TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF ABREVIATIONS

ABM	Analog Behavioral Modeling
ABV	Assertion Based Verification
ADC	Analog-to-Digital Converter
AMS	Analog Mixed Signal
ASIC	Application Specific Integrated Circuits
ASL	Analog Specification Language
ATE	Automated Test Equipment
AWG	Arbitrary Waveform Generators
BDD	Binary Decision Diagram
CRV	Constrained Random Verification
CAD	Computer-Aided Design
DAC	Digital-to-Analog Converter
dB	Decibel
DfT	Design-for-Testability
DUT	Device Under Test
DUV	Design Under Verification
FFT	Fast Fourier Transform
FSM	Finite State Machine
HDL	Hardware Description Language

IC	Integrated Circuit
IP	Intellectual Property
LPN	Labeled Petri Net
LSB	Least Significant Bit
OTA	Overall Timing Accuracy
PARD	Periodic And Random Deviation
PCB	Printed Circuit Board
PDN	Power Distribution Network
PLL	Phase Locked Loop
PSL	Property Specification Language
RF	Radio Frequency
RMS	Root-Mean Square
RTL	Register Transfer Logic
RTV	Run-Time Verification
SFDR	Spurious Free Dynamic Range
SNDR	Signal-to-Noise and Distortion Ratio
SoC	System on Chip
SPICE	Simulation Program with Integrated Circuit Emphasis
SVA	System-Verilog Assertions
THD	Total Harmonic Distortion

TMS	Time Measurement System
TSV	Through-Silicon Vias
VCO	Voltage Controlled Oscillator
VRM	Voltage Regulator Module

INTRODUCTION

System-on-Chip (SoC) designs are pervasive in our modern life. They are deployed at the heart of various product systems such as popular lifestyle consumer electronics (smartphones, laptops, cameras, etc.), automotive embedded systems, avionic systems and advanced telecommunications computing architecture. A SoC enables integration of different intellectual property (IP) units, radio frequency (RF) modules, advanced microprocessor and analog/mixed-signal (AMS) circuitry into a single chip. Cornerstones of SoCs are AMS designs (Ken *et al.*, 2000) which are integrated circuits (IC) needed at the interface with the real world. Unlike digital designs, the verification of AMS designs presents several unique challenges. Indeed, the performance of analog circuits is expressed directly in terms of continuous electrical quantities and is generally sensitive to environment factors such as signal noise and current leakage. Among the important functionalities of AMS designs are generating timing references, frequency synthesis and circuit biasing which is required for correct and stable function of the SoC. Moreover, front and back end AMS designs are used to convert between analog and digital data representations. As such, they play a crucial role in the overall SoC performance and it is essential that they are devoid of functional errors. However, these systems are built by humans and they can make errors during the design. These errors can occur for a variety of reasons such as some confusion in understanding the systems specifications, misinterpretation of any specification requirement, etc. Unfortunately, a large number of errors are often discovered after the release of the design as stated in several manufacturer errata documents (Advanced Micro Devices, 2005; International Business Machines Corporation, 2005; Intel Corporation, 2000, 2007, 2010; Texas Instruments, 2018).

The aftermath of functional bugs can be avoided when they are detected early in the design process. However, a functional bug released in the final design product entails modifications to the manufacturing process, known as respins, to get a flawless design. Respins cause significant time-to-market delays and financial losses. A typical example is the Pentium FDIV

bug that affected the floating point unit of original Pentium processor models. The company attributed the bug to missing entries in the lookup table employed by the floating-point division circuitry. Ultimately, the affected chips were recalled at a cost of $475 million for the company (Markoff, 2008). The same defect today would be much more costly due to faster ramp up timelines. Therefore, it is essential to ensure that the design operates correctly and in accordance with the manufacturer's specifications at early design phases. Oftentimes, the primary reason for errors in AMS ICs is due to a misunderstanding or ambiguous information about the original design specifications (Rashinkar *et al.*, 2001). Ambiguity may arise from partially missing details about specification requirements (e.g., maximum allowable tolerances on external environmental parameters) or test requirements deemed necessary to guarantee that performance specifications are met. In either case, ambiguous definition of design specifications leads to yield loss and respins.

Nowadays, design verification is by far the most critical path in the chip design process (Wile *et al.*, 2005), depicted in Figure 0.1, where significant efforts are invested to improve first pass quality and optimize yields.

A study (Wile *et al.*, 2005), depicted in Figure 0.2, reveals that improving verification productivity drives the bug discovery earlier in the design process and thus decreases schedule and costs. This is not surprising, as in reality, if a bug is caught early during verification, it costs little to fix. However, it is more costly when caught during test and even the most costly if the customer finds the problem as illustrated in Figure 0.3.

Over the last decade, various techniques have been proposed to remove latent issues and to help minimize the number of respins. Yet, advancement in the discipline required for successful chip and system verification has not evolved at a sufficient speed to handle the ever increasing design complexity. As a result of this deficiency, a considerable portion of the total design cycle time is devoted to the verification process. According to (Cadence Design Systems, 2007), 70 % of the

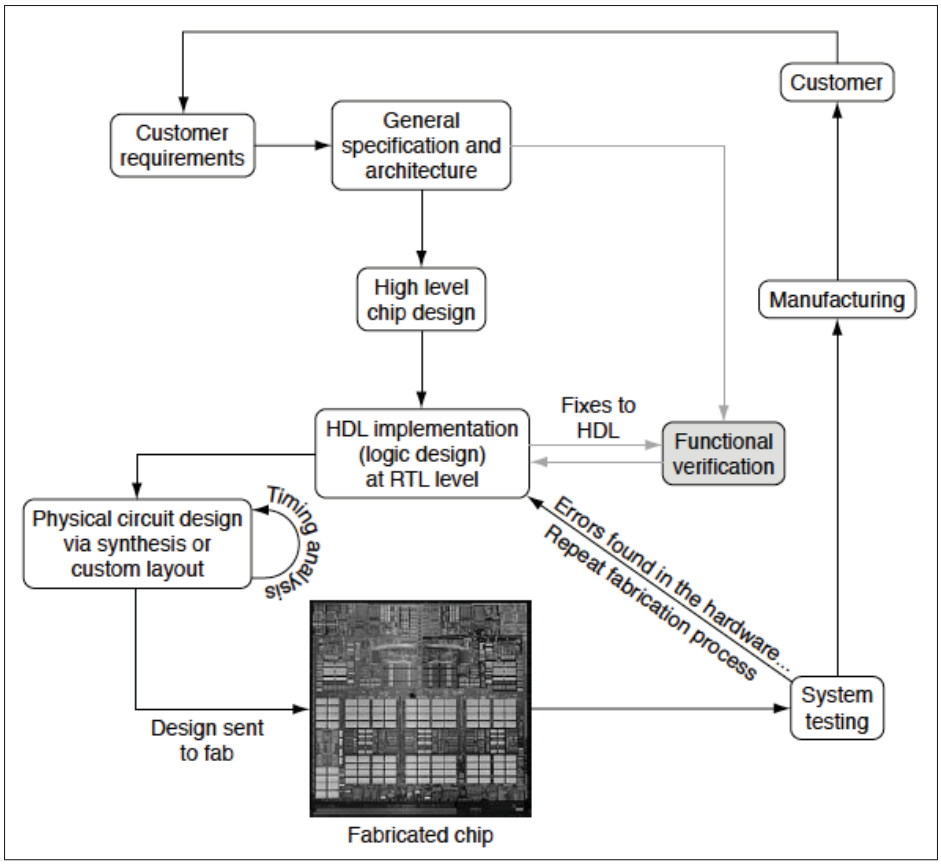

Figure 0.1 The chip design process (Wile *et al.*, 2005)

product respin are due to functional bugs, with industry/research team devoting about 80 % of effort to pre-silicon verification (Robert, 2005). In pre-silicon verification, which is performed at different abstraction levels, the task is to ensure that the logic in a software model of the hardware design operates correctly under all circumstances as stipulated by the specifications. Once first silicon prototypes are available, post-silicon validation tests come into play to detect manufacturing defects. All in all, the ultimate goal of pre-silicon verification and post-silicon validation is to ensure that the final chip product strictly obeys its predefined specifications. In spite of the complementarity of the two approaches, their application at different stages in the chip design process still leaves room for potential undiscovered bugs.

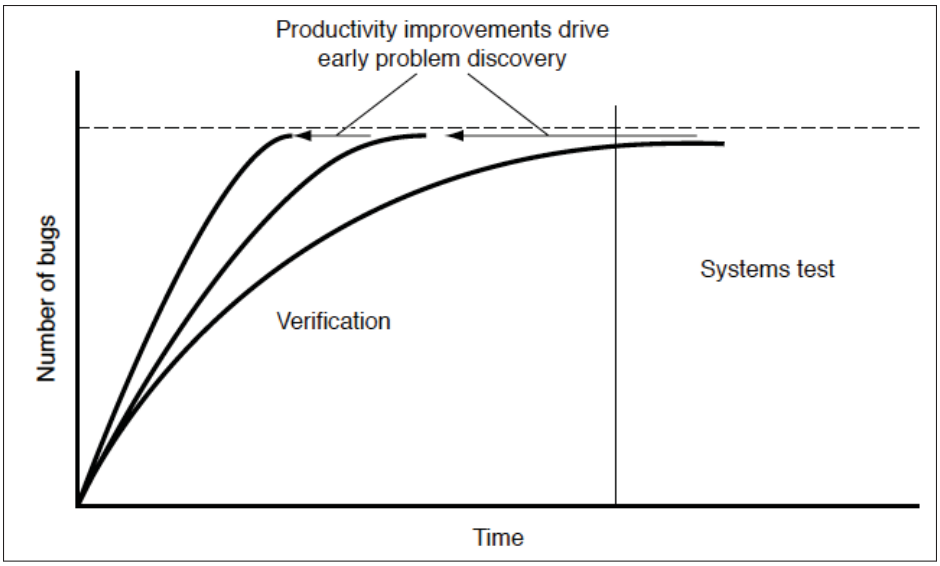

Figure 0.2 Verification Productivity vs. Bug Discovery
(Wile *et al.*, 2005)

A key issue not yet addressed by traditional verification techniques is to early and efficacy account for specification testing requirements when we verify an AMS design. Indeed, one aspect of test which is usually beyond the scope of functional verification, is the presence of additional requirements. These include, for example, adequate interfaces between the tester and the device under test (DUT), extra test access points or specific sub-circuits which may be incorporated into the design structure to make the test cost effective or simply feasible. Moreover, probing devices may alter their functionality in unexpected ways. This is in addition to other challenges that may arise from various parasitic effects such as noise and operating environment conditions.

To fully analyze this impact on the overall AMS design behavior and meet the performance specification, it is necessary to adequately define the additional specification and test requirements and to assign appropriate values to their associated elements early on in the design process. However, in reality, adequate values for some specification and test requirements are seldom clearly defined or provided in the design preliminary datasheet. This is because usually

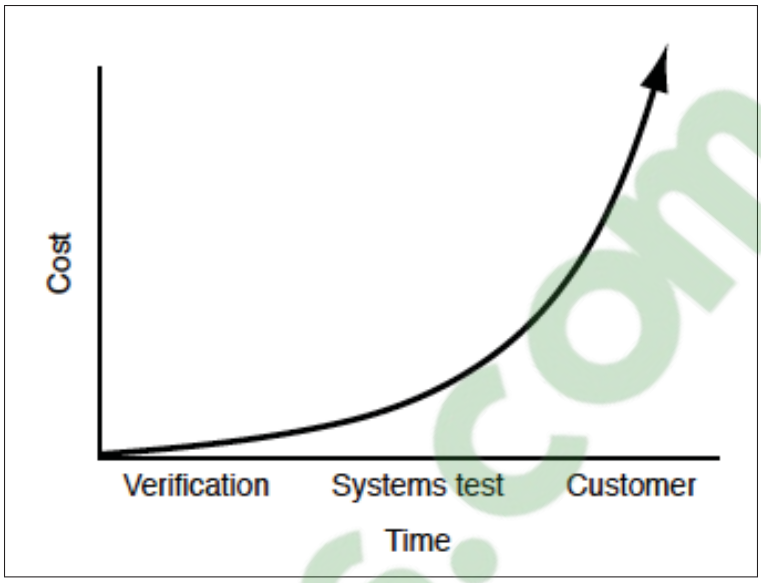

Figure 0.3 The costs of undetected bugs over time
(Wile *et al.*, 2005)

these additional requirements are not allocated higher priority before the test phase, until fabrication is complete. For some type of requirements which, if not adequately defined, it could be too late to make the necessary changes during the test phase. It is therefore of great utility to the verification engineer to build appropriate modeling and verification paradigm to better handle specification testing requirements at a high level of abstraction.

Problem Statement

While AMS modules account only for a small part of the whole SoC (25 % or even less), they are the source of more than 75 % of design problems and risks (Eisawy, 2012). Various verification methodologies, each addressing specific verification challenges, have been shown to be useful for detecting and eliminating design failures. Nevertheless, poor "first time" design success rates, falling to 28 % (FarWest Research and Mentor Graphics, 2007), illustrate the lack of cohesive and efficient techniques to allow a predictable verification process that leads to the highest possible confidence in the correctness of designs. In the current state-of-the art, multiple approaches were concerned with high-level modeling and verification of AMS

designs to allow early detection of functional bugs (Pichon *et al.*, 1995; Navin *et al.*, 1997; Gerlach & Rosenstiel; Bonnerud *et al.*, 2001; Bjornsen & Ytterdal, 2003; Dubikhin *et al.*, 2016). Though, these attempts to solve the AMS verification problem fall short in addressing some real-world uncertainties related to the design behavior due to:

- The functionality of analog blocks within AMS systems is most directly expressed in terms of continuous electrical quantities and is normally influenced by higher order physical effects when designing in deep submicron (e.g., current leakage, increased parasitics, etc.) in addition to the fact that these systems usually act upon unpredictable environmental conditions like temperature effects and random noise effects that can alter their behavior in unexpected ways. Several simulation-based verification techniques were focussed on studying the circuit behavior under such conditions at the circuit-level which is extremely costly in terms of computation time and memory resources.

- An AMS design has to meet two conflicting demands since on the one hand it has to represent the real physical behavior of the design as accurately as possible and on the other hand it must be sufficiently simple in order to maintain the computing time for verification reasonably short. Therefore, moving circuit analysis to a higher level of abstraction while trading off some accuracy is extremely valuable in detecting circuit failures earlier in the design cycle and consequently in raising confidence in the end product.

- Verification of AMS designs is primarily based on checking their conformance with respect to an initial product design specification. There can be inconsistencies in the specification itself which could result in an erroneous design behavior. Up to the present time, the problem of conflicts or inaccuracies in the specification plans still does not go

away completely. Consequently, finding a way to detect and avoid incomplete or ambiguous specifications that result in costly rectification work is also extremely valuable in detecting design failures earlier in the design process.

To sum up, a key for a sound verification of AMS designs is an appropriate model that best describes their functional behavior in the real operating environment and efficiently account for the additional specification and test requirements, in particular those that can affect the design, during the early design phases. This will help in the early discovery of out-of-specification failures and will allow designers to make the necessary changes before committing designs to manufacture which will reduce the risk of costly errors. In this thesis we present a novel constraint-driven verification methodology for 2D and 3D AMS designs that leverages specific specification and test requirements to gain a better analysis and understanding of each design specification and consequently to keep track of any inconsistent, incomplete or ambiguous specification information and to enable early out-of-specification failures detection.

Thesis Objectives

The main objective of this thesis is to enhance the detection of hidden out-of-specification failures, usually dealt with during the test phase, at higher levels of abstraction. Our main focus lies on specification errors caused by missing or incomplete definition about their specification and test-related constraints. This is achieved through the development of a unified modeling and verification approach for automatically monitoring specifications of AMS designs in the presence of specification and test-related constraints. In particular, we aim at developing:

- A modeling method for AMS designs that efficiently account for the additional specification and test-related constraints and allows to predict, in an early design phase, the circuit behavior under extreme testing conditions (e.g., operational temperature levels, random noise effects etc). Modeling requires the definition of at least one test scheme

for each design specification to demonstrate specification compliance. Yet, even if tests are conducted at the end of the process, they are commonly defined as soon as the design high-level implementation (even if partially complete) exists and the global design input/output (I/O) interfaces are defined. This early definition of test schemes serves two purposes: 1) it avoids omission of important specification and test-related constraints during the design phase, such that the tests cannot be adequately performed or are not cost effective, and 2) it helps in the early discovery of missing, incomplete or misunderstood specifications, by forcing a detailed analysis of all specifications.

- A verification framework for systematic analysis of the completeness and coherence of AMS design specifications with respect to specification and test-related constraints. This could be done through: 1) building a verification checklist in order to ensure that all the specifications are covered using at least one test and none of them is missed, 2) building a list of specification-related constraints, and 3) extracting the relevant test-related constraints and integrating them into the constraint list and consequently into the verification process.

Thesis Contributions

The proposed approach will allow us to study some of the effects in a traditional mixed-signal test environment at a higher level of abstraction, during design verification phase. This is very useful and essential for the performance evaluation of the fabricated AMS design. The contribution of this thesis can be summarized with the following points:

- We provide a method to generate a set of additional constraints defined based on the testing techniques and strategies that are necessary to guarantee proper AMS circuit functionality especially under extreme conditions.

- We introduce an automated modeling approach which enables specification and test-related constraints integration and analysis during design verification stage in order to detect possible specification conflicts or inaccuracies that could lead to complete system failure and hence facilitate the total post-fabrication chip testing process.

- We extend the constraint-based verification approach to handle set of performance features derived from the mixed-signal tester. We perform a simulation-based technique to assess the impact of the tester inaccuracies on the reliability of the test. Advantages of the proposed approach are robustness and flexibility to account for external tester characteristics. The approach estimates the acceptance/rejection of the circuit with respect to the tester's characteristics.

- The whole thesis framework is developed as a Constraint-based Property Checker Matlab tool for automatic modeling and verification of AMS designs in the presence of specification and test-related constraints. The tool is implemented using Matlab based object-oriented approach in form of object classes and functions.

Thesis Organization

The remainder of this thesis is organized as follows: Chapter 1 provides a more in-depth look into specification and test-related constraints, focusing on physical constraints as well as those affecting the AMS design. Likewise, fundamentals of analog behavioral modeling as well as mixed-signal testing for 2D and 3D ICs are presented to equip the reader with some concepts that are going to be used in the rest of this thesis. Chapter 2 presents the different verification techniques applied to 2D and 3D ICs, along with a literature overview on the relevant work. Chapter 3 describes the proposed methodology for modeling and verifying AMS designs in the presence of specification and test-related constraints. The second part of Chapter 3 presents an

extension of the methodology to handle additional set of performance features derived from the mixed-signal tester components used for specifications tests. The effectiveness of this methodology is demonstrated for several benchmarks in Chapter 4 namely, a commercial frequency synthesizer IC based on a Charge Pump Phase-Locked Loop (CP-PLL), a sigma-delta modulator and a 3D clock tree IC. A number of interesting functional specifications are analyzed and verified (part of this work was submitted for publication in the IET Computers & Digital Techniques journal). Finally, some conclusions and prospective future research directions is presented.

CHAPTER 1

PRELIMINARIES

This chapter presents some basic definitions of specification and test-related constraints that will be used as a part of the proposed modeling and verification framework. It also highlights the fundamentals of analog behavioral modeling as well as mixed-signal testing for 2D and 3D ICs.

1.1 Specifications and Test-related Constraints

In system verification, we verify that a design is correct with respect to specifications. Even if the design is proven to be flawless, there is still a question of how complete the specifications are, and whether they effectively cover all possible behaviors of the design. The challenge of writing complete specifications with a clear understanding of the systems functionality and operational concepts is even more crucial in the design process. It turns out that one of the main problems is incomplete or incorrect specifications, which miss essential informations or have ambiguous information or may not be kept up as modifications take place in the design process. Such deficiencies usually result in substantial design errors that might slip through the verification process. In this thesis, we take the verification process a step further by eliciting and analyzing additional specification and test requirements in order to achieve better errors detection as early as possible in the design process. To do so, we apply two types of constraints: specification-related constraints and test-related constraints. These constraints are used in the proposed modeling and verification approach as a means to ensure that specifications are complete and unambiguous.

In general, specifications define the overall performance (power, speed, etc.) to be reached during test for a given set of environmental conditions (voltage, temperature, etc.). We define each {performance, condition} set as a specification-related constraint. Each specification should have at least one test procedure used to ensure final design compliance. The associated test procedures are usually defined prior to manufacture, as soon as the design initial model is

established. This is useful to efficiently account for testing requirements, in particular those that can affect the design, during the early design phases. Indeed, additional requirements may come from the test itself. For example, testers are constrained by the limited amount of memory available to store test patterns and responses, as well as the limited number of available tester channels and the maximum frequency at which they can operate. Additionally, measurement inaccuracies present a common problem when analog cores are tested in a mixed-signal test environment based on digital processing (Best, 2003). This problem, gets even worse when we have noisy DC power supply lines, improper grounding of the wafer probe, and a lack of proper noise shielding of the wafer probe station (Lau, 2002). Add to this, extra test access points or special interface circuits that may be needed to make the test cost effective, if not simply feasible. Probing designs may alter their functionality in unexpected ways: this impact must be considered during verification. In this thesis, we define a test-related constraint as any additional requirement linked to the test itself. We differentiate between two main categories of test-related constraints: physical constraints (such as the interface between testers and the DUT, input-signal conditions, input impedance, etc.) and constraints that can affect the design. We define test-related constraints affecting the AMS design as those implemented by adding specific sub-circuits or components to the design structure. Such constraints are mandatory to achieve adequate specification tests.

In what follows, we explain through concrete examples how omitting important specification and test-related constraints can affect AMS designs performance.

1.2 Examples Showing the Impact of Specification and Test-related Constraints on AMS Designs performance

Several concrete examples to highlight the impact of omitting important specification and test-related constraints during the early design phase are presented in (Shapiro *et al.*, 1995; Burns & Roberts, 2001; Comte, 2003). These include not providing adequate test points for function performance measurement and inappropriate consideration early on in the design phase on how large amounts of data will be reduced, analyzed, and reported.

A particular example consists of a cellular telephone voice-band interface device used to convert digital voice samples into an audio signal for the telephone's earpiece (Burns & Roberts, 2001). The test scheme used for the audio interface is illustrated in Figure 1.1.

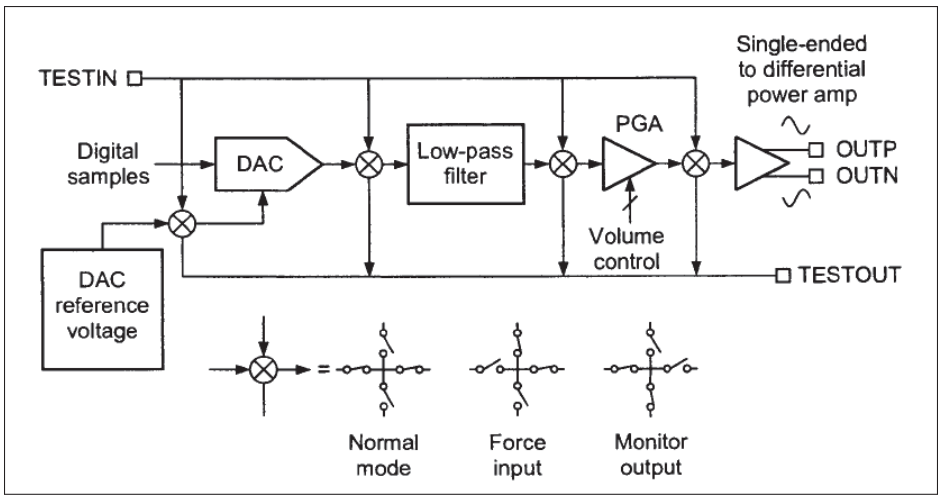

Figure 1.1 Audio interface test scheme (Burns & Roberts, 2001). The different operation modes detailed at the bottom of the figure apply to all switches.

This device has failed the gain error specification tests. The cause of failure was from lack of early consideration of test-related constraints. In fact, failure analysis revealed that the Digital-to-Analog Converter (DAC) was contributing additional gain error of 0.2dB due to a parasitic resistance in the DAC reference voltage used to set the full-scale range of the DAC. Likewise, the power amplifier sub-block was introducing a gain error of -0.7 dB due to inappropriate transistor sizing which explains the total channel gain error of -0.5dB failing by -0.45dB (the gain error specification is 0dB $\pm$ 0.05 dB as defined in the device's datasheet).

In this particular case, it is very likely that the design error was not caught by the verification process because of an underestimation of the multiple test access points impact on the design performance. Incomplete specifications of sub-blocks might also have played some role. These test points were added for analog signal observability purposes and represent a good example of test-related constraints. If these test points were of great help identifying the design root

cause of failure, their presence could have been leveraged to detect the error before fabrication. Indeed, an early definition of sub-block tests would have forced a more detailed analysis of the test access point impact as well as a more rigorous definition of the sub-block specifications. More specifically, the overall gain error specification would have been explicitly broken down. Each of these specifications would therefore have been more rigorously verified with respect to its specification and test-related constraints, namely the appropriate values of the parasitic resistance and the power amplifier transistor sizes, using a compatible verification checklist. This would insure that none of these informations is missed. Both specification and test-related constraints for mixed-signal integrated circuits represent thus a key for a first pass silicon success.

Another example to highlight different types of specification and test-related constraints is presented in (Comte *et al.*, 2003). It consists of an ADC device that was shown out of compliance with the required specifications due to the effect of additional constraints summarized in Table 1.1 (columns 2 and 3).

Table 1.1 Specification and Test-related constraints for an 8-bit ADC dynamic specifications

Specifications	**Specification-related Constraints**	**Test-related Constraints**
1. SNDR (Signal-to-Noise and Distortion Ratio) min=48 2. SFDR (Spurious Free Dynamic Range) max=-55 dB 3. THD (Total Harmonic Distortion) max=-55dB	• The SNDR, SFDR and THD parameters are dependent upon the amplitude deviations of the stimulus. • Possible measurement errors due to input signal variation. $\Rightarrow$Stimulus amplitude < FS FS represents the ADC full scale range.	• Number of samples $\geq$1024 • Number of periods $\geq$103 • Stimulus p-p amplitude $\leq$ FS-4 LSB

A typical test setup on the industrial ATE (Automated Test Equipment) HP83000, illustrated in Figure 1.2, was used for the ADC specifications tests. Indeed, test conditions defining the test stimulus are configured in the waveform synthesizer. The later generates a sine-wave signal with an input frequency (f_{in}), an amplitude (A_{in}) and an offset (V_o). This stimulus is applied on the converter input and the resulting samples are acquired in the capture memory at the rate of the sampling frequency (f_s). These samples are then transferred to the CPU for further processing.

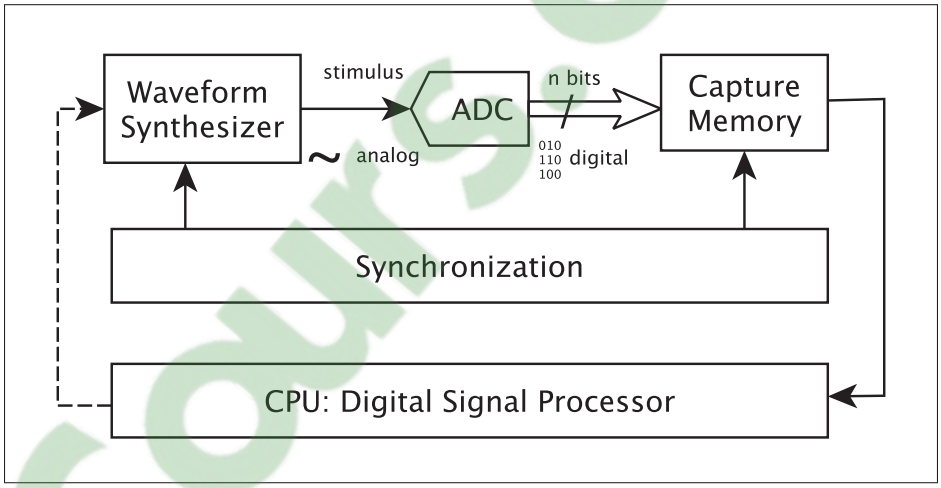

Figure 1.2 ADC testing environment (Comte *et al.*, 2003)

Spectral analysis of the ADC under test is based on the exploitation of the Fast Fourier Transform (FFT) of the digital samples acquired at the converter output when a pure sine wave is applied to its input. The resulting spectrum is analyzed to evaluate the ADC dynamic specifications presented in Table 1.1 (column 1). In this particular example, the most prominent test-related constraints susceptible of having an effect on the ADC dynamic specifications are: the number of samples N considered to perform the FFT, the number of periods M of the input sine-wave during acquisition, and the input signal amplitude (A_{in}). The number of samples N taken into account for the analysis is an important factor to consider for low-cost testing. Theoretically, FFT only requires at least one sample per code is present in the data record. However, this demands a perfect synchronization usually difficult to ensure in practice.

A lack of early consideration of the aforementioned constraints (see Table 1.1, columns 2 and 3), in particular the input signal amplitude, has led to inaccurate measurement results and the ADC was reported in out-of-specification (Comte *et al.*, 2003). In fact, in a testing environment it is not possible to precisely guarantee the value of the generated input stimulus amplitude while the values of all the dynamic specifications are sensitive to this amplitude. For instance, a deviation in the input signal peak-to-peak amplitude of less than 0.1 LSB can result in a variation of 20dB or more in the measured harmonic distortion specification which does not represent the ADC performance.

In summary, there are many examples illustrating the effects of overlooked specification and test-related constraints on AMS designs performance roughly ranging from subtle performance degradation to complete design failure. In this thesis, we present the first attempt to include all these constraints into the verification process which in turn:

- Ensures that all the specification and test-related constraints are defined and none of them is missed.
- Guarantees that each of these constraints is within the adequate range of operating conditions (imposes a routine to check for each of the specification and test-related constraints and assign a value within the tolerance limits as will be explained in Chapter 3).

If the specification-related constraints are commonly used (under different forms), it is the first time that test-related constraints are applied during the verification process, to the best of our knowledge. The basic idea will be to integrate all the specification and test-related constraints into the AMS design model, initially provided by the designer, at a high level of abstraction[1]. We use the PSpice Analog Behavioral Modeling (Wilson, 1989) or the Matlab/Simulink tool environment (The Mathworks Inc., 2011) to implement the AMS models in the presence of specification and test-related constraints. A brief description of PSpice's ABM feature will be presented in the subsequent section.

[1] The proposed modeling and verification methodology is flexible and can handle different levels of abstraction depending on the circuit configuration input.

1.3 Analog Behavioral Modeling using PSpice

Analog Behavioral Modeling (ABM) feature of PSpice allows a flexible description of electronic devices, i.e, a mathematical relationship is used to model a circuit segment with no need to design the segment component by component which significantly reduces the total simulation time. There are two main applications of behavioral modeling in analog simulation domain: modeling new system types and black-box modeling of complex systems. Extensions introduced into the PSpice simulator in order to enable these applications are detailed in (Wilson, 1989).

To illustrate the use of the ABM feature of PSpice, consider a simple Voltage Controlled Oscillator (VCO) modeled as a sinusoidal function with the following form:

$$OUT = sin((twopi * fc * time) + phi) \tag{1.1}$$

Its behavioral model using ABM elements is shown in Figure 1.3, where *twopi*, *fc* (frequency of the signal) and *phi* (phase angle of the signal) are all constant global parameters defined with a parameter block (PARAMETERS part).

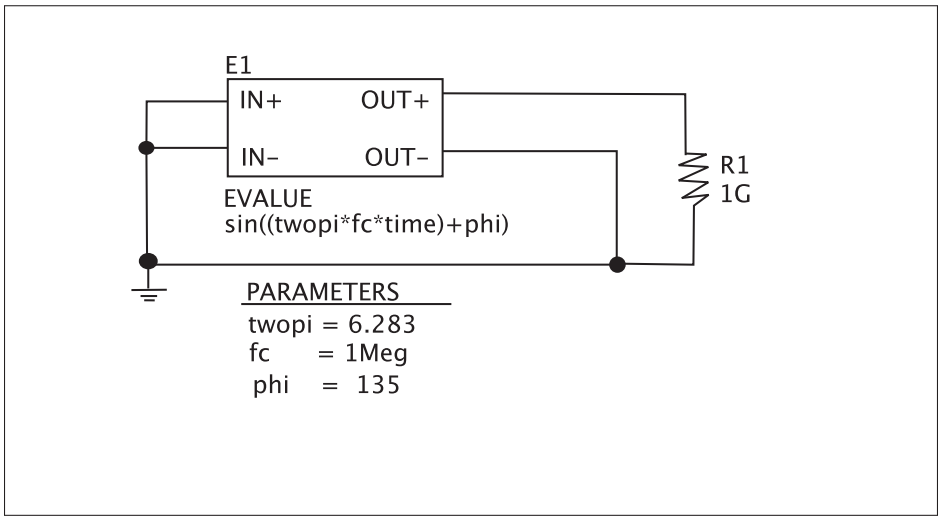

Figure 1.3 VCO behavioral model (Wilson, 1989)

1.4 Fundamentals of Mixed Signal Testing

The main purpose of functional testing is to ensure that the fabricated chip operates in conformance with its specifications. Based on the system specifications, the first step in the test process is to specify the testability features (e.g., scan, test points, etc.) followed by the set of tests defining the test plan. Once the test plan is approved, an initial test program is defined. The latter may undergo continual modifications while the design is still not yet finalized. The process of establishing such test program requires several steps, starting from the design specification to the generation of the test code (Burns & Roberts, 2001). In what follows we will examine these steps as well as the test program functionality in some detail.

1.4.1 Test Specification Process

One of the key elements in the design flow is the specification document also known as datasheet. It should provide explicit information about the functionality requirements that must be satisfied by the design. Moreover, it represents a formal communication channel between the designer and the test engineer. However, it may contain mistakes and ambiguities that must be corrected earlier in the design process. In practice, the specification datasheet has many sections (Burns & Roberts, 2001). Among these, some of the most important to test engineering are: the device description, principles of operation, electrical characteristics, timing diagrams and package/pinout information. A test is derived for the device description and principles of operation to ensure that the device fulfills the requirements. Same goes for the electrical parameters of the design that should be tested in all modes of operation. Typically, a test plan consists of several major components (Burns & Roberts, 2001). Most noteworthy among these are:

- **Test Code and Digital Patterns**: Test code and digital patterns are the predominant elements of the mixed-signal test program. The former is used to monitor pertinent elements such as the order and timing of instrument settings, signal generation and signal measurements that compose each measurement in the test program. Digital patterns are

made up of a sequence of vectors, where a vector defines the required input to the DUT and the expected output value. In mixed signal testing, these patterns must be executed at a precise frequency.

- **Simulation Code**: Simulation code allows the simulation of the design model with respect to the instructions in the test program. The obtained design responses are compared to the expected test limits.

- **Debuggability**: Test program debugging process is used to locate hardware problems such as bad Device Interface Board (DIB) layout and broken tester modules. Likewise, it determines measurement correlation errors as well as intermittent bugs. The most prominent test techniques and measurements that make up a mixed signal test program are: continuity test, leakage test, supply current tests and DC measurement. The latter involves some major problems such as accuracy and repeatability[2], mainly due to a series of factors summarized in Table 1.2.

Table 1.2 Components of error in a given specification of accuracy (Burns & Roberts, 2001)

Errors	**Description**
Systematic errors	Usually show up consistently from measurement to measurement and are mainly caused by consistent errors in the measurement instruments. These kind of errors can often be reduced through calibration.
Random errors	Generally caused by thermal noise or other sources in either the DUT or the tester hardware.
Quantization errors	Caused by the conversion from an analog signal (e.g., input voltage or current) to a finite set of possible digital output results from the ADC.

[2] Accuracy and repeatability are one of the most exasperating aspects of mixed signal testing. Many efforts are devoted to solve accuracy and repeatability problems. A successful resolution of a perplexing accuracy problem is a significant achievement in a test engineer's day (Burns & Roberts, 2001).

1.4.2 Mixed-Signal Tester Overview

A typical mixed-signal tester architecture is illustrated in Figure 1.4. It consists of multiple subsystems such as waveform digitizers, arbitrary waveform generators (AWG) and digital pattern generators. Most mixed-signal testers have several common building blocks, namely DC sources, digital subsystem, AC source and measurement, time measurement system and computing hardware (Burns & Roberts, 2001). In what follows, we provide a brief description of these building blocks.

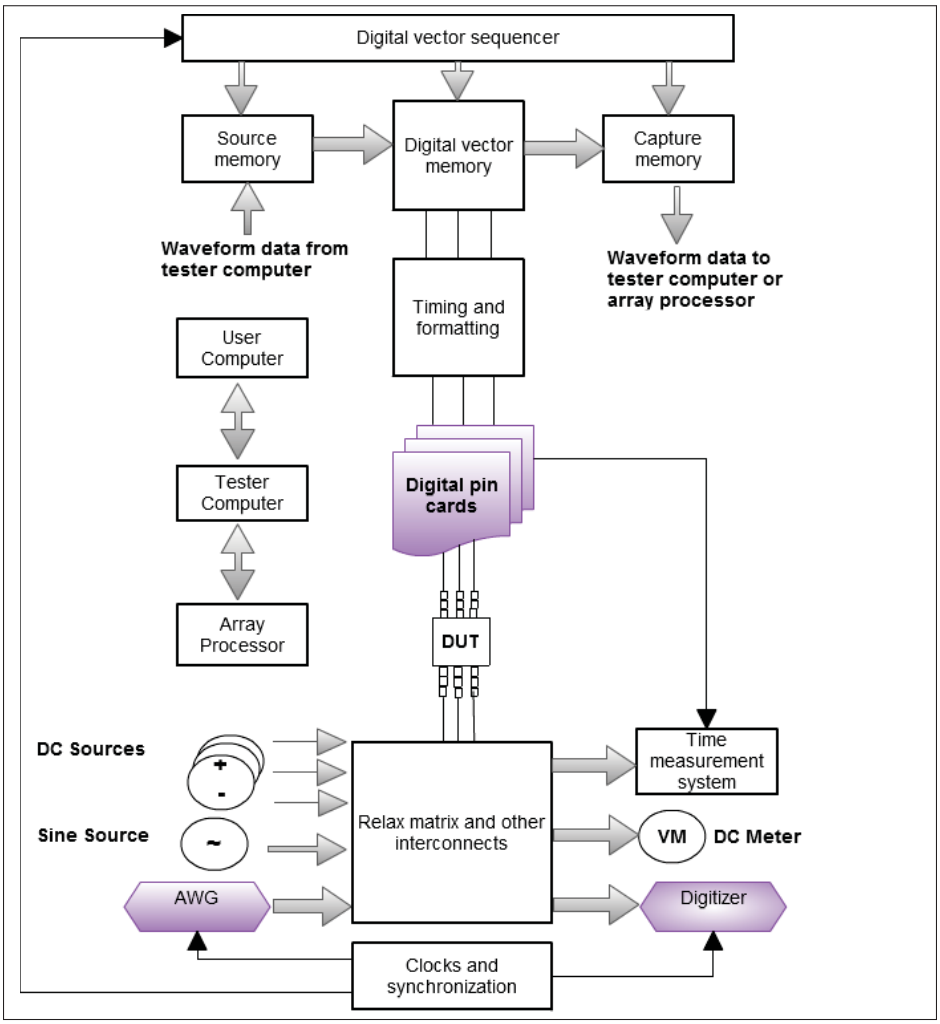

Figure 1.4 Mixed-signal tester architecture (Burns & Roberts, 2001).

- **DC sources**: The DC subsystem is one of the elements that make up most of mixed-signal testers. It consists of several components including multimeters, voltage current sources, precision voltage references, calibration sources and relay matrices. Each of these components serves a different purpose such as providing fast measurements with high-accuracy, or producing the DC voltages or currents that are required to power up the DUT and stimulate its DC inputs. Another important purpose is to maintain flexible interconnections between the tester instruments and the DUT.

- **Digital subsytem**: Another common feature that make up the bulk of most mixed signal testers is the digital subsystem. It is mainly used to compare the outputs of the DUT with the expected results in order to ensure that the device has been correctly manufactured.

- **AC source and measurement**: An efficient way to test AC performance is to use DSP-Based testing (Burns & Roberts, 2001). The approach involves a stimulus/measurement pair namely the AWG and the waveform digitizer. An AWG consists of a bank of waveform memory, a DAC that converts the waveform data into stepped analog voltages and a programmable low pass filter that smoothes the stepped signal into continuous waveform. It is used to convert digital samples from a waveform memory into continuous time waveforms. Conversely, a waveform digitizer converts continuous-time waveforms into digitized representations. Both the AWG and the waveform digitizer operate from clock sources that are synchronized to each other and to the digital pattern's frame loop repetition rate. Such synchronization of sample rates between the AWG, digitizers and digital pattern generators is another distinguishing characteristic of mixed signal tester.

- **Time measurement system**: The time measurement system (TMS) is used to measure various parameters such as frequency, rise and fall times, jitter and propagation delay. Accurate timing measurements necessitate a high-quality signal path between the DUT output and the TMS instruments.

- **Computing hardware**: A typical mixed-signal tester includes several computers and signal processors serving different purposes such as editing and compiling a test program or performing mathematical operations on the data collected during each test.

1.4.2.1 Test Techniques for Two-dimensional AMS Integrated Circuits

Several test strategies have been proposed in theory and in practice for testing AMS circuits. In what follows, we present the most common analog test methods and their measurement setups. We also provide a brief description of how analog tests can be performed using digital sampling techniques.

The simplest analog measurement setup is composed of a signal generator to stimulate the DUT and an instrument for output readings on the DUT parameters. The signal generator can produce waves of any shape (e.g., sinusoid, square-wave, etc.) that fit the purpose of the test. Input signals are selected based on the type of measurement to be carried out. The following are the four main measurement categories (Roberts, 1996):

1. DC measurements: used for measuring the static behavior of the design like leakage currents, output resistance, transfer characteristics and offsets.
2. AC measurements: used for measuring the small and large-signal frequency response behavior of the design. Distortion measurements also pertain to this test.
3. Transient or time-domain measurements: used for measuring the behavior of designs prone to signal shapes when used in their intended application.
4. Noise measurements: used for measuring the variations in the signal that usually show up at the design's output when its input is set to zero.

A typical setup to enable most of the aforementioned measurements (1, 2 and 4) is illustrated in Figure 1.5. It includes a sinusoidal signal generator with variable amplitude and frequency

control. The output of the DUT is first filtered using a bandpass filter. Next, the power associated with the filtered output signal, once stabilized, is measured using a true-RMS power meter. Transient-type measurements need specific equipment to generate and capture the adequate test signal, e.g., bit-error rate (Roberts, 1996).

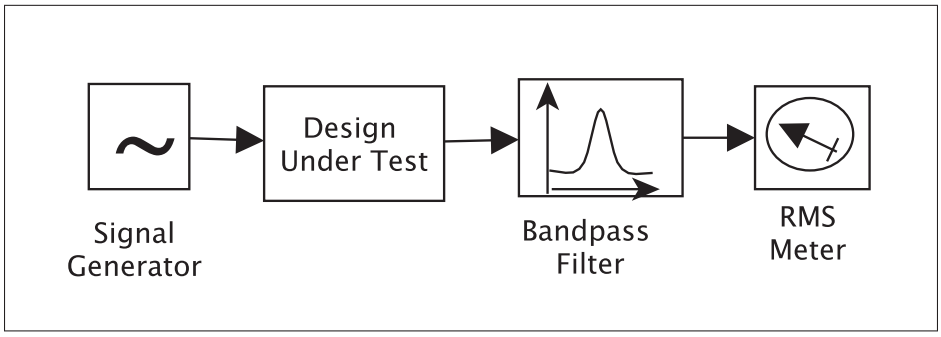

Figure 1.5 Typical analog test setup (Roberts, 1996).

While the pure analog approach to specifications measurements suffers from some problems (e.g., relatively slow when AC specifications should be tested at multiple frequencies), DSP-based testing approach allows faster and more accurate measurements (Burns & Roberts, 2001). In a DSP-based measurement system, the input test signal is numerically computed by a digital signal processor (DSP) and then fed to a digital-to-analog (D/A) converter block as illustrated in Figure 1.6. The resulted signal is then applied to the DUT whose response is digitized by an analog-to-digital (A/D) converter block and injected into the DSP for further processing. Depending on which measurement is required the appropriate software would be loaded in place (Roberts, 1996).

The approach is flexible in terms of program logic and able to pipeline the different phases of the test procedure. A second advantage is that it allows possible reuse of the same hardware for multiple test functions.

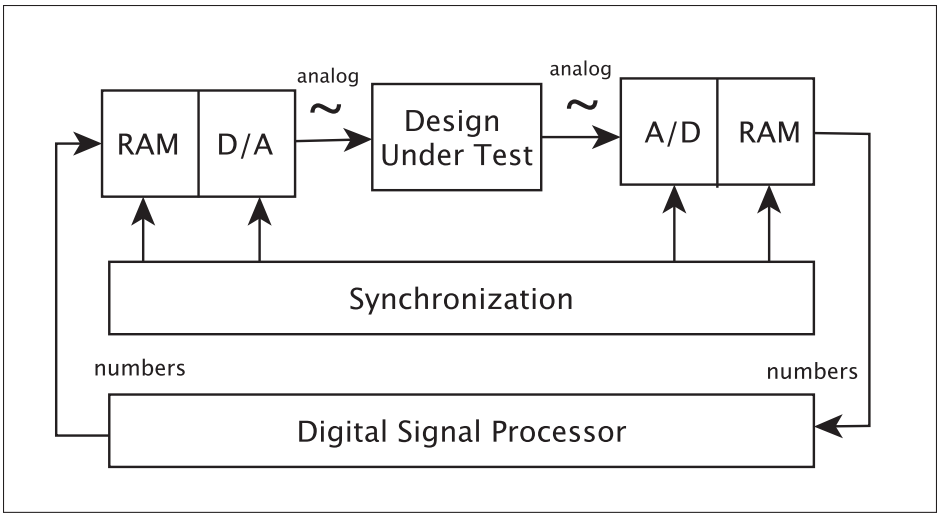

Figure 1.6 DSP-based measurement system (Roberts, 1996).

A different approach consists of adding the test circuitry into the AMS design making it easier to test. This paradigm has come to be known as Design-for-Testability (DfT) (Roberts, 1996). DfT approaches, which are popular today, refer to design modifications that enable improved access to internal circuit elements such that they can be controlled and/or observed more easily. Such design modifications can be physical such as adding a test access point to a net or simply integrating additional circuit elements for testability improvements. Examples of DfT methods applied to AMS designs are provided in (Roberts, 1996).

1.4.2.2 Test Techniques for Three-dimensional Integrated Circuits

The test process of 3D ICs can be split into two main phases: pre-bond testing and post-bond testing. The former allows testing dies before they are stacked together, while the later allows testing dies after they are stacked (Todri-Sanial & Tan). In comparison with the classical 2D ICs, the test process is faced with the following additional challenges (Marinissen *et al.*, 2010):

- Probing on multiple small probe points and thin wafers handling.
- Fault models and corresponding tests for TSV-based interconnects with the associated intradie defects.

- DfT methodologies adapted for testing the individual or stacked dies.
- Test optimization for increased efficiency and reduced Cost.

Testing approaches to address the previous problems are presented in (Todri-Sanial & Tan). These include, for example, a holistic approach that allows a unified testing of wires, microbumps and TSVs for interposer stacks, in accordance with the IEEE 1149.1 standard (Wang *et al.*, 2015). At−speed tests as well as fault models are proposed for both interposer stacks (Wang *et al.*, 2015) and TSV-based 3D ICs (Taouil *et al.*, 2015). Moreover , DfT methodologies are adapted to perform prebond and postbond testing (Lee & Chakrabarty, 2009). Such approaches necessitate access to all modules within 3D stacked IC and must be based on well-defined components and testing interfaces (Todri-Sanial & Tan).

In summary, performance specifications are evaluated based on predefined test schemes (established as soon as the design high-level implementation exists). Depending on which test strategy to apply, a set of additional requirements becomes apparent. It is notably more efficient and advantageous when these requirements become another design constraint to be considered during the early design phases. Early definition of the components of the test apparatus makes it possible to incorporate salient specifications and test constraints into the design process. This in turn can aid the designer in finding out-of-specification failures caused by overlooked test constraints. We will show in Chapters 3 and 4 how this can be efficiently done.

CHAPTER 2

LITERATURE OVERVIEW

This chapter provides an overview of previous research endeavors related to AMS verification. It comprises two main parts. The first part presents the state-of-the-art verification techniques applied to 2D ICs. The second part highlights recent advances in 3D ICs.

2.1 Introduction

Functional verification is meant to ensure that a design performs its intended function as defined by the specifications. One of its greatest challenges is detecting design behaviors which may violate the expected property specifications. A wide variety of methods, including simulation-based and formal methods, have been proposed in the literature to tackle this challenge. The most widely used verification method for AMS circuits (such as ADCs, VCOs and operational amplifiers) is simulation-based verification. Recently, however, formal methods have emerged as a promising complement to traditional simulation-based techniques and have been applied to ensure the quality and correctness of AMS circuits.

In the remaining of this chapter, we point out the different strengths and weaknesses of current AMS verification techniques. First, we overview of simulation-based methods applied to 2D ICs, followed by formal methods. We devote the last part of the chapter for a survey of the various research directions in verification techniques for 3D ICs.

2.2 System Verification Techniques for Two-dimensional AMS Integrated Circuits

Design verification techniques fall into two broad categories: simulation-based techniques and formal-based techniques. In this section we will introduce these techniques and their applications to AMS designs followed by discussions of related works.

2.2.1 Simulation-Based Techniques

Traditionally, the verification of AMS designs is carried out using simulation. In simulation-based verification, a set of input stimuli is first applied to the design under verification (DUV). As inputs are propagated through the DUV via a simulation engine, a monitor routine evaluates its output against the expected output as defined in the specification documents. To identify a design error using a simulation-based approach, every input stimulus should facilitate a way to trigger (i.e., sensitize) a bug at some point in the design. The problem of high-quality stimulus generation has resulted in the appearance of constrained-random simulation, also referred to as constrained-random verification (CRV).

Constrained-random verification (Yuan *et al.*, 2006) is considered as a very effective way in improving AMS verification quality. The idea is to verify the functionality of the design by attaching it to a testbench which generates the appropriate stimuli to drive while monitoring its output. This technique is recognized as runtime verification (Kundert & Chang, 2006). In runtime verification, a correctness property specification is checked against the current execution of a system (online monitoring) or a finite set of recorded executions (offline monitoring) using a monitor.

In order to effectively detect property violations, assertions are used in runtime verification. Assertion based verification (ABV) (Vijayaraghavan & Ramanathan, 2005) is one of the widely used verification technique as it improves the verification quality and decreases the debugging time of complex AMS designs. An assertion simply expresses a property specification. If it is violated, a failure message appears notifying the user that the property specification being monitored has failed (Foster & Krolnik, 2010). For Example, if the desired DC voltage level at the output node of an amplifier design is between 0.85V and 0.95V, we can use the following assertion:

if $DC(V(out)) \notin [0.85V, 0.95V]$ **then**
    Violation = '1'
end

Advantages of using assertions include enhanced error detection and decreased debugging time as a result of improved observability. Likewise, assertions can be used efficiently with simulation and formal verification. An assertion based verification environment is illustrated in Figure 2.1.

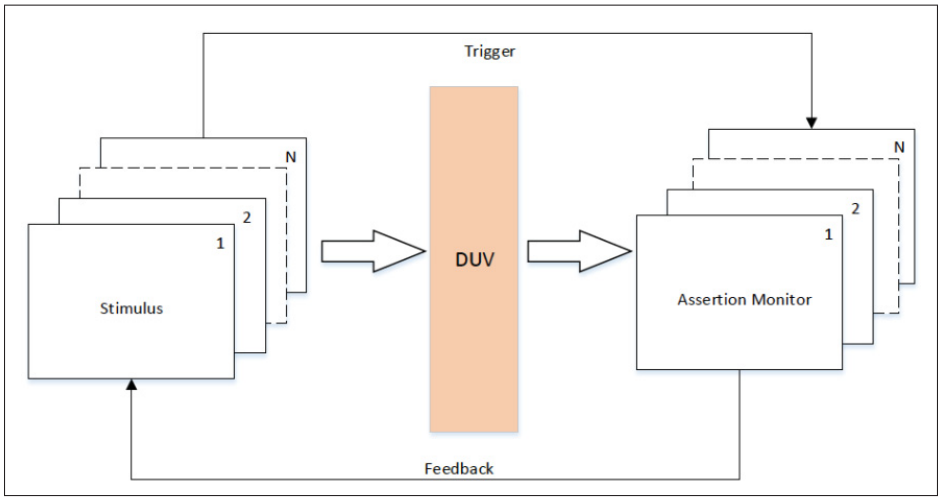

Figure 2.1 Assertion Based Verification Environment

A specific assertion is constructed to monitor a desired property specification for a given DUV. The later is stimulated using a stimulus as shown in Figure 2.1. Both the stimulus and the monitor can be precisely specified using AMS hardware description languages (AMS HDLs). The overall environment allows to conduct simulations of the DUV to achieve runtime verification (Foster & Krolnik, 2010). The feedback and trigger signals are used as communication signals between the stimulus generator and the monitor. This communication mechanism can be performed in an automated fashion and serves to guide the selection of test cases during regression testing.

Simulation-based verification plays a significant role as part of functional verification methodologies. While it remains the predominant verification technique, formal verification methods started lately gaining attention for proving the correctness of AMS designs.

2.2.2 Formal Verification Techniques

In contrast to simulation, formal verification aims to prove the correctness of a design for all possible input signals and initial conditions. It uses mathematically rigorous techniques to exhaustively verify the design without any need for a test bench or input stimuli. This has the advantage of removing uncertainty on corner cases that might escape traditional simulation techniques, but at the cost of increased complexity of analysis. Despite the important progress achieved in the digital domain, the application of formal methods to the analog domain is still hindered by some fundamental problems inherent within the continuous nature of analog signals. In fact, formal methods for the verification of AMS designs are compelled to deal with an infinitely large state space due the presence of continuous state variables such as voltages and currents. The commonly used formal verification method to cope with infinite state spaces is theorem proving (Cyrluk *et al.*, 1995).

Theorem proving methods were developed to prove design specifications using formal deduction based on a set of inference rules. Both the design under verification and its target specification are expressed as formulas in some mathematical logic. While there has been some success (Peng & Greenstreet, 2015), such deductive methods require a significant amount of human expertise and interaction making their application to complex designs very difficult and time consuming.

As an alternative, automated state space exploration methods can be applied to check conformance of relatively small designs. State space exploration is an essential approach to verification of finite-state systems. There are basically two categories of state space methods: equivalence checking and model checking methods.

Equivalence checking methods are used to determine whether two system models are functionally similar with respect to their input-output behavior (Kropf, 1999). The two models could be at the same or different levels of abstraction. Equivalence between models can be based on specific properties such as transient or steady state response properties in the time domain or frequency domain. In contrast to theorem proving, these methods do not require to construct a mathematical proof, nevertheless their correctness depends on the exploration and comparison of the reachable state spaces.

In model checking (Clarke *et al.*, 1999), the task is to check whether a system satisfies a given specification or not. The system model is expressed in terms of state transition system describing all its possible behaviors. The specification is described by a temporal logic formula.

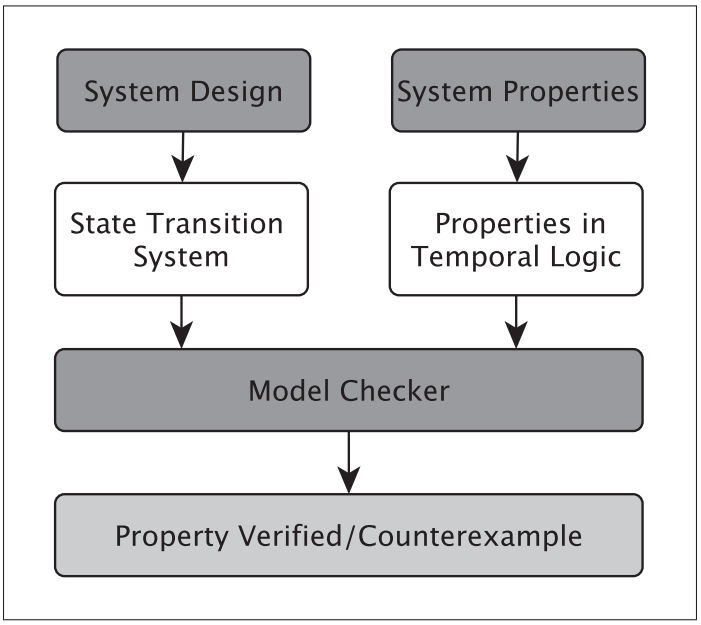

Figure 2.2 A Typical Model Checking System
(Wile *et al.*, 2005)

Given a system model and a specification as inputs, the model checking algorithm (depicted in Figure 2.2) proceeds by exhaustively searching for specification violations in the system state-space. In case the specification is violated, a counterexample describing the failure points is generated.

In summary, the main obstacle in applying state space exploration methods in practice is the problem of state space explosion as the state transition system grows exponentially with the number of state variables. This makes these methods applicable only to designs of small size.

In what follows we will report on the practical application of the previous techniques in the verification of AMS designs.

2.2.3 Relevant Work

Verification approaches for AMS designs have been primarily developed in theory and in practice at the transistor level (Kundert *et al.*, 1988, 1989, 1990; Buhler *et al.*, 2006). Part of these approaches were concerned with the verification in the presence of different kinds of constraints. For instance, the effects of noise (Paper *et al.*, 2005), fluctuations and technology variations (Ankele *et al.*, 1989) were investigated in (Kundert *et al.*, 1990). Further constraints related to the manufacturing steps such as *local oxidation, photolithography, ion implantation, and etching* were considered in (Buhler *et al.*, 2006).

With respect to our main focus, the main drawback of such techniques is that circuit analysis at the transistor level is done late in the design process and cannot achieve high efficiency in detecting specification errors. Moreover, circuit analysis at the transistor level is very costly in terms of time and memory resource allocation, which make them less attractive for early design exploration. In order to tackle these challenges, a new trend of design methodology (Horowitz, 2011) has complemented the traditional transistor level verification with the behavioral level modeling and verification at a higher level of abstraction. High-level models of AMS designs, written in high level languages like Matlab or C, are flexible for modification and easy to maintain in addition to the fact that they allow much faster preliminary simulation.

Interesting attempts to verify AMS design specifications at high level of abstraction were reported in (Pichon *et al.*, 1995; Navin *et al.*, 1997; Gerlach & Rosenstiel; Bonnerud *et al.*, 2001; Bjornsen & Ytterdal, 2003; Dubikhin *et al.*, 2016).

Specifications monitoring of AMS designs using assertions was proposed in (Mukhopadhyay *et al.*, 2009), where the authors proposed an approach for integrating analog assertions into existing commercial simulators. A similar assertion-based verification approach was proposed in (Riordan & Bhattacharya, 2012). Assertions languages such as Property Specification Language (PSL) (Committee *et al.*, 2005a) and System-Verilog Assertions (SVA) (Committee *et al.*, 2005b) were extended to SPICE circuit simulator to facilitate the transfer of specifications across multiple circuit representations. The approach was used to model and verify the behavior of several benchmarks.

Simulation was complemented by symbolic methods in (Al Sammane *et al.*, 2007), where the authors proposed to build property observers from PSL specifications to check simulation traces of discrete-time designs. The approach was implemented in Mathematica and Matlab. It was used to verify the stability specification of a Sigma-Delta ($\Sigma\Delta$) modulator and the PLL locking time property of a frequency synthesizer. In (Havlicek & Little, 2011), the authors proposed real-time extensions to SVA regular expressions. The extensions were built on the already existing definitions of Timed Regular Expressions (TRE) in (Asarin *et al.*, 2002). Application of TRE to mixed-signal specifications, with the notion of feature-indented assertions, was presented in (Ain *et al.*, 2016). Standard features such as rise time, peak overshoot and settling time were formally expressed and evaluated in a simulation-based environment. In a similar spirit the authors in (da Costa & Dasgupta, 2015; Ain & Dasgupta, 2015) proposed an approach for quantitative evaluation of mixed-signal specifications expressed as regular expressions. In (da Costa & Dasgupta, 2015), emphasis was placed on monitoring systems modeled as hybrid automaton using formal methods.

Several theoretical and practical tools for the modeling and automatic monitoring of mixed-signal circuit simulations were proposed in (Ferrere, 2016). In the same context, monitoring specifications expressed in real-time temporal logic such as Metric Temporal Logic (MTL) (Koymans, 1990) or Signal Temporal Logic (STL) (Maler & Nickovic, 2004) was studied in (Maler & Nickovic, 2004) for continuous-time simulation traces, and (Thati & Rosu, 2005) for discrete-time traces. The monitoring and simulation in (Maler & Nickovic, 2004) was carried

out in Matlab/Simulink environment (The Mathworks Inc., 2011). Applications of this work to AMS designs are summarized in (Maler & Ničković, 2013). A similar solution was presented in (Mukherjee *et al.*, 2012), where additional problems related to synchronizing a checker for AMS assertions with the AMS simulator was studied.

In (Nickovic & Maler, 2007), the authors proposed an analog monitoring tool (AMT). The idea was to synthesize signal temporal logic (STL) into timed automata (Maler *et al.*, 2005, 2006). The latter was used to monitor simulation traces of analog signals in an offline or incremental fashion. The verification of a DDR2 SDRAM memory using AMT was proposed in (Jones *et al.*, 2008). Further applications of AMT to evaluate the simulation traces of a DSI3 protocol implementation in an automotive airbag system was presented in (Nguyen & Ničković, 2016).

Monitoring the transient response of nonlinear analog circuits using extended temporal logic, Computational Tree Logic (Hafer & Thomas, 1987) for analog circuit verification, was presented in (Dastidar & Chakrabarti, 2005, 2007). A finite state machine (FSM) model, constructed by means of repeated SPICE simulations, was used to capture the circuit behavior under all possible input waveforms. In (Frehse *et al.*, 2006b), an online monitoring technique was proposed. The work is concerned with the verification of oscillator circuits specifications. Linear hybrid automata (LHA) was employed as a monitor to analyze the reachability of time domain features. A different online monitoring approach for analog systems was implemented in (Zaki *et al.*, 2006). The authors used interval based methods and automata theoretic approaches to prove the system specifications within a fixed time period of the interval arithmetics simulation. The methodology was used to verify the oscillation property of a tunnel diode oscillator.

A more recent approach to verify the stability specification of a third-order modulator using affine arithmetic was proposed in (Radojicic & Grimm, 2016). The approach was used to verify the impact of variations in parameters, inputs, or initial conditions on particular specifications. Another approach using affine arithmetic was proposed in (Grabowski *et al.*, 2006a) to tackle

the problem of process variation and device mismatch. It was applied to an analog bandpass filter and the results were compared to a Monte Carlo simulation.

Focused on the verification in the presence of noise, researchers in (Ankele *et al.*, 1989; Best, 2003; Kundert, 2003; Kundert & Chang, 2006; Wang, 2009; Kundert & Chang, 2009; Jaykar *et al.*, 2011) presented a detailed analyses of noise (e.g., sampling jitter, and kT/C noise) at a high level of abstraction. Likewise, attractive contribution through a time-domain numerical integration techniques for behavioral noise analysis were presented in (Thain & Connelly, 1995; Demir, 1997; Mathis & Thiessen, 2009). In a similar numerical approach, the effects of noise (e.g., thermal, shot and flicker) combined with process variation were investigated in (Narayanan *et al.*, 2009, 2010b, 2013). Process variation effects, on the other hand were analyzed in a statistical run-time verification environment in (Seghaier *et al.*, 2015).

In a different approach, the authors in (Narayanan *et al.*, 2010a) used the MetiTarski toolset to verify saturation specification of an Op-Amp in the presence of noise and process variation. MetiTarski is an automatic theorem prover for real-valued elementary functions such as ln, exp, sin, cos, etc. Similar research was done in (Denman *et al.*, 2009), where MetiTarski was used to verify properties of analog circuits namely, a tunnel diode oscillator and an operational amplifier. Theorem proving was initially used to verify the specifications of non-ideal logical circuitry in the analog domain in (Hanna, 1994). In an attempt to automate the proposed approach, the author opted instead for constraint based techniques (Hanna, 1998). PVS theorem prover was adopted in (Ghosh & Vemuri, 1999) to check the functional equivalence between synthesized VHDL-AMS designs and their behavioral specifications. The approach is concerned with DC and low frequency behaviors. In similar but more elaborated fashion, the author in (Hanna, 2000) proposed an approach for specifying and reasoning about implementations of digital designs presented at the analog level of abstraction. The behavior of devices like diodes and resistors is characterized by conservative approximation methods based on piecewise-linear predicates on currents and voltages.

To overcome the incompleteness of simulation and the complexity of formal methods, the authors in (Steinhorst & Hedrich, 2010b, 2011) proposed a property verification and equivalence checking methodology for analog circuits based on formal automatic input stimuli generation. The methodology enables transient simulation using an efficient state space-guided input stimuli generator covering the circuit's complete dynamic behavior. It was applied to a Sallen-Key biquad lowpass filter.

Focused on equivalence checking of analog circuits with strong nonlinearities, researchers in (Steinhorst & Hedrich, 2010a) proposed an approach based on canonical state space transformation. Conformance between two systems under verification is performed by numerically comparing their transformed system functions in the canonical state space. A failure occurs if the comparison reveals that the results are different with respect to an initially predefined error value. Similarily, in the approaches to equivalence checking of analog circuits (Hedrich & Barke, 1995; Hartong *et al.*, 2004; Zaki *et al.*, 2008), sampling methods were used to check conformance between nonlinear analog circuits.

Another equivalence checking approach for functional verification of VHDL-AMS designs was presented in (Salem, 2002). It is based on equivalence checking, rewriting systems and simulation combined into one verification environment. Given two VHDL-AMS architectures describing different abstraction levels for the same design, the idea was to divide the specification and implementation code into digital, analog and data converter components. Equivalence of digital parts was performed using formal equivalence checking, while rewriting techniques and name matching were applied to analog components. The outputs were fed to comparators to be checked using simulation.

A more recent equivalence checking approach for AMS circuits was implemented in (Lim *et al.*, 2015). Analog cells were written in SystemVerilog and compared against their implementation at the transistor level, while digital blocks were validated using digital validation tools. The approach was applied to a analog cells of a single-slope ADC and a serial link receiver. While these approaches are concerned with equivalence checking in time domain,

others tend to focus on verification in frequency domain. For example, the approach proposed in (Balivada *et al.*, 1995) allows to prove equivalence between two designs (e.g., specification and implementation) of analog circuits represented by linear transfer functions. This is done by first discretizing the transfer functions to the Z-domain using bilinear transformation which allows to represent the design in terms of digital elements. The resulting discrete realizations are encoded into Ordered Binary Decision Diagrams (OBDDs) and compared using techniques for checking compatibility of states in finite state machines. This work was extended in (Seshadri & Abraham, 2001) to verify conformance between the transfer functions under the influence of parameter variations using global optimization techniques. The approach avoids loss of precision due to the bilinear transformation. On a similar subject, equivalence checking with parameters variations was investigated in (Hedrich & Barke, 1998).

Apart from equivalence checking methods, model checking was also used to validate AMS specifications. It was primary applied to verify digital designs at the transistor level in (Kurshan & McMillan, 1991). The developed method introduced an a priori abstraction of the state space based on partitioning the continuous state space into fixed size multidimensional cubes which is computationally expensive. Instead, the authors in (Greenstreet & Mitchell, 1999; Yan & Greenstreet, 2007) advocate the use of discretization and projection techniques of the state space to reduce its dimension while maintaining an over approximation of the circuit dynamic behavior. The idea was adapted in a series of works (Dang *et al.*, 2004; Gupta *et al.*, 2004; Frehse *et al.*, 2006a). These approaches were tailored to verify the specifications of several benchmarks namely, a biquad low-pass filter (Dang *et al.*, 2004), a tunnel diode oscillator (Gupta *et al.*, 2004) and voltage controlled oscillators (Frehse *et al.*, 2006a).

Research activities concerned with the verification of AMS designs were proposed in (Bemporad & Morari, 1999; Dang *et al.*, 2004; Gupta *et al.*, 2004; Freibothe *et al.*, 2006). In a more recent example (Althoff *et al.*, 2011), the authors proposed an approach for the verification of transient and invariant specifications of a charge-pump PLL. The latter was modeled using hybrid automaton with linear continuous dynamics and uncertain parameters. The idea behind this approach is to compute accurate over-approximations of reachable sets using uncertain

parameters to represent the range of possible switching times. It was applied to the verification of the PLL locking time and stability specifications. An upper bound on the worst-case lock time in the presence of random phase error and charge pump current variations was provided. A different effort for using reachability analysis methods in the verification of AMS systems was proposed in (Fisher *et al.*, 2014). The authors developed a model checking tool (LEMA) using both explicit zone-based methods (Little *et al.*, 2011) as well as implicit BDD and SMT-based methods (Walter *et al.*, 2008). The specifications were expressed using the Language for Analog/Mixed-Signal Properties (LAMP) (Fisher *et al.*, 2015). The AMS circuit behavior was modeled with a labeled Petri net (LPN) model (Walter *et al.*, 2008; Little *et al.*, 2011). LEMA tool uses model generation techniques to generate the circuit LPN model from simulation traces (Little *et al.*, 2010; Batchu, 2010; Kulkarni, 2013). It was used to model and verify several AMS designs such as DACs, phase interpolators and voltage controlled oscillators.

Verification of time constraints of analog signals (e.g., rise time, fall time) using model checking algorithms was proposed in (Grabowski *et al.*, 2006b). The authors proposed extending their previous method for model checking of integrated analog circuits to take into account time constraints. Extensions were concerned with the development of the analog specification language ASL (Steinhorst *et al.*, 2006) tailored to express analog circuit specifications like gain, rise time, offset and slew rate.

In summary, there are a variety of methods for the verification of functional specifications of AMS designs including simulation, semi-formal and formal methods. Within these were specific methods for modeling and verification in the presence of various parasitic effects such as noise, manufacturing variations and environment constraints at a high level of abstraction. Table 2.1 summarizes the main characteristics of important works among the above mentioned ones. The table presents the class of systems verified, the models used, the type of constraints considered, the monitoring methods, analysis domains, the verification technique, the tools used, and the case studies verified.

Table 2.1 Verification Techniques for 2D ICs.

(a) Summary Table

	(Nickovic & Maler, 2007)	(Al Sammane *et al.*, 2007)	(Mukhopadhyay *et al.*, 2009)	(Narayanan *et al.*, 2009, 2010b, 2013)	(Riordan & Bhattacharya, 2012)
Type of Systems	Analog	AMS	AMS	Analog/RF	AMS
Models	Behavioral	Behavioral	Schematic/Behav.	Behavioral	Multiple Abstraction Levels
Type of Constraints	-	-	-	Environmental, Noise, Process Variation	-
Monitoring Type	Offline/Online	Offline	Online	Online/Offline	Online
Analysis Domain	Time	Time	Time	Time	Time
Verification Technique	Simulation	Simulation	Simulation	Simulation	Simulation
Tools	AMT & Matlab	Matlab	Cadence/Synop.	Matlab	Spice
Case Studies	Flash Memory	PLL, ΔΣ Mod	Power Management Unit	Tunnel Diode, Colpitts Oscillator, PLL-based Freq. Synthesizer	ADC

(b) Summary Table (Cont')

	(Mukherjee *et al.*, 2012)	(Seghaier *et al.*, 2015)	(Walter *et al.*, 2008)	(Althoff *et al.*, 2011)	(Little *et al.*, 2010; Kulkarni, 2013; Fisher *et al.*, 2014)
Type of Systems	AMS	AMS	AMS	AMS	AMS
Models	Behav./Transistor Level	Behavioral	Behavioral	Behavioral	Behavioral
Type of Constraints	-	Process Variation	-	-	-
Monitoring Type	Online	Offline	-	-	-
Analysis Domain	Time	Time	Time	Time	Time
Verification Technique	Simulation	Simulation	Model Checking	Reachability Analysis	Model Checking
Tools	Cadence AMS	Matlab	ATACS	MATLAB	LEMA
Case Studies	Clock Generation Unit	Ring Oscillator, Charge Pump PLL	Switched Capacitor Integrator, PLL	Charge Pump PLL	DACs, Phase Interpolators, VCOs

To the best of our knowledge, no empirical research exists addressing the impact of additional specification and test requirements on the AMS design performance at early design stages, during the verification phase, as a means to systematically make sure that specifications are complete and well understood. In this thesis, we take the verification process a step further by introducing two types of constraints: specification-related constraints and test-related constraints. Next, we develop a unified modeling and verification methodology to assess the impact of these additional constraints on the AMS design's overall performance. We apply our methodology to a commercial frequency synthesizer by (Freescale Semiconductor, 1999). With reference to traditional simulation-based approaches, in particular (Mukhopadhyay *et al.*, 2009; Mukherjee *et al.*, 2012; Mukherjee & Li, 2016), we differentiate ourselves by integrating additional specification and test-related constraints within the verification approach at an early design stage. Moreover, we provide an automated mechanism to guide the verification process, leveraging these constraints to gain a better analysis of each design specification and consequently to keep track of any inconsistent, incomplete or ambiguous specification information and to enable early specification error detection.

2.3 System Verification Techniques for Three-dimensional Integrated Circuits

3D ICs have emerged as an attractive solution to get rid of the planar constraint from IC design (Banerjee *et al.*, 2001; Das *et al.*, 2004). The performance and cost savings of such ICs are presented in (Ferri *et al.*, 2007; Dong & Xie, 2009). Indeed, the 3D IC technology enables shorter interconnect wires and improves performance and density, in addition to the easy reuse of IP blocks, heterogeneous technology integration and a reduction of the form factor. Technologies presently used for 3D integration incorporate: wire bonding, microbump, and through-silicon vias (TSV) that combine different dies in a single stack die stacking, (Davis *et al.*, 2005; MIT Lincoln Labs., 2006). Die stacking allows heterogeneous integration of circuits of different materials (e.g. RF CMOS, SiGe) (Xue *et al.*, 2003). This greater flexibility, however, comes at the price of increased verification challenges.

2.3.1 Relevant Work

Recently, significant research effort has been invested into the area (Saraswat *et al.*, 2000; Rahman, 2001; Huang & Lei, 2002; Banerjee *et al.*, 2001; Xie *et al.*, 2006; Cong *et al.*, 2006; Hogan & Petranovic, 2009; Petranovic & Chow, 2011; Yahalom, 2016). The simplest approach (Petranovic & Chow, 2011), is to first perform design rule check (DRC), layout versus schematic (LVS), and parasitic extraction (PEX) on the individual dies separately, and then to consider die-to-die interfaces. Some additional parasitics were considered, namely, capacitive and inductive coupling between dies (Lee *et al.*, 2009; Chuang *et al.*, 2013; Koo *et al.*, 2013), TSV-to-TSV noise coupling (Xu *et al.*, 2010; Song *et al.*, 2011; Liu *et al.*, 2011; Gope *et al.*, 2013), noise coupling between TSVs and other devices (Khan *et al.*, 2009; Lin *et al.*, 2012; Lee *et al.*, 2012), parasitics associated with microbumps as well as local variation of device parameters (Selvanayagam *et al.*, 2009; Matsumoto *et al.*, 2013; Santos *et al.*, 2014). Furthermore, noise coupling between noisy digital blocks and sensitive analog circuits is outlined and considered in (Yahalom, 2016). The impact of process variations on the performance of 3D ICs is also introduced and studied in (Ferri *et al.*, 2007; Garg & Marculescu, 2009; Ozdemir *et al.*, 2010; Xu *et al.*, 2012).

In addition to the verification issues, another set of issues, to be addressed very early in the design process, is coming from the test process (Lee & Chakrabarty, 2009; Petranovic & Chow, 2011; Chung, 2012). In fact, the move towards 3D technology is hindered by a misunderstanding of 3D testing issues and by the absence of adequate test techniques (Lee & Chakrabarty, 2009). Industrial experts have reported various test challenges, namely, the lack of probe access for wafers, inadequate test access to internal stacking modules, design testability, thermal concerns, test economics and additional defects resulting from unique processing steps (e.g., wafer thinning, alignment, bonding) (Lee & Chakrabarty, 2009). The 3D IC test process involves two main phases : pre-bond testing and post-bond testing (Lee & Chakrabarty, 2009; Marinissen & Zorian, 2009; Cho *et al.*, 2011; Noia & Chakrabarty, 2011). The former allows detection of defects that are caused by the TSV process itself, while the later allows detection of faults caused by thinning, alignment, and bonding (Lee & Chakrabarty, 2009; Marinis-

sen & Zorian, 2009; Marinissen, 2013; Minas *et al.*, 2010). Pre-bond testing, in particular, presents several new challenges (Lee & Chakrabarty, 2009; Marinissen & Zorian, 2009; Gambino *et al.*, 2015). In particular, a die in 3D stacks might include only partial logic and not completely functional circuits which restricts the number of tests to be performed to any single layer with partial circuits. We tackle this last challenge through early definition of test(s) for each design specification; as we extract their specification and test-related constraints and incorporate them into the verification process to gain a better understanding and analysis of each test and consequently to ensure that each die is testable prior to bonding.

Similar to the 2D ICs case, there was no attempt to consider both specification and test-related constraints during the verification of 3D ICs in order to ensure that the specifications are correct, complete and unambiguous. We provide a first attempt to assess the impact of these constraints on 3D ICs performance, while monitoring their specification properties. We apply our verification methodology to the 3D clock tree example used in (Zhao *et al.*, 2011) and we show that some specifications may degrade or even cannot be verified without adding specific specification and test-related constraints.

2.4 Summary

In this chapter, we presented a summary of the main verification techniques and their application to 2D and 3D mixed-signal IC designs. A look at the literature shows that there are multiple approaches to modeling, analysis and verification that differ in several respects. In broad terms, the difference is in the emphasis on system models and on whether or not they emphasize analysis and verification results.

The various research directions in mixed-signal systems resulted in different perceptions towards modeling and verification issues. At one end of the spectrum, there are simulation-based approaches that remain predominantly used to validate a design, which we adopt in this thesis. Generally, these approaches are capable of dealing with complex systems but do not consider its exhaustive behavior. At the other end of the spectrum, there are formal-based approaches that offer a more accurate and rigorous analysis but suffer from the state space explosion problem and are applicable only to designs of moderate size. There are additional approaches spanning the rest of the spectrum that combine simulation with formal-based methods. Unfortunately, formal methods are still lagging in industry behind simulation-based verification, in particular for mixed-signal designs, which makes them less attractive to us.

Current approaches to high-level system modeling and simulation provide effective means for analyzing the behavior of a mixed-signal design. Nevertheless, in reality, they do not guarantee that the design will maintain the same behavior with the effects of specification and test-related constraints.

In the next chapter, we will present our unified modeling and verification methodology for specification and test-related constraints analysis in 2D and 3D mixed-signal designs.

CHAPTER 3

PROPOSED VERIFICATION METHODOLOGY

This chapter presents a constraint-driven verification methodology for AMS designs. The methodology is based on the early insertion of test(s) associated with each design specification. It allows to model and verify an AMS design with specification and test-related constraints. These constraints are extracted from the design preliminary datasheet and used to drive our verification methodology. First, we consider test-related constraints as additional requirements, such as additional test points or special interface circuits, needed to make the test cost effective or simply feasible. Second, we enlarge on this idea to handle mixed-signal tester's instrument characteristics that are also introduced as input constraints. A detailed description of the proposed methodology including its major phases will be presented.

3.1 Introduction

In order to achieve a robust and effective functional verification, there are specific initial requirements which must be targeted. Current verification methodologies have been shown to be useful for detecting and eliminating AMS design failures. Though, poor "first time" design success illustrates the lack of cohesive and efficient techniques to allow a predictable verification process that leads to the highest possible confidence in the correctness of AMS designs. A key issue not yet addressed in the current state of the art methodologies is a lack of early and efficacy account for specification testing requirements. One aspect of testing which is usually beyond the scope of functional verification, is the presence of additional requirements. These include, for example, adequate interfaces between the tester and the DUT or extra test access points for function performance measurement. Moreover, other challenges may arise from the impact of various parasitic effects such as noise and operating environment conditions. To tackle this, we propose in this chapter a new constraint-driven verification for monitoring specifications of AMS designs in the presence of specification and test-related constraints. The monitoring is done using assertion based runtime verification technique. This is carried out

in an offline fashion by first running the complete simulations and then monitoring the AMS design specifications on offline simulation traces.

3.2 Constraint-driven Verification Methodology

The idea behind our methodology is to integrate both specification and test-related constraints at a high level of abstraction, that is throughout the verification process. Although test is the last stage shown in the design process, it should be defined as soon as the IC high level implementation is defined which forces the definition of at least one test scheme covering each of the IC specifications. This early definition of test schemes serves two purposes: 1) it prevents not considering test and specification-related constraints during the design phase, such that the tests cannot be adequately performed or are not cost effective, and 2) it helps in the early discovery of missing, incomplete or misunderstood specifications, by forcing a detailed analysis of all specifications.

As mentioned in chapter 1, a specification-related constraint is defined as the set of {performance, condition}, that is the overall design performance (power, speed, etc.) to be reached during test for a given set of environmental conditions (voltage, temperature, etc.). Moreover, a test-related constraint is defined as any additional requirement coming from the test itself. It may be an additional sub-circuit or a test access point which may be incorporated into the design structure to make the test simply feasible. In this thesis we focus on physical test-related constraints (such as the interface between testers and the device under test, input-signal conditions, input impedance, etc.) as well as those that can affect the design. We mean by test-related constraints affecting the AMS design those implemented by adding specific components or sub-circuits to the design. These constraints are deemed necessary to achieve adequate specification tests.

3.2.1 Monitoring specifications of AMS designs in the presence of specification and test-related constraints:

The overall constraint-driven verification methodology is illustrated in Figure 3.1. It consists of three major phases: constraints extraction, modeling and verification.

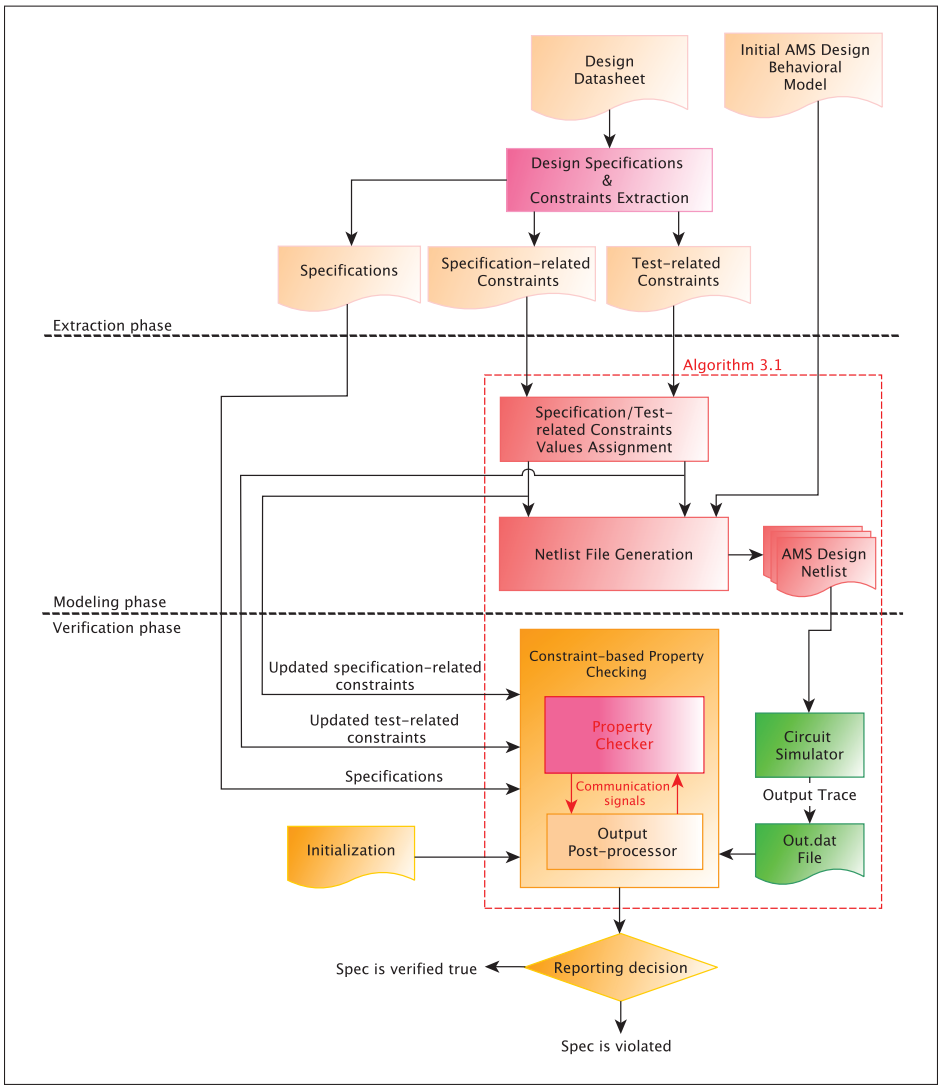

Figure 3.1 Constraint-driven Verification Methodology

We have developed a two-phase algorithm (Algorithm 3.1) that takes care of the automated modeling and verification tasks. The implementation of these tasks is described in Algorithm 3.1.

Algorithm 3.1 Constraint-based Modeling and Verification Algorithm

```
   Require: A_M, 𝕊, ℂ ;          /* A_M: AMS design model, 𝕊:{S_i}_{i=1..N} set of
   specifications, ℂ:{C_i}_{i=1..N} set of specification and test-related
   constraints */
 1 for each specification S_i of 𝕊 do
   |  ------------------------Modeling phase------------------------
 2 |  C_i = Specification/TestRelatedConstraintsValuesAssignment(S_i);
 3 |  GenerateCircuitNetlist();
   |  ------------------------Verification phase----------------------
   |  /* Constraint-based Property Checking starts here:    */
 4 |  O_T = Circuitobj.GetCircuitOutput(node_i);
 5 |  O_Tp=OutputPost-processor(O_T);
 6 |  PropertyChecker(O_Tp, S_i, C_i);
 7 |  status=UpdateStatus();
 8 |  if ∼ status then
 9 |  |  S_i ← success;
   |  else
10 |  |  S_i ← fail ;                  /* a violation signal is generated */
   |  end
11 |  Graphics() ;                                   /* Plot results */
   end
```

In the following sections, we elaborate on each of the extraction, modeling and verification phases.

3.2.1.1 Extraction Phase

The first step in our methodology is to manually extract the design specifications, specification-related constraints as well as the specifications test schemes from the design preliminary datasheet (see Figure 3.2). Moreover, relevant test-related constraints are manually extracted from the predefined test schemes. We call this step a 'pre-verification' step as we introduce additional specification and test-related constraints and as we have to check whether these constraints are defined or not for a target specification.

For each design specification, we extract a set of specification and test-related constraints that can differ from one specification to another. This is because each specification has its own

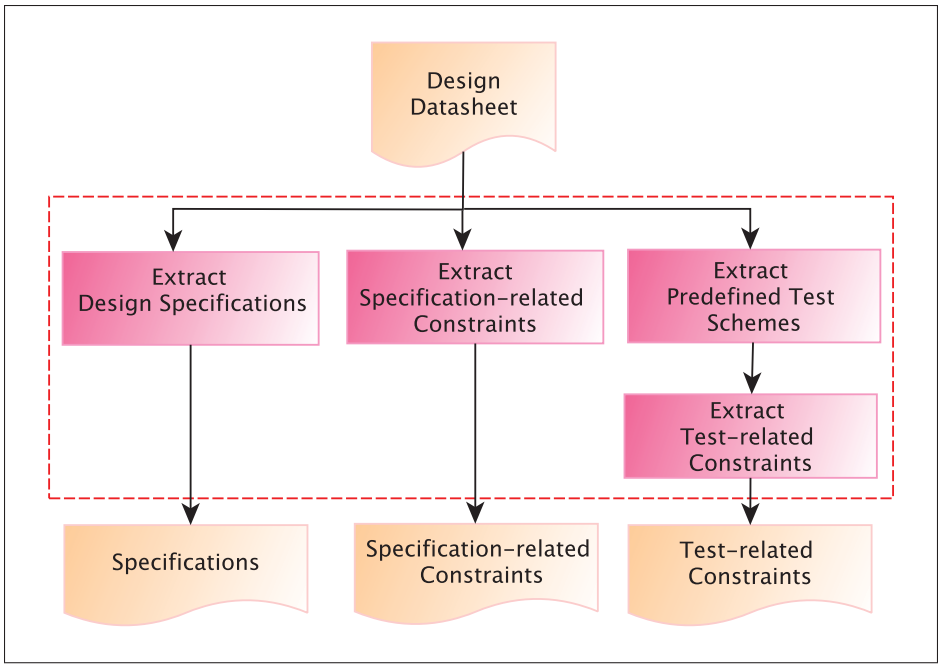

Figure 3.2 Design Specifications & Constraints Extraction

test scheme that is not necessarily the same for all the design specifications. The result of the extraction phase is the design system specifications $System_{Specs}$ which can be formalized as follows:

$$System_{Specs} = \begin{cases} S_1 + C_1 : \{ \delta_1, \Gamma_1 \} \\ S_2 + C_2 : \{ \delta_2, \Gamma_2 \} \\ . \\ . \\ . \\ S_N + C_N : \{ \delta_N, \Gamma_N \} \end{cases}$$

where S_i and C_i represent, respectively, the design specification and the set of extracted constraints (δ_i: specification-related constraints, Γ_i: test-related constraints); i $\in$ [1, N].

At this point, the extracted specification and test-related constraints are either defined using initial nominal values or simply missing. These constraints must be assigned appropriate values in order to allow adequate specification test. As explained next, an example of such an

assignment would be to use values at the limits of the tolerance for a given parameter. This assignment task is done during the subsequent modeling phase.

3.2.1.2 Modeling Phase

The modeling phase iterates between two steps: constraints values assignment and netlist file generation.

- Constraints Values Assignment:

In this step, each constraint is assigned a specific value. Indeed, during the extraction phase, it may be determined that adequate values for some specification and test-related constraints are not clearly specified or not provided in the design preliminary datasheet. The reason behind this is usually specification and test-related constraints are not allocated higher priority before the test phase. For some type of constraints, which have not been adequately, it could be too late to make the necessary changes during the test phase. Examples include specification and test-related constraints that can affect the design (illustrative examples of such constraints, will be presented in chapter 4).

To complement the definition of certain constraints values, we perform a worst-case tolerance analysis. Through this analysis, we identify the tolerance levels on each of these constraints. This is useful to predict, in an early design phase, the circuit behavior under extreme test conditions (e.g., operational temperature levels, random noise levels, etc.).

Tolerance limits define the range of values in which specifications are guaranteed to pass the test. These limits are obtained by changing each constraint's value in predefined percentage increments or decrements. The implementation of this task is described in Algorithm 3.2.

Once the required tolerance limits defined, we assign a value within these limits to each constraint (line 2 in Algorithm 3.3). In doing so, we provide the set of assigned specification and test-related constraints that will be used in the next netlist file generation step.

Algorithm 3.2 Constraints Tolerance Limits Variation

Require: C_i, *Delta* ; /* C_i: Specification and test-related constraints, *Delta*: percent variations from the initial constraints nominal values*/
1 **for** *each constraint c of* C_i **do**
2 | $[lower_{limit}, upper_{limit}]$ = *ComputeToleranceLimits* (*c*, *Delta*(*c*))
3 | ***return*** $[lower_{limit}, upper_{limit}]$
end

Algorithm 3.3 Constraints Values Assignment

Require: C_i, $Tolerance_{Array}$; /* C_i: Specification and test-related constraints, $Tolerance_{Array}$: Set of constraints tolerance limits intervals */
Ensure: $C_i \subset Tolerance_{Array}$
1 **for** *each constraint c of* C_i **do**
2 | $c = AssignValueWithinToleranceLimits\ ()$
end
3 ***return*** C_i

- Netlist File Generation:

Based on the initial AMS design behavioral model (provided by the designer) and the extracted specification and test-related constraints, a netlist file that represents the AMS design behavioral model in the presence of specification and test-related constraints is automatically generated using *GenerateCircuitNetlist()* function (line 3 in Algorithm 3.1). The detailed implementation of this function is outlined in Algorithm 3.4.

Indeed, during netlist generation, the algorithm operates in two modes as illustrated in Figure 3.3 :

- 1. A read mode (lines 1-2 in Algorithm 3.4) that reads the input files containing the initial AMS design behavioral model and the extracted specification and test-related constraints.

- 2. A create mode (lines 3-10 in Algorithm 3.4) during which the algorithm integrates the extracted specification and test-related constraints into the initial AMS design behavioral model.

  Constraints integration requires the position at which additional specification and test-related constraints should be inserted (*insertPosition* parameter). This is determined a priori using a *Search* function that finds the position of a target constraint within the initial AMS design behavioral model. Next, based on the *insertPosition* parameter, the algorithm either copies the content of the initial AMS design behavioral model (lines 5-6 & lines 8-9 in Algorithm 3.4) or simply inserts a given constraint (line 7 in Algorithm 3.4) into the newly created netlist file (line 3 in Algorithm 3.4). This step is repeated until all additional constraints are inserted.

  The output is a file with a specific extension, such as '.cir', that contains the new AMS design model with the additional specification and test-related constraints which is then fed into the circuit simulator, as illustrated in Figure 3.1.

The netlist generation also includes another pre-verification step during which it is checked that a test scheme is defined for each specification. In case there is no test scheme defined for a given specification, one must be elaborated and the associated test related constraints must be extracted for the netlist generation to be successful.

Detecting such missing test schemes is one of the benefits of our approach. The implementation of this pre-verification step is described in Algorithm 3.5.

The pre-verification step depends on two parameters $TEST_{flag}$ and $SELECT$. $TEST_{flag}$ is a control parameter used to make sure that there is at least one test scheme defined for each specification. In this case, based on the $SELECT$ value the algorithm reads either the set of specification-related ($SELECT = 0$) constraints or the set of test-related ($SELECT = 1$) constraints (line 2-5 in Algorithm 3.5). The algorithm then returns the set of constraints (line 6) otherwise an error message indicating a missing test scheme is generated (line 7).

Algorithm 3.4 Netlist File Generation

```
   Require: A_M, C_i, insertPosition ;          /* A_M: AMS design model, C_i:
   Specification and test-related constraints, insertPosition: The position
   where a given constraint should be inserted*/
   Ensure: C_i ⊄ ∅
   ----------Read mode----------
1  ReadInitialDesignModel (A_M);
2  ReadExtractedSpecification/TestRelatedConstraints (C_i);
   ----------Create mode----------
3  CreateNewNetlistFile ();
4  for each constraint c of C_i do
5  |  for j = 1: insertPosition(c) do
6  |  |  CopyInitialDesignModel(A_M(j))
   |  end
7  |  InsertConstraint (c)
8  |  for j = (insertPosition(c) + length(c)) : length(A_M) do
9  |  |  CopyInitialDesignModel(A_M(j))
   |  end
   end
10 return DesignBehavioralModelWithAdditionalConstraints
```

Algorithm 3.5 Netlist File Generation: Pre-verification step

```
   Require: TEST_flag, SELECT
   Ensure: TEST_flag ≥ 1
1  while TEST_flag ≥ 1do
2  |  if SELECT = 0 then
3  |  |  C_i (: , 1)= read (SpecificationRelatedConstraints);
4  |  else if SELECT = 1 then
5  |  |  C_i (: , 2)= read (TestRelatedConstraints) ;
   |  end
6  |  return C_i ;
   end
7  UNDEFINED TEST SCHEME FOR THE GIVEN SPECIFICATION! ;
```

Both the AMS design netlist with and without specification and test-related constraints can be written using any standard circuit simulator such as PSpice A/D (Wilson, 1989). After netlist file generation, we automatically simulate the resulted AMS design netlist.

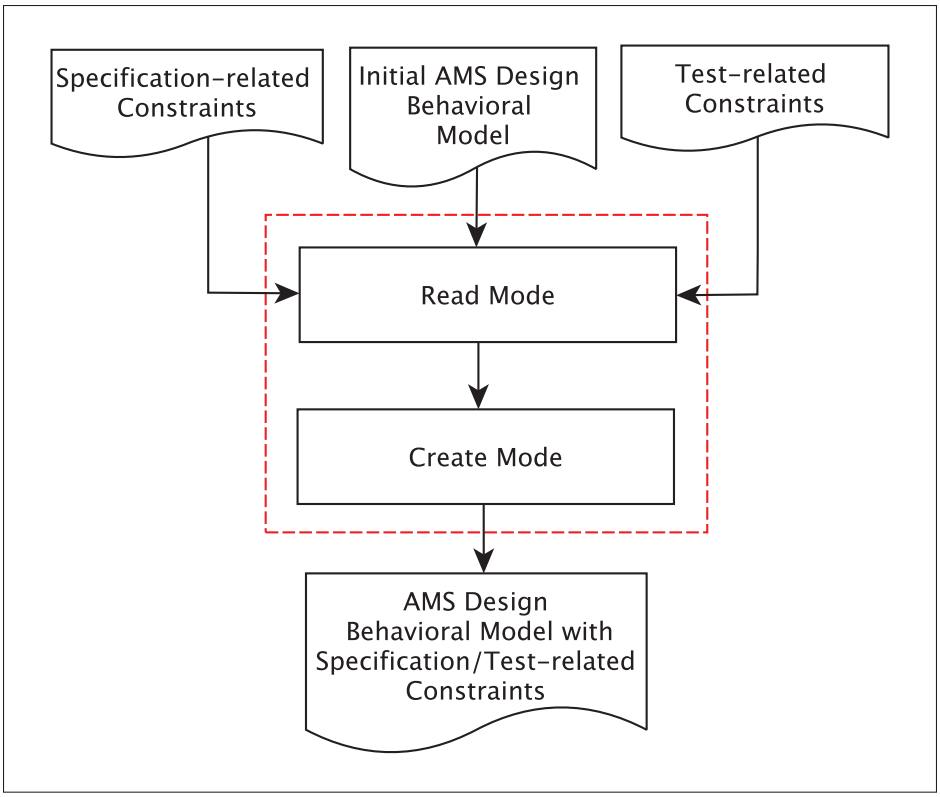

Figure 3.3 Netlist File Generation

3.2.1.3 Verification Phase

Once the modeling phase is complete, a simulation-based verification phase is conducted to assess the impact of the extracted constraints on the AMS design performance. The methodology supports different levels of abstraction and different simulation engines. Here, it is implemented using Matlab and Orcad PSpice environments.

The first step of the verification phase is to simulate the resulting AMS design netlist from the modeling phase, using PSpice. The goal of this simulation is to obtain the circuit output under specification and test-related constraints variations, which will be contained in the created simulation result file (Out.dat). Next a Matlab/Pspice interface is used for post-processing the simulation results from PSpice (lines 4-5 in Algorithm 3.1).

Thereafter, given the AMS circuit output (Out.dat), the set of specifications ($\mathbb{S}$), and the set of specification and test-related constraints ($\mathbb{C}$), we use the constraint-based property checking algorithm (lines 4-11 in Algorithm 3.1) in a Matlab simulation environment. The later is composed of two main sub-algorithms: an output post-processor used for post-processing the simulation results from PSpice and a property checker.

Accordingly, the PSpice output trace (O_T), property specification, specification and test-related constraints, simulation parameters and initializations are evaluated using the constraint-based property checking algorithm as shown in Figure 3.1.

The communication signals between the property checker and the output post processor (see Figure 3.1) are used as trigger signals to start a new monitoring process (a new specification input).

The implementation of the property checker is shown in Algorithm 3.6. *Checkproperty()* function (line 1) evaluates whether or not a target specification S_i is satisfied for a given offline simulation trace O_{Tp} in the presence of the extracted specification and test-related constraints C_i. In fact, it asserts $Monitor_{flag}$=1 if a specification violation occurs. Otherwise, it reports that the property specification is satisfied.

Algorithm 3.6 Property Checker

Require: O_{Tp}, S_i, C_i ; /* O_{Tp}: Post-processor output, S_i: Specification, C_i: Specification and test-related constraints */

1 *Checkproperty*(O_{Tp}, S_i, C_i) ;

2 **return** $Monitor_{flag}$;

Generally, property monitoring can be as simple as watching the circuit output (i.e., voltage, current etc.). Conversely, it can be more complex when a certain amount of post-processing is necessary to make sense of the output data. Depending on how the design output signal should be observed, in an online or offline fashion (Yuan *et al.*, 2006), the property checker is created. Although, online monitoring is more feasible when verifying simple specifications,

offline monitoring allows the verification of more complex specifications at the cost of high memory consumption.

The verification results obtained from the property checker allow us to draw certain conclusions about the design, such as how do these additional constraints affect the AMS design performance, and what are the benefits of addressing them in the early design phases.

3.2.2 Monitoring specifications of AMS designs in the presence of specification and test-related constraints with tester's components characteristics:

In this section we extend the previous constraint-based verification approach to handle additional performance characteristics derived from the mixed-signal tester components. The extended version of the proposed methodology is illustrated in Figure 3.4.

The idea behind the verification approach is that including additional characteristics related to the tester components with specification and test-related constraints makes the specification definition complete and unambiguous. Indeed, not only specification and test-related constraints can impact and alter the AMS design functionality in unexpected way but also the tester hardware. In that, the problem of specification test result divergence, between an ATE and a practical customer environment due to their different tester component characteristics, has become a major concern according to (Nakura *et al.*, 2016). The typical example is the tester's power supply block sometimes too good, compared with a practical customer power supply, which results in test-escapes or less better than the customer power supply which results in yield loss (Arabi, 2010).

In this thesis we also tackle this problem through early definition of the tester characteristics used to ensure that the AMS design specification is met. Our methodology employs specific tester components characteristics, as they are extracted and used during the modeling and verification phases. In case, the solution to compensate for the difference between an ATE and a practical customer environment involves additional interface elements, our methodology allows designers to make the necessary changes during early design phases. This, in turn, reduces

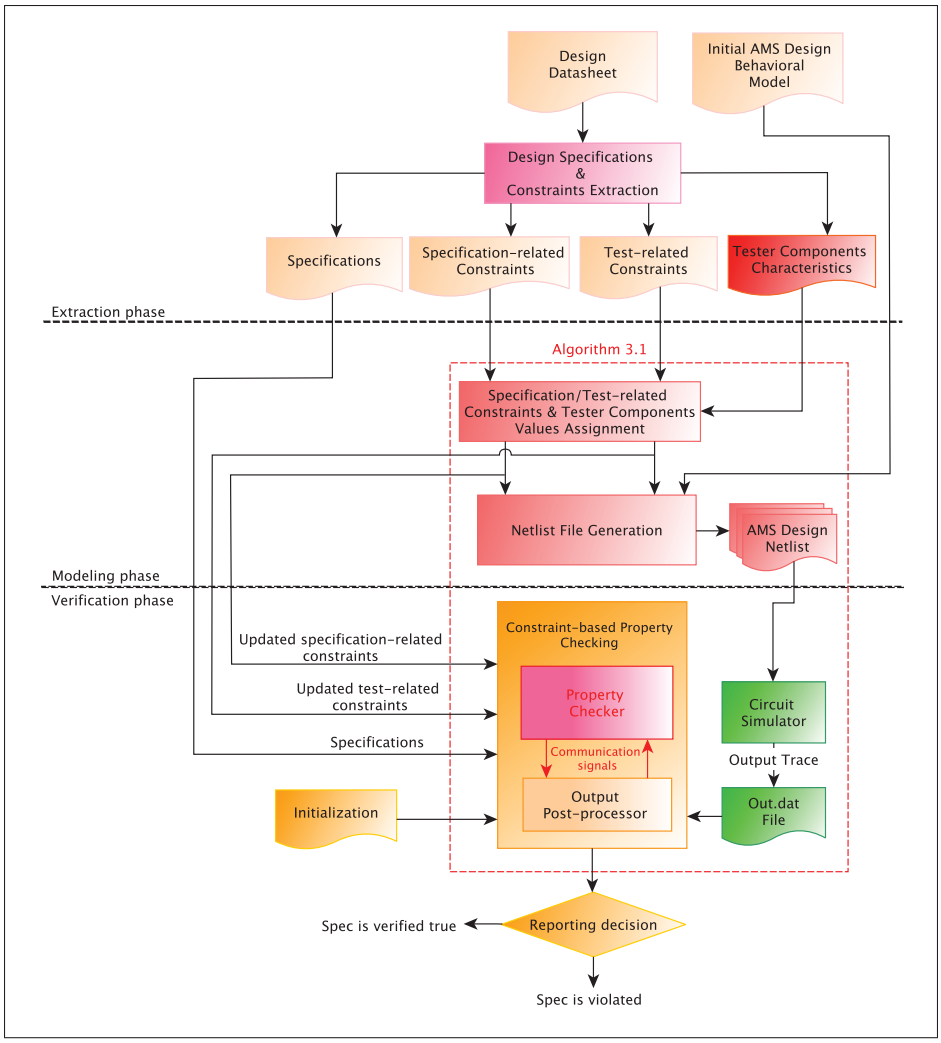

Figure 3.4 Extended version of the constraint-driven verification methodology

the risk of costly errors caused by incomplete, ambiguous or missing details in the specification documents.

3.2.2.1 Extraction Phase

We repeat the previous extraction phase, except we add another step to extract these characteristics as illustrated in Figure 3.5.

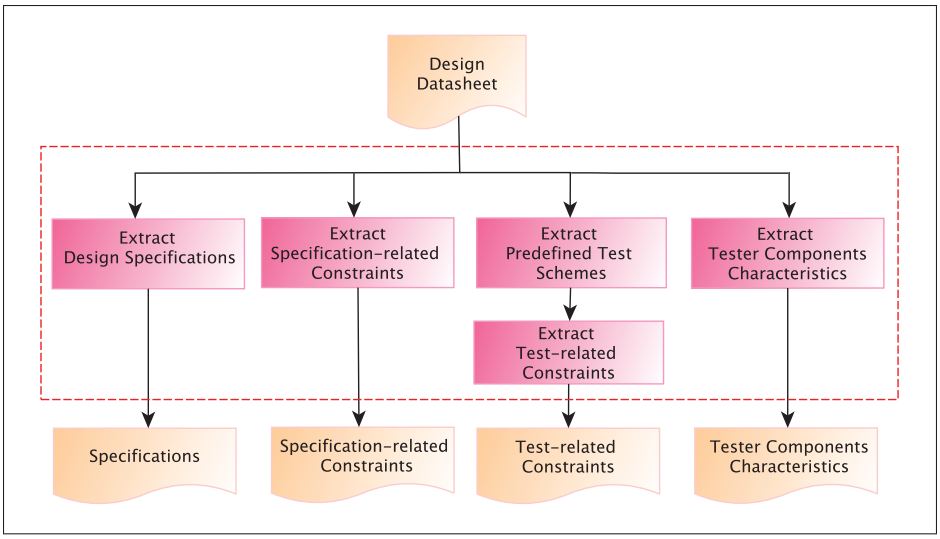

Figure 3.5 Extraction Phase: Extended version

3.2.2.2 Modeling Phase

As in the previous modeling phase, the function *Specification/TestRelatedConstraintsValuesAssignment* (line 2 in Algorithm 3.1) is used to assign values to the extracted tester components characteristics. These are then used to generate the new AMS design netlist.

The implementation of the netlist file generation and its corresponding pre-verification step is shown in Figure 3.6 and Algorithm 3.7, respectively.

3.2.2.3 Verification Phase

The verification phase is the same as in the constraint-based property checking algorithm (lines 4-11 in Algorithm 3.1). Subsequently, given the AMS design netlist in the presence of specification and test-related constraints as well as the tester components characteristics, we perform a simulation based technique to assess the impact to the tester inaccuracies on the reliability of the test results.

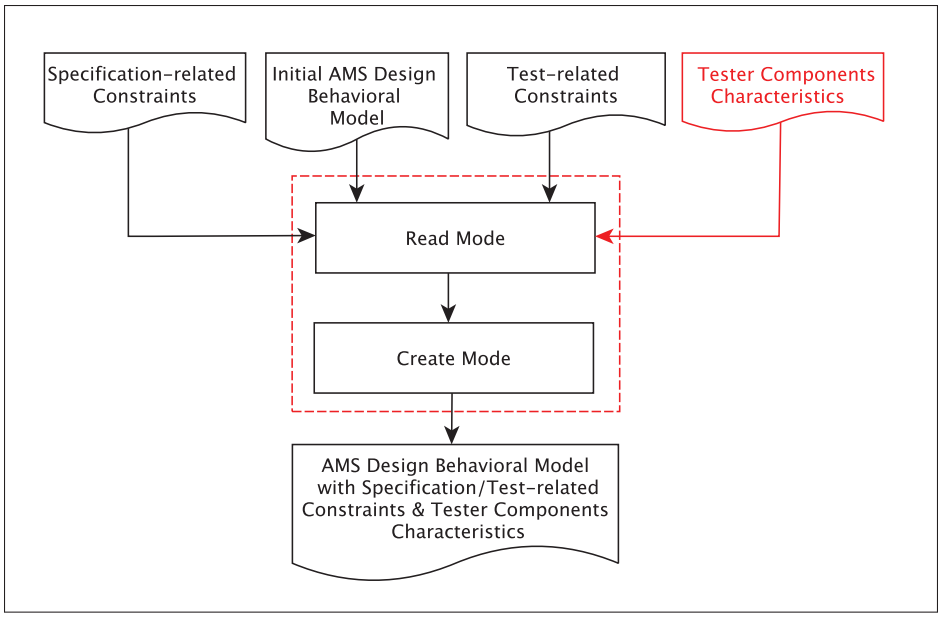

Figure 3.6 Netlist File Generation: Extended version

Algorithm 3.7 Netlist File Generation for the extended version: Pre-verification step

Require: $TEST_{flag}$, *SELECT*
Ensure: $TEST_{flag} \geq 1$
1 **while** $TEST_{flag} \geq 1$ **do**
2 **if** *SELECT = 0* **then**
3 C_{in} *(: , 1)= read (SpecificationRelatedConstraints);*
4 **else if** *SELECT = 1* **then**
5 C_{in} *(: , 2)= read (TestRelatedConstraints) ;*
  else
6 C_{in} *(: , 3)= read (TesterComponentsCharacteristics) ;*
  end
7 ***return*** C_{in} *;*
end
8 *UNDEFINED TEST SCHEME FOR THE GIVEN SPECIFICATION! ;*

Advantages of the extended approach are robustness and flexibility to account for external mixed-signal tester components characteristics. Likewise, the approach estimates the acceptance/rejection of AMS circuits with reference to these characteristics.

3.3 Summary

In this chapter, we presented a constraint-driven verification methodology for monitoring specifications of AMS designs. The proposed methodology includes three phases: constraints extraction, modeling and verification. It requires the definition of at least one test scheme for each specification. In case there is no test scheme defined for a given specification, a failure message is generated to notify that one must be elaborated. This early definition of specification test schemes allows to consider specification and test-related constraints such that the tests can be adequately performed. Likewise, it helps in the early discovery of missing, incomplete or misunderstood specifications, by forcing a detailed analysis of all specifications. We use behavioral models to capture the AMS design functionality at high level of abstraction, however the methodology can be easily applied at lower levels of abstraction. First, we derive and model specification and test-related constraints for each specification. These constraints are then added to the initial AMS design high-level behavioral model that is then simulated. Next, an assertion based runtime verification is carried out in a MATLAB simulation environment.

We also extend our constraint-based verification methodology to handle additional set of performance features derived from the mixed-signal tester components used for specifications tests. The combination of specification and test-related constraints with tester's instrument characteristics enriches us to assess the impact of the mixed-signal tester inaccuracies on the reliability of the specification test. The methodology estimates the acceptance/rejection of the AMS design with respect to the tester's characteristics.

In the next chapter, we apply our methodology to several illustrative examples, namely 2D and 3D AMS designs. Each phase of the proposed methodology is explained in more detail. Several interesting functional specifications are analyzed and verified. A comparison of our verification results with those obtained using previous approaches is illustrated.

CHAPTER 4

CASE STUDIES

This chapter illustrates our verification methodology through three concrete examples namely, a commercial frequency synthesizer IC based on a charge pump phase locked loop (CP-PLL), a 3D clock tree IC and a sigma-delta modulator. First, it outlines the relevant specification and test-related constraints of each case study followed by its behavioral model. Second, it describes several interesting functional specifications and illustrates the obtained verification results.

4.1 Commercial Frequency Synthesizer IC based on a Charge Pump Phase-Locked Loop

The functional specifications of the frequency synthesizer IC are listed in Table 4.1. In this example we wish to verify the lock time, output frequency and reference spurs specifications. As mentioned in Chapter 3, the first step in our methodology is to manually extract the design specifications, specification-related constraints as well as the specifications test schemes from the design preliminary datasheet. Indeed, the extracted test scheme used for the target specifications is presented in Figure 4.1, where the frequency synthesizer IC is the Device Under Test (DUT).

Table 4.1 Specifications of the CP-PLL frequency synthesizer IC (Freescale Semiconductor, 1999).

Parameter	**Symbol**	**Value**
Lock time	T_{lock}	$\leq$ 0.2 ms
Output Frequency	F_{out}	[100, 550]MHz
Supply Voltage	V_{pos}	[1.8, 3.6]V
PFD Output Current	$[PD_{outL}, PD_{outH}]$	[1.1, 4.4]mA
Charge Pump Voltage	$[VCP_{Lo}, VCP_{Hi}]$	[0, 5]V
VCO Gain	K1	4e6 Hz/volt
Reference Spurs	Ref_{spur}	$<$ -60 dB

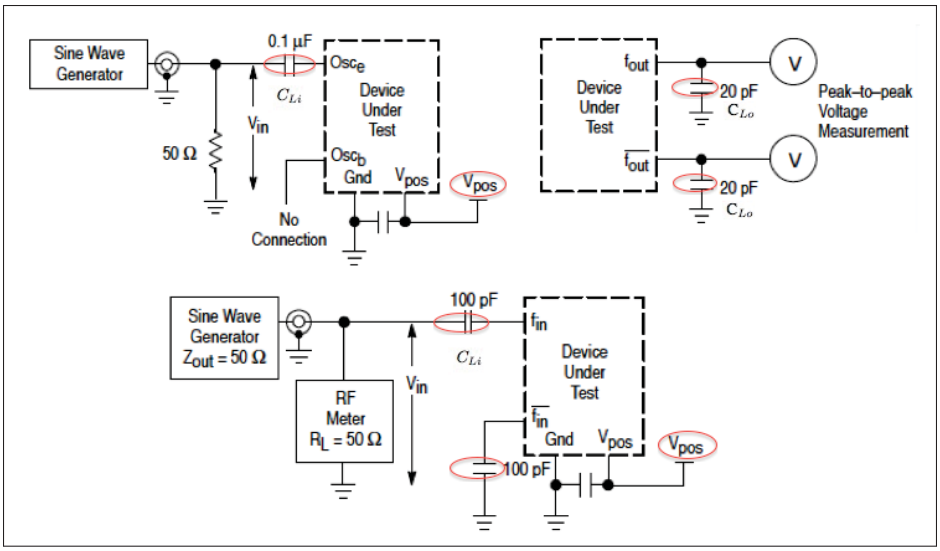

Figure 4.1 Specification Test Circuit
(Freescale Semiconductor, 1999).

The measurements at the test point consist simply of the voltmeter, oscilloscope and spectrum analyzer readings of the specifications of interest. A detailed description of the procedures used in testing each of the extracted specifications and the IC application circuit (see Figure 4.2) are illustrated in (Freescale Semiconductor, 1999).

4.1.1 Identification of Relevant Specification and Test-related Constraints

Based on the extracted test scheme and the application circuit which are used for the output frequency, reference spurs and lock time specifications, we define test-related constraints as the requirements related to the physical connection interface between the DUT and the test instruments (including the DUT input/output connecting components). In order to fully analyze the impact of these constraints on the frequency synthesizer behavior, we add a variation constraint on the capacitors (C_{Li} and C_{Lo}) used to connect the DUT to the test instruments (including waveform generators, voltmeters, oscilloscopes, spectrum analyzer, etc) as illustrated in Figure 4.1. The capacitance variations are within 10% of their initial value (given in the design datasheet). In addition, we consider specification-related constraints as the requirements on the DUT power supply (V_{pos}) for proper operation (see Figure 4.2). To get more realistic

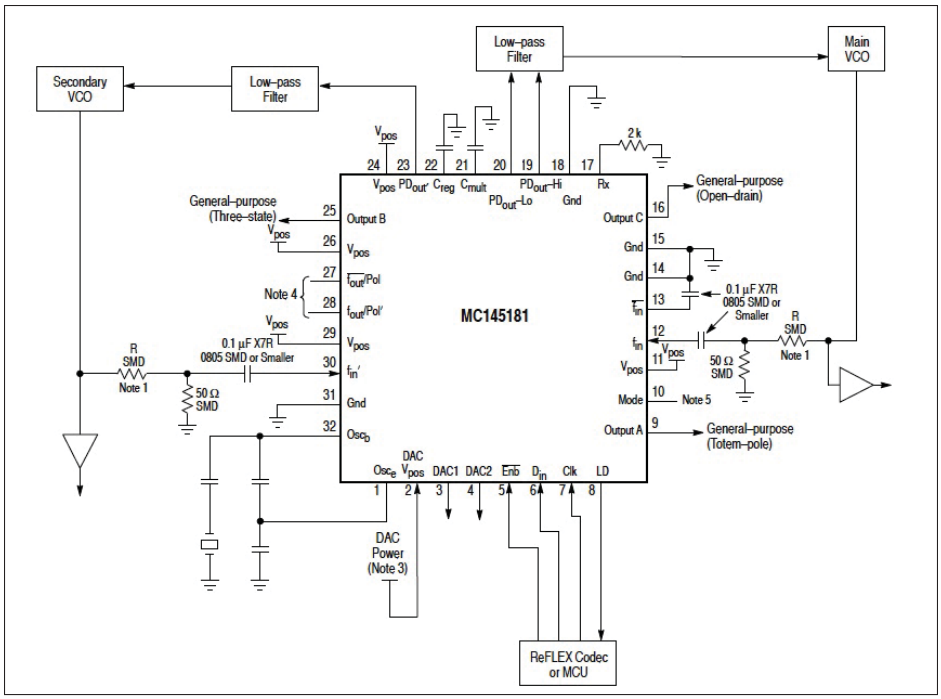

Figure 4.2 Application Circuit (Freescale Semiconductor, 1999).

results with respect to power fluctuations due to noise, we add to the initial frequency synthesizer behavioral model a circuit-level model of a power distribution network (PDN) (Smith *et al.*, 2000), illustrated in Figure 4.3. The PDN has noise contributions from the chip, package, printed circuit board (PCB) and the voltage regulator module (VRM) (Smith *et al.*, 2001).

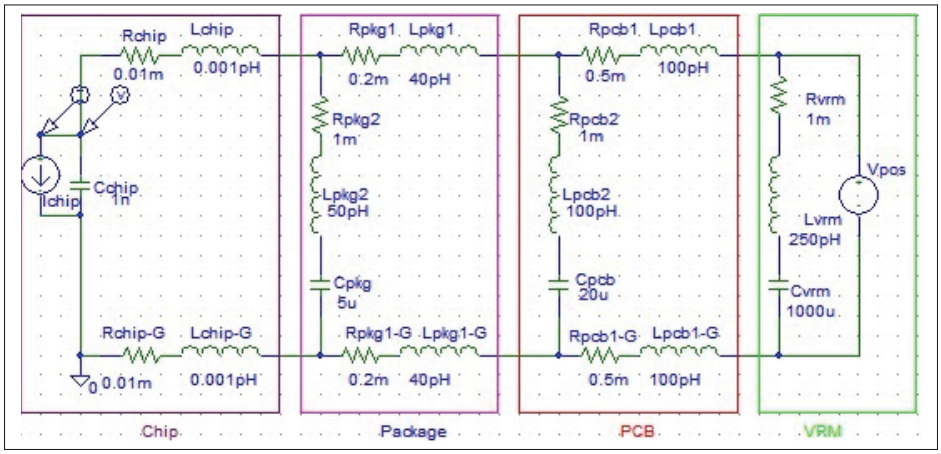

Figure 4.3 Power Distribution Network

The impedance profile of the derived PDN (seen by the chip pads) is presented in Figure 4.4. Similarly, we assume that we have a noisy input voltage module.

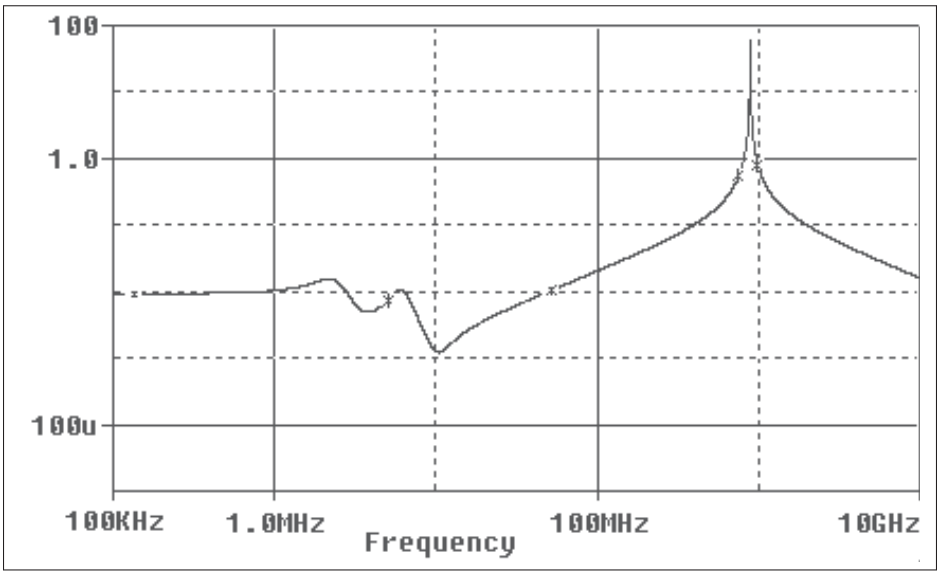

Figure 4.4 Power Distribution Network impedance profile

The peak-to-peak noise value is approximately 10% of the nominal power supply value presented in the datasheet, which in this case is 350mV peak-to-peak (note that the circuit is supplied with 3.6 V power in this application). The amplitude distribution of the voltage noise is Gaussian. The values of these specification and test-related constraints are chosen in such a way that the CP-PLL frequency synthesizer IC is guaranteed to operate within the specifications listed in the datasheet.

To sum up, we formalize the system of specification and test-related constraints as follows:

$$spec/test_{cst} = \begin{cases} C_{Li} \in [\mathrm{C}_{Limin}, \quad C_{Limax}] \\ C_{Lo} \in [\mathrm{C}_{Lomin}, \quad C_{Lomax}] \\ V_{supply-noise} \leq \varepsilon \end{cases}$$

where C_{Li} and C_{Lo} represent the DUT input/output connection elements and ε represents the maximum tolerable noise level.

In what follows, we assess the impact of the aforementioned constraints on the CP-PLL based frequency synthesizer performance.

4.1.2 CP PLL based frequency synthesizer Behavioral Model

Before introducing the CP-PLL based frequency synthesizer model we recall some general concepts related to PLL modeling.

- PLL Modeling

PLLs are a class of AMS circuits that are commonly used in many applications including frequency synthesis, phase/frequency modulation and clock data recovery. Various behavioral models have been built (Li & Meiners, 2000; Andreu *et al.*, 2001; Acco *et al.*, 1999) in order to facilitate PLLs analysis and to detect out-of-specification failures at early design stages. These abstracted models provide a good accuracy and represent a notable attempt to evaluate the PLL performance before committing to silicon. Generally, simulation-based verification for high-level design models is conducted to evaluate the PLL's functional specifications under particular initial conditions and parameters values. Likewise, the circuit functionality is verified under the assumption that these specifications are correct, complete and accurate. Nevertheless, typical specifications (provided in the design datasheet) might appear as not clear and explicit enough to allow a predictable verification process that guarantees correct functionality of the end product.

In this case study, we have reproduced the behavioral model of the commercial CP-PLL based frequency synthesizer IC (MC145181) provided in (Freescale Semiconductor, 1999), a datasheet from Freescale. The IC block diagram is presented in Figure 4.5.

To illustrate the impact of specification and test-related constraints, two sets of simulations are performed here, with and without these constraints.

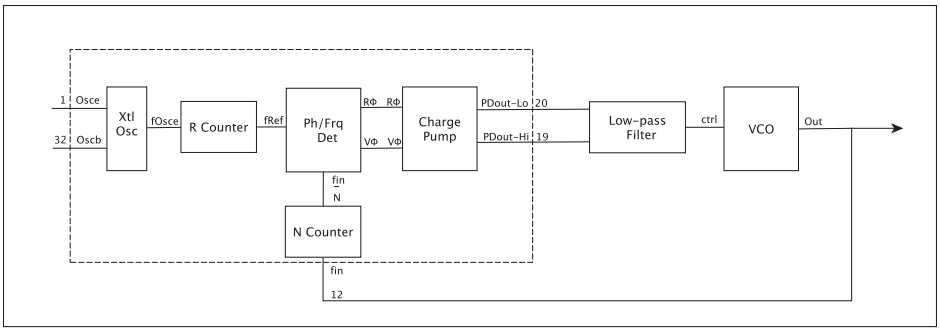

Figure 4.5 CP-PLL based frequency synthesizer Block Diagram (Freescale Semiconductor, 1999).

- **PSpice Implementation of the CP-PLL Frequency Synthesizer for the Case without Specification and Test-related Constraints:**

To simulate the operation of the CP-PLL frequency synthesizer in Figure 4.5, we need a model for the phase-frequency detector (Ph/Frq Det) & the charge pump, a model for the low-pass filter and a model for the VCO. The reference frequency f_{Ref} is the frequency of the reference oscillator divided by the value programmed into the R counter (f_{Ref}= f_{Osce}/R). In this PSpice implementation, f_{Ref} is used directly (to avoid overloading the circuit simulation with a higher frequency followed by a counter). In the same manner, the input amplifier and N counter are folded into the model for the VCO.

The behavioral model of the CP-PLL based frequency synthesizer is presented in Figure 4.6. It is developed using Analog Behavioral Modeling (ABM) feature of PSpice (refer to section 1.3 in chapter 1 for more details about ABM feature of PSpice). It consists of a phase-frequency detector (Ph/Frq Det) with a reference oscillator (HB1 block), a loop filter, and a voltage-controlled oscillator (VCO) represented by HB2 block in Figure 4.6. Each of these components is modeled using analog behavioral block. For ease of use, the models are developed as parameterized subcircuits.

The output signals from the reference oscillator (f_{Ref}), included in HB1 block and presented in Figure 4.8, and the VCO (Out) are compared to determine the lead/lag relationship and a compensating up (PD_{out}-Hi) or down (PD_{out}-Lo) current signal is generated. The loop filter is

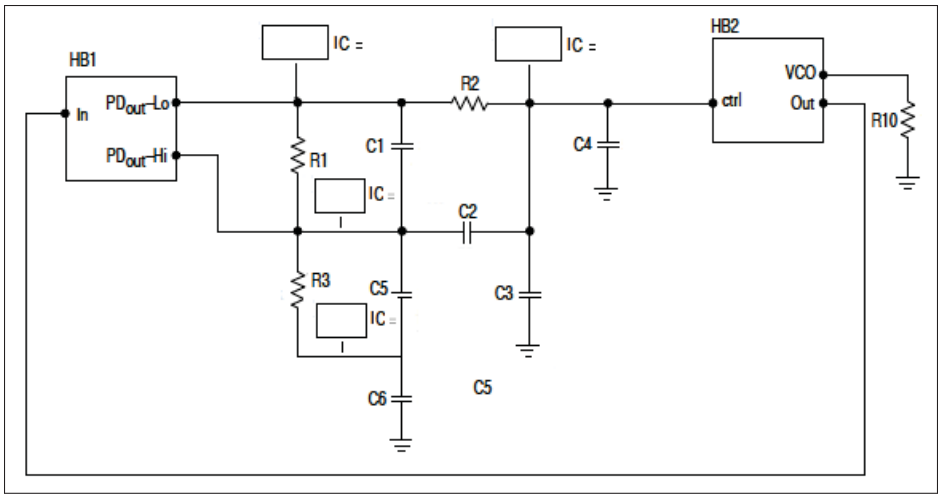

Figure 4.6 CP-PLL based frequency synthesizer ABM Model (Freescale Semiconductor, 1999)

then used to convert the current from the PFD to a tuning voltage for the VCO. Following are the ABM models for the blocks in Figure 4.6.

- HB1 block

It includes a reference Oscillator (HB block) and a PFD with dual charge pumps as shown in Figure 4.7. The PFD subcircuit has two inputs, one from the reference oscillator (f_{Ref}) and the other from the VCO (In). The two signals are compared to determine the lead/lag relationship and a compensating up (PD_{out}-Hi) or down (PD_{out}-Lo) signal is generated.

The ABM model for simulating the reference oscillator (HB Block) is shown in Figure 4.8.

$$Ref = \begin{cases} sin(t_w * f_r * time) & \text{if } V(shift) < 1 \\ sin(4 * t_w * f_r * time) & \text{else} \end{cases}$$

Its output is defined by an IF-THEN statement to produce a frequency that is either the reference frequency (fr), if the input signal "shift" is low, or four times fr if it is high. A limiter/gain block is used to convert the sine wave output into a square wave.

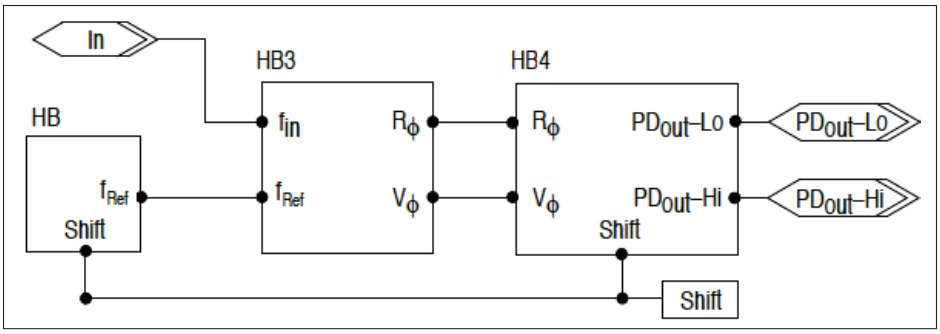

Figure 4.7 HB1 block: PFD with Dual Charge Pumps & Reference Oscillator ABM Models (Freescale Semiconductor, 1999)

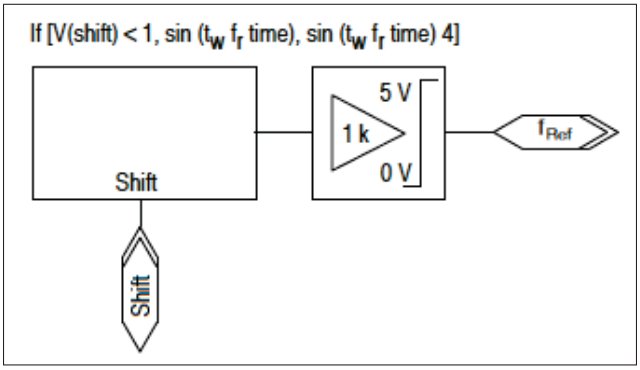

Figure 4.8 HB block: Reference Oscillator ABM Model (Freescale Semiconductor, 1999)

More details about the implementation of internal sub-blocks (HB3 and HB4) is illustrated in (Freescale Semiconductor, 1999) (more specifically on Figures 41-47 in this datasheet).

- HB2 Block

This block represents the VCO and is implemented using a sine wave generator with a control element as presented in Figure 4.9. An analog behavioral block is employed as a sine wave generator and a GVALUE element is used as a control element. The equation for the sine wave

generator is given by:

$$Out = \begin{cases} sin((\frac{t_w}{N+S_z})f_c * time + v(int)) & \text{if } V(turbo) < 1 \\ sin((\frac{4*t_w}{N+S_z})f_c * time + v(int)) & \text{else} \end{cases}$$

where f_c is the VCO frequency when the control voltage is zero.

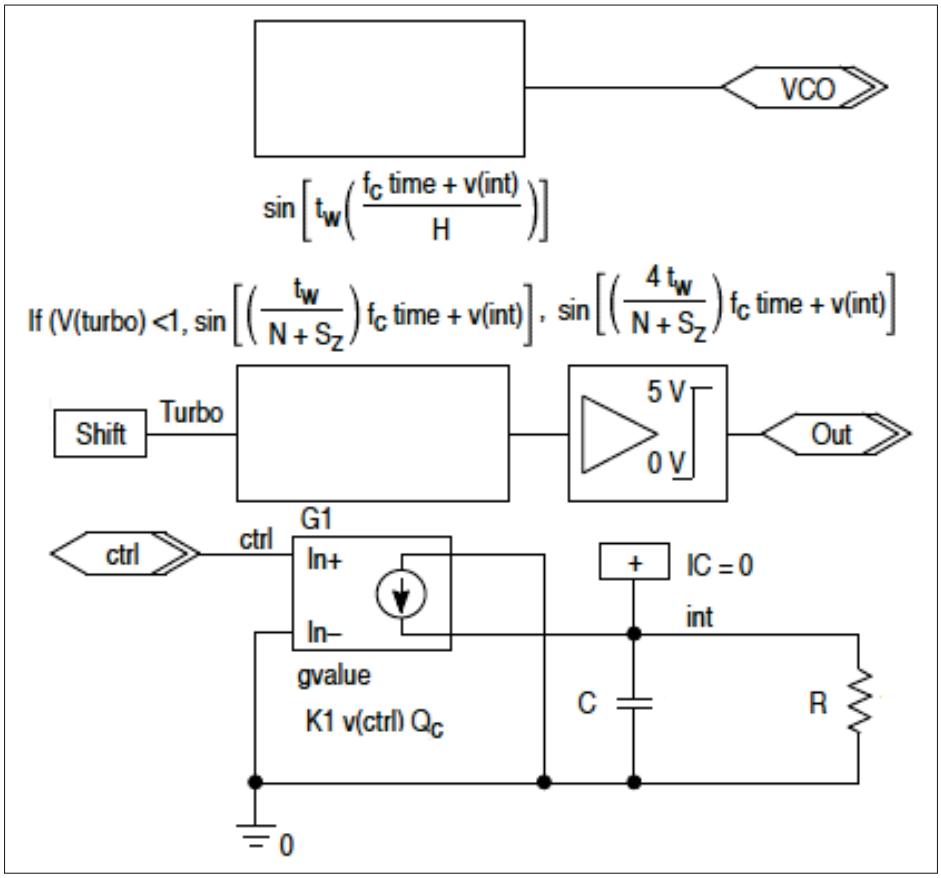

Figure 4.9 HB2 block: VCO ABM Model
(Freescale Semiconductor, 1999)

The GVALUE element acts as an integrator and its output is given by:

$$v(int) = K1 * v(ctrl) * Q_c \tag{4.1}$$

where K1 is the VCO gain in Hz/V.

- **PSpice Implementation of the CP-PLL Frequency Synthesizer for the Case with Specification and Test-related Constraints:**

In order to assess the impact of specification and test-related constraints some changes must be made to the previous PSpice implementation. The R and N counters are considered in the loop model as presented in Figure 4.5. By including the two counters, the reference frequency f_{Ref} is not directly used. It is rather the frequency of the oscillator divided by the value programmed into the R counter. Likewise the input amplifier and N counter are unfolded from the VCO model.

In what follows, we verify the CP-PLL based frequency synthesizer IC specifications with and without specification and test-related constraints (spec/test$_{cst}$).

4.1.3 Verification Results

In this section we report the results obtained with the CP-PLL based frequency synthesizer IC parameters listed in Table 4.2 where f_c and K1 represent, respectively, the VCO frequency when the control voltage is zero and the VCO gain in rad/V. With respect to our methodology, these verification results are obtained by simulating the CP-PLL based frequency synthesizer netlist, with the power distribution network (PDN) model in Figure 4.3, using PSpice.

In this case study, we have simulated the MC145181 main loop that covers 100 to 550 MHz frequency band (see Figure 4.2). An external reference $f_{Osce} \in [9, 80]$ MHz is used. This reference signal is AC coupled into Osc_e pin (the coupling capacitor is in series with the load R_L =50 Ω) and the Osc_b pin is left floating as illustrated in Figure 4.1. Since the phase detector is chosen to run at 25 KHz (f_{Ref}= 25 kHz), the reference f_{Osce} must be divided down to 25 KHz by the R counter (f_{Ref}= f_{Osce}/R). The main VCO is AC coupled into f_{in} pin as presented in Figure 4.2. The Output frequency is measured using the test method illustrated in Figure 4.1.

Table 4.2 Parameters for the CP-PLL based frequency synthesizer Model (Freescale Semiconductor, 1999).

Parameter	**Value**	**Unit**
f_{Ref}	25	KHz
f_c	544.6	MHz
K1	4	MHz/V
R1	60.4	KΩ
R2	20	KΩ
R3	40.2	KΩ
R4	1	KΩ
C1	330	pF
C2	330	pF
C3	3300	pF
C4	330	pF
C5	33	pF
C6	50	pF

In order to assess the impact of the previously defined specification and test-related constraints $spec/test_{cst}$, we have redefined the frequency synthesizer's system specifications by taking into account the specification and test-related constraints as follows:

$$\text{System}_{Specifications}=\begin{cases} T_{lock} \leq 0.2\text{ms} \\ F_{out} \in [100, 550]\text{MHz} \\ \text{Ref}_{spur} < -60dB \\ C_{Li} \geq 350\text{ pF} \quad at\ Osc_e\ pin \\ C_{Li} \geq 310\text{ pF} \quad at\ f_{in}\ pin \\ C_{Lo} \geq 310\text{ pF} \\ V_{supply-noise} \leq 350\text{ mV P-P} \end{cases}$$

Indeed, a more rigorous definition of the input connecting capacitor C_{Li} at Osc_e input pin has to be $\geq$ 350 pF. This is because the minimum frequency at this pin can be as low as 9MHz ($f_{Osce} \in$ [9, 80] MHz as specified in the IC datasheet) and since the AC coupling capacitance with the input load impedance forms a high pass filter with a cut-off frequency: f_c= 1/(2 π R_L C_{Li}).

The same applies to input pin f_{in} (see Figure 4.2) the input connecting capacitor at this pin must be $\geq$ 310 pF since the minimum tunable frequency of the VCO used is 10 MHz. Likewise C_{Lo} must be $\geq$ 310 pF to adequately perform the output frequency signal measurements.

The first specification that we have verified is the lock time specification. It is the maximum time required for the CP-PLL synthesizer to switch from one frequency to another, which must be $\leq$ 0.2 ms. In the normal application of our methodology, the AMS design behavioral model simulation takes into account all the specification and test-related constraints. However, as mentioned previously, to illustrate the impact of these constraints, two sets of simulations are performed here, with and without these constraints.

The verification results without specification and test-related constraints, corresponding to the verification method used in (Freescale Semiconductor, 1999), are depicted in Figure 4.11.

These results indicate that no specification violation has occurred and $T_{lock} \leq$ 0.2 ms. The rest of the verification results without specification and test-related constraints are summarized in Table 4.3 (middle column). Likewise, the verification results of T_{lock} as well as the rest of the CP-PLL synthesizer specifications for different values of C_{Li}, C_{Lo} and $V_{supply-noise}$, when specification and test-related constraints are applied, are reported in Table 4.3 (right column).

Table 4.3 Simulation results with and without specification and test-related constraints

Parameter performance	**Results without specification/test-related constraints**	**Results using Constraint-Based-PC Algorithm**
Lock time: T_{lock}	0.2 ms	0.7 ms/does not lock
Operating Frequency: F_{out}	[100, 550] MHz	[95, 372.6] MHz
Reference Spurs: Ref_{spur}	-63 dB	-33.69 dB
Verification Status	**Verified**	**Violated**

These verification results indicate that the CP-PLL frequency synthesizer IC fails in the presence of specification and test-related constraints, if these constraints are not satisfied, and that

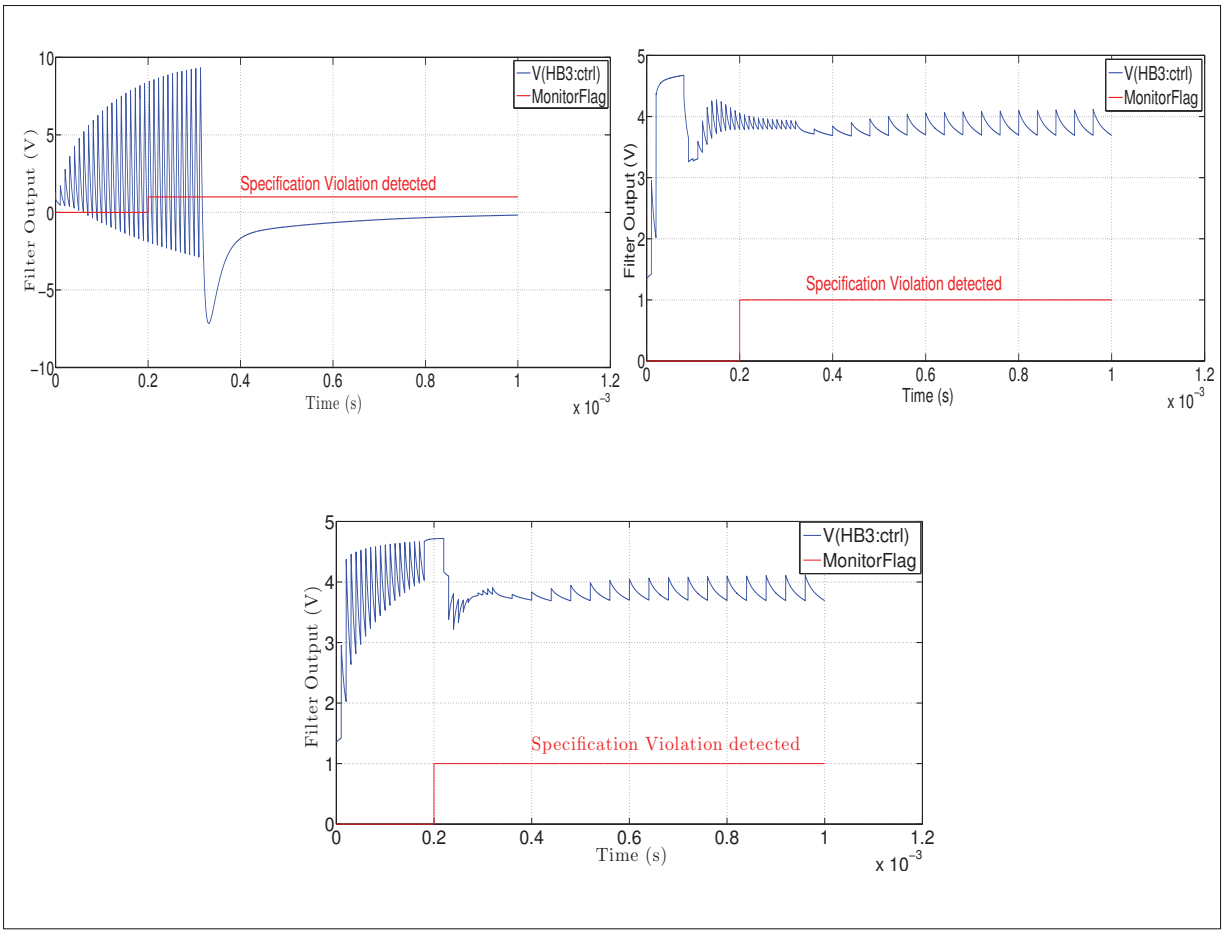

Figure 4.10 Simulation results of the lock time specification: Results with specification and test-related constraints

two main scenarios were observed; either the IC does not get locked to the input frequency and a specification violation is reported, as illustrated in Figure 4.10, or it gets locked to the input frequency with a maximum settling time of 0.7 ms (see Table 4.3, right column) and an output frequency ranging from 95 MHz to 372.6 MHz, which in both cases violates the desired system specifications in Table 4.1.

In summary, the obtained results clearly show the relevancy of the proposed approach as it finds realistic operation conditions with which specifications are not met, while the IC was previously verified with existing techniques. As the different component values were directly taken from (Freescale Semiconductor, 1999), our results show that some additional specifica-

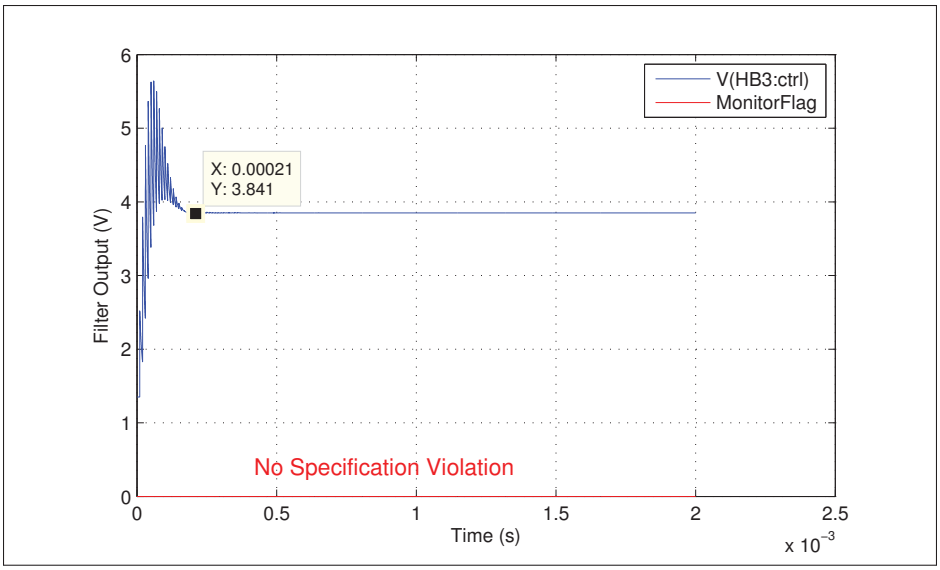

Figure 4.11 Simulation results of the lock time specification: Results without specification and test-related constraints

tion and test-related constraints on those component values would be required to make sure that the IC will perform as predicted. Finding this early in the design process allows designers to make the necessary changes. Note that in our methodology, the specification violations are automatically detected by the constraint-based property-checking algorithm, namely the verification phase of Algorithm 3.1 in chapter 3.

4.2 3D Clock Tree

In this section, we present another case study, namely a 3D clock tree. Through this example, we highlight different type of test-related constraints, in particular those that affect the design. We show the importance of early consideration of such constraints on the design performance. In this case study, we use the circuit models in (Zhao *et al.*, 2011) which are not behavioral models but can still be used as our methodology supports different levels of abstraction. Indeed, the proposed verification methodology is flexible and in some cases, it may take lower level description of the circuit instead of its behavioral description. The 3D clock tree is composed of two dies and simulated using Spice circuit simulator. We replace the clock source for pre-bond

and post-bond operations with the CP-PLL frequency synthesizer (Freescale Semiconductor, 1999) studied in the previous section.

The circuit models used for post-bond and pre-bond test (Zhao *et al.*, 2011) are presented in Figure 4.12, where node A is the clock source for post-bond operation. Sink C on die-0 and sink E on die-1 have loading capacitances of C_{LC} and C_{LE}, respectively. A TSV with a TSV-buffer is used to connect node B and D. The sub-tree on die-1 (edge (D,E)) is connected to F (the pre-bond testing clock source) using a transmission gate (TG). Capacitors C_{LC} and C_{LE} have the same value of 5fF. The wires (A,B), (B,C), (D,E) and (F,D) have the same length of 500μm. Pre-bond test of the 3D clock tree requires that each die in the stack be separated and tested individually using a single test probe.

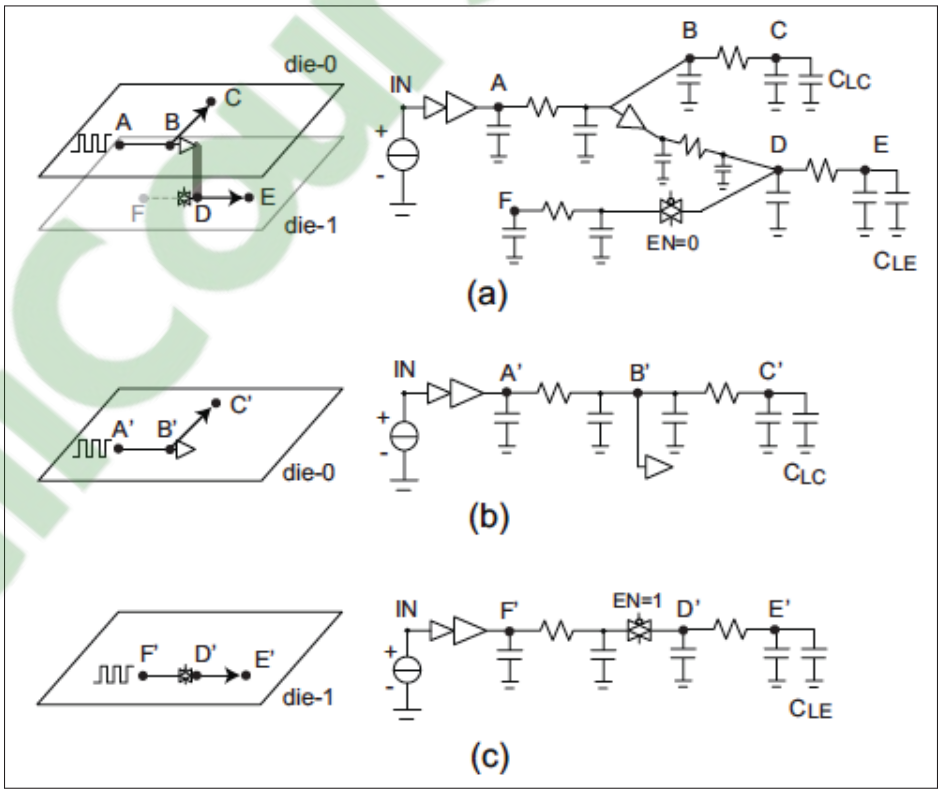

Figure 4.12 Specification Test Circuits for: (a) post-bond 3D clock tree, (b) pre-bond testable 2D clock tree on die-0, (c) pre-bond testable 2D clock tree on die-1 (Zhao *et al.*, 2011)

4.2.1 Identification of Relevant Specification and Test-related Constraints

The pre-bond test of each die in the 3D clock tree (die-0 and die-1 in Figure 4.12) requires a fully connected clock tree so that the minimum-skew clock signal can be delivered to the sinks on die-0 and die-1 using a single test probe. However, it is shown in (Minz *et al.*, 2008) and (Zhao & Lim, 2010) that only one die in 3D clock trees with multiple TSVs, has a fully connected 2-D clock tree while the other dies in the stack have several small, isolated subtrees. This in turn makes pre-bond test of the 3D clock tree next to impossible (Zhao *et al.*, 2011) as each subtree necessitates its own probe pad. Additionally, the overall timing accuracy (OTA) of the testing equipment is more than $\pm$100 ps (Verigy V93000 SoC Series Pin Scale Digital Cards) which makes dealing with many probe pads to test a clock signal very hard especially when testing with multiple probes is very costly.

Another constraint appears during the pre-bond test of the 2D clock tree in die-0 (see Figure 4.12) as the skew specification of this die cannot be met without the presence of the downstream capacitances of the subtree in die-1. This consequently may either hinder or impede the test process.

In accordance with the previously defined specification and test-related constraints, there are three major design modifications in order to enable efficient pre-bond and post-bond testing of the 3D clock tree IC:

$$\text{spec/test}_{cst} = \begin{cases} \textit{1- Redundant tree construction for die-1} \\ \textit{(Red_{die-1}; die-0 already contains a fully connected clock tree).} \\ \textit{2-TSV-buffer insertion} \\ \textit{(to avoid the high skew situation during the pre-bond test of die-0).} \\ \textit{3-TG insertion} \\ \textit{(to connect or disconnect the redundant tree).} \end{cases}$$

4.2.2 3D Clock Tree Model

As mentioned before, we use the 3D clock tree with two dies from (Zhao *et al.*, 2011), and we replace the clock source for pre-bond and post-bond operations with the CP-PLL frequency synthesizer (Freescale Semiconductor, 1999) studied in the previous section. We use technology parameters from the 45 nm predictive technology model (Verigy V93000 SoC Series Pin Scale Digital Cards) with a unit-length wire resistance of 0.1/μm and a unit-length wire capacitance of 0.2 fF/μm. The values of the sink capacitance are within the range [5 fF, 80 fF]. The buffer parameters are set to: R_d= 122 Ω, C_L= 24 fF, and td = 17 ps. We use 10 μm$\times$10μm via-last TSVs with 20μm height and 0.1μm liner oxide thickness. TSV parasitics (Synopsys Raphael) are set to R_{TSV}= 0.035 Ω and C_{TSV}= 15.48 fF. The supply voltage (Vdd) is set to 1.2-V^2. The maximum load capacitance for each buffer *cmax* is 300 fF.

In Spice simulation, wire segments and TSVs are represented as π models and clock buffers and TSV-buffers are represented as inverter pairs. The specification of the target 3D clock tree is defined as follows (Zhao *et al.*, 2011):

$$\text{Skew}_{spec}=\begin{cases} \text{Skew}_{Post-bond} \leq 42.21\text{ps} \\ \text{Skew}_{Pre-bond\,(die-0)} \leq 42.21\text{ps} \\ \text{Skew}_{Pre-bond\,(die-1)} \leq 54.13\text{ps} \end{cases}$$

where $Skew_{Post-bond}$ is the clock skew specification for post-bond test (Figure 4.12 (a)), $Skew_{Pre-bond\,(die-0)}$ and $Skew_{Pre-bond\,(die-1)}$ are the clock skew specifications for the pre-bond testable 2D clock tree on die-0 (Figure 4.12 (b)) and die-1 (Figure 4.12 (c)) respectively.

4.2.3 Verification Results

In this section we report the results obtained with the 3D clock tree in Figure 4.12 (a), where die-0 contains the clock source and has a fully connected tree while die-1 has a sub-tree (edge (D,E)).

In order to assess the impact of the previously defined specification and test-related constraints $spec/test_{cst}$, we have redefined the skew specification as follows:

$$\text{Skew}_{spec/test-cst} = \begin{cases} \text{Skew}_{Post-bond} \leq 42.21 \text{ ps} \\ \text{Skew}_{Pre-bond(die-0)} \leq 42.21 \text{ ps} \\ \text{Skew}_{Pre-bond(die-1)} \leq 54.13 \text{ ps} \\ 1-\text{Red}_{die-1} = \text{edge (F,D)} \\ 2-\text{TSV}_{buffer} = (R_d = 122\ \Omega,\ C_L = 24 \text{ fF, td} = 17 \text{ ps}) \\ 3-\text{TG}_{param} = \begin{cases} \text{C}_{nodeD,Gr} = 14.2 \text{ fF} \quad \textit{when TG is off} \\ (\text{C}_{In,Gr} = 16.4 \text{ fF},\ C_{Out,Gr} = 18.4 \text{ fF} , \\ \text{R}_{In,Out} = 108\ \Omega\ ) \quad \textit{when TG is on} \end{cases} \end{cases}$$

where $\text{C}_{nodeD,Gr}$ is the capacitance between node D and the ground when the transmission gate (TG) is off, $C_{In,Gr}$, $C_{Out,Gr}$ and $R_{In,Out}$ are the capacitance between the TG input and the ground, the capacitance between the TG output and the ground and the resistor between the TG input and output respectively, when the transmission gate (TG) is on.

These additional specification and test-related constraints are passed as input arguments to *GenerateCircuitNetlist()* function (see line 3 in Algorithm 3.1 in chapter 3), during the modeling phase, in order to generate the 3D clock tree netlist that is then fed into Spice circuit simulator.

We have simulated the 3D clock tree using our constraint-based property checking algorithm and compared the obtained results with the clock skew specification $Skew_{spec/test-cst}$. The simulation results for the post-bond clock skew give 42 ps for the global 3D clock tree, a clock skew value of 42.3 ps and 54.19 ps for the 2D clock trees on die-0 and die-1 respectively. This is in the case when the previous specification and test-related constraints are respected otherwise the algorithm reports a specification violation.

In summary, these verification results show that, after die separation in order to verify the two dies (die-0 and die-1) individually, the clock skew specification on die-1 (which does not con-

tain a fully connected clock tree) cannot be verified without adding the redundant tree Red_{die-1} (edge (F,D)) and a specification violation is reported. Likewise, there is a skew degradation during the pre-bond verification of die-0 as the downstream capacitances of the sub-tree (edge (D,E)) in die-1 are not present. Though, once a TSV-buffer is inserted on die-0, the clock skew specification is reported to be true and there is no change to the delay at the sink C on die-0 after die separation since the buffer shields off all the downstream capacitance. Additionally, the clock skew verification was not possible after die bonding without the use of a TG to disconnect the redundant sub-tree on die-1 and the specification is reported to be false.

While pre-bond testing process is particularly difficult, if not impossible (Zhao *et al.*, 2011; Lee & Chakrabarty, 2009; Gambino *et al.*, 2015), these results clearly show the relevancy of the proposed approach as it finds out-of-specification situations due to a lack of early consideration of specification and test-related constraints on the clock skew specification. These additional constraints are essential elements for an efficient pre-bond and post-bond verification of the 3D clock tree IC and consequently for an efficient pre-bond and post-bond test.

4.3 Sigma-Delta Modulator

In addition to specification and test-related constraints, the proposed methodology takes into account the tester components characteristics during the modeling and verification phases as illustrated in Figure 3.4, chapter 3. In this section, the extended approach is illustrated on a sigma-delta modulator. The effects of specification and test-related constraints with tester's instrument characteristics on the AMS design performance are analyzed at a high level of abstraction.

The sigma-delta modulator (Figure 4.13) specifications are listed in Table 4.4. In this case study, we are interested in checking the signal-to-noise and distortion ratio (SNDR) and resolution (Rbit) specifications. The first step in specifications analysis is to identify and manually extract the set of specification and test-related constraints as well as the tester components characteristics as described in chapter 3.

Table 4.4 Specifications of the sigma-delta modulator (Brigati *et al.*, 1999).

Parameter	**Symbol**	**Value**
Signal-to-Noise and Distortion Ratio	SNDR	101.5 dB
Resolution	Rbit	16 bits
Signal Bandwidth	BW	22.05 kHz
Sampling Frequency	Fs	11.2896 MHz
Oversampling Ratio	R	256
Samples Number	N	65536
Integrator gains	b, b2	0.5
Power Supply	Vref	1V

Specification and test-related constraints are defined based on the test method described in chapter 1 (Figure 1.2). Similarily, the tester components characteristics are derived from the industrial HP83000 ATE datasheet (Dantes HP83000, 1998) used for the specifications test. In this example we consider only one tester component, namely the HP83000 ATE power supply component characteristics.

4.3.1 Identification of Relevant Specification and Test-related Constraints and Tester Components Characteristics

In general, for ADC performance specifications, the most important constraints we are interested to consider are summarized in chapter 1 (Table 1.1, columns 2 and 3). We apply these specification and test-related constraints to the 16-bit sigma-delta modulator presented in (Brigati *et al.*, 1999). Likewise, we add the requirements related to the physical connection interface between the DUT and the test instruments. In addition, we consider the tester component characteristics as the periodic and random deviation (PARD) specification of the HP83000 ATE power supply block. PARD is defined as the sum of all ripple and noise voltages measured over a certain bandwidth (Crandall, 1997). Note that, the variation in the analog input signal amplitude for the HP83000 ATE, is estimated in (Comte *et al.*, 2003) based on the summation of a constant term and a term proportional to the nominal value of the signal amplitude. The

total variation is given by the following expression:

$$\Delta A = (\frac{0,01}{100}) * A + c \tag{4.2}$$

where A and represent, respectively, the input signal amplitude and the constant term. This constant term is estimated as being 4mV for low amplitude signals and 5mV for high amplitude signals (Comte *et al.*, 2003).

To sum up, we formalize the system of specification and test-related constraints with the tester component characteristics as follows:

$$spec/test_{cst-with-T.C_{ch.}} = \begin{cases} \textit{Possible measurement errors due to input signal amplitude variation.} \\ \textit{Number of samples} \geq N \\ \textit{Number of periods} \geq M \\ \textit{Stimulus p-p amplitude} < \text{FS} \\ C_{Li} \in [\text{C}_{Limin}, \quad C_{Limax}] \\ PARD \leq \rho \end{cases}$$

where N, M and FS represent, respectively, the number of samples considered to perform the FFT, the number of periods of the input sine-wave during acquisition and the ADC full scale range. C_{Li} and ρ represent, respectively, the DUT input load capacitance and the acceptable maximum level of the tester power supply PARD.

4.3.2 Sigma-Delta Modulator Behavioral Model

A sigma-delta modulator is one of the most effective forms of ADCs. Its applications involve communication systems, sensors and professional audio. In this case study, we have reproduced the behavioral model of the sigma-delta modulator presented in (Brigati *et al.*, 1999). The model is illustrated in Figure 4.13. It is developed in a Matlab/Simulink environment (The Mathworks Inc., 2011).

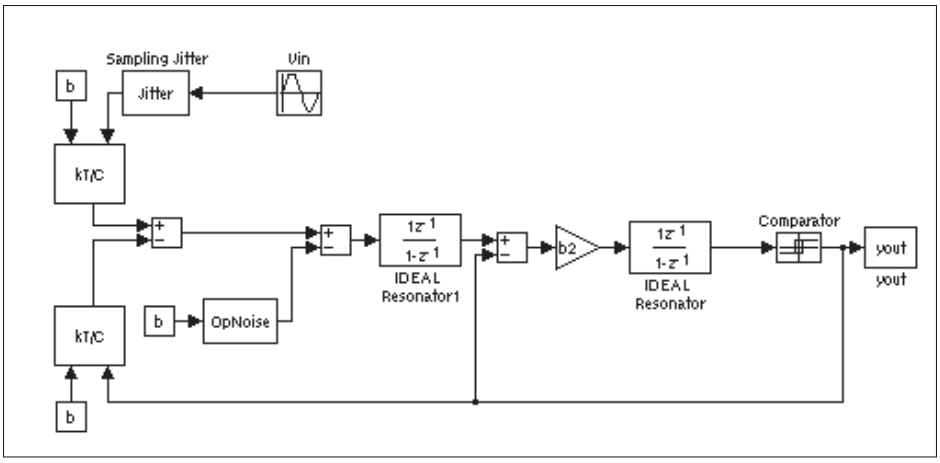

Figure 4.13 Sigma-Delta Modulator Behavioral Model
(Brigati *et al.*, 1999)

In order to enable a more realistic performance analysis, at the behavioral level, the essential non-idealities affecting the sigma-delta modulator performance have to be taken into account in the design model. These include sampling jitter, kT/C noise and operational amplifier parameters (noise, finite gain, finite bandwidth, slew-rate and saturation voltages). A detailed description of the design model, the considered effects and the implementation details is provided in (Brigati *et al.*, 1999).

In our example, only the non-idealities of the first integrator are considered as illustrated in Figure 4.13. More specifically, sampling jitter, kT/C noise and operational amplifier noise are considered separately (combined effects are not considered in this work).

In what follows, we verify the sigma-delta modulator specifications in the presence of specification and test-related constraints with the tester component characteristics as formalized in (spec/test$_{cst-with-T.C_{ch.}}$).

4.3.3 Verification Results

We simulated the sigma-delta modulator behavioral model for both ideal case (without any non-linearity) and when non-idealities (e.g., sampling jitter, kT/C noise and operational amplifier noise) are introduced. These non-idealities are considered separately.

Firstly, the simulations were conducted without specification and test-related constraints and tester components characteristics. The simulation results for the ideal sigma-delta modulator are shown in Figure 4.14. Table 4.5 (columns 2 and 3) summarizes the results obtained when non-idealities including sampling jitter, kT/C noise and operational amplifier noise are considered separately. The obtained results are the same as in (Brigati *et al.*, 1999) and no specification violation detected.

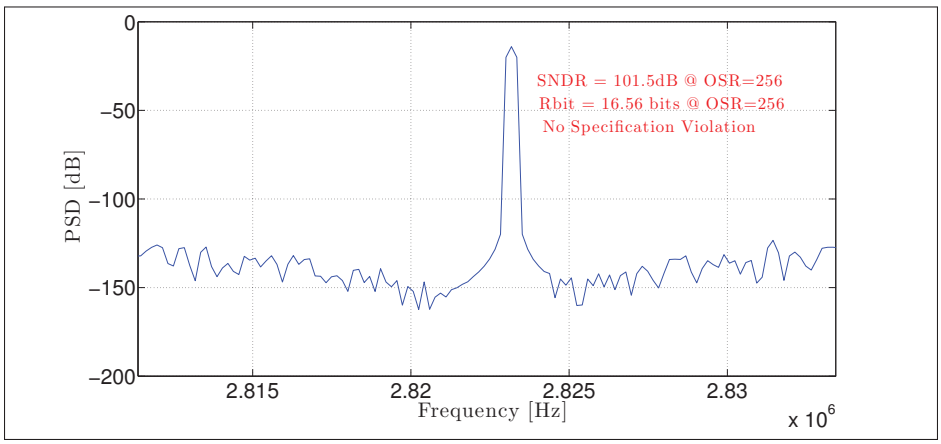

Figure 4.14 Simulation results of the SNDR and Resolution (Rbit) specifications: Results without specification and test-related constraints and tester components characteristics

Secondly, in order to assess the impact of the previously defined specification and test-related constraints as well as the tester component characteristics (spec/test$_{cst-with-T.C_{ch.}}$), we have redefined the SNDR and resolution (Rbit) specifications as follows:

$$spec/test_{cst-with-T.C_{ch.}} = \begin{cases} SNDR \quad 101.5 \text{ dB} \\ Resolution \quad 16 \text{ bits} \\ Number\ of\ samples \geq 65536 \\ Number\ of\ periods \geq 16388 \\ Stimulus\ p\text{-}p\ amplitude < 1V \\ C_{Li} \in [10, \quad 100] \text{ nF} \\ PARD \leq 4\text{mV} \end{cases}$$

The variation in the analog input signal amplitude for the HP83000 ATE, is calculated using Equation 4.2 which gives ΔA= 0.004V. The simulations were conducted for different values of *PARD* (tester power supply characteristic), with HP83000 power supply characteristic (*PARD* $\leq$ 4mV) and with the HP6655A characteristic (*PARD* $=$ 7mV).

Table 4.5 Simulation results with and without specification and test-related constraints and tester components characteristics

	Res. without spec/test-rel. csts & tester comp. charac.		**Res. using Constraint-Based-PC Algorithm**			
			HP 83000 P.S charac.		**HP 6655A charac.**	
	SNDR [dB]	Rbit [bits]	SNDR [dB]	Rbit [bits]	SNDR [dB]	Rbit [bits]
Ideal modulator	101.5	16.56	[101.1 101.5]	[16.50 16.56]	95.60	15.59
Sampling jitter ($\Delta\tau$ = 8 ns)	98.6	16.09	[98.40 98.80]	[16.06 16.13]	90.9	14.81
Switches (kT/C) noise (Cs = 5 pF)	98.7	16.11	[98.50 99.30]	[16.07 16.20]	91.8	14.95
Op-amp noise (Vn = 50 μVrms)	96.6	15.75	[95.60 97.20]	[15.59 15.86]	80.2	13.02
Verification Status	**Verified**		**Verified**		**Violated**	

The results are summarized in Table 4.5 (columns 4-7). From Table 4.5, it can be noted that, the SNDR and resolution (Rbit) specifications have failed due to high level of the *PARD* specification of the tester power supply and test conditions. In reality, one may come across such a level since a power supply with less better quality is less expensive and attractive solution especially in a customer test environment.

In this case, for the HP6655A power supply (with *PARD* = 7mV), the obtained results (Table 4.5 (columns 6-7)) correspond to the worst-case value of each specification, considering the ideal modulator and the modulator with non-idealities including sampling jitter, kT/C noise and operational amplifier noise. The effect of specification and test-related constraints in combination with the tester power supply characteristics and the modulator non-idealities can significantly degrade the required resolution (13bits) as shown in Figure 4.15.

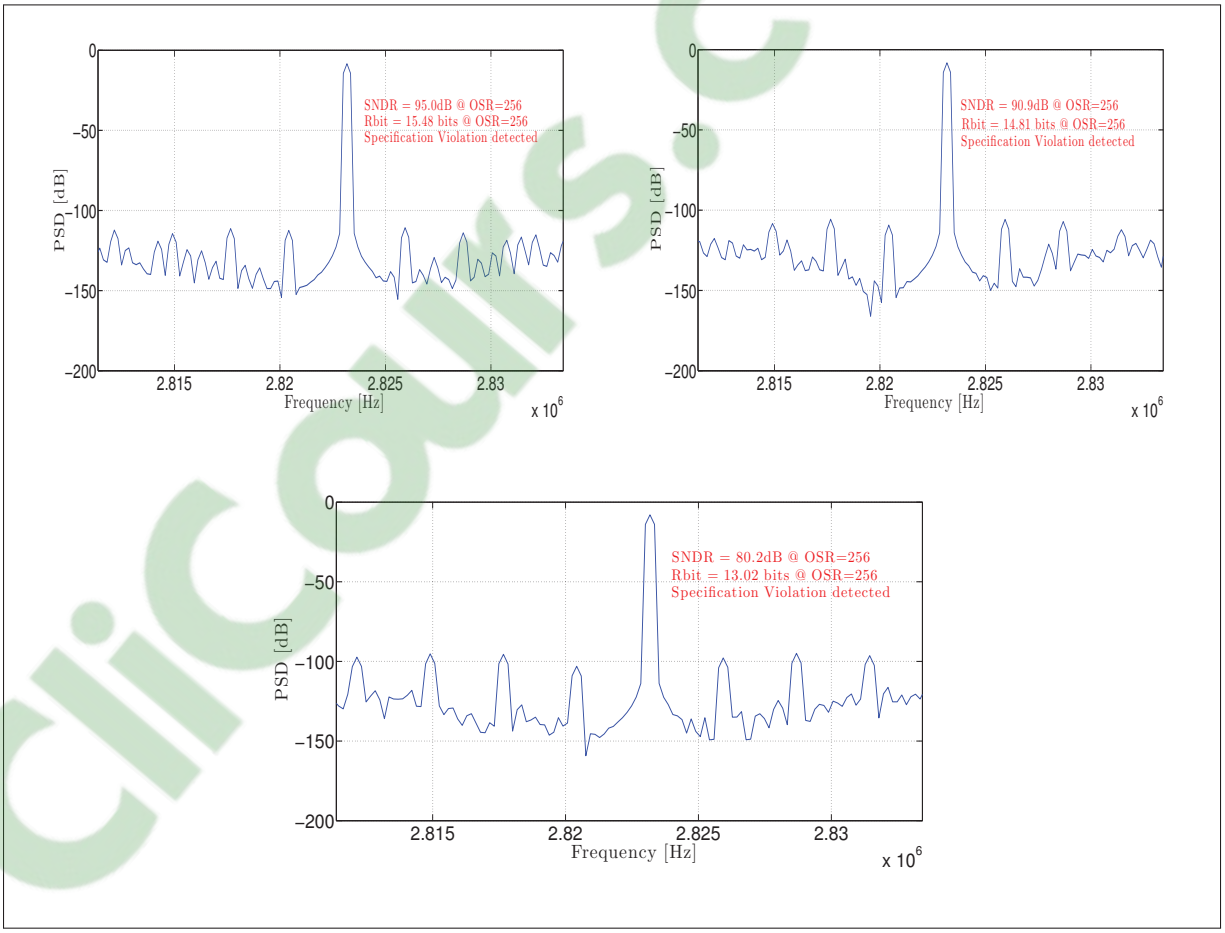

Figure 4.15 Simulation results of the SNDR and Resolution (Rbit) specifications:Results with specification and test-related constraints and tester components characteristics

It is interesting to note that the authors in (Brigati *et al.*, 1999) have shown that the SNDR and resolution (Rbit) specifications are satisfied. However, that is not the case when the effect of

specification and test-related constraints combined with the tester power supply characteristics are considered. In fact, only a complete and detailed definition including all the additional specification and test-related constraints as well as the tester power supply characteristics ensures that the required specifications are satisfied (Table 4.5, columns 4-5) otherwise the design performances degrade and the algorithm reports a specification violation. The effect of spec-

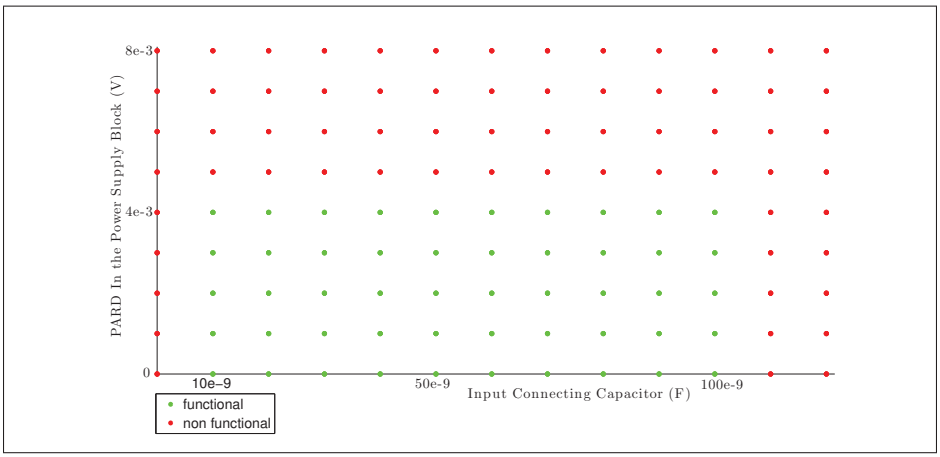

Figure 4.16 Shmoo Plotting of Sigma-Delta Modulator Results.

ification and test-related constraints combined with the tester power supply characteristics on the sigma-delta modulator can be observed using shmoo plots as shown in Figure 4.16. The figure is shown only for the input connecting capacitor with respect to the requirements related to the physical connection interface between the DUT and the test instruments. The capacitance values are generated using *Specification/TestRelatedConstraintsValuesAssignment* (line 2 in Algorithm 3.1, chapter 3) function. At each capacitance value, the power supply *PARD* is swept based on the tester characteristics used and the result is treated by writing either '1' for functional or '0' for non-functional design. It can be clearly seen that for higher values of *PARD* ($>$ 4mV), the sigma-delta modulator failed its specifications.

The above simulation results were derived for a sigma-delta modulator with non-idealities considered separately and with test-related constraints only on the input connecting capacitor. Current research interests entail combining all the modulator non idealities and adding test-

related constraints on the output connecting interface. The methodology will be also applied to a 3D sigma-delta modulator.

4.4 Complexities and Costs of the Proposed Constraint-based Verification Approach

In the following, we first recapitulate the performance of the proposed constraint-based verification algorithm using the previous benchmark circuits. We next discuss the complexity and costs of our method. Table 4.6 shows simulation run-times for the three circuits studied in the previous section. As can be seen from the table 4.6, the total elapsed time necessary to process all the verification tasks can be several hours. For example, it takes about 2 hours to adequately verify the frequency synthesizer IC reference spur specification at frequencies around 500 MHz with data accumulation in the 1 to 2 Gbyte range.

Table 4.6 Simulation Run-Times of the constraint-based property checking algorithm

Circuit	Specifications	Time
Frequency Synthesizer IC	Lock time Output Frequency Reference Spurs	387 seconds 387 seconds 2 hours 6 minutes
3D Clock Tree	Clock Skew	27.8 seconds
Sigma-delta Modulator	SNDR Resolution	3.2 minutes 3.2 minutes

In general, the overall complexity of our method depends on the number of input specifications as well as the number of their specification and test-related constraints. Indeed the running time of the algorithm involves computing all the required specification and test-related constraints for each input specification, checking if they are adequately defined and within the specified range of values which comes at the cost of additional CPU time and sometimes high memory consumption. This means, for a given set of n input specifications with m input constraints it takes O(n*m) to verify the circuit benchmark specifications.

4.5 Summary

In this chapter, we used the proposed methodology to verify the specifications of a CP-PLL frequency synthesizer and a 3D clock tree IC, where a number of out-of-specification failures was detected due to specification and test-related constraints violation, thereby proving that the proposed approach is efficient in finding bugs early in the design process. We showed that the guaranteed specifications (with respect to the design datasheet) apply only with a complete and detailed definition including all the additional specification and test-related constraints otherwise the design performances degrade and the algorithm reports a specification violation. Finding these out-of-specification failures early in the design process is advantageous and valuable as it allows designers to make the necessary changes before committing designs to manufacture which reduces the risk of costly errors caused by incomplete details in the specification documents.

In addition to specification and test-related constraints, tester components characteristics were considered during the modeling and verification phases in order to assess the impact of the mixed-signal tester inaccuracies on the reliability of the specifications test. The extended approach was applied to sigma-delta modulator where a number of out-of-specification failures was detected due to specification and test-related constraints and tester components characteristics violation. Advantages of the extended approach are robustness and flexibility to account for external mixed-signal tester components characteristics. Likewise, the approach estimates the acceptance/rejection of AMS circuits with reference to these characteristics. Nevertheless, the approach presents some limitations as it inherits the coverage limitation drawbacks from standard simulation techniques. In addition, the verification of some complex specifications needs the gathering of simulation data which can cost a lot of memory resources.

CONCLUSION AND RECOMMENDATIONS

The need for efficient verification approaches that allow first pass success to complex AMS designs is becoming more of a requirement rather than a luxury. That was motivated by the fast production ramp-ups and tight budget constraints. Indeed, the AMS design process confronts two major obstacles: The first obstacle is that the complexity inherent in such systems places a burden on written specifications documents (specifications requirements tend to suffer from ambiguity). The second obstacle is the stringent testing requirements that must be accounted for to adequately meet these specifications. In this thesis, we have presented a constraint-based verification methodology that addresses both obstacles. The methodology is based on the early insertion of test(s) associated with each design specification. It allows to integrate specification and test-related constraints in a simulation-based verification environment at the system level.

The concept of early insertion of tests clearly demonstrates its feasibility to make apparent hidden out-of-specification failures, usually dealt with during the test phase after committing designs to fabrication, at higher levels of abstraction. In that it avoids the omission of important specification and test-related constraints such that the tests cannot be properly achieved. In addition, it helps in the early discovery of ambiguous, missing and incomplete specifications by forcing a detailed analysis of all specifications. Since industrial verification techniques are based on simulation, we believe that our methodology can be quite helpful in developing successful AMS designs. The thesis contributes in three main directions.

1. Firstly, it provides a systematic analysis of the completeness and coherence of AMS designs specifications with respect to specification and test-related constraints. Accordingly, it allows to perform a worst-case tolerance analysis through which the tolerance levels on each of the specification and test-related constraints are identified. These are then used to assign the appropriate values (oftentimes not clearly specified or not provided in the specifications documents) to the constraints. The analysis showed its ef-

fectiveness to complement (or to add when missing) the definition of certain constraints values.

2. Secondly, it presents a first attempt to automatically assess the impact of test-related constraints on AMS designs performance at a high level of abstraction. The idea is to integrate specification and test-related constraints into the AMS design behavioral model whose output is evaluated in a simulation-based verification environment. The advantage of the proposed methodology is that the best compromise between functionality, performance, and test can be reached since specifications and test requirements become another set of prioritised constraints that must be considered. The methodology has been gainfully applied on a CP-PLL frequency synthesizer and a 3D clock tree verification, where different out-of-specification failures were detected due to specification and test-related constraints violation, thereby enhancing the detection of hidden bugs as early as possible in the design process.

3. Thirdly, it presents a methodology to handle additional set of performance features derived from the mixed-signal tester components used for specifications tests. The combination of specification and test-related constraints with tester's instrument characteristics allows to assess the impact of the mixed-signal tester inaccuracies on the reliability of the specifications test. The methodology estimates the acceptance/rejection of the AMS design with respect to the tester's characteristics. The practical effectiveness of the proposed framework is compared for a sigma-delta modulator.

The approach we outlined in this thesis is a first step towards integrating specification and test-related constraints into verification strategies of AMS designs. In this light, our findings open a new avenue for research that may contribute to the development of successful complex AMS designs. As future work, more investigation is needed to improve the approach to verify important specifications related to industrial problems. For example, adding more features to

strengthen the capabilities of the approach to handle industrial 3D ICs. Also, the constraints extraction from the specifications documents is done manually, investigating alternative implementations is needed to automate the extraction. As formal verification techniques such as model checking started lately gaining more attention in industry, a research direction to use model checking for proving the properties of 3D ICs in the presence of specification and test-related constraints can be explored.

APPENDIX I

A CONSTRAINT-BASED PROPERTY CHECKING TOOL FOR 2D AND 3D AMS DESIGNS

The overall thesis framework is established as a Constraint-based Property Checker tool for automatic modeling and verification of AMS designs in the presence of specification and test-related constraints. The tool is implemented using Matlab based object-oriented approach in form of object classes and functions. Figure I-1 illustrates the UML package diagram summarizing the tool's classes and their relationships.

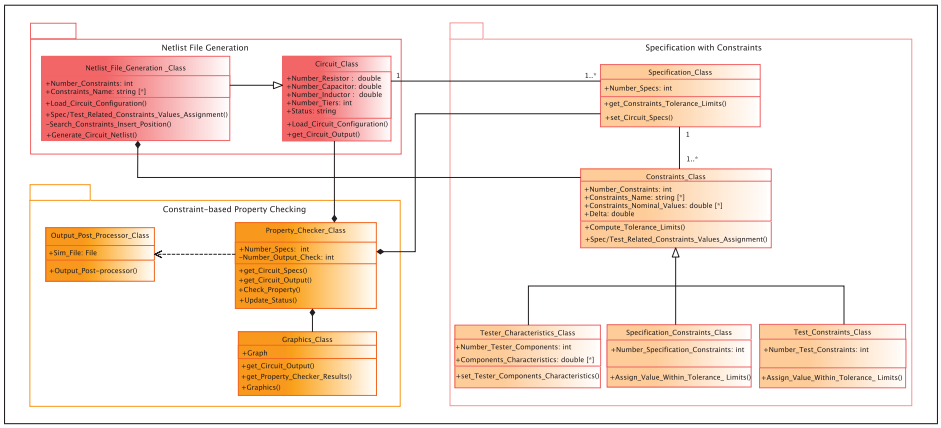

Figure-A I-1 UML Package Diagram for the Constraint-based Property Checking Tool

The classes are defined as follows:

1. **Specification_Class**: This class defines the circuit specifications with specification and test-related constraints. The method *get_Constraints_Tolerance_Limits* is used to get the tolerance limits for each constraint. These tolerance limits define the range of values in which specifications are guaranteed to pass the test. *set_Circuit_Specs* is used to set the specifications and their associated constraints to the specified intervals. This is useful to complement the definition of certain constraints values that might not be defined in the design preliminary datasheet.

2. **Constraints_Class**: This class provides the tolerance levels on each of the specifications constraints based on their nominal values. This is done using *Compute_Tolerance_Limits* method in which each constraint's value is varied in a predefined percentage increment or decrement *Delta*. *Specification/Test_Related_Constraints_Values_Assignment* method is used to assign a value to each constraint parameter. An example of such an assignment would be to use values at the limits of the tolerance for a given constraint parameter.

3. **Circuit_Class**: This class defines the AMS design model initially provided by the designer. *Load_Circuit_Configuration* returns the initial circuit netlist with the initial conditions of the circuit current and voltages, step size, and simulation parameters. On the other hand, *get_Circuit_Output* is responsible for calling the PSpice circuit simulator with the obtained input netlist and saving the generated output. It also parses the output report generated by the simulator and extracts the required data.

4. **Netlist_File_Generation_Class**: Based on the initial AMS circuit netlist, this class generates a new netlist file that represents the AMS design behavioral model with the additional specification and test-related constraints. The output is a file with a specific extension, such as '.cir' which is then fed into the PSpice circuit simulator. During netlist generation, *Generate_Circuit_Netlist* method operates in two modes: 1. A read mode that reads the input files containing the initial AMS circuit netlist and the extracted specification and test-related constraints. 2. A create mode during which it integrates the specification and test-related constraints into the initial AMS circuit netlist. Constraints integration requires the position at which additional specification and test-related constraints should be inserted. This is determined a priori using *Search_Constraints_Insert_Position* method that finds the position of a target constraint within the initial AMS circuit netlist. This class includes a built-in checker that checks whether a test scheme is defined for each specification. In case there is no test scheme defined for a given specification, an error message indicating a missing test scheme is generated. This means a successful netlist file generation requires the definition of a test scheme together with the associated constraints for each specification.

5. **Output_Post_Processor_Class**: This class is used when a certain amount of post processing is necessary to make sense of the extracted output data. Depending on the circuit output, the *Output_Post-processor* method is used for post-processing the simulation results from PSpice circuit simulator.

6. **Property_Checker_Class**: This class performs the specifications monitoring on offline simulation traces. *Check_property()* method is used to evaluate whether or not a target specification is satisfied for a given simulation trace in the presence of specification and test-related constraints.

7. **Graphics_Class**: This class allows to display the verification results obtained from the *Property_Checker_Class*.

BIBLIOGRAPHY

Acco, P., Kennedy, M. P., Mira, C., Morley, B. & Frigyik, B. 1999. Behavioral Modeling of Charge Pump Phase Locked Loops. *in Proc. ISCAS'99*, 1, 375-378.

Advanced Micro Devices. 2005, Aug. Inc. Revision Guide for AMD Athlon (TM) 64 and AMD Opteron(TM) Processors.

Ain, A. & Dasgupta, P. 2015. Monitoring AMS Simulation: From Assertions to Features. *In VLSI Design*, pp. 429-434.

Ain, A., da Costa, A. A. B. & Dasgupta, P. 2016. Feature Indented Assertions for Analog and Mixed-Signal Validation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(11), 1928-1941.

Al Sammane, G., Zaki, M. H., Dong, Z. J. & Tahar, S. 2007. Towards Assertion Based Verification of Analog and Mixed Signal Designs Using PSL. *in: Languages for Formal Specification and Verification, Forum on Specification and Design Languages*.

Al-Sammane, G., Zaki, M. H. & Tahar, S. 2007, April. A Symbolic Methodology for the Verification of Analog and Mixed Signal Designs. *Proc. IEEE/ ACM Design Automation and Test in Europe (DATE'07), Nice, France*.

Althoff, M., Yaldiz, S., Rajhans, A., Li, X., Krogh, B. H. & Pileggi, L. 2011. Formal Verification of Phase-Locked Loops Using Reachability Analysis and Continuization. *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 659-666.

Andreu, D., Stepan, D. & Josse, S. 2001. Introduction to Behavioral Modeling and Phase Loop Locked Behavioral Modeling. *Newsletter www.spice-club.com*.

Ankele, B., Holzl, W. & O'Leary, P. 1989, Jul. Enhanced MOS Parameter Extraction and SPICE Modelling for Mixed Analogue and Digital Circuit Simulation. *in Proc. IEEE Int. Conf. Microelectron. test Struct*, pp. 133-137.

Arabi, K. 2010. Power Noise And Its Impact On Production Test and Validation Of SOC Devices. *IEEE VLSI Test Symposium (VTS), 28th*, pp. 285-285.

Asarin, E., Caspi, P. & Maler, O. 2002. Timed Regular Expressions. *Journal of the ACM*, 49(2), 172-206.

Balivada, A., Hoskote, Y. & Abraham, J. A. 1995. Verification of Transient Response of Linear Analog Circuits. *In: IEEE VLSI Test Symposium*, pp. 42-47.

Banerjee, K., Souri, S. J., Kapur, P. & Saraswat, K. C. 2001. 3-D ICs: A Novel Chip Design for Improving Deep-Submicrometer Interconnect Performance and Systems-on-Chip Integration. *Proceedings of the IEEE*, 89(5).

Banerjee, S., Mukhopadhyay, D. & Chowdhury, D. R. 2005. Computer Aided Test (CAT) Tool for Mixed Signal SOCs. *IEEE VLSI Design*, pp. 787-790.

Batchu, S. 2010. *Automatic Extraction of Behavioral Models from Simulations of Analog/Mixed-Signal (AMS) Circuits*. (Master's thesis, University of Utah).

Bemporad, A. & Morari, M. 1999. Verification of Hybrid Systems Via Mathematical Programming. *In Hybrid Systems: Computation and Control, LNCS 1569*, pp. 31-45.

Best, R. E. 2003. *Phase-Locked Loops: Design, simulation and applications* (ed. 5). McGraw Hill Publishers.

Bjornsen, J. & Ytterdal, T. 2003. Behavioral Modeling and Simulation of High-Speed Analog-to-Digital Converters using SystemC. *In Proceedings of the 2003 International Symposium on Circuits and Systems*, 3.

Bonnerud, T. E., Hernes, B. & Ytterdal, T. 2001. A mixed-signal, functional level simulation framework based on SystemC for system-on-a-chip applications. *in Proc. IEEE Conference on Custom Integrated Circuits (CICC'01)*, pp. 541-544.

Brigati, S., Francesconi, F., Malcovati, P., Tonietto, D., Baschirotto, A. & Maloberti, F. 1999. Modeling sigma-delta modulator non-idealities in SIMULINK (R). *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'99)*, 2, 384-387.

Buhler, M., Koehl, J., Bickford, J., Hibbeler, J., Schlichtmann, U., Sommer, R., Pronath, M. & Ripp, A. 2006. DFM/DFY Design for Manufacturability and Yield - Influence of Process Variations in Digital, Analog and Mixed-Signal Circuit Design. *IEEE/ACM Design, Automation and Test in Europe*, pp. 387-392.

Burns, M. & Roberts, G. 2001. *An Introduction to Mixed-Signal IC Test and Measurement*. New York: Oxford University Press.

Cadence Design Systems. 2007. *Using a SoC Functional Verification Kit to Improve Productivity, Reduce Risk, and Increase Quality.* Consulted at http://w2.cadence.com/whitepapers/SoC_fv_Kit_wp.pdf.

Capture CIS version 9.2.3. Cadence Design Systems, Inc.

Cho, M., Liu, C., Kim, D. H., Lim, S. K. & Mukhopadhyay, S. 2011. Pre-bond and Post-bond Test and Signal Recovery Structure to Characterize and Repair TSV Defect Induced Signal Degradation in 3-D System. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 1(11), 1718-1727.

Chuang, Y.-L., Yuan, C.-S., Chen, J.-J., Chen, C.-F., Yang, C.-S., Changchien, W.-P., Liu, C. C. & Lee, F. 2013. Unified Method for Heterogeneous Integration of CoWoS Technology. *IEEE Electronic Components and Technology Conference (ECTC)*, pp. 852-859.

Chung, K. 2012. DFT Challenges in 3D ICs. *Test Technology Workshop*.

Clarke, E. M., Grumberg, O. & Peled, D. 1999. *Model Checking*. MIT press.

Committee, D. A. S. et al. 2005a. IEEE Standard for Property Specification Language (PSL), Version 1.1 Standard IEEE 1850. *http://www. eda. org/vfv*.

Committee, D. A. S. et al. 2005b. IEEE Standard for SystemVerilog Unified Hardware Design, Specification, and Verification Language Standard IEEE 1800. *http://www. edastds. org/sv/*.

Comte, M. 2003. *Etude des Corrélations entre Paramètres Statiques et Dynamiques des CAN en vue d'optimiser leur Flot de Test*. (Ph.D. thesis, Université Montpellier II).

Comte, M., Bernard, S., Azais, F., Bertrand, Y. & Renovell, M. 2003, Sept. A New Methodology for ADC Test Flow Optimization. *in Proc. 2003 International Test Conference*, pp. 201-209.

Cong, J., Jagannathan, A., Ma, Y., Reinman, G., Wei, J. & Zhang, Y. 2006, January. An Automated Design Flow for 3D Microarchitecture Evaluation. *Proceedings of Asia and South Pacific Design Automation Conference*.

Crandall, E. 1997. *Power Supply Testing Handbook: Strategic Approaches in Test Cost Reduction*. Springer Science & Business Media.

Cyrluk, D., Rajan, S., Shankar, N. & Srivas, M. K. 1995. Effective Theorem Proving For Hardware Verification. *Theorem Provers in Circuit Design*, pp. 203-222.

da Costa, A. A. B. & Dasgupta, P. 2015. Formal Interpretation of Assertion-based Features on AMS Designs. *IEEE Design & Test*, 32(1), 9-17.

Dang, T., Donzé, A. & Maler, O. 2004. Verification of Analog and Mixed-signal Circuits using Hybrid System Techniques. *In Formal Methods in Computer-Aided Design, LNCS 3312*, pp. 14-17.

Dantes HP83000. 1998. Multimedia Series - Technical Description, Integrated Measurement Systems (IMS), Inc. Beaverton, Oregon, 1998.

Das, S., Fan, A., Chen, K.-N., Tan, C. S., Checka, N. & Reif, R. 2004. Technology, Performance, and Computer-Aided Design of Three-Dimensional Integrated Circuits. *Proceedings of the International Symposium on Physical Design*, pp. 108-115.

Dastidar, T. R. & Chakrabarti, P. 2005. A Verification System for Transient Response of Analog Circuits Using Model Checking. *In IEEE International Conference on VLSI*, pp. 195-200.

Dastidar, T. R. & Chakrabarti, P. 2007. A Verification System for Transient Response of Analog Circuits. *In ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 12(3), 1-39.

Davis, W. R., Wilson, J., Mick, S., Xu, J., Hua, H., Mineo, C., Sule, A. M., Steer, M. & Franzon, P. D. 2005. Demystifying 3D ICs: The Pros and Cons of Going Vertical. *IEEE Design and Test of Computers*, 22(6), 498-510.

Demir, A. 1997. *Analysis and Simulation of Noise in Nonlinear Electronic Circuits and Systems*. (Ph.D. thesis, University of California, Berkeley).

Denman, W., Akbarpour, B., Tahar, S., Zaki, M. H. & Paulson, L. C. 2009. Formal Verification of Analog Designs Using MetiTarski. *In Proc. Formal Methods in Computer-Aided Design*, pp. 93-100.

Dong, X. & Xie, Y. 2009, Jan. System-Level Cost Analysis and Design Exploration for Three-Dimensional Integrated Circuits (3D ICs). *Design Automation Conference, ASP-DAC. Asia and South Pacific*, pp. 234-241.

Dubikhin, V., Sokolov, D., Yakovlev, A. & Myers, C. J. 2016. Design of Mixed-Signal Systems with Asynchronous Control. *IEEE Design & Test*, 33(5), 44-55.

Eisawy, A. 2012. Improving Analog/Mixed- Signal Verification Productivity, Mentor Graphics.

FarWest Research and Mentor Graphics. 2007. EDA Market Statistics Service Report.

Ferrere, T. 2016. *Assertions and Measurements for Mixed-Signal Simulation*. (Ph.D. thesis, Université Grenoble-Alpes, France).

Ferri, C., Reda, S. & Bahar, R. 2007, Nov. Strategies for Improving the Parametric Yield and Profits of 3D ICs. *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 220-226.

Fisher, A., Kulkarni, D. & Myers, C. J. 2015. A New Assertion Property Language for Analog/Mixed-Signal Circuits. In *In M.-M. Louerat and T. Maehne, editors, Languages, Design Methods, and Tools for Electronic System Design, volume 311 of Lecture Notes in Electrical Engineering* (pp. 45-65). Springer.

Fisher, A. N., Batchu, S., Jones, K., Kulkarni, D., Little, S., Walter, D. & Myers, C. J. 2014. LEMA: A Tool For the Formal Verification of Digitally-Intensive Analog/Mixed-Signal Circuits. *In Proc. International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1017-1020.

Foster, H. D. & Krolnik, A. C. 2010. *Creating Assertion-Based IP (Integrated Circuits and Systems)*. Springer.

Freescale Semiconductor. 1999. Dual 550/60 MHz PLL Frequency Synthesizer with DACs and Voltage Multiplier, MC145181 datasheet, Motorola, Inc. 1999. Rev 1.

Frehse, G., Krogh, B. H. & Rutenbar, R. A. 2006a. Verifying Analog Oscillator Circuits Using Forward/Backward Abstraction Refinement. *In IEEE/ACM Design, Automation and Test in Europe*, pp. 257-262.

Frehse, G., Krogh, B. H., Rutenbar, R. A. & Maler, O. 2006b. Time Domain Verification of Oscillator Circuit Properties. *Electronic Notes in Theoretical Computer Science*, 153(3), 9-22.

Freibothe, M., Schonherr, J. & Straube, B. 2006. Formal Verification of The Quasi-Static Behavior of Mixed-Signal Circuits by Property Checking. *Electronic Notes in Theoretical Computer Science*, 153(3), 23-35.

Fulde, M. 2010. Variation Aware Analog and Mixed-Signal Circuit Design in Emerging Multi-Gate CMOS Technologies. *Springer Series in Advanced Microelectronics*.

Gambino, J. P., Adderly, S. A. & Knickerbocker, J. U. 2015. An Overview of Through-Silicon-Via Technology and Manufacturing Challenges. *Microelectronic Engineering*, 135, 73-106.

Garg, S. & Marculescu, D. 2009. System-Level Process Variability Analysis and Mitigation for 3D MPSoCs. *Design, Automation & Test in Europe Conference & Exhibition, DATE'09.*, pp. 604-609.

Gerlach, J. & Rosenstiel, W. System Level Design Using the SystemC Modeling Platform. In *System Design Automation*. Springer.

Ghosh, A. & Vemuri, R. 1999. Formal Verification of Synthesized Analog Designs. *In ACM/IEEE International Conference on Computer Design*, pp. 40-45.

Gope, D., Chatterjee, S., De Araujo, D., Chakraborty, S., Pingenot, J. & Camposano, R. 2013. Device Physics Aware 3D Electromagnetic Simulation of Through-Silicon-Vias in System Modeling. *in 3D Systems Integration Conference (3DIC), 2013 IEEE International*, pp. 1-5.

Grabowski, D., Grimm, C. & Barke, E. 2006a. Semi-Symbolic Modeling and Simulation of Circuits and Systems. *In Proc. International Symposium on Circuits and Systems (ISCAS)*, pp. 983-986.

Grabowski, D., Platte, D., Hedrich, L. & Barke, E. 2006b. Time Constrained Verification of Analog Circuits using Model-Checking Algorithms. *Electronic Notes in Theoretical Computer Science*, 153(3), 37-52.

Greenstreet, M. R. 1996. Verifying Safety Properties of Differential Equations. *In Computer Aided Verification LNCS*, 1102, 277-287.

Greenstreet, M. R. & Mitchell, I. 1999. Reachability Analysis Using Polygonal Projections. *In Hybrid System: Computation and Control, LNCS 1569*, pp. 103-116.

Gupta, S., Krogh, B. H. & Rutenbar, R. A. 2004. Towards Formal Verification of Analog Designs. *In Proc. IEEE/ACM Conference on Computer Aided Design*, pp. 210-217.

Hafer, T. & Thomas, W. 1987. Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree. *International Colloquium on Automata, Languages, and Programming*, pp. 269-279.

Hanna, K. 1994. Reasoning About Real Circuits. In *In Theorem Proving in Higher Order Logic, LNCS 859* (pp. 235-253). Springer.

Hanna, K. 1998. Automatic Verification of Mixed-Level Logic Circuits. In *In Formal Methods in Computer-Aided Design, LNCS 1522* (pp. 133-166). Springer.

Hanna, K. 2000. Reasoning About Analog-Level Implementations of Digital Systems. *Formal Methods in System Design*, 16(2), 127-158.

Hartong, W., Klausen, R. & Hedrich, L. 2004. Formal Verification for Nonlinear Analog Systems: Approaches to Model and Equivalence Checking. In *Advanced Formal Verification* (pp. 205-245). Kluwer.

Havlicek, J. & Little, S. 2011. Realtime Regular Expressions for Analog and Mixed-Signal Assertions. *In Formal Methods in Computer Aided Design FMCAD Inc*, pp. 155-162.

Hedrich, L. & Barke, E. 1995. A Formal Approach to Nonlinear Analog Circuit Verification. *In IEEE/ACM International Conference on Computer-Aided Design*, pp. 123-127.

Hedrich, L. & Barke, E. 1998. A Formal Approach to Verification of Linear Analog Circuits with Parameter Tolerances. *In IEEE/ACM Design, Automation and Test in Europe*, pp. 649-654.

Hogan, M. & Petranovic, D. 2009, Sept. Robust Verification of 3D-ICs: Pros, Cons and Recommendations. *3D System Integration, 2009. 3DIC 2009. IEEE International Conference on*, pp. 1-6.

Horowitz, M. 2011. Digital Analog Design. *Workshop on Frontiers in Analog Circuit Synthesis and Verification*.

Huang, W.-H. & Lei, Y. V. 2002. 2-D and 3-D Performance Analysis of a Digitally Controlled CMOS Class-E Amplifier. *Technical report, Massachusetts Institute of Technology, Course 6.374, Fall 2002*.

Intel Corporation. 2000, Feb. Intel(R) Strong ARM(R) SA-1100 Microprocessor Specification Update.

Intel Corporation. 2007, Sep. Intel Core 2 Duo and Intel Core 2 Solo Processor for Intel Centrino Duo Processor Technology Specification Update.

Intel Corporation. 2010, Jul. Intel Core i7-900 Desktop Processor Series Specification Update.

International Business Machines Corporation. 2005, Jul. IBM PowerPC 750GX and 750GL RISC Microprocessor Errata Notice.

Jaykar, S., Palsodkar, P. & Dakhole, P. 2011. Modeling of Sigma-Delta Modulator Non-Idealities in MATLAB/Simulink. *IEEE International Conference on Communication Systems and Network Technologies*, pp. 310-315.

Jeff, D. 2017, June 7. Chip Test Shifts Left. Semiconductor Engineering. Consulted at https://semiengineering.com/chip-test-shifts-left.

Jones, K., Konrad, V. & Ničković, D. 2008. Analog Property Checkers: A DDR2 Case Study. *In Proc. Formal Verification of Analog Circuits (FAC).*

Ken, K., Henry, C., Dan, J., Gilles, L., Enrico, M. & Fred, S. 2000. Design of Mixed-Signal Systems-On-a-Chip. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 19(12), 1561-1571.

Khan, N. H., Alam, S. M. & Hassoun, S. 2009. Through-Silicon Via (TSV)-Induced Noise Characterization and Noise Mitigation Using Coaxial TSVs. *In IEEE 3D System Integration (3DIC)*, pp. 1-7.

Koo, K., Kim, M., Kim, J. J., Kim, J. & Kim, J. 2013. Vertical Noise Coupling from On-Chip Switching-Mode Power Supply in a Mixed-Signal Stacked 3-D-IC. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 3(3), 476-488.

Koymans, R. 1990. Specifying Real-Time Properties with Metric Temporal Logic. *Real-time systems*, 2(4), 255-299.

Kropf, T. 1999. *Introduction to Formal Hardware Verification*. Springer.

Kulkarni, D. 2013. *Improved Model Generation and Property Specification for Analog/Mixed-Signal Circuits*. (Master's thesis, University of Utah).

Kundert, K., White, J. & Sangiovanni-Vincentelli, A. 1988, November. An Envelope-Following Method for the Efficient Transient Simulation of Switching Power and Filter Circuits. *IEEE International Conference on Computer-Aided Design: Digest of Technical Papers*.

Kundert, K. 2003. Predicting the Phase Noise and Jitter of PLL-Based Frequency Synthesizers. https://www.designers-guide.com.

Kundert, K. & Chang, H. 2006. Verification of Complex Analog Integrated Circuits. *IEEE Custom Integrated Circuits Conference*, pp. 177-184.

Kundert, K. & Chang, H. 2009. Top-Down Design and Verification of Mixed-Signal Circuits. *Designers Guide Consulting, http://www.designers-guide.com/docs/tddv.pdf.*

Kundert, K. S., White, J. & Sangiovanni-Vincentelli, A. 1989. A Mixed Frequency-Time Approach for Distortion Analysis of Switching Filter Circuits. *IEEE Journal of Solid-State Circuits*, 24(2), 443-451.

Kundert, K. S., White, J. K. & Sangiovanni-Vincentelli, A. 1990. *Steady-State Methods for Simulating Analog and Microwave Circuits*. Kluwer Academic Publishers.

Kurshan, R. P. & McMillan, K. L. 1991. Analysis of Digital Circuits Through Symbolic Reduction. *IEEE Transactions on Computer-Aided Design*, 10(11), 1356-1371.

Lau, W. Y. 2002. Measurement Challenges For On-wafer RF-SOC Test. *In Proceedings of Electronics Manufacturing Technology Symposium*, pp. 353-359.

Lee, H.-H. S. & Chakrabarty, K. 2009. Test Challenges for 3D Integrated Circuits. *IEEE Design & Test of Computers*, 26(5), 26-35.

Lee, M., Cho, J. & Kim, J. 2012. Noise Coupling Analysis Between TSV and Active Circuit. *in: IEEE Elec. Des. Advanced Packaging Systems Symp. (EDAPS)*, pp. 45-48.

Lee, S., Kim, G., Kim, J., Song, T., Lee, J., Lee, H., Park, K. & Kim, J. 2009. Modeling and Analysis of Die-to-Die Vertical Coupling in 3-D IC. *Electronics Packaging Technology Conference, 2009. EPTC'09. 11th*, pp. 707-711.

Li, W. & Meiners, J. 2000. Introduction to Phase-Locked Loop System Modeling. *Analog Applications Journal, Texas Instruments Incorporated*, 5-10.

Lim, B. C., Jang, J.-E., Mao, J., Kim, J. & Horowitz, M. 2015. Digital Analog Design: Enabling Mixed-Signal System Validation. *IEEE Design & Test*, 32(1), 44-52.

Lin, L. J.-H., Chang, H.-P., Wu, T.-L. & Chiou, Y.-P. 2012. 3D Simulation of Substrate Noise Coupling from Through Silicon Via (TSV) and Noise Isolation Methods. *in: IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*, pp. 181-184.

Little, S., Walter, D., Jones, K., Myers, C. & Sen, A. 2010. Analog/Mixed-Signal Circuit Verification Using Models Generated from Simulation Traces. *International Journal of Foundations of Computer Science*, 21(2), 191-210.

Little, S., Walter, D., Myers, C., Thacker, R., Batchu, S. & Yoneda, T. 2011. Verification of Analog/Mixed-Signal Circuits Using Labeled Hybrid Petri Nets. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(4), 617-630.

Liu, C., Song, T., Cho, J., Kim, J., Kim, J. & Lim, S. K. 2011. Full-chip TSV-to-TSV Coupling Analysis and Optimization in 3D IC. *in Proceedings of the 48th ACM/EDAC/IEEE Design Automation Conference, New York, NY*, pp. 783-788.

Maler, O. & Nickovic, D. 2004. Monitoring Temporal Properties of Continuous Signals. In *In Formal Modelling and Analysis of Timed Systems, LNCS 3253* (pp. 152-166). Springer.

Maler, O. & Ničković, D. 2013. Monitoring Properties of Analog and Mixed-Signal Circuits. *International Journal on Software Tools for Technology Transfer*, 15(3), 247-268.

Maler, O., Nickovic, D. & Pnueli, A. 2005. Real Time Temporal Logic: Past, present, future. In *In Formal Modeling and Analysis of Timed Systems, LNCS 3829* (pp. 2-16). Springer.

Maler, O., Nickovic, D. & Pnueli, A. 2006. From MITL to Timed Automata. In *In Formal Modeling and Analysis of Timed Systems, LNCS 4202* (pp. 274-289). Springer.

Marinissen, E. J. 2013. Creating Options for 3D-SIC Testing. *In: IEEE Int. Symp. VLSI Design, Automation, and Test (VLSI-DAT)*, pp. 1-7.

Marinissen, E. J. & Zorian, Y. 2009. Testing 3D Chips Containing Through-Silicon Vias. *Proc. ITC Paper ET1. 1*.

Marinissen, E. J., Chi, C.-C., Verbree, J. & Konijnenburg, M. 2010. 3D DfT Architecture for Pre-bond and Post-bond Testing. *In Proc. 3D Systems Integration Conference (3DIC)*, pp. 1-8.

Markoff, J. 2008. Burned once, Intel prepares new chip fortified by constant tests. *New York Times*.

Mathis, W. & Thiessen, T. 2009. On Noise Analysis of Oscillators Based on Statistical Mechanics. *International Conference on Mixed Design of Integrated Circuits & Systems,*, pp. 472-477.

Matsumoto, K., Ibaraki, S., Sueoka, K., Sakuma, K., Kikuchi, H., Mori, H., Orii, Y., Yamada, F., Fujihara, K., Takamatsu, J. et al. 2013, Oct. Thermal Design Guideline and New Cooling Solution for a Three-Dimensional (3D) Chip Stack. *In 3D Systems Integration Conference (3DIC), 2013 IEEE International*, pp. 1-8.

Minas, N., De Wolf, I., Marinissen, E. J., Stucchi, M., Oprins, H., Mercha, A., Van der Plaas, G., Velenis, D. & Marchai, P. 2010. 3D Integration: Circuit Design, Test, and Reliability Challenges. *In: IEEE 16th International On-Line Testing Symposium*, pp. 217-217.

Minz, J., Zhao, X. & Lim, S. K. 2008. Buffered Clock Tree Synthesis for 3D ICs Under Thermal Variations. *Proceedings of the 2008 Asia and South Pacific Design Automation Conference*, pp. 504-509.

MIT Lincoln Labs. 2006. MITLL Low-Power FDSOI CMOS Process: Designer Guide. MIT, Cambridge, MA.

Mukherjee, P. & Li, P. 2016. Using Presilicon Knowledge to Excite Nonlinear Failure Modes in Large Mixed-Signal Circuits. *IEEE Design & Test*, 33(5), 28-34.

Mukherjee, S., Dasgupta, P., Mukhopadhyay, S., Little, S., Havlicek, J. & Chandrasekaran, S. 2012. Synchronizing AMS Assertions With AMS Simulation: From Theory To Practice. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 17(4).

Mukhopadhyay, R., Panda, S., Dasgupta, P. & Gough, J. 2009. Instrumenting AMS Assertion Verification on Commercial Platforms. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 14(2), 21:1-21:47.

Nakura, T., Terao, N., Ishida, M., Ikeno, R., Kusaka, T., Iizuka, T. & Asada, K. 2016. Power Supply Impedance Emulation to Eliminate Overkills and Underkills Due to The Impedance Difference Between ATE and Customer Board. *Test Conference (ITC), IEEE International*, pp. 1-8.

Narayanan, R., Zaki, M. H. & Tahar, S. 2009. Using Stochastic Differential Equation for Assertion Based Verification of Noise in Analog/RF Circuits. *IEEE International Mixed-Signals, Sensors, and Systems Test Workshop*, pp. 1-8.

Narayanan, R., Akbarpour, B., Zaki, M. H., Tahar, S. & Paulson, L. C. 2010a. Formal Verification of Analog Circuits in the Presence of Noise and Process Variation. *In Proc. Design, Automation & Test in Europe (DATE)*, pp. 1309-1312.

Narayanan, R., Zaki, M. H. & Tahar, S. 2010b. Using Stochastic Differential Equation for Verification of Noise in Analog/RF Circuits. *Journal of Electronic Testing*, 26(1), 97-109.

Narayanan, R., Seghaier, I., Zaki, M. H. & Tahar, S. 2013. Statistical Run-Time Verification of Analog Circuits in Presence of Noise and Process Variation. *IEEE Transactions on Very Large Scale Integration Systems*, 21(10), 1811-1822.

Navin, V. K., Ray, T., Hassoun, M. M., Black, W. C., Lee, E. K., Soenen, E. G. & Geiger, R. L. 1997. A Simulation Environment for Pipelined Analog-to-Digital Converters. *in Proc. IEEE International Symposium on Circuits and Systems (ISCAS'97)*, 3, 1620-1623.

Nguyen, T. & Ničković, D. 2016. Assertion-based Monitoring in Practice - Checking Correctness of An Automotive Sensor Interface. *Science of Computer Programming*, 118, 40-59.

Nickovic, D. & Maler, O. 2007. AMT: A Property-based Monitoring Tool for Analog Systems. In *In Formal Modelling and Analysis of Timed Systems, Austria, LNCS 4763* (pp. 304-319). Springer.

Noia, B. & Chakrabarty, K. 2011. Testing and Design-For-Testability Techniques for 3D Integrated Circuits. *Proc. ATS*, pp. 474-479.

Ozdemir, S., Pan, Y., Das, A., Memik, G., Loh, G. & Choudhary, A. 2010. Quantifying and Coping with Parametric Variations in 3D-Stacked Microarchitectures. *Proceedings of the 47th Design Automation Conference*, pp. 144-149.

Paper, P., Deen, M. J. & Marinov, O. 2005. Noise in Advanced Electronic Devices and Circuits. *in Proc. AIP Int. Conf. Noise Phys. Syst. 1/f Fluctuat.*, 780(1), 3-12.

Peng, Y. & Greenstreet, M. 2015. Integrating SMT with Theorem Proving for Analog/Mixed-Signal Circuit Verification. *NASA Formal Methods Symposium*, pp. 310-326.

Petranovic, D. & Chow, K. 2011. 3D-IC System Verification Methodology: Solutions and Challenges. *Electronic Design Processes Symp.*, pp. 1-5.

Pichon, F., Blanc, S. & Candaele, B. 1995, March. Mixed-Signal Modelling in VHDL for System-on-Chip Applications. *in Proc. European Design and Test Conference (ED&TC 1995)*, pp. 218-222.

Radojicic, C. & Grimm, C. 2016. Formal Verification of Mixed-Signal Designs Using Extended Affine Arithmetic. *In Proc. Ph. D. Research in Microelectronics and Electronics (PRIME)*, pp. 1-4.

Rahman, A. 2001. *System-Level Performance Evaluation of Three-Dimensional Integrated Circuits*. (Ph.D. thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA 02139).

Rashinkar, P., Paterson, P. & Singh, L. 2001. *System-on-a-Chip Verification: Methodology and Techniques*. Kluwer Academic Publishers.

Riordan, D. & Bhattacharya, P. 2012. PSL/SVA Assertions in SPICE. *Proceedings of The Design and Verification Conference and Exhibition*.

Robert, H. 2005, January. How to Boost Verification Productivity. *EE Times*.

Roberts, G. W. 1997. Metrics, Techniques and Recent Developments in Mixed-Signal Testing. *Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design*, pp. 514-521.

Roberts, G. 1996. Metrics, Techniques and Recent Developments in Mixed-Signal Testing. *International Conference on Computer Aided-Design*, pp. 514-521.

Salem, A. 2002. Semi-Formal Verification of VHDL-AMS Descriptions. *In IEEE International Symposium on Circuits and Systems*, pp. 333-336.

Santos, C., Vivet, P., Colonna, J.-P., Coudrain, P. & Reis, R. 2014, Dec. Thermal Performance of 3D ICs: Analysis and Alternatives. *In 3D Systems Integration Conference (3DIC), 2014 International*, pp. 1-7.

Saraswat, K. C., Banerjee, K., Joshi, A., Kalavade, P., Kapur, P. & Souri, S. 2000. 3-D ICs: Motivation, Performance Analysis, and Technology. *In Proc. ESSCIRC*, pp. 406-414.

Seghaier, I., Zaki, M. H. & Tahar, S. 2015. Statistically Validating the Impact of Process Variations on Analog and Mixed Signal Designs. *In Proceedings of the 25th edition on Great Lakes Symposium on VLSI. ACM, New York, NY, USA*, pp. 99-102.

Selvanayagam, C. S., Lau, J. H., Zhang, X., Seah, S., Vaidyanathan, K. & Chai, T. 2009. Nonlinear Thermal Stress/Strain Analyses of Copper Filled TSV (Through Silicon Via) and Their Flip-Chip Microbumps. *IEEE Transactions on Advanced Packaging*, 32(4), 720-728.

Seshadri, S. & Abraham, J. A. 2001. Frequency Response Verification of Analog Circuits Using Global Optimization Techniques. *Journal of Electronic Testing*, 17(5), 395-408.

Shapiro, W., Murray, F. & Howarth, R. 1995. Space Mechanisms Lessons Learned Study, Volume II-Literature Review. *In: NASA Technical Memorandum*, 107047.

Smith, K., Hanaway, P., Jolley, M., Gleason, R., Strid, E., Daenen, T., Dupas, L., Knuts, B., Marinissen, E. J. & Van Dievel, M. 2011. Evaluation of TSV and Micro-bump Probing for Wide I/O Testing. *In Proc. International Test Conference (ITC)*, pp. 1-10.

Smith, L., Roy, T. & Anderson, R. 2000, Oct. Power Plane SPICE Models for Frequency and Time Domains. *Electrical Performance of Electronic Packaging Conference*.

Smith, L. D., Anderson, R. E. & Roy, T. 2001, Jul. Chip-Package Resonance in Core Power Supply Structure for a High Power Microprocessor. *In Pacific Rim/ASME International Electronic Packaging Technical Conference and Exibition*.

Song, T., Liu, C., Kim, D. H., Lim, S. K., Cho, J., Kim, J., Pak, J. S., Ahn, S., Kim, J. & Yoon, K. 2011. Analysis of TSV-to-TSV Coupling with High-Impedance Termination in 3D ICs. *in:12th International Symposium on Quality Electronic Design*, pp. 1-7.

Steinhorst, S. & Hedrich, L. 2010a. Advanced Methods for Equivalence Checking of Analog Circuits with Strong Nonlinearities. *Formal Methods in System Design*, 36(2), 131-147.

Steinhorst, S. & Hedrich, L. 2010b. Improving Verification Coverage of Analog Circuit Blocks by State Space-Guided Transient Simulation. *In Proc. International Symposium on Circuits and Systems (ISCAS)*, pp. 645-648.

Steinhorst, S. & Hedrich, L. 2011. Formal Methods for Verification of Analog Circuits. In Li, P., Silveira, L. M. & Feldmann, P. (Eds.), *Simulation and Verification of Electronic and Biological Systems* (pp. 173-192). Springer Science & Business Media.

Steinhorst, S., Jesser, A. & Hedrich, L. 2006. Advanced Property Specification for Model Checking of Analog Systems. *In Analog'06*, pp. 63-68.

Synopsys Raphael. Consulted at http://www.synopsys.com.

Taouil, M., Masadeh, M., Hamdioui, S. & Marinissen, E. J. 2015. Post-bond Interconnect Test and Diagnosis for 3-D Memory Stacked on Logic. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(11), 1860-1872.

Texas Instruments. 2018, May. Mixed Signal Microcontroller, MSP430G2955 Device Erratasheet (Rev. H).

Thain, W. E. & Connelly, J. A. 1995. Simulating Phase Noise in Phase Locked Loops with a Circuit Simulator. *IEEE International Symposium on Circuits and Systems, Vol. 3*, pp. 1760-1763.

Thati, P. & Rosu, G. 2005. Monitoring Algorithms for Metric Temporal Logic Specifications. *Electronic Notes in Theoretical Computer Science*, 113, 145-162.

The Mathworks Inc. 2011. MATLAB User Guide. Consulted at http://www.mathworks.com/.

Todri-Sanial, A. & Tan, C. S. *Physical Design for 3D Integrated Circuits*. CRC Press.

Verigy V93000 SoC Series Pin Scale Digital Cards. Consulted at http://www1.verigy.com.

Vijayaraghavan, S. & Ramanathan, M. 2005. *A Practical Guide for SystemVerilog Assertions*. Springer Science & Business Media.

Walter, D., Little, S., Myers, C., Seegmiller, N. & Yoneda, T. 2008. Verification of Analog/Mixed-Signal Circuits Using Symbolic Methods. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(12), 2223-2235.

Wang, R., Chakrabarty, K. & Bhawmik, S. 2015. Interconnect Testing and Test-path Scheduling for Interposer-based 2.5-D ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(1), 136-149.

Wang, Z. 2009. *Runtime Verification of Analog and Mixed Signal Designs*. (Master's thesis, Concordia University).

Wile, B., Goss, J. C. & Roesner, W. 2005. *Comprehensive Functional Verification: The Complete Industry Cycle*. USA: Morgan Kaufmann.

Wilson, I. M. 1989. Analog Behavioral Modeling Using PSPICE. *Proceedings of the 32nd Midwest Symposium on Circuits and Systems*, pp. 981-984.

Xie, Y., Loh, G. H., Black, B. & Bernstein, K. 2006. Design Space Exploration for 3D Architectures. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2(2), 65-103.

Xu, H., Pavlidis, V. F. & De Micheli, G. 2012. Effect of Process Variations in 3D Global Clock Distribution Networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 8(3), 1-25.

Xu, Z., Beece, A., Zhang, D., Chen, Q., Chen, K.-N., Rose, K. & Lu, J.-Q. 2010, Nov. Crosstalk Evaluation, Suppression and Modeling in 3D Through-Strata-Via (TSV) Network. *in 3D Systems Integration Conference (3DIC), 2010 IEEE International*, pp. 1-8.

Xue, L., Liu, C. C., Kim, H.-S., Kim, S. K. & Tiwari, S. 2003. Three-Dimensional Integration: Technology, Use, and Issues for Mixed-Signal Applications. *IEEE Transactions on Electron Devices*, 50(3), 601-609.

Yahalom, G. 2016. *Analog-Digital Co-existence in 3D-IC*. (Ph.D. thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science).

Yan, C. & Greenstreet, M. R. 2007. Circuit-Level Verification of a High-Speed Toggle. *IEEE International Conference on Formal Methods in Computer-Aided Design*, pp. 199-206.

Yuan, J., Pixley, C. & Aziz, A. 2006. *Constraint-based Verification*. Springer Science & Business Media.

Zaki, M. H., Tahar, S. & Bois, G. 2006. A Practical Approach for Monitoring Analog Circuits. *Proceedings of the 16th ACM Great Lakes Symposium on VLSI (GLSVLSI'06)*, pp. 330-335.

Zaki, M. H., Tahar, S. & Bois, G. 2008. Formal Verification of Analog and Mixed Signal Designs: A Survey. *Microelectronics Journal*, 39(12), 1395-1404.

Zhao, X. & Lim, S. K. 2010. Power and Slew-Aware Clock Network Design for Through-Silicon-Via (TSV) Based 3D ICs. *Proceedings of the 2010 Asia and South Pacific Design Automation Conference*, pp. 175-180.

Zhao, X., Lewis, D. L., Lee, H.-H. S. & Lim, S. K. 2011. Low-Power Clock Tree Design For Pre-Bond Testing of 3-D Stacked ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(5), 732-745.