstrated by our all-encompassing validation strategy, which ultimately aims to raise the performance and dependability of analog-to-digital conversion systems and, consequently, the entire area of electronic circuit design.

3.1 Flash ADC

Flash analog-to-digital converters, also known as parallel ADCs, are the fastest way to convert an analog signal to a digital signal. Flash ADCs are suitable for applications requiring very large bandwidths. It is one of the simplest and fastest ADC architectures in literature. Flash ADC consists of string of resistors and high-speed comparators. The reference voltage is the FSR voltage divided by N resistors, which would create a voltage divider and the smallest division is called LSB.

$$1\,\text{LSB} = \frac{\text{FSR}}{2^N} \tag{3.1}$$

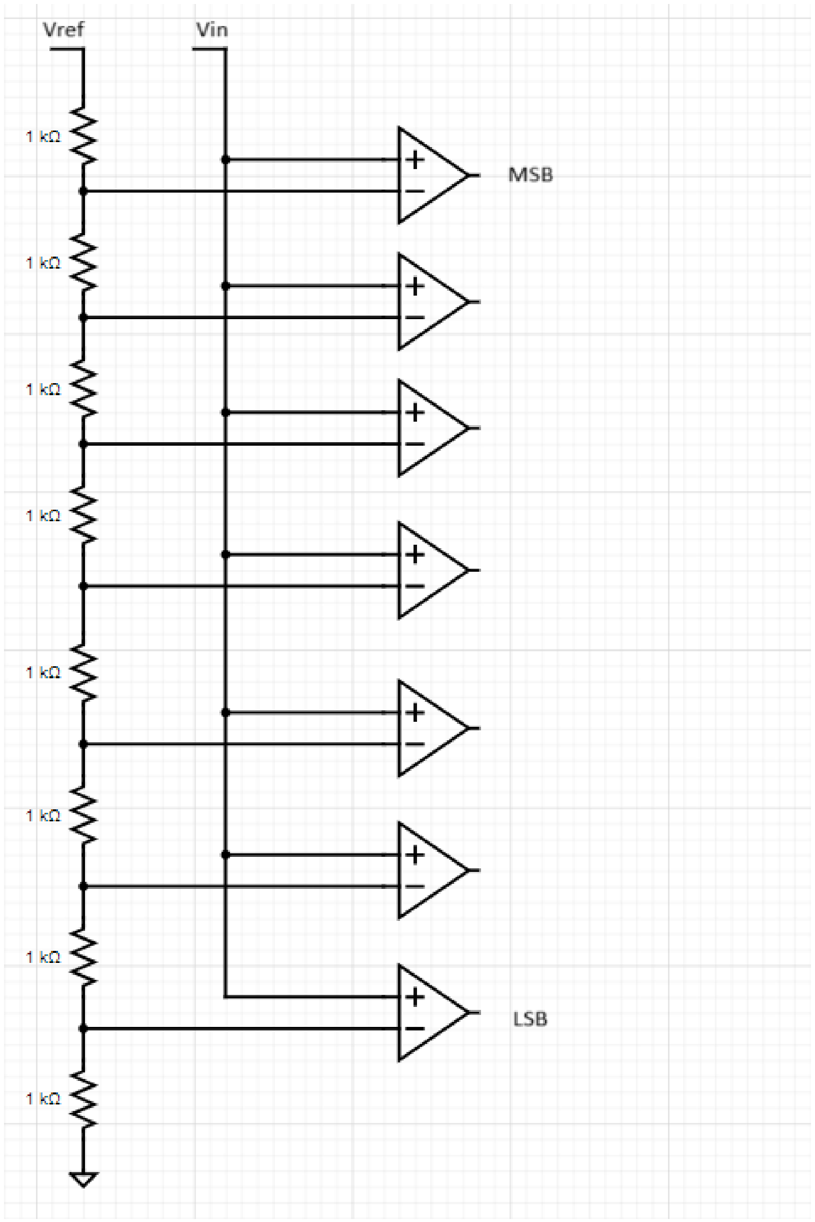

Figure 3.1: Flash ADC

The gray code output, get after all the comparators are then passed through another digital circuit which converts gray scale into 3-bit digital output. In this project, considering the output at the end of to be final output as after that it can have some digital logic gates which are not considered.

3.1.1 Comparator

A comparator is a device that compares an input voltage with a reference voltage and indicates which is greater by means of a digital output. If positive terminal is greater than negative terminal, then the comparator output will be 1. For inverted comparator, the working is exactly opposite, then the output will be 0. The comparator amplification is very high gain, as it should be able to amplify a small difference. These comparators are essential components in various applications from level detectors, oscillators to ADC to make quick decisions based on small input changes.

Comparator architecture:

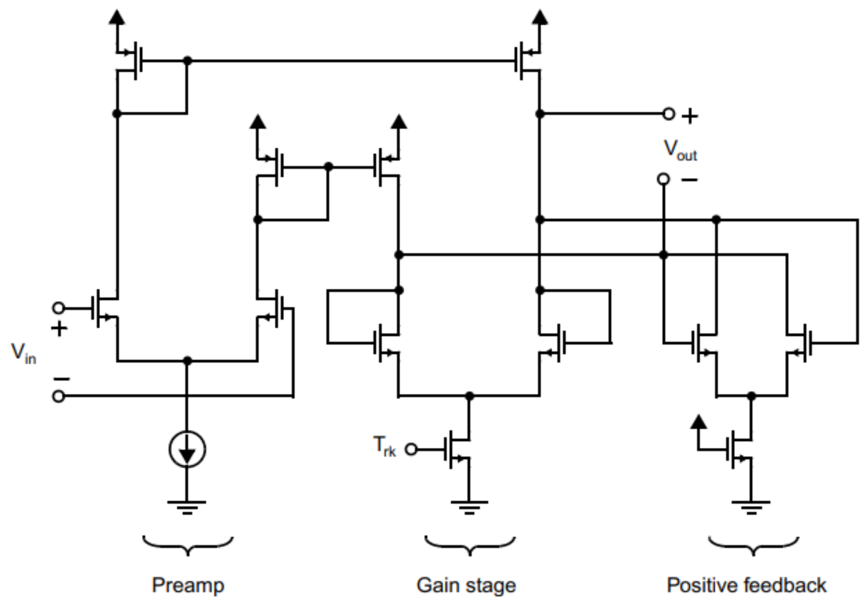

Figure 3.2: Simple Comparator Architecture

The comparator designed in this project is a two-stage comparator that has a preamplifier and a positive feedback track and hold latch stage. The preamplifier is a like differential input pair which converts the voltage into current. This current is then mirrored into the next stage which consists of a cross coupled pair which is positive feedback and diode connected load. After this there is a self-biased differential

Operational transconductance amplifier which converts differential input into a single ended and based on which input is larger, the output will be lower or higher value. Then there is an inverter which converts this into a digital signal.

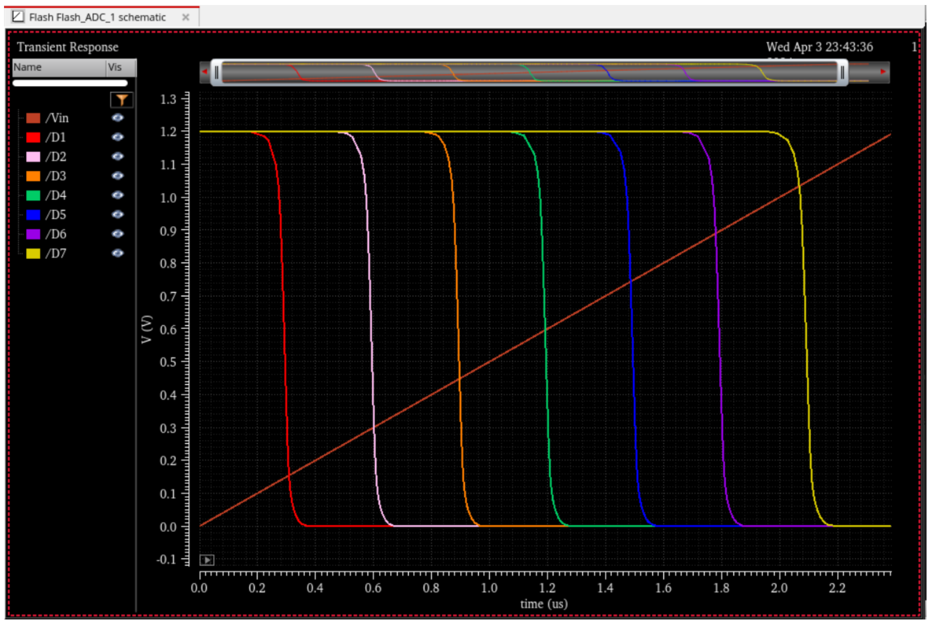

Figure 3.3: Ideal Response of 3-Bit Flash ADC

During the transient analysis with a ramp input, we observe the ideal response of a 3-bit Flash ADC. Each comparator trips correspondingly as the ramp input increases.

Chapter 4

EXPERIMENTAL RESULTS

The flash ADC consists of a resistor string and a string of comparators. Since the resistor string is already modeled at the behavioral level, the module to model here is the comparator.

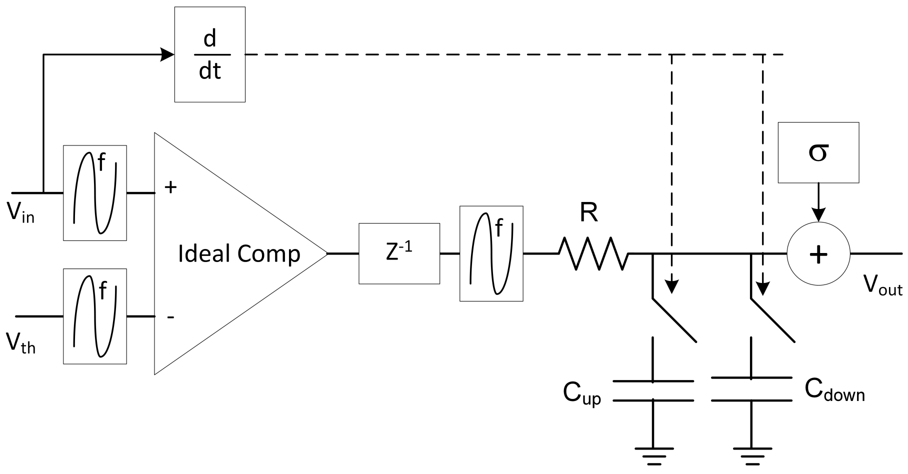

Figure 4.1: High-level Model of the Comparator

The high-level model for the comparator is shown in Figure 3. For each fault, the model parameters are computed separately and no fault collapsing or ordering has been employed. Extraction of the fault model takes on average 5 seconds (some faults have a higher accuracy threshold compared to the others). Overall, 95 hard and parametric faults are evaluated for the comparator. For all evaluated faults, the input hash function has only one parameter, a DC offset. A first-order RC circuit has proven to be sufficient to model the rise time and fall time delays (no faults resulted in ringing or oscillatory behavior). The output hash function is of first degree despite the highly non-linear behavior of the comparator, which is already captured in the ideal model. Most faults have resulted in hysteresis behavior where the rise and fall time responses are different depending on whether the output is rising or falling. This can be included in the model in multiple ways. The derivative block has been chosen for its general applicability, although the ideal response could also serve this purpose effectively.

At the comparator level, fault response similarities are evaluated as DTW or ED metrics. The thresholds are adjusted per fault based on the difference between faulty and fault-free responses. After the fine-tuning phase of the fitting algorithm, each fault response is modeled with a worst-case error metric. To analyze the fault modeling effects in more detail, have grouped the faults into five categories: (a) faults that shift the offset of the input hash function (IHF), (b) faults that shift the output hash function offset (OHF), (c) faults that shift the output hash function gain (OG), (d) faults that shift the baseline delay (D), and (e) faults that shif the RC value at the output (RC). A fault can be included in more than one category since it can change multiple parameters. For each category, RMS (root-mean-square) was calculated of the similarity error, which is shown in Table 4.1. Table I shows that faults in each category are modeled with similar efficacy and there are no significant differences between the fault categories.

Table 4.1: Similarity Errors for Different Fault Categories

Fault Cat. Metric	**Baseline**	**IHF**	**D**	**OHF**	**OG**	**RC**
DTW	1800	8.4	5.8	16.7	0.2	7.2
ED	500	4.0	4.6	4.8	0.0	4.6

Next, evaluated the accuracy of the fault simulations at the system level using a 3-bit ADC as a demonstration. To enable full-scale transistor-level simulations, have chosen to limit the resolution of the ADC to three bits. For the 3-bit ADC, there are 672 faults to simulate, where the majority of them are within the 7 comparators.

The ADC is simulated for its static parameters, including DNL, INL, gain error, and offset error. Some faults result in very large deviations in these parameters whereas some faults result in only marginal deviations. For a more detailed analysis of the accuracy, have grouped the faults into the same aforementioned categories. These represent two of the fault responses, one that causes catastrophic shifts and another that causes marginal shifts.

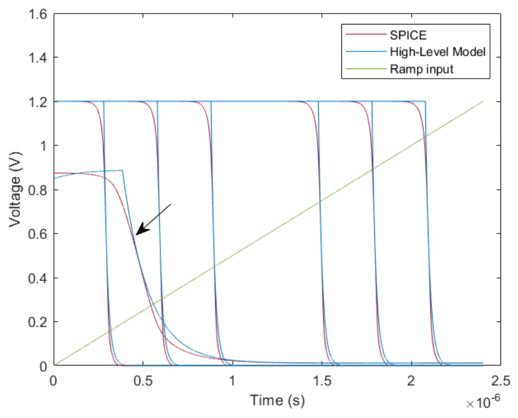

Figure 4.2: Fault Response with Catastrophic Shift

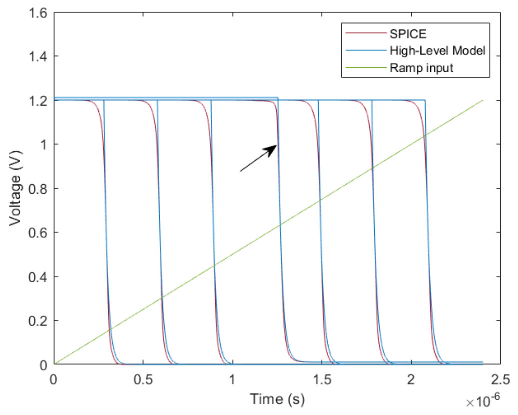

Figure 4.3: Fault Response with Marginal Shift

Table 4.2 shows the RMS errors (evaluated over each fault category) of determining DNL, offset error, and gain error with a ramp input, for all fault categories. DNL and offset errors are with reference to 1LSB; the gain error is with reference to 1. Hierarchical fault simulations and full transistor-level fault simulations are in agreement with the maximum error for hierarchical fault simulation being less than 1% of 1 LSB. As an example, Figure 4 shows the simulation results for 2 randomly selected faults.

Table 4.2: RMS Error in Estimating DNL, G and V_{off}

Fault	**DNL1**	**DNL2**	**DNL3**	**DNL4**	**DNL5**	**DNL6**	**G**	**Voff**
IHF	2e-4	7e-4	4e-4	3e-4	3e-4	2e-4	3e-6	0
D	8e-4	6e-3	2e-3	1e-3	1e-3	5e-4	2e-4	2e-6
OHF	1e-3	6e-3	2e-3	2e-3	2e-3	8e-4	9e-6	9e-7
OG	6e-4	1e-2	8e-4	1e-3	9e-4	1e-4	2e-6	0
RC	8e-4	8e-3	2e-3	1e-3	1e-3	4e-4	1e-5	1e-6

4.1 Simulation Time

Table 4.3 compares the computation times between the hierarchical and full transistor-level simulations for three circuits: a 3-bit flash ADC and an 8-bit flash ADC. Computation times for the hierarchical simulations (HL) include transistor-level simulations for each block, model fitting time, and behavioral simulations whereas computation time for full transistor-level simulations (SP) includes only the SPICE simulation time. The time savings for the hierarchical simulations increase with the increasing complexity of the circuits.

Table 4.3: Fault Simulation Time Comparison

	3b ADC	8b ADC
SP	14s	30min
HL	10s	15min
Reduction	1.4x	2x

To implement the fault clustering used the template model values as data points. Since the duration of transistor-level simulations is constant and determined by the capabilities and limitations of the simulation tools, the focus shifts towards optimizing the remaining components of the simulation process. Model fitting and the search for optimal template values consume the bulk of the time, accounting for approximately 50% of the total simulation time. While attempts to streamline this process could potentially expedite simulations, they risk compromising the model's accuracy, leading to significant discrepancies between the simulated and actual circuit behaviors. Meanwhile, performance parameter calculations, representing about 20% of the total time, emerge as a viable option for optimization.

Our dataset consists of five systematically processed and scaled variables, with a data range spanning from 1 to 10-15. Employed several normalization techniques to facilitate more effective grouping and extensively evaluated their performance, including MinMax Scaler, MaxAbsolute Scaler, and Standard Scaler, among others. After careful consideration, selected the MinMax Scaler and Standard Scaler as the best choices based on the number of clusters and error (elbow curve), ultimately choosing the Standard Scaler for its superior performance with respect to cluster cohesion and error metrics.

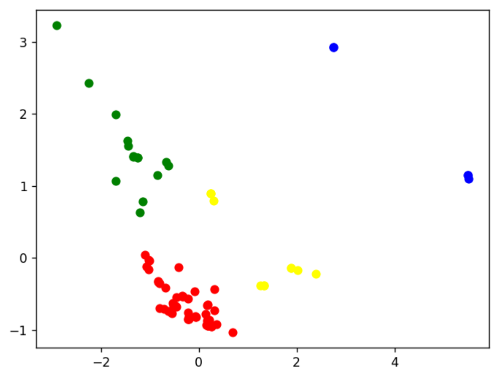

Figure 4.4: Fault Clusters and Sizes

Since the duration of transistor-level simulations is constant and determined by the capabilities and limitations of the simulation tools, the focus shifts towards optimizing the remaining components of the simulation process. Model fitting and the search for optimal template values consume the bulk of the time, accounting for approximately 50% of the total simulation time. While attempts to streamline this process could expedite simulations, they risk compromising the model's accuracy, leading to significant discrepancies between the simulated and actual circuit behaviors. Meanwhile, performance parameter calculations, representing about 20% of the total time, emerge as a viable option for optimization.

Using data clustering techniques offers a strategic solution to streamline the calculation of performance parameters, resulting in significant time savings. This approach is particularly effective in Flash ADCs due to the recurring nature of faults in the comparators. In a Flash ADC with 7 comparators that may each have up to 100 unique faults, there could be a total of 700 distinct fault simulations. Observed that the average cluster size is around 10 for the data collected. However, by employing fault clustering for all ADC faults, a substantial reduction in computational overhead can be achieved, revolutionizing simulation efficiency. Table 4.4 shows the improvement in simulation time from implementing fault clustering.

Table 4.4: Fault Simulation Time Comparison with Clustering

	3b ADC	8b ADC
SP	156min	12288hrs
HL	14min	143hrs
Reduction	11x	85x

As a result, the simulation time is reduced tenfold without compromising the model's accuracy in analyzing fault impacts. This optimization strategy highlights the importance of innovative approaches to simplify the complexities of ADC fault modeling, and demonstrates the potential for significant improvements in simulation efficiency. These advancements make it possible to create accurate and efficient fault models, thus improving the efficacy of simulations in capturing the nuanced behaviors of Flash ADCs under fault conditions.

Chapter 5

CONCLUSION

This work aims to close the gap between existing transistor- level fault simulation tools, known for their accuracy, and behavioral abstraction levels, characterized by shorter simulation times. Our approach involves incorporating functional enhancements into the high-level models of individual circuit blocks. The model enhancements are chosen from a predefined set of templates through a two-tier algorithm that aims to align the responses between SPICE simulations and HL simulations. Two similarity metrics with dynamically adjustable thresholds are used for model fitting. Experimental findings on case study, a flash ADC , demonstrate that highly accurate fault simulations can be attained at a significantly reduced simulation time. Data clustering algorithm also decreases the sim time by 4 folds. The image processing for layout can be further continued and can be completed to fruit probability of shorts and improve overall fault model.

REFERENCES

[1] Sadia Azam, Dall'Ora, et al. Analog defect injection and fault simulation techniques: A systematic literature review. *IEEE TCAD*, pages 1–1, 2023.

[2] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, 1994.

[3] Mayukh Bhattacharya, Beatrice Solignac, and Michael Durr. Application of sampling in industrial analog defect simulation. In *2022 IEEE International Test Conference (ITC)*, pages 436–445, 2022.

[4] D. Binu and B.S. Kariyappa. A survey on fault diagnosis of analog circuits: Taxonomy and state of the art. *Int. Jour. of Electronics and Communications*, 73:68–83, 2017.

[5] F. Duvivier and M. Rivier. Approximation of critical area of ics with simple parameters extracted from the layout. In *Proceedings of International Workshop on Defect and Fault Tolerance in VLSI*, pages 1–9, 1995.

[6] Osman Emir Erol and Sule Ozev. Knowledge-and simulation-based synthesis of area-efficient passive loop filter incremental zoom-ADC for built-in self-test applications. *ACM TODAES*, 24(1):1–27, 2018.

[7] Muhammad Umer Farooq, Likun Xia, Hussin, et al. High-level fault modeling and fault propagation in analog circuits using NLARX automated model generation technique. *IEEE ICIAS*, 2:846–850, 2012.

[8] W.G. Fenton, T.M. McGinnity, and L.P. Maguire. Fault diagnosis of electronic systems using intelligent techniques: a review. *IEEE Trans. on Systems, Man, and Cybernetics*, 31(3):269–281, 2001.

[9] Enrico Fraccaroli and Franco Fummi. Analog fault testing through abstraction. In *IEEE DATE*, pages 270–273, 2017.

[10] Enrico Fraccaroli, Davide Quaglia, and Franco Fummi. Simulation-based holistic functional safety assessment for networked cyber-physical systems. In *FDL*, pages 5–16, 2018.

[11] Yaping He, Xiaohua Luo, and Nianxiong Tan. Critical area computation based on equidistance line for small layouts. In *2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA)*, pages 219–218, 2013.

[12] Chung Kin Ho, P.R. Shepherd, et al. Hierarchical fault diagnosis of analog integrated circuits. *IEEE TCAS-I*, 48(8):921–929, 2001.

[13] Mehmet Ince, Ender Yilmaz, Wei Fu, Sule Ozev, et al. Fault-based built-in self-test and evaluation of phase locked loops. *ACM TODAES*, 26(3), jan 2021.

[14] Mehmet Ince, Ender Yilmaz, Wei Fu, Joonsung Park, Krishnaswamy Nagaraj, LeRoy Winemberg, and Sule Ozev. Digital built-in self-test for phased locked loops to enable fault detection. In *IEEE ETS*, pages 1–6. IEEE, 2019.

[15] Mehmet Ince, Ender Yilmaz, and Sule Ozev. Enabling fast process variation and fault simulation through macro modelling of analog components. In *IEEE NATW*, pages 1–6, 2018.

[16] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[17] Rohit J Kate. Using dynamic time warping distances as features for improved time series classification. *Data Mining and Knowledge Discovery*, 30:283–312, 2016.

[18] A. Khouas and A. Derieux. Fault simulation for analog circuits under parameter variations. *Journal of Electronic Testing*, 16:269–278, 2000.

[19] Y. Kiliç and M. Zwoliński. Behavioral fault modeling and simulation using VHDL-AMS to speed-up analog fault simulation. *Analog Integrated Circuits and Signal Processing*, 39:177–190, 2004.

[20] Zhenbao Liu, Taimin Liu, Junwei Han, Shuhui Bu, Xiaojun Tang, and Michael Pecht. Signal model-based fault coding for diagnostics and prognostics of analog electronic circuits. *IEEE Trans. on Ind. El.*, 64(1):605–614, 2016.

[21] Hui Luo, Youren Wang, Hua Lin, and Yuanyuan Jiang. Module level fault diagnosis for analog circuits based on system identification and genetic algorithm. *Measurement*, 45(4):769–777, 2012.

[22] N. Nagi and J.A. Abraham. Hierarchical fault modeling for analog and mixed-signal circuits. In *IEEE VTS*, pages 96–101, 1992.

[23] Evanthia Papadopoulou. Net-aware critical area extraction for opens in vlsi circuits via higher-order voronoi diagrams. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(5):704–717, 2011.

[24] Haoyang Shen, Adam Blaq, Deepu John, and Barry Cardiff. A foreground mismatch and memory harmonic distortion calibration algorithm for TIADC. *IEEE TCAD-I*, 70(3):1110–1120, 2022.

[25] Kristina P. Sinaga and Miin-Shen Yang. Unsupervised k-means clustering algorithm. *IEEE Access*, 8:80716–80727, 2020.

[26] H. Spence. Automatic analog fault simulation. In *AUTOTESTCON*, pages 17–22, 1996.

[27] Joseph Staudinger et al. Memory fading volterra series model for high power infrastructure amplifiers. In *IEEE RWS*, pages 184–187, 2010.

[28] Chauchin Su, Shenshung Chiang, and Shyh-Jye Jou. Impulse response fault model and fault extraction for functional level analog circuit diagnosis. In *IEEE ICCAD*, pages 631–636, 1995.

[29] Gurusubrahmaniyan Subrahmaniyan Radhakrishnan and Sule Ozev. Adaptive modeling of analog/RF circuits for efficient fault response evaluation. *Journal of Electronic Testing*, 27:465–476, 2011.

[30] Stephen Sunter. Analog fault simulation - a hot topic! In *IEEE ETS*, pages 1–5, 2020.

[31] Michail Vlachos, Marios Hadjieleftheriou, Dimitrios Gunopulos, and Eamonn Keogh. Indexing multi-dimensional time-series with support for multiple distance

measures. In *ACM International Conference on Knowledge Discovery and Data Mining*, pages 216–225, 2003.

[32] R. Voorakaranam, S. Chakrabarti, et al. Hierarchical specification-driven analog fault modeling for efficient fault simulation and diagnosis. In *IEEE ITC*, pages 903–912, 1997.

[33] L. Xia, M.U. Farooq, et al. Survey and evaluation of automated model generation techniques for high level modeling and high level fault modeling. *Journal of Electronic Testing*, 29:861–877, 2013.

[34] Likun Xia, Muhammad Umer Farooq, and Ian M Bell. High level fault modeling of analog circuits through automated model generation using Chebyshev and Newton interpolating polynomials. *Analog Int. Circ. and Signal Proc.*, 82:265–283, 2015.

[35] Rui Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

[36] Chenglin Yang, Jing Yang, Zhen Liu, and Shulin Tian. Complex field fault modeling-based optimal frequency selection in linear analog circuit fault diagnosis. *IEEE TIM*, 63(4):813–825, 2013.

[37] Ender Yilmaz, Anne Meixner, and Sule Ozev. An industrial case study of analog fault modeling. In *IEEE VTS*, pages 178–183, 2011.

[38] Ender Yilmaz and Sule Ozev. Defect-based test optimization for analog/RF circuits for near-zero DPPM applications. In *IEEE ICCD*, pages 313–318, 2009.

[39] Ranran Zhou, Peter Poechmueller, and Yong Wang. An analog circuit design

and optimization system with rule-guided genetic algorithm. *IEEE TCAD*, 41(12):5182–5192, 2022.

[40] Vladimir Zivkovic and Art Schaldenbrand. Requirements, for industrial analog fault-simulator. In *IEEE SMACD*, pages 61–64, 2019.