8-2014

EARLY PERFORMANCE PREDICTION METHODOLOGY FOR MANY-CORES ON CHIP BASED APPLICATIONS

Boray S. Deepaksubramanyan
Syracuse University

ABSTRACT

Modern high performance computing applications such as personal computing, gaming, numerical simulations require application-specific integrated circuits (ASICs) that comprises of many cores. Performance for these applications depends mainly on latency of interconnects which transfer data between cores that implement applications by distributing tasks. Time-to-market is a critical consideration while designing ASICs for these applications. Therefore, to reduce design cycle time, predicting system performance accurately at an early stage of design is essential. With process technology in nanometer era, physical phenomena such as crosstalk, reflection on the propagating signal have a direct impact on performance. Incorporating these effects provides a better performance estimate at an early stage. This work presents a methodology for better performance prediction at an early stage of design, achieved by mapping system specification to a circuit-level netlist description.

At system-level, to simplify description and for efficient simulation, SystemVerilog descriptions are employed. For modeling system performance at this abstraction, queueing theory based bounded queue models are applied. At the circuit level, behavioral Input/Output Buffer Information Specification (IBIS) models can be used for analyzing effects of these physical phenomena on on-chip signal integrity and hence performance.

For behavioral circuit-level performance simulation with IBIS models, a netlist must be described consisting of interacting cores and a communication link. Two new netlists, IBIS-ISS and IBIS-AMI-ISS are introduced for this purpose. The cores are represented by a macromodel automatically generated by a developed tool from IBIS models. The generated IBIS models are employed in the new netlists. Early performance prediction methodology maps a system specification to an instance of these netlists to provide a better performance estimate at an early stage of design. The methodology is *scalable* in nanometer process technology and can be *reused* in different designs.

EARLY PERFORMANCE PREDICTION METHODOLOGY FOR MANY-CORES ON CHIP BASED APPLICATIONS

By

Boray S. Deepaksubramanyan
B.E., University of Pune, 2003
M.S., Syracuse University, 2006

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical and Computer Engineering

Syracuse University
August 2014

Acknowledgments

To family, for family.

For me, it is "Tea Machine."

People.

TABLE OF CONTENTS

Acknowledgments **iv**

List of Tables **ix**

List of Figures **x**

1 Introduction **1**

1.1 Introduction and Motivation 1

1.2 System Modeling and Simulation Overview 4

1.2.1 System-Level Simulation 4

1.2.2 Circuit-Level Simulation 4

1.2.3 Performance Metrics of Interest 5

1.3 Outline 5

2 Why Early Performance Prediction? **8**

2.1 Introduction 8

2.2 ASIC Design Flow and Relevant Simulation Tools 9

2.2.1 Simulation Tools 10

2.3 IBIS Models for Behavioral SI Simulation 13

2.3.1 Generating an IBIS File 14

2.3.2 SPICE to IBIS Tool 15

2.4 Higher-Level System Specification 17

2.4.1 Computational Models to Describe System Behavior 18

2.4.2 Specifying Communication - Protocol Modeling 20
2.5 Summary - Need for Early Performance Prediction 24

3 A Method to Generate Models of Circuits Specified by IBIS Models 25
3.1 Introduction 25
3.2 Background on Moment Matching 26
3.2.1 Asymptotic Waveform Evaluation and Obtaining System Poles . . . 26
3.2.2 Reduced Pole Approximation 28
3.2.3 Delay Metrics Based on Distribution Function 30
3.2.4 Moment Matching Background 30
3.3 Weibull Distribution Function as a Regression Function 31
3.3.1 Weibull c.d.f. Based Regression 33
3.4 IBIS Models Based Macromodel Generation Method 35
3.5 Implementation and Results 42
3.6 Proposed Model Generation Method 43
3.7 Summary 44

4 A Tool for Behavioral Signal Integrity Analysis 46
4.1 Introduction 46
4.2 Factors Affecting Link Performance 47
4.2.1 Jitter Taxonomy 48
4.2.2 Jitter Models 49
4.2.3 Utilizing Equalizers to Minimize ISI 51
4.2.4 Crosstalk Analysis 53
4.3 Netlists for On-Chip Link Analysis 54
4.3.1 Interconnect Topologies 55
4.3.2 IBIS-ISS Netlist 57
4.3.3 IBIS-AMI-ISS Netlist with Equalizers 57

4.4 PCI Express Protocol and Netlists . . . 58
4.4.1 PCIe PHY Layer . . . 59
4.4.2 PCI Express Jitter Budget . . . 59
4.4.3 IBIS-AMI-ISS Netlist for PCIe Spec. . . . 60
4.5 Simulation-Based Performance Estimation at PHY . . . 61
4.5.1 Efficient Topology Selection . . . 62
4.5.2 Efficient Topology Selection Algorithm . . . 63
4.6 Automated Behavioral On-Chip Link Analysis . . . 64
4.6.1 Experimental Setup . . . 64
4.6.2 Netlist Generation Template . . . 65
4.6.3 Behavioral Link Analysis Method . . . 66
4.6.4 Behavioral Link Analysis Tool . . . 67
4.6.5 Implementation . . . 68
4.7 Summary . . . 71

5 Early Performance Prediction Methodology 73

5.1 Introduction . . . 73
5.2 System Modeling and Specification . . . 74
5.2.1 System Modeling . . . 74
5.2.2 Specifying Interconnects with TLMs . . . 75
5.2.3 Protocol Timing Model . . . 77
5.3 Performance Spec. and Modeling at System Level . . . 78
5.3.1 Performance Specification . . . 78
5.3.2 Performance Modeling using Queueing Models . . . 81
5.4 Early Performance Prediction Methodology . . . 82
5.4.1 System Architecture and Task Mapping . . . 83
5.4.2 Slot-Based Scheduling for Data Transactions . . . 84
5.4.3 Efficient Early Performance Prediction Method . . . 86

5.5 Implementation 87
5.5.1 System-Level Template 88
5.5.2 Circuit-Level Template 88
5.5.3 Tool Components 89
5.6 Summary 89

6 Conclusion **91**
6.1 Summary 91
6.2 Conclusions 92
6.2.1 Features of Early Performance Prediction 93

A EEPPM Tool **94**

LIST OF TABLES

3.1 Performance parameters, 22nm 43
3.2 Performance parameters, 16nm 43

4.1 PCI Express Jitter Budget [45] 60
4.2 PCI Express Jitter Budget 60
4.3 Results: 22nm 70
4.4 Results: 16nm 70

LIST OF FIGURES

1.1 Generic Many-Core ASIC Structure 2

2.1 ASIC Design Flow Overview 11

2.2 Early Performance Prediction - Mapping System Spec. to Behavioral IBIS-ISS Netlist 12

2.3 Structure of IBIS Models 13

2.4 Setup to Obtain V/I Data 15

2.5 Setup to obtain V/t Data 16

2.6 Typical transient response waveform from generated IBIS file 16

2.7 S2IBIS tool flow 17

2.8 TLM Two-Phase Protocol 22

2.9 TLM Four-Phase Protocol 22

3.1 Weibull c.d.f. for different parameter values 32

3.2 Typical IBIS model V/t Rising Waveform 35

3.3 IBIS Component Characterization 36

3.4 Inverter: V/t rising waveform 37

3.5 Weibull c.d.f. 37

3.6 IBIS Model Generation Tool 41

4.1 Circuit-level Behavioral Link Analysis with Generated IBIS Models 50

4.2 Block Diagram 3-tap FIR Filter 51

4.3 Block Diagram 3-tap FIR Filter 53

4.4 Presented Netlists for Serial Link Topologies 55
4.5 Serial Link Netlist With Equalization 58
4.6 Results for 16nm Process Technology for Various T-line Topologies 71
4.7 Results for 22nm Process Technology for Various T-line Topologies 71

5.1 Communication Protocol 77
5.2 Look Up Table: General Structure and Examples 87

A.1 Snapshot of Early performance prediction tool 95

CHAPTER 1
INTRODUCTION

1.1 Introduction and Motivation

Rapid advancement in process technology has allowed multiple cores to be integrated on integrated circuits. Current microprocessors consist of multiple on-chip cores with central processing (CPU) and graphics processing (GPU) cores integrated on the same die. As technology advances further into nanometer process technology allowing more transistors to be fabricated on-chip, many-core based processors, application specific integrated circuits (ASICs) with added functionality are becoming essential to meet performance goals while maintaining the power budget. These cores may be homogeneous (same size, type) or heterogeneous as described in [1]. Considering the current processors/systems trends, a future with many heterogeneous cores can be envisioned, each performing different functionality to implement complex applications.

These cores are interconnected through a highly efficient interconnect network. Selecting appropriate interconnect network topology considering on-chip constraints is an active research and development topic. Knowledge of many well-researched domains such as high performance computing, parallel systems, on-chip networks, ASIC system design is required for creating products that can execute multiple high intensity applications concur-

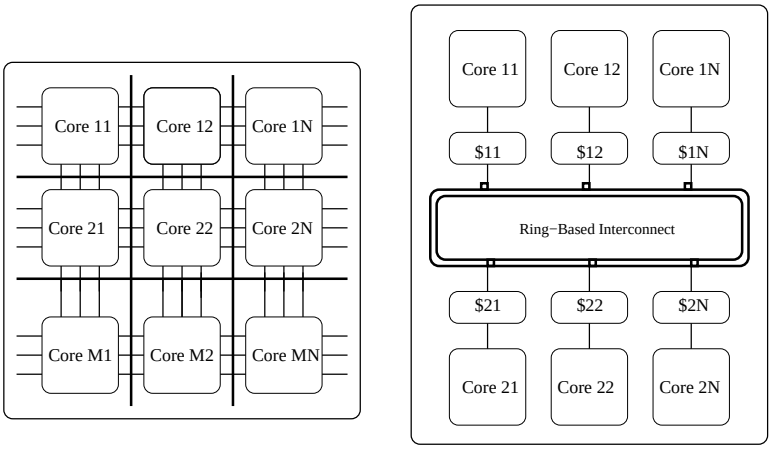

Fig. 1.1: Generic Many-Core ASIC Structure

rently for a real-world user experience. Recent examples of technology advancement include many-integrated cores (MIC) on a chip [14] that utilizes multiple cores on processors for fast computations required by concurrent applications. This many-core "co-processor" supplements the core processor in high performance computing applications.

For the applications mentioned above that have high computing needs and perform multiple functionalities requiring high amounts of data transfer, communication delay between cores is the main performance bottleneck as the application tasks is distributed to a few cores. And with process technology in nanometer era, evaluating signal integrity has become critically important to meet the performance requirement as eventually, it is the VLSI chip that has to perform the required functionality. System performance, defined primarily by latency at the system-level is therefore a function of circuit-level, layout-level issues. Therefore, physical phenomena affecting signal integrity (SI) such as reflections due to impedance mismatch, intersymbol interference and crosstalk between adjacent wires that transport data from one core to another must be accounted for to obtain a more accurate measure of system performance.

Early performance prediction methodology, the contribution of this work incorporates signal reliability information at an early stage of design. As these factors are more pronounced in nanometer process technology, and predicted to worsen with technology ad-

vancement according to the International Technology Roadmap for Semiconductors (ITRS) [2] [23], a **scalable** prediction method is essential. The developed methodology is scalable in nanometer technology.

Consider a generic many-core ASIC design as shown in Figure 1.1. Each core in the figure is tasked to perform specific functions for each application. A set of these cores would have dedicated memory and Input/Output (interconnect) units to transfer the result of tasks' computation. The performance of the chip depends mainly on the overall time taken for data transfer over interconnects. Studying the factors affecting interconnect performance is neccessary at an early stage of chip design. The design flow for processors and ASICs is a multi-phase process comprising of multiple steps. And in each phase, the system is designed with reference to the performance requirements. Computer-aided analysis or simulation of these complex systems is undertaken at each phase by selecting a suitable model of the system at that abstraction level to study various metrics required by the user.

To study ASIC performance at the system-level, a C-like higher-level specification language is employed to simplify system description and perform efficient simulation. With factors affecting signal integrity as the main issues in evaluating system performance at the circuit-level, for behavioral simulation at this abstraction level, Input/Output Buffer Information Specification (IBIS) model description is used. To accurately estimate performance at an early phase of design (i.e. at the system-level), knowledge of circuit-level issues that impact performance is required. The goal of the presented work is to provide this circuit-level process technology dependent information at an early stage of design. This is achieved by mapping the system description to its equivalent circuit-level IBIS models based netlist (the generated IBIS-ISS netlist and created "IBIS-AMI-ISS" netlist). An overview of simulation models used at system- and circuit- levels of abstraction that are utilized in this work is presented below.

1.2 System Modeling and Simulation Overview

1.2.1 System-Level Simulation

At the system-level, to efficiently model and simulate modern integrated circuits, higher-level description languages have been developed to simplify system description and to aid fast simulations. HDLs such as SystemC and SystemVerilog are good examples. A complete system, such as the one shown in Figure 1.1, that consists of sub-systems performing different functions can be specified for fast simulation and early analysis with these description languages. Several work in recent years have added SystemC, SystemVerilog capabilities to efficiently specify and simulate such systems. In this work, system is specified with C-like SystemVerilog HDL constructs [18]. To allow higher level specification and performance modeling, communication links are modeled using the transaction-level modeling (TLM) paradigm.

1.2.2 Circuit-Level Simulation

At the circuit-level, SI analysis involves studying the effects of these phenomena on signal quality at the receiver end. Three types of electrical simulation tools are used to perform SI analysis; SPICE based circuit simulators, Electromagnetic (EM) simulators and Behavioral simulators. Circuit simulators solve differential equations corresponding to various circuit elements to predict current and voltage at different nodes of the circuit. EM simulators solve Maxwell's equations and simulate electric and magnetic fields at different locations. Both simulation methods are very accurate but computationally slow [19].

Behavioral simulators, on the other hand, use precompiled tabulated data to simulate voltages and currents. Therefore, they are accurate and computationally faster than circuit and EM simulators. Input/Output Buffer Information Specification (IBIS) models are used for behavioral SI simulations. IBIS models contain non-linear electrical characteristics of the input and output pins of a component. They do not include transistor-level

implementation details. Therefore, these models preserve intellectual property (IP) and are computationally very fast. Since IP is protected, IBIS models can be re-used in different designs. It can also be used at an early stage of design process as there is no need for circuit specifics.

1.2.3 Performance Metrics of Interest

In this application domain of interest, with the focus on interconnect performance, the main system-level performance metric evaluated is latency and to an extent throughput. Latency is defined as the amount of time (or number of clock cycles) needed to transfer data from source to destination. This performance metric is impacted by physical layer factors such as crosstalk and reflection on the communication link. The study of the impact of these physical phenomena is the main focus of early performance prediction methodology. Another performance metric is throughput, though isn't the focus in this work. System throughput is defined as the rate at which data is transferred from source to destination. The importance of a particular metric is application specific. For applications considered in this work, transmission latency is the primary performance metric of focus.

1.3 Outline

For systems with many processing cores on a chip, low energy and high performance is required to sustain the dramatic development in processor and ASICs over the past four decades [1]. From an architect's perspective, a good knowledge of performance bounds/estimates at an early stage is important to make accurate decisions. For these bounds/estimates to be accurate, the architect would require a good knowledge of circuit-level issues that would impede performance. Early performance prediction method is a contribution in this direction. Evaluating system performance accounting for factors affecting signal integrity would provide a more realistic and accurate estimate of overall

performance at early phase of design cycle.

Contributions

Due to increasing design complexity, it is critical to analyze the functionality accurately at an early stage of design process. Computer-aided analysis or simulation tools are used for this purpose. ASIC and microprocessor design flow consists of multiple levels of abstractions and at each level, simulation tools have been developed for comprehensive analysis. At each abstraction layer, different groups of engineers are involved in design and verification of the system and work independently from one other. To reduce the overall design time, it is therefore beneficial to develop a method that provides a better prediction of system performance by factoring the physical phenomena affecting signal quality. This would aid in meeting time-to-market requirements. The developed early performance prediction method is a contribution in this direction. The following are the contributions of this work:

1. A tool to generate macromodel for a circuit specified by behavioral IBIS models
2. A knowledge-based method for behavioral SI analysis with generated macromodel
    - Creation of IBIS-ISS netlist based on IBIS-Interconnect subcircuit specification (IBIS-ISS).
    - Creation of IBIS-AMI-ISS netlist, that enhances the IBIS-ISS standard to incorporate IBIS Algorithmic Modeling Interface (AMI).
3. Early performance prediction method that maps a system specification to the generated netlist to obtain a better performance estimate at an early stage of design.

Dissertation Overview

The remainder of this document is organized as follows: Chapter 2 describes the need for early performance prediction methodology. First, general ASIC design flow is described

with various design phases. The developed early performance prediction method is outlined with the phases in design flow along with future trends in perspective. Simulation spec. and models at both system-level and circuit-level are discussed with focus the reliability of transferred signal. Chapter 3 presents a method to generate a macromodel of a circuit specified by IBIS models. The method, based on moment matching method is discussed. Chapter 4 presents a knowledge-based tool to perform behavioral signal integrity analysis with IBIS models. The tool incorporates the generated IBIS models discussed in previous chapter to create two new netlists, IBIS-ISS and IBIS-AMI-ISS netlist. A knowledge-based template is created to perform behavioral statistical eye analysis using the new netlists that contain surveyed transmission line models for interconnects. Peripheral Component Interface (PCI) Express (PCIe) protocol is employed for this work. Chapter 5 presents the early performance prediction methodology. A higher-level C-like specification of the system is mapped to an instance of the created netlist. Mathematical model applied for performance modeling and estimation is discussed. Finally, Chapter 6 concludes the work with a discussion on the usefulness of the developed methodology.

CHAPTER 2
WHY EARLY PERFORMANCE PREDICTION?

2.1 Introduction

This chapter presents an overview of ASIC design flow and simulation tools to provide the reader a perspective on the developed early performance prediction methodology. Multi-core and many-core integrated circuits are the computing engines in a wide range of systems/applications such as laptops, smartphones, tablets, gaming consoles, vehicles, aeronautics and so on. Each of these systems perform specific functions tailored to the required application. For example, modern microprocessors integrate CPU cores and GPU cores on the same die along with internal memory. The trend is to integrate more of these cores for highly efficient computing and are used in simulation of various high-performance applications and graphics (gaming softwares). A very recent example is the reporting of detailed simulation of the human heart, now possible with new IBM Blue Gene computing system [15]. This ASIC chip comprises of multiple cores (16 cores) performing computation along with dedicated on-chip memory, all managed by a layered interconnect fabric.

Designing and ensuring the proper operation of these ASICs is a complex task and is

handled by a series of steps. This chapter begins by an overview of the IC design flow from concept to tapeout. Each phase of design comprises of multiple intermediate steps optimized for user specified constraints such as power, timing, area etc. At each level of abstraction, comprehensive simulation is performed for validation and verification of functionality. Behavioral simulation method is of particular focus in this work and is discussed at both system spec. level and circuit level. IBIS models are widely used for performing behavioral signal integrity simulations at the circuit-level. IBIS models are discussed in detail in the subsequent sections. A good understanding of these behavioral models is required for the development of IBIS based behavioral serial link performance analysis tool presented in Chapter 3.

In the following section, background in system-level modeling with a higher-level specification language is introduced. Basic models of computation that are used for this work are discussed. A C-like description for modeling communication protocols is also described. These hardware descriptions are developed to simplify system specification and reduce simulation time. This spec. is provided as input to the early prediction method. Both system-level spec. and IBIS models background are presented with the design flow (Figure 2.1) in perspective. The goal is to motivate the need for early performance prediction method. The developed methodology itself is presented in Chapter 5.

2.2 ASIC Design Flow and Relevant Simulation Tools

ASIC design flow from specification to tapeout is discussed briefly, with a reference flow shown in Figure 2.1. System is first specified at a higher-level (behavioral-level) of abstraction in hardware oriented languages (HDLs) such as SystemVerilog, SystemC and other C-extensions to simplify complex ASIC design descriptions. These descriptions are *synthesized* to a gate-level netlist by modern synthesis tools in an automated fashion (mentioned here, but not the focus of this work). The synthesis process itself comprises of

multiple intermediate steps with optimizations for power, timing, area as per the requirements of the application (steps 1 through 4). The goal of back-end ASIC design flow (steps 4 through 8) is to map the generated netlist to standard cells for each component that is available as a library. A series of steps are performed to obtain an optimized final layout, which is converted to GDSII format and sent for fabrication. The focus here is on the simulation aspect of the flow. Depending on the spec., the required components' standard cells are invoked and interconnected to obtain final layout. In nanometer era, interconnect performance dictates overall system performance for applications discussed in Chapter 1. Therefore, many of the tools developed to automate the ASIC design flow at each phase of front-end design and back-end design specifically focus on timing and signal integrity aspects to meet performance requirement.

2.2.1 Simulation Tools

At each design phase, comprehensive simulations are performed to verify the functionality and ensure the correctness of design. System specification in higher-level languages allow fast simulations of systems as they simplify system description by abstracting relevant information. At behavioral level (system-level), simulation tools have been developed for (early) performance analysis. Similarly, mature gate-level and circuit-level simulation tools are available that analyze synthesized designs and ensure correct working of design. Examples include Cadence Spectre, Synopsys HSPICE and others. For fast behavioral signal integrity simulations with IBIS models at the circuit-level (or layout level), simulation tools exists such as Mentor Graphics Hyperlynx, Cadence Sigrity and so on. These simulations are performed at a later phase of design cycle (Step 6). Since custom ASIC design involves the use of standard cells, IBIS models for each of the cells would also be available.

From a system designer or architect's perspective, system-level simulations provide valuable understanding of the system to be designed. However, the accuracy of the models is critical for a good estimate of the system performance. A useful contribution would be

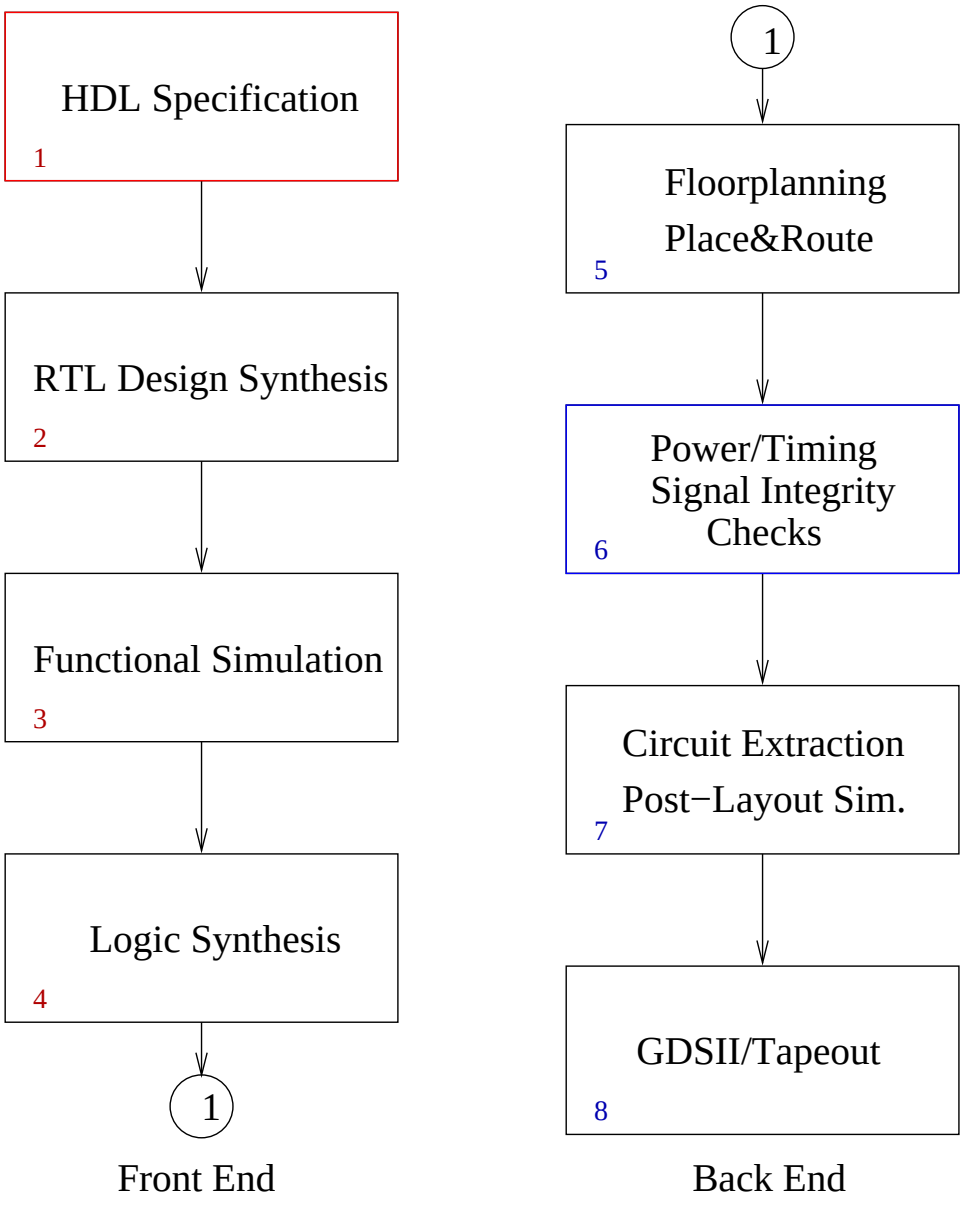

Fig. 2.1: ASIC Design Flow Overview

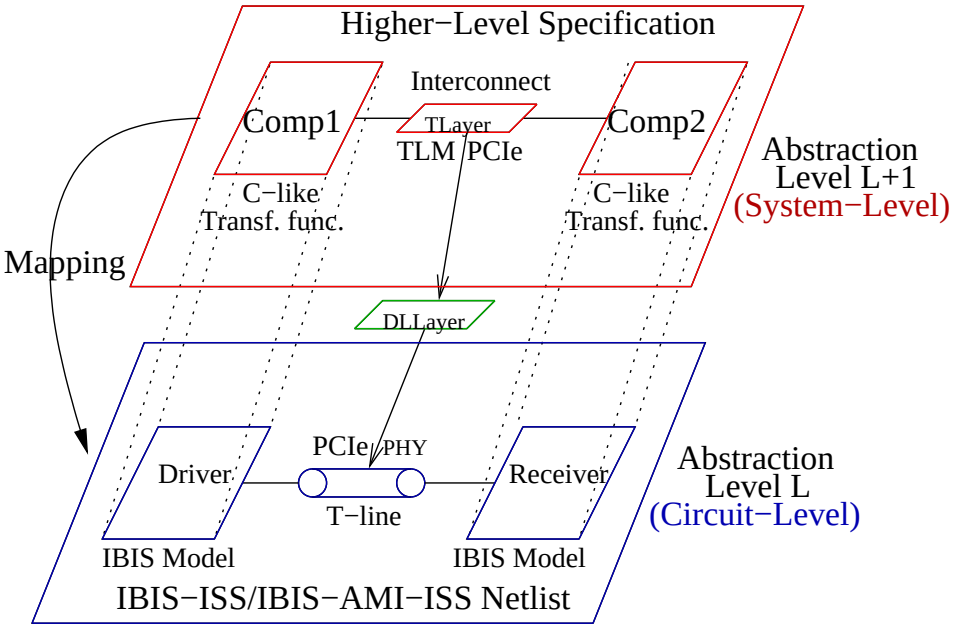

Fig. 2.2: Early Performance Prediction - Mapping System Spec. to Behavioral IBIS-ISS Netlist

to provide the architect with simulation information from later phase of design (at lower abstraction level) during system-level simulation phase. This would necessitate the need for process tech. dependent mapping at an early stage of design to improve the prediction accuracy and lead to first-time-correct implementations. The presented work is a step in this direction.

To appreciate early performance prediction methodology, a good understanding of specification and modeling at different abstraction levels is needed. The reader would recognize that the specification and models are vary significantly as the abstraction levels discussed in this work represent different domains in electrical engineering and computer engineering. The following section describes these models (specifications) in detail. Figure 2.2 shows the mapping of models at different abstraction level that forms the crux of *early prediction methodology*.

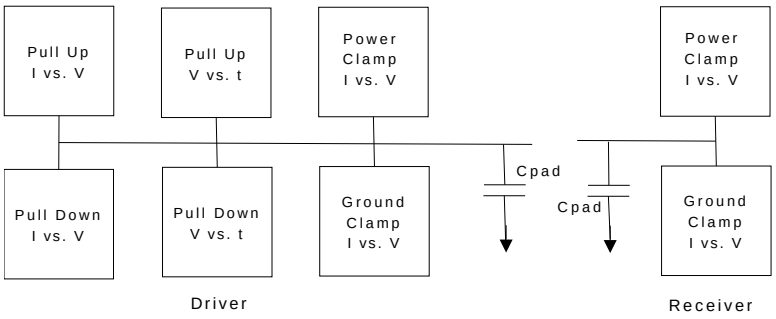

Fig. 2.3: Structure of IBIS Models

2.3 IBIS Models for Behavioral SI Simulation

This section describes the Input/Output Buffer Information Specification (IBIS) models in detail. IBIS is a behavioral specification (ANSI/EIA standard) that describes the non-linear electrical characteristics of the input and output pins of a component through tabulated Voltage-Current (V/I) and Voltage-time (V/t) data. IBIS models were introduced by Intel Corporation in early 1990s to aid signal integrity simulations by allowing interoperability of models between vendors. Unlike SPICE, IBIS models do not include transistor-level implementation details and thus preserves intellectual property (IP). This allows for faster simulation than SPICE. IBIS models can be created either manually by extracting V/I and V/t data or using automation tool such as S2IBIS. Both manual procedure and the automation tool to create IBIS models are discussed.

Figure 2.3 shows the structure of the input pin (receiver) and output pin (driver) of an IBIS file. The driver consists of a pull-up device (with V/I, V/t data) and a pull-down device (with V/I, V/t data). Behavioral simulation with IBIS models involves analyzing effects of impedance mismatch, crosstalk and reflections on an electrical signal propagating on a wire modeled as a transmission line from the driver to receiver. IBIS models are used mainly in signal integrity simulation of printed circuit boards.

In this work, IBIS models are used for on-chip signal integrity simulations. To be able to use the IBIS models to perform processor performance simulations in an automated manner, a macromodel needs to be generated from the IBIS model with estimates of per-

formance parameters; for example the signal rise time.

Using the V/t data, each component is characterized by the poles (transfer function) of the specific component at a particular technology node and its edge rate (rise and fall times). Manual procedure to extract V/I and V/t data is discussed next. This is followed by a brief discussion on the automated tool. The generated IBIS files form the input to the proposed model generation tool described in Chapter 3.

2.3.1 Generating an IBIS File

The procedure for generating data to create IBIS model file is described here. Steps for extracting voltage-current and voltage-time data of a component is discussed. The ANSI/EIA IBIS Specification [7] outlines the procedure to obtain V/I and V/t data [10].

Extracting V/I Data

To construct the V/I curves for the output pin, the first step is to understand the working of the component and determine how to put the component's output to logic low and logic high states. For this, a transistor-level netlist of the component is provided. Once the component is placed at a particular output state, the output pin is connected to an independent voltage source, as shown in Figure 2.4. The simulator is set to DC analysis mode. The current flow into the pad is then measured as the source is swept across a range from -VDD to 2VDD. Separate curves are generated for pull-up and pull-down devices by setting the output of the component to logic low and high respectively [10]. The steps are repeated for each component.

Extracting V/t Data

Next, the procedure to generate tabulated transient response (voltage-time data) is described. To obtain V/t (rise time and fall time) curves for each output pin, the simulator is set to transient analysis mode [10]. A signal with appropriate slew rate is applied at the

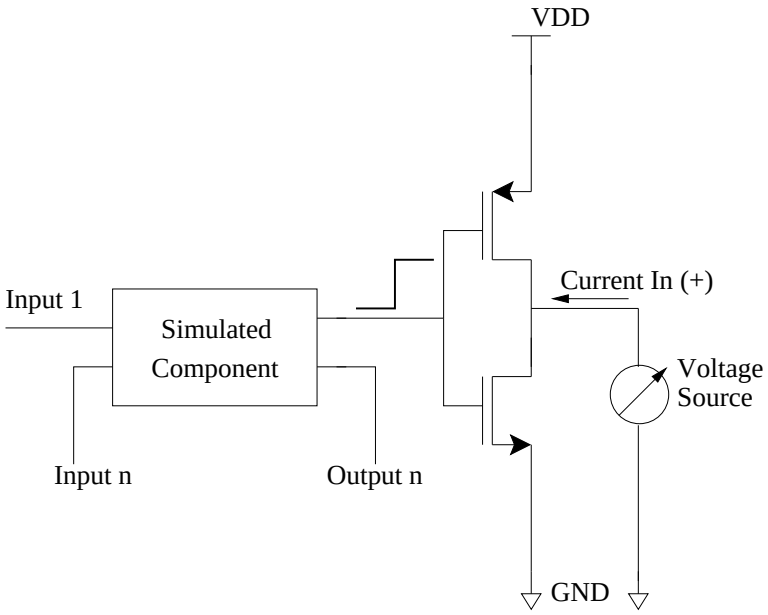

Fig. 2.4: Setup to Obtain V/I Data

input of the simulated component as shown in Figure 2.5. V/t tables are created when the output is driving into a test load, typically a 50Ω resistor that is connected to the appropriate supply [10], as described in Figure 2.5. The test load can be selected or specified by the user. The test load used to generate the V/t tables must be described in the model. If the automation tool, discussed next, is used to generate IBIS data, then the test load must be specified in the user defined *.s2i* file. The signal integrity simulation tool will use this information from the model to reconcile the output waveform to reflect the behavior when driving the actual system load. Figure 2.6 shows a typical rising V/t waveform obtained from an IBIS file.

2.3.2 SPICE to IBIS Tool

Spice to IBIS conversion tool (S2IBIS) is a platform independent tool that converts a SPICE netlist of a component (circuit) to its IBIS model. The tool was developed at North Carolina State University [9]. Figure 2.7 shows the block diagram of S2IBIS tool flow [12]. The tool consists of a command file that contains the header information and user defined description

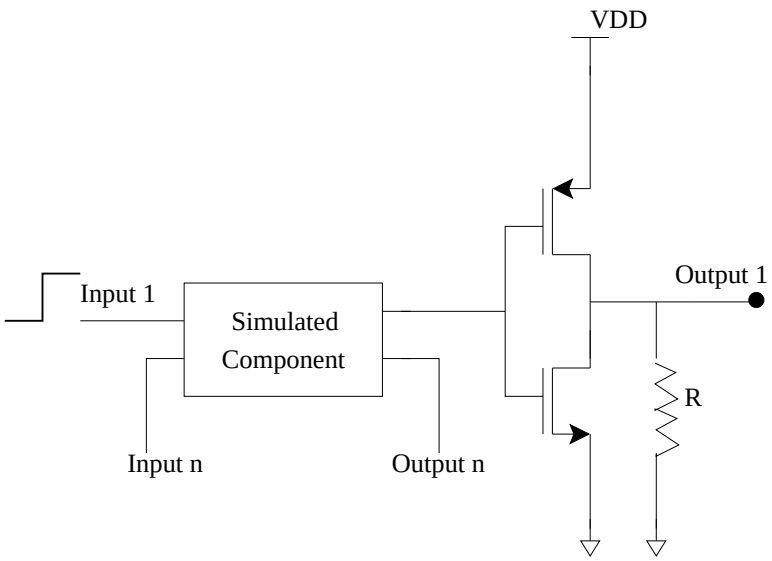

Fig. 2.5: Setup to obtain V/t Data

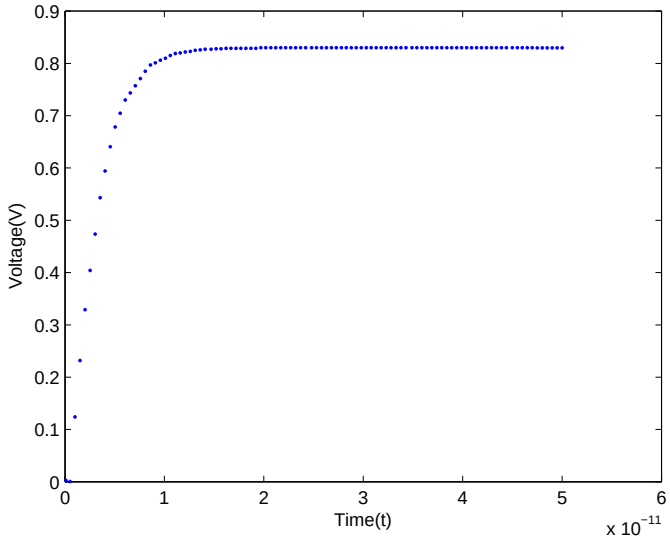

Fig. 2.6: Typical transient response waveform from generated IBIS file

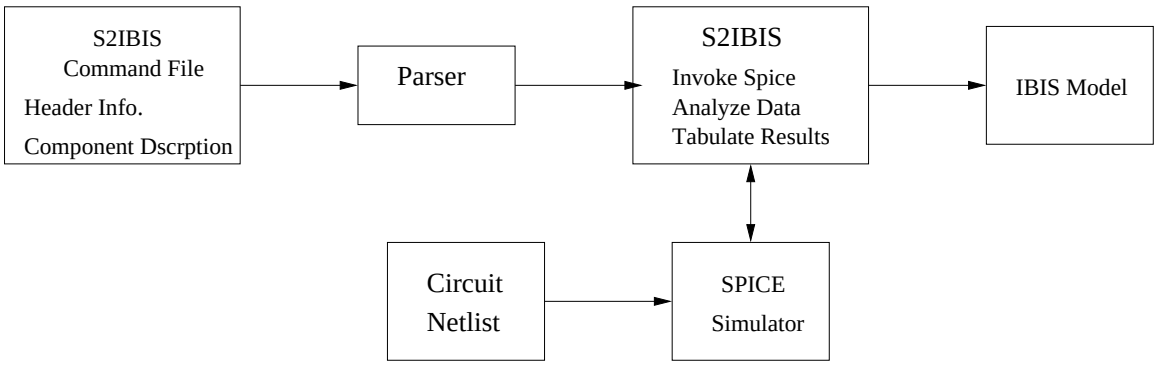

Fig. 2.7: S2IBIS tool flow

of the component. This command file (Example: example.s2i) has to be edited by the user and details of the Spice netlist file, the (technology) model file, supply voltage and other parameters are incorporated. The S2IBIS tool accepts this information through a parser and invokes a SPICE simulator to analyze data and extract relevant V/I and V/t data. Acceptable SPICE simulators include HSPICE, SPICE2, SPICE3. Finally, the data is stored into a file in the standard IBIS model format. For this work, IBIS-AMI (IBIS versions 5 and 6) models are created manually with specific information required for this academic work as S2IBIS generates models upto IBIS version 3. IBIS version 5 onwards support Algorithmic Modeling Interface (AMI).

With the understanding of data generation method for IBIS models, a method to generate macromodels of circuits specified by IBIS models is presented in Chapter 3. In the following section, system specification background is reviewed with focus of the presented work.

2.4 Higher-Level System Specification

As complexity of integrated circuits increases, there arises a need to describe these systems at a higher level of abstraction to simplify system description and improve simulation time. For specifying systems at this higher abstraction level (above RTL), called system-level, new hardware description languages [16] [62] have been developed. The most prominent

ones are SystemVerilog and SystemC. System-level languages provide constructs/facilities of computation and communication of the system to be modeled separately. SystemVerilog HDL [18] (IEEE 1800 Standard) builds on the existing, widely used Verilog HDL by adding new constructs that allows simplification of complex system descriptions. These include separate description of processing elements (cores) and the intercommunication between them, allowing abstract description of interfaces. Similarly, SystemC [16] (IEEE 1666 standard) on the other hand has been built ground-up with C++ programming language.

SystemVerilog, SystemC allow users to specify systems at a higher level of abstraction (above RTL, Verilog/VHDL). At the higher-level, system functionality is effectively described with minimal information of lower-level details. This allows for faster and more efficient simulation of modern systems. SystemVerilog comprises of extensions over Verilog HDL with specific constructs for interconnect description (`interface`), allowing the specification at system-level, where separation of computation and communication is required. SystemC on the other hand, has been built ground up. It is basically a set of classes built over the standard C++ programming language with the addition of timing and concurrency constructs. This allows hierarchical description of hardware components and ease of extensibility. During the course of this work, both have been studied, but SystemVerilog constructs have been used to develop early performance prediction methodology.

2.4.1 Computational Models to Describe System Behavior

Models of computation (MOC) relevant for modeling many-core ASICs have been defined and implemented over the past years. These MOCs have been implemented in SystemC and are available as part of standard libraries [54] [59]. Of the MOCs implemented, bounded process network is of particular interest along with timed synchronous dataflow. These MOCs can be specified in a similar way using SystemVerilog as well. This work started with SystemC, however SystemVerilog has been used for system specification.

Bounded Kahn Process Network (BKPN)

BKPN MOC consists of computational process nodes connected by unidirectional FIFO channel. BKPN implementation in SystemC is available as part of heterogeneous computation model library [54]. Both unbounded and bounded KPN are implemented with bounded FIFO channel with buffer size N, where $1 < N < \infty$. The behavior of BKPN is given by [54]:

Rules for communication:

1. Communication via bounded FIFO's ($1 < N < \infty$). Size of the buffer is fixed.
2. Blocking read and write implementation. Read until fifo empty, write until buffer is full
3. No data can be written when buffer is full. No read access when buffer is empty

Rules for computation: Computation is through continuous mapping. That means, a token is produced only after consuming some tokens to perform a computation

Synchronous Dataflow

Timed synchronous dataflow (TSDF) MOC is implemented for example in the SystemC-AMS standard [59]. It is based on the synchronous dataflow model of computation with sampling at discrete time stamps. TSDF model is well suited for work focussing on signal processing applications. The rules that describe the behaviour of a TSDF MOC are given below:

Rules for communication:

1. Tokens transferred at a rate specified by the user.
2. Communication through the HetSC uc_arc_seq FIFO channel implementation.

Rules for computation:

1. Computation is through continuous mapping.

2. Functional relation.

3. Input arc is read first to produce the output arc.

2.4.2 Specifying Communication - Protocol Modeling

To simulate detailed communication protocols such ARM AMBA, Intel PCI Express at a higher level of abstraction, transaction-level modeling (TLM) method has been developed over the past decade. TLM provides mechanisms for modeling behavior of the communication link with predefined and user defined tasks and functions. The most recent version of standardized TLM, TLM 2.0 has been incorporated into SystemC [58] specification language. TLM is a concept that isn't specific to a particular description language and can be specified in other system-level languages such as SystemVerilog. The modeling of interconnects with TLMs using SystemVerilog is discussed in subsequent chapters.

Transaction Level Modeling

Transaction level modeling (TLM) [62] specifies modeling communication between processes as a sequence of transactions. The previous section highlighted computation models for higher-level component specification. TLM allows behavioral description and modeling of interconnection between processes. It provides means to abstract communication interface between the components using tasks and/or functions. By raising the abstraction level, significant simulation speedup is achieved while studying the system behavior. TLMs can be purely at the software level, or at the hardware level by abstracting the notion of time. They can be described in any system-level description language.

TLM 2.0, defined and adopted by Open SystemC Initiative (OSCI) defines two coding styles to specify protocols at different modeling abstractions; loosely-timed (LT) TLM and approximately-timed TLM. In loosely-timed model an abstract notion of timing (if any)

is provided. Approximately-Timed model contains more detailed timing information as compared to LT model. The level of timing detail could be an integer number or could be clock cycle count accurate, but not clock cycle-accurate. Loosely-timed (LT) modeling approach is suitable for early software-level analysis. Approximately-timed (AT) models on the other hand are very useful for hardware performance modeling and analysis at an early phase of the design process. Therefore, approximately-timed modeling approach has been referenced to model the communication between components in this work. The protocol model itself is briefly discussed below in general. In this discussion, a point-to-point producer-consumer transaction model is considered with producer referenced as the initiator and consumer as the target.

Two-Phase and Four-Phase AT TLM Approach

Modeling interconnect protocol for performance prediction requires a level of timing detail. OSCI TLM2 describes two base protocol specifications with different timing phases. They are categorized as two-phase AT modeling and four-phase AT modeling approaches. Two phase AT implementation consists of two phases (timing points) to model the protocol. For this work, a combination of two-phase and four-phase AT protocol has been used to model the communication protocol at the transaction level, with PCIexpress protocol specification as a reference. Both two-phase and four-phase AT TLM models are used for this work. All timing phases are non-blocking calls. The two timing phases of 2-phase AT protocol are:

1. (Begin) Request - Initiator to target
2. (End) Response - Target to initiator

The four timing phases of the 4-phase AT protocol are:

1. Begin Request (Beg-Req)- Start of transaction from initiator to target
2. End Request (End-Req) - Acknowledgement from the target to initiator

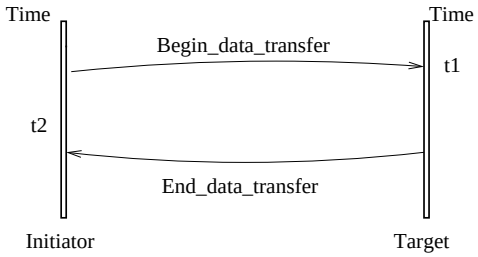

Fig. 2.8: TLM Two-Phase Protocol

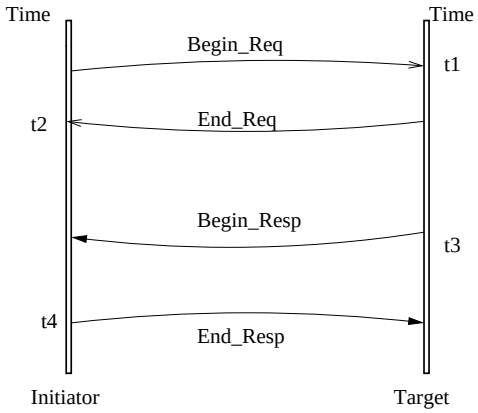

Fig. 2.9: TLM Four-Phase Protocol

3. Begin Response (Beg-Rsp) - Begin the transaction from target to initiator

4. End Response (End-Rsp) - End of transaction from initiator to target

The timing information at each phase of the protocol can be specified explicitly by the **wait()** statement or implicitly by noting the time taken by each step in the protocol. Figure 2.9 shows a typical implementation of the protocol. Using the AT-TLM coding style and the two-phase, four-phase base protocol model, communication interfaces can be modeled at different timing abstraction levels.

Specifically, to model the interaction between cores in a many-core based systems, functional and architects TLM abstraction views are useful. The architects view transaction level model is well suited for architectural exploration, performance modeling and estimation at early stage of design. Timing information is specified as an integer at this abstract level and can be refined to clock cycle count accurate. Therefore, this view is referenced while specifying the interconnect functionality in this work. A typical system

description would consist of the instruction followed by its delay specification for performance analysis. A set of function calls are described for modeling the communication aspect. Open Cores Protocol - IP (OCP-IP), provides detailed description of various TLM views that includes functional view, programmers view and verification view for different research and development group. Chapter 5 provides code snippets used in this work as examples of functional and architectural view specification and modeling style.

Communication/Interface Channels

During the initial phase of this work, SystemC based modeling and specification was studied. Although not used directly, the following paragraph summarizes some advances that are applicable with other HDLs. To broaden the usage of system-level HDLs (SystemC in this work) for efficient simulation, *converter channels* as means for connecting dissimilar models of computation have been developed in [51]. These adaptable channels are required to connect heterogeneous models of computation due to their dissimilar behavior. Of particular interest for this work is the connection between a BKPN MOC and TSDF MOC. BKPN model has an abstract notion of time. Conversion to TSDF is essentially a process of adding more detailed timing information. Considering the behavior of TSDF, the timing information is essentially discretized. Similarly, interfaces for other computation models are developed.

This work utilizes SystemVerilog based TLM as the language builds upon widely used Verilog HDL and newer constructs defined in the IEEE 1800 standard are particularly useful for research and development in this work. Authors in [105] provide a set of SystemVerilog snippets that model asynchronous communication between two systems. In similar work, a handshaking protocol based interface spec. is provided that can be used in async comm. These facilities can be used as an input specification for the developed early prediction method.

2.5 Summary - Need for Early Performance Prediction

The goal of this chapter was to provide the reader a perspective of the need for early performance prediction, the contribution of this work. In this context, the VLSI design flow overview was first presented highlighting the main steps from HDL specification to final tapeout. Each phase of the design flow in itself comprises of multiple steps that perform specific functions to meet the requirements. During each design phase, simulation plays an indispensable role providing valuable feedback to the tool (and user) and several tools exist that perform accurate simulation (Examples: SPICE, ModelSim). This work focusses on mapping system description used at two different abstraction levels for efficient simulation to obtain a better estimate of performance.

Specifically, a methodology for early performance prediction that aids an architect or system designer to better predict system performance at an early stage of design is developed. The method maps a SystemVerilog based system spec. (used for efficient simulation) to the generated IBIS models based netlist. This work is useful because it aims to reduce the design cycle time by providing a more accurate estimate at an early design phase, aiding faster time-to-market requirement. Aspects related to early performance prediction are discussed in subsequent chapters. The next chapter describes the model generation tool development. Given an IBIS file of a circuit, the tool estimates the system performance parameters and generates the macromodel.

CHAPTER 3
A METHOD TO GENERATE MODELS OF CIRCUITS SPECIFIED BY IBIS MODELS

3.1 Introduction

At the circuit-level, behavioral signal integrity simulations are performed with IBIS models as discussed in Chapter 1. A serial link has to be represented by a netlist to employ signal integrity simulation tools to evaluate performance. For the serial link, rise time and fall time information of the signal is required. This infomation needs to be extracted from IBIS model file. This chapter describes methods for generating models of a circuit specified by IBIS model file. The developed model generation method generates a macromodel that comprises of the estimates of poles, rise time and fall time of the specified circuit, described in detail in this section. For the model generation tool to estimate the aforementioned parameters, the data must first be fitted to a mathematical model. Then the model parameters can be determined. Using circuit and systems theory, the model parameters are used to obtain the macromodel. In this work, Weibull cumulative distribution function (WCDF) has been selected as the regression function. With this regression function, regression analysis is performed on the IBIS V/t data. Moment matching method is then employed to obtain

the system poles.

The remainder of the chapter is organized as follows: Section 3.2 discusses the background on moment matching method applied for estimating system poles. Section 3.3 describes the selected mathematical model, the Weibull distribution function and previous work on development of Weibull delay metric based on the distribution function. This would motivate the reason for using Weibull distribution function for this work. Section 3.4 presents a method to generate a macromodel of a circuit specified by IBIS models. The method is summarized as a pseudocode for reference. Section 3.5 provides the results and implementation details. This is followed by another method for macromodel generation. Due to "human" constraints, this hasn't been implemented, however the method itself is well documented. Section 3.6 summarizes the developed method.

3.2 Background on Moment Matching

3.2.1 Asymptotic Waveform Evaluation and Obtaining System Poles

Asymptotic waveform evaluation is a general technique for simulation of large linear(ized) circuits [28]. In AWE, a form of Pade's approximation is employed to approximate a system transfer function to a reduced order model. Once the AWE model is developed, it can be reused for different applications. Consider a lumped linear time-invariant (LTI) system. This system can be described by the following state equation:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{3.1}$$

Here, **u** is a generic input excitation vector and **x** is an n-dimensional state vector. The solution for this is the following homogeneous equation [28]:

$$\dot{\mathbf{x_h}} = \mathbf{A}.\mathbf{x_h} \tag{3.2}$$

The Laplace transform for the above equation is:

$$\mathbf{X_h(s)} = (\mathbf{sI} - \mathbf{A})^{-\mathbf{1}}\mathbf{x_h(0)} \tag{3.3}$$

The equation above can be approximated by expanding it as a Taylor series about s = 0, shown below.

$$\mathbf{X_h(s)} = -\mathbf{A}(\mathbf{I} + \mathbf{A^{-1}s} + \mathbf{A^{-2}s^2} + ...)\mathbf{x_h(0)} \tag{3.4}$$

Now, the transfer function of this system in terms of circuit moments (m_{ci}) can be expressed as follows:

$$H(s) = m_{c0} + m_{c1}s + m_{c2}s^2 + m_{c3}s^3 + ... \tag{3.5}$$

Here, the generalized circuit moments are defined as:

$$m_{ci} = \frac{(-1)^i}{i!}\int_0^\infty t^i.e^{-st}dt, \quad i = 0, 1, 2... \tag{3.6}$$

The circuit moments can be obtained by comparing Equations 3.4 and 3.5 as it follows from the definition of Laplace transform:

$$X(s) = \int_0^\infty x(t).e^{-st}dt = \sum_{k=0}^{\infty}\frac{(-1)^k}{k!}s^k\int_0^\infty t^k.e^{-st}dt \tag{3.7}$$

These time-domain circuit moments are now matched to a lower order frequency domain function in poles and residue form (Pade's approximation):

$$H(s) = \sum_{l=0}^{q}\frac{k_l}{s - p_l} \tag{3.8}$$

As many circuit moments from the equation above can be matched to a lower order model.

Note:

AWE method for timing analysis uses Pade's approximation to evaluate the lower order model. The idea of Pade's approximation is to approximate a transfer function $H(s)$ by an order-limited rational function $H_q(s)$, where q is the order.

$$H_q(s) = \frac{1 + b_1 s + b_2 s^2 + ... + b_n s^n}{1 + a_1 s + a_2 s^2 + ... + a_n s^m} \tag{3.9}$$

where, $n = m - 1$.

To obtain a q^{th} order reduced model of an n^{th} order system, $2q$ circuit moments are required. Pade's approximation then converts these $2q$ moments into a q^{th} order reduced model. Poles and residues can be calculated using the steps outlined in [28].

3.2.2 Reduced Pole Approximation

In [29], the authors present a *two-pole approximation* that is derived from the first three moments of the impulse response of the circuit/system. The system transfer function is approximated to the first three moments of the impulse response. Based on these circuit moments, two system poles are obtained by a set of equations. The first two poles are approximated/calculated to the first three moments using the following equations [29]:

$$p_1 = -\left(\frac{m_{c2}}{m_{c3}}\right) \tag{3.10}$$

$$p_2 = p_1 . \left(\frac{m_{c0}.m_{c2} - {m_{c1}}^2}{m_{c1}.m_{c3} - {m_{c2}}^2}\right) \tag{3.11}$$

where, m_{ci}, $i = 0, 1, 2, ...$ are the circuit moments.

The approximate residues can be obtained from the two poles and three circuit moments [29]. The approximate poles above are derived with the aid of AWE, discussed earlier in this chapter. The transfer function for a given system can be approximated by a reduced

order model based on Pade's approximation, the denominator can be further approximated by considering coefficients of 's' upto the second order. In [28], it was shown that the ratio of successive moments asymptotically approaches to the value of the dominant pole, i.e.,

$$p_1 = \lim_{j \to \infty} \left(\frac{m_{cj}}{m_{c(j+1)}} \right) \tag{3.12}$$

To obtain two poles, the first three moments of the circuit are required. They can be calculated by the following set of equations: The ratio of two successive poles in terms of circuit poles, $\frac{p_j}{p_{j+1}}$ can be expressed as below [29]:

$$p_2 = p_1 . \left(\frac{1 + \frac{k_2}{k_1} . (\frac{p_1}{p_2})^{j+1} + \frac{k_3}{k_1} . (\frac{p_1}{p_3})^{j+1} + ... + \frac{k_n}{k_1} . (\frac{p_1}{p_n})^{j+1}}{1 + \frac{k_2}{k_1} . (\frac{p_1}{p_2})^{j+2} + \frac{k_3}{k_1} . (\frac{p_1}{p_3})^{j+2} + ... + \frac{k_n}{k_1} . (\frac{p_1}{p_n})^{j+2}} \right) \tag{3.13}$$

Assuming that the given circuit has been approximated to 'N' poles, the j^{th} moment is related to poles and residues by the following expression:

$$m_{cj} = \Sigma_{i=1}^{N} \frac{k_i}{(p_i)^{j+1}} \tag{3.14}$$

The detailed derivation of the equations above are given in [29]. The same equations can be used to obtain additional poles of the system from higher order moments. The presented pole approximation method has been applied to estimate the poles of the system in the developed model generation method to be described in the next section. In [36], the authors present a method to obtain delay and slew metrics based on the first three circuit moments. The application is for interconnects modeled as RC trees and more accurately estimating the delay values for near end and far end nodes. In the following section, a survey of delay metrics proposed based on various probability distributions is discussed in some detail.

3.2.3 Delay Metrics Based on Distribution Function

For circuits modeled as RC model, delay metrics have been proposed that define and calculate delay metrics based on probability distribution functions. Delay metrics based on the Gamma, Weibull, Lognormal and Beta distribution functions have been reported in earlier work. Weibull distribution function based delay metric is of particular interest as it has been applied in this work.

3.2.4 Moment Matching Background

Moment matching method involves matching the circuit moments to the central moments of a selected distribution function. Several works explore this method in various domains. Here, this method along with regression, estimates the poles of a circuit specified by tabulated simulated data value, a contribution [38].

Recall that the transfer function of an LTI system in terms of circuit moments (m_{ci}) is given by:

$$H(s) = m_{c0} + m_{c1}s + m_{c2}s^2 + m_{c3}s^3 + ... \tag{3.15}$$

Circuit moments are defined as:

$$m_{ci} = \frac{(-1)^i}{i!}.\int_0^\infty t^i.e^{-st}dt, \quad i = 0, 1, 2... \tag{3.16}$$

The central moments for a particular distribution function or p.d.f. are given in [32]. The goal here is to match these to the circuit moments given by Equation 3.16. From the above equations, matching the circuit moments and moments of a distribution, we obtain the following relations:

$$m_{c1} = -m_1 \tag{3.17}$$

$$m_{c2} = 2!.m_2 - {m_1}^2 \tag{3.18}$$

$$m_{c3} = -3!.m_3 + 3m_1m_3 - 2{m_2}^2 \tag{3.19}$$

Based on this, the relation between circuit moments (m_{ci}) and central moments (μ_i) can be obtained. Weibull distribution function is specifically applied in this work. To obtain the macromodel, WCDF is used as the regression function. This is discussed in the following section.

3.3 Weibull Distribution Function as a Regression Function

Weibull distribution function is often used in reliability engineering. There are two models of Weibull distribution function; two parameter and three parameter. For the presented work, two-parameter Weibull distribution function has been used and will therefore be the focus of the following paragraphs. A two-parameter Weibull probability density function (WPDF) with parameters 'α' and 'β' is defined as:

$$P(t) = \alpha.\beta^{\alpha}.t^{\alpha-1}.e^{-(t/\beta)^{\alpha}} \tag{3.20}$$

Where, t > 0 and α, β > 0.

Parameter 'α' is known as shape parameter.

Parameter 'β' is known as scale parameter.

The Weibull cumulative distribution function (WCDF) is defined as:

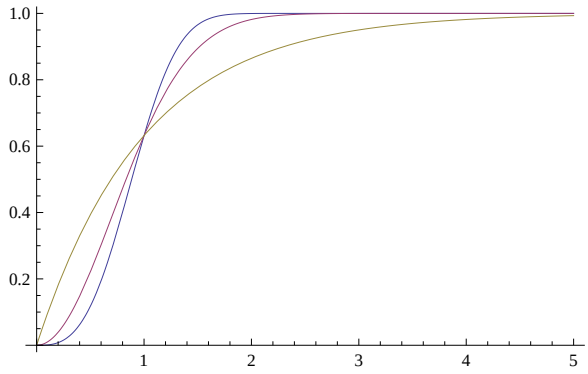

Fig. 3.1: Weibull c.d.f. for different parameter values

$$F(t) = 1 - e^{-(\frac{t}{\beta})^{\alpha}} \tag{3.21}$$

For α < 1, α = 1 and α > 1, three basic shapes of WCDF can be obtained as shown in Figure 3.5.

It can be seen from Figure 3.5 that the WCDF is able to model various waveform shapes that a circuit modeled as RC network can take for a step (or ramp) input. Important properties of the distribution function are well catalogued and given below. The *mean*, *variance* and *skewness* of Weibull distribution function is given by:

$$\mu = \beta.\Gamma(1+\theta) \tag{3.22}$$

$$\sigma^2 = \beta^2.(\Gamma(1+2\theta) - \Gamma^2(1+\theta))$$

$$\sigma = \sqrt{\beta^2.(\Gamma(1+2\theta) - \Gamma^2(1+\theta))} \tag{3.23}$$

$$\gamma = \frac{(2.\Gamma(1+1\theta)^3 - 3\Gamma(1+1\theta)\Gamma(1+2\theta) + \Gamma(1+3\theta))}{(-\Gamma(1+1\theta)^2 + \Gamma(1+2\theta))^{3/2}} \tag{3.24}$$

where, $\theta = 1/\alpha$

Moment matching technique involves matching the moments of a selected distribution to the moments of the underlying circuit. Since, the simulation data is provided, first regression is required before applying moment matching. The details of this methodology is

discussed in the next section.

3.3.1 Weibull c.d.f. Based Regression

The mathematical background required for model generation methodology is discussed in this section. Regression analysis is first performed with IBIS data to regression coefficients. Using the estimated value of these coefficients, moment matching method is applied to obtain the estimated poles, rise time and fall time of the specified circuit. The steps for the presented methodology is described in detail below. Weibull distribution function is reviewed as it is the selected regression function in this work. Then, regression analysis with Weibull distribution function is described.

Regression Function: Two parameter Weibull distribution function was introduced in the previous chapter. Here, the key properties are provided for reference. As was noted earlier, Weibull based delay metric is computationally fast and relatively accurate. The p.d.f., c.d.f., mean and variance of a Weibull distribution with parameters 'α' and 'β' are given by:

$$P(t) = \alpha.\beta^{\alpha}.t^{\alpha-1}.e^{-(\frac{t}{\beta})^{\alpha}} \tag{3.25}$$

$$F(t) = 1 - e^{-(\frac{t}{\beta})^{\alpha}} \tag{3.26}$$

$$\mu = \beta.\Gamma(1+\theta) \tag{3.27}$$

$$\sigma^2 = \beta^2.(\Gamma(1+2\theta) - \Gamma^2(1+\theta)) \tag{3.28}$$

Where, $t > 0$ and α, β > 0, and $\theta = 1/\alpha$.

The Gamma function [110] is given by: $\Gamma(t) = \int_0^{\infty} y^{x-1}.e^{-y}dy$. Regression analysis is performed on IBIS model file by fitting the Weibull CDF to the transient analysis data. The rise and fall times are obtained with the estimated values of 'α' and 'β'. Then, applying moment matching technique, system poles are determined (estimated). Depending on the application, these parameters can be utilized to obtain other parameters. Detailed

discussion follows.

Regression Analysis: Ordinary least square regression method has been applied for this work with Weibull c.d.f. as the regression function due to its simple functional form and ability to fit various waveform shapes. For this function, least square regression analysis is performed as follows. Consider the WCDF, reproduced below:

$$F(t) = 1 - e^{-(\frac{t}{\beta})^{\alpha}} \tag{3.29}$$

If we take the natural logarithm on both sides of the equation above, we get:

$$\ln F(t) = \ln[1 - e^{-(t/\beta)^{\alpha}}]$$
$$\ln F(t) = \ln(1) - \ln(e^{-(t/\beta)^{\alpha}})$$
$$\ln F(t) = (t/\beta)^{\alpha}$$

Taking natural logarithm again,

$$\ln .\ln F(t) = \ln (t/\beta)^{\alpha}$$
$$\ln .\ln F(t) = \alpha . \ln(t - \beta)$$
$$\ln .(\ln F(t)) = \alpha . \ln(t) - \alpha . \ln(\beta) \tag{3.30}$$

Equation 3.30 above is now of the form of a linear regression line, y = $\beta_0 . x$ + β_1, where y = $\ln .(\ln F(t))$, β_0 = α and β_1 = $-\alpha . \ln(\beta)$. Least square regression can be performed on Equation 3.30 following the steps outlined earlier. The estimated values of 'α' and 'β', '$\hat{\alpha}$' and '$\hat{\beta}$' respectively, are then used for obtaining the system poles, rise time and fall time of the circuit specified by IBIS model generated by the model generator. The model generation method is discussed next.

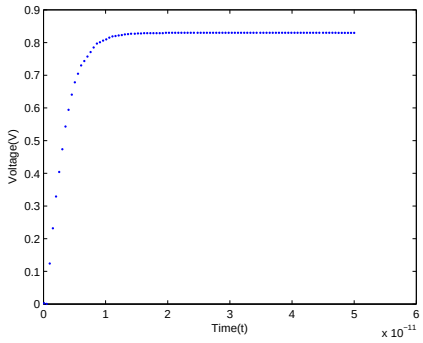

Fig. 3.2: Typical IBIS model V/t Rising Waveform

3.4 IBIS Models Based Macromodel Generation Method

This section describes the implemented model generation tool. The model generator accepts IBIS voltage-time data as input and generates the estimated values of rise time, fall time and poles of the system. The methodology consists of two steps: macromodeling and parameter estimation. In the next chapter, this work is applied to aid early performance prediction.

Macromodeling: A macromodel is an approximation of an unknown function [24] [25]. Constructing a macromodel consists of the following steps: *data generation* and *regression*. IBIS model data generation and formatting steps were described in detail in Chapter 2. Regression is performed on the tabulated data in IBIS models. Regression analysis involves computing the coefficients of the selected functional form. In this work, Weibull cumulative distribution function has been used as the regression function. This is because it can fit various waveform shapes of the Voltage-time (V/t) curves of an IBIS model obtained with transient analysis of circuits as shown in Figure 3.2. The regression method applied here is discussed next.

Weibull c.d.f. Based Macromodel: The Weibull cumulative distribution function (WCDF), with parameters 'α' and 'β' is given as:

$$F(t) = 1 - e^{-(\frac{t}{\beta})^{\alpha}} \tag{3.31}$$

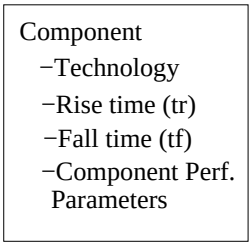

Fig. 3.3: IBIS Component Characterization

The key statistical characteristics, mean (μ), variance(σ^2) and skewness (γ) of the distribution function are given in terms of the Gamma function ($\Gamma(t)$) in Equation 3.27 and Equation 3.28.

Advantages of Weibull c.d.f. as Macromodel:

1. The parameters of Weibull distribution function can be calculated relatively easily as they are based on the Gamma function.

2. A simple model can be obtained as two parameters are sufficient for regression.

A typical V/t rising waveform obtained from an IBIS file is shown in Figure 3.2. As can be seen, the rising transient response waveform is a monotonically increasing curve. Therefore, the data-set can be fitted to a distribution function. In addition to being monotonically increasing, the data set is also positively skewed. Due to these characteristics, Gamma distribution, Beta distribution [34] and Weibull distribution functions may be used as the regression function. Each of these distribution functions along with associated delay and slew metrics were discussed in background work. Weibull c.d.f (WCDF) is selected as the macromodel as the parameters of WCDF are based on the Gamma function, which can be calculated quickly and accurately. In addition, Weibull c.d.f takes a simple functional form. Figure 3.5 shows the Weibull c.d.f. waveform for different values of its parameters α and β.

By changing the values of 'α' and 'β' of WCDF, various rising and falling waveform data of IBIS models (examples include Figures 3.2 and 3.4) can be fitted. In the developed model generation tool, regression analysis is performed over V/t data of an IBIS file by

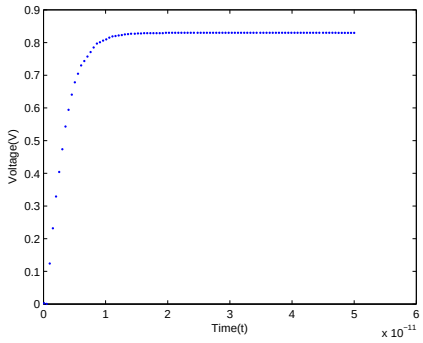

Fig. 3.4: Inverter: V/t rising waveform

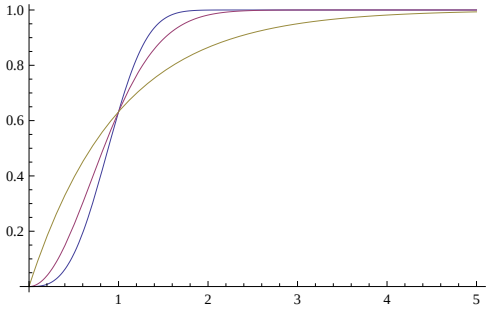

Fig. 3.5: Weibull c.d.f.

fitting the data to a Weibull CDF and obtaining the estimated values of its parameters $\hat{\alpha}$, $\hat{\beta}$. With this information, the mean and variance of the distribution is estimated from Equations 3.36 and 3.37. Once the parameters are estimated, rise and fall times can be obtained directly by solving Equation 3.29 for the 20% delay value and 80% delay value. Solving Equation 3.29 for time t_ϕ and substituting F(t) to be 'ϕ' as in [20], we get:

$$t_\phi = \beta.ln(1/(1-\phi))^\theta \tag{3.32}$$

The 20% and 80% delay points on the waveform can be calculated by substituting ϕ = 0.2 and ϕ = 0.8 respectively in the above equation.

$$t_{0.2} = \beta.ln(1/(1-0.2))^\theta = \beta.ln(0.125)^\theta \tag{3.33}$$

$$t_{0.8} = \beta.ln(1/(1-0.8))^\theta = \beta.ln(5)^\theta \tag{3.34}$$

Therefore, the slew (rise time and fall time) values can be obtained by subtracting Equation 3.33 from Equation 3.34. The estimated value of rise time and fall time through the proposed method are more accurate as the recorded data provided in IBIS file is the analog output of a digital pin.

Parameter Estimation: Once the rise and fall times are estimated, the next step of the model generation tool is estimating poles of the component or circuit that the IBIS file represents. Consider that the simulated component in Figure 2.2 from Chapter 2 is a CMOS inverter. When a step input is applied to the inverter, it's output goes low, the pull-up section of the output buffer is activated and rising waveform is obtained. Similarly, pull-down section of the output buffer is activated when input of inverter is logic low. The V/I curves for the output buffer is obtained with transistors fully on as discussed previously [10]. The steps for parameter estimation are discussed below.

1. Moment Matching: After obtaining the coefficients of the regression function, rise time and fall time were estimated. The next step of model generation is to estimate the poles of the circuit. The key idea applied here is *moment matching*, a technique where the moments of the selected distribution function are matched to the moments of the circuit. The underlying circuit is assumed to be modeled as an RC circuit.

   It has been shown that the impulse response of a RC circuit can be modeled by a probability density function [20] as it meets the following requirements:

$$\begin{aligned} h(t) &> 0 \,\forall t \\ \int_0^\infty h(t)dt &= 1 \end{aligned} \tag{3.35}$$

   It is well known that the integral of the impulse function is the step function. The step response of an LTI system would therefore be a cumulative distribution function. The key step in moment matching technique is to match the mean, variance, skewness and

other higher order central moments of a selected distribution function to the moments of the impulse response of the circuit. Since Weibull distribution function is used here, it's mean and central moments are matched to the circuit moments. Moment matching equations are given below:

$$\mu = -m_{c1} \tag{3.36}$$

$$\sigma^2 = 2m_{c2} - {m_{c1}}^2 \tag{3.37}$$

The next step to relate component transient response and its moments is to match the circuit moments of the simulated component (VLSI inverter) to the mean and variance of the proposed macromodel (Weibull CDF). Matching Equation 3.27 to Equation 3.36 and Equation 3.28 to Equation 3.37, the following relationship is obtained between the circuit moments and the mean and central moments of the Weibull distribution function:

$$m_{c1} = -\beta.\Gamma(1+\theta) \tag{3.38}$$

$$\begin{aligned} 2.m_{c2} - {m_{c1}}^2 &= \beta^2.(\Gamma(1+2\theta) - \Gamma^2(1+\theta)) \\ m_{c2} = 0.5.\beta^2.(\Gamma(1+2\theta) - \Gamma^2(1+\theta) + \Gamma^2(1+\theta)) \\ m_{c2} &= \frac{1}{2}\beta^2.(\Gamma(1+2\theta)) \end{aligned} \tag{3.39}$$

Similarly, higher order circuit moments can be obtained by matching them to higher order central moments of Weibull distribution function. Once the circuit moments

are calculated, as many system poles as required can be estimated.

2. Estimating Poles : Once the circuit moments are calculated, poles of the transfer function of the circuit can be obtained by matching the transfer function to a reduced order Pade's approximation as discussed in the previous chapter [28]. Approximate poles are calculated in this work using the *three moments approximation* method [36], discussed in previous chapter, for a LTI system described by a reduced order transfer function, the first two poles are approximated with the first three circuit moments by the following equation:

$$\begin{aligned} p_1 &= -\left(\frac{m_{c2}}{m_{c3}}\right) \\ p_2 &= p_1 . \left(\frac{m_{c0}.m_{c2} - {m_{c1}}^2}{m_{c1}.m_{c3} - {m_{c2}}^2}\right) \end{aligned} \tag{3.40}$$

Equation 3.40 provides the dominant pole and the second pole approximations. Additional poles and residues can be estimated using this method depending upon the requirements of an application, but in this work, upto two poles are estimated.

Similarly, the residues can also be obtained using the circuit moments and the approximated poles. From AWE, we know that:

$$\begin{aligned} \frac{k_1}{p_1} + \frac{k_2}{p_2} &= m_0 = 1 \\ \frac{k_1}{{p_1}^2} + \frac{k_2}{{p_2}^2} &= -m_1 \end{aligned} \tag{3.41}$$

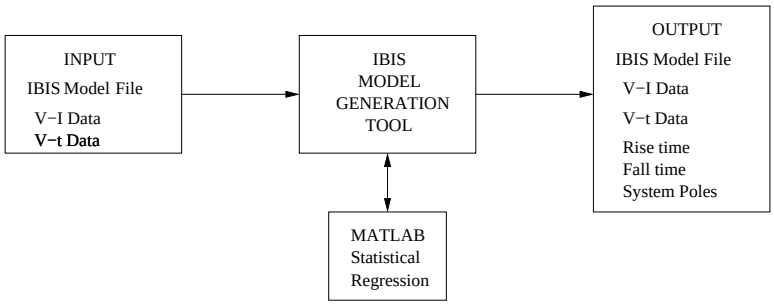

Fig. 3.6: IBIS Model Generation Tool

Solving for k_1 and k_2, we obtain the residues as

$$k_1 = \left(\frac{1 + m_1.p_2}{p_1 - p_2}\right).{p_1}^2 \tag{3.42}$$

$$k_2 = \left(\frac{1 + m_1.p_1}{p_1 - p_2}\right).{p_2}^2 \tag{3.43}$$

Once the poles and residues are obtained, the transfer function of the circuit can be written out in the standard poles and residues form. In this work, the transfer function is automatically generated. In the time domain, the function can be written in poles and residues form as :

$$f(t) = k_1.e^{-p_1.t} + k_2.e^{-p_2.t} \tag{3.44}$$

and the transfer function can be written as:

$$H(s) = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} \tag{3.45}$$

All the steps mentioned above are automatically generated by the developed tool. Once the tool obtains the transfer function, other characteristics may be derived from the transfer function. To the best of our knowledge, this work is the first attempt at extracting poles, rise time and fall time of a circuit specified by IBIS models.

Algorithm

The model generation methodology is summarized below as a pseudocode:

1. Fit Weibull c.d.f. to IBIS V-t data
2. Perform regression to obtain estimated values of α and β, $\hat{\alpha}$ and $\hat{\beta}$
3. Obtain estimated t_r and t_f with Equation 3.33 and Equation 3.34
4. Match Weibull p.d.f. moments to circuits impulse response moments
5. Obtain estimated poles using Equation 3.40
6. Obtain the estimated residues using Equation 3.43
7. Generate the transfer function and time-domain equation of the circuit and store the result in the IBIS file

3.5 Implementation and Results

The developed model generation tool, shown in Figure 3.6 is implemented with Perl programming language [108] [109]. It inputs V/t data from IBIS models and calls MATLAB to perform regression analysis on the data by following the steps outlined above. The code snippets provided by authors in [47] were modified for utilizing Weibull distribution function. The estimated values of poles, rise time and fall time obtained are then written back into the same IBIS file.

Results

In this section, the results obtained by the model generator is presented. The proposed methodology has been applied to a CMOS inverter at 22nm and 16nm technology nodes. The inverters were sized with the following ratios: at 22nm, $(\frac{W}{L})_P = \frac{144n}{22n}$ and $(\frac{W}{L})_N = \frac{400n}{22n}$,

Table 3.1: Performance parameters, 22nm

Parameters	Inverter1	Inverter2
Rise time	2.8302ps	2.7053ps
Fall time	0.5302ps	0.3088ps
Poles (pMOS)	5.642e+010, -3.41e+010	6.183e+010, -4.101e+010
Poles (nMOS)	-1.0918e+010, 4.5099e+010	-9.882e+009, 4.0167e+010

Table 3.2: Performance parameters, 16nm

Parameters	Inverter1	Inverter2
Rise time	2.9185ps	2.560ps
Fall time	0.4591ps	0.3342ps
Poles (pMOS)	8.1986e+010, -9.198e+010	5.818e+010,-3.0087e+010
Poles (nMOS)	3.8762e+010, -1.4336e+011	-9.882e+009, 4.0167e+010

whereas at 16nm $(\frac{W}{L})_P = \frac{80n}{16n}$ and $(\frac{W}{L})_N = \frac{400n}{16n}$. IBIS files were generated using the S2IBIS tool. HSPICE [107] was used as the Spice simulation engine along with the S2IBIS tool. Predictive technology transistor model files used for creating IBIS files were referenced from [13]. The steps of the proposed methodology were coded in MATLAB to generate the results. The time complexity of the method is determined by the time taken to compute WCDF parameters. Tables I and II provide results for 22nm and 16nm technology nodes respectively.

3.6 Proposed Model Generation Method

For linear (linearized) circuits, a method to generate a macromodel given IBIS file description was implemented in [38]. In this work, the goal is to extend this to simple nonlinear examples. Transistor based circuit such as a CMOS inverting amplifier would form a good example. Understanding the numerical methods employed in SPICE (S2IBIS tool invokes SPICE program to generate IBIS data) is required and is well documented in [73]. Trapezoidal approximation for numerical integration is used during transient analysis simulation.

The proposed model generation method is described below. The main step is to relate the regression function to the selected companion model parameters and then to match them

with the given circuit. Most importantly, the model parameters require the knowledge of breakpoint voltages so that appropriate time steps can be determined with a discretization error bound and the circuit parameters be obtained.

Proposed Algorithm

The pseudocode for the proposed algorithm is given below.

1. Perform segmented regression on V-I data to obtain voltage breakpoints. Record the breakpoints and note corresponding time points in V-t data.

2. With the breakpoints, select a step size for time stepping V-t data assuming that the simulator used LTE based algorithm. Specify the error tolerance (the bound) on step size.

3. Select step size such that the error bound is met and the traversed points include the recorded breakpoints.

4. Build the companion model assuming trapezoidal integration scheme and obtain model parameters.

5. Obtain required circuit parameters from model parameters.

Currently, the IBIS file for CMOS amplifier has been generated with S2IBIS tool. The linearized model was generated with the method described in [38]. The method described above can be used to generate the circuit parameters. Many semiconductor companies provide IBIS models of their products (eg. [86]).

3.7 Summary

This chapter presented a tool to generate a macromodel of a circuit specified by IBIS models. Regression analysis with moment matching method estimates the values of rise time,

fall time and poles of the circuit under a first assumption. The outputs of the tool is an IBIS model file with estimated rise time, fall time and transfer function. WCDF was used as a regression function. Figure 3.6 shows the implemented tool. Another method for generating a macromodel of a circuit specified by IBIS model file is proposed in the previous section. The method enumerates steps to obtain a companion model of the circuit.

The next chapter presents two new netlists for on-chip behavioral signal integrity analysis with the generated IBIS models. These netlists are used for statistical eye analysis to obtain performance metrics of interest. In Chapter 5, a new early performance prediction method is described that utilizes these netlists to provide a better performance estimate at an early stage of design. When viewed from the higher abstraction level, these netlists constitute a new abstraction layer at the circuit-level.

CHAPTER 4
A TOOL FOR BEHAVIORAL SIGNAL INTEGRITY ANALYSIS

4.1 Introduction

The previous chapter described a method to generate a macromodel of a circuit specified by IBIS models. This chapter presents a tool to evaluate performance of point-to-point on-chip interconnects considering factors that affect signal integrity such as intersymbol interference (ISI), crosstalk and reflections. Point-to-point link topology is employed as it provides maximum signal integrity. Of particular focus is behavioral simulation utilizing generated IBIS models. Analysis with these behavioral models have the advantages of *fast simulation time*, *good accuracy* and *re-usability*. This link is represented by a **netlist** comprising of a transmitter (called driver) circuit, an interconnect structure and a receiver circuit. The developed tool generates the required IBIS models and assembles a netlist in IBIS Interconnect Subcircuit Specification (IBIS-ISS) format. The netlist is then employed for behavioral signal integrity analysis to evaluate serial link performance. If further performance improvement is required, equalizers at the receiver (and transmitter) are added to the IBIS-ISS netlist. This new netlist, called IBIS-AMI-ISS netlist, follows the IBIS

Algorithmic Modeling Interface (AMI) standard (IBIS versions 5.1, 6.0). To achieve this, the existing IBIS-ISS standard has to be enhanced to incorporate AMI models.

The remainder of the chapter is organized as follows: Section 4.2 describes various factors that affect signal integrity. These factors must be accounted for while evaluating link performance. The following section introduces two new netlists descriptions that can be and are used for evaluating signal integrity. Advantages are highlighted. Many core processors that are designed to perform intense computations require a robust on-chip communication protocol for efficient communication. Section 4.4 describes the PCI express (PCIe) protocol employed for this work. PCIe supports point-to-point communication links. The next section discusses simulation-based approach for estimating performance of serial links described by the new IBIS-ISS, IBIS-AMI-ISS netlists. Section 4.6 describes a tool for on-chip link performance analysis that employs the developed netlists. A template that follows the PCIe spec., required for simulation is discussed. Finally, the work is summarized.

4.2 Factors Affecting Link Performance

This section discusses the technical background required for serial link analysis. Factors affecting the performance of serial link are discussed. Mathematical model for each physical phenomena affecting the performance are highlighted. Jitter is broadly defined as the deviation of a signal from its ideal value. There are two sources of jitter: time jitter and voltage (amplitude) jitter. Some definitions related to electrical signaling and jitter are given below:

1. Unit Interval (UI): Unit interval is the ideal time duration of the single bit of data. It is defined as the reciprocal of data rate. For example, PCIe Generation 3 has a UI of 125ps.

2. Bit Error Rate (BER): BER is defined as the ratio of the number of bits received in error to the number of bits transmitted.

3. Eye Diagram: An eye diagram is a composite view of all the bits superimposed on each other. Reduced eye opening indicates the presence of jitter.

4.2.1 Jitter Taxonomy

Sources of jitter can be classified broadly as deterministic jitter and random jitter. Deterministic jitter comprises of three major subcomponents, each of which is described below.

Deterministic Jitter

Subcomponents (sources) of deterministic jitter are:

- Data Dependent Jitter (DDJ): This type of jitter is caused by variation in incoming data sequence. DDJ is subdivided into two components: intersymbol interference (ISI) and duty cycle distortion (DCD). ISI occurs when the previous data bit affects the current data bit. DCD on the other hand is a type of jitter that causes distortion in the duty cycle of the pulse being transmitted. The main cause of DCD is different rise times and fall times of the driver. Reference [50] shows these effects in relation to the clock signal. DCD is usually tested by passing a string of alternate ones and zeroes, reducing the possibility of ISI. For signal integrity analysis, a pseudo-random bit sequence is generated with a linear feedback shift register with the rise time and fall time data obtained by the model generation tool.

- Bounded Uncorrelated Jitter (BUJ): This is caused mainly due to coupling between adjacent wires carrying information.

- Periodic Jitter (PJ): Periodic jitter refers to variations in the signal's period caused due to electromagnetic interference [75].

Random Jitter

Another component that constitutes total jitter is random jitter. This is usually modeled by the Gaussian distribution characterized by its standard deviation (σ). Otherwise, a specific distribution function is specified. The mathematical model for random jitter is given by the following:

$$J_{RJ} = \frac{1}{\sigma.\sqrt{2\pi}} e^{-(\frac{x^2}{2\sigma^2})} \tag{4.1}$$

The signal integrity analysis tool HSPICE Stateye follows the Gaussian distribution function [53].

The total jitter is therefore expressed as the sum of the deterministic jitter and random jitter. Deterministic jitter is specified by the peak-to-peak value whereas the random jitter is specified by the root mean squared value, due to its random nature. Therefore, to obtain the total jitter value, random jitter is first converted from root mean square value to peak to peak value and then summed with deterministic jitter. The equations for calculating jitter are described below [50]:

$$TJ_{p-p} = DJ_{p-p} + RJ_{p-p} \tag{4.2}$$

$$RJ_{p-p} = k_{\sigma}.RJ_{RMS} \tag{4.3}$$

Where, k_{σ} is the multiplicative coefficient to convert from RMS value to peak-to-peak value. For a BER of 10^{-12}, k_{σ} = 14.07. PCI express (PCIe) jitter budget is defined and documented in [45]. PCI special interest group (PCI-SIG) specifies allowable deterministic and random jitter for the transmitter, the receiver and the serial link.

4.2.2 Jitter Models

Mathematical models for jitter components are defined in this section [75]. Jitter exhibits stochastic behavior and can therefore be modeled as a probability density function (p.d.f.).

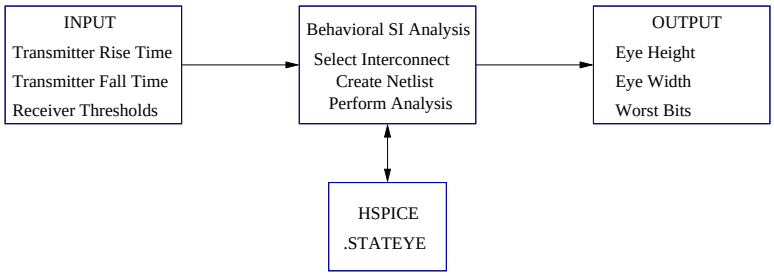

Fig. 4.1: Circuit-level Behavioral Link Analysis with Generated IBIS Models

The bit error rate is its integral, the cumulative distribution function. This can be expressed mathematically as:

$$BER = \int_{-\infty}^{v_{mid}} f_v(x) = \int_{v_{mid}}^{\infty} f_v(x) = 0.5 \tag{4.4}$$

where, $f_v(x)$ is the jitter p.d.f.

Deterministic jitter subtypes can be described by a particular p.d.f. In general, they can be described by the dual-dirac p.d.f., defined in Equation 4.8. Random jitter, on the other hand, is typically modeled by Gaussian distribution function, with a single parameter, its standard deviation (σ).

$$J_{RJ} = \frac{1}{\sigma.\sqrt{2\pi}} e^{-(\frac{x^2}{2\sigma^2})} \tag{4.5}$$

Models for deterministic jitter depends on the specific type and each is modeled by a particular distribution function. For deterministic jitter subtypes namely, intersymbol interference, duty-cycle distortion and periodic jitter, the mathematical models are given as below:

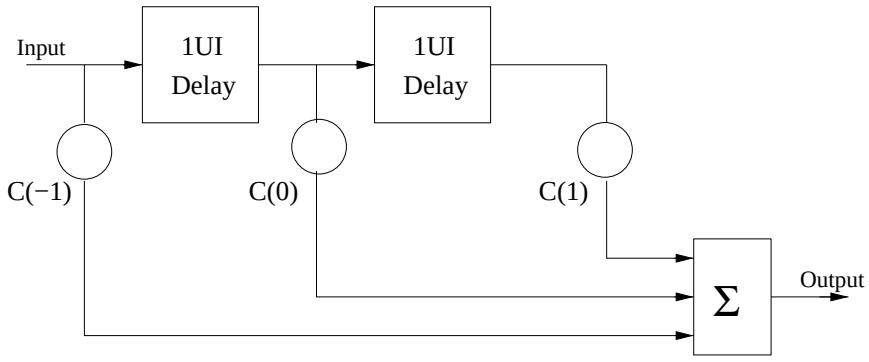

Fig. 4.2: Block Diagram 3-tap FIR Filter

$$J_{ISI}(x) = \Sigma_{i=1}^{N} P_i.\delta(x - x_i) \tag{4.6}$$

$$J_{DCD}(x) = \frac{\delta(x - W/2)}{2} + \frac{\delta(x + W/2)}{2} \tag{4.7}$$

$$J_{PJ} = A.cos(\omega.t + \theta) \tag{4.8}$$

$$J_{PJ} = \frac{1}{\pi.\sqrt{A^2 - x^2}} \tag{4.9}$$

Each of the deterministic jitter types are bounded and can be characterized by a particular probability density function. However, a reasonable density function that can be used to describe this form of jitter is the dual Dirac function, described by Equation 4.8.

4.2.3 Utilizing Equalizers to Minimize ISI

Consider the system shown in Figure 1.1 from Chapter 1 that is implementing an application that has distributed tasks among cores and the result is transferred over the interconnect. The rate of this data transfer increases with each generation of communication protocol. At data rates of 8 Gbps and beyond (state-of-the-art PCIe Gen. 3 specification), the effects of intersymbol interference (ISI) is highly pronounced due to which the received eye is closed. To open the eye to improve performance, equalizers are employed at the receiver and/or transmitter. Several equalizer implementations have been documented in

recent literature, primarily based on finite impulse response filters. At the transmitter side, feed-forward equalizers, built with 3-tap FIR filters are usually employed to pre-emphasize the input signal to the interconnect. A block diagram representation is shown in Figure 4.2. Within the IBIS specification, the algorithmic model interface (AMI) description enhancement (ver 5.1, 6.0) is utilized. Finite Impulse Response (FIR) based feed-forward equalizer (FFE) is suggested at the transmitter end. At the receiver end, a decision feedback equalizer (DFE) is employed that comprises of the FIR filter structure along with a decision block that decides on the logic value of the received signal. Both FFE and DFE are described below with focus on relevant background for this work.

Feed-forward Equalizers

At the transmitter side, the signal to be transmitted is emphasized. An FIR filter structure is capable of emphasizing the signal. At the circuit level, current mode or voltage mode drivers are employed to implement the pre-emphasis functionality, with current-mode driver, current-sources are added to boost the current driven to the interconnect, the transmission-line when there is a bit logic transition (current steering method). From the FIR filter implementation that comprises of unit-delays and multiplying coefficients summed together to obtain the emphasized signal at the input of the transmission-line. At the block level, FIR filter consists of delay circuits (delay of one unit interval) along with a multiplicative coefficient summed together to obtained an output signal that comprises of more signal levels than the input signal at the same data rate. PCIe Gen 3 recommends a 3-tap FIR filter at the transmitter end [43]. A 3-tap filter scheme is shown in Figure 4.2. An implemented topology for current-mode circuit-level implementation is given in [44].

Decision Feedback Equalizers

DFE comprises of a decision block along with the FIR filter structure in feedback path. The goal of the decision block is to determine the logic value of the received signal. The

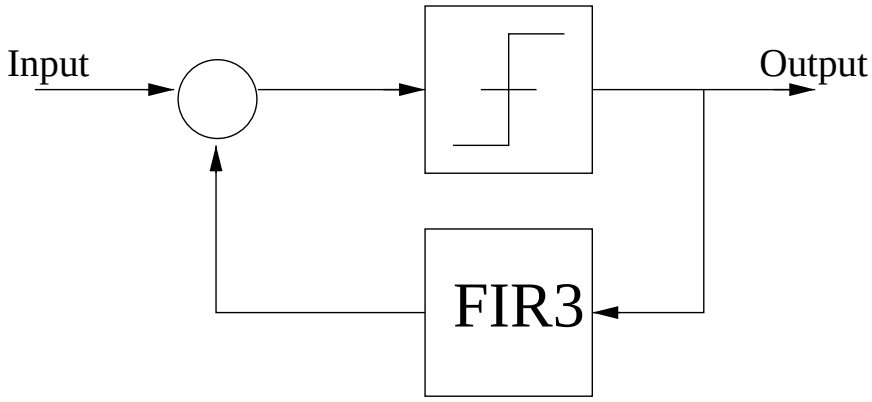

Fig. 4.3: Block Diagram 3-tap FIR Filter

decision block is implemented with sense amplifier, comparators. DFE works by minimizing/eliminating the postcursor of the received signal caused due to ISI. The decision block compares the previous value against a threshold to determine the logic value. Figure 4.3 shows the block diagram of DFE implementation.

4.2.4 Crosstalk Analysis

An important factor while considering the overall link performance is the effect of crosstalk noise on wires running close to each other. Crosstalk occurs when a signal propagating on a wire (called aggressor) induces a spike (noise) on a quiet adjacent wire (called victim). This has an adverse impact on signal integrity and therefore affects performance. There are two crosstalk subtypes: Near end crosstalk (NEXT) and Far end crosstalk (FEXT). The focus here is on NEXT caused mainly due to capacitive coupling between adjacent wires (on a chip). To evaluate this, wires running in parallel and in close proximity to each other is considered with one or more wires acting as aggressors and the impact on the performance is studied on the victim. In this work, crosstalk is quantified with the eye height metric obtained by simulating the victim's wire (Alternatively, crosstalk can be analyzed by performing transient analysis of the coupled wires with SPICE).

The interconnects are modeled as coupled T-lines with appropriate termination scheme for each process technology. The developed tool then automatically selects the model that

closely meets the requirements. Both single-ended and differential schemes are analyzed.

From a signal reliability perspective, data related error due to crosstalk is classified as functional error and timing error. Both errors impact the correct data reception at the receiver end. Mathematical model for analysis of these two crosstalk metrics are detailed in [87]. Consider a point-to-point link comprising of a transmission line connecting D flip-flops at driver and receiver end. Functional error occurs when a voltage spike is induced on the victim's line by the aggressor and the receiver incorrectly samples it as logic high (or low) causing data reception error. Depending on the protocol specification, this would require data retransmission, affecting overall latency. Timing reliability is caused due to a slowdown or speedup in victim's signaling, leading to incorrect sampling of data at the receiver due to setup and hold time violations. A generic analytical study is provided in Ref. [88].

4.3 Netlists for On-Chip Link Analysis

A netlist needs to be formed to simulate a system. In this section, new netlists comprising of generated IBIS models (described in Chapter 3) is introduced. The netlists represent a point-to-point connection between two circuits as shown in Figure 4.4. Its components are:

- Transmitter (Driver): IBIS file generated by the model generation tool consisting of generated parameters (Figure 4.1).
- Receiver: IBIS file with specified input pin model.
- Wires/Interconnects: Single-ended and differential T-line schemes along with various termination schemes.
- Equalizers: Algorithmic model (AMI) description for equalizers if used. They may be employed at the transmitter end and receiver end.

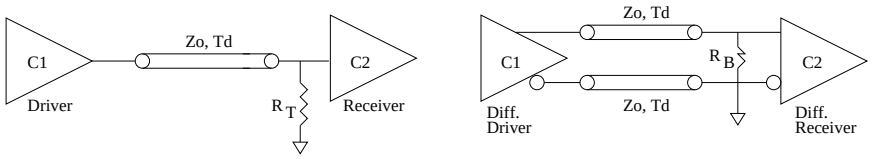

Fig. 4.4: Presented Netlists for Serial Link Topologies

This work presents two new netlists that utilize the generated IBIS models described in previous chapter. For circuit-level behavioral simulation, two netlists are created:

- IBIS-ISS netlist - This netlist is generated using the IBIS-ISS specification. This specification is a set of guidelines and isn't a netlist description by itself, a requirement for behavioral serial link simulation.

- IBIS-AMI-ISS netlist - The description of this netlist, shown in Figure 4.5, enhances the spec. by EIA/ANSI standard by introducing the capability of incorporating equalizers to improve serial link performance beyond 8Gbps data rates.

This would discussed in detail in subsequent sections. First, interconnect topologies that are suggested for serial links is discussed in the following section. Figure 4.4 shows a netlist that incorporates these interconnect topologies.

4.3.1 Interconnect Topologies

Lossless (LC) and low-loss transmission lines (T-line) models are incorporated in the netlist (ITRS recommendations, [3]). In current nanometer process technology, edge rates are in pico seconds (ps) range. This signal's edge rate is much less than the propagation delay of the line, requiring T-lines to be used as an alternative for RC wires. T-Lines are currently used for global on-chip interconnects [71] in addition to repeated RC wires for local interconnects. T-lines have lower latency and energy per bit dissipation, outperforming RC wires in nanometer process technology as demonstrated in [89]. LC T-lines can therefore be used for relatively short distances as well.

When to consider transmission lines for interconnects?

With process technology in the nanometer era and the edge rates in the pico seconds (ps) range, LC transmission lines are becoming a necessary alternative to RC wires for interconnects. The important question that arises is: When should one model the interconnection with transmission lines? Consider that the interconnect delay be given by T_D and the edge rate at the driver end be given by t_{rf}. A rule of thumb for considering T-lines as on-chip interconnect is given by the following equation.

$$t_{rf} \leq \frac{T_D}{5} \tag{4.10}$$

A more detailed work describing the guidelines for considering transmission line effects for an on-chip interconnect topology is given in [27]. For lossless and lossy transmission lines with parameters R,L,G,C, the characteristic impedance is given by:

$$Z_0 = \sqrt{\frac{L}{C}} \tag{4.11}$$

$$Z_{lossy} = \sqrt{\frac{R + j\omega L}{G + j\omega C}} \tag{4.12}$$

To alleviate reflection and intersymbol interference to improve signal reception, several termination schemes can be considered for each T-line type. Source and load termination schemes are employed with resistance (and capacitance) values for each technology node. Each of these topologies provides advantages and certain limitations, discussed briefly below.

When resistive load termination scheme is employed, the maximum eye voltage at the receiver end reduces as compared to interconnect topology with no termination scheme, but this eye voltage remains fairly constant as bit rate increases. This is advantageous from a scalability point of view [72]. Another approach developed is capacitive source termination of RC wires represented by the π-model which boosts the signal on the wire,

thereby improving the interconnect latency.

To perform simulation, transmitter and receiver are specified by generated IBIS file [38]. Interconnects are modeled by LC T-lines. Rise and fall time values from driver are imported into a LFSR spec. to generate a PRBS23 sequence that is propagated along the link. HSPICE Stateye tool [107] is employed for statistical eye analysis.

The developed tool, based on the spec., automatically selects the topology that meets the requirements. For each process technology, analysis is performed with single-ended and differential T-lines.

4.3.2 IBIS-ISS Netlist

In this work, a new netlist for behavioral signal integrity simulations is described. This netlist, called IBIS-ISS netlist is based on the IBIS interconnect subcircuit specification (ISS). The contribution of ISS is allowing incorporation of existing SPICE based netlist structure for describing interconnects. This work is, to the best of our knowledge, the first attempt at defining a netlist that utilizes this new specification with two purposes: allowing fast, scalable behaviroral signal integrity simulations and using this netlist for a better performance prediction at an early stage of design. The netlist defined in this work consists of a generated IBIS model file as driver, a SPICE interconnect subcircuit description of transmission lines and an input model of IBIS file as receiver. The netlist is shown in Figure 4.4.

4.3.3 IBIS-AMI-ISS Netlist with Equalizers

As mentioned in previous section, at high data rates, receiver eye is closed due to inter-symbol interference. For simulation-based performance estimation at the physical-layer abstraction, an enhanced netlist has to be created that incorporates equalizers to minimize ISI into the IBIS-ISS netlist described in previous section. This netlist, called IBIS-AMI-ISS netlist would comprise of IBIS-AMI description of equalizers incorporated along with

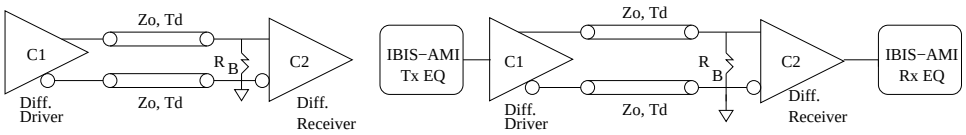

Fig. 4.5: Serial Link Netlist With Equalization

the generated IBIS-ISS netlist. This would require additions to the IBIS interconnect subcircuit specification. Algorithmic Modeling Interface (AMI) capability was added to the IBIS specification with the specific purpose of describing equalizers at the algorithmic-level and allow linear time invariant and nonlinear time varying. Algorithmic description of equalizers can be specified using IBIS version 5 and later. IBIS version 6 (September 2013) has an updated, more comprehensive description in this aspect. Depending upon the tool implementation, for a particular equalizer description, tap coefficients can be directly specified that is used to compute the output signal from the current/input signal assuming a FIR filter structure.

To perform simulation, transmitter/driver and receiver are specified by generated IBIS file [38]. IBIS-ISS or IBIS-AMI-ISS netlist is created as shown in Figure 4.4 and Figure 4.5. Rise and fall time values from driver are imported into the LFSR template to generate a PRBS23 sequence that is propagated along the link. A set of algorithmic description of 3-tap equalizers is first created following a pre-defined constraint that depends on the protocol. For simulation, one of the equalizer description is invoked to form a IBIS-AMI-ISS netlist to generate the receiver eye diagram. The developed tool, based on the spec., automatically selects the required netlist.

4.4 PCI Express Protocol and Netlists

Peripheral Component Interface - Express (PCIe) is a serial, dual simplex, point-to-point communication protocol. It is a well defined multi-layered third generation protocol developed by Intel Corporation [46]. PCIe physical layer (PHY) specification is described in

some detail as this specification is referenced while specifying jitter budget and creating a template file for interconnect analysis. Jitter model and jitter budget for the link are well documented for PCIe protocol [45].

4.4.1 PCIe PHY Layer

PCIe protocol is defined with a layered architecture with the transport layer specification independent of the physical layer spec. This layered architecture allows future technological advances to be incorporated in the PHY layer specification, with the transaction layer and data-link layer specification remain unchanged. The advantage of this approach is that standard Transaction Layer IPs can be developed at a higher abstraction level for fast functional and performance simulation that can be used with future protocol generations. These IPs also aid in design reuse. Several proposals for future interconnect topologies have appeared in recent literature with optical interconnects being prominent alternative to current practices (Generation 4 continues electrical interconnects).

The PHY layer of PCIe is composed of two sub-layers, the logical layer and physical layer. PCIe PHY follows a serial dual-simplex point-to-point topology. Two lanes on opposite direction transport the data payload independently. Communication link with 50Ω characteristic impedance is required as per the standard. LC transmission lines are used to implement the interconnect link [2], due to improved latency performance as process technology scales to nanometer era. The unit interval is 125ps for Gen 3 with 128/130b encoding to reduce the 25% performance loss due to 8b/10b encoding used in Gen 2.

4.4.2 PCI Express Jitter Budget

Jitter budget of PCIe communication protocol is summarized. The values of jitter components both at the transmitting end and the receiving end are tabulated for reference. The model and derivation of Jitter budget is documented in [45]. This is summarized here for reference. Tables 4.1 and 4.2 tabulates the jitter budget for a PCIe link with standard values

Table 4.1: PCI Express Jitter Budget [45]

Jitter Type	Transmitter	Receiver	Link
Det. Jitter (p-p)	25ps	25ps	12.5ps
Random Jitter (p-p)	1.6ps	1.6ps	0s
Total Jitter (sum)	26.6ps	26.6ps	12.5ps

Table 4.2: PCI Express Jitter Budget

Jitter Type	Transmitter	Receiver	Link
Det. Jitter (p-p)	7ps	11.8ps	5ps
Random Jitter (p-p)	0.8ps	1.8ps	0.8s
Total Jitter (sum)	7.8ps	13.6ps	5.8ps

of total jitter, random jitter and clock jitter. These values are referenced while creating a template for SI analysis.

4.4.3 IBIS-AMI-ISS Netlist for PCIe Spec.

IBIS-AMI-ISS netlist can be used to represent a lane in PCIe based point-to-point link. PCIe Gen 3 specifies the use of a 3-tap FIR filter for pre-emphasis while a DFE for receiver equalization. IBIS-AMI-ISS netlist can be used to specify this when required. For most portion, this would require an enhancement to IBIS-ISS spec. to specifically incorporate the AMI description of equalizers. This has been defined and implemented in this work.

In the next section, an approach to estimating performance at physical layer level is discussed. There are two approaches to estimating performance; analytical approach and simulation-based approach. Analytical approach involves selecting a mathematical model to effectively describe the behavior of the link and the parameters of the model provide an estimate of the performance. The choice of the appropriate model is critical to obtain resonable estimates. Simulation-based approach involves performing computer-aided analysis by providing appropriate input vectors (values) and studying the obtained outputs, the estimated values. This approach is useful for serial link analysis where signal integrity is main focus. Through this approach, factors affecting signal integrity and therefore reliable data transfer can be effectively studied.

4.5 Simulation-Based Performance Estimation at PHY

Hybrid modeling [90] consists of a combination of analytical model and simulation-based model. This approach is well suited for the early performance prediction methodology, presented in the next chapter. In this section, simulation-based approach for performance parameter estimation at the physical level is discussed. The system model maps to the estimated parameters to obtain a more accurate performance prediction. Consider a simple synchronous x2 link. At PHY, the link is modeled by bridged differential T-lines for a given process technology. Since the goal is to predict performance, a method to extract relevant parameters from T-line topology is presented. For a given process technology, the method to obtain these values at the physical layer is summarized by the following algorithm:

1. Perform Stateye analysis at the input port and output port.
2. Unfold the obtained eye diagrams at both ports.
3. Generate the histogram from the unfolded waveforms.
4. Measure the mean of histogram at each port. These values correspond to λ and μ coefficients respectively.
5. Record the values in a LUT.
6. Repeat for each new topology.

modeling the link as a queueing system at the system-level, the problem of obtaining a more accurate performance estimate at system-level reduces to estimating its birth (λ) and death (μ) coefficients. The queue model at system-level is mapped to an instance of LUT with generated mean-variance values from simulation of the IBIS-ISS (or IBIS-AMI-ISS) netlist, described in the previous section.

4.5.1 Efficient Topology Selection

A method to obtain efficient topologies is presented in this section. The method applies mean-variance analysis. Mean-variance analysis, developed by Markowitz is widely used in financial sector in portfolio theory. In that domain, for an investment, the goal is to maximize return for a given risk. Or, obtain returns such that the risk is minimized. Mean-Variance analysis provides a set of investments that meet the above criteria and is called the efficient portfolio. In this work, this mean-variance analysis method is applied to obtain "low-risk" interconnect topologies to obtain efficient performance prediction of ASICs.

For performance and reliability analysis of modern ASICs, the problem at hand is to obtain interconnect topologies that minimize the risk (of errors) in data reception. This risk is caused due to crosstalk noise affecting signal integrity of interconnects, the main performance bottleneck at the physical level. In this work, crosstalk noise is mapped to the variance (std. deviation) of the histogram of the model obtained by simulation. Then, to obtain an efficient estimate of performance considering the reliability of signal, a set of interconnect topologies (or a single topology) is selected from a list of interconnects with the mean-variance rule. The selected topology or topologies is known as the efficient frontier, defined below. This selection rule is defined as follows [91]. Consider two interconnect topologies, IntcA and IntcB with mean and variance values (μ_a, σ_a) and (μ_b, σ_b) respectively. Interconnect topology IntcA is said to mean-variance dominate topology IntcB if the following holds:

$$\mu_a < \mu_b \;\; and \;\; \sigma_a < \sigma_b \tag{4.13}$$

$$or$$

$$\mu_a > \mu_b \;\; while \;\; \sigma_a \leq \sigma_b \tag{4.14}$$

A topology or topologies that *do not* meet the above criteria is known as the *efficient*

frontier. To analyze crosstalk, the aggressor is driven by a PRBS and its effect on the victim is simulated and reported. The mean-variance rule is applied at the input and output ports of the interconnect to study the impact of crosstalk. Then, from the obtained efficient set of topologies, the required netlist can be generated. Therefore, variance or std. deviation is used as a rule to select the component with lower crosstalk influence. Consider interconnect topologies with mean, std. deviation values obtained from the generated histogram (Algorithm Section V-B). Considering the mean-variance rule, the topology selection algorithm above is updated to obtain efficient topologies as given below.

4.5.2 Efficient Topology Selection Algorithm

For a given process technology, do the following:

1. Perform Stateye analysis at the input port and output port.
2. Unfold the obtained eye diagrams at both ports.
3. Generate the histogram from the unfolded waveforms.
4. Measure the mean and std. deviation of histogram at each port. These values correspond to (λ,σ_1) and (μ,σ_2) coefficients respectively.
5. Record the values in a look-up table (LUT).
6. Repeat for each new topology.
7. Record mean, variance values (to output) in a LUT.
8. Obtain the efficient frontier with Markowitz's rule.
9. Record the efficient frontier in another LUT for mapping.

The obtained look-up tables would be used by the early performance prediction tool to select the netlist(s) that closely meets the requirements.

4.6 Automated Behavioral On-Chip Link Analysis

In this section, a tool to perform behavioral on-chip link analysis is presented. The important advantages of the tool are scalability and reusability. The output IBIS files produced by the model generation tool are then used to compose a system (i.e., create a netlist) and analyze performance.

Circuit C1 behaves as the transmitter (called driver) while circuit C2 is the receiver. The two circuits are connected by an interconnect link modeled as a transmission line with characteristic impedance (Z_0) and delay time (TD). Interconnect link performance analysis at the physical layer involves the following: First, the user selects the required driver, receiver and specifies the performance requirement. Then, the automation tool will generate the rise time and fall time information from the circuit specified by IBIS models and passes on this information to the Stateye tool. Stateye analysis tool will generate a PRBS sequence with the obtained rise time and fall time data. An interconnect topology (with a particular Z_0, TD) is then selected. Finally, link performance analysis is performed and the results obtained are recorded.

The application of model generation tool is to obtain, in an automated fashion, a more accurate estimate of performance parameters and aid in early performance prediction. The rise and fall time information are first generated with the developed model generation tool, discussed in the previous section, for C1 (and C2). The model parameters are passed to a SI analysis tool (HSPICE [107] Stateye) to analyze the system performance considering the effects of reflection due to impedance mismatch.

4.6.1 Experimental Setup

For crosstalk analysis, to model the coupling capacitors between adjacent wires, SPICE T-line U-model can be employed. Advanced tools have the capability to consider the effects of aggressors on victim wire. For most portion, differential topology is used. For each

process technology, differential signaling scheme is implemented with resistive termination of $2.Z_0$. A template for each process technology is created with appropriate jitter values for PCIe Gen3, Gen4.

4.6.2 Netlist Generation Template

Lossless (LC) transmission lines (T-line) are used to model serial point-to-point communication link between two circuits in this work [4]. LC T-lines are a necessary alternative to RC based interconnects in the nanometer regime [66], where the signal frequency and data transfer rates are in the multi-giga hertz range. In addition, ITRS projection data [4] also recommends LC T-lines as an alternative to RC wires for on-chip global interconnects. Recent work have demonstrated that LC T-lines outperform RC lines in nanometer process technology [57]. With this as background, for this work, LC T-lines, specified by characteristic impedance (Z_0) and delay time (T_D) are utilized for modeling the communication link. Source termination and load termination T-line topologies are also applied to reduce signal reflection. The goal is to facilitate detailed analysis. The characteristic impedance and delay time of a LC T-line in terms of L and C are given by [23]:

$$Z_0 = \sqrt{\frac{L}{C}}...(ohms) \tag{4.15}$$

$$T_D = \sqrt{L.C}...(ps/mm) \tag{4.16}$$

For PCIe protocol, typical value of characteristic impedance is 50Ω. The IBIS models are considered to be generated with this value as reference. Variations are considered so that the effects of impedance mismatch can be studied with the developed tool. Based on the specification provided by the user, the developed tool will select LC topologies that closely meets the requirements. The communication protocol in this work follows the PCIe base specification. PCIe is an established computer peripheral protocol that is widely used

for connecting multiple integrated circuits on a board. The same protocol is adopted here for on-chip interconnections.

Deterministic jitter (DJ) in the form of periodic jitter has been applied along with a predetermined duty cycle distortion (DCD) value. Both parameters can be updated depending on the user requirement. The driver rise time and fall time values are imported as inputs for the pseudo random bit sequence generator (PRBS). A linear feedback shift register implements the PRBS. LC transmission lines based interconnects are primarily used in this work. Recent research work [57] have demonstrated that LC T-lines have lower latency and energy per bit dissipation as compared to RC based interconnects, which makes them a good replacement for RC wires. They do however come at a cost of greater area consumption. Latency is of primary interest in this work.

4.6.3 Behavioral Link Analysis Method

The steps for the automated SI analysis methodology is given below. For each process technology, do the following:

1. Generate the estimated rise time (t_r) and fall time (t_f) information with the model generation tool.

2. Pass t_r and t_f values to HSPICE®Stateye simulation file.

3. Invoke Statistical eye (Stateye) analysis tool and perform analysis as follows:

    - Generate a pseudo random bit sequence (PRBS) with the obtained t_r and t_f values at the driver end.
    - Select a model for the interconnect (transmission line) with a particular Z_0, T_D (extracted from RLGC) parameters.
    - Perform stateye analysis on the selected topology and obtain the BER.

4. With the IBIS input model file of the receiver, compare the obtained value with input threshold values (V_{inl} and V_{inh}).

5. End if requirements are met. Else, repeat the procedure.

Performance Metrics Evaluated

For a serial link with an IBIS driver, a receiver and a T-line interconnect, performance metrics considered are eye height and eye width for a BER of less than 10^{-12}. Both eye height and eye width are estimated using the Stateye tool.

4.6.4 Behavioral Link Analysis Tool

System-level simulation with statistical eye analysis method provides fast and accurate performance evaluation of the communication link. The link comprises of a driver specified here by IBIS input model, a receiver specified by an IBIS model file and an interconnect structure (lossless transmission line).

Link Analysis Method

Serial link analysis can be performed by the following methods: s-parameters based method or behavioral models based method. The focus of this work is on behavioral signal integrity analysis with IBIS models.

To perform link analysis, link specification (PCIe PHY) is incorporated into a template as discussed above. Jitter components specified are periodic jitter, duty-cycle distortion representing deterministic jitter, random jitter modeled with a Gaussian distribution. Built-in models for certain jitter types are available with (academic) commercial tools, but would have to be invoked appropriately by the user. This knowledge-based template for automated analysis has to be built by the user.

Based on user specification, the tool will automatically generate link netlist by selecting

a particular interconnect topology and estimate the eye height and eye width at the receiver of the link. The tool will output link netlists that closely meet the specification. This allows for fast evaluation of the link with available topologies and reduce the design cycle time. The IBIS-ISS, IBIS-AMI-ISS netlists are reusable in different design. The developed method consists of the following steps (pseudo code below):

Given: Characteristic impedance (Z_0)

Spec.: Delay time, Minimum eye height, Minimum eye width, BER $< 10^{-12}$.

1. Search and select interconnect models that meet characteristic impedance $Z_0 = 50\Omega$ requirement. Obtain the time delay (T_D) value of each interconnect
2. Compare delay value with specification and choose a particular link topology
3. Invoke Stateye analysis tool and perform serial link analysis
4. Obtain the output performance parameters
5. Compare results against spec. Meets requirements? Yes, end. Else, repeat with relaxed constraint

4.6.5 Implementation

The behavioral SI analysis tool has been implemented with Perl scripting/programming language. The modified IBIS models, generated by IBIS model generator, forms the input to the tool. A template with standard values of deterministic jitter (periodic and DCD) and random jitter based on PCIexpress specifications is created. HSPICE Stateye analysis is then automatically invoked for serial link analysis with generated IBIS models forming the driver and receiver. The contribution of this work is that it aids in faster system-level SI analysis compared with SPICE based simulations. Based on the parameters of the T-line, characteristic impedance Z_0 and delay time T_D are obtained. These parameters form the

input for the serial link topology selection procedure. For in most cases, a characteristic impedance of 50Ω is taken as a standard value.

Examples and Results

This section lists some examples and results demonstrating the automated link analysis method. Examples for different topologies are presented with tabulated results for both 22nm and 16nm process technology nodes. For each technology node, input specifications are Z_0, maximum delay T_D and the length (l) of the T-line. The tool takes these as input from the user, selects a list of interconnects that meet the requirements, and produces the estimated values of the eye height and eye width for the topologies that meet the criteria. The flow of the tool has been shown earlier in Figure 4.4. The main steps involved in the analysis method are: one, utilizing 't_r' and 't_f' obtained by the model generation tool to generate a more accurate PRBS at the driver end and two, selecting an interconnect model (transmission line) with appropriate parameters. These steps are repeated for each technology node.

For a given process technology, various T-line topologies are supported. Source terminated topology and load terminated topology are considered in each case and their effect on performance metrics is tabulated. As mentioned earlier, eye height and eye width are the outputs of the behavioral tool. The information obtained can be used by a system designer to evaluate serial link performance quickly and accurately. The developed tool can be integrated with the system specified at a higher abstraction level to aid early performance analysis.

For characteristic impedance Z_0 = 50Ω, and a maximum specified interconnect delay, the performance parameters obtained by the tool are tabulated below. Each table lists the results obtained for that process technology. Further description of the experimental work is discussed in the following paragraph.

For each process technology, a minimum value for the performance metric is set. The

Table 4.3: Results: 22nm

Link Length(mm)	Delay(ps)	Max. Eye Ht.(V)	Eye Wdt.(s)
1.0	1.2	0.289589	2.91218e-012
1.0	2.0	0.289589	2.91218e-012
1.0	3.0	0.290225	2.91218e-012,
2.5	1.2	0.289572	2.91218e-012
2.5	2.0	0.291133	2.91218e-012
2.5	3.0	0.292141	5.18848e-010
5.0	1.2	0.289544	2.91218e-012
5.0	2.0	0.289544	2.91218e-012
5.0	3.0	0.290668	4.89736e-010

Table 4.4: Results: 16nm

Link Length(mm)	Delay(ps)	Max. Eye Ht.(V)	Max. Eye Wdt.(V)
1.0	1.2	0.241635	7.63562e-012
1.0	2.0	0.241635	7.63562e-012
1.0	3.0	0.241635	7.63562e-012
2.5	1.2	0.24163	7.63562e-012
2.5	2.0	0.24163	7.63562e-012
2.5	3.0	0.24163	7.63562e-012
5.0	1.2	0.241622	7.63562e-012
5.0	2.0	0.241622	7.63562e-012
5.0	3.0	0.241622	7.63562e-012

tool will create an interconnect link with generated IBIS models, search and select T-lines that meet the specifications and perform behavioral SI analysis. From the tabulated results, it will compare with the performance requirement and provide the user with the link(s) that closely meets the requirement. With this information, a systems engineer can select appropriate serial link topologies that meet the specification. Another important advantage is that these topologies can be *reused* in different designs since the methodology is based on IBIS files that contains tabulated data instead of SPICE circuit description, preserving IP and therefore allowing reuse. This reduces the overall design cycle time. Figure 4.6 and Figure 4.7 show maximum eye height trend for a given process technology for various topologies.

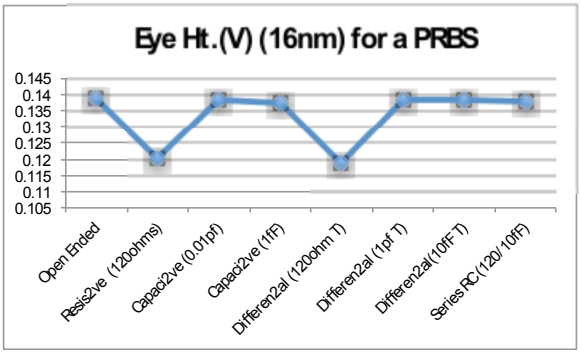

Fig. 4.6: Results for 16nm Process Technology for Various T-line Topologies

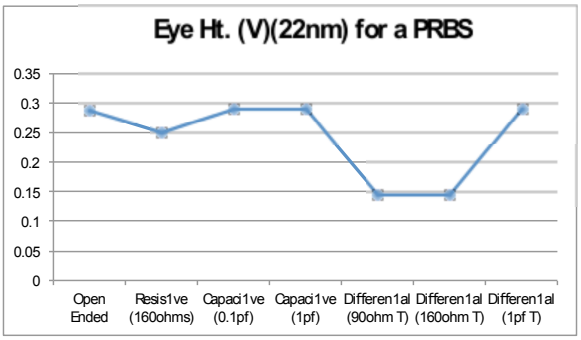

Fig. 4.7: Results for 22nm Process Technology for Various T-line Topologies

4.7 Summary

This chapter presented a tool for behavioral serial link analysis. Two new netlists are described, the IBIS-ISS netlist and IBIS-AMI-ISS netlist. The netlist consists of generated IBIS model files (Chapter 3) and an T-line based interconnect topology. To perform serial link analysis, the rise time and fall time information from generated IBIS model file is imported into a template with LFSR based PRBS generator. This template, the input to the tool, consists of a set of interconnect topologies and associated models. PCIe Gen3, Gen4 jitter spec. is incorporated. Based on the user specification, the tool selects the generated IBIS model as driver, an interconnect topology and receiver to create an IBIS-ISS netlist. Differential T-line topology is used as provides good noise immunity for a wide data rate range. If the spec. isn't met, IBIS-AMI-ISS netlist is invoked with appropriate equalizers and the netlist that best meets the requirements is selected. HSPICE Stateye is used. From

the results, it is observed that the eye height reduces with advancement in process tech. with values around 50mV for 16nm process tech. Equalizers are used to open up the eye at the receiver and pre-emphasize the signal at the transmitter end. The next chapter presents the early performance prediction methodology, where the system specification is mapped to the generated netlist to obtain a more accurate performance prediction at an early stage of design.

CHAPTER 5
EARLY PERFORMANCE PREDICTION METHODOLOGY

5.1 Introduction

This chapter describes the developed *early performance prediction* methodology. An accurate system performance analysis at an early stage of design has useful advantages. First, it improves overall design cycle time since a more accurate estimate is available, enhancing the prospects of first-time-correct design. Second, with this early insight, an engineer could modify the existing designs to further improve overall design performance. For a reasonably good performance prediction, selection of appropriate performance model(s) is important. Queueing theory based model is employed for this purpose as system performance is primarily a function of interconnect latency. This chapter describes a methodology to map the performance model obtained from a given abstract C-like specification to a lower level model to obtain a more accurate performance estimate at an early phase of design. This is the essence of early performance prediction.

The rest of this chapter is organized as follows: The next section describes C-like system description to specify systems at a higher-level of abstraction. Specifically, intercon-

nect spec. is discussed in some detail. Then, in Section 5.3, performance modeling at the system-level and performance estimation method at the circuit-level (PHY from interconnect perspective) is discussed. Section 5.4 presents the early performance predicition methodology for many-core systems. System specification in SystemVerilog is mapped to a generated circuit-level IBIS-ISS or IBIS-AMI-ISS netlist (Section 4.5 describes the estimation method) to obtain an improved performance prediction at an early design phase. The method is implemented using PERL.

5.2 System Modeling and Specification

Modeling and specification of modern systems at a higher abstraction level was discussed in Chapter 2. In this section, specific system-level modeling and specification constructs for this work are described. System-level spec. allows clear separation of computation processes and communication interfaces. This allows easier integration of multiple IPs, required to build these systems. First, computational process model applied is discussed. This is followed by specification and modeling of interconnects using TLMs along with the interconnect protocol.

5.2.1 System Modeling

Communication between two cores is point-to-point (pin-to-pin) connection with a particular interconnect network. This communication at the transaction layer is viewed as the producer-consumer model, with the description at the system-level. In this model, a producer generates the data (tokens) to be written. This data is transferred to another computer core through this interconnect, modeled as a (first-in-first-out (FIFO)) buffer. The consumer consumes tokens from the interconnect at a particular rate. The buffer is required for storing tokens when the production rate is greater than the consumption rate and for meeting the protocol requirements. Bounded Kahn Process Network (BKPN) model of

computation is well suited to represent this model. System-level HDLs such as SystemC and SystemVerilog currently have the capability to specify this abstract model.

A point-to-point link in a bounded process network consists of two processes connected by a finite size FIFO channel. Performance primarily depends on the interconnect buffer (FIFO for example) and the implemented communication protocol. The next section describes specifying and modeling the interconnect functionality using TLMs. The selected computation model determines the netlist at this abstaction layer. SystemVerilog HDL is directly referenced to specify the TLM, netlist.

5.2.2 Specifying Interconnects with TLMs

An overview of transaction level modeling developed to raise the abstraction level of interconnect specification to specify its behavior was presented in Chapter 2. In this section, TLM specification of the system to model interconnect performance with SystemVerilog `interface` (equivalent to SystemC `channels`) is described. System performance is specified by C-like *wait()* statements and are invoked in SystemVerilog and SystemC to specify timing information. In SystemVerilog, protocol timing spec. can be described as `tasks` or `functions` that encapsulate the interconnect functionality within the *interface* construct. Given this spec., the tool evaluates various link topologies by creating circuit-level IBIS-ISS or IBIS-AMI-ISS netlists and maps it to a selected netlist that meets the performance spec. PCIe protocol base spec. guidelines are followed at each abstraction level for this academic work.

SystemVerilog constructs and definitions pertinent to transaction-level modeling (TLM) are enumerated below for reference:

1. Interconnect: An interconnect module describes the communication behavior between two sub-systems. In TLM, this is described by a (set of) function calls. SystemVerilog `interface` construct specifies interconnects and encapsulates the func-

tionality.

2. Interconnect Protocol: Describes the rules of communication between two processors and can be encapsulated within the interconnect module with C-like functions. In SystemVerilog, `task` and `function` constructs allow description of interconnect behavior.

3. Netlist: A system-level netlist is created by connecting modules with the interconnect. Ports specify the entry and exit points for computing processes and the interconnect. In SystemVerilog, interconnects are described by `interface` and have module ports description incorporated along with the *direction* (input, output) with `modport` construct.

For analyzing performance, timing information is required and implied by the protocol. For example, consider a handshaking based communication between two processes. In this protocol, request (REQ) and acknowledge (ACK) signals are used for handshaking. The delay associated with each step in the protocol is explicitly encapsulated within a **wait()** statement. Performance is evaluated considering the total time taken to complete the requested transaction. Selecting a suitable buffer for interface leads to application of appropriate performance model(s). For this work, a queueing theory model is applied for performance analysis.

Combination of 2-phase and 4-phase approximately-timed (AT) protocol model is utilized to model the communication protocol. AT model given in TLM 2.0 spec. has an approximate notion of time with no clock cycle time information. The four phases required to complete a transaction are: Start (Begin-Req and End-Req), Transmit (Send-Data), Receive acknowledgement (ACK) or No ACK. Available spec. template [58] were initially modified and used. Specific coding styles pertaining to architects view has been employed. Functional view and architect's view coding guidelines are appropriate at the specification abstraction layer. The protocol is specified with SystemVerilog HDL (initially the work

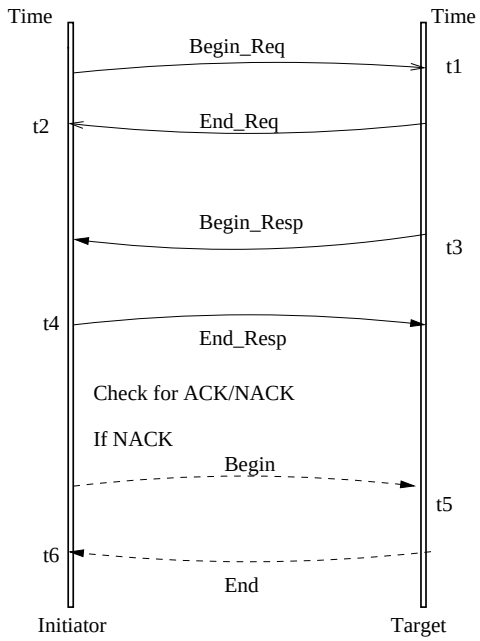

Fig. 5.1: Communication Protocol

involved SystemC. Newer SystemVerilog constructs are well suited for this work).

5.2.3 Protocol Timing Model

Acknowledgement/No-Acknowledgement (ACK/NAK) protocol [46] is implemented with a combination of four-phase and two-phase approximately timed TLM protocol model. The 2-phase model is required in addition to 4-phase for the case where no acknowledgment (NAK) is received, requiring data to be transmitted again to complete the transaction. The protocol model incorporating this is given in Figure 5.1. This specification is then integrated along with the initiator process node (the transmitter) and target process node (the receiver) specification described in C-like SystemVerilog description. Further details on TLM 2.0 specification and associated function calls is given in [58]. The following section describes the birth-death process queueing system model. This model has been employed for performance analysis for a given TLM description. To evaluate interconnect performance, the sevice time model is required.

5.3 Performance Spec. and Modeling at System Level

Performance analysis models, as opposed to functional models, propagate the delay associated with transmitting data rather than actual data values. Performance spec. would comprise of first specifying the time delay of data transmission and delay of various phases of the underlying protocol in addition to computation time. The system is modeled using a higher-level HDL such as SystemVerilog, SystemC (and/or other C-like languages). Then, the delay values are usually extracted and mapped to a selected performance model. Since the system performance is driven primarily by the interconnect fabric, the appropriate choice would require a combination of system-level model and lower-level (PHY) model.

To analyze performance at the physical layer, simulation-based approach is required to study the factors affecting signal integrity. Hybrid modeling approach involves a combination of analytical and simulation-based models. This approach has been employed for early performance estimation in this work. To estimate performance at the system-level, the analytical model employed is the birth-death queueing system model. This adequately models point-to-point (PCIe) links for performance analysis. At the circuit-level, since circuit step response (IBIS output) can be appropriately modeled by an exponential distribution function as shown in previous work [38], $M/M/1/K$ queueing model (bounded FIFO of size K) with exponential arrival time and service time is employed. The goal is to provide a better estimate of performance parameters by mapping to values extracted from circuit-level netlist.

5.3.1 Performance Specification

Specifying interconnect performance at the system-level is discussed in this section. C-like specification is primarily used to model system at the higher abstraction level. TLMs as described earlier, provides means to specify interconnect behavior at various abstraction levels. A brief survey of relevant system-level behavioral interconnect specification is de-

scribed, with SystemVerilog description language. Recommended basic coding guidelines followed are documented. As mentioned in Chapter 2, **wait ()** statement specifies the performance at this level. In SystemVerilog, `interface` construct encapsulates interconnect behavior, specified by `task` and `function` keywords that provide means to describe the intended functionality. A typical/general construct for specifying performance is given below [101].

```
interface intfc (); //interconnect
int fifo [$:1024]; //Bounded queue
modport in (import task read,. . . ); //netlist
modport out (import task write,. . .);
task automatic write ();//functionality
task automatic read ();
endinterface intfc;
module modname();
always
wait(condition);
unique if (condition) //transfer data;
else //protocol;
endmodule
```

The interconnect behavior is specified within each `function` or `task`. In the example above, interconnect is modeled as a FIFO using the SystemVerilog **queue** construct. Depending on the requirement, there are several ways to describe performance specification. In the explicit approach, delay value is described by an integer number. Implicit approach, on the other hand would involve including the protocol rule within the wait() statement. This leads to a model where timing annotations at different time points are required to be calculated to yield the final result. An example would be asynchronous communication with handshaking protocol including status signals. Example spec. (Ref: [105] for recent

work) is provided below. Here, performance information is extracted from the specification and applied to the selected mathematical model for performance analysis. Some examples with implicit performance spec. is given below. In these cases, the spec. is mapped to a TLM protocol described in Chapter 2 with timing annotations.

```
wait (condition1);
wait (condition1 or condition2);
```

For example, for the communication protocol shown in Figure 5.1, the condition in the statement above corresponds to the ACK signal.

```
wait (ACK);
```

For **parallel** data transfer over multiple lanes, *fork-join* construct is used in an `always` block with the `module` description. PCIe protocol timing parameters are adhered to with x2, x4, x8, x16 lanes. Each lane itself is specified with PCIe jitter parameters. The multi-phase TLM timing spec. is mapped to the performance spec. described above. If the performance functionality is specified in modules rather than interfaces, then `interface` construct incorporates it through the `export` keyword within the `modport` description.

To ensure reliability of data transfer and therefore system performance, conditions for **deterministic** data transfer between two cores/sub-systems are specified by the following rules:

1. Each port assigns to one and only one process at a given point in time
2. An interface instance connects to only two processes (producer, consumer).

These conditions are implemented in SystemVerilog with `unique` keyword along with conditional statements, `unique if`, `unique case` statements, within the `always` construct with/without sensitivity list for synchronous/asynchronous intercommunication [105]. An alternate approach is to specify the behavior as a `task` within the `interface` and invoke it from the SystemVerilog module.

Once the user has an accurate system specification that incorporates the aforementioned

constructs, this spec. forms the input to the early performance prediction tool. To model interconnect performance at this abstraction level, queueing theory based model is employed as discussed in the next section. The prediction method extracts the required timing information from the spec. and passes it on to the selected performance model. This will be discussed further in the implementation section.

5.3.2 Performance Modeling using Queueing Models

Point-to-point interconnect link is modeled with a bounded FIFO queueing system. The model can be analyzed as a birth-death process or using the mean-value analysis and fundamental rules namely Little's law, forced-flow law. Using Kendall's notation, point-to-point link is modeled as $M/M/1/K$ queue, with FIFO size of K units. Little's law is given by:

$$L = \lambda.W \tag{5.1}$$

Here,

- L: Average number of items in the FIFO.
- W: Average waiting time for item in queue.
- λ: Average number of units arriving per unit time.

Consider that the average value for arrival is represented by 'λ', average value for departure is 'μ' and utilization is given by 'ρ'. Link utilization is given by the utilization law as below:

$$\rho = T.S \tag{5.2}$$

Here,

- T: System throughput, the number of jobs that can be completed in a given time.

- S: Average service requirement, defined as the ratio of amount of time the queue is busy to the completion time.

The link latency (time spent) and throughput metrics can be estimated with the performance metrics for a $M/M/1/K$ queue, given by the following relations:

$$W_q = \frac{\rho}{\mu - \lambda}, L_q = \lambda.W_q \tag{5.3}$$

$$\rho = \frac{\lambda}{\mu} \tag{5.4}$$

$$P_B = \frac{(1-\rho).\rho^K}{1-\rho^{K+1}} \tag{5.5}$$

$$T = \rho.(1 - P_B) \tag{5.6}$$

$$W(\rho \neq 1) = \frac{1}{\lambda}\left(\frac{\rho}{1-\rho} - \frac{K+1}{1-\rho^{K+1}}.\rho^{K+1}\right) \tag{5.7}$$

$$W(\rho = 1) = \frac{K}{2.\lambda} \tag{5.8}$$

Here, W_q is the wait time in the queue, P_B is the blocking probability, 'λ' and 'μ' are also known as birth and death coefficients when the queue is viewed as a birth-death stochastic process. For evaluating interconnect performance, the **wait time** metric is of primary importance. M/M/1/K model is the most widely used model and appropriate for this work. Several good references provide detailed discussions on other queueing models that can be employed for link analysis.

5.4 Early Performance Prediction Methodology

An algorithm for early performance prediction is presented in this section. The goal is to provide a more accurate estimate for the given abstract performance specification. This would allow an architect or systems engineer to predict system performance more accurately at an early stage thereby reducing design cycle time.

5.4.1 System Architecture and Task Mapping

With increasing complexity of applications and with multiple cores integrated on a chip, mapping these application(s) to a set of cores to ensure perf. requirements are met is an important task. At the system-level, several recent works have attempted to address this issue for different specifications/requirements. A brief survey is presented with focus on methods relevant to this work. In general, graph-based approach is extensively employed in the proposed solutions. This is due to its ease of representation. Depending on the number of applications and number of available resources (computational cores), mapping is performed by partitioning the cores for each application to ensure performance (and power) requirements are met.

In [96], methods for mapping multiple applications onto generic many core chips is discussed. A 2D-mesh based and/or ring based on-chip interconnect topology is assumed and modeled as a tree. The applications are modeled as directed acyclic graphs (DAGs) and mapping algorithms are developed to map a DAG representation to the tree model, subject to constraints. The focus in this work is on performance (interconnects), therefore the constraint is the latency that can be specified by the edges of the DAG. The computational speed of a particular core is represented by the vertices of DAG.

The authors in [95] present a method to map a KPN specification onto a multi-processor chip. Two methods are presented; independent communication-aware mapping, where the core mapping is provided and system-mapping where processing elements and communication primitives are to be mapped. The KPN model trace DAG is scheduled and then mapped onto cores. Recent work summarized above derives from some earlier work that employed graph-based approach for mapping, due to its ease of representation, for example [98]. Readers interested in the state-of-the-art survey of mapping methods employed for wide range of applications are referred to [99].

In both mesh-based and ring-based topologies, a tree (special case of DAG) effectively models the topology. Starting from a core (the central one in case of the mesh topology)

that constitutes the root node of the tree, neighboring cores form the child nodes of the root node and adjoining cores form the subsequent child nodes to complete the tree structure. For this work, with latency constraints as metric, related tasks of the application are mapped considering the least distance metric (nearest neighbor). As many tasks of the application are mapped onto the same core (note that related tasks or instructions are assumed to be known).

Given a set of processing elements, and tasks, a method to map tasks to the least number of PEs with communication latency as constraint is given below. For each task:

1. Model the many core system and tasks as a DAG.
2. Create a list of computing cores with the same order as DAG.
3. From the task graph, identify subtasks.
4. Map a subtask to the first computing core to which no task is assigned.
5. With the topology in perspective, map the dependent tasks if any to the nearest computing core to minimize latency.
6. End if the mapping is within the latency bounds.

Next, transaction scheduling considering data-parallelism is discussed.

5.4.2 Slot-Based Scheduling for Data Transactions

Transaction scheduling considering data-level parallelism is applied in this work. Slot-based, contention free scheduling method, described in [92] is employed to improve performance by considering physical concurrency. Communication between cores and associated memory on many-core chip would be based on a specific interconnect topology such as 2D mesh and ring topology. Consider that ring topology has been employed for communication between cores (current implementations such as many integrated cores employ

ring-based topology since it requires fewer wires than other topologies). To schedule concurrent transactions, slot-based scheduling method is applied. Each transaction comprises of multiple unit transactions, where a unit transaction consists of a fixed number of atomic transactions. A slot is defined as the time taken to transmit a unit transaction. Fixed period transactions are considered here, i.e. the transaction is available for the amount of time specified by its period. The total time required to complete the transaction depends upon the core execution time transaction delay which depends upon the distance between the communicating cores. As described in [92], a data transaction is specified by a list T_i = $\{e_i, p_i, sp_{ti}, ep_{ti}\}$.

Here, 'e_i' is the execution time (expected time to complete the transaction), 'p_i' is the period of the transaction, 'sp_{ti}' is the start point of the transaction and 'ep_{ti}' is the end point of transaction 'i'. At any given time, for a particular interconnect segment, transaction overlaps are not allowed. Non-overlapping transactions are then scheduled concurrently. Consider two transactions T_1 and T_2, with start and end points being (sp_{t1}, ep_{t1}) and (sp_{t2}, ep_{t2}) respectively required to be transferred over the interconnect. These transactions are said to overlap and cannot be transferred concurrently if:

$sp_{t1} < ep_{t2}$ and $sp_{t2} < ep_{t1}$.

Transactions that do not meet the above criteria are scheduled concurrently. With this background, a simple slot-based communication-centric data-level scheduling methodology is implemented. The steps involved are given below. The method was implemented with PERL. For a given set of transactions:

1. Create a list of time slots.
2. Arrange transactions in ascending order of their first end points.
3. Obtain sets of transactions that overlap from rule provided above.
4. Map overlapping transactions onto consecutive time slots.
5. Map remaining transactions onto the same order of time slots.

Considering PCIe ordered set, a unit transaction would consist of 16 bytes. Thus, slot time can be calculated as the product of the size of the unit transaction and the bit unit interval, which is 125ps for PCIe Gen3. Each transaction therefore comprises of multiple unit transactions that could be scheduled concurrently.

5.4.3 Efficient Early Performance Prediction Method

Early performance prediction method is given below. This method utilizes hybrid modeling approach discussed in the previous section.

Input: SystemVerilog specification, Latency requirement

Output: IBIS-ISS, IBIS-AMI-ISS netlist

For a given process technology, do the following:

1. From system description, extract the functional and communication spec.
2. Functional spec., currently in transfer function form is mapped to the modified IBIS model file that contains generated transfer function.
3. Model each PTP link as single birth-death queueing system (M/M/1/K). Refine obtained delay value based on PCIe spec.
4. Invoke algorithm Sec.IV-B, perform behavioral SI analysis.
5. Invoke algorithm Sec.V-B and obtain the estimates of (λ, σ_1) and (μ, σ_2).
6. Obtain the efficient frontier with mean-variance analysis.
7. With each topology in efficient set, calculate system utilization (ρ).
8. For topology selection, do the following:
    - If $\rho \neq 1$, obtain Latency metric by calculating average wait time from Equation 5.7, end.

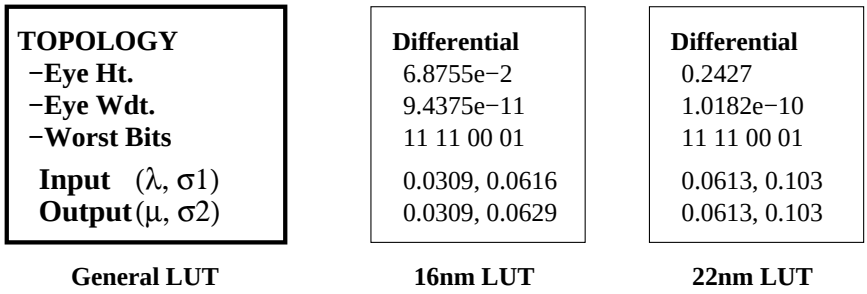

Fig. 5.2: Look Up Table: General Structure and Examples

- Else, obtain Latency metric by calculating average wait time from Equation 5.8.

9. Calculate throughput if required from Equation 5.7.

10. Compare calculated value with spec. Meets requirements? Output the final netlist. Else, repeat procedure with another topology.

5.5 Implementation

The methodology is implemented with Perl. Single and differential ended T-lines are used with appropriate signaling schemes. IBIS models provided by the user are updated with generated parameters by the IBIS model generation tool. Early performance prediction method begins by importing these parameters to a template comprising of a set of interconnect models, created for each process technology. PCIe Gen3 spec. for DJ (periodic and DCD) and RJ are included. Stateye analysis tool is automatically invoked for serial link analysis with this template. The LUT comprises of a list of interconnect topologies. Each topology has performance metrics tabulated as shown in Figure 5.2. Then, from the LUT, the tool will select a topology that meets the requirements by applying a selection procedure. The output file automatically generated is the required IBIS-ISS netlist. The generated IBIS models of CMOS inverter is used to demo the method. C-like system specification is mapped to an instance of IBIS-ISS netlist so that the performance requirements are met. Performance metrics are obtained with Stateye tool. The implemented tool takes

a few minutes to obtain the required netlist.

5.5.1 System-Level Template

SystemVerilog [6] based transaction-level model template is employed. Interconnects are specified using `interface` construct that contain tasks and functions to describe its functionality. Within them, `wait()` statement encapsulates the timing information that is specified either explicitly with an integer number or implicitly with a conditional statement, for example: `wait(25)`, `wait(condition1)`. Implicit timing is specified with a specific communication protocol. Here, PCIe ACK/NAK protocol is used to model the behavior, with each condition carrying a timing annotation.

5.5.2 Circuit-Level Template

IBIS-ISS and IBIS-AMI-ISS netlists are created with appropriate signaling schemes to study circuit-level interconnect performance at 22nm and 16nm. IBIS models provided by the user are updated with generated parameters by model generation tool. A template is created consisting of T-line topologies, equalizer models, a PRBS along with serial link (Stateye) analysis facility. The ports and interconnect are specified with PCIe Gen 3 Jitter parameters. With HSPICE Stateye, RJ [107] with peak-to-peak value of 1.6ps has been applied along with periodic jitter and DCD. During the early prediction procedure, the tool invokes the behavioral serial link template and generates an IBIS-ISS or IBIS-AMI-ISS netlist comprising of the driver, selected T-line topology, the receiver with equalizers to complete the point-to-point interconnect.

For IBIS-AMI based netlist simulation, AMI object provided with HSPICE is used. The template developed for Stateye based simulation is modified to incorporate algorithmic description of the equalizer at the transmitter end and receiver end. AMI parameter file consists of a set of rules to implement AMI object functionality according to the specified requirement (a choice of LTI based simulation versus the nonlinear simulation with AMI

Getwave function). Choice of LTI/non-LTI equalization along with tap coefficients can be specified. The goal is to improve receiver performance by opening the eye. In the event that the tool is unable to assemble an IBIS-ISS netlist that meets the spec., equalizers are employed (to improve performance) and to generate an IBIS-AMI-ISS netlist that closely meets the spec.

5.5.3 Tool Components

Early performance prediction method, implemented in PERL begins by extracting the `interface` information from the SystemVerilog spec. and records the necessary timing information. This data forms the input to the selected mathematical model. The mathematical model is then mapped to the generated circuit-level macromodel, the netlist. The practical limitation is the availability of interconnect models that can meet the requirement. PCIe Gen3 spec. for DJ and RJ are included. Stateye is automatically invoked for serial link analysis with the developed circuit-level template.

5.6 Summary

This chapter presented a methodology to predict performance of a modern ASIC based systems at an early stage of design process. The important advantages of the methodology are scalability in nanometer technology and reusability. The system was specified in SystemVerilog and modeled as a process network. Interconnect behavior was modeled with transaction level modeling paradigm. For early prediction, hybrid modeling approach was employed that consists of a combination of analytical model and simulation-based model. Bounded FIFO queueing system was applied as the analytical model for system-level analysis. To obtain an accurate prediction, simulation-based performance estimation methodology was employed to estimate the required parameters at the physical layer. System specification is then mapped to the generated parameters from IBIS-ISS or IBIS-AMI-ISS

netlist to obtain a better performance estimate at an early stage. In other words, the estimated values of the parameters of the hybrid model is obtained for early prediction.

For this application, the method itself comprises of multiple steps. A functional specification is translated to a predefined set of tasks (example: instructions) that are mapped to respective processing element(s). A schedule is formed. Then, with the interconnect TLM, the method takes a system-level netlist as input and produces the IBIS-ISS (or IBIS-AMI-ISS) netlist as the output. Currently, the generated CMOS inverter IBIS macro-models along with specific T-lines based interconnect at the circuit-level forms the IBIS-ISS netlist. With the IBIS-AMI object file, equalizer coefficients are specified to form the IBIS-AMI-ISS netlist. The behavioral system-level spec. in SystemVerilog is described with specific constructs, with snippets provided in Chapter 4.

The scalability of the method is demonstrated with example and results. The method has been implemented in Perl scripting and programming language. The goal of the tool is to aid an architect to design first-time-correct systems and reduce design cycle time. The developed method would be useful in many-core ASIC based design and development.

CHAPTER 6
CONCLUSION

6.1 Summary

As VLSI systems enter many-core era (with a thousand cores on a chip on the horizon), several research and development opportunities exists. Designing and implementing these ICs is a complex task accomplished by detailed multi-layer abstraction flow from a higher-level specification required to describe these systems to the final implementation. One important requirement is accurate prediction of system performance at an early stage of design. A method to accomplish this is called **early performance prediction**, the contribution of this work. As reasoned out in Chapter 2, the goal of early performance prediction is to reduce design time by enhancing the prospects of first time correct implementation. This is achieved by mapping a higher-level (system-level) spec. to a model generated at the circuit-level, thereby providing insights into circuit-level issues at an early phase of design. The following are the contributions of this work:

1. A tool to generate a macromodel of a circuit specified by IBIS models. Two methods are presented with one implemented. The tool is scalable in nanometer technology and generated models are reusable in different designs.

2. A tool for behavioral signal integrity analysis (HSPICE Stateye) of serial links.

    - A new IBIS-ISS netlist based on IBIS-ISS specification.
    - A new netlist IBIS-AMI-ISS, created to incorporate on-chip equalizers with IBIS-AMI specification. This would require enhancements to IBIS-ISS specification.

3. An early performance prediction methodology that provides a more accurate performance estimate of the system at early stage of design phase. The created IBIS-ISS, IBIS-AMI-ISS netlists are viewed as a new abstraction layer that provides the required performance information at a the early phase of design.

6.2 Conclusions

To predict performance more accurately at an early stage of design cycle, the developed methodology incorporates factors affecting signal integrity (SI) at the physical layer to better predict performance of these applications. In this tool, the system is described at system-level with SystemVerilog. Interface construct encapsulates the communication between processes. PCI express protocol is referenced for the communication between two cores due to its layered architecture. For early performance prediction, this specification is mapped to a new (generated) netlist (IBIS-ISS netlist, IBIS-AMI-ISS netlist) of the system at the circuit level specified by IBIS models for behavioral SI analysis. Some important conclusions drawn from this work are discussed below:

1. **Scalability** of the methodology is required due to worsening signal integrity issues in nanometer technology.

2. **Adaptability** is essential as due to limits of physics, new devices beyond CMOS are introduced. This is achieved by creating a new abstraction layer, based on behavioral

IBIS models in this work. Subsequent work could borrow from the spirit of this contribution.

3. **Ease of interoperability** due to this method would allow interaction of products from multiple vendors to provide the best solutions to the user.

6.2.1 Features of Early Performance Prediction

Specific advantages of the early performance prediction methodology with respect to conclusions drawn above are enumerated below:

1. Is **"open-source"** - interoperable, easy-to-integrate with other tools/vendors. Since the tool requires behavioral IBIS models as input, it can be employed in various designs without modifications, a very useful feature

2. Is **"long-term"** - independent of process technology, T-lines with tech. must be considered. Therefore, in future, changes in device technology (eg. shifting from CMOS technology to FinFET or Silicon-on-insulator (SOI)) would not impact the methodology, another important advantage.

APPENDIX A
EEPPM TOOL

Efficient Early Performance Prediction tool has been scripted using PERL. The inputs to the tool are the system specification and generated IBIS macromodels. The output is the IBIS-ISS or IBIS-AMI-ISS netlist that can be further analyzed. A snapshot of the scripted Early performance prediction tool is shown below.

There are two phases for the tool implementation. In the first phase, the tool takes SystemVerilog spec. as input and outputs scheduled transactions after mapping spec. to appropriate architecture. This data then forms the input to the selected mathematical model.

In the second phase, the system-level model is mapped to the generated macro-model at the circuit-level. This macro-model is an instance of the IBIS-ISS netlist or IBIS-AMI-ISS comprising of a driver, interconnect model following SPICE netlist and receiver. IBIS-AMI-ISS netlist would contain on-chip equalizer descriptions in IBIS-AMI format in addition to the IBIS-ISS netlist.

The system specified in SystemVerilog forms the input to the tool. An example specification is attached for reference. The output of the tool is IBIS-ISS netlist also provided for reference. Inverters driver and receiver models are used.

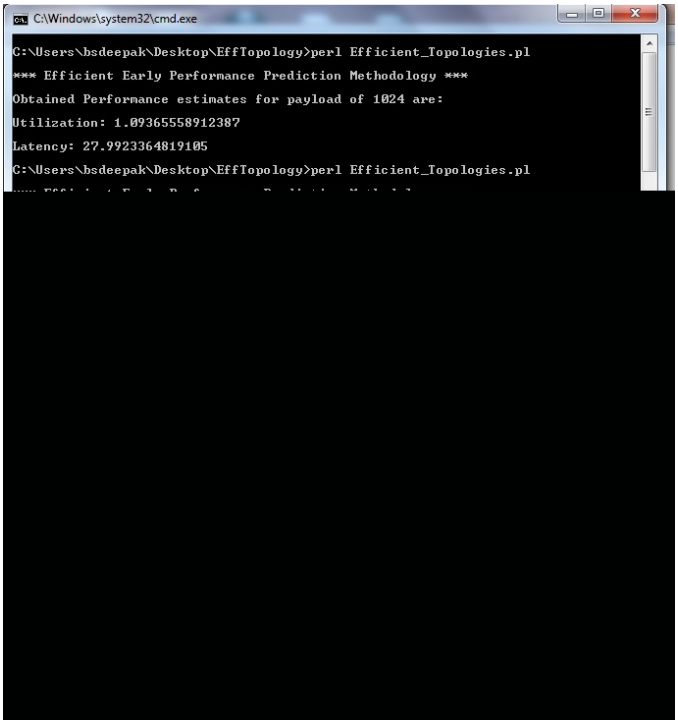

Fig. A.1: Snapshot of Early performance prediction tool

BIBLIOGRAPHY

[1] S. Borkar, A. Chien, The future of microprocessors. Commun. ACM 54, 5 (May 2011), 67-77.

[2] International Technology Roadmap for Semiconductors (ITRS) 2009, www.itrs.net

[3] ITRS, Interconnect Roadmap 2011, http://www.itrs.net

[4] International Technology Roadmap for Semiconductors (ITRS) 2011, Interconnect Roadmap, http://www.itrs.net

[5] I/O Buffer Information Specification (IBIS) , http://www.eigroup.org/ibis

[6] IEEE 1800 SystemVerilog Standard, IEEE 2012.

[7] EIA/ANSI IBIS Forum, I/O Buffer Information Specification, Version 5, 2008.

[8] EIA/ANSI IBIS Group , I/O Buffer Information Specification - Interconnect Subcircuit Specification, Version 1, 2011.

[9] S2IBIS3 tool, http://www.ece.ncsu.edu/erl/ibis/s2ibis3/s2ibis3.htm

[10] M. Mirmak, IBIS Modeling Cookbook, Government Electronics and Information Technology Association and The IBIS Open Forum, 2005.

[11] A. Verma, P. Franzon, Spice to IBIS tool (s2ibis3): http://www.ece.ncsu.edu/erl/ibis/s2ibis3/s2ibis3.htm

[12] A. Verma, A. Glaser, S. Lipa, M. Steer, P. Franzon, The development of a Macro-modeling tool to develop IBIS models, IEEE 2003.

[13] W. Zhao, Y. Cao, New generation of Predictive Technology Model for sub-45nm early design exploration, IEEE Transactions on Electron Devices, vol. 53, no. 11, pp. 2816-2823, November 2006.

[14] A. Heinecke, M. Klemm, and H.-J. Bungartz, GPGPU to Many-Core: Nvidia Fermi and Intel Many Integrated Core architecture, Computing in Science and Engineering, 14(2), Mar. 2012.

[15] IBM Blue Gene team, Design of the IBM Blue Gene/Q Compute chip, IBM Journal of Research and Development , vol.57, no.1/2, pp.1:1-1:13, Jan.-March 2013

[16] IEEE 1666 Standard, SystemC 2.3, www.accellera.org

[17] Open SystemC Initiative (OSCI), IEEE 1666 Standard - SystemC, Download from http://www.systemc.org

[18] IEEE 1800 Standard - SystemVerilog, Download from http://www.accellera.org

[19] S. Bogatin, Signal and Power Integrity, Prentice-Hall, 2010.

[20] F. Liu, C. Kashyap and C. J. Alpert, A Delay Metric for RC Circuits based on the Weibull Distribution, Proceedings of IEEE International Conference on Computer Aided Design (ICCAD), 2002.

[21] F. Liu, C. Kashyap and C. J. Alpert, Delay and Slew Metric Using the Lognormal Distribution, Proceedings of Design Automation Conference, 2003.

[22] C. S. Amin, F. Dartu, Y.I. Ismail, Weibull Based Analytical Waveform Model, Proceedings of IEEE International Conference on Computer Aided Design (ICCAD), 2003.

[23] H. Ito, J. Inoue, S. Gomi, H. Sugita, K. Okada, and K. Masu, On-Chip Transmission Line for Long Global Interconnects, IEEE 2004.

[24] A. Agarwal, R. Vemuri, Hierarchical performance macromodels of feasible regions for synthesis of analog and RF circuits, Proceedings of IEEE/ACM International Conference on Computer-Aided Design, 2005.

[25] R. Harjani et al., Feasibility and Performance Region Modeling of Analog and Digital Circuits, Analog Integrated Circuits and Signal Processing- special issue on Macromodeling, vol. 10, no. 1-2, pp. 23Ű43, 1996.

[26] R. Trihy, R. Rohrer, Nonlinear Macromodeling with AWE, Proceedings of Analog Integrated Circuits and Signal Processing, 6, 265-274 (1994)

[27] A. Deutsch, G. V. Kopcsay, P. Restle, G. Katopis, W. D. Becker, H. Smith, P. W. Coteus, C. W. Surovic, B. J. Rubin, R. P. Dunne, T. Gallo, K. A. Jenkins, L. M. Terman, R. H. Dennard, G. A. Sai-Halasz, and D. R. Knebel, "When are transmission-line effects important for on-chip interconnections", Proc. 47th Electronic Components Technol. Conf., pp.704 -712 1997.

[28] L.T. Pillage, R.A. Rohrer, Asymptotic waveform evaluation for timing analysis, IEEE Transactions on Computer-Aided Design for Integrated Circuits, April 1990.

[29] B. Tutuianu, F. Dartu, and L. Pileggi, An explicit RC-circuit delay approximation based on the first three moments of the impulse response, Proc. IEEE/ACM Design Automation Conf., 1996, pp. 611Ű616.

[30] J. Rubenstein, P. Penfield and M. A. Horowitz, Signal Delay in RC Tree Networks, IEEE Trans. on CAD, July 1983.

[31] W. Zhao, Y. Cao, The Elmore delay as a bound for RC trees with generalized input signals, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 16, NO. 1, JANUARY 1997.

[32] R. Kay, L. Pillege, PRIMO - Probability Interpretation of Moments for Delay Calculation, Proceedings of Design Automation Conference 1998.

[33] R. Yin, L. Pillage, h-gamma: An RC Delay Metric Based on a Gamma Distribution Approximation of the Homogeneous Response, proceedings of ICCAD, 1998

[34] Jun-Kuei Zeng, and Chung-Ping Chen, Interconnect delay and slew metrics using the beta distribution, Proceedings of Design Automation and Test in Europe (DATE) 2010, pp. 1329-1332.

[35] W.C. Elmore, The Transient Response of Damped Linear Network with Particular Regard to Wideband Amplifiers, Journal of Applied Physics, Volume 1, Jan 1948.

[36] J. Sun, Y. Zheng, Q. Ye, T. Ye, Interconnect delay and slew metrics using the first three moments,Proceedings of the Sixth International Symposium on Quality Electronic Design (ISQED) 2005.

[37] C. V. Kashyap, C. J. Alpert, F. Liu, and A. Devgan. PERI: A technique for extending delay and slew metrics to ramp inputs. TAU, pages 57Ű62, Dec. 2002.

[38] B. Deepaksubramanyan, C.Y.R. Chen, A. Nunez, A Tool to Generate Models Based on Behavioral IBIS Models, Proceedings of 55th IEEE Midwest Symposium on Circuits and Systems (MWCAS), August 2012.

[39] PCI Express Architecture, PCI Express Jitter Budget and BER 1.0, PCI-SIG 2005.

[40] Intel PCI Express, PCIe Specification, http://www.pcisig.com/

[41] PCI Express SIG, PCI Express Base Specification PCIExpress3, http://www.pcisig.com/specifications/pciexpress/base3/

[42] PCI Express SIG, PCI Express Base Specification 2, http://www.pcisig.com/specifications/pciexpress/base2/

[43] D. Rennie, Designing to the new PCI express 3.0 equalization requirements, Synopsys, PCI SIG 2012.

[44] S. K. Lee, S. H. Lee, D. Sylvester, D. Blaauw, J. Y. Sim, A 95fJ/b current-mode transceiver for 10mm on-chip interconnect, IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC),(pp. 262-263), February 2013.

[45] PCI Express Architecture, PCI Express Jitter Budget and BER 1.0, PCI-SIG 2005.

[46] Mindshare Inc, PCI Express System Architecture, Addison-Wesley Develpers Press, 2003.

[47] F. Farelo, A. Kaw, Linear and Nonlinear Regression code snippets in Matlab, Univ. of South Florida, 2007.

[48] Stephan Schulz, J. Becker, Thomas Uhle, Karsten Einwich, and S. Sonntag, Transmitting TLM transactions over analogue wire models, In Proceedings of the Conference on Design, Automation and Test in Europe (DATE '10), European Design and Automation Association, 3001 Leuven, Belgium, Belgium, 1608-1613.

[49] Kuo A, Farahmand T, Ou N, Tabatabaei S, Ivanov A , Jitter models and measurement methods for high-speed serial interconnects, Test Conference, 2004 Proceedings, pp 1295- 1302, Oct 2004.

[50] Kossel M.A., Schmatz M.L. , Jitter measurements of high-speed serial links, Design and Test of Computers, IEEE , vol.21, no.6, pp. 536- 543, Nov.-Dec. 2004

[51] Markus Damm, Jan Haase, Christoph Grimm, Fernando Herrera, and Eugenio Villar, Bridging MoCs in SystemC specifications of heterogeneous systems, EURASIP J. Embedded Syst. 2008, Article 7 (January 2008), 16 pages.

[52] SystemC-AMS, SystemC Analog/Mixed Signal Extensions, http://www.systemcams.org

[53] HSPICE, HSPICE Signal Integrity Analysis Reference, 2011 http://hspice.com

[54] Fernando Herrera and Eugenio Villar, A framework for heterogeneous specification and design of electronic embedded systems in SystemC, ACM Trans. Des. Autom. Electron. Syst. 12, 3, Article 22 (May 2008), 31 pages.

[55] I. Getreu and D. Teegarden, An introduction to behavioral modeling, Microelectronics J., vol. 24, no. 7, pp. 708Ű716, 1993.

[56] Reinaldo Bergamaschi, Indira Nair, Gero Dittmann, Hiren Patel, Geert Janssen, Nagu Dhanwada, Alper Buyuktosunoglu, Emrah Acar, Gi-Joon Nam, Dorothy Kucar, Pradip Bose, John Darringer, and Guoling Han. 2007. Performance modeling for early analysis of multi-core systems. In Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis (CODES+ISSS '07). ACM, New York, NY, USA, 209-214.

[57] Yulei Zhang, Xiang Hu, Alina Deutsch, A. Ege Engin, James F. Buckwalter, and Chung-Kuan Cheng. 2009. Prediction of high-performance on-chip global interconnection. In Proceedings of the 11th international workshop on System level interconnect prediction (SLIP '09). ACM, New York, NY, USA, 61-68.

[58] OSCI SystemC Transaction Level Modeling (SystemC TLM 2.0), http://www.accellera.org/activities/committees/systemc-tlm, 2011.

[59] OSCI SystemC Analog/Mixed Signal Specification (SystemC-AMS 2.0, Draft), http://www.accellera.org/activities/committees/systemc-ams, March 2012

[60] Martin Barnasconi, SystemC-AMS Extensions: Solving the Need for Speed, Design Automation Conference, Knowledge Center Article 2010.

[61] OSCI SystemC-AMS, SystemC-AMS User Guide, Accelera Group, 2010

[62] Laurent Maillet-Contoz. 2010. Standards for system level design. In Proceedings of the International Conference on Computer-Aided Design (ICCAD '10). IEEE Press, Piscataway, NJ, USA, 332-335.

[63] F. Herrera, E. Villar, C. Grimm, M. Damm, and J. Haase, A general approach to the interoperability of HetSC and SystemC-AMS, in Proceedings of the Forum on Design Languages (FDL Š07), Barcelona, Spain, September 2007.

[64] W. Lihua, Design and Simulation of PCI Express Transaction Layer, Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on , vol., no., pp.1-4, 11-13 Dec. 2009

[65] IBM Research, Early Analysis Tools, http://domino.research.ibm.com/comm/ research.nsf/pages/r.da.slate.html

[66] Yulei Zhang, Xiang Hu , Deutsch, A., Engin, A.E., Buckwalter, J.F. , Chung-Kuan Cheng, Prediction and Comparison of High-Performance On-Chip Global Interconnection, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on Volume: 19 , Issue: 7, 1154 - 1166, July 2011

[67] R. Ho, K. Mai, and M. Horowitz, The Future of Wires, Proc. IEEE, vol. 89, no. 4, pp. 490-504, Apr., 2001.

[68] R. Ho , T. Ono , F. Liu , R. Hopkins , A. Chow , J. Schauer and R. Drost High-speed and low-energy capacitively-driven on-chip wires, IEEE Solid-State Circuits Conference, pp.412 -413 2007.

[69] Ou N, Farahmand T, Kuo A, Tabatabaei S, Ivanov A, Jitter models for the design and test of Gbps-speed serial interconnects, Design and Test of Computers, IEEE , vol.21, no.4, pp. 302- 313, July-Aug. 2004

[70] A. Tsuchiya et.al, Design guideline for resistive termination of on-chip high-speed interconnects, IEEE CICC, 2005.

[71] H. Ito et.al., On-Chip Transmission Line for Long Global Interconnects, IEEE 2004.

[72] A. Tsuchiya et.al, Performance Limitation of On-Chip Global Interconnects for High Speed Signaling, IEEE CICC 2004.

[73] L. W. Nagel, SPICE2: A Computer Program to Simulate Semiconductor Circuits, UC Berkeley Tech Report UCB/ERL M520, 1975.

[74] C. A. Thompson, A study of numerical integration techniques for use in the companion circuit method of transient analysis, ECE Technical Reports. Paper 297, Purdue University, 1992.

[75] A. Kuo et.al., Jitter models and measurement methods for high-speed serial interconnects, IEEE Test Conference, Oct 2004.

[76] Ning Dong, J. Roychowdhury, General-Purpose Nonlinear Model-Order Reduction Using Piecewise-Polynomial Representations, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.27, no.2, pp.249-264, Feb. 2008

[77] Jaijeet Roychowdhury, "Numerical Simulation and modeling of Electronic and Biochemical Systems", Foundations and Trendső in Electronic Design Automation: Vol. 3: No 2-3, pp 97-303, 2009.

[78] Michal Rewienski and Jacob White, A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. In Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design (ICCAD '01). IEEE Press, Piscataway, NJ, USA, 252-257.

[79] Michal Rewienski, Jacob K. White: A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. IEEE Trans. on CAD of Integrated Circuits and Systems 22(2): 155-170 (2003)

[80] Ning Dong and Jaijeet Roychowdhury. 2003. Piecewise polynomial nonlinear model reduction. In Proceedings of the 40th annual Design Automation Conference (DAC '03). ACM, New York, NY, USA, 484-489.

[81] Ayman I. Kayssi, Karem A. Sakallah, and Timothy M. Burks, Analytical Transient Response of CMOS Inverters, IEEE Transactions on Circuits and Systems I, Fundamental Theory and Applications, Vol 39, No. 1, Jan 1992.

[82] J. Katzenelson, An Algorithm for Solving Nonlinear Resistive Network, The Bell System Technical Journal, October 1965.

[83] T. Fujisawa, E. S. Kuh, Piecewise-Linear Theory of Nonlinear Networks, SIAM Journal on Applied Mathematics, Vol. 22, No. 2 (Mar., 1972), pp. 307-328.

[84] W.J. McCalla, D.O. Pederson, Elements of Computer-Aided Circuit Analysis, IEEE Transactions on Circuit Theory, Vol. CT-18, No. 1, 1971.

[85] Synopsys HSPICE, StatEye Analysis, http://www.hspice.com, March 2013.

[86] Texas Instruments IBIS Models, http://www.ti.com/adc/docs/

[87] B. Halak, A. Yakovlev, Statistical analysis of crosstalk-induced errors for on-chip interconnects, IET Comput. Digit. Tech., Vol. 5, Iss. 2, 104Ű112, 2011.

[88] Wei-Yu Chen, S.K. Gupta, M.A. Breuer, Analytical models for crosstalk excitation and propagation in VLSI circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.21, no.10, 1117- 1131, Oct 2002.

[89] Y. Zhang et. al., Prediction and Comparison of High-Performance On-Chip Global Interconnection, IEEE Transactions on VLSI Systems, July 2011.

[90] E. Lazowska, J. Zahorjan, G. Graham, K. Sevcik, Quantitative System Performance: Computer System Analysis Using Queueing Network Models, Prentice Hall, 1984.

[91] Princeton University, ECO525 Class Notes on Mean Variance Analysis, http://www.princeton.edu/ markus/teaching/Eco525/.

[92] B. D. Bui, R. Pellizzoni, M. Caccamo, "Real-Time Scheduling of Concurrent Transactions in Multidomain Ring Buses," IEEE Transactions on Computers, vol. 61, no. 9, pp. 1311-1324, Sept., 2012

[93] H. Yviquel, E. Casseau, M. Wipliez, M. Raulet, "Efficient multicore scheduling of dataflow process networks," Signal Processing Systems (SiPS), 2011 IEEE Workshop on , vol., no., pp.198,203, 4-7 Oct. 2011

[94] Shekhar Borkar, Thousand core chips: a technology perspective, In Proceedings of the 44th annual Design Automation Conference (DAC '07), ACM, New York, NY, USA, 746-749.

[95] J. Castrillon, A. Tretter, R. Leupers, and G. Ascheid. 2012. Communication-aware mapping of KPN applications onto heterogeneous MPSoCs, Proceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 1266-1271.

[96] B. Yang, L. Guang, T. Säntti, J. Plosila, Mapping multiple applications with unbounded and bounded number of cores on many-core networks-on-chip, Microprocessors and Microsystems, Volume 37, Issues 4-5, June-July 2013, Pages 460-471.

[97] Peh L. S., Keckler S. W., Vangal S. , On-Chip Networks for Multicore Systems, In Multicore Processors and Systems (pp. 35-71) Springer, 2009.

[98] A. Kumar Singh, T. Srikanthan, A. Kumar, and W. Jigang, Communication-aware heuristics for run-time task mapping on NoC-based MPSoC platforms. J. Syst. Archit. 56, 7 (July 2010), 242-255.

[99] A. Kumar Singh, M. Shafique, A. Kumar, and J. Henkel, Mapping on multi/many-core systems: survey of current and emerging trends. In Proceedings of the 50th Annual Design Automation Conference (DAC '13)., ACM, New York, USA, 2013.

[100] J. Castrillon, R. Leupers, G. Ascheid, Maps: Mapping concurrent dataflow applications to heterogeneous MPSoCs, IEEE Transactions on Industrial Informatics, Vol. 9, No. 1, 2013.

[101] J. Bromley, Towards a Practical Design Methodology with SystemVerilog Interfaces and Modports, Design and Verification (DVCon) Conference, 2007.

[102] J. Bromley, Seamless Refinement from Transaction Level to RTL Using SystemVerilog Interfaces, SNUG, 2008.

[103] S. Sutherland et al, SystemVerilog for Design: A Guide to Using SystemVerilog for Hardware Design and Modeling, 2nd Edition, July 2006.

[104] S. Sutherland, What, If Anything, In SystemVerilog Will Help Me With FPGA-based Designs?, Design Conference (DesignCon12), 2012.

[105] M. Renaudin, A. Fonkoua, . Tiempo Asynchronous Circuits System Verilog Modeling Language. In Asynchronous Circuits and Systems (ASYNC), 2012 18th IEEE International Symposium on (pp. 105-112), 2012.

[106] W. Stallings, Queueing Analysis, www.williamstallings.com/studentsupport.htm

[107] Synopsys HSPICE, http://www.hspice.com, 2011.

[108] James Lee, Begining Perl, Apress, 2010.

[109] T. Christiansen, and N. Torkington, Perl Cookbook, O'Reilly Media, 2003.

[110] Wolfram Alpha, http://www.wolframalpha.com

VITA

NAME OF AUTHOR: Boray S. Deepaksubramanyan

PLACE OF BIRTH: Bangalore (now Bengaluru), Karnataka, India

DATE OF BIRTH: July 17, 1981

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

Syracuse University, New York, U.S.A.

University of Pune, Maharashtra, India

DEGREES AWARDED:

Bachelor of Engineering in Industrial Electronics, 2003, University of Pune, India

Master of Science in Electrical Engineering, 2006, Syracuse University, U.S.A.

PROFESSIONAL EXPERIENCE:

- Graduate Teaching Assistant at Syracuse Univesity with Prof. S.K. Chin, Prof. Q. Qiu, Prof. K.J. Deris, Prof. E. Ercanli (2006-2012). Courses include CSE 664-VLSI Design Methods, CSE 464-Introduction to VLSI Design, CSE 261-Digital Logic Design, CSE 381-Computer Architecture.

- Grader/Lab Assistant at Syracuse University with Prof. N. Mansouri, Prof. A. Nunez-Aldana. CSE 464-Introduction to VLSI Design, CSE 471, 671 - Embedded System Design.
- Graduate Engineering Project: SystemC implementation of IEEE Clock Synchronization Protocol, Lockheed Martin Corporation (2006-2007). Principal Investigator - Dr. Adrian Nunez. Related courses Principal Investigators - Dr. Fred Schlereth, Dr. Duane Marcy.