MACROMODELING OF NONLINEAR DRIVER AND RECEIVER CIRCUITS

A Thesis
Presented to
The Academic Faculty

by

Bhyrav Mutnury

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2005

MACROMODELING OF NONLINEAR DRIVER AND RECEIVER CIRCUITS

Approved by:

Dr. Madhavan Swaminathan (Advisor)
Professor, Department of ECE
Georgia Institute of Technology

Dr. Abhijit Chatterjee
Professor, Department of ECE
Georgia Institute of Technology

Dr. Sung-Kyu Lim
Asst. Professor, Department of ECE
Georgia Institute of Technology

Dr. David C. Keezer
Professor, Department of ECE
Georgia Institute of Technology

Dr. Suresh K. Sitaraman
Professor, Department of ME
Georgia Institute of Technology

Date Approved: 08-03-2005

ACKNOWLEDGEMENTS

First, I want to thank my advisor, Professor Madhavan Swaminathan, for his guidance and support during my graduate studies. He is an outstanding scientist, mentor, and a tremendous source of motivation. I will always be grateful for his valuable advice and insight. I would also like to extend my gratitude to the Ph.D. committee: Professor Abhijit Chatterjee, Professor David C. Keezer, Professor Sung Kyu Lim, and Prof. Suresh K. Sitaraman. I appreciate their time and effort in serving on my committee. I extend special thanks to all current and graduated members of the research group. Your friendship, assistance, and opinions will always be appreciated. I would especially like to mention Nanju Na, Sungjun Chun, Sung-Hwan Min, Vinu Govind, Woopoung Kim, Erdem Matoglu, Jinwoo Choi, Jinseong Choi, Sidharth Dalmia, Jifeng Mao, Prathap Muthana, Rohan Mandrekar, Amit Bavisi, Tae Hong Kim, Wansuk Yun, Raghavan Madhavan, Di Qian, Joongho Kim, Krishna Srinivasan, Subramanian Natarajan Lalgudi, Souvik Mukherjee, Krishna Bharat and Lixi Wang. I would like to thank Moises Cases, Nam Pham, and Daniel de Araujo from IBM, Austin, for their support and encouragement throughout my PhD. I would like to thank James Libous at IBM for his support and encouragement for my studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The signal integrity, power integrity, and timing analysis of today's high-speed digital systems are computationally exhaustive, both in terms of CPU memory required and simulation time consumed. One way to reduce this complexity is to use macromodels of the sub-circuits comprising these high-speed digital systems. Since digital driver/receiver circuits have a major share in this computational load, modeling digital driver/receiver circuits accurately to capture their nonlinearity becomes a big challenge. The contribution of this thesis is to generate black-box macromodels of driver/receiver circuits that result in huge computational speed-up compared to actual transistor-level driver/receiver circuits and at the same time maintain high accuracy. It is always useful to have a black-box modeling approach as the modeling technique is independent of the knowledge of the internal logic of the circuit being modeled. This would make the modeling approach more robust and more applicable to a wide variety of circuits. Driver/receiver macromodels have been extended to multiple ports to take into account the effect of non-ideal power and ground nodes in this thesis.

CHAPTER I

INTRODUCTION

The rapid advance in semiconductor technology is pushing high-performance electronic systems toward higher operating frequency, higher power dissipation, and lower supply voltage, which pose tremendous challenges for designers. It can be seen from Figure 1.1 that the operating frequency for Intel microprocessors has been doubling almost every two years (Moore's Law).

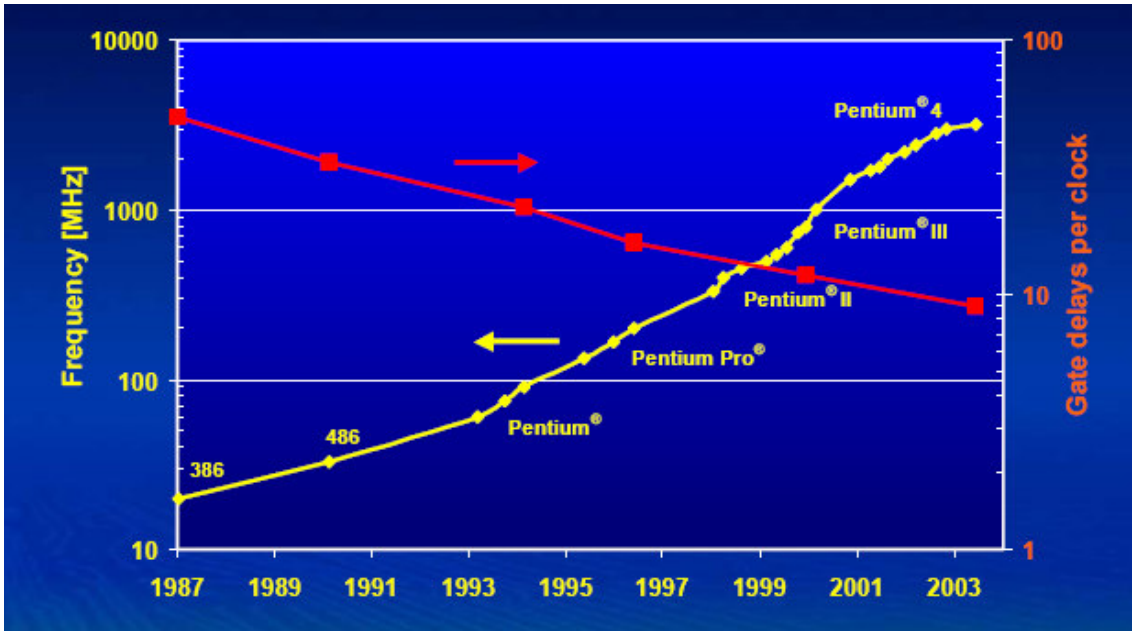


Figure 1.1 Operating frequency trends in Intel microprocessors.

With the operating frequency increasing, parasitic effects that were previously ignored cannot be overlooked anymore for accurate system level analysis. Figure 1.2 shows how the power supply voltage for microprocessors has been decreasing with each

generation of processors. The decrease in power supply voltage and increase in operating frequency and power consumption has left little room for error in modeling today's high-speed systems.

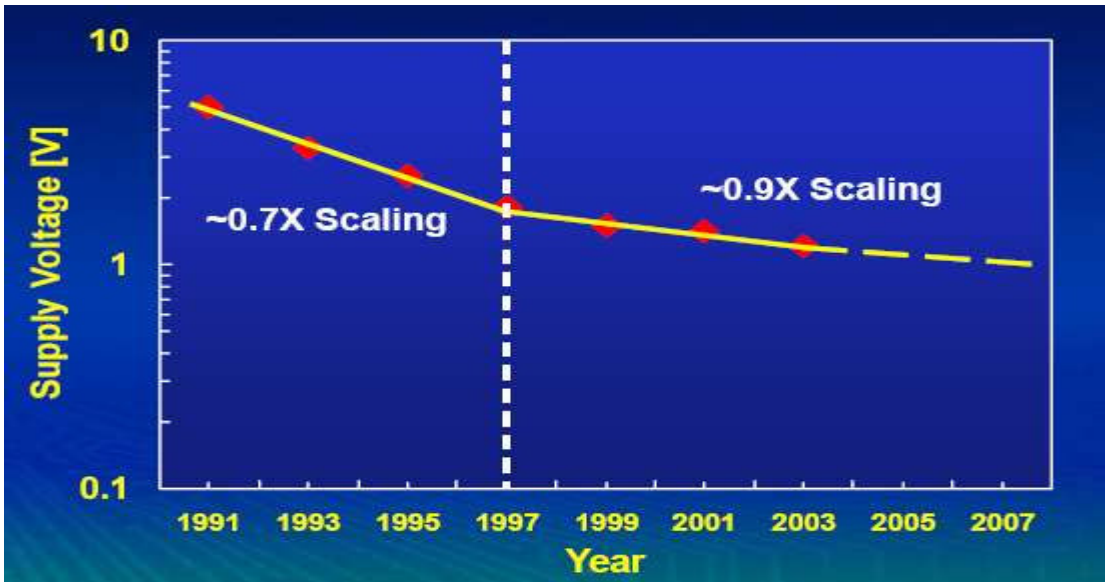


Figure 1.2 Power supply voltage trends in microprocessors.

The number of failures caused by signal and power integrity problems is on the rise because existing design tools and modeling methodologies cannot address these issues efficiently. Signal integrity, power integrity, and timing analysis of high-speed digital systems are becoming more and more complex, both in terms of CPU memory requirement and simulation time consumed. Analyzing signal and power integrity problems is important for meeting the design specifications. One way to reduce the complexity of the problem is by using macromodels of the sub-circuits comprising these high-speed digital systems. Since digital driver/receiver circuits play an important role in the signal integrity analysis of these high-speed digital systems, modeling digital driver/receiver circuits to capture their nonlinearity accurately is a big challenge. The

focus of this thesis is on the nonlinear macromodeling of driver and receiver circuits for efficient signal and power integrity analysis.

1.1 Macromodeling

Macromodeling of a circuit involves producing a reduced order model or behavioral model with the original circuit's input and output ports such that the macromodel runs faster than the original circuit while accurately modeling the actual circuit. Macromodeling can be broadly classified into (1) passive or linear macromodeling and (2) active or nonlinear macromodeling. There has been an increasing demand for integrating the electromagnetic behavior of passive structures into conventional computer-aided design (CAD) tools so that designers can take into account the electromagnetic effects during the design and analysis of multi-GHz electronic systems. In the past, a lot of work was done on the model order reduction of integrated chip (IC) interconnects and modeling passive circuits [A1]-[A2].The macromodel can be constructed using two methods. One method is to construct the macromodel from the moments that are the characteristics of the circuit. In [A3]-[A6], explicit or implicit moment-matching techniques have been used to construct the macromodel by generating and matching the moments using Padé approximation. The other method is to capture the frequency-dependent data using a macromodel after extracting the port behavior of the circuit either from an electromagnetic simulator or from measurements. In [A7]-[A11], the macromodel has been constructed by capturing measured or simulated frequency data using least squares approximation and vector-fitting.

Compared to work done on model order reduction for linear time invariant (LTI) resistance-inductance-capacitance networks, the problem of nonlinear macromodeling is less explored [A12]-[A15]. It is always useful to have modeling techniques that are black-box in nature, in which a circuit can be modeled independently of the knowledge of its internal logic [A16]. This would make the modeling approach more robust and more applicable to a wide variety of circuits. Figure 1.3 shows a black-box representation of an N-port device.

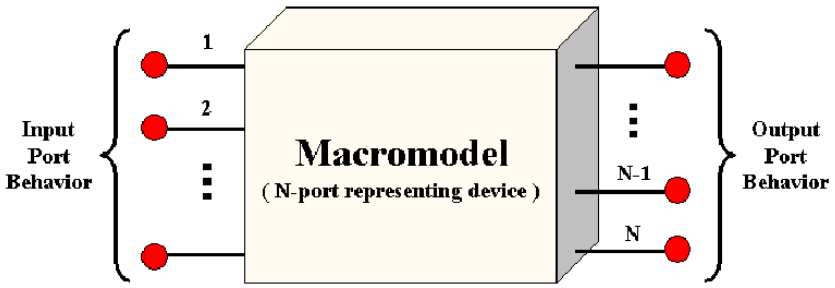


Figure 1.3 Black-box macromodel of an N-port device.

Nonlinear macromodeling is a new field of research. In the past, work has been done on modeling weakly nonlinear analog and RF circuits using Volterra series [A17]-[A18]. More recently, there has been some work on macromodeling digital aggressors for power and ground supply noise prediction. In [A19], independent ideal current sources have been utilized in predicting the simultaneous switching noise (SSN) effects and capacitor controlled ideal switches have been utilized to imitate the switching behavior of a digital cell in [A20]-[A21]. These methods result in an approximate prediction of the SSN. In [A22], linear time-varying (LTV) abstractions have been used to capture the aspects of digital switching nonlinearity. All the above mentioned modeling techniques model a

digital cell block in a mixed signal environment to capture the power and ground noise effects. The effect of power and ground noise on the driver output and receiver input signal and vice-versa is not taken into account. Scalability and extension of the above modeling approaches to multiple ports is a complicated procedure. The focus of this thesis is on the macromodeling of digital driver and receiver circuits for the generation of black-box models, as shown in Figure 1.4, for both efficient signal and power integrity analysis.

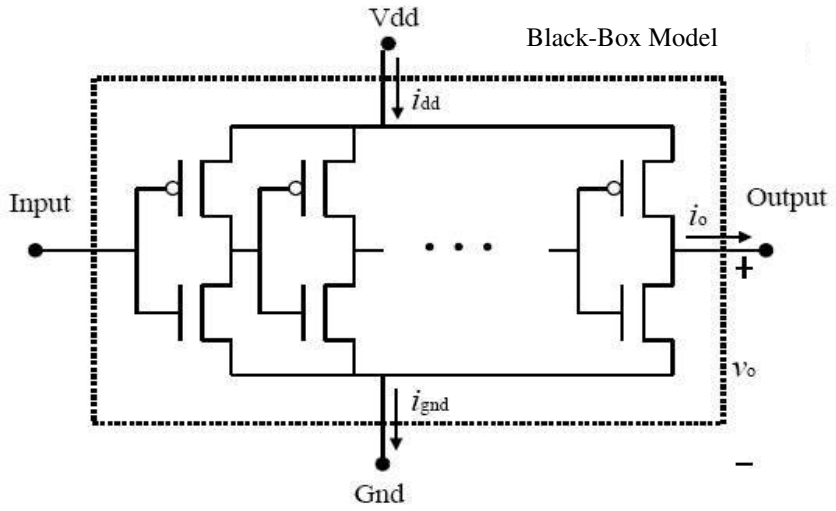


Figure 1.4 Black-box macromodel of a nonlinear circuit.

1.2 Rationale for Nonlinear Driver/Receiver Macromodeling

According to the International Technology Roadmap for Semiconductors (ITRS) 2004, the power supply voltage of ICs in year 2005 is 0.9 V [A23]. The power supply voltage will steadily reduce to 0.8 V by 2007 and 0.6 V by 2013. Both the on-chip clock frequency and chip-chip (off-chip) frequencies will increase in years to come. The on-chip and off-chip frequencies for 2005 are 5.2 and 3.1 GHz, 9.2 and 4.8 GHz in year

2007, and 22 and 14.9 GHz in 2013, respectively. With the complexity of interconnects and packages increasing and the rise and fall time of the signals decreasing, the previously neglected electromagnetic effects cannot be ignored anymore. With the decrease in the margin of error, accurate analysis of power and signal integrity issues is becoming important [A24].

Signal integrity refers to a broad set of interconnect design issues such as signal reflections, impedance mismatch, and crosstalk coupling. On the other hand, power integrity refers to a set of power supply design issues such as resonance, IR voltage drop, and SSN.

Digital driver circuits drive electronic signals through lossy interconnects in high-speed digital systems. These electronic signals get deformed as they propagate through transmission lines because of crosstalk, attenuation, and impedance mismatch. Receiver circuits receive the distorted signal from interconnects and feed it to the processors. Digital driver and receiver circuits play an important role in high-speed digital systems. Since digital driver/receiver circuits are complex nonlinear dynamic systems containing a very complex functional part and a high number of pins (several hundreds for modern microprocessors), accurate macromodeling of digital I/O drivers is a significant challenge.

In order to perform efficient signal integrity and power integrity analysis on today's high-speed systems, nonlinear driver and receiver macromodels should have all of the below mentioned characteristics.

1. Nonlinear macromodels protect the intellectual property (IP) information of transistor-level driver/receiver circuits and minimize reverse engineering.

2. Driver and receiver macromodels consume less simulation time and CPU memory compared to transistor-level circuits without losing accuracy.
3. Sensitive issues like SSN and crosstalk can be accurately captured using these macromodels.
4. Macromodels are extendable to multiple ports for including power supply and ground node effects.
5. Macromodels are scalable to include temperature and process variations.
6. Macromodels can be developed in a generic SPICE like format that is universally acceptable independent of the platform being used.

Since the existing nonlinear macromodels do not meet all the above mentioned requirements, there is always a need for accurate nonlinear macromodels that satisfy all the above requirements. The accuracy of signal integrity and electromagnetic compatibility (EMC) simulations depends on the accuracy of the available macromodels of circuit elements. Having accurate macromodels is of paramount importance in the design of fast circuits. These models also capture sensitive effects like waveform distortion, crosstalk, overshoots, and radiation.

1.3 Existing Driver and Receiver Macromodels

Input/output buffer information specification (IBIS) is the present industry standard for driver and receiver modeling [A25]. In this section, a description of IBIS driver and receiver models, their accuracy, and their limitations are discussed.

1.3.1 IBIS Driver Models

IBIS is the input/output buffer information specification from the electronics industry alliance (EIA). It is a modeling technique that provides a simple table-based buffer model for semiconductor devices [A26]-[A29]. IBIS models can be used to characterize current/voltage (I/V) output curves, rising/falling transition waveforms, and package parasitic information of the device. It is important to note that an IBIS model is also intended to provide nonproprietary information about driver/receiver circuits. Furthermore, there are many different SPICE formats in the industry today, and not all are compatible with one another [A30]. This process of converting one transistor-level SPICE netlist compatible with another SPICE format is time consuming. IBIS models are compatible with all SPICE formats, saving a lot of time and labor. An IBIS model can be generated either by measurement, which requires having a well-controlled environment and measurement devices, or by using a SPICE generated netlist and running multiple SPICE simulations to get the necessary current voltage tables and voltage transition tables [A31].

IBIS behavioral models are based on DC current vs. voltage curves along with a set of rise and fall times of the driver output voltage and packaging parasitic information of the I/O buffer. A typical IBIS behavioral model representation is shown in Figure 1.5.

An IBIS model consists of pull-up and pull-down transistors, power and ground clamp diodes, input and output die capacitance (C_comp), and package characteristics (the values of the lead inductance (L_package), resistance (R_package), and capacitance (C_package)) [A32]-[A33]. IBIS modeling takes into account 1) DC steady-state I/V

characteristics of the pull-up and pull-down transistors, 2) I/V characteristics of the power and ground clamps, and 3) IBIS transition waveforms.

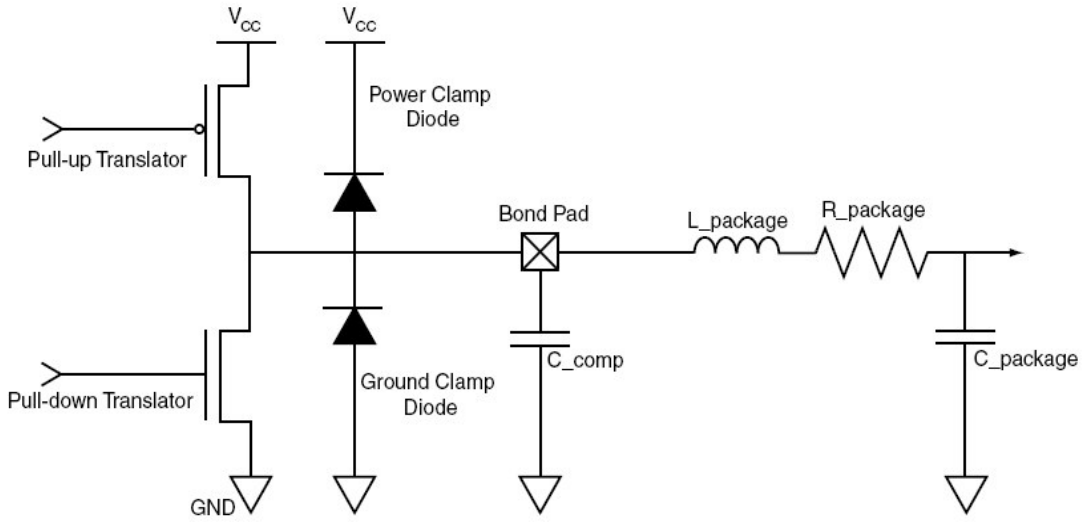


Figure 1.5 IBIS driver model schematic.

To generate pull-up and pull-down curves, voltage sources are connected at both driver input and output. The output voltage source tries to drive high (for the pull-up curve) or low (for the pull-down curve). The output voltage is swept from (Vgnd - Vcc) to 2Vcc, and the output current at each output voltage is recorded. If the driver has an enable input, the sweep is performed a second time with the driver disabled. This gives the performance of the clamping structure that may be present. The pull-down curve is a result of subtracting the ground clamp I/V curve from the logic LOW I/V curve, since this is where the pull-down transistor is active, as shown in Figure 1.6. Similarly, the pull-up curve is generated by subtracting the power clamp I/V curve from the logic-HIGH I/V curve, as shown in Figure 1.7. Again, the full range is from $-V_{cc}$ to $2V_{cc}$.

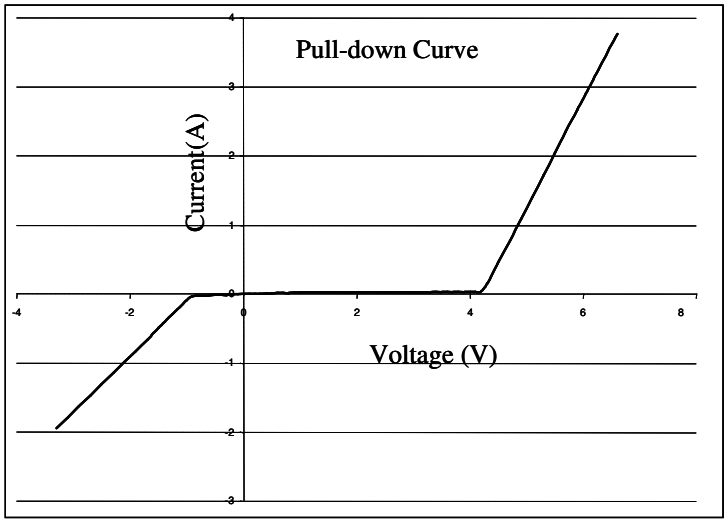


Figure 1.6 IBIS driver pull-down curve.

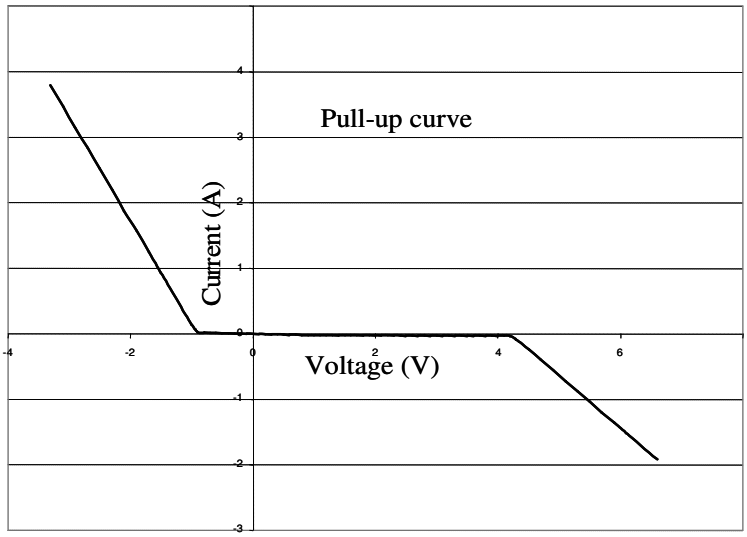


Figure 1.7 IBIS driver pull-up curve.

The ground clamp curve is derived from the ground relative data gathered while the buffer is in a high-impedance state and illustrates the region where the ground clamp diode is active. A voltage source is attached to the associated pin and the output voltage is swept from (Vgnd – V_{cc}) to (Vgnd + V_{cc}). The power clamp curve is derived from the

V_{cc} relative data gathered while the buffer is in a high impedance state (the driver is disabled) and shows the region where the power clamp diode is active. This measurement ranges from V_{cc} to $2V_{cc}$ [A33].

The pull-up and power clamp curves are V_{cc} relative, i.e., the voltage values are referenced to the V_{cc} pin. The output current of a pull-up or power clamp configuration depends on the voltage between the output and V_{cc} pin and not the voltages between the output and the ground pins. The voltages in IBIS tables are derived as shown:

$$Vtable = Vcc - Voutput \tag{1.1}$$

$$Vtable = Voutput \tag{1.2}$$

Equation (1.1) represents voltages for pull-up and for power clamp devices and equation (1.2) represents voltages for pull-down and ground clamp devices.

An IBIS model can also provide rising and falling *v-t* waveforms, which illustrates the transitions from GND to V_{cc} and from V_{cc} to GND. These curves can be taken from SPICE simulations when the buffer output is terminated appropriately for the appropriate stimulus at buffer input. The ramp rates are taken when the output voltage varies from 20% to 80% V_{cc} for the rising waveform and from 80% to 20% V_{cc} for the falling waveform. In calculating the ramp rates, the effect of parasitics is ignored. These ramp rates are much faster than slew rates in which the package parasitics are taken into account. A typical rising and falling waveform plot for an IBIS driver model is shown in Figure 1.8.

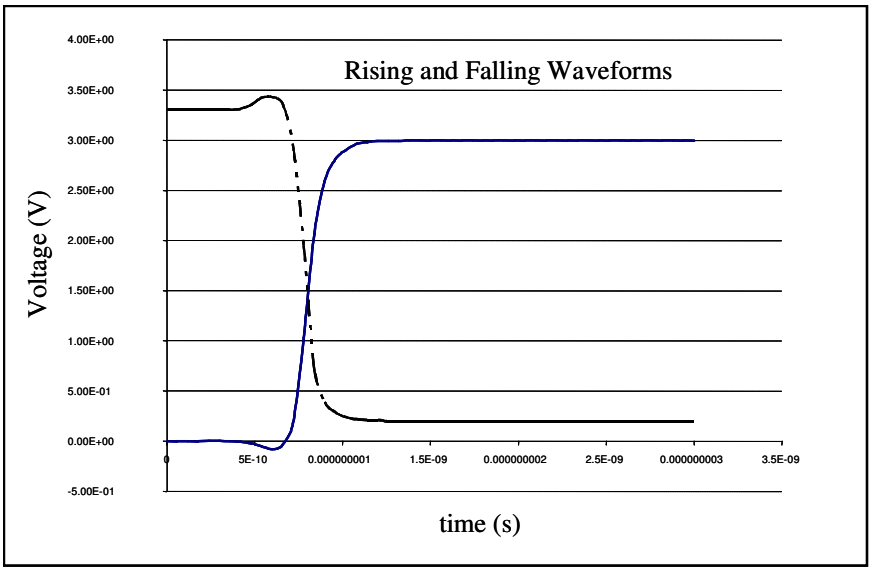


Figure 1.8 IBIS driver rising and falling waveforms.

1.3.2 IBIS Driver Modeling Limitations

IBIS driver modeling is popular and widely used as it is commercially available, has large sets of libraries, and IBIS models run faster than actual transistor-level driver models. However, IBIS models have limitations. One of the limitations is that the physical effects to be considered are decided a-priori when the equivalent circuit is defined, leaving little or no possibility for including the effects inherent to the device. IBIS models also fail to accurately capture the dynamic characteristics of the driver as the modeling technique relies heavily on static characteristics. The extension of IBIS driver models to multiple ports only results in less accurate models [A35]. IBIS models cannot accurately model sensitive effects like SNN and crosstalk accurately [A36].

Figure 1.9 compares the accuracy of an IBIS driver model with a transistor-level driver circuit. A test case has been generated where a driver circuit was connected to an ideal 25-ohm ideal transmission line that got terminated at the far end with a 1 pF

capacitance. The transmission line had a delay of 0.2 ns. The driver circuit was generated by cascading seven inverters in series, making the driver circuit weakly nonlinear. The driver was operated at 1 GHz. The near-end and the far-end voltage waveforms of the transmission line were measured for the transistor-level driver circuit and the IBIS driver model. It can be clearly seen that the IBIS model cannot accurately capture the magnitude and timing information even for a weakly nonlinear driver circuit.

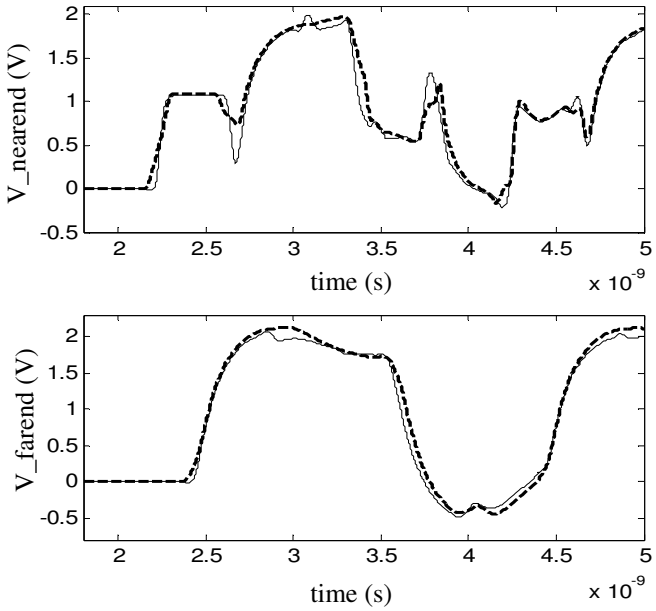


Figure 1.9 Voltage waveforms at the near-end and the far-end of the transmission line for transistor-level driver circuit (straight line) and IBIS model (dashed line).

The same driver circuit was modeled using spline function with finite time difference (SFWFTD) modeling approach. A detailed explanation of SFWFTD modeling approach is given in Chapter 2. It can be seen from Figure 1.10 that the near-end and far-end voltage waveforms using SFWFTD macromodel accurately matches with the actual driver circuit voltage waveforms.

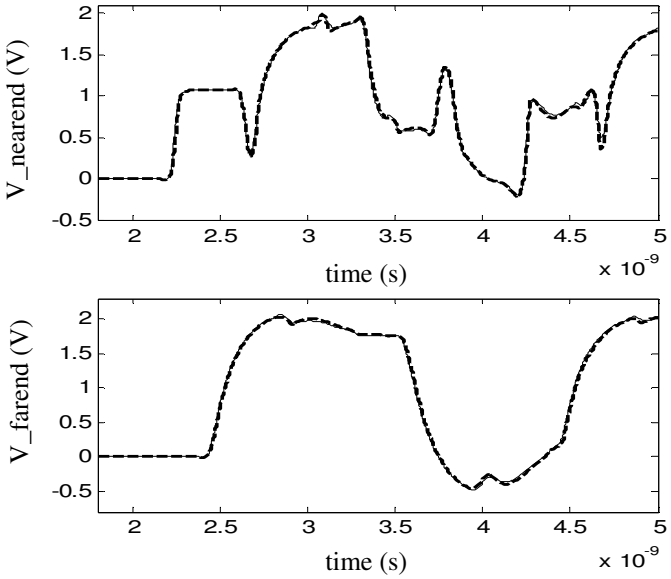


Figure 1.10 Voltage waveforms at the near-end and the far-end of the transmission line for transistor-level driver circuit (straight line) and spline function with finite time difference model (dashed line).

Another test case was generated where 10 identical weakly nonlinear drivers were each connected to a 25-ohm ideal transmission line that was in turn terminated by a 25-ohm matched resistor. Figure 1.11 shows the voltage at the near-end and far-end of the transmission line along with the SSN when all the 10 drivers switch simultaneously. Figure 1.12 shows the voltage waveforms at the near-end and the far-end of the transmission line along with SSN when 10 IBIS driver models switch simultaneously. It can be seen from Figure 1.12 that IBIS models fail to capture sensitive effects like SSN accurately. This is another limitation of IBIS driver models. IBIS driver models cannot be extended to multiple ports without losing accuracy.

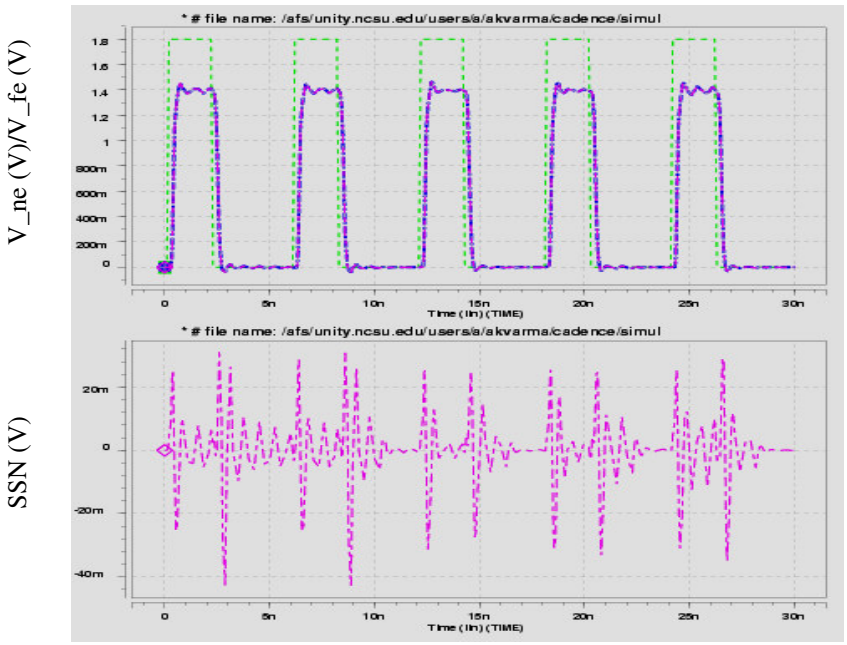


Figure 1.11 Voltage waveforms at the near-end and the far-end of the transmission line for a transistor-level driver circuit and SSN when multiple drivers are switching.

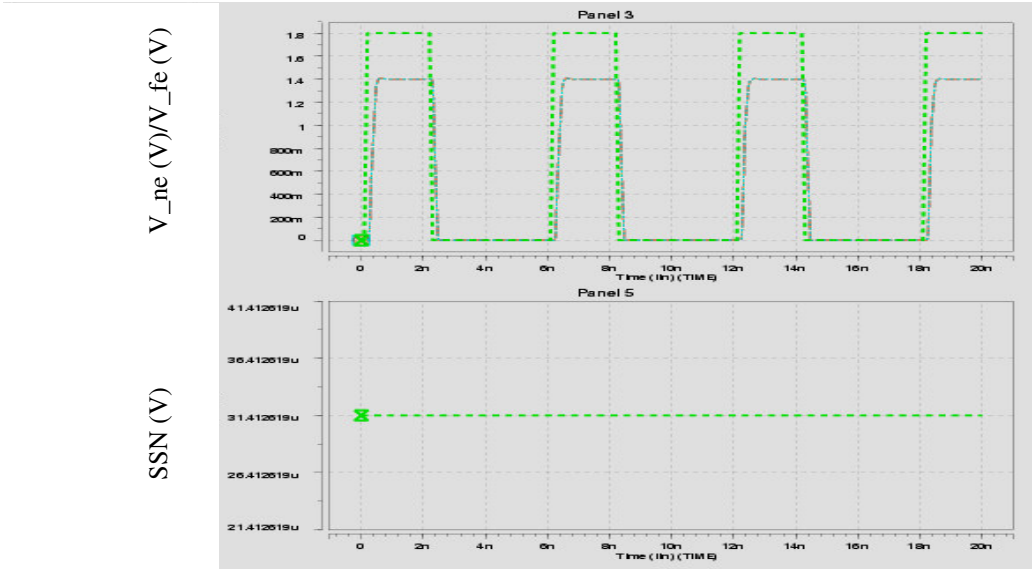


Figure 1.12 Voltage waveforms at the near-end and the far-end of the transmission line for an IBIS driver model and SSN when multiple IBIS driver models are switching.

1.3.3 IBIS Receiver Models

Receiver modeling also plays an important role in analyzing signal integrity issues for today's high-speed digital systems. IBIS receiver models are current industry standard.

In IBIS, a typical receiver circuit contains a ground (GND) clamp and a power clamp, as shown in Figure 1.13. The power and GND clamps represent the electrostatic discharge (ESD) structure. The IBIS receiver circuit also requires a logic voltage high threshold (v_{ih}) and a voltage low logic threshold (v_{il}) for the input. The IBIS simulator uses these logic threshold values to compute signal integrity issues such as overshoot/undershoot and noise margins [A37].

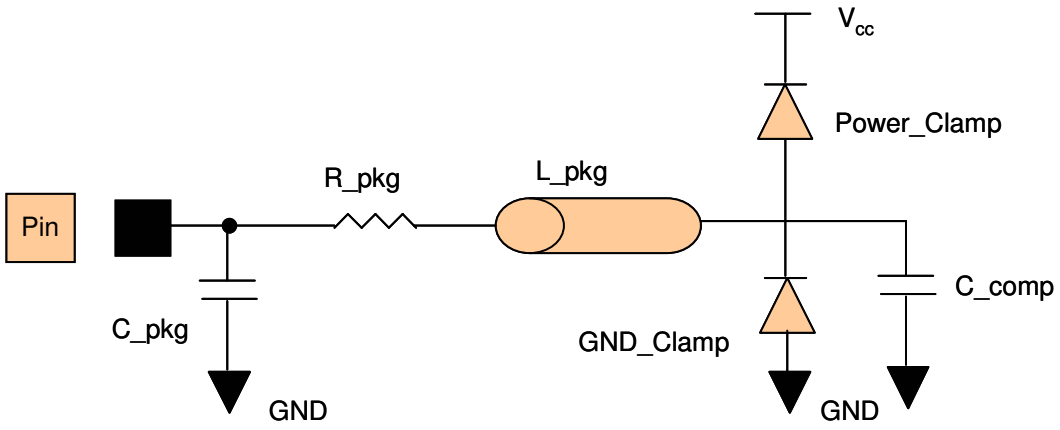


Figure 1.13 IBIS receiver model schematic.

The input model also includes the package parasitics and the input die capacitance, C_comp. The C_comp parameter is connected to the input, usually with reference to ground when the IBIS file is used in the simulator. It is the capacitance seen when looking from the pad back into the buffer. C_comp is a key parameter, especially for

receiver inputs. In IBIS receiver modeling, the power and the GND clamp data are generated following the same procedure used for an IBIS driver model. The sweep voltage range will be –Vcc to Vcc for the GND clamp and Vcc to 2Vcc for the power clamp curve because the power clamp data is relative to Vcc as shown in equation (1.1).

IBIS receiver models are highly based on the static characteristics of the receiver. A combination of clamping diodes does not accurately model the loading characteristics of the receiver circuit. The output characteristics of the receiver circuit cannot be modeled accurately by taking only the threshold voltages into account, as the delay information through the receiver is lost.

1.4 Proposed Research and Dissertation Outline

The key contribution of this work has been to develop different macromodels for different types of driver and receiver circuits that result in huge computational speed-up compared to actual transistor-level driver/receiver circuits and at the same time preserve accuracy. Both driver and receiver modeling approaches are black-box in nature. These macromodeling approaches can be extended to multiple ports to take into account the effect of power and ground nodes. Additionally, this work also focuses on macromodeling differential driver circuits and modeling driver circuits with pre-emphasis or pre-compensation effect. It is envisioned that the proposed research work will contribute toward generating macromodels for all classes of driver and receiver circuits. Figure 1.14 shows the flow chart of the general procedure involved in modeling driver and receiver circuits. It can bee seen that coming up with PWL voltage sources (identification signals) is ad hoc and it takes few hours to generate the macromodels of

driver and receiver circuits. But the time involved in modeling them is off-set by the fact that once driver or receiver circuits are modeled they become part of a library and can be used over and over again for numerous system level simulations.

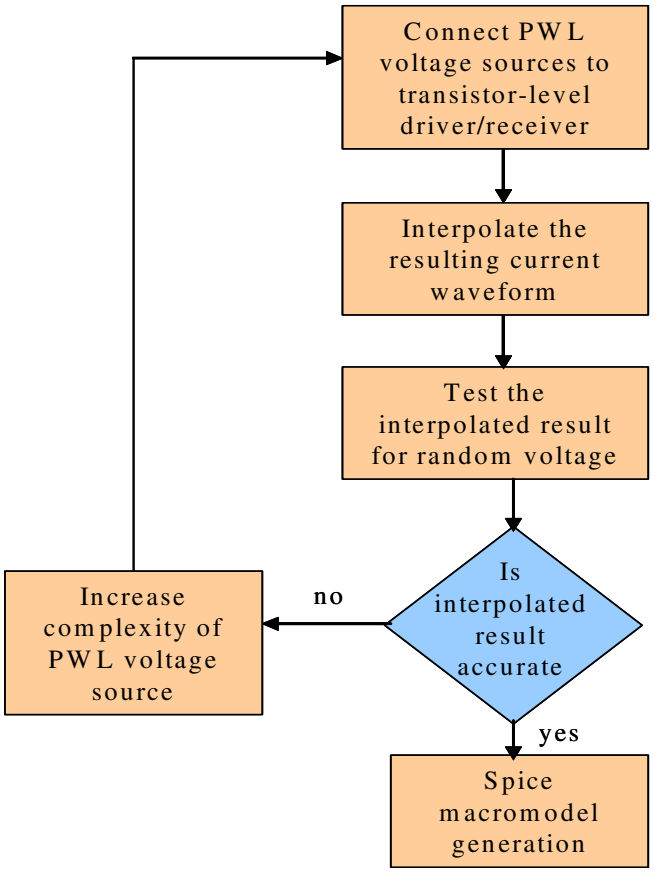


Figure 1.14 Procedure for driver/receiver modeling

The following items are discussed in various sections of the dissertation.

1. Modeling driver circuits using the radial basis function (RBF) modeling approach. In this approach, a nonlinear relation is drawn between the driver output current and output voltage using summation of Gaussian functions. This represents previous work done by the University of Torino, Italy [A38]. The advantages and limitations of this method are discussed and validated on various test cases.

2. Modeling driver circuits based on their complexity:
    a) Weakly Nonlinear Driver Circuits: Driver circuits where the static characteristics dominate the dynamic characteristics of the driver. A modeling methodology based on static characteristics has been proposed to model these driver circuits. Static characteristic macromodels are accurate and (10 – 100)X faster than actual weakly nonlinear transistor-level driver circuits
    b) Moderately Nonlinear Driver Circuits: Spline function with finite time difference (SFWFTD) approach has been proposed to model moderately nonlinear driver circuits. This methodology takes into account both the static and the dynamic characteristics of the driver circuit. SFWFTD macromodels are (10 – 40)X faster than transistor-level driver circuits, depending on the driver being modeled
    c) Highly Nonlinear Driver Circuits: A modeling methodology based on recurrent neural networks (RNN) has been proposed to model highly nonlinear driver circuits. RNN network is a powerful nonlinear interpolation tool that can model highly nonlinear feedback memory systems. RNN models also provide (10 – 40) X speed-up over transistor-level driver circuits, depending on the kind of driver being modeled.
    d) Pre-compensation and pre-emphasis driver circuits are becoming popular in transmitting signal though lossy interconnects. Pre-emphasis driver circuits have been modeled accurately by including the effect of pre-compensation in the weighting functions.

All the above models are weakly sensitive to the external load connected to the driver. All these macromodels can capture sensitive effects like crosstalk and power supply noise

accurately. The accuracy and computational speed-up of the above macromodels have been tested on different test cases and results have yielded good correlation with the actual transistor-level circuits.

3. Extension of the above modeling methods to multiple ports.

   The developed modeling methodologies have been extended to multiple ports. These macromodels can capture the effect of power supply voltage and driver local ground on the driver output signal. These macromodels can also capture the effect of driver output signal on the power supply voltage and local ground. Capturing these sensitive relations can lead to the accurate modeling of sensitive effects like SSN when multiple drivers are switching. Macromodels generated using the proposed modeling techniques consume less CPU time and CPU memory compared to transistor-level driver circuits.

A full driver level simulation taking power supply node and ground node effects on the driver output has been performed using the above macromodels. Test results show good correlation between the transistor-level driver circuits and macromodels.

4. Modern high-speed digital interfaces have turned to low-voltage differential signaling (LVDS) because of their numerous advantages over single-ended signaling. Differential signals have lower voltage swings than single-ended signals. Differential signals have a reduced electromagnetic interference (EMI) effect and crosstalk coupling. A modeling methodology based on an RNN network has been proposed to macromodel differential drivers with and without the pre-emphasis/pre-compensation effect. The macromodels based on RNN networks are 10X faster and consume 10X less CPU memory compared to transistor-level driver circuits.

5. A modeling methodology for macromodeling receiver circuits has been proposed. Receiver circuits are difficult to model, as the input to the receiver is analog in nature and the output is digital. Receiver modeling can be divided into macromodeling receiver input characteristics (where the receiver acts like a capacitive load) and receiver output characteristics (that forms input to logic circuits).
   a) Input characteristics of the receiver have been modeled by expressing receiver input current as a function of receiver input voltage using RNN function or SFWFTD approach.
   b) Output characteristics of the receiver have been captured by using a combination of voltage transfer characteristics of the receiver and a finite time delay element. The voltage transfer characteristics of the receiver can create the output voltage signature of the receiver accurately. The time delay of the signal through the receiver circuit has been accurately captured by the finite time delay element.
   c) The receiver circuit modeling technique has been extended to multiple-ports to include the effect of power supply voltage on both the receiver input and the receiver output characteristics.

The accuracy and simulation speed-up of the receiver macromodels has been compared to transistor-level receiver circuits for various test cases.

6. A scalable macromodeling approach for driver circuits has been proposed. The output voltage and current of a driver circuit are dependent on the power supply voltage, temperature, and process variation of the driver circuit. Variations in the above parameters affect the output voltage and the output current of driver circuits. Scalable driver macromodels that take into account the effect of temperature, power supply

voltage, and process variations help in efficiently analyzing signal integrity issues efficiently at an early stage of a design process. The scalability of the RNN modeling approach for both differential and single-ended driver circuits has been shown for some test cases.

The remainder of this thesis is organized as follows. In Chapter 2, the RBF modeling approach to model driver circuits is discussed along with its limitations. Chapter 2 also presents different macromodeling techniques for different classes of drivers. Depending on the complexity of the driver being modeled, a different nonlinear macromodeling method is required. Extension of the driver modeling approach to multiple ports has been discussed in Chapter 3. The accuracy of the modeling approach has been tested on various test cases. Modeling receiver circuits is described in Chapter 4. Receiver circuits also play an important role in signal integrity and power integrity analysis. In this chapter, modeling receiver input and output characteristics are explained in detail. Chapter 5 discusses the modeling of differential driver circuits. Differential signaling reduces EMI effects, crosstalk coupling, and high voltage swings. Macromodeling differential driver circuits with and without pre-emphasis is discussed in this chapter. Chapter 6 discusses the scalability of driver circuits. Chapter 7 concludes the thesis with future work.

CHAPTER II

MACROMODELING OF DIGITAL DRIVERS

It has been seen from Chapter I that while IBIS models are fast, they cannot accurately model the nonlinear dynamic characteristics of driver circuits. Therefore, IBIS models are more suitable to model driver circuits that are weakly nonlinear in nature.

In this chapter, radial basis function (RBF) modeling approach for modeling driver circuits has been discussed [A38]. This approach is the previous work done by Prof. Canavero's group at University of Torino, Italy [A38]. In this modeling approach, the driver output current is expressed in terms of driver output voltage using a summation of radial basis functions, usually Gaussian functions. RBF functions can accurately model the nonlinearity of driver circuits, but RBF models also have their limitations. In this chapter, the advantages and limitations of this method are discussed and validated on various test cases.

Driver circuits can be broadly classified into three groups depending on their nonlinearity: 1) weakly nonlinear driver circuits, 2) moderately nonlinear driver circuits, and 3) highly nonlinear driver circuits. In weakly nonlinear driver circuits, the static nonlinear characteristics dominate the dynamic nonlinear characteristics. These driver circuits have little or no memory in them. In this chapter, a modeling approach based on static characteristics has been proposed for weakly nonlinear driver circuits. Moderately nonlinear driver circuits have memory in them and therefore, the dynamic characteristics

cannot be ignored. Driver circuits with memory act as a feedback system; the output at time instance‘t’ is dependent on previous time instances of the output. The nonlinearity of a circuit can be gauged based on the number of previous time steps needed to model the output accurately. In case of moderately nonlinear driver circuits, the output value at previous one or two time instances has to be taken into account. For highly nonlinear driver circuits, the output values of previous two or more instances have to be taken into account. Spline function with finite time difference (SFWFTD) modeling has been proposed to model moderately nonlinear driver circuits in this chapter. For highly nonlinear driver circuits, recurrent neural network (RNN) modeling has been proposed. Highly nonlinear circuits have large memory or feedback effect. In a highly nonlinear driver circuit, dynamic characteristics of the driver circuit dominate the static characteristics. RNN is a special branch of artificial neural networks (ANN) that model nonlinear systems with feedback accurately. All the above mentioned macromodels are black-box in nature, faster than transistor-level driver circuits, weakly sensitive to the external load connected, and at the same time maintain high accuracy.

In this chapter, static characteristics modeling technique, SFWFTD modeling technique, and RNN modeling technique are discussed in detail in sections 2.3, 2.4, and 2.5, respectively. Spice netlist generation for all these macromodels has also been described in these sections. Macromodel to hardware measurement correlation has been shown in section 2.6. Pre-emphasis/pre-compensation driver circuits that are used to drive signals through extremely lossy lines have been modeled in section 2.7. The accuracy of these macromodels has been tested on numerous test cases and results show good correlation between the macromodel and the transistor-level circuit waveforms.

2.1 Radial Basis Function Based Modeling

RBF modeling technique is a parametric modeling technique in which a nonlinear relation is drawn between the output current and the output voltage of a driver circuit. The output current and output voltage are related using a piece-wise (PW) parametric equation as shown below:

$$i_o(k) = w_1(k) f_1(\Theta_1, x(k)) + w_2(k) f_2(\Theta_2, x(k)) \tag{2.1}$$

$$f_n(\Theta_n, x(k)) = \sum_{j=1}^{M} \theta_{nj} \phi\left(\left|x - cn\,j\right|, \beta\right),\ n = 1, 2 \tag{2.2}$$

In equation (2.1), i_o is the driver output current, f_1 and f_2 are the sub-models that relate the driver output current to the output voltage for driver input HIGH and LOW, respectively [B1]. The transition from one logic state to another is done with the help of weighting functions w_1 and w_2. The time-varying weighting functions act as switches between the sub-models f_1 and f_2. Sub-models f_1 and f_2 are expressed as a summation of radial basis functions, as shown in equation (2.2), where M is the number of basis functions needed for f_1 or f_2 to accurately model the digital driver. Gaussian, Multi-quadric, and thin-plate spline are some of the radial basis functions as shown in Figure 2.1. In equation (2.2), Φ is the asymptotically increasing or decreasing basis function and θ_j is the weight of the basis function Φ. The centers of the basis functions are defined by c_j and the width or the spread parameter is defined by β [B2].

The *regressor vector x* in equation (2.2) collects the past r samples of the driver output voltage (v_o) and the driver output current (i_o) along with the present sample of the driver output voltage. The parameter r has been called the dynamic order of the model. The

dynamic order for driver changes, depending on the complexity of the driver that is being modeled [B3].

$$x^T(k) = \begin{Bmatrix} i_o(k-1), i_o(k-2), ..., i_o(k-r), \\ v_o(k), v_o(k-1), ..., v_o(k-r) \end{Bmatrix} \tag{2.3}$$

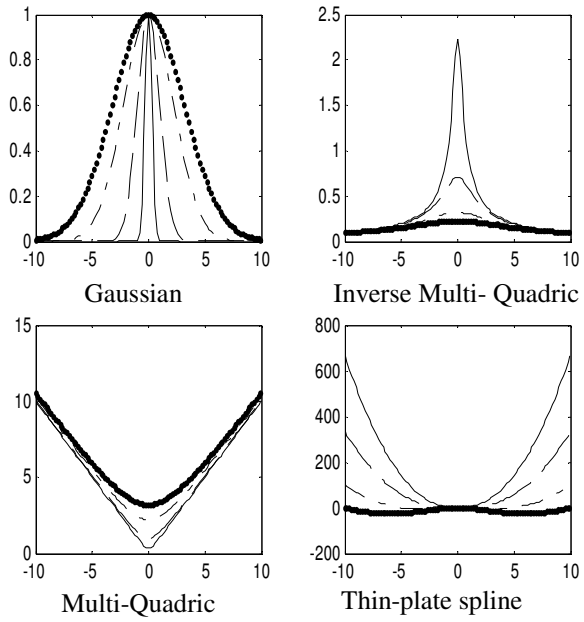


Figure 2.1 Different radial basis functions with varying widths.

This modeling technique can be extended to multiple ports by changing the *regressor vector x*. To include the effect of the power supply v_{dd}, the *regressor vector x* should be modified as shown:

$$x^T(k) = \begin{Bmatrix} i_o(k-1), i_o(k-2), ..., i_o(k-r), \\ v_o(k), v_o(k-1), ..., v_o(k-r), \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-r) \end{Bmatrix} \tag{2.4}$$

A set of identification signals are used at the output of the driver for driver input HIGH and LOW to estimate f_1 and f_2, respectively. These identification signals are generated from a piece-wise linear (PWL) voltage source connected at the driver output.

The voltage waveform has different slopes and different rise times to capture all the dynamic and static characteristics of the driver. These identification signals have to be carefully generated to excite all the dynamic characteristics of the driver [A38]. Figure 2.2 shows the voltage identification signal at the output of the IBM driver ('Bt3350') to estimate sub-model f_1 for 1ns rise time. Figure 2.3 shows the corresponding current waveform at the IBM driver output when the input is held HIGH.

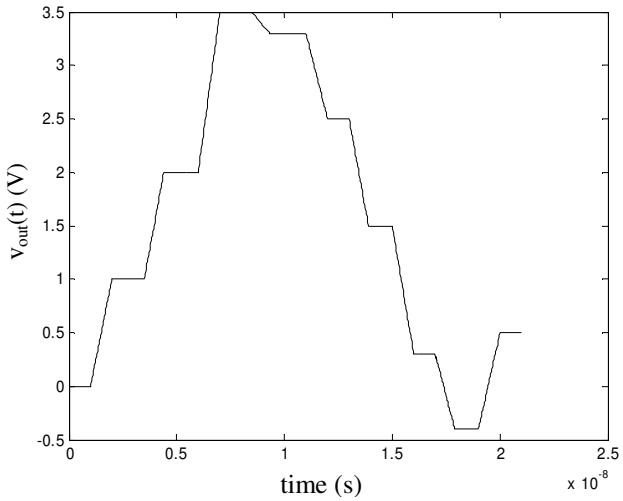


Figure 2.2 Voltage Identification signal at the driver output to estimating f_1 for 1ns rise time.

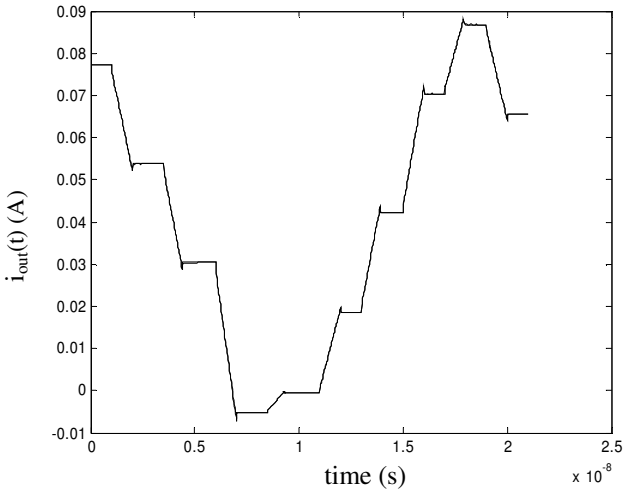


Figure 2.3 Current Identification signal at the driver output corresponding to Figure 2.2.

Sub-models f_1 and f_2 are parameterized using the classical Gram-Schmidt (CGS) method in [B4]. Sub-models $f_{1,2}$ in equation (2.1) can be expressed as:

$$f_n = \Phi\theta \; ; \quad n = 1,2 \tag{2.5}$$

$$\textit{where,} \quad f_n = \left[f_n(1), f_n(2), ..., f_n(N)\right]^T , \tag{2.6}$$

$$\Phi = \left[\Phi_1, \Phi_2, ..., \Phi_M \right], \tag{2.7}$$

$$\textit{and} \quad \Phi_i = \left[\Phi_i(1), \Phi_i(1), ..., \Phi_i(N)\right]^T , \quad 1 \le i \le M \tag{2.8}$$

$$\theta = \left[\theta_1, \theta_2, ..., \theta_M \right]^T \tag{2.9}$$

where N is the number of data points and M is the number of basis functions needed to estimate sub-models f_1 or f_2.The regression matrix Φ can be decomposed into

$$\Phi = WA \tag{2.10}$$

where A is an $M \; X \; M$ triangular matrix with 1's on the diagonal and 0's below the diagonal and W is an $N \; X \; M$ matrix with orthogonal columns w_i.

$$W^T W = D \tag{2.11}$$

where D is a diagonal matrix with elements d_i.

$$d_i = w_i^T w_i = \sum_{t=1}^{N} w_i(t) w_i(t), \; 1 \le i \le M \tag{2.12}$$

The CGS method computes one column of A at a time, orthogonalizing Φ_i into set of orthogonal basis vector [B5]. At the k^{th} stage, it makes the k^{th} column orthogonal to each

of the *k-1* previously orthogonalized columns and the operation repeats for $k = 2,3, \ldots, M-1, M$.

Once sub-models f_1 and f_2 are estimated, the time-varying weighting functions w_1 and w_2 can be calculated by solving equation (2.1) for two different loads. A resistive load is picked as one load and a series connection of a resistor and a battery is picked as another load.

2.1.1 Limitations of RBF Modeling Approach

Even though RBF modeling approach accurately models transistor-level driver circuits, it has some inherent limitations. One of the limitations is that the CGS method that is used to estimate the RBF function parameters is very sensitive to round-off errors. If Φ from equation (2.10) is ill-conditioned, then the resulting W would lose its orthogonality and re-orthogonalization would be necessary. Another limitation with RBF modeling approach is that with the decrease in input rise time for a driver circuit, the dynamic characteristics of the driver start dominating the static characteristics, which will lead to an increase in the number of basis functions needed to accurately model the driver circuit, as shown in Table 2.1. Table 2.1 shows that with the decrease in rise time for an IBM driver ('AGP'), there is an increase in the number of basis functions needed to model the driver circuit.

Increase in the number of basis functions results in increase in the complexity of the RBF driver models which in turn results in increase in the simulation time. Figure 2.4 shows how the number of basis functions increase the simulation time for the RBF driver

modeling. From numerous experiments it has been found that increase in basis functions also results in poor numerical convergence with Hspice circuit simulator [B8].

Table 2.1 Driver input rise time Vs. number of basis functions needed.

Identification signal rise times	Gaussian function	
	f_1	f_2
1ns	6	6
0.8ns	5	5
0.5ns	11	11
0.3ns	13	17

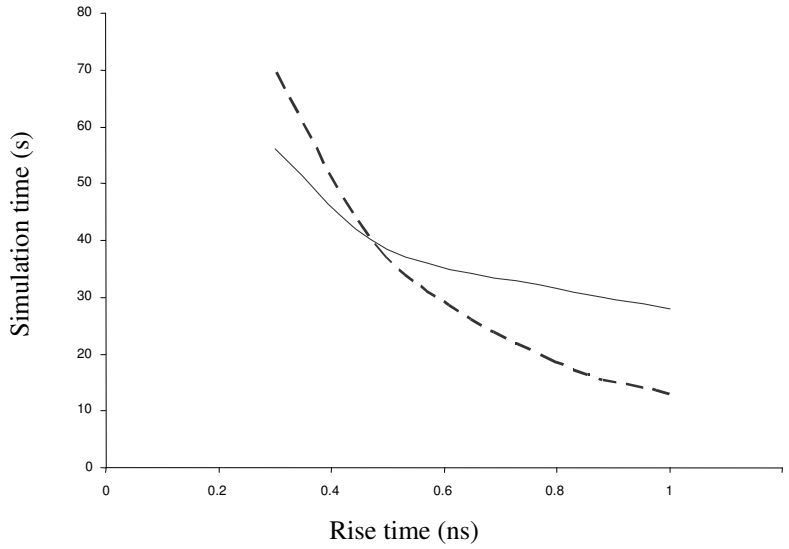


Figure 2.4 Driver input rise time Vs. simulation time required for transistor-level IBM driver (straight line) and RBF driver model (dashed line).

Also, RBF models cannot accurately model highly nonlinear driver circuits. In highly nonlinear circuits the memory or feedback effect is predominant. A test case has been generated to demonstrate the failure of RBF models in capturing highly nonlinear driver

circuits. A detailed description of the test case is discussed later in section 2.4.2 of this chapter. Finally, RBF modeling approach has some numerical convergence issues with Hspice when the modeling approach is extended to multiple ports in order to capture sensitive effects like simultaneous switching noise (SSN).

2.2 Classification of Driver Circuits

Representation of sub-models f_1 and f_2 using RBFs is not the best way to model nonlinear driver circuits. In this chapter, it has been proposed that different representations of sub-models f_1 and f_2 result in efficient modeling of different groups of driver circuits. To model weakly nonlinear driver circuits IBIS and static characteristic models are efficient. RBF and SFWFTD models are efficient in modeling moderately nonlinear driver circuits. RNN models are efficient in modeling highly nonlinear driver circuits.

To decide which category the transistor-level driver circuit belongs to, a PWL voltage source is connected at the output of a driver circuit. This PWL voltage source is similar to the one explained in section 2.1 with different rise times and steady state values. If the transition from one steady state to another in the PWL voltage source results in similar transformation of the resulting driver output current signature, then the driver is weakly nonlinear. IBIS and static characteristic models are efficient in modeling these driver circuits. Static characteristic models are better than IBIS models and the former can be extended to multiple ports. Sometimes the resultant driver output current signature has some additional dynamic characteristics every time the PWL voltage source makes a transition from one steady state to another. These dynamic characteristics result in current

spikes at transitions and make the current signature different from the voltage signature. Drivers that belong to this category are termed as moderately nonlinear. RBF and SFWFTD are efficient in modeling these driver circuits. SFWFTD models are simple to generate and do not have convergence issues with Hspice compared to RBF models. If the resultant current signature does not resemble the PWL voltage waveform then the driver is highly nonlinear and it has memory or feedback in it. RNN models are efficient in modeling highly nonlinear driver circuits.

2.3 Static Characteristic Modeling

For weakly nonlinear driver circuits, the speed and memory advantages of RBF models are not predominant as explained in section 2.1. For this class of drivers, IBIS models are more suitable as they retain both speed and accuracy. The problem with IBIS models is that these models cannot be extended to multiple ports to capture sensitive effects like SSN. Static characteristic models can be extended to multiple ports without losing accuracy. In driver circuits with little memory, the static characteristics of the driver circuit dominate the dynamic characteristics for normal excitations. Therefore, a static characteristic relation can be used in relating the driver output current to the output voltage [B6]. The driver output current can be expressed as:

$$i_o(v_o(t) = w_1(t) f_1(v_o(t)) + w_2(t) f_2(v_o(t)) \tag{2.13}$$

where i_o is the driver output current, v_o is the driver output voltage, and w_1 and w_2 are weighting functions that help sub-models f_1 and f_2 transit from one state to another. Sub-models f_1 and f_2 in equation (2.13) can be represented as:

$$f_n(v_o(t)) = a_{m,n}{v_o}^m + a_{(m-1),n}{v_o}^{(m-1)} + \ldots a_{o,n}; n = 1,2 \tag{2.14}$$

In equation (2.14), *a's* are constants that depend on the kind of driver being modeled and the value of *m* is of the order 1 to 5. For an IBM driver ('BagpV3V2'), the output current vs. output voltage plot when the driver input is held HIGH is shown in Figure 2.5

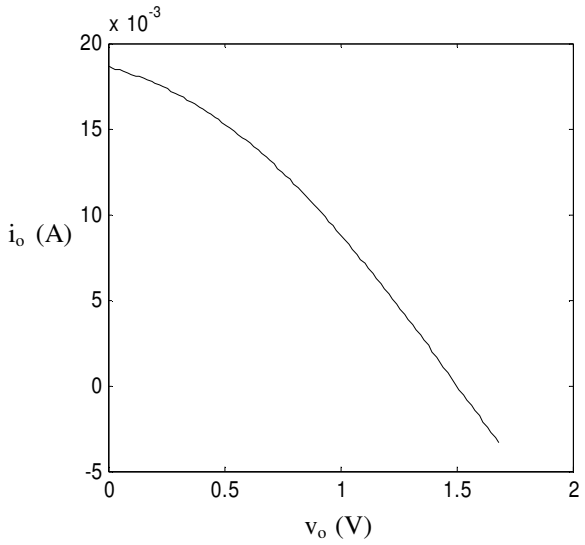


Figure 2.5 DC relation between driver output current and output voltage when driver input is HIGH.

It can be seen from Figure 2.5 that the output current can be expressed as a function of output voltage, as shown in equation (2.14). Similarly, the driver output current can be expressed using equation (2.14) when the driver input is held LOW. The whole process of computing sub-models f_1 and f_2 is computationally simple.

Once sub-models f_1 and f_2 are estimated, weighting functions w_1 and w_2 can be estimated by linear inversion of equation (2.13) for two different loads as shown:

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} f_{a1} & f_{a2} \\ f_{b1} & f_{b2} \end{bmatrix}^{-1} \begin{bmatrix} i_a \\ i_b \end{bmatrix} \tag{2.15}$$

Figure 2.6 shows typical weighting functions $w_1(t)$ and $w_2(t)$ from equation (2.15). Since two equations are needed to calculate two unknowns, the driver output is terminated with two different loads. The choice of the loads should test sub-models f_1 and f_2 in the range of interest of load variation. Usually, a resistor is used as the first load and a resistor with DC voltage source is used as the second load. The driver power supply is usually picked as the DC source.

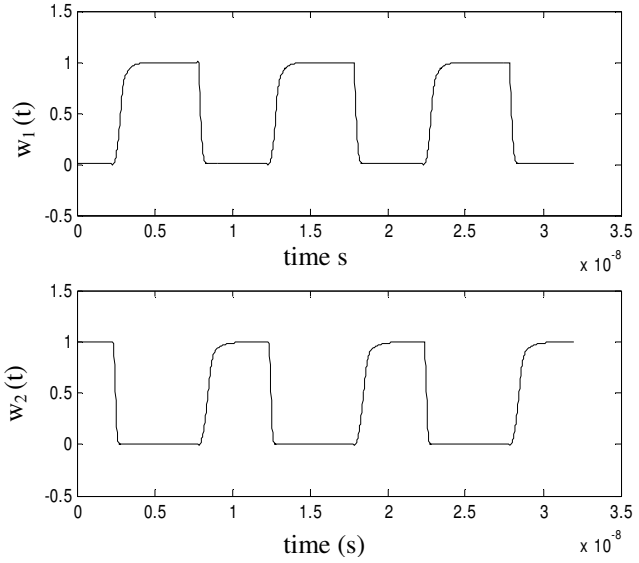


Figure 2.6 Weighting functions $w_1(t)$ and $w_2(t)$.

A SPICE equivalent circuit for the static characteristic modeling technique can be generated using PWL voltage sources, voltage-dependent current sources and voltage-dependent voltage sources. The weighting functions w_1 and w_2 can be represented using PWL voltage sources. A voltage-dependent voltage source can be used to represent sub-models f_1 and f_2. A voltage-dependent current source can be used to capture the relation between driver output current and output voltage. A schematic spice netlist for static characteristic model is shown in Figure 2.7

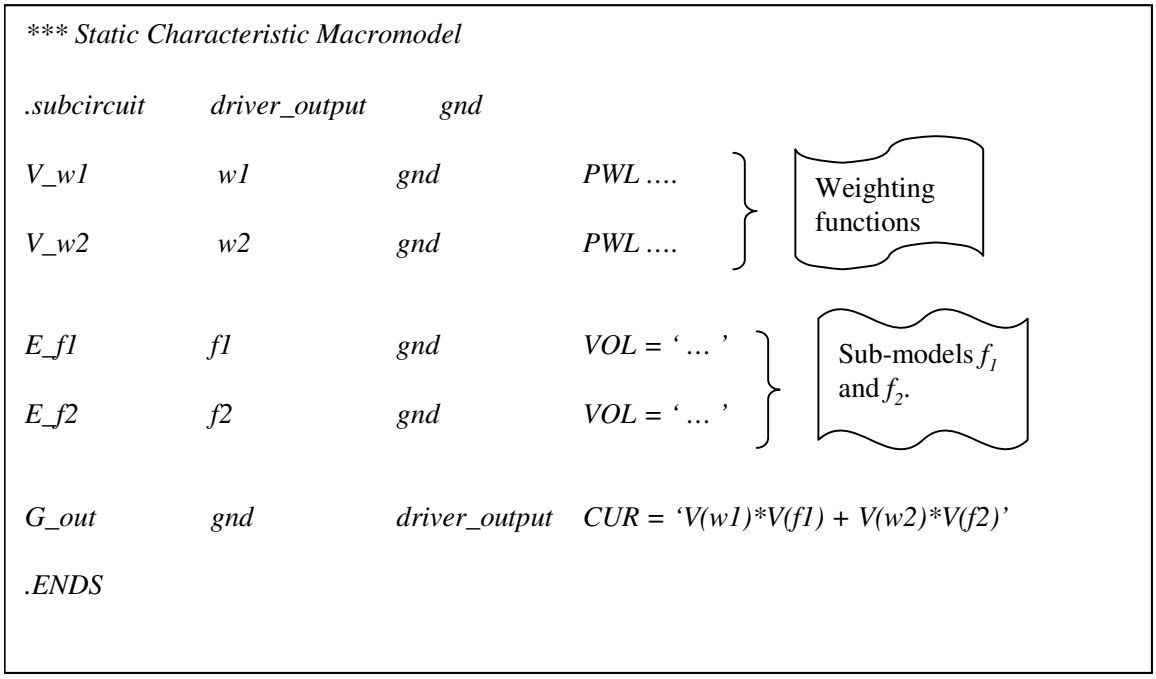


Figure 2.7 Spice netlist representation for a static characteristic model.

2.3.1 Test Results

In this section, the accuracy and computational speed-up of static characteristic models and RBF models has been studied on an IBM driver (‘Bt3350pd_c’) for different driver rise times. The test vehicle was an IBM encrypted spice netlist with a 3.3 V power supply. The driver circuit is 130 KB in size. The rise time of the driver was reduced from 1 ns to 0.3 ns to compare RBF and static characteristic models with respect to simulation time and accuracy. Load insensitiveness of both the models has also been tested during this process by loading them with transmission lines of different characteristic impedance for each case.

Case 1: A test case was generated where the IBM driver was connected to a 100-ohm ideal transmission line with a line delay of 1 ns and excited with a 0.8 ns rise time. RBF model required six basis functions for f_1 and six basis functions for f_2. The dynamic order

r was one. A linear relation was used to model sub-models f_1 and f_2 in static characteristic models. Figure 2.8 shows the near-end and far-end voltage waveforms of the transmission line for all the three models (RBF, static characteristic and encrypted transistor-level driver spice circuit).

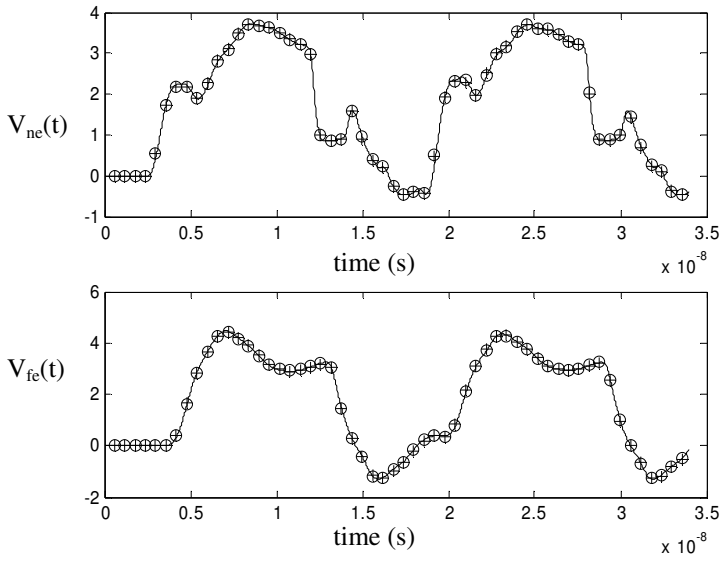


Figure 2.8 Near-end $V_{ne}(t)$ and far-end $V_{fe}(t)$ waveforms on the 100-ohm transmission line connected to IBM transistor model (straight line), RBF model ('o') and static characteristic model ('+').

The actual encrypted model took 35 s, the RBF model took 21 s and the static characteristic model took 4 s. It can be seen from Figure 2.3 that the voltage waveforms of static characteristic model and RBF model matches well with transistor-level circuit waveforms. All the simulations were carried out on a SUN ULTRA-10 workstation.

Case 2: In this test case, the driver was connected to an ideal transmission line of characteristic impedance 75-ohms with a line delay of 1 ns. In the RBF model, sub-model f_1 required six basis functions and f_2 required nine basis functions, and the dynamic order was one. A quadratic polynomial was used to model both f_1 and f_2 of the static

characteristic model. The near-end and far-end voltage waveforms were computed, as shown in Figure 2.9. The actual encrypted IBM model took 27 s the RBF model took 27 s and the linear model took 4 s. The voltage waveforms of the static characteristic model match well with the transistor-level driver circuit. All the simulations were carried out on a SUN ULTRA-10 workstation.

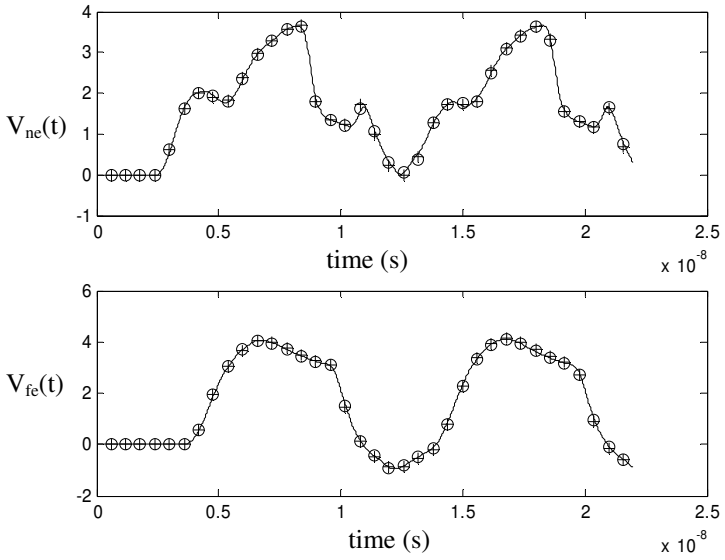


Figure 2.9 Near-end $V_{ne}(t)$ and far-end $V_{fe}(t)$ waveforms on the 75-ohm transmission line connected to IBM transistor model (straight line), RBF model ('o') and static characteristic model ('+').

For weakly non-linear drivers, static characteristic models are better replacements to RBF models as they can capture the same nonlinearity with less computational time and the modeling process is effortless [B6]. RBF models tend to become more and more complex as the rise time decreases, losing their edge in computational time over transistor-level driver models as shown in Figure 2.10.

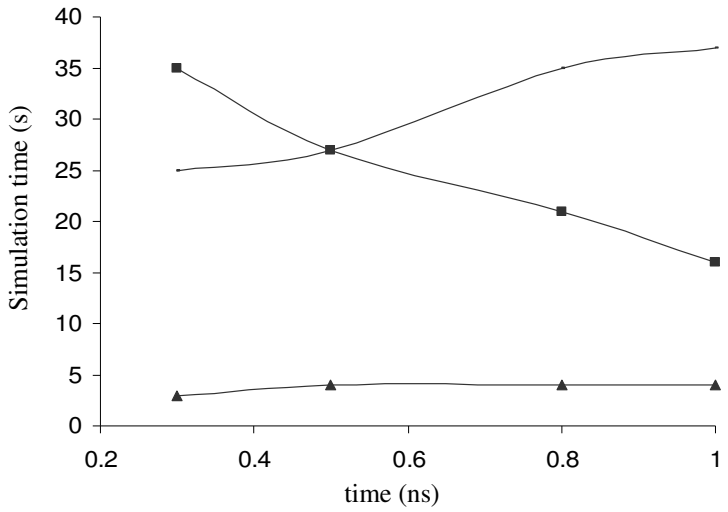


Figure 2.10 Time taken for simulation for different rise-times a) IBM model (straight line), b) RBF model (■), and c) static characteristic model (▲).

As the rise time of the driver input decreases, dynamic characteristics of the driver cannot be ignored. The static characteristic modeling methodology does not take into account the dynamic characteristics of the driver circuit. Static characteristic models cannot accurately model moderately nonlinear driver circuits where the dynamic characteristics are predominant as shown in Figure 2.11.

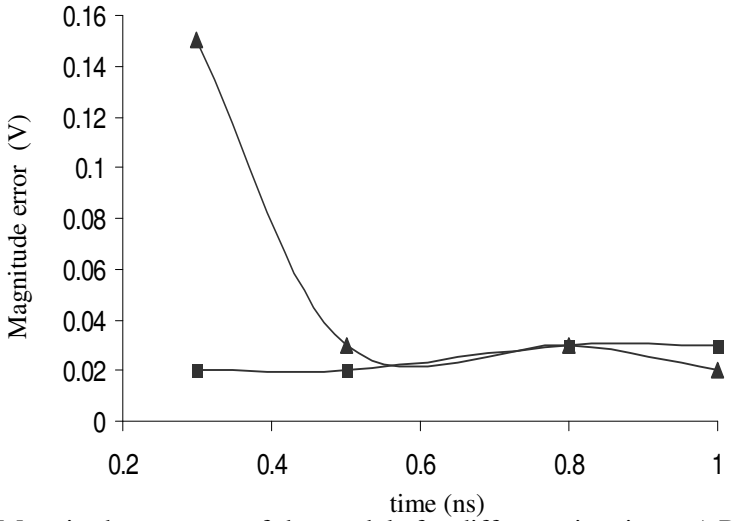


Figure 2.11 Magnitude accuracy of the models for different rise-times a) RBF model (■) and b) static characteristic model (▲).

2.4 Spline Function with Finite Time Difference (SFWFTD) Modeling

For moderately nonlinear driver circuits, the dynamic characteristics start to dominate the static characteristics and therefore, static characteristic models cannot accurately capture the nonlinearity. Figure 2.12 shows a PWL voltage source connected at the output of an IBM driver ('AGPV3V2') when the driver input is held HIGH. The accuracy of the static model can be determined from Figure 2.12, where it can be seen that the static model current deviates from the original current response generated from the PWL voltage source connected at the output of the driver.

Since the deviation is a result of the failure to capture the dynamic characteristics, the static modeling methodology can be modified to include the previous time instances of the driver output current so that the dynamic behavior of the driver can be captured.

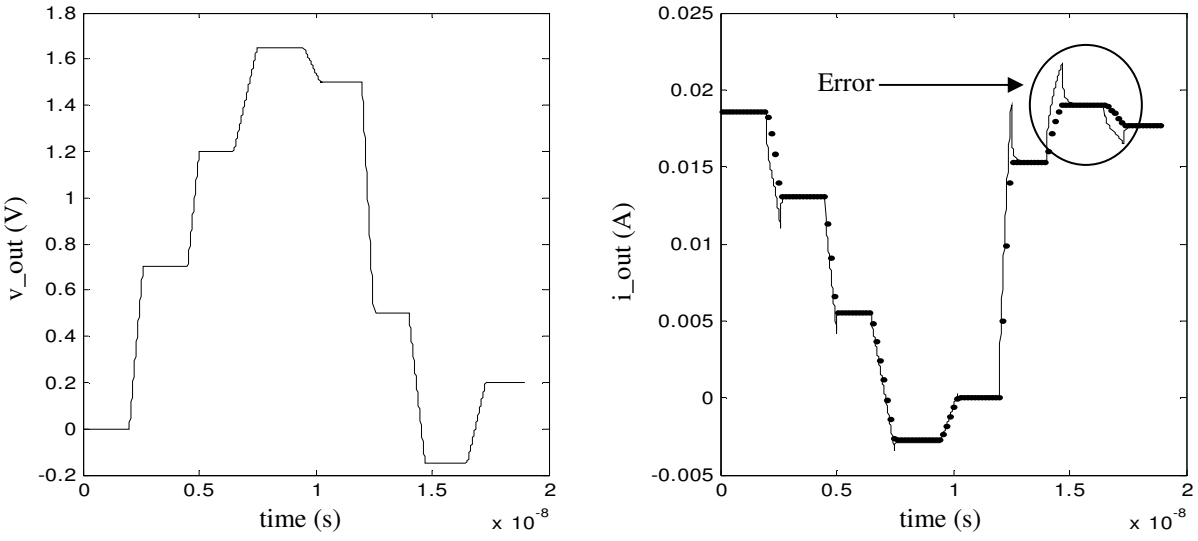


Figure 2.12 PWL voltage source connected at the output of driver for input HIGH and the corresponding output current from transistor-level driver circuit (straight line) and static characteristic model (dotted line).

When the driver input is set HIGH, the current at the output, i_{oh}, can be expressed as sub-model f_1 (from now on $f_{1,s}$), as shown in equation (2.14). Sub-model $f_{1,s}$ at time instance '*k-1*' can be expressed as:

$$f_{1,s}(k-1) = i_{oh}(k-1) = a_{m,1}{v_o}^{m}(k-1) + a_{(m-1),1}{v_o}^{(m-1)}(k-1) + \ldots\ldots + a_{0,1} \tag{2.16}$$

Incremental change in the driver output current Δi_{oh} is the difference between the present instance *(k)* and previous time instance *(k-1)* values of sub-model $f_{1,s}$ as shown:

$$f_{1,s}(k) - f_{1,s}(k-1) = i_{oh}(k) - i_{oh}(k-1) = \Delta i_{oh} \tag{2.17}$$

$$\text{(Or)}\ \ f_{1,s}(t) - f_{1,s}(t-\Delta t) = \Delta i_{oh} \tag{2.18}$$

Once Δi_{oh} is calculated, the first derivative of driver output current i'_{oh} can be approximated as:

$$\frac{f_{1,s}(t) - f_{1,s}(t-\Delta t)}{\Delta t} = \frac{\Delta i_{oh}}{\Delta t} = i'_{oh} \tag{2.19}$$

where Δt is the sampling time. The effect of dynamic behavior when the driver input is HIGH is captured in i'_{oh}. Similarly, the effect of dynamic behavior when the driver input is LOW is captured by $\Delta i'_{ol}$. Therefore, dynamic behavior can be added to static sub-models $f_{1,s}$ and $f_{2,s}$ as shown:

$$\begin{aligned} f_1(k) &= f_{1,s}(k) + p * i'_{oh} \\ f_2(k) &= f_{2,s}(k) + pp * i'_{ol} \end{aligned} \tag{2.20}$$

where *p* and *pp* are constants whose magnitude can be estimated by calculating the least mean square error between $f_{1,s}$/$f_{2,s}$ and the transistor-level driver output current values for inputs HIGH/LOW, respectively [B7]-[B8].It can be seen from Figure 2.13 that with the

inclusion of one previous time instance of the driver output current, the modeled and simulated output current values match accurately.

It is important to note that there is no limitation on the number of previous output current time instances that can be added to the static sub-models $f_{1,s}$/$f_{2,s}$ as shown:

$$\begin{aligned} f_1(k) &= f_{1,s}(k) + p * i'_{oh} + q * i''_{oh} + ... \\ f_2(k) &= f_{2,s}(k) + pp * i'_{ol} + qq * i''_{ol} + ... \end{aligned} \tag{2.21}$$

Typically, for most of the driver circuits, spline function with finite time difference (SFWFTD) models need one previous time instance to accurately model the driver output voltage characteristics.

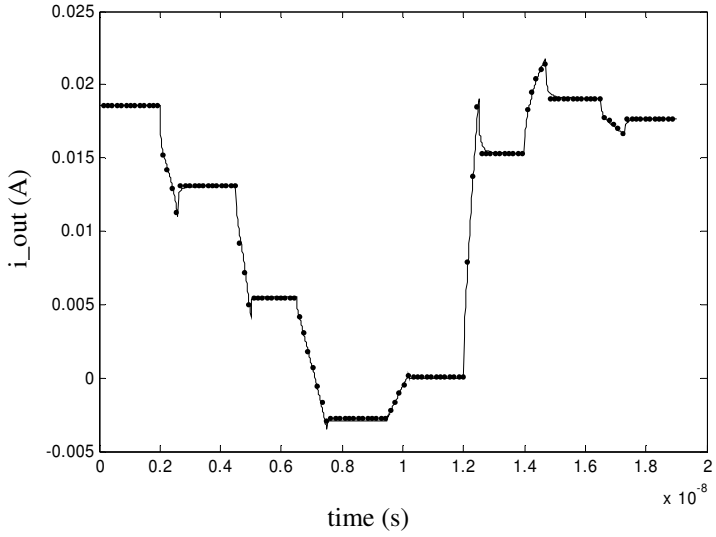


Figure 2.13 Output current from an IBM transistor-level model (straight line) and from SFWFTD model (dotted line).

SFWFTD method can accurately model moderately nonlinear driver circuits. This modeling approach is simple and efficient compared to RBF modeling approach where macromodels have numerical convergence problems with Hspice.

A SPICE equivalent circuit of SFWFTD model is similar to the static characteristic model spice netlist. The weighting functions w_1 and w_2 can be represented using PWL voltage sources. A voltage-dependent current source can be used to represent spline functions of sub-models f_1 and f_2. The dynamic behavior of the driver can be captured using state equations. Assuming,

$$E(k) = f_1(k) \tag{2.22}$$

$$Since \quad f_1(k) - f_1(k-1) = E(t) - E(t-\Delta t) \tag{2.23}$$

$$\frac{E(k) - V(k)}{R} = i(k) = E(k) - V(k) \tag{2.24}$$

$$C\frac{dV}{dt} = i(t) \tag{2.25}$$

Figure 2.14 shows the equivalent circuit representation of equations (2.23) to (2.25) for capturing the dynamic characteristics (previous time instances).

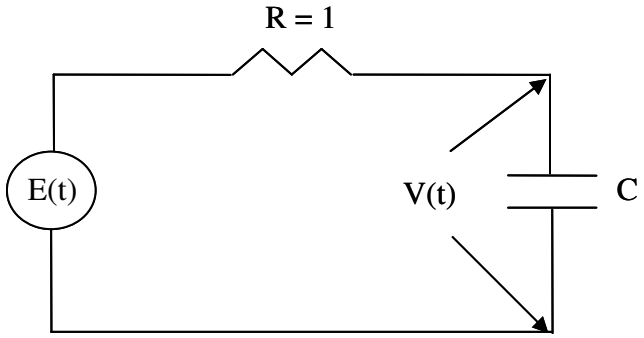


Figure 2.14 Circuit representation of dynamic characteristics.

It can be seen from equations (2.24) and (2.25), that the previous time instance of sub-models f_1 can be estimated easily. Similar procedure can be repeated to estimate the previous time instance of sub-model f_2 [B8].

The value of capacitor 'C' is based on the sampling time step. Similarly, the above technique can be used to capture the previous time instances for driver output current and voltage. Same procedure can be repeated to estimate more than one past time instance of driver output current and voltage. Figure 2.15 shows a schematic of spice netlist for SFWFTD method.

```
*** Spline Function with Finite Time Differnce Approximation  Macromodel

.subcircuit     driver_output        gnd

V_w1            w1          gnd          PWL ….

V_w2            w2          gnd          PWL ….


E_f1            x1          gnd          VOL = ' … '
R_f1            x1          x2           1
C_f1            x2          gnd          C

E_f2            y1          gnd          VOL = ' … '
R_f2            y1          y2           1
C_f2            y2          gnd          C


G_out           gnd         driver_output   CUR = 'V(w1)*( … ) + V(w2)*( … )'

.ENDS
```

Finite time difference approximation using state equations.

Figure 2.15 Spice netlist representation for SFWFTD approximation model.

2.4.1 Test Results

In this section, the accuracy and the simulation speed of SFWFTD model is studied for different rise times on IBM ('AGPV3V2') driver circuit. This driver is an encrypted spice transistor-level circuit with a 1.5 V power supply and 800 KB in size.

Case 1: A test vehicle was designed in which a driver was connected to a transmission line which is terminated by a 5 pF capacitance. The voltage waveforms at the near-end of the transmission line were measured. The IBM encrypted driver was connected to a 100-ohm ideal transmission line that has a line delay of 0.5 ns. The driver was given an input pulse with 0.5 ns rise time and a period of 10 ns. It took 10 RBFs for f_1 and six RBFs for f_2 to accurately capture the nonlinearity of the driver. The dynamic order r used for RBF model was one. In case of SFWFTD model, a third order spline function with one previous time instance was needed for sub-models f_1 and f_2 to accurately model the driver. The value of p and pp was 5*sampling time which was 20 ps. IBM driver took 12 minutes 51 s for simulation, RBF model took 18 s, SFWFTD model took 7 s for simulation on a SUN ULTRA-10 workstation. The voltage waveform at the near-end of the transmission line is plotted in Figure 2.16. Both SFWFTD model and RBF model consumed less memory compared to the actual IBM driver model and ran faster than the actual transistor-level circuit. It can be seen from Figure 2.16 that the waveform of the macromodels matched well with the actual IBM driver voltage waveform.

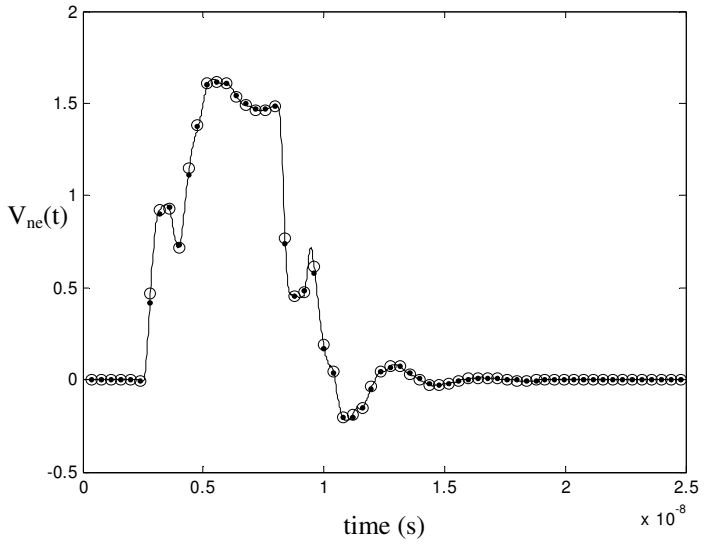


Figure 2.16 Near end $V_{ne}(t)$ voltage waveform for IBM driver (straight line), SFWFTD model (dotted line) and RBF model (0).

Case 2: A test case was generated where an IBM driver was connected to a 75-ohm transmission line with a line delay of 0.2 ns. The transmission line was terminated with a 1 pF. A pulse with 0.2 ns rise time and a period of 2 ns was given. The near and far end voltage waveforms on the transmission line were measured, as shown in Figure 2.17. RBF model took 10 basis functions to model f_1 and eight to model f_2. A third-order spline function with one previous time instance was needed for sub-models f_1 and f_2 to accurately model the driver. The value of *p* and *pp* is 5*sampling time, which was 20 ps. The IBM driver took 934 s for simulation, the RBF model took 35 s and the SFWFTD model took 14 s for simulation. All simulations were carried out on a SUN ULTRA-10 workstation. It can be seen that the voltage waveforms from the macromodels match well with the transistor-level driver circuit waveforms.

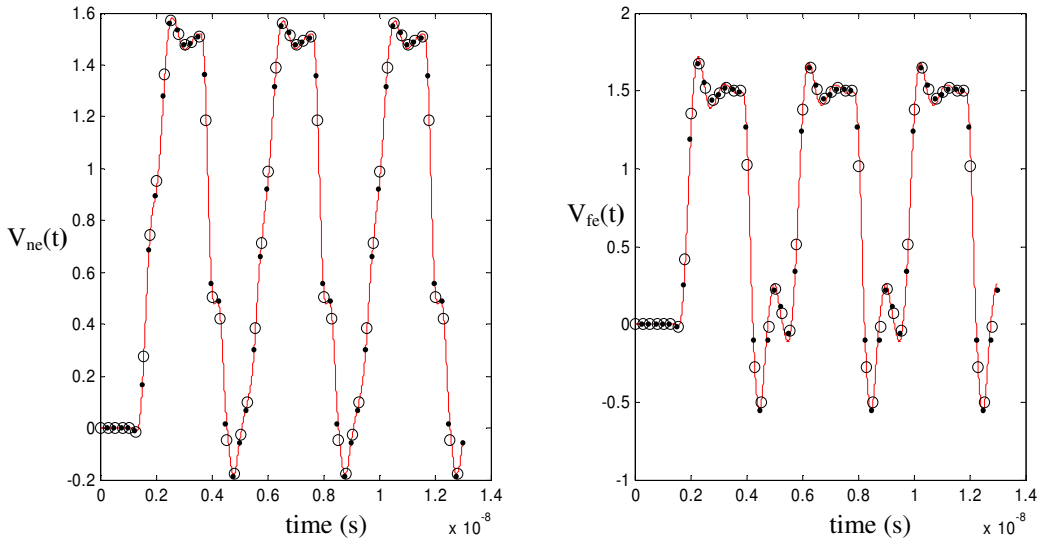


Figure 2.17 Near-end $V_{ne}(t)$ and far-end $V_{fe}(t)$ voltage waveforms for IBM driver (straight line), SFWFTD model (dotted line) and RBF model (0).

2.5 Recurrent Neural Network Modeling

Recurrent Neural Networks is a branch of neural networks that models systems with memory or feedback effect accurately. A RNN has the capability of learning and then representing dynamic system behavior. It has been used in areas such as signal processing, speech recognition, system identification, and control [B9]–[B11]. Figure 2.18 shows a schematic of a typical recurrent neural network. It can be seen from Figure 2.18 that the output, O(t), at time 't' is also dependent on the previous time instances of output (O(t-1), O(t-2), …, O(t-tr)). The number of previous output time instances that are required to model the system accurately is dependent on the complexity of the system being modeled.

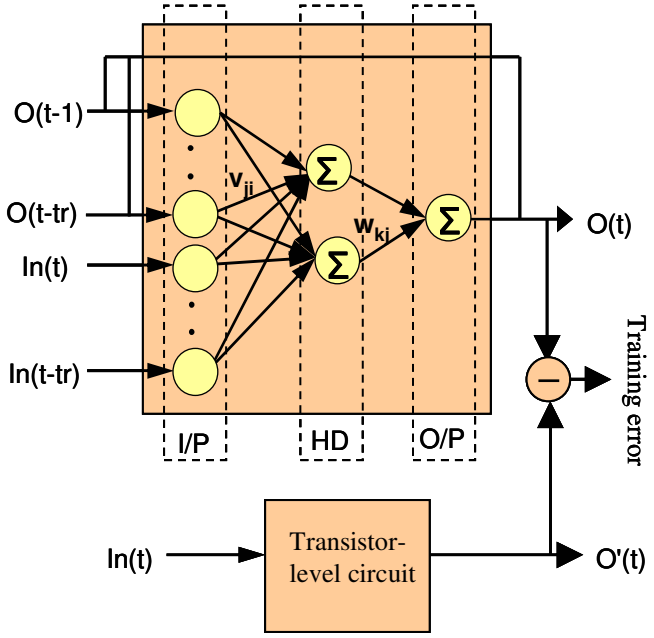


Figure 2.18 Schematic of Recurrent Neural Network (RNN) model.

Recurrent neural networks (like ANNs) typically have three layers: the input layer, the hidden layer and the output layer. The inputs are fed into the input layer that is connected to the hidden layer through weights v_{ij}. The number of hidden layers and neurons can be increased or decreased depending on the complexity of the system being modeled.

The summation of product of the inputs with weights (v_{ij}) is stored in the hidden layer and is passed through hyperbolic tangential function to the output layer [B12]-[B14]. The output layer stores the summation of product on hidden layer outputs with weights (w_{jk}).

$$O(t) = \sum_{j=1}^{M} w_{kj}\, g\left(\sum_{i=1}^{N} v_{ji} x_i + v_{oj}\right) + w_{ok} \tag{2.26}$$

$$g(x) = (e^{ax} + e^{-ax})/(e^{ax} - e^{-ax}) \tag{2.27}$$

$$x(k) = \begin{Bmatrix} In(k), In(k-1), ..., In(k-t_r) \\ O(k), O(k-1), ..., O(k-t_r) \end{Bmatrix} \tag{2.28}$$

In equation (2.26), *x* is the vector that contains the inputs to the neural network, *M* is the number of hidden neurons and *N* is the number of inputs. Depending on the complexity of the system being modeled, the number of hidden neurons *M* can be increased or decreased. Function g(x) is a hyperbolic function. The output O(t) is compared with the actual output O'(t) to calculate the training error. Each dynamic response O(t) from RNN macromodel is also called a RNN output trajectory.

The weights are estimated to minimize the difference between RNN trajectory O(t) and transistor-level circuit data O'(t).

$$\min_{\Phi} \frac{1}{2} \sum_{k=1}^{N_t} (O'(k) - O(k))^2 \tag{2.29}$$

where Φ is a vector that contains all the weights of the neural network and N_t is the total number of time samples. In order to train the macromodel, derivatives of the error function with respect to each parameter in the RNN are required to form a Jacobian matrix. Since, the output at time '*t*' is dependent on previous output time instances, conventional back-propagation method is not applicable for neural-network training. A training scheme based on back propagation through time (BPTT) should be used to train the RNN model. Gradient-based optimization algorithms, such as the Levenberg–Marquardt and quasi-Newton methods are used to estimate the weights of the RNN macromodel [B15]-[B17].

2.5.1 RNN Driver Modeling

It was seen in the previous section that SFWFTD models accurately model moderately nonlinear driver circuits. But SFWFTD technique has limitations. When the transistor-level driver circuit models are highly nonlinear, SFWFTD method fails to capture the high nonlinearity present in the driver circuits. One solution for modeling these highly nonlinear driver circuits is to use RNN networks [B8].RNN functions can model the nonlinearity of these complex drivers. Sub-models f_1 and f_2 can now be expressed using RNN functions as shown:

$$f_n = \sum_{k=1}^{M} b_{kj}\, g\left(\sum_{j=1}^{N} a_{ji} x_i + a_{oj}\right) + b_{ok}; n = 1,2 \tag{2.30}$$

where

$$g(x) = (e^{x} + e^{-x})/(e^{x} - e^{-x}) \tag{2.31}$$

In equations (2.30) and (2.31), b and a are weights associated with the neural network, N represents number of hidden neurons, M represents number of outputs, and x is the regressor vector as shown:

$$x^{T}(k) = \begin{Bmatrix} i_o(k-1), i_o(k-2), ..., i_o(k-r), \\ v_o(k), v_o(k-1), ..., v_o(k-r) \end{Bmatrix} \tag{2.32}$$

where i_o and v_o are driver output current and voltage, respectively.

In RNN training, all weights in all of the layers are adjusted till the modeled current matches with transistor-level driver output current. A modified back propagation through time (MBPTT) algorithm is used to estimate the weights of the RNN network [B18]-[B19]. This algorithm takes into account the feedback effect of the output in training the neural network, which makes RNN functions more robust in modeling highly nonlinear

signatures. Depending on the nonlinearity of the driver to be modeled, neural networks can be modified to increase the hidden neurons.

Spice netlist generation for RNN models is similar to SFWFTD models. Previous time instances of driver output current and voltage are estimated using equations (2.23)–(2.25). Sub-models f_1 and f_2 are now replaced by hyperbolic tangential functions instead of SFWFTD models.

2.5.2 Test Results

A test case with IBM (SDRAM) transistor-level driver was generated to test the accuracy of SFWFTD, RBF, and RNN macromodels. The IBM driver was connected to a 50-ohm ideal transmission line and terminated with a 2 pF capacitance. The driver was given an input pulse with 0.2 ns rise time at 200 MHz. The voltage waveforms at the near-end and far-end of the transmission lines were measured from the RBF and the SFWFTD models. A fourth-order polynomial was used to model sub-models f_1 and f_2 in the SFWFTD macromodel. Sub-models f_1 and f_2 in the RBF model needed eight and nine basis functions, respectively. The dynamic order for RBF models was two. It can be seen from Figure 2.19 that both models failed to accurately capture the nonlinearity of the driver. Both the timing and the magnitude of the resulting voltage waveforms do not match accurately with the transistor-level driver circuit waveforms.

For RNN model, sub-models f_1 and f_2 needed three hyperbolic tangent functions, each with two previous time instances of driver output current and output voltage. It was found that the RNN model gives high accuracy for the same test case, as shown in Figure 2.20.

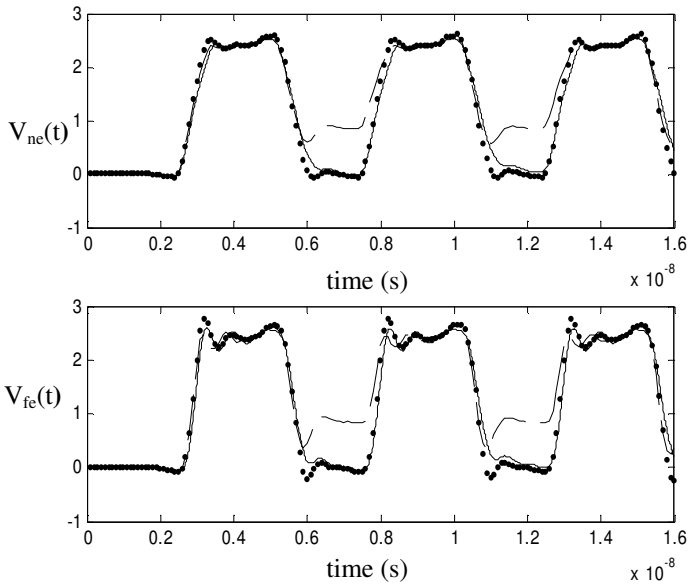


Figure 2.19 Near end ($V_{ne}(t)$) and far end ($V_{fe}(t)$) voltage waveforms on transmission line for IBM transistor-level driver model (straight line), RBF model (dashed line) and SFWFTD model (dotted line).

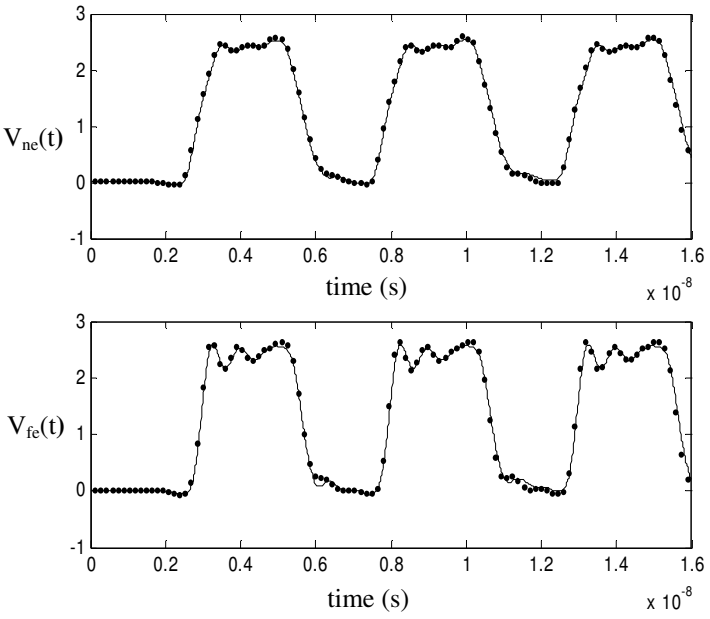


Figure 2.20 Near end ($V_{ne}(t)$) and far end ($V_{fe}(t)$) voltage waveforms on transmission line for IBM transistor-level driver model (straight line) and RNN model (dotted line).

It can be seen that for highly nonlinear driver circuits, a macromodel based on RNN is a good replacement for SFWFTD model.

2.6 Measurement to Model Correlation

Macromodels of driver and receiver circuits can be generated from (1) voltage and current information obtained from transistor-level SPICE simulations and (2) voltage and current information obtained from laboratory measurements. In this thesis, voltage and current information for modeling driver and receiver circuits is obtained from transistor-level SPICE simulations. Obtaining the current and voltage information from measurement is a difficult task and can sometimes lead to measurement errors. If the voltage waveforms from transistor-level driver and receiver SPICE netlists accurately correlate with the measurement results, then the macromodels generated from transistor-level SPICE netlists accurately match with measurement results.

Laboratory measurements ware carried out at room temperature and nominal voltage for the Altera CCT FPGA on-chip driver circuit. In this approach, the lab measurement was taken using Altera Stratix checkout board. The same on board topology was used in the simulation. Altera introduced Stratix devices, the industry's biggest and fastest FPGAs. Stratix II FPGAs feature a new and innovative logic structure that allows designers to conserve device resources by packing more functionality into less area, dramatically reducing device costs [B20].

The transistor-level Hspice circuit simulation and the lab results differ a little bit because of discontinuities from vias. The parasitic effect of connectors has not been fully accounted for in the simulations. Only the transmission line models were included. The

probe used for probing the signals has long ground wire which introduce some inductance, adding some ringing on the signal. The simulations included the driver circuit (output buffer) model, transmission line model (W-element) and receiver circuit (input buffer) model, and wherever necessary, extracted models of any discontinuities present, such as vias. The simulations also included package models for both the driver and receiver circuits. Appropriate termination scheme was used for each I/O standard. The transmission line model used for the correlation was extracted using a 2D field solver from the layout of the board. The transmission line length was measured from the trace between the driver and the receiver. The board used for correlation had EP1S40F1508 device on it. To obtain the lab measurement, designs were generated in Quartus II software which is the Altera FPGA based software. Required I/O standard and default current strengths were assigned to the pins using the Quartus II software.

Figure 2.21 shows the lab measurement test set-up for the CCT FPGA on-chip driver/receiver circuit. An appropriate parallel on-chip termination was used for the test set-up. CCT FPGA driver/receiver circuit has a core power supply of 1.5 V and an I/O power supply of 3.3V. A rise time of 1 ns was used for the driver circuit.

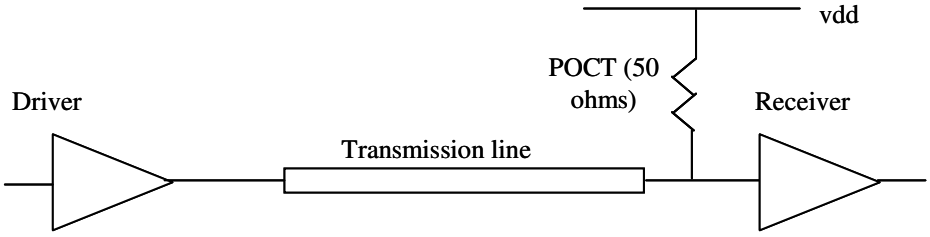


Figure 2.21 CTT parallel on-chip termination setup scheme.

Figure 2.22 shows the voltage waveform at receiver input from lab measurement for Figure 2.21 test set-up. Figure 2.23 shows the voltage waveform at the receiver input from the Hspice transistor-level driver/receiver circuit simulation.

It can be seen that voltage waveform from the transistor-level simulation matches closely with the laboratory measurement waveform. RNN macromodels for driver and receiver circuits have been generated from the data obtained from transistor-level driver and receiver circuits. A detailed explanation of receiver circuit modeling using RNN is discussed later in chapter V. Figure 2.23 shows the voltage waveform at the receiver input for the Hspice RNN macromodel simulation. It can be seen that the RNN macromodel voltage waveform matches well with the transistor-level Hspice simulation.

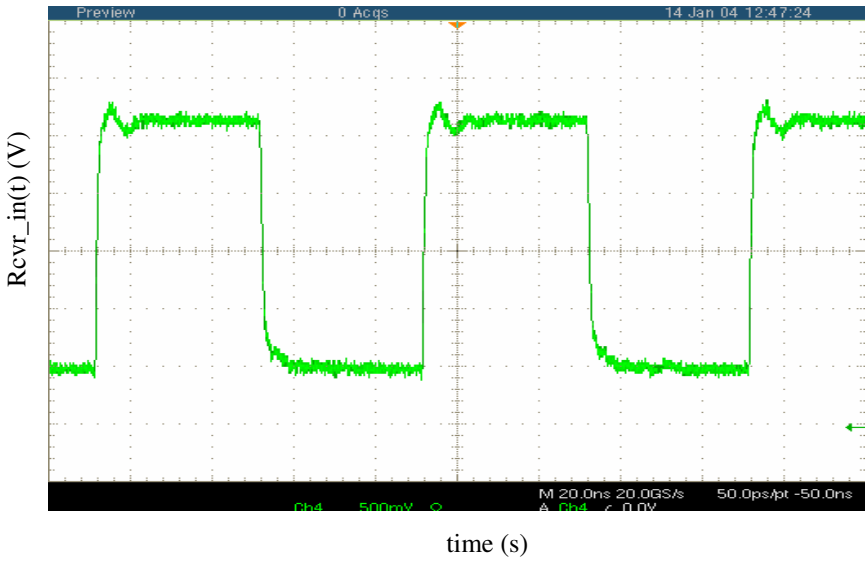


Figure 2.22 CTT parallel on-chip termination measurement result.

It can be seen from Figures 2.22 and 2.23 that the measurement and RNN macromodel results match well. Table 2.2 shows the correlation between the Hspice transistor-level circuit model, measurement result, and RNN macromodel simulation

result for the above test set-up. Table 2.2 compares the result in terms of rise time and peak to peak amplitude. The rise time measurement is from 10% to 90% in the Hspice transistor-level simulation, RNN macromodel simulation and lab result, for the single-ended on-chip CCT driver/receiver circuit. It can be seen that there is 0.3 ns error in the rise time and a 0.1 V error in the modeling the magnitude of the signal. It should be noticed that this error could be partly because of measurement inaccuracies and partly for not incorporating all the parasitic effects in the Hspice transistor-level circuit simulation. There is a 50X speed-up in simulation time for RNN macromodel compared to transistor-level Hspice circuit and a 7X reduction in memory.

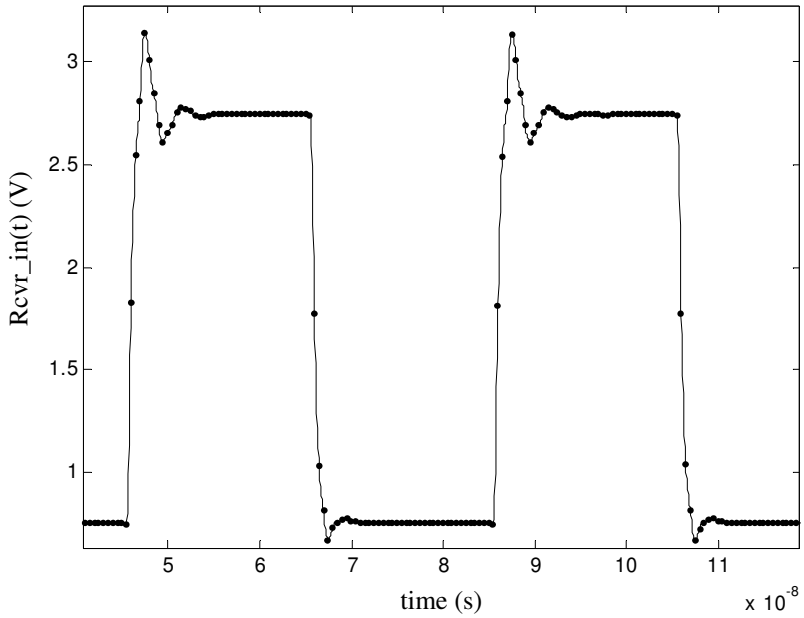


Figure 2.23 Voltage waveform at receiver input for CTT parallel on-chip termination a) transistor-level circuit Hspice simulation result (straight line) and b) RNN macromodel result (dotted line).

Table 2.2 CTT parallel on-chip termination measurement Vs simulation result.

	Measurement	**Simulation**	**Macromodel**
Rise Time (ns)	1.0	0.95	0.97
High (v)	2.15	2.1	2.05

2.7 Pre-emphasis Driver Modeling

Pre-emphasis (Pre-compensation) drivers are important in signal integrity analysis of large digital systems. Pre-emphasis drivers are effective in driving signals though lossy transmission lines. These drivers boost the magnitude of high frequency spectral components of signals, thus ensuring that the signal reaches the receiver without affecting the logic even after attenuation. Pre-emphasis drivers are useful in reducing Inter-Symbol Interference (ISI). Figure 2.24 shows a two-tap FIR driver pre-compensation scheme and its corresponding output waveform can be seen in Figure 2.25. The amount of pre-compensation is shown by the parameter 'R' in Figure 2.24 which varies from 0.0 (no pre-compensation) to < 1.0 for higher levels of pre-compensation. Two-tap pre-emphasis drivers usually incorporate four voltage levels: HIGH, LOW, Strong HIGH and Strong LOW. Pre-emphasis comes into effect only when the signal bit makes transition from one state to another. For example, in Figure 2.25, when the signal makes a transition from LOW to HIGH state, the signal level is boosted to a higher magnitude to reach strong HIGH. When the signal bit stays at the same logic level, pre-compensation stays the same and the signal stays in HIGH state. The same theory holds good for HIGH to LOW transition.

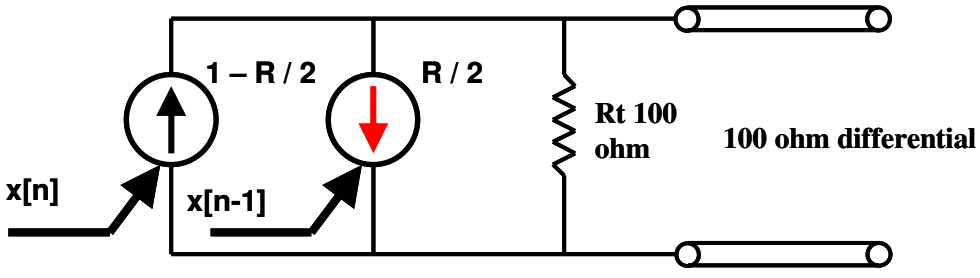


Figure 2.24 A two-tap FIR driver pre-compensation scheme.

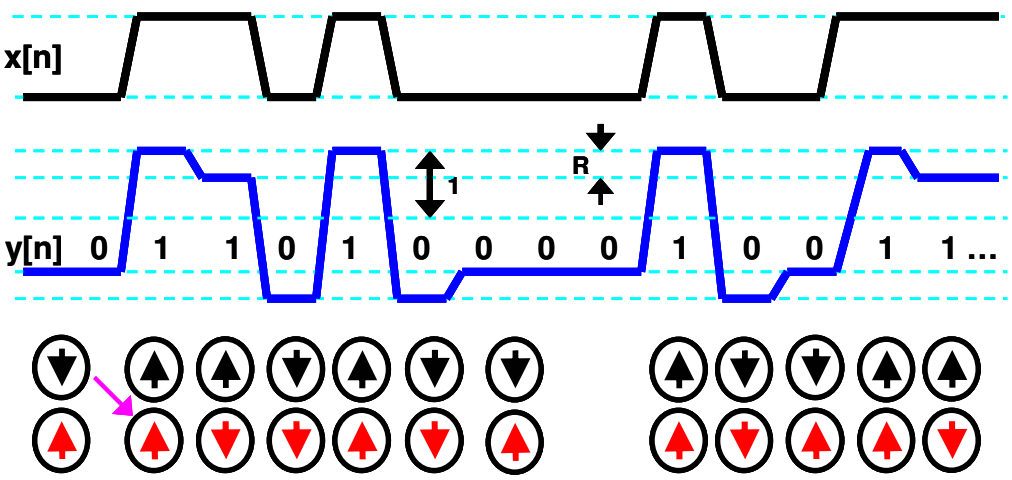


Figure 2.25 Input and output voltage waveforms for a two-tap FIR pre-compensation driver.

Modeling of pre-emphasis drivers is similar to modeling ordinary single-ended drivers. Sub-models f_1 and f_2 that model the driver nonlinearity for input HIGH and LOW states, respectively, have to be carefully estimated. In case of a two-tap FIR pre-compensation driver, f_1 estimates the driver non-linearity for HIGH state and f_2 estimates the non-linearity for LOW state, but care should be taken in ensuring that the range of sub-models f_1/f_2 should be from strong LOW to strong HIGH states. This ensures that sub-models f_1/f_2 can capture the non-linearity of the driver with pre-emphasis when the output voltage swings from strong LOW to strong HIGH and vice-versa. The weighting functions that are calculated from linear inversion of equation (2.13) take into account the effect of pre-emphasis. Figure 2.26 shows the weighting functions w_1 and w_2 for IBM driver ('BBPICMPTERM_A') for two different sets of loads as explained in the previous section. It can be seen from Figure 2.26 that the weighting functions take into account the effect of pre-emphasis.

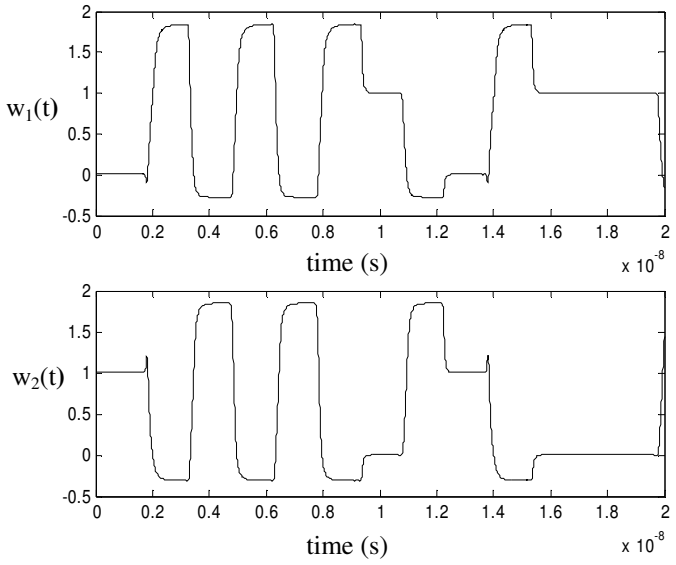


Figure 2.26 Weighting function w_1 and w_2 for IBM pre-emphasis driver.

2.7.1 Test Results

A test case was designed to test the accuracy of SFWFTD modeling technique in modeling pre-emphasis driver circuits. SFWFTD modeling technique was used to model an IBM driver ('BBPICMPTERM_A') that had a power supply voltage of 1.2 V and a rise time of 50 ps. The driver was connected to a 50-ohm ideal transmission line with a delay of 0.5 ns which in turn was terminated with a 2 pF capacitance. A random bit pattern (010101100111…) was given to the driver and voltage waveforms at the near-end and far-end of the transmission lines were measured. It can be seen from Figure 2.27 that SFWFTD macromodel accurately captures the nonlinearity of the driver. A third order cubic polynomial was used in sub-models f_1 and f_2. The value of p and pp were 10*sampling time and 9*sampling time, respectively. A sampling time of 10 ps was used for this test case. SFWFTD macromodel was 47 X faster than the IBM transistor-level

driver model. The simulations were run on an IBM PC 1.5-GHz processor. It can be seen from Figure 2.27 that pre-emphasis driver circuits can be modeled accurately.

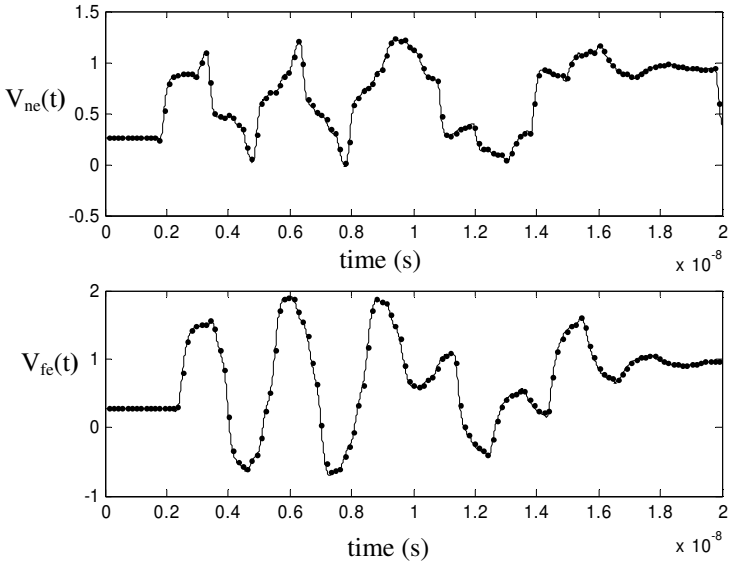


Figure 2.27 Voltage waveform at the near-end ($V_{ne}(t)$) and far-end ($V_{fe}(t)$) of the transmission line from IBM transistor-level driver (straight line) and SFWFTD model (dotted line).

2.8 Summary

In this chapter a comparison study between static characteristic models, SFWFTD models, RBF models, and RNN models was performed. It has been found that static characteristic models and IBIS models can accurately model driver circuits with little or no memory. RBF models tend to consume more simulation time for weakly nonlinear driver circuits. Since IBIS models cannot be extended to multiple ports without losing accuracy, static characteristic models are better replacements for IBIS models. Moderately nonlinear drivers have memory effect in them; therefore, static characteristic models cannot accurately model them. SFWFTD models are better choice to model

moderately nonlinear driver circuits. SFWFTD models consume less simulation time compared to RBF models and at the same time maintain the same accuracy. In addition, SFWFTD models are easy to generate compared to RBF models. RBF models tend to become more and more complex as the rise time decreases, resulting in convergence issues with SPICE. For highly nonlinear driver circuits with large memory or feedback effect, SFWFTD models fail to capture the nonlinearity accurately. Therefore, RNN models are better replacements to SFWFTD models. RNN functions can accurately model nonlinear systems with feedback effect.

In this chapter, it is shown that above modeling approaches can accurately model driver circuits with pre-emphasis effect. Table 2.3 shows the computational speed-up and accuracy of driver macromodels with respect to transistor-level driver circuits. It can be seen that driver macromodels are accurate and at the same time maintain huge computational speed-up.

Table 2.3 Computational speed-up and mean square error for driver macromodels.

Driver Type	Method	Computational Speed-up	Mean Square Error
AGP	SFWFTD Test Case	66X	10^{-4}
SDRAM	RNN Test Case	5X	$(1.2\text{-}1.3)10^{-3}$
PICM	SFWFTD Test Case	47X	10^{-4}

Driver circuits can be modeled using voltage and current information from measurements or from transistor-level spice netlists. It is always simple and efficient to generate macromodels for driver and receiver circuits from transistor-level spice netlists. It has been shown that macromodels generated from SPICE netlists accurately match with the measurement results.

CHAPTER III

MACROMODELING OF DRIVERS WITH MULTIPLE PORTS

In chapter I it has been shown that different macromodels can be used depending on the type of driver circuit being modeled. It is important to note that the modeling approach can be extended to multiple ports to include the effect of power and ground noise. In a driver circuit, power supply noise and ground noise affect the driver output voltage [C1]. Switching of multiple drivers simultaneously results in large transient current through power distribution system (PDS). These currents result in simultaneous switching noise (SSN) that can result in false triggering of logic circuits [C1]-[C3]. The relation between driver output voltage, power supply voltage, and ground voltage has to be accurately captured to model sensitive effects like SSN accurately. Existing driver modeling approaches like IBIS and RBF cannot accurately capture the effect of ground bounce on signal and vice-versa [B8]. It has been shown in Chapter I that IBIS models cannot model SSN accurately. RBF models on the other hand tend to have convergence issues with SPICE when extended to multiple ports. Extension of driver macromodels to include the effect of power and ground ports is the focus of this chapter.

In this chapter, section 3.1 discusses the effect of power distribution network (PDN) on signal and power integrity of high-speed digital systems. Extension of driver modeling technique to multiple ports is described in detail in section 3.2. The effect of power supply port and ground port on the driver output voltage has been discussed for spline

function with finite time difference (SFWFTD) and RNN models. The accuracy of these macromodels has been demonstrated on various test cases in section 3.4. Section 3.5 summarizes the chapter.

3.1 Effect of Power Distribution Network (PDN)

With increasing clock speeds and decreasing supply voltages in today's high-speed digital systems, maintaining the signal and power integrity for future systems is becoming one of the most important issues. The transient current injected into the power distribution planes builds up energy due to the resonant cavity and causes voltage fluctuations and circuit delays [C12]. This leads to unwanted effects on the PDS such as ground bounce, power supply compression, and electromagnetic interference (EMI). A major problem in the (PDS) is SSN induced by power/ground plane inductance. The purpose of the PDS is to supply a constant, noise-free voltage to the integrated circuits in a system. To achieve this, the PDS must exhibit very low impedance over a large frequency bandwidth where the noise voltages exist. As a result, an important area in high-speed digital systems is design and modeling of power/ground planes arising in PDNs.

Complementary metal oxide semiconductor (CMOS) microprocessors and application specific integrated circuits (ASICs) in a modern digital system consist of a large number of internal circuits and external circuits (I/O drivers). A power distribution network (PDN) for the typical high-speed digital system is shown in Figure 3.1. The PDN is used to deliver power to core logic and I/O circuits in the modern system [C2].

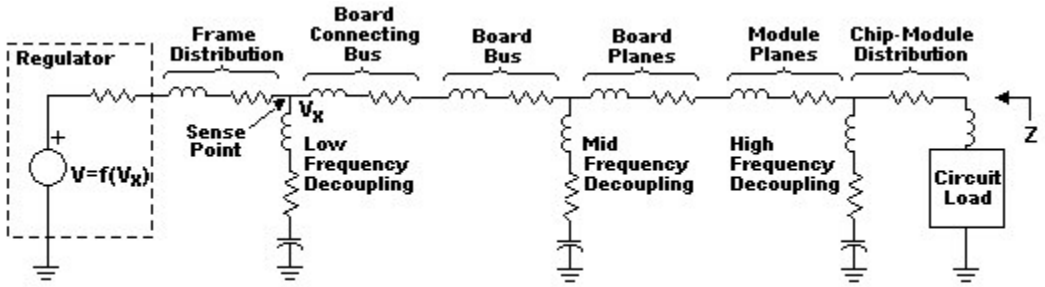


Figure 3.1 Power distribution network (PDN) for the typical high-speed digital system.

Figure 3.2 shows a simple scenario of charging and discharging of an inverter.

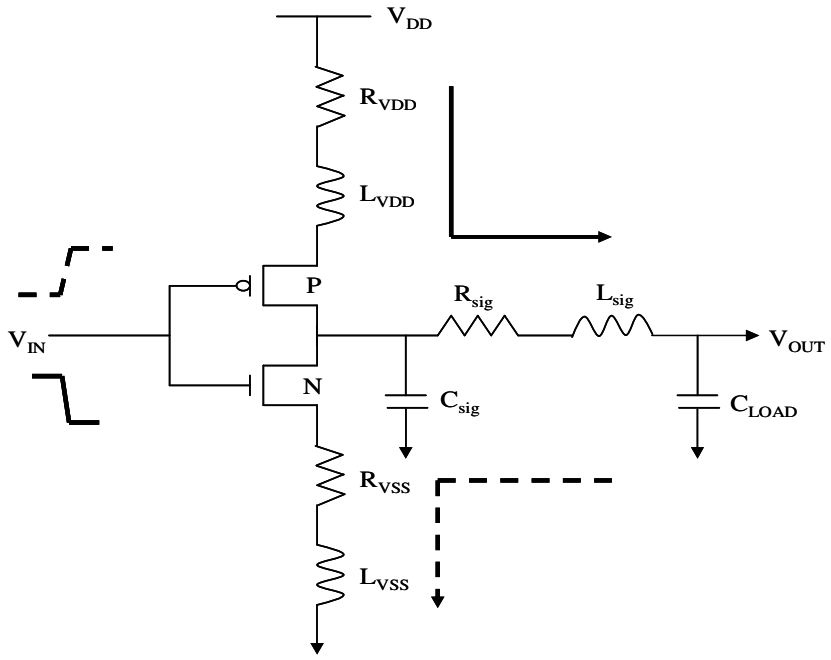


Figure 3.2 Switching of a CMOS inverter.

It can be seen that when the input switches from HIGH to LOW, there is current surge from PDS. When many drivers switch from LOW to HIGH or HIGH to LOW at the same time, very large transient currents must be delivered by the PDS. Even a small inductance in the PDS will generate a noise voltage, which could cause false triggering of

other gates, which is commonly known as the SSN problem. It has been recognized that the power supply noise induced by large numbers of simultaneously switching circuits in the power distribution network can limit their performance [C4]-[C5].

The main issues associated with power distribution are the IR voltage drops and the inductive effects [C6]. When DC current (I) is supplied to circuit loads, the finite resistance (R) of the package metal layers, which includes vias, interconnects and power/ground planes, causes a voltage drop given by Ohm's Law. Since the IR voltage drop can vary across the chip, the supply voltage for all the circuits may not be the same. This variation of the DC supply voltage can cause the false transitioning of the circuits for spurious input signals. During the high-to-low or low-to-high transitions of the circuits, the inductive effect occurs more seriously due to a time-varying current. Since metal layers are inherently inductive, the time varying current causes a voltage fluctuation to the supply voltage. Hence, the supply voltage oscillates around the DC level with time [C13]. This inductive effect leads to the following effects:

1. The inductance of the power distribution network causes the circuits to slow down by introducing excessive time delays to the supply voltage of the circuits as shown in Figure 3.3.
2. Noise glitches on the power supply may cause false switching of the circuits on both the sending and receiving chips. Both these effects should be minimized for increasing the reliability of systems.

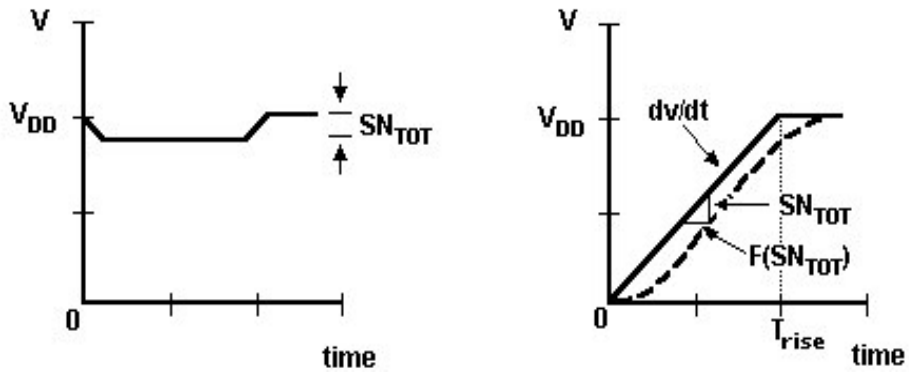


Figure 3.3 Graphical derivation of signal delay due to switching noise [C2].

With advances in silicon technology, power supply voltage has reduced according to the scaling rules while the amount of power required has increased. As a result, the current delivery requirement for the power distribution network has increased greatly and the tolerance for the power supply noise has decreased. This has reduced the tolerance for error in modeling SSN, hence, there is a necessity for macromodels of driver circuits that can accurately model sensitive effects like SSN.

3.2 Non-Ideal Power Supply and Ground Nodes

3.2.1 Non-Ideal Power Supply Node

If the power supply of the driver circuit is ideal, then the issue of SSN when multiple drivers switch does not arise. But in reality, the power supply of a driver circuit is non-ideal. Figure 3.4 shows a schematic of a driver circuit where the power supply is non-ideal.

To incorporate the effect of the power supply node (v_{dd}), a relation should be drawn between driver power supply current (i_{dd}) and driver power supply voltage. However, the

driver power supply current is not only a function of driver power supply voltage but also a function of driver output voltage (v_o) as shown below:

$$i_{dd}(k) = w_{1,dd}(k) f_{1,dd}(v_o(k), v_{dd}(k)) + w_{2,dd}(k) f_{2,dd}(v_o(k), v_{dd}(k)) + w_{3,dd}(k) \tag{3.1}$$

where $f_{1,dd}$ and $f_{2,dd}$ are the power supply sub-models that relate the power supply current to power supply voltage for driver inputs HIGH and LOW, respectively [C7]. In equation (3.1), weighting functions $w_{1,dd}$, $w_{2,dd}$, and $w_{3,dd}$ help sub-models $f_{1,dd}$ and $f_{2,dd}$ in transitioning from one state to another.

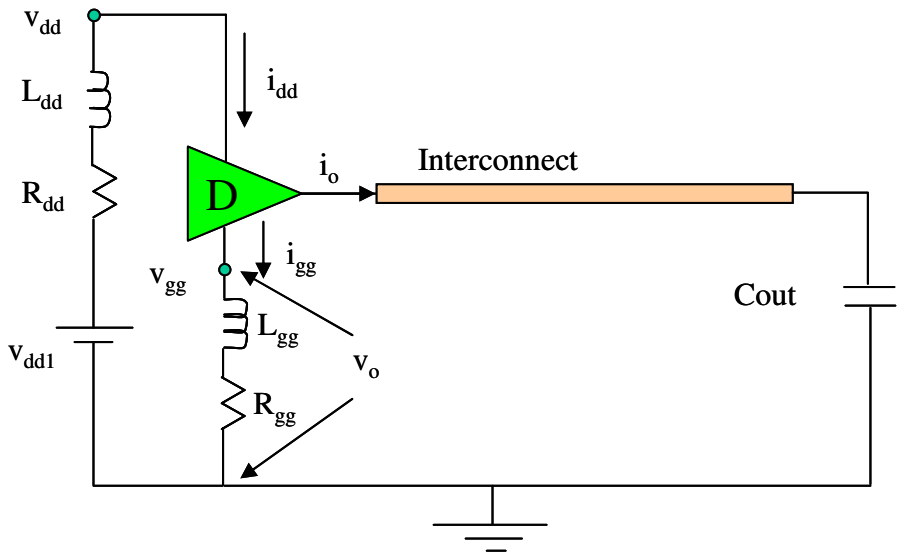


Figure 3.4 A driver with a non-ideal power supply v_{dd} is connected to a transmission line.

3.2.1.1 Spline Function with Finite Time Difference (SFWFTD) Method

Sub-models $f_{1,dd}$ and $f_{2,dd}$ in equation (3.1) can be expressed as a combination of static sub-models ($f_{1,sd}$ and $f_{2,sd}$) and dynamic sub-models ($f_{1,dy}$ and $f_{2,dy}$) as shown below:

$$f_{n,dd}(v_o(k), v_{dd}(k)) = f_{n,sd}(v_o(k), v_{dd}(k)) + f_{n,dy}(v_o(k), v_{dd}(k)); \quad n = 1,2 \tag{3.2}$$

The static sub-models $f_{1,sd}$ and $f_{2,sd}$ can be calculated through a double *dc* sweep at driver output and at driver power supply. This relationship can be represented as:

$$f_{n,sd}(v_o(k), v_{dd}(k)) = (a_{n,r}{v_o}^{r}(k) + a_{n,(r-1)}{v_o}^{(r-1)}(k) + \ldots + a_{n,0}) * (b_{n,s}{v_{dd}}^{s}(k) + b_{n,(s-1)}{v_{dd}}^{(s-1)}(k) + \ldots + b_{n,0}) \quad (3.3)$$

where *a* and *b* are constants whose values depend on the driver being modeled. The value of *n* is one or two depending on whether the driver input is HIGH or LOW. The values of *r* and *s* are positive and usually less than five for most driver circuits.

The dynamic sub-models $f_{1,dy}$ and $f_{2,dy}$ can be expressed as:

$$f_{n,dy}(v_o(k), v_{dd}(k)) = p_{n,d}\Delta i'_{n,o} + q_{n,d}\Delta i'_{n,dd} + pp_{n,d}\Delta i''_{n,o} + qq_{n,d}\Delta i''_{n,dd} \ldots \quad (3.4)$$

where n = 1,2

In equation (3.4), dynamic characteristic sub-model ($f_{n,dy}$) constants $p_{n,d}$, $q_{n,d}$, $pp_{n,d}$ and $qq_{n,d}$ are estimated by connecting PWL voltage sources both at the driver output and at the driver power supply and measuring the error between static power supply current and transistor- level driver power supply current. Figure 3.5 shows PWL voltage sources connected at driver output and power supply, respectively, for IBM driver ('HSTL_A') when the driver input is held HIGH and LOW.

In equation (3.4), $p_{n,d}$, $q_{n,d}$, $pp_{n,d}$ and $qq_{n,d}$ depend on the driver being modeled. Termination of the driver output with two different loads results in two different equations. Since, three unknown weighting functions exist, one method to solve the problem is by assuming $w_{1,dd} = (1 - w_{2,dd})$. Thus, weighting functions $w_{1,dd}$, $w_{2,dd}$ and $w_{3,dd}$ can be calculated once $f_{1,dd}$ and $f_{2,dd}$ are estimated for two different loads as shown below:

$$\begin{bmatrix} w_{1dd} \\ w_{3dd} \end{bmatrix} = \begin{bmatrix} f_{1a,dd} - f_{2a,dd} & 1 \\ f_{1b,dd} - f_{2b,dd} & 1 \end{bmatrix}^{-1} \begin{bmatrix} i_a - f_{2a,dd} \\ i_b - f_{2b,dd} \end{bmatrix} \tag{3.5}$$

Figure 3.6 shows the weighting functions generated for IBM('HSTL_A') driver circuit using equation (3.5).

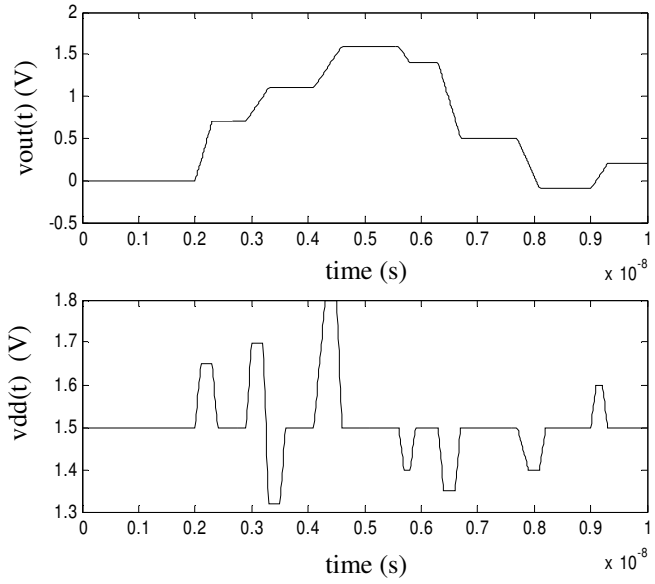


Figure 3.5 PWL voltage sources connected at the driver output and the power supply node to calculate the dynamic characteristics of SFWFTD approach.

Similarly, driver output current is not only a function of driver output voltage but also a function of driver supply voltage. The procedure for estimating driver output current is similar to estimating driver power supply current. This can be calculated as:

$$i_o(k) = w_1(k) f_1(v_o(k), v_{dd}(k)) + w_2(k) f_2(v_o(k), v_{dd}(k)) \tag{3.6}$$

where $$f_n(v_o(k), v_{dd}(k)) = f_{n,s}(v_o(k), v_{dd}(k)) + f_{n,d}(v_o(k), v_{dd}(k)); n = 1,2 \tag{3.7}$$

$$\begin{aligned} f_{n,s}(v_o(k), v_{dd}(k)) = (aa_{n,t} {v_o}^t(k) + aa_{n,(t-1)} {v_o}^{(t-1)}(k) + ... + aa_{n,0}) * \\ (bb_{n,u} {v_{dd}}^u(k) + bb_{n,(u-1)} {v_{dd}}^{(u-1)}(k) + ... + bb_{n,0}) \end{aligned} \tag{3.8}$$

where $n = 1, 2;\ t >= 1$ and $u >= 1$

$$f_{n,d}(v_o(k), v_{dd}(k)) = p_n \Delta i'_{n,o} + q_n \Delta i'_{n,dd} + pp_n \Delta i''_{n,o} + qq_n \Delta i''_{n,dd} \ldots ;\ n = 1,2 \quad (3.9)$$

where p_n, q_n, pp_n *and* qq_n are constants.

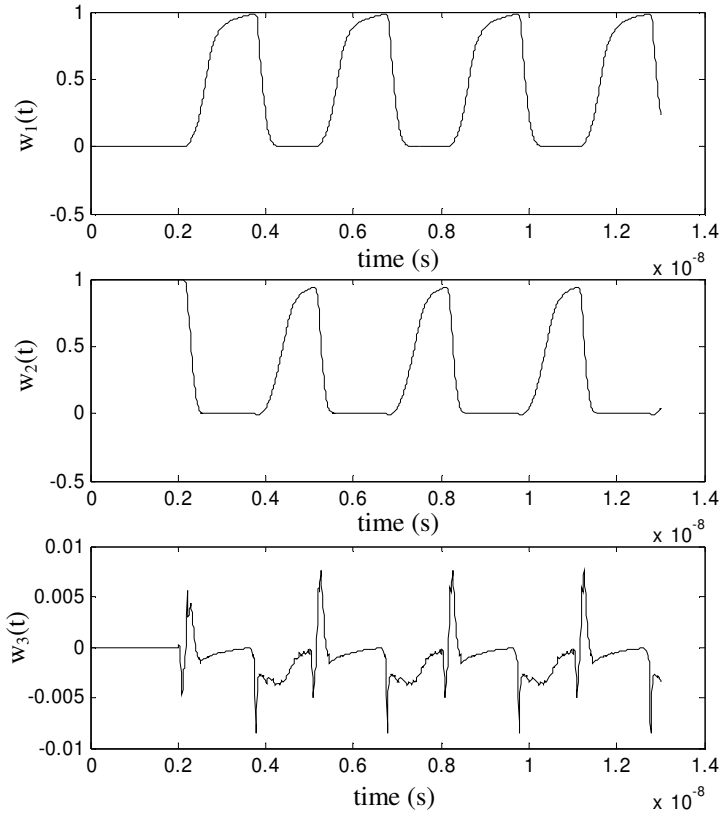


Figure 3.6 Weighting functions w_{1dd}, w_{2dd}, and w_{3dd} that help sub-models f_{1dd} and f_{2dd} transition from one state to another.

Equations (3.6), (3.7), (3.8) and (3.9) provide the relation between driver output current in terms of driver output voltage and driver power supply voltage.

Figure 3.7 shows the resultant IBM driver output current and the power supply current from PWL voltage sources connected at driver output and power supply, as

shown in Figure 3.5 when the driver input is HIGH. It can be seen from Figure 3.7 that the SFWFTD models the driver output and power supply current accurately.

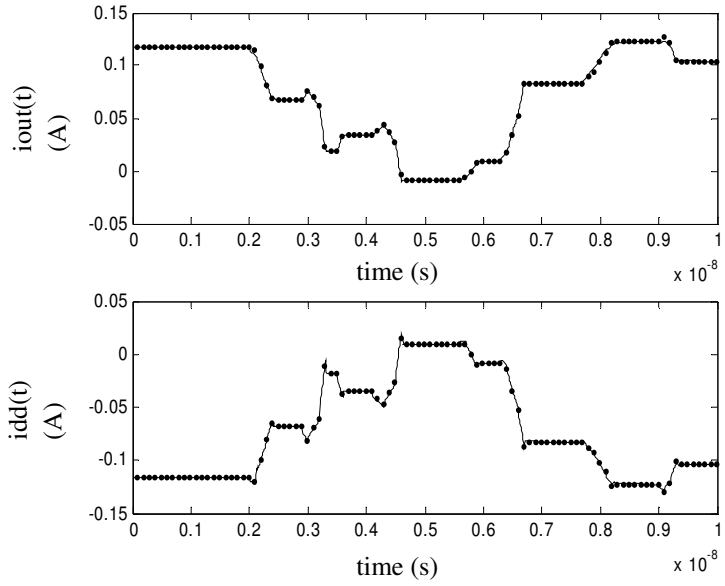


Figure 3.7 Driver output current and power supply current from transistor level driver (straight line) and SFWFTD model (dotted line) when the driver input is HIGH.

Similarly, SFWFTD models can model the driver output current and power supply current accurately when the driver input is LOW.

3.2.1.1 Recurrent Neural Network Approach

RNN modeling approach can also be extended to multiple-ports to incorporate non-ideal power supply node using the above technique. Sub-models $f_{1,dd}$ and $f_{2,dd}$ of equation (3.1) and f_1 and f_2 of equation (3.6) are now represented using hyperbolic tangential function as shown below:

$$f_n = \sum_{k=1}^{M} b_{kj}\, g\left(\sum_{j=1}^{N} a_{ji} x_i + a_{oj}\right) + b_{ok}\,;\; n = 1,2 \tag{3.10}$$

where

$$g(x) = (e^{x} + e^{-x})/(e^{x} - e^{-x}) \tag{3.11}$$

where, b and a are weights associated with the neural network, N represents number of hidden neurons, M represents number of outputs and x is the regressor vector as shown:

$$x(k) = \begin{Bmatrix} i_o(k-1), i_o(k-2), ..., i_o(k-r) \\ v_o(k), v_o(k-1), ..., v_o(k-r) \\ i_{dd}(k-1), i_{dd}(k-2), ..., i_{dd}(k-r) \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-r) \end{Bmatrix} \tag{3.12}$$

The regressor vector, x, takes into account both the past and present samples of driver output current, output voltage, power supply current, and power supply voltage [C8]. RNN is trained using modified back propagation through time (BPTT) algorithm [B19]. RNN sub-models can also model the driver power supply current and driver output current accurately.

A spice netlist of SFWFTD and RNN modeling approach can be generated from voltage-dependent voltage sources, voltage-dependent current sources and state equations as explained in Chapter II.

3.2.2 Non-Ideal Power Supply and Ground Nodes

In real high speed systems, there is always a local ground and a global ground. Any voltage fluctuations on the local ground can affect the driver output voltage. Figure 3.3 shows a schematic where the driver has non-ideal power supply and ground.

In order to model the effect of ground node, a relation between driver output current, driver output voltage, driver ground voltage, and driver power supply voltage should be accurately captured. Equation (3.6) represents the driver output current relation. Sub-

models f_1 and f_2 are modeled using RNN networks as shown in equation (3.10). The regeressor vector x is represented as shown below:

$$x(k) = \begin{Bmatrix} i_o(k-1), i_o(k-2), ..., i_o(k-r1) \\ v_o(k), v_o(k-1), ..., v_o(k-r2) \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-r3) \\ v_{gg}(k), v_{gg}(k-1), ..., v_{gg}(k-r4) \end{Bmatrix} \tag{3.13}$$

It can be seen from equation (3.13) that x contains all the previous time instances of driver output voltage (v_o), driver power supply voltage (v_{dd}), and driver ground voltage (v_{gg}). The selection of scalar constants, *r1*, *r2*, *r3,* and *r4*, depends on the driver circuit being modeled.

To model f_1 and f_2, the driver output current is measured when PWL voltage sources are connected at driver output, driver power supply node, and driver ground port. These PWL voltage sources excite both the static and dynamic characteristics of the driver circuit. Figure 3.8 shows a typical PWL voltage source setup used to model IBM DDR2 driver circuit. It can be seen from Figure 3.8 that the PWL voltage sources have different rise times and different magnitudes to excite the nonlinearity of the driver circuit.

Figure 3.9 shows IBM DDR2 driver output current (straight line) when the driver input is set HIGH. It can be seen that the driver output current was accurately modeled using RNN network (dotted line). A RNN network with two hidden neurons was required to model the driver output current. Similarly, the driver output current can also be modeled for driver input LOW.

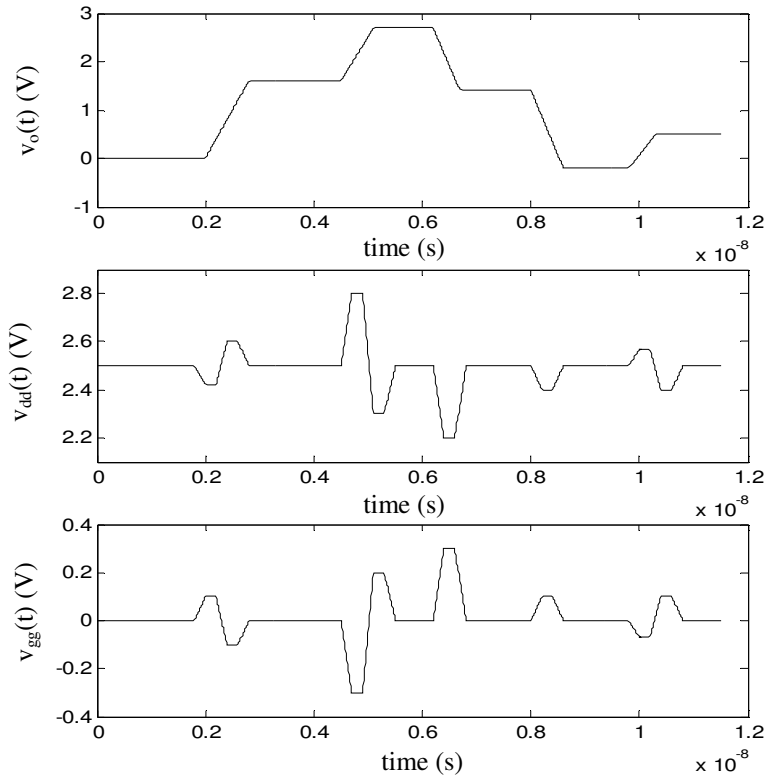


Figure 3.8 PWL voltage sources connected at driver output, driver power supply, and driver ground ports.

Weighting functions w_1 and w_2 help in transitioning sub-models f_1 and f_2 from one logic state to another. Since two unknowns are present, two equations are required to solve for two unknowns. The driver output current is measured for two different terminations of driver output. The driver output is terminated with a resistive load and the driver sub-models f_{1a} and f_{2a} are estimated and the sub-models f_{1b} and f_{2b} are estimated when the driver output is terminated with a resistance and a DC voltage source. Weights functions w_1 and w_2 can be estimated as:

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} f_{1a} & f_{2a} \\ f_{1b} & f_{2b} \end{bmatrix}^{-1} \begin{bmatrix} i_a \\ i_b \end{bmatrix} \tag{3.14}$$

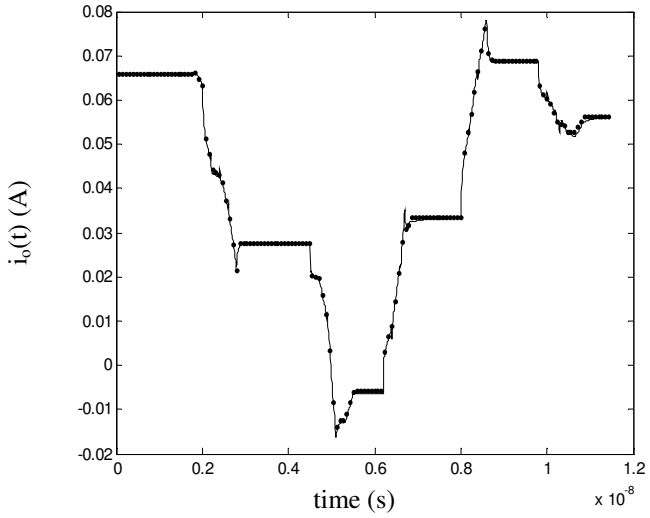


Figure 3.9 Driver ground current i_o when the driver input is held HIGH from IBM DDR2 driver (straight line) and RNN model (dotted line).

To model the power supply noise, a relation between driver power supply current and driver power supply voltage is drawn as shown in equation (3.1). Sub-models f_{1dd} and f_{2dd} are modeled using RNN networks as shown in equation (3.10) and (3.11). The regressor vector *xdd* contains the present and previous samples of driver output voltage and power supply voltage as shown below:

$$xdd(k) = \begin{Bmatrix} i_{dd}(k-1), i_{dd}(k-2), ..., i_{dd}(k-p1) \\ v_o(k), v_o(k-1), ..., v_o(k-p2) \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-p3) \end{Bmatrix} \tag{3.15}$$

The values of scalar constants *p1*, *p2,* and *p3* in equation (3.15) are dependent on the kind of driver circuit being modeled. The driver power supply current for driver input LOW and HIGH is measured when PWL voltage sources shown in Figure 3.8 are connected at driver output, power supply node, and ground ports. Figure 3.10 shows the driver power supply current (i_{dd}) when the driver input is set HIGH. It can be seen that the RNN model

(dotted line) can accurately model the power supply current using two hidden neurons. Similarly, the power supply current can be modeled for driver input LOW.

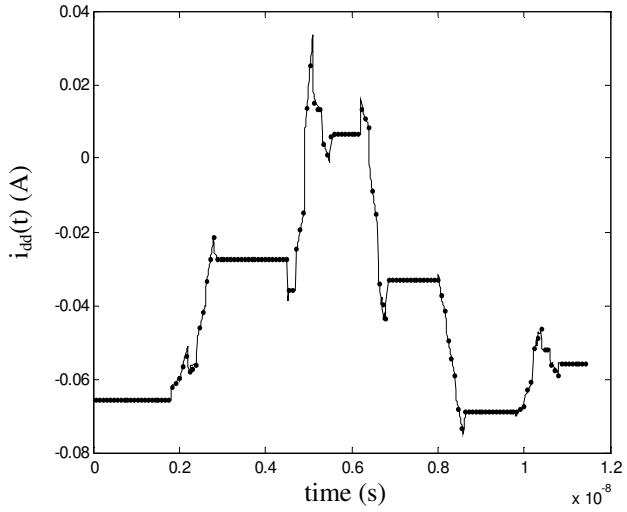


Figure 3.10 Driver ground current i_{dd} when the driver input is held HIGH from IBM DDR2 driver (straight line) and RNN model (dotted line).

Weighting functions w_{1dd}, w_{2dd}, and w_{3dd} help in transitioning sub-models f_{1dd} and f_{2dd} from one state to another. Since three unknowns are present in equation (3.1) and two equations are present to estimate them, the problem is solved by assuming w_{1dd} = (1 - w_{2dd}).

The ground noise is modeled by drawing a relation between driver ground current and driver ground voltage as shown below:

$$i_{gg}(k) = w_{1gg}(k) f_{1gg}(k) + w_{2gg}(k) f_{2gg}(k) + w_{3gg} \tag{3.16}$$

$$xgg(k) = \begin{Bmatrix} i_{gg}(k-1), i_{gg}(k-2), ..., i_{gg}(k-q1) \\ v_o(k), v_o(k-1), ..., v_o(k-q2) \\ v_{gg}(k), v_{gg}(k-1), ..., v_{gg}(k-q3) \end{Bmatrix} \tag{3.17}$$

In equation (3.16), sub-models f_{1gg} and f_{2gg} relate the driver ground current to driver ground voltage and output voltage for driver input HIGH and LOW, respectively. The

values of scalar constants *q1*, *q2,* and *q3* depend on the kind of driver being modeled. Sub-models f_{1gg} and f_{2gg} are estimated by measuring the driver ground current using the PWL voltage sources shown in Figure 3.8. Figure 3.11 shows the driver ground current when the driver input is HIGH. It can be seen that the RNN model (dotted line) accurately models the driver ground current.

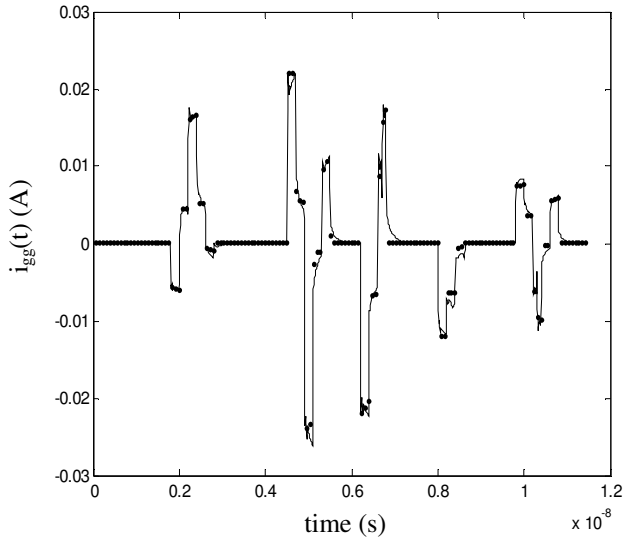


Figure 3.11 Driver ground current i_{gg} when the driver input is held HIGH from IBM DDR2 driver (straight line) and RNN model (dotted line).

Once sub-models f_{1gg} and f_{2gg} are estimated using the PWL voltage sources for driver input HIGH and LOW, weighting functions w_{1gg}, w_{2gg}, and w_{3gg} are calculated. The procedure to calculate the weighting functions is similar to the one used to estimate weighting functions for power supply node. Since three weights are present for two equations, it is assumed that $w_{1gg} = 1 - w_{2gg}$.

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} f_{1a} - f_{2a} & 1 \\ f_{1b} - f_{2b} & 1 \end{bmatrix}^{-1} \begin{bmatrix} i_a - f_{2a} \\ i_b - f_{2b} \end{bmatrix} \tag{3.18}$$

Figure 3.12 shows weighting functions w_{1gg}, w_{2gg}, and w_{3gg} estimated for IBM DDR2 driver circuit [C11].

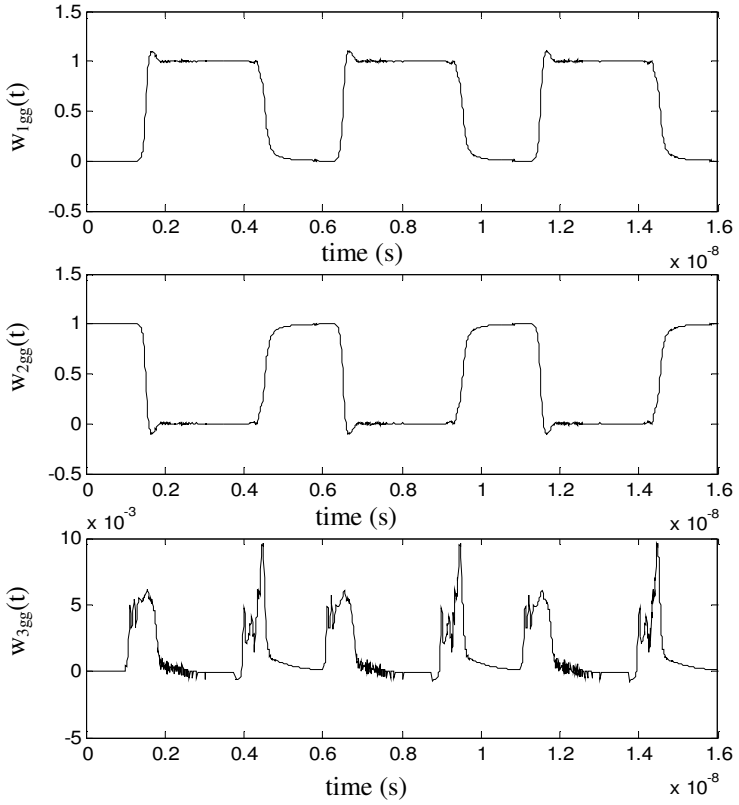


Figure 3.12 Weighting functions w_{1gg}, w_{2gg}, and w_{3gg} for IBM DDR2 driver.

Once all the weighting functions for driver output, driver power supply node, and driver ground port are calculated, a spice netlist for RNN macro-model was generated. All the weights are represented using PWL voltage sources in the spice netlist. Sub-models for driver output, power supply node, and ground port are represented using voltage-dependent voltage sources and voltage-dependent current sources.

3.3 Test Results

Case 1: A test case was generated in which 16 IBM drivers (HSTL_A) were connected to a plane pair model which was generated using the cavity resonator method [C9]. In the cavity resonator method, the impedance between two ports on the plane can be expressed as:

$$Z_{ij}(w) = \sum_{n=0}^{\infty}\sum_{m=0}^{\infty} \frac{N_{mni}N_{mnj}}{jwC_{mn} + \dfrac{1}{jwL_{mn}} + G_{mn}} \tag{3.19}$$

The equivalent circuit for equation (3.19) can be implemented by using parallel resonant circuits and ideal transformers [C9]-[C10].

The cavity resonator plane pair had dimensions of 10 cm × 6 cm. It had six ports on each plane, V_{dd} and *Gnd*, as shown in Figure 3.13. Sixteen drivers were connected at port one and all the drivers were connected to 50-ohm transmission lines which in turn were terminated by 1 pF capacitors. All capacitors were terminated at port three. The power supply node v_{dd} was at port four.

Three additional ports were used for probing. All 16 drivers were identical. The resulting SSN was calculated using both actual transistor-level driver model and SFWFTD model. Figure 3.14 shows the near-end and far-end voltage waveforms on transmission line one. It can be clearly seen that the modeled and actual transistor-level model waveforms match accurately.

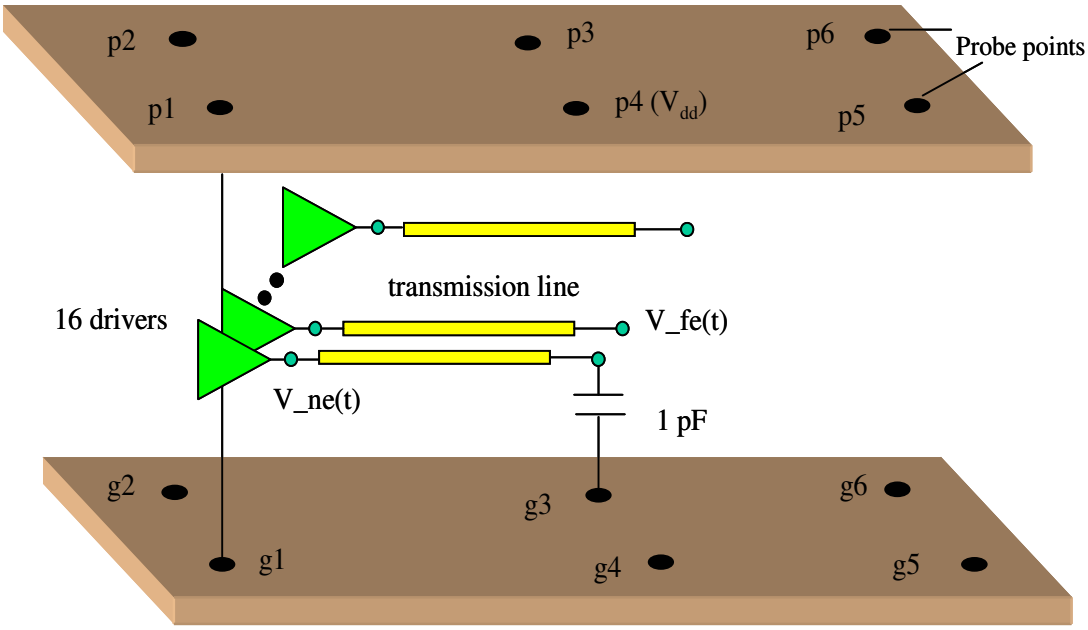


Figure 3.13 Plane pair model generated using cavity resonator method. Both planes have six ports each.

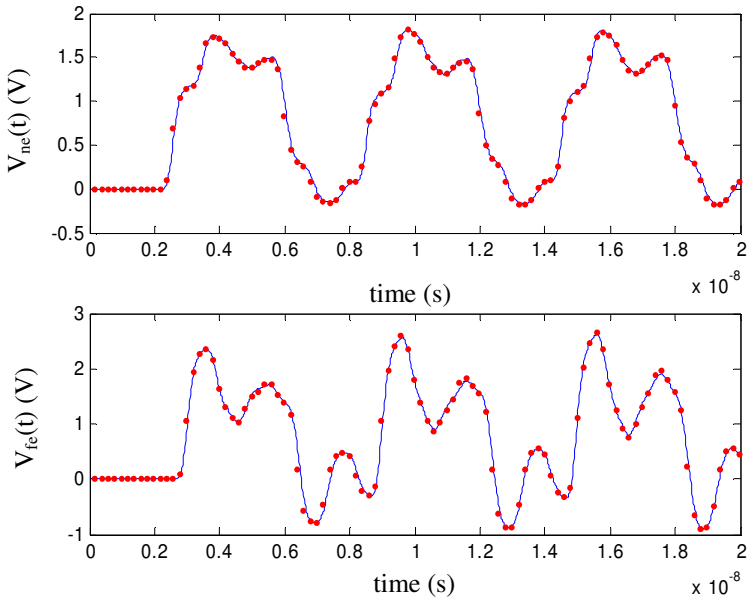


Figure 3.14 Near-end ($V_{ne}(t)$) and far-end ($V_{fe}(t)$) voltage waveforms on transmission line # 1 for actual transistor level driver model (straight line) and SFWFTD model (dotted line).

Figure 3.15 shows the SSN at ports one, three, and five, respectively. It can be clearly seen that the SSN was accurately modeled using SFWFTD method.

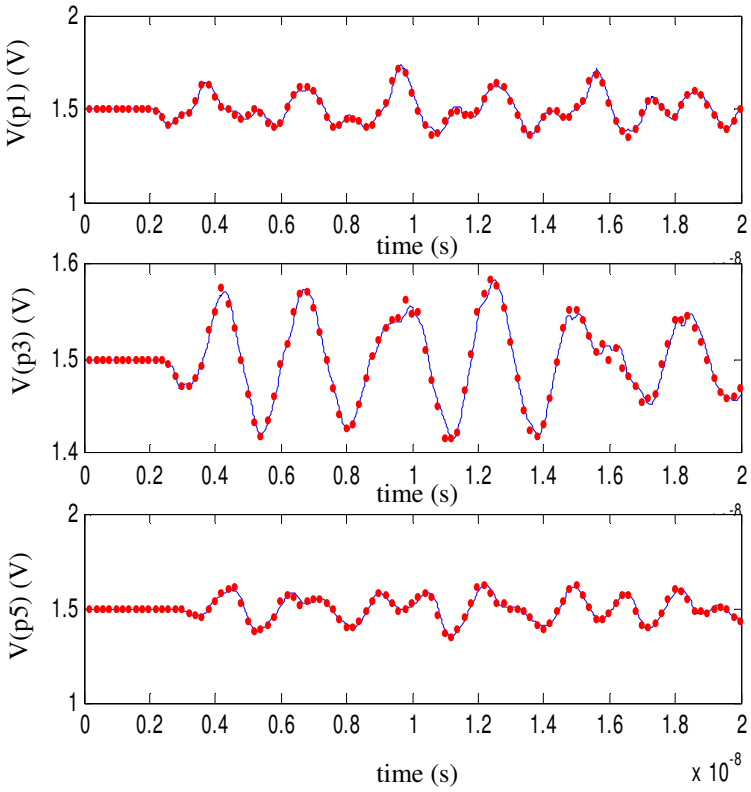


Figure 3.15 Simultaneous Switching Noise (SSN) at ports p1, p3, and p5 when 16 identical drivers are switching together. SSN from actual transistor level driver model (straight line) and SFWFTD model (dotted line).

It is known that SSN can affect the integrity of a signal. Figure 3.16 shows the voltage waveforms at the near-end and the far-end of the transmission line for ideal power supply and non-ideal power supply nodes. The distortion of the voltage waveform in the presence of SSN can be seen from Figure 3.16. The mean square error (MSE) between the signals with and without SSN is of the order 10^{-2}.

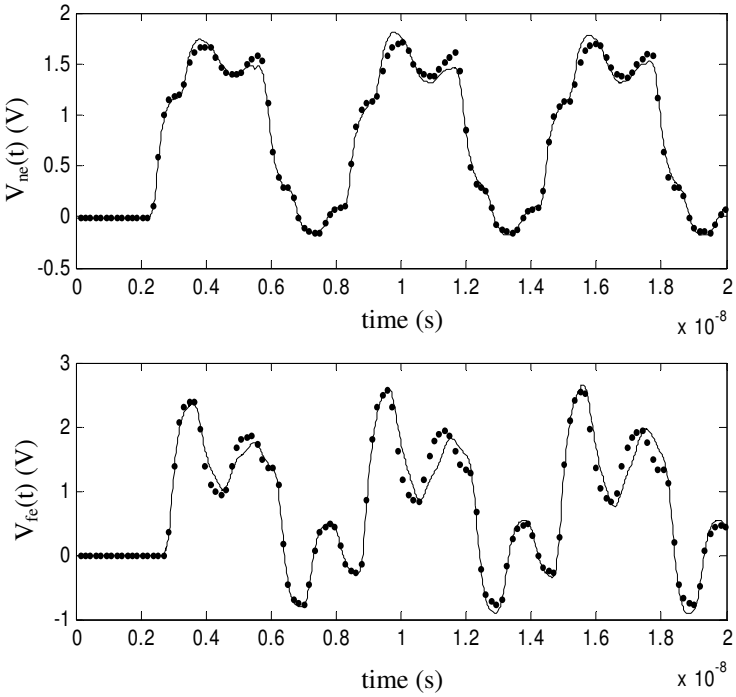


Figure 3.16 Near-end ($V_{ne}(t)$) and far-end ($V_{fe}(t)$) voltage waveforms on transmission line # 1 for SFWFTD model with SSN (straight line) and SFWFTD model with ideal v_{dd} (dotted line).

Case 2: A comparison study was done between simulation time taken by the actual IBM ('HSTL_B') transistor-level driver circuit and SFWFTD model when the number of drivers switching were increased from 2 to 16. Figure 3.17 shows the plot for simulation time consumed Vs. number of drivers switching.

It can be seen that when 16 drivers were switching, the SFWFTD model is 25-30 times faster than the transistor level driver circuit. These simulations were carried out on SUN ULTRA-10 machine for 30 ns time period.

Figure 3.18 shows the percentage peak noise error when the number of drivers switching was increased from 2 to 16. The percentage peak noise error between the actual transistor-level driver circuit and SFWFTD model was calculated when the number of drivers switching was increased from 2 to 16 using the relation shown:

$$\%\,Peak\ Noise\ Error = \frac{(Peak\ Noise\ of\ HSTL\ driver - Peak\ Noise\ of\ Spline\ model)}{Peak\ Noise\ of\ HSTL\ driver} * 100 \quad (3.20)$$

It can be clearly seen that when 16 drivers are switching, the percentage peak noise error was less than 3 %. From these results it can be seen that SFWFTD method can accurately capture SSN effect at the same time taking significantly less time for simulation.

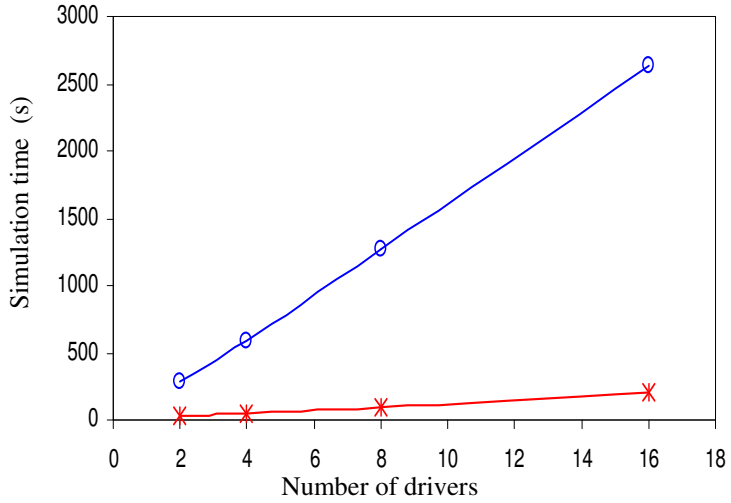


Figure 3.17 Simulation time Vs. number of drivers switching for transistor level driver (0) and SFWFTD model (*).

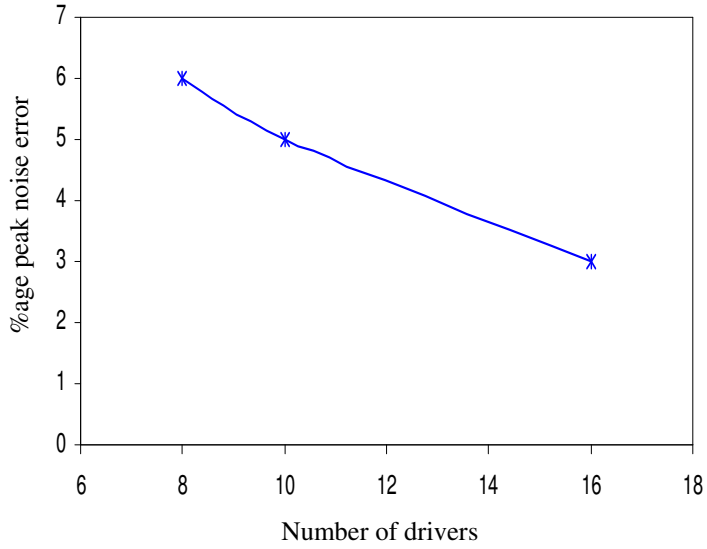


Figure 3.18 Percentage peak noise error Vs. number of drivers switching for SFWFTD model.

Case 3: In this test case, four IBM drivers ('DDR2') were connected to a plane pair modeled using the cavity resonator method. IBM 'DDR2' driver has a power supply voltage of 2.5 volts and operational frequency of 250 MHz with a rise time of 1.25 ns. The plane pair had dimensions of 6 cm X 4 cm with four ports on V_{dd} plane and four ports on *Gnd* plane. All the drivers were identical, driving ideal 50-ohm transmission lines and were connected at port one. The power supply node was at port three. All the transmission lines were terminated at port two using a 2 pF capacitance and port four was used for probing. Figure 3.19 shows the plane pair used to model the power supply noise when four drivers are simultaneously switching. The IBM driver was modeled using RNN. The regressor vector x consists of present samples of power supply voltage and driver output voltage and past samples of driver output current, power supply current, power supply voltage and output voltage. The RNN model for sub-functions $f_{1d,2d}$ required one hidden layer with two hidden neurons. Modified BPTT training algorithm was used to estimate the weights of the RNN model [B19]. It can be seen from Figure 3.20 that RNN model captures SSN accurately when all four drivers are switching simultaneously.

RNN model is 6-7 times faster than the transistor level model when the time domain simulations were carried out for three cycles. The computational time speed up between the RNN model and actual transistor level model increases with the increase in complexity of the circuit.

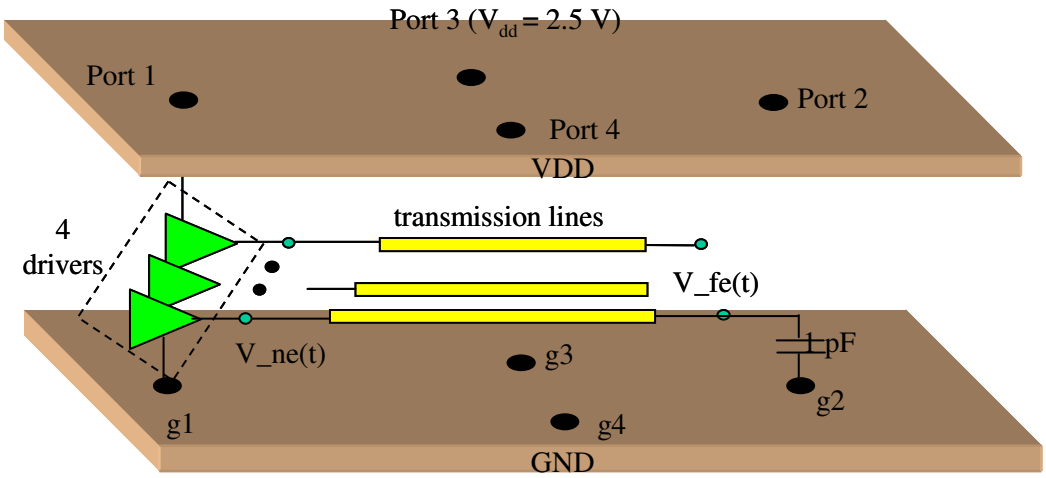


Figure 3.19 Plane pair model generated using cavity resonator method. Both planes have four ports each.

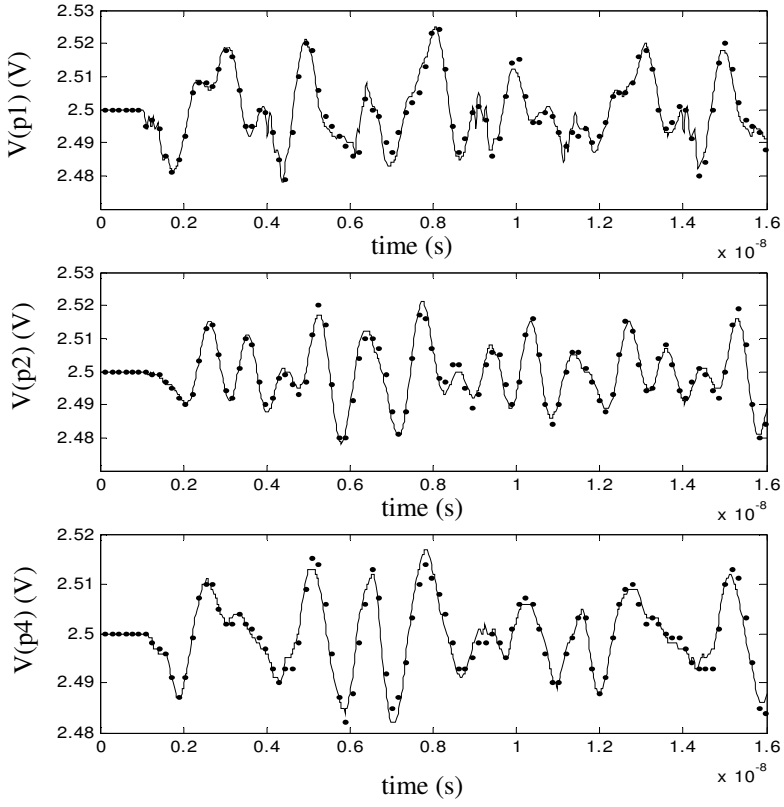


Figure 3.20 Simultaneous Switching Noise (SSN) at ports p1, p2, and p4 when four identical drivers are switching together. SSN from actual transistor level driver model (straight line) and RNN model (dotted line).

Case 4: In this test case, six IBM DDR2 driver circuits were connected to six ideal 50-ohm transmission lines that were terminated with 2 pF capacitor. The transmission lines line a delay of 0.5ns. The input pulse given to the driver circuits has a rise time of 0.25 ns. The driver circuit's ground port was connected to ideal ground through a 0.1 nH inductor. The voltage at the near-end and far-end of the transmission line one is measured for both the transistor-level circuit and the RNN model. It can be seen from Figure 3.21 that both the near-end and far-end waveforms match accurately. The noise at the ground port was also plotted for transistor level driver circuit and RNN model.

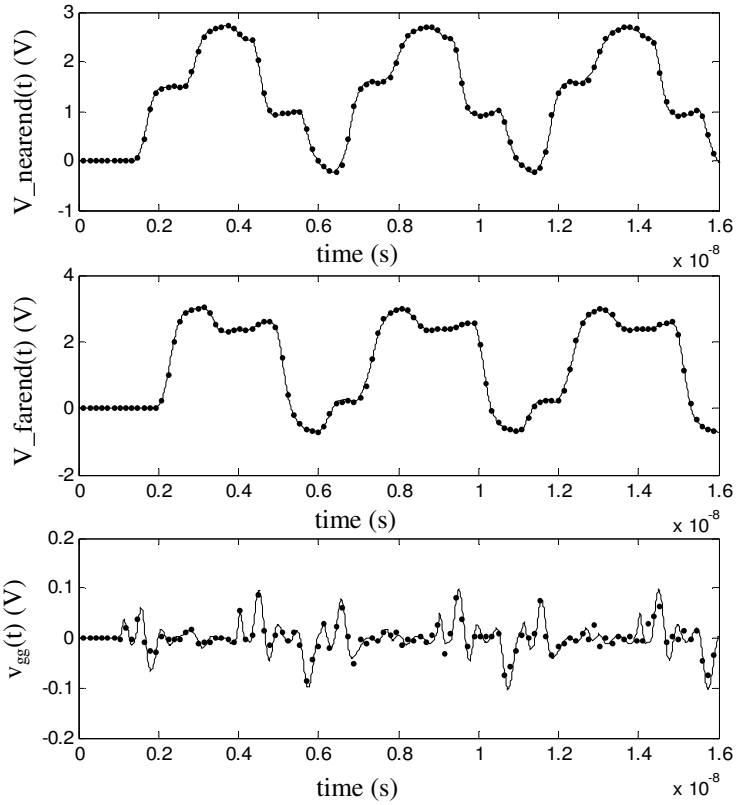


Figure 3.21 Voltage waveforms at near-end of the transmission line, far-end of the transmission line, and local ground from IBM DDR2 driver circuit (straight line) and RNN model (dotted line).

It can be seen from Figure 3.21 that RNN models can be extended to multiple ports and can also capture sensitive effect like SSN accurately. IBM transistor-level driver circuit took 1350 seconds for simulation and 172414 Kbytes of memory and RNN model took 60 seconds for simulation and 1513 Kbytes of memory. It can be seen that RNN models were faster consumed less memory and accurately model the ground noise and driver output voltage. All the simulations were run on DELL 2 GHz personal computer.

Case 5: A test case was generated where six IBM ('DDR2') drivers were connected to six 50-ohm ideal transmission lines with 0.5 ns time delay. All the transmission lines were terminated with a 2 pF capacitance. All the driver circuits had a rise time of 0.25 ns. The power supply of 2.5 V is supplied to the driver through a 0.1 nH inductance. The local ground of the driver circuit was connected to the ideal ground through a 0.1 nH inductance. The voltage waveforms at the near-end of the transmission line, the power supply node, and ground node were plotted when all the drivers were switching simultaneously. It can be seen from Figure 3.22 that the voltage waveforms from the RNN model (dotted line) match well with the actual transistor-level IBM DDR2 voltage waveforms. The transistor-level driver circuit took 886 seconds for simulation and 173096 Kbytes of memory. RNN model took 49 seconds and 7485 Kbytes of memory for simulation. All the simulations are run on DELL 2-GHz personal computer. It can be seen that RNN models are accurate, faster and consume less memory than transistor-level driver circuits.

Table 3.1 summarizes the computational speed-up and memory reduction driver macromodels have compared to transistor-level driver circuits.

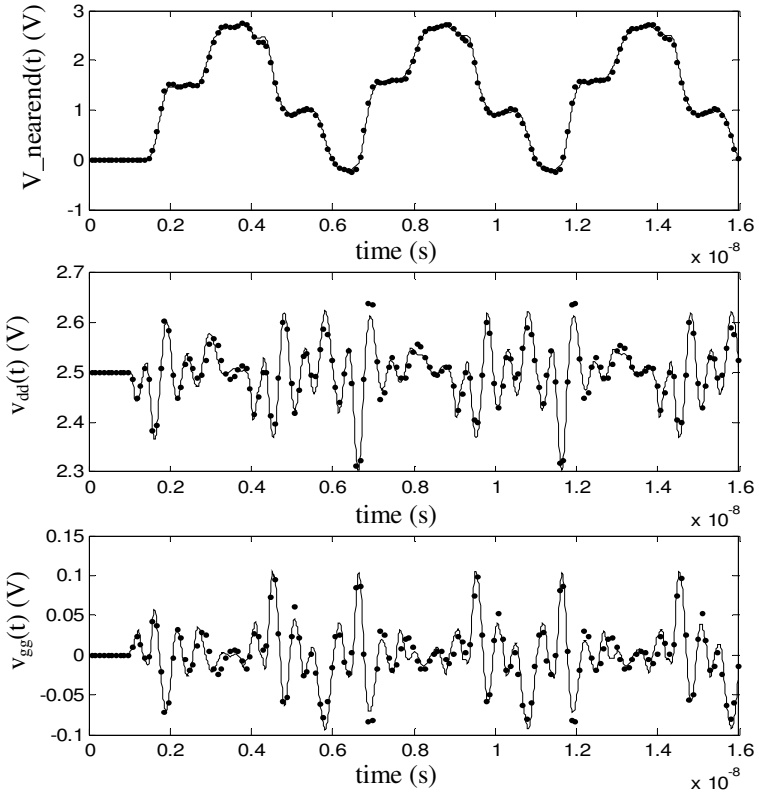


Figure 3.22 Voltage waveforms at near-end of the transmission line, power supply node, and local ground from IBM DDR2 driver circuit (straight line) and RNN model (dotted line).

Table 3.1 Memory reduction and computational speed-up for driver macromodels.

	Memory Reduction	Computational Speed-up
Test Case1	75X	30X
Test Case2	80X	25X
Test Case3	55X	7X
Test Case4	113X	23X
Test Case5	25X	18X

3.4 Summary

In this chapter, driver modeling approach has been extended to multiple ports to include the effect of both power supply and ground noise. It has been seen that both SFWFTD models and RNN models can be extended to multiple ports to include the effect of non-ideal power supply port. RNN macromodels have also been extended to include the effect of non-ideal ground port. It is known that modeling sensitive effects like SSN accurately is a big challenge. Existing driver models like IBIS cannot model SSN. The effect of non-ideal power supply and ground ports on driver output voltage and vice-versa has been captured accurately.

Various test cases have been generated to model the accuracy of SFWFTD and RNN macromodels. It was seen from results that both SFWFTD and RNN models were accurate in modeling sensitive effects like SSN. Both SFWFTD and RNN models consume less CPU simulation time and less CPU memory compared to transistor-level driver circuits.

CHAPTER IV

MACROMODELING OF DIFFERENTIAL DRIVER CIRCUITS WITH PRE-EMPHASIS

Modern high speed digital interfaces have turned to low voltage differential signaling (LVDS) because of its numerous advantages over single-ended signaling. Differential signals have lower voltage swings than single-ended signals due to self-referencing which in turn leads to faster circuits with low power consumption. Differential signals have reduced electromagnetic interference (EMI) effects as the opposite currents carried on the two traces cancel the electric and magnetic fields [D1]. They are also less sensitive to crosstalk coupling. With advanced differential drivers featuring advanced high-speed techniques such as slew rate control and pre-compensation to drive differential signals, accurate macromodeling of digital differential drivers is a huge challenge. In order to simulate the operation of LVDS links for the assessment of signal integrity and electromagnetic compatibility (EMC) problems, suitable macromodels of differential drivers are needed. Therefore, differential driver macromodels should be accurate, computationally fast, and consume less CPU memory providing the designer increased coverage and faster simulation. The macromodels must also be efficient and accurate enough to predict sensitive effects like reflections and crosstalk.

It is always advantageous to have a black-box modeling approach that is independent of the knowledge of the internal logic of the differential driver [D2]. Figure 4.1 shows the

black-box representation of a differential driver circuit. Differential driver macromodeling is different from single-ended driver modeling as the output current at port P is dependent on output voltages at ports P and N. Similarly, current at port N is dependent on voltages at ports P and N.

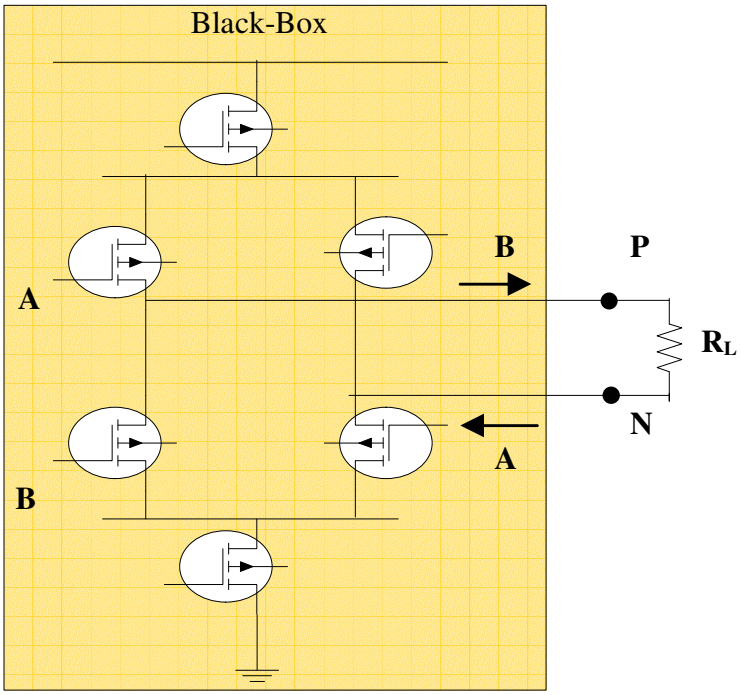


Figure 4.1 Black-box model of a differential driver.

In this chapter, input/output buffer information specification (IBIS) differential driver models have been discussed along with their limitations in section 4.1. Recurrent neural network (RNN) modeling methodology has been proposed to accurately model differential driver circuits in section 4.2. Pre-emphasis differential driver circuits are efficient in driving signals through extremely lossy transmission lines. Section 4.3 discusses modeling of differential drivers with two-bit and three-bit pre-emphasis effect

in detail. Spice netlist generation for RNN modeling approach has been discussed in section 4.4. Test results validating the accuracy and the computational speed-up of the macromodeling methodology have been shown in section 4.5. Section 4.6 summarizes the chapter.

4.1 IBIS Differential Driver Model

IBIS is the current industry standard for driver models [D3]. IBIS models are behavioral models that protect proprietary information and run faster than transistor-level driver models. Figure 1.5 represents an IBIS differential output buffer model. It can be seen that IBIS models use a combination of push-up and pull-down circuits that have driver circuit's static I-V characteristic information. IBIS models also have rising and falling *v-t* waveforms that help transitioning from one logic state to another [D4]. A more detailed description on IBIS single-ended buffer model functioning is given in chapter I. It has been shown in chapter I that IBIS models fail to capture the dynamic characteristics of driver circuits accurately as they are primarily based on the static characteristics.

IBIS format for differential buffer model is exactly similar to a single-ended buffer model. To create a differential IBIS model, IBIS data is extracted as if it was a single-ended device. In IBIS differential driver model, for each differential output pin a separate single-ended buffer model is used. But in practice these buffer models are not independent. This has led to discussion in the IBIS community as to what is the best method to create differential IBIS models [D5]-[D6]. As a result, there are several methods that have been proposed and are used by different vendors and industry. IBIS differential driver models have limitations in terms of accuracy as they are modeled as

single-ended devices. In this paper, a modeling technique based on RNN has been proposed to accurately model differential driver circuits. This modeling methodology models the differential driver by deriving nonlinear port relationships between its output currents and output voltages. The proposed modeling technique is a black-box modeling approach with little dependence on the external load connected to the driver circuit.

4.2 Differential Driver Modeling

The nonlinear relation between the differential driver output voltages and currents can be captured using a nonlinear RNN relation as:

$$i_{op}(t) = w_{1p}(t) f_{1p}(v_{op}, v_{on}, i_{op}) + w_{2p}(t) f_{2p}(v_{op}, v_{on}, i_{op}) \tag{4.1}$$

$$i_{on}(t) = w_{1n}(t) f_{1n}(v_{op}, v_{on}, i_{on}) + w_{2n}(t) f_{2n}(v_{op}, v_{on}, i_{on}) \tag{4.2}$$

where i_{op} and i_{on} are the currents at the output ports P and N of the differential driver, as shown in Figure 4.1. The output voltages at ports P and N are represented by v_{op} and v_{on}, respectively. Sub-models f_{1p} and f_{2p} capture the nonlinear relation between driver port P current and driver ports P and N voltages when the differential driver input is set HIGH and LOW, respectively. Similarly, f_{1n} and f_{2n} capture the nonlinearity of port N for input HIGH and LOW, respectively. Weighting functions w_{1p} and w_{2p} help in transitioning sub-models f_{1p} and f_{2p} from one state to another. Similarly, weighting functions w_{1n} and w_{2n} help in transitioning sub-models f_{1n} and f_{2n} from one state to another. Equations (4.1) and (4.2) approximate the external device behavior including the information on state transitions without assumptions on the device internal structure. One of the important challenges is to accurately model sub-models f_{1p}, f_{2p}, f_{1n}, and f_{2n}. Calculation of weighting functions w_{1p}, w_{2p}, w_{1n}, and w_{2n} is dependent on the accuracy of sub-model estimation.

In [D7] and [D8], differential driver output currents, i_{op} and i_{on}, were represented as a sum of a static mapping and dynamic function. Differential driver output port P current was represented as:

$$f_{1p}(v_1, v_2) = \hat{i}_{Hop}(v_1, v_2) + \bar{i}_{Hop}(v_1, v_2, t) \tag{4.3}$$

$$f_{2p}(v_1, v_2) = \hat{i}_{Lop}(v_1, v_2) + \bar{i}_{Lop}(v_1, v_2, t) \tag{4.4}$$

where $\hat{i}_{Hop}$ and $\hat{i}_{Lop}$ capture the static characteristics of driver current i_{op} when the driver input is fixed at HIGH logic state and LOW logic state, respectively. The dynamic characteristics of the driver output port P are captured using $\bar{i}_{Hop}$ and $\bar{i}_{Lop}$ when the driver input is held HIGH and LOW, respectively. Static characteristics functions, $\hat{i}_{Hop}$ and $\hat{i}_{Lop}$, are represented using artificial neural networks as shown below:

$$\hat{i}_{H/Lop}(v_1, v_2) = \sum\nolimits_n (a_n \tanh(b_o + b_1 v_{op} + b_2 v_{on}) \tag{4.5}$$

where a_n, b_o, b_1, and b_2 are scalar constants for ANN hyperbolic tangential function.

The dynamic functions, $\bar{i}_{Hop}$ and $\bar{i}_{Lop}$, are represented as:

$$\bar{i}_{H/Lop}(k) = \alpha_{01}\bar{i}_{H/Lop}(k-1) + \ldots + \alpha_{10} v_{op}(k-1) + \ldots + \alpha_{20} v_{on}(k-1) + \ldots \tag{4.6}$$

where α_{01}, α_{10}, and α_{20} are constants [D8].

The model representation defined by equations (4.3) and (4.4) is reminiscent of spline function with finite time difference (SFWFTD) models for single-ended driver circuits that capture both the static and the dynamic characteristics. This modeling approach is efficient for driver circuits with moderate nonlinearity. When the transistor-level driver circuits are highly nonlinear, the above modeling approach cannot accurately capture the nonlinearity present in the driver circuit and the estimation of dynamic characteristic

parameters becomes difficult. This has been demonstrated for single-ended driver circuits in [C5]. One solution for modeling these highly nonlinear driver circuits is by use of RNN networks, as RNN functions are powerful nonlinear interpolation functions which can model the nonlinearity of these complex drivers.

To model the nonlinearity of the differential driver, two piece-wise linear (PWL) voltage sources are connected at the end of the driver output ports for each input logic state. The two output voltage sources excite the nonlinearity of the differential driver in the required voltage range. Figure 4.2 shows PWL voltage sources connected at the end of an IBM differential driver ('bsdb25') when the differential driver input is set HIGH.

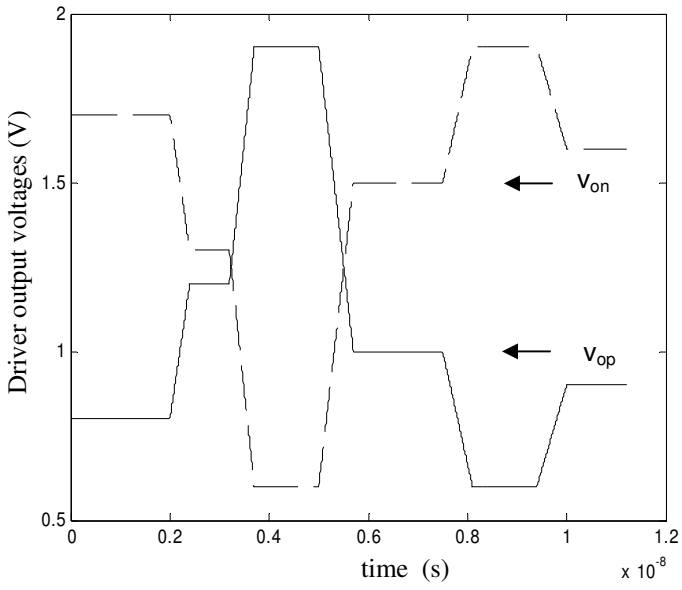


Figure 4.2 PWL voltage sources at the driver outputs.

The resultant current waveforms from these voltage sources are shown in Figure 4.3. These differential driver current waveforms should be accurately modeled to model the driver circuit accurately. The same procedure is repeated when the differential driver

input is held LOW.

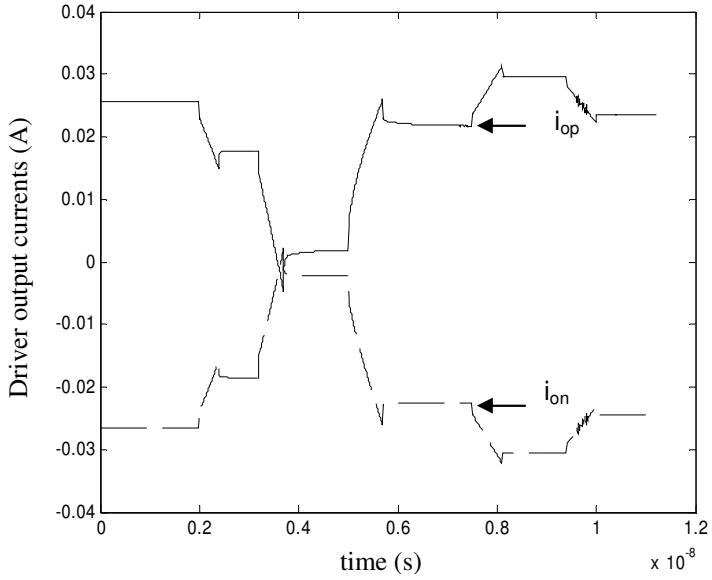


Figure 4.3 Resultant output current waveforms from PWL voltage sources.

The nonlinear relation between the voltages and currents can be captured using a nonlinear RNN relation as shown below:

$$f_{qp/n} = \sum_{k=1}^{M} b_{kj}\, g\Big(\sum_{j=1}^{N} a_{ji} x_i + a_{oj}\Big) + b_{ok} \; ; \; q = 1,2 \tag{4.7}$$

where,

$$g(x) = (e^{x} + e^{-x})/(e^{x} - e^{-x}) \tag{4.8}$$

$$x^{T}(k) = \begin{Bmatrix} i_{op/n}(k-1), i_{op/n}(k-2), ..., i_{op/n}(k-r), \\ v_{op}(k), v_{op}(k-1), ..., v_{op}(k-r), \\ v_{on}(k), v_{on}(k-1), ..., v_{on}(k-r) \end{Bmatrix} \tag{4.9}$$

Sub-models f_{1p}, f_{2p}, f_{1n}, and f_{2n} are expressed using a summation of hyperbolic tangential functions as shown in equation (4.7).

In equation (4.7), b and a are weights associated with the neural network, N

represents number of hidden neurons, M represents number of outputs, and x is a vector that takes into account all the previous and present samples of differential driver output voltages and past samples of output current. The number of samples of output voltages and output current that are included in the RNN model depends on the complexity of the differential driver. RNN uses modified back propagation through time (BPTT) algorithm to estimate the weights [B19]. A detailed description of RNN modeling method is given in chapter II for modeling highly nonlinear single-ended driver circuits.

Figure 4.4 shows the plot for sub-models f_{1p} and f_{1n} and differential driver output currents when the input is HIGH. It can be seen that RNN sub-models accurately model the nonlinearity of the differential driver output currents.

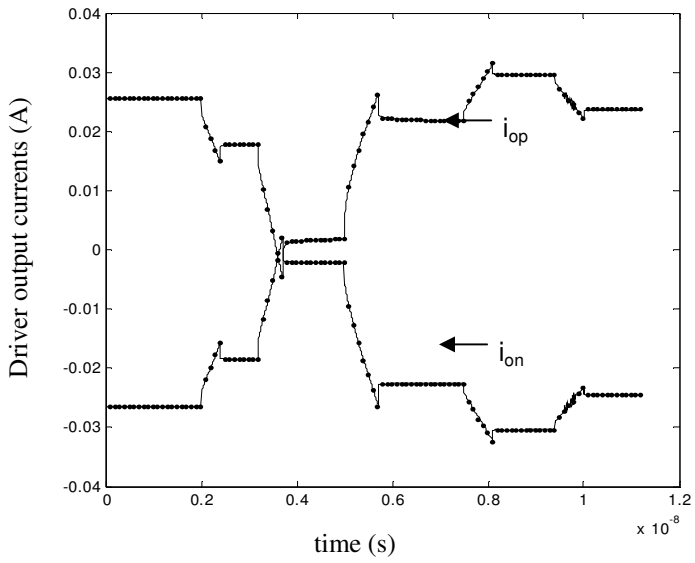


Figure 4.4 Differential driver output current waveforms (straight lines) from PWL voltage sources and from sub-models f_{1p} and f_{1n} (dotted lines).

Weighting functions w_{1p} and w_{2p} can be estimated from linear inversion of equation (4.1) for two different loads. One of the loads is resistive impedance and the other is a combination of resistor and a dc voltage source. The weighting functions for output port

P can be estimated as shown below:

$$\begin{bmatrix} w_{1p} \\ w_{2p} \end{bmatrix} = \begin{bmatrix} f_{1pa} & f_{2pa} \\ f_{1pb} & f_{2pb} \end{bmatrix}^{-1} \begin{bmatrix} i_{opa} \\ i_{opb} \end{bmatrix} \tag{4.10}$$

where i_{opa} and i_{opb} represent the differential driver output currents from two different load terminations. Usually a 100-ohm differential load can be selected as one of the loads. A combination of 100-ohm differential load and a DC voltage source can be selected as the second load. The goal is to excite sub-models f_{1p}, f_{2p}, f_{1n}, and f_{2n} in the range of differential driver output voltage variation. Figure 4.5 shows weighting functions w_{1p} and w_{2p} that were calculated using equation (4.10)

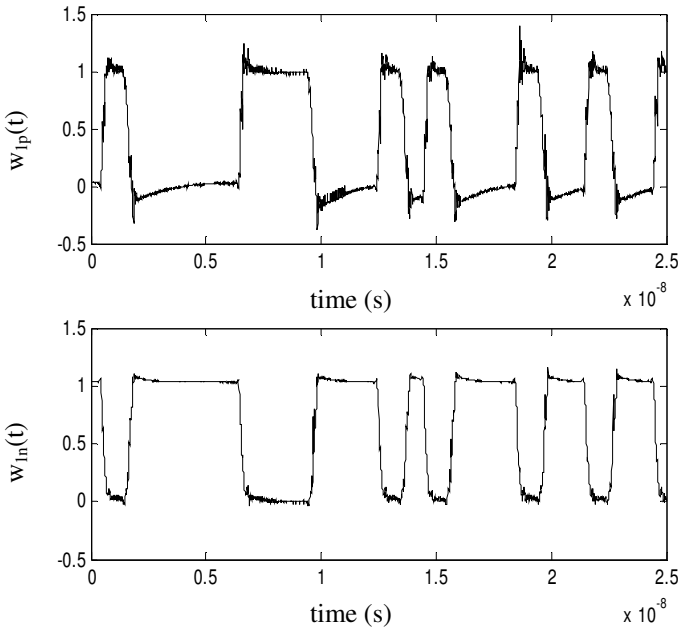


Figure 4.5 Weighting functions w_{1p} and w_{2p} for IBM driver.

It can be seen that these weighting functions enable sub-models f_{1p} and f_{2p} in transitioning from one state to another. Similarly, weighting functions w_{1n} and w_{2n} enable sub-models f_{1n} and f_{2n} in transitioning from one state to another.

4.3 Pre-emphasis Differential Driver Modeling

A source for electrical characteristics of interconnections for high-performance systems can be found in [D9]. The most dominant limit of copper interconnect is loss [D10] and there are multiple methods to address high loss in signaling channels [D11]. Pre-emphasis (Pre-compensation) drivers are important in maintaining signal integrity of lossy channels. These drivers boost the magnitude of high frequency spectral components of signals, thus acting as a high pass filter ensuring that the signal reaches the receiver without affecting the logic even after the channel loss (which acts as a low pass filter). Pre-emphasis drivers are useful in reducing Inter-Symbol Interference (ISI). A detailed explanation of operation of pre-emphasis scheme is given in chapter II, section 5. To accurately model the nonlinearity of pre-emphasis differential drivers care should be taken in estimating the range of the voltage variation of sub-models (f_{1p}, f_{2p}, f_{1n}, and f_{2n}). For a two-bit pre-emphasis, the four voltage states of a differential driver can be represented as 00 (LOW), 01 (Strong LOW), 10 (Strong HIGH), and 11 (HIGH). Since strong LOW and strong HIGH result in large voltage swings, sub-models f_{1p} and f_{1n} should capture strong HIGH and sub-models f_{2p} and f_{2n} should capture strong LOW accurately. The weighting functions are obtained from equation (4.10). Figure 4.6 shows weighting functions w_{1p} and w_{2p} for a two-bit pre-emphasis scheme. It can be seen from Figure 4.6 that the effect of the pre-emphasis is included in the weighting functions.

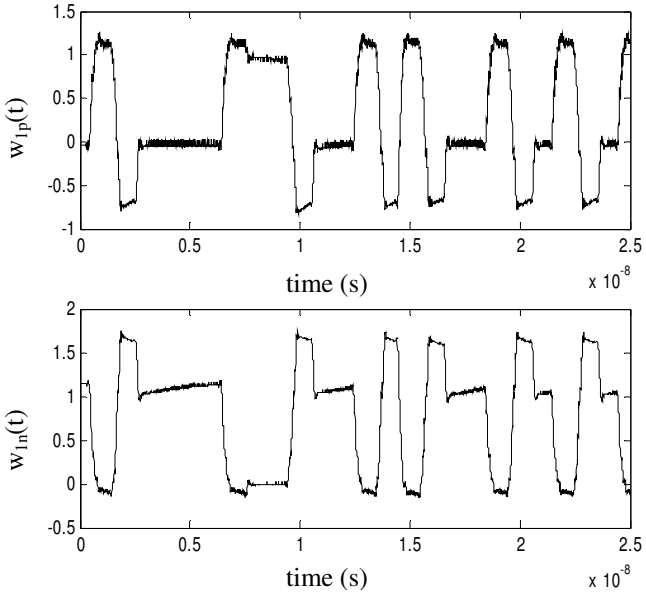


Figure 4.6 Weighting functions w_{1p} and w_{2p} for a two-bit pre-emphasis.

The principle used to model two-bit pre-emphasis can also be extended to model a three-bit pre-emphasis differential driver. A three-bit pre-emphasis driver will have 2^3 combinations of input voltage variations. To estimate sub-models f_{1p} and f_{2p} accurately, the right voltage combination needs to be picked. The eight voltage variation combinations can be expressed in binary form as 000, 001, 010, 011, 100, 101, 110, and 111. In the binary combination, least significant bit (LSB) represents the third bit in the pre-compensation. The second LSB represents the second bit in the pre-compensation. In 100, the first bit voltage is HIGH and the other bits are LOW, and this results in a strong HIGH voltage and similarly, 011 results in a strong LOW voltage. It can be seen that for modeling f_{1p}, the voltage combination that results in a full three-bit pre-emphasis is 100 and to model f_{2p} that results in three-bit pre-compensation is 011. Similarly, sub-models f_{1n} and f_{2n} are also estimated for 100 and 011 voltage combinations. Care should be taken in properly estimating the voltage variations in modeling sub-models f_{1p}, f_{2p}, f_{1n}, *and* f_{2n}.

Figure 4.7 shows the weighting functions w_{1p} and w_{1n} that are calculated for a three-bit pre-compensation scheme in IBM driver using equation (4.10). It can be seen that the three-bit pre-compensation has been taken into account in the weights. Since the worst case combination of voltage variations is used in estimating sub-models f_{1p},f_{2p}, f_{1n}, and f_{2n} for output ports P and N, the macromodel accurately models the three-bit pre-compensation differential driver.

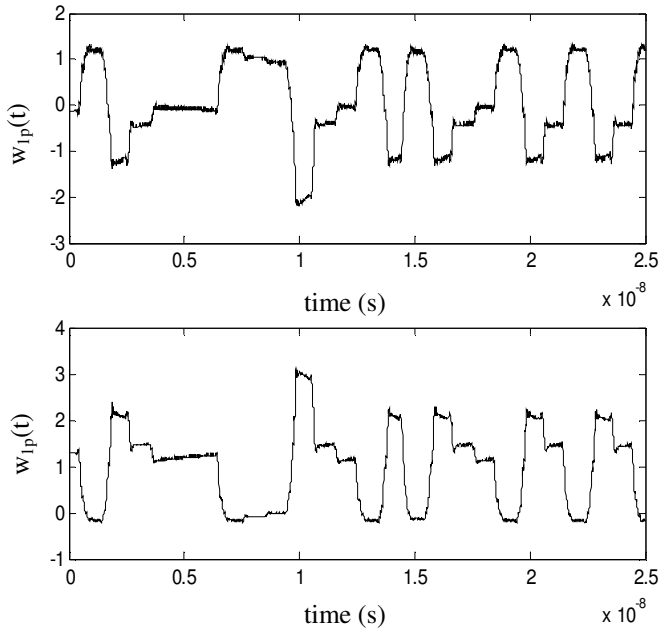


Figure 4.7 Weighting functions w_{1p} and w_{2p} for a three-bit pre-emphasis.

4.4 Spice Netlist

Spice circuit generation of RNN modeling methodology for differential driver circuit is similar to spice circuit generation for single ended driver circuits. Spice circuit can be generated using PWL voltage sources, voltage-dependent current sources, and voltage-dependent voltage sources. The weighting functions w_{1p}, w_{2p}, w_{1n}, and w_{2n} are represented using PWL voltage sources. A voltage-dependent current source has been used to

represent sub-models f_{1p}, f_{2p}, f_{1n}, and f_{2n}. Previous time instances of driver output voltages and currents are calculated using state equations. A more detailed description of representing previous time instances of driver output voltage and current is provided in Chapter II. A schematic of spice netlist for a differential driver circuit is given in appendix A

4.5 Test Results

Three test cases were designed to validate the accuracy of the modeling methodology on an IBM transistor-level differential driver ('bsdb25'). The IBM driver was operated at 1 GHz with a power supply voltage of 1.8V. The driver can be operated with a three-bit pre-emphasis. The IBM driver was designed for a 100-ohm differential load. Figure 4.8 shows the test set-up for all the three test cases. The first test case involved modeling IBM ('bsdb25') differential driver without pre-emphasis. The second test case involved modeling the IBM driver with two-bit pre-compensation. The third test case involved modeling the transistor-level IBM driver using three-bit pre-emphasis. The simulations were carried out on a Pentuim-4 1.8 GHz windows PC with 512 MB RAM.

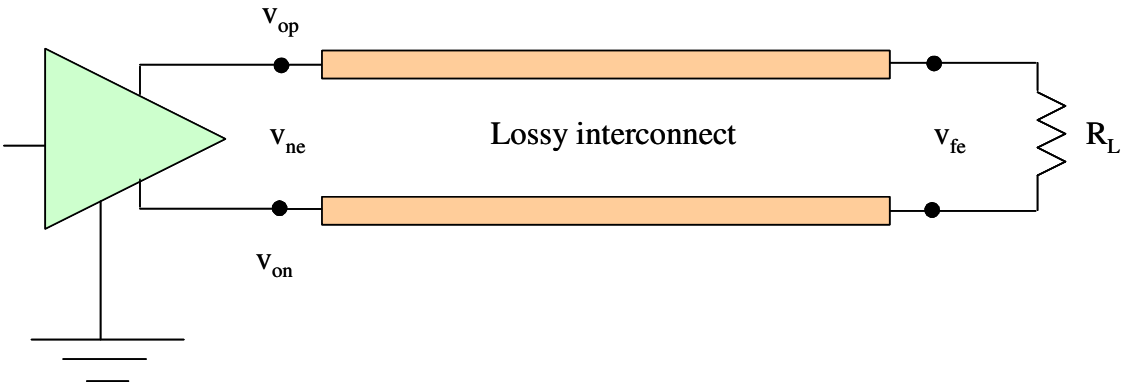


Figure 4.8 Test set-up for IBM differential driver.

Case 1:

In this test case, the IBM driver ('bsdb25') was connected to a differential 90-ohm lossy transmission line that was 12 inches long. The transmission line was in turn terminated with a 130-ohm resistor as shown in Figure 4.8. The characteristic impedance mismatch of the transmission line and load termination mismatch result in reflections at the far-end of the transmission line that would test the macromodels accurately for worst case reflections. The voltage waveforms at the near-end of the transmission line for both the output ports were measured from the RNN macromodel and actual transistor-level driver model. It can be seen from Figure 4.9 that the near-end voltage waveforms for both the transistor-level driver model and the RNN macromodel match accurately.

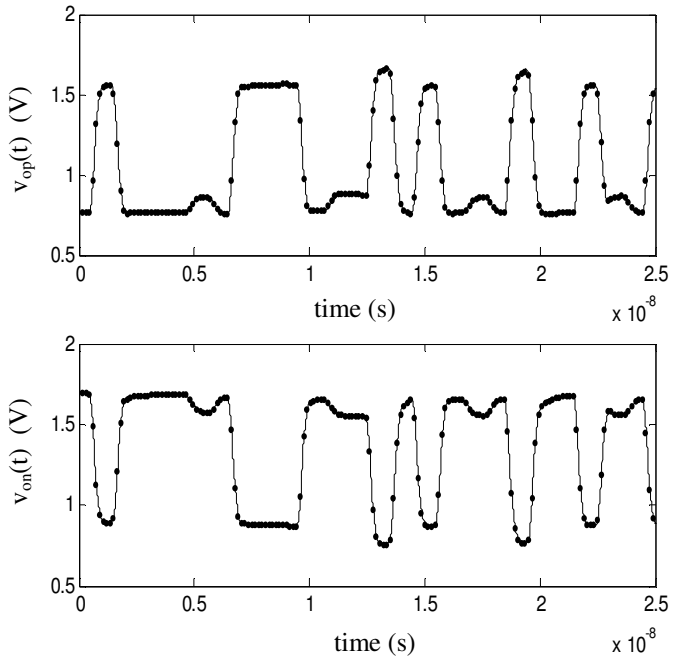


Figure 4.9 Voltage waveforms at the near-end of the transmission line for IBM transistor-level driver (straight line) and RNN macromodel (dotted line).

The IBM transistor-level differential driver model took 282 seconds for simulation where as, the RNN macromodel took 19 seconds, with a resulting speed-up of 15 X.

Case 2:

A test case was generated where the IBM driver ('bsdb25') was connected to a 100-ohm differential lossy transmission line that was 12 inches long. The transmission line was in turn terminated with a 70-ohm resistor as shown in Figure 4.8. The voltage waveforms at the near-end of the transmission line of both the output ports were measured from the RNN macromodel and the actual transistor-level driver model. It can be seen from Figure 4.10 that the near-end voltage waveforms for both the transistor-level driver model and the RNN macromodel match accurately. The IBM transistor-level differential driver model took 221 seconds for simulation where as, the RNN macromodel took 19 seconds, with a resulting speed-up of 12X.

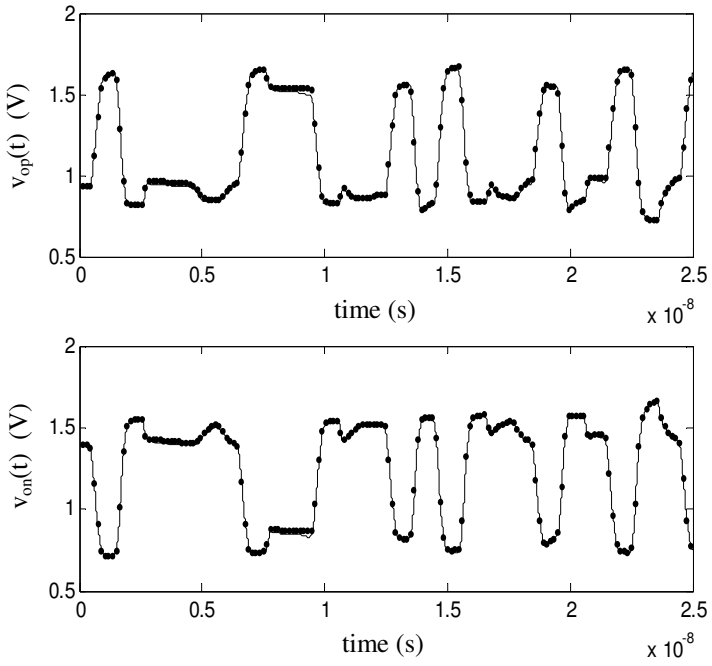


Figure 4.10 Voltage waveforms at the near-end of the transmission line for IBM transistor-level driver (straight line) and RNN macromodel (dotted line).

Case 3:

In this case, the IBM differential driver was modeled to capture the three-bit pre-emphasis effect. The IBM driver was connected to a 100-ohm lossy differential transmission line and the transmission line was terminated using a 130-ohm resistor as shown in Figure 4.8. The termination load was mismatched to test the accuracy of the macromodel for voltage reflections. Voltage waveforms at near-end and far-end of both the output ports were measured using RNN macromodel and actual transistor-level driver model. It can be seen from Figure 4.11 that the RNN macromodel waveforms match well with both the near-end and the far-end voltage waveforms of the transistor-level driver model.

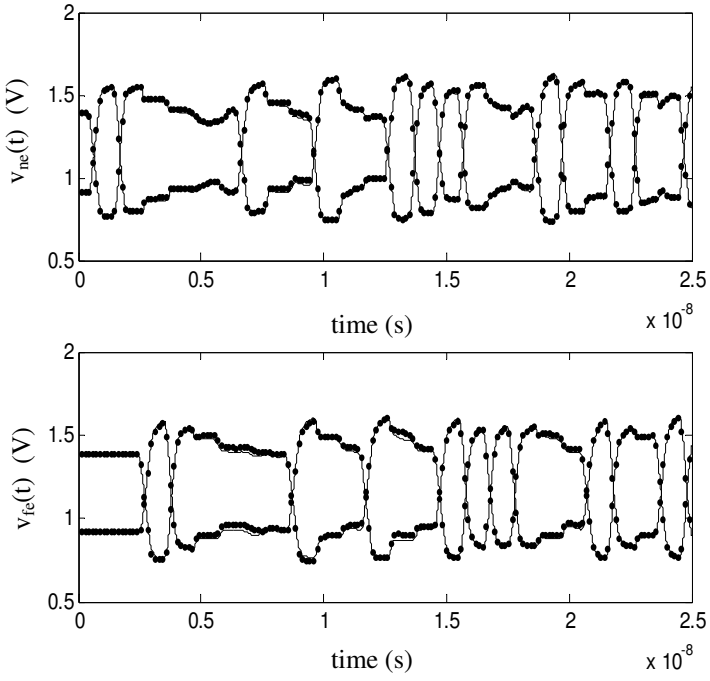


Figure 4.11 Voltage waveforms at the near-end and far-end of the transmission line for IBM transistor-level driver (straight line) and RNN macromodel (dotted line).

The simulation time consumed by both the driver models was calculated. The IBM transistor-level driver model took approximately 205 seconds and the RNN macromodel

took about 20 seconds. The computational speed advantage of RNN macromodel is independent of the complexity of the driver circuit. The speed advantage increases with the increase in simulation time and complexity of the circuit.

Table 4.1 shows a comparison between IBM transistor-level driver circuit and RNN macromodel for all three test cases in terms of CPU time and CPU memory consumed. It can be seen that the RNN macromodel consumes 10 times less memory compared to transistor-level driver model.

Table 4.1 Comparison between IBM driver and RNN macromodel.

Case	CPU Time (s)		CPU Memory (KB)		Mean Square Error
	IBM	RNN	IBM	RNN	
1	282	19	13683	1363	5.1e-4
2	221	19	13683	1373	4.5e-4
3	205	20	13640	3378	5.4e-4

4.6 Summary

In this chapter, a modeling methodology based on RNN has been used to model differential driver circuits with and without pre-emphasis. It has been shown in this chapter that IBIS, which is the present industry standard for black-box modeling of driver circuits models differential driver circuits as two independent single-ended driver models. The interaction between the two output ports is not captured in IBIS. It has also been shown in chapter I that IBIS models cannot accurately capture the behavior of single-ended driver circuits.

RNN functions are powerful interpolation functions that have been used to model nonlinear systems with feedback accurately. The accuracy of RNN functions to model highly nonlinear single-ended driver circuits has been shown in chapter II. In this chapter,

RNN modeling approach has been extended to model differential driver circuits. RNN modeling methodology has been tested on various test cases with and without pre-emphasis and results showed good accuracy. RNN macromodels are faster than transistor-level differential circuits by one or two orders of magnitude and consume less memory thus providing the designer faster and accurate results which reduces time-to-market and allows additional time for increased coverage thus improving design quality.

CHAPTER V

RECEIVER MACROMODEING

Receiver circuits are important in analyzing signal integrity and power integrity issues in today's high speed digital systems. Modeling receiver circuits is more complicated than modeling driver circuits as the input to the receiver circuit is not digital but analog in nature. Receiver macromodels like driver macromodels should protect the intellectual property of the actual transistor-level receiver circuit. A black-box macromodel of a receiver circuit can help in efficient time domain analysis of today's high-speed digital systems. Figure 5.1 shows a scenario where both the driver and receiver circuits have been replaced with their black-box equivalent macromodels. The focus of this chapter is on accurate modeling of receiver circuits.

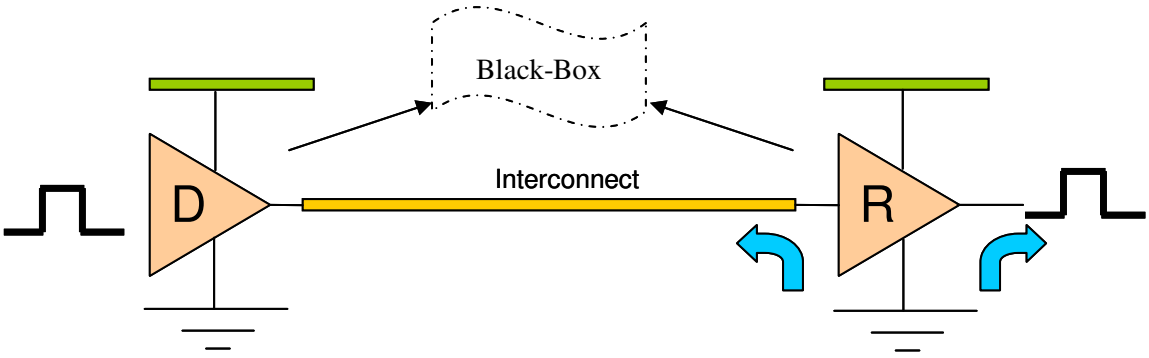


Figure 5.1 Black-box models of driver and receiver circuits.

A few modeling techniques to model receiver circuits have been proposed in the past and IBIS receiver models are popular amongst them. IBIS receiver models use power and

ground (‘GND’) clamping diodes with high and low voltage threshold values [E1]. IBIS receiver models are based on the static characteristics of the receiver and the delay of the signal through the receiver is not taken into account. Also in the past, receiver input characteristics have been modeled using a shunt capacitor and a shunt nonlinear resistor defined by the *i-v* port static characteristics of the receiver. Receiver input characteristics have also been modeled using a combination of linear and nonlinear basis functions [E3]. These modeling approaches result in complex macromodels with large number of basis functions. None of these macromodeling techniques have addressed the issue of modeling the output characteristics of the receiver circuit, which plays an important role in timing analysis of high-speed digital systems. In order to model receiver circuits efficiently, both their input as well as output characteristics have to be accurately captured.

In this chapter, a detailed description of existing modeling approaches and their limitations is discussed in section 5.1. In section 5.2, receiver modeling has been divided into two parts, modeling of receiver input characteristics and modeling of receiver output characteristics. Receiver input characteristics have been modeled using spline function with finite time difference (SFWFTD) and recurrent neural network (RNN) functions. Receiver output characteristics have been modeled using voltage transfer characteristics (VTC) with finite time delay element. Receiver modeling approach has also been extended to multiple ports to include the effect of power supply noise on receiver input and output voltages in section 5.3. In section 5.4, spice netlist generation for the proposed method is discussed. A few test cases have been generated to test the accuracy of the proposed modeling approach .The accuracy of extension of receiver modeling approach

to multiple ports has also been tested on few test cases in section 5.5. Section 5.6 summarizes the chapter.

5.1 Receiver Modeling Approaches – Prior Art

IBIS receiver models are the present industry standard for modeling receiver circuits. In IBIS, a typical receiver circuit contains a ground (GND) clamp and a power clamp, as shown in Figure 5.2. The power and GND clamps represent the electrostatic discharge (ESD) structure. IBIS receiver circuit also has a voltage high logic threshold (v_{ih}) and a low voltage logic threshold (v_{il}) for the input. The IBIS simulator uses these logic threshold values to compute signal integrity issues such as overshoot/undershoot and noise margins [E1].

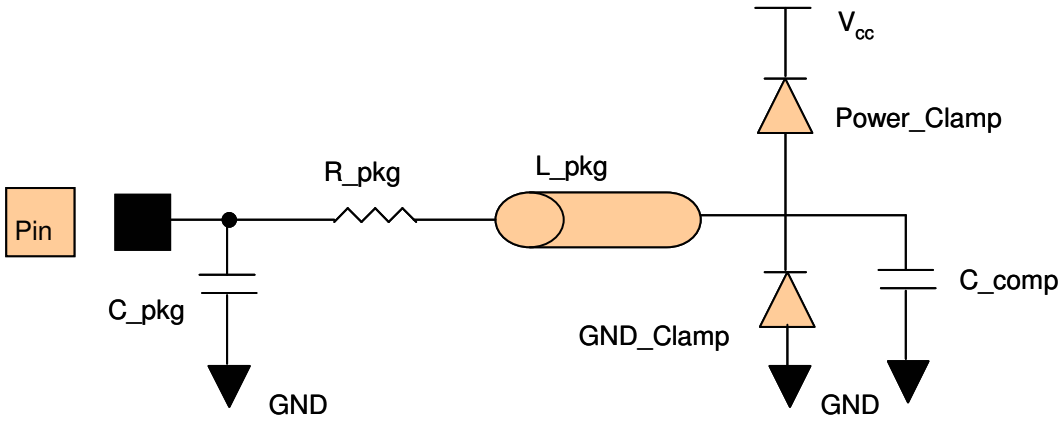


Figure 5.2 IBIS receiver model schematic.

The input model also includes package parasitics and input die capacitance, C_comp. The C_comp parameter is connected to the input, usually with reference to ground when IBIS file is used in the simulator. It is the capacitance seen when looking from the pad back into the buffer. C_comp is a key parameter for receiver inputs. In IBIS receiver

modeling, the power and GND clamp data is generated following the same procedure used for an IBIS driver model. The sweep voltage range will be –Vcc to Vcc for the GND clamp and Vcc to 2Vcc for the power clamp curve. A more detailed description is given in chapter I.

In [E2], the receiver is modeled using a shunt capacitor and a shunt nonlinear resistor defined by the *i-v* port static characteristics of the receiver. The capacitor and nonlinear resistor take into account the static and dynamic behavior of the receiver circuits. This modeling approach only gives a rough approximation of the receiver input current (i_{in}). In [E3], another modeling technique was proposed that involves expressing the receiver input current, i_{in}, as shown below:

$$i_{in}(k) = i_l(k) + i_{nl}(k) \tag{5.1}$$

where i_l refers to the linear behavior model of the receiver input current and i_{nl} refers to the nonlinear behavior model of the receiver input current. The nonlinear model takes into account the effects of the receiver input current behavior in the voltage range, where the effects of protection circuits cannot be neglected. An autoregressive with extra input (ARX) method has been used in [A38], [E3] to estimate i_l, as shown below:

$$i_l(k) = \Theta_l{}^T x_l(k) \tag{5.2}$$

$$x_l(k) = [i_{in}(k-1),...,i_{in}(k-r_l), v_{in}(k),...,v_{in}(k-r_l)]^T \tag{5.3}$$

Equations (5.2) and (5.3) define a linear combination of the components in the regressor vector x_l, where Θ_l is the vector of parameters collecting the unknown coefficients and r_l is the dynamic order of the sub-model.

The nonlinear model of the receiver input current is represented as:

$$i_{nl}(k) = g_u(\Theta_u, x_u(k)) + g_d(\Theta_d, x_d(k)) \tag{5.4}$$

where g_u and g_d are the radial basis function (RBF) models for up and down protection circuits, respectively. Θ_u and Θ_d are the model parameters for g_u and g_d, respectively. Previous time instances of receiver input current and voltage for up and down protection circuits are contained in x_u and x_d, respectively [E3]. This modeling technique accurately models the receiver input current.

5.1.1 Limitations of Receiver Modeling Techniques

IBIS receiver models are primarily based on the static characteristics of the receiver.. The output characteristics of the receiver circuit cannot be modeled accurately by taking only the threshold voltages into account, as the delay information through the receiver is not captured. A combination of clamping diodes with loading capacitor is used to model the input characteristics. This model does not take into account the memory effect of nonlinear receiver circuits.

Representing the loading characteristics of a receiver as a combination of linear ARX model and nonlinear RBF model results in accurate modeling, but the methodology involved is highly complex. In [A38], the number of basis functions needed to model g_u and g_d is of the order 16 to 19, respectively. The number of basis functions required to model the receiver accurately is receiver dependent. Depending on the complexity of the receiver, the number of basis functions might change. But the macromodels become complex as the number of basis functions and the dynamic order of the models increase (the number of previous time instances of receiver input current and voltage), making the macromodels computationally slower and more susceptible to convergence problems.

None of these macromodeling techniques have addressed the issue of accurately modeling the output characteristics of the receiver, which plays an important role in doing timing analysis of high-speed digital systems.

5.2 Receiver Modeling Methodology

In this section, receiver circuits have been modeled by dividing the modeling approach into two parts: modeling of receiver input characteristics and modeling of receiver output characteristics. Receiver input characteristics have been modeled using SFWFTD approach for moderately nonlinear receiver circuits and RNN modeling approach for highly nonlinear receiver circuits. Receiver output characteristics have been modeled using VTC with finite time delay element.

5.2.1 Receiver Input Characteristics Modeling

5.2.1.1 Spline Function with Finite Time Difference (SFWFTD) Model

The input current of any receiver circuit can be expressed using spline function and finite time difference approximation [E4]. For any receiver, the DC input current in terms of input voltage can be expressed as:

$$i_{in}(k) = f_{in,s}(k) = A_m {v_{in}}^m(k) + A_{(m-1)} {v_{in}}^{(m-1)}(k) + ... + A_0 \; ; m \geq 1 \qquad (5.5)$$

where *A*s are constants, v_{in} is the receiver input voltage, i_{in} is receiver input current, and the value of m is dependent on the kind of receiver being modeled. Figure 5.3 shows an IBM receiver AGPV3V2 input current characteristics. It can be seen from Figure 5.3 that the DC receiver input current can be accurately modeled using equation (5.5).

Equation (5.5) does not capture the dynamic response of the receiver. The dynamic characteristics can be included by taking the previous time instances of the receiver input current and input voltage into account. Sub-model $f_{in,s}$ at time instance *'k-1'* can be expressed as:

$$f_{in,s}(k-1) = i_{in}(k-1) = A_m {v_{in}}^m(k-1) + A_{(m-1)} {v_{in}}^{(m-1)}(k-1) + ... + A_0 \qquad (5.6)$$

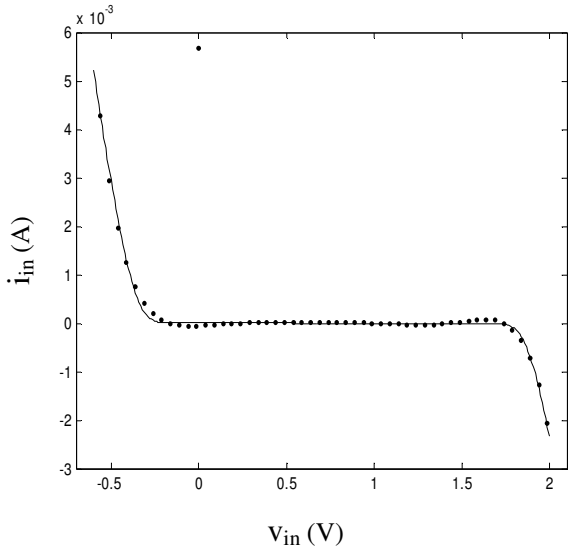


Figure 5.3 Receiver DC input current Vs. input voltage.

Incremental change in the receiver input current Δi_{in} is the difference between the present time instance *(k)* and previous time instance *(k-1)* values of sub-model $f_{in,s}$ as shown:

$$f_{in,s}(k) - f_{in,s}(k-1) = i_{in}(k) - i_{in}(k-1) = \Delta i_{in} \qquad (5.7)$$

$$\text{(Or)} \quad f_{in,s}(t) - f_{in,s}(t-\Delta t) = \Delta i_{in} \qquad (5.8)$$

Once Δi_{in} is calculated, first derivative of receiver input current i'_{in} can be approximated as:

$$\frac{f_{in,s}(t) - f_{in,s}(t-\Delta t)}{\Delta t} = \frac{\Delta i_{in}}{\Delta t} = i'_{in} \tag{5.9}$$

where Δt is the sampling time.

Figure 5.4 shows a PWL voltage source connected at the input of an IBM receiver ('AGPV3V2'). Since the loading effect of the receiver input is similar to a capacitor, the capacitive effect can be included to model the dynamic characteristics.

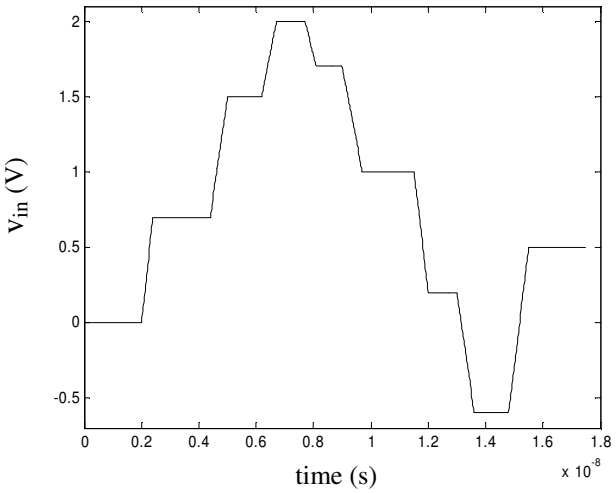


Figure 5.4 A PWL voltage sources connected at the receiver input to calculate the dynamic characteristics.

Therefore, the input current of a receiver circuit can be expressed as:

$$i_{in} = f_{in}(i_{in}, v_{in}) \tag{5.10}$$

$$f_{in}(k) = f_{in,s}(k) + p * i'_{in} + pp * v'_{in} \tag{5.11}$$

where v'_{in} is the first derivative of the receiver input voltage, p and pp are constants whose magnitude can be estimated by calculating the least mean square (LMS) error between transistor-level receiver input current and modeled receiver input current. It can

be seen from Figure 5.5 that the modeled receiver input current (dotted line) and the simulated transistor-level receiver input current (straight line) match accurately.

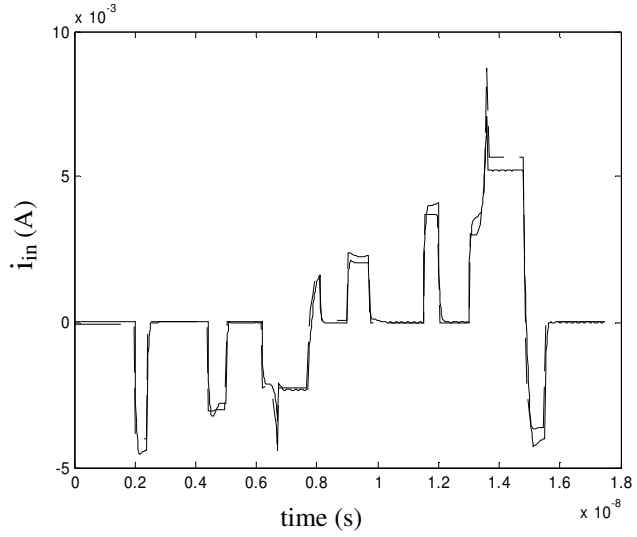


Figure 5.5 IBM ('AGPV3V2') receiver input current from a PWL voltage source connected at receiver input (straight line) and SFWFTD model (dashed line).

SFWFTD modeling technique can accurately model the input characteristics of a moderately nonlinear receiver circuit. For a highly nonlinear receiver circuit, when a PWL voltage source is connected at the input of the receiver circuit, as shown in Figure 5.4, the effect of clamping diodes is reflected in the resulting current signature. SFWFTD models cannot accurately capture such nonlinearity and RNN models are better replacements to model these highly nonlinear receiver circuits.

5.2.1.2 Recurrent Neural Network (RNN) Model

In order to accurately model the input characteristics of a highly nonlinear receiver circuit, the input current, i_{in}, should be accurately modeled. Since the effect of clamping diodes is more predominant in highly nonlinear receiver circuits, a more powerful interpolation technique should be used to model the receiver input current accurately.

Since the input current will be a function of receiver input voltage and input current, the receiver input current can be expressed as:

$$i_{in}(k) = f_{in}(x(k)) \tag{5.12}$$

$$x(k) = \begin{Bmatrix} i_{in}(k-1), i_{in}(k-2), ..., i_{in}(k-p), \\ v_{in}(k), v_{in}(k-1), ..., v_{in}(k-p) \end{Bmatrix} \tag{5.13}$$

where *x(k)* takes into account all the previous time instances of receiver input voltage, v_{in}, and input current, i_{in}. Function f_{in} is a nonlinear RNN function as shown below:

$$f_{in}(x) = \sum_{k=1}^{M} b_{kj}\, g\left(\sum_{j=1}^{N} a_{ji} x_i + a_{oj}\right) + b_{ok} \tag{5.14}$$

$$g(x) = (e^{x} - e^{-x})/(e^{x} + e^{-x}) \tag{5.15}$$

where a and b are weights associated with the recurrent neural network, N represents number of hidden neurons, and M represents number of outputs. The weights are estimated using a modified back propagation through time BPTT algorithm [B19]. A detailed description on RNN is given in chapter II. To estimate f_{in} accurately, a good training data is required that should excite both the linear and the nonlinear regions of the receiver input current. A PWL voltage source with different rise times and different amplitudes connected at the input of the receiver should accurately capture the nonlinear characteristics and the effect of clamping diodes of the receiver circuit. This PWL voltage source should be designed to generate good training data for the neural networks. Figure 5.6 shows the resultant input current for IBM ('DDR2') receiver when a PWL voltage source is connected at receiver input.

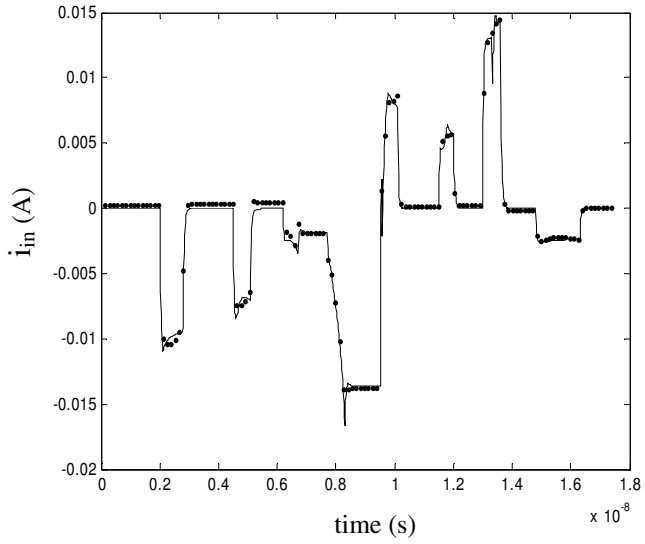


Figure 5.6 IBM DDR2 receiver input current from a PWL voltage source connected at receiver input (straight line) and RNN model (dotted line).

Figure 5.6 shows that the trained RNN model current (dotted line) matches well the transistor-level IBM receiver input current (straight line) [E5]. For the above example, a RNN model with five hidden neurons and with two previous time instances resulted in an accurate model.

5.2.2 Receiver Output Characteristics Modeling

The output voltage of the receiver can be modeled using a combination of static receiver characteristics ($v_{in} - v_{out}$) and a finite delay element to capture the input-output delay of the receiver. Figure 5.7 shows the static characteristic behavior of an IBM ('DDR2') receiver. Any interpolation technique can be used to represent Figure 5.7. In this chapter, the static behavior has been approximated using a artificial neural network (ANN) function as shown below:

$$v_{out}(k) = b_1 g\left(\sum_{j=1}^{N} (a_j v_{in} + a_o)\right) + b_o \tag{5.16}$$

where *as* and *bs* are weights associated with the neural network, *N* represents number of hidden neurons. Back-propagation algorithm was used to train the neural network. In this particular case, a neural network with five hidden neurons was used to capture the static behavior. The output voltage of the receiver circuit from equation (5.16) does not include the delay the signal undergoes when it passes through the receiver. This delay has to be added to the receiver output to compensate for the lost timing information.

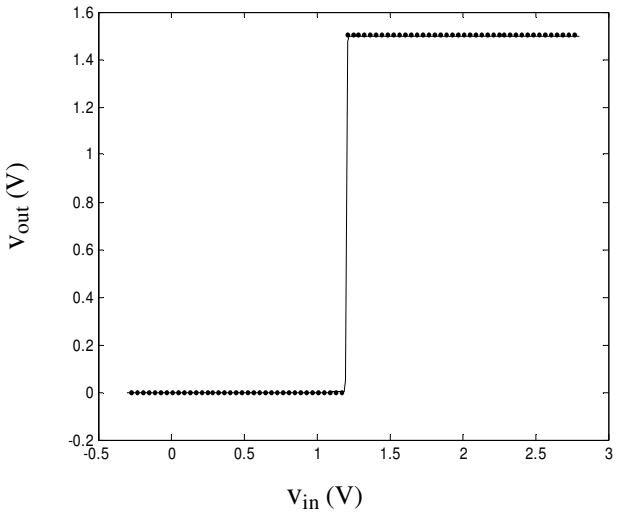


Figure 5.7 Receiver static characteristics for IBM ('DDR2') receiver (straight line) and neural model (dotted line).

The input-output delay of a receiver is dependent on the slew rate of the input voltage and independent of the frequency of excitation for normal range of operations. Figure 5.8 shows the surface plot between input-output delay, frequency of input voltage, and slew rate of the input voltage for IBM ('DDR2') receiver. It can be clearly seen from Figure 5.8 that the delay is a function of input slew rate.

The delay for the particular slew rate can be easily estimated using Figure 5.8 and the relation between input and output voltages can be captured using the static relation. Thus,

the combination of the two would result in accurate modeling of the receiver output voltage.

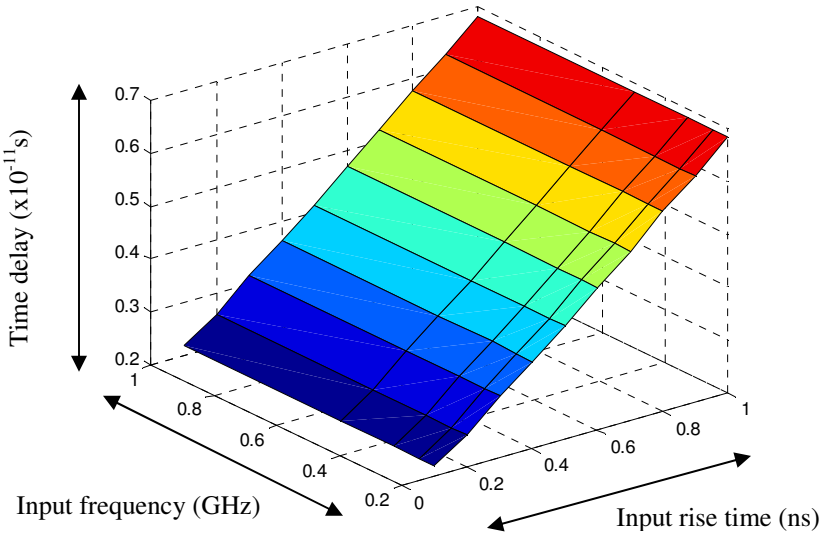


Figure 5.8 Surface plot between slew rate, frequency and time delay for IBM DDR2 receiver.

Figure 5.9 shows the procedure to model receiver output characteristics. Capturing the static characteristics of the receiver circuit and time delay through the receiver circuit accurately is essential for accurate modeling of receiver circuits.

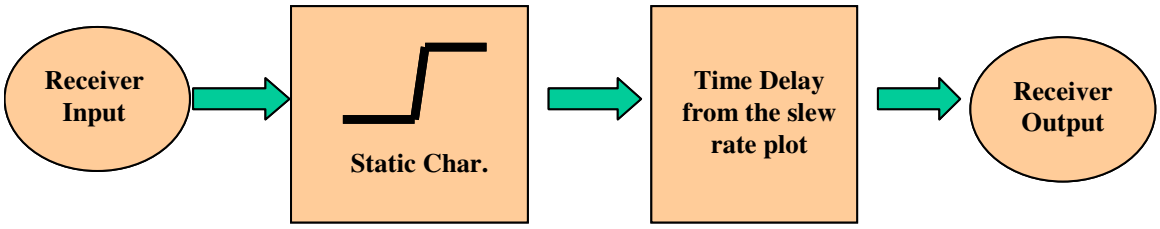


Figure 5.9 Procedure to model the output characteristics of a receiver circuit.

5.3 Extension to Multiple Ports

5.3.1 Receiver Input Characteristics Modeling

The receiver modeling approach discussed in the previous section can be extended to multiple ports. Switching of many receiver circuits simultaneously can result in simultaneous switching noise (SSN). To estimate SSN, the effect of non-ideal power supply port should be taken into account. Therefore, it is important that the modeling methodology be extendable to multiple ports. Figure 5.10 shows a scenario where the power supply is non-ideal. Inductance L_{dd} and resistance R_{dd} represent the power plane parasitics that result in noise. To incorporate the effect of the power supply node (v_{dd}), a new relation should be drawn between receiver power supply current (i_{dd}) and receiver power supply voltage.

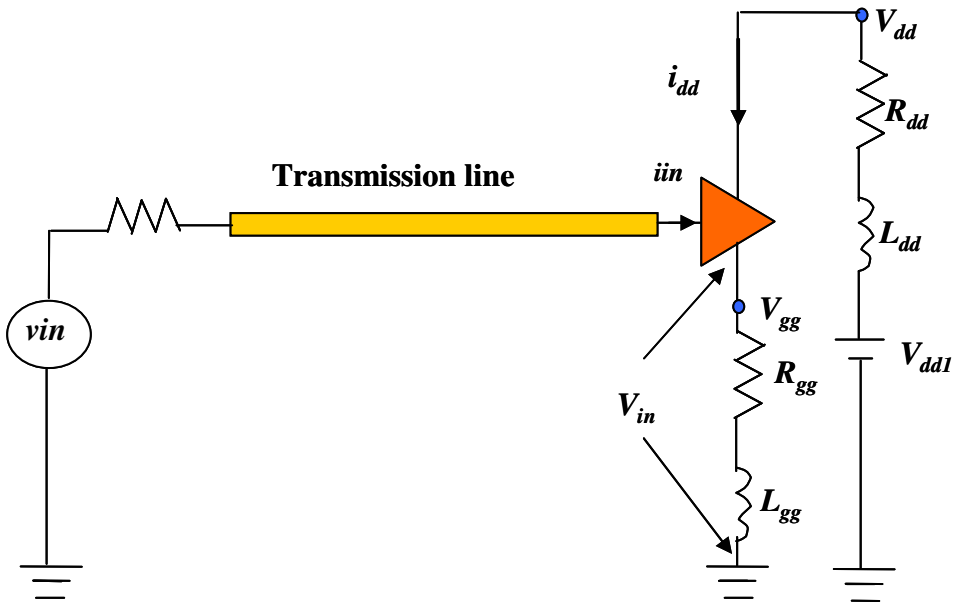


Figure 5.10 Schematic of a receiver circuit with non-ideal power supply node.

However, the receiver power supply current is not only a function of receiver power supply voltage but also a function of receiver input voltage (v_{in}), as shown below:

$$i_{dd}(k) = f_{dd}(x_{dd}(k)) \tag{5.17}$$

$$x_{dd}(k) = \begin{Bmatrix} i_{dd}(k-1), i_{dd}(k-2), ..., i_{dd}(k-p_1), \\ v_{in}(k), v_{in}(k-1), ..., v_{in}(k-p_2) \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-p_3) \end{Bmatrix} \tag{5.18}$$

where f_{dd} is a RNN function as shown in equation (5.14). Vector x_{dd} takes into account the previous and present samples of receiver input voltage and power supply voltage along with power supply current. In equation (5.18), p_1, p_2 and p_3 are constants that depend on the kind of receiver being modeled. The number of previous time instances depends on the complexity of the receiver being modeled. If the receiver being modeled is complex, then more number of previous time instances should be taken into account.

Similarly, the receiver input current is not only a function of the receiver input voltage but also a function of receiver power supply voltage. The procedure for estimating the receiver input current is similar to estimating the receiver power supply current. Receiver input current is represented in terms of receiver input voltage and receiver power supply voltage as shown below:

$$i_{in}(k) = f_{in}(x_{in}(k)) \tag{5.19}$$

$$x_{in}(k) = \begin{Bmatrix} i_{in}(k-1), i_{in}(k-2), ..., i_{in}(k-q_1), \\ v_{in}(k), v_{in}(k-1), ..., v_{in}(k-q_2) \\ v_{dd}(k), v_{dd}(k-1), ..., v_{dd}(k-q_3) \end{Bmatrix} \tag{5.20}$$

where f_{in} is a RNN function as shown in equation (5.14). In equation (5.20), q_1, q_2 and q_3 are constants that depend on the kind of receiver being modeled. The number of previous time instances depends on the complexity of the receiver being modeled.

Figure 5.11 shows PWL voltage sources connected at the receiver input and receiver power supply, respectively, for IBM driver ('DDR2'). Figure 5.12 shows the resultant IBM receiver input current and the power supply current, respectively. It can be also seen from Figure 5.12 that the RNN function (dotted line) accurately models the receiver input and power supply current.

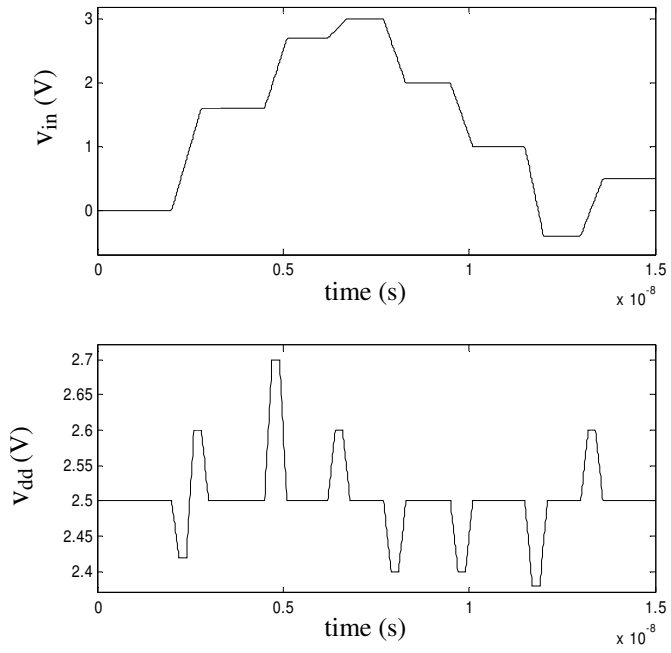


Figure 5.11 PWL voltage source connected at receiver input and receiver power supply.

5.3.2 Receiver Output Characteristics Modeling

The output characteristics of a receiver circuit are dependent on the power supply node (v_{dd}) and receiver reference voltage (v_{ref}). When multiple receivers are switching simultaneously, the noise generated at power supply node, v_{dd}, can affect the output of the receiver. Hence, the output voltage of a receiver circuit can be expressed as:

$$v_{out} = f_{out}(v_{in}, v_{dd}, v_{ref}) \tag{5.21}$$

Variation in power supply affects the magnitude of the receiver output voltage. This effect can be captured by representing the receiver output voltage as:

$$v_{out\_new} = v_{out} * \frac{v_{dd}}{VDD} \tag{5.22}$$

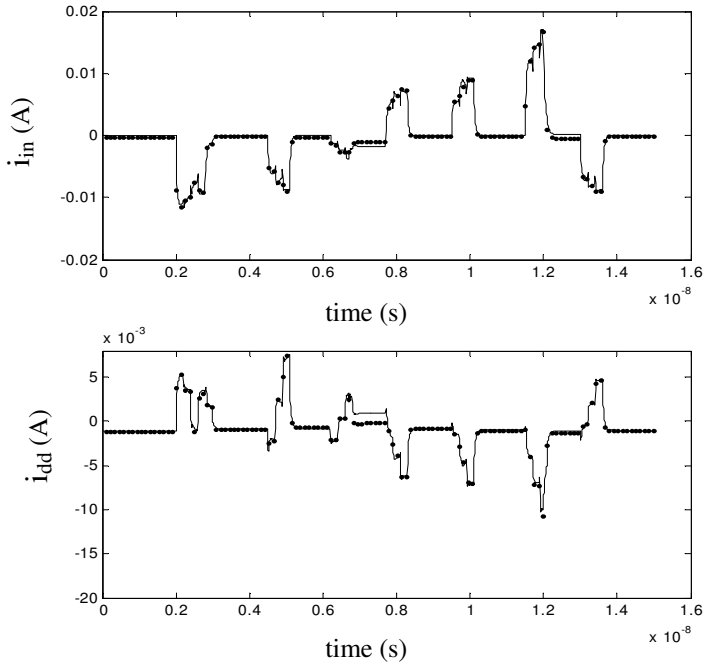


Figure 5.12 IBM DDR2 receiver input current from a PWL voltage source connected at receiver input and receiver power supply (straight line) and RNN model receiver input and power supply (dotted line).

Equation (5.22) accurately represents the effect of v_{dd} on v_{out} as the effect of v_{dd} variation is felt when the receiver output is at logic HIGH. Variation of v_{ref} can affect the delay of the signal through the receiver. Figure 5.13 shows the variation of v_{ref} for IBM DDR2 receiver circuit for a particular slew rate.

It can be seen from Figure 5.13 that the change of v_{ref} can result in variation of delay through the receiver circuit. Increase in v_{ref} results in increase of time delay through a receiver circuit. This variation is monotonic in nature and can be captured using a linear

or quadratic function. Figure 5.14 shows the procedure to include the effect of v_{dd} and v_{ref} on receiver output characteristics.

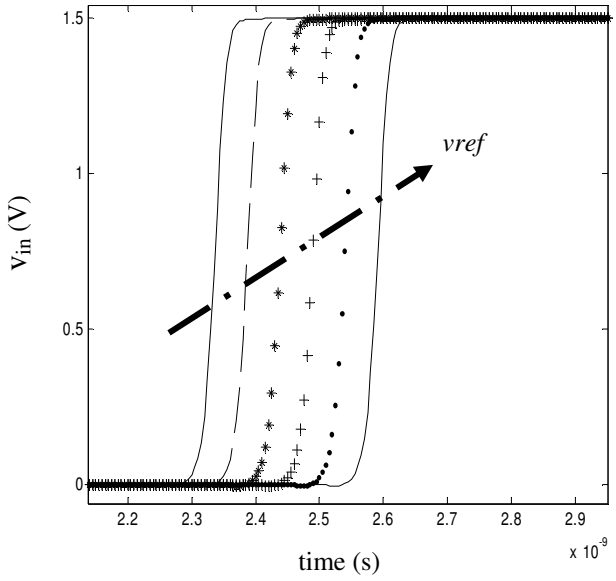


Figure 5.13 Effect of *vref* on the time delay through the receiver circuit.

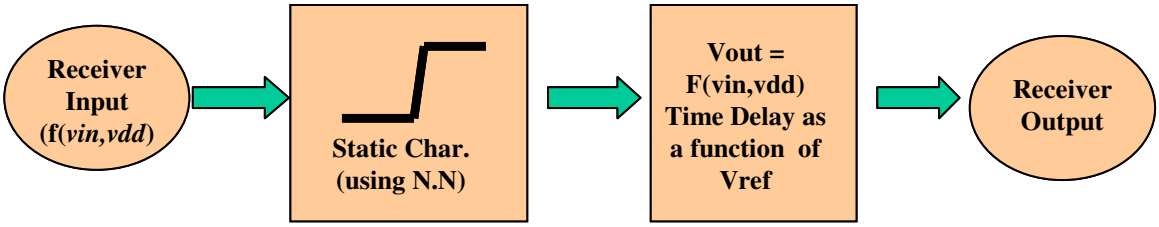


Figure 5.14 Procedure to model the output characteristics of a receiver circuit taking vref and vdd effect into account.

5.4 Spice Netlist Generation

Spice netlist generation of the proposed receiver modeling approach is similar to spice netlist generation of driver circuits. The input characteristics relation between receiver input current, input voltage, and power supply voltage can be expressed using

voltage-dependent voltage sources. The previous time instances of receiver input voltage, input current, and power supply voltage can be expressed using state equations (similar to driver circuits). A detailed explanation of spice netlist generation for drivers is given in chapter II. Static characteristics in modeling receiver output characteristics can be expressed using voltage-dependent voltage source. The time delay element can be incorporated using an additional voltage-dependent voltage source. A schematic of the spice netlist is given in Appendix B.

5.5 Test Results

5.5.1 Two-port Results

Test Case 1

A test case was setup where a 50-ohm PWL voltage source was connected to an ideal 50-ohm transmission line with 0.5 ns time delay which in turn fed the IBM ('DDR2') receiver. The voltage at the input of the receiver and the current at the end of the transmission line were measured using the behavioral RNN macromodel and the actual transistor-level receiver. IBM ('DDR2') receiver had a 2.5 V power supply. The PWL voltage source had amplitude of 3.5 V to excite the nonlinearity of the receiver. It can be seen from Figure 5.15 shows that the macromodel and the actual IBM results match very well. The actual IBM driver took 96 seconds and the macromodel took 4 seconds for the same simulation. All the simulations were carried on an IBM 2-GHz PC with 512 MB RAM.

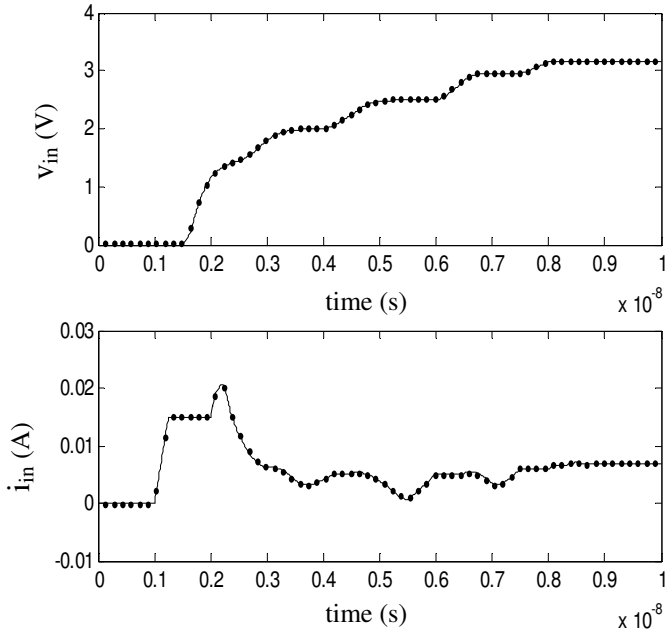


Figure 5.15 Voltage at receiver input and current at receiver input for IBM DDR2 (straight line) and RNN receiver macromodel input (dotted line).

Test Case 2

A more realistic test case was setup where an IBM ('AGPV3V2') driver was connected to a 75-ohm ideal transmission line with a 0.5 ns time delay. The transmission line was in turn connected to an AGPV3V2 receiver. A SFWFTD model was used to model the IBM driver ('AGPV3V2') and receiver circuit was also modeled using SFWFTD modeling technique. A seventh degree polynomial was used to model the DC characteristics of the receiver and the values of *p* and *pp* were 0 and 2.3. The driver was excited with a 1.5 V pulse with 0.3 ns rise time. The voltages at the near-end of the transmission line, the input of the receiver, and the output of the receiver were plotted as shown in Figure 5.16.

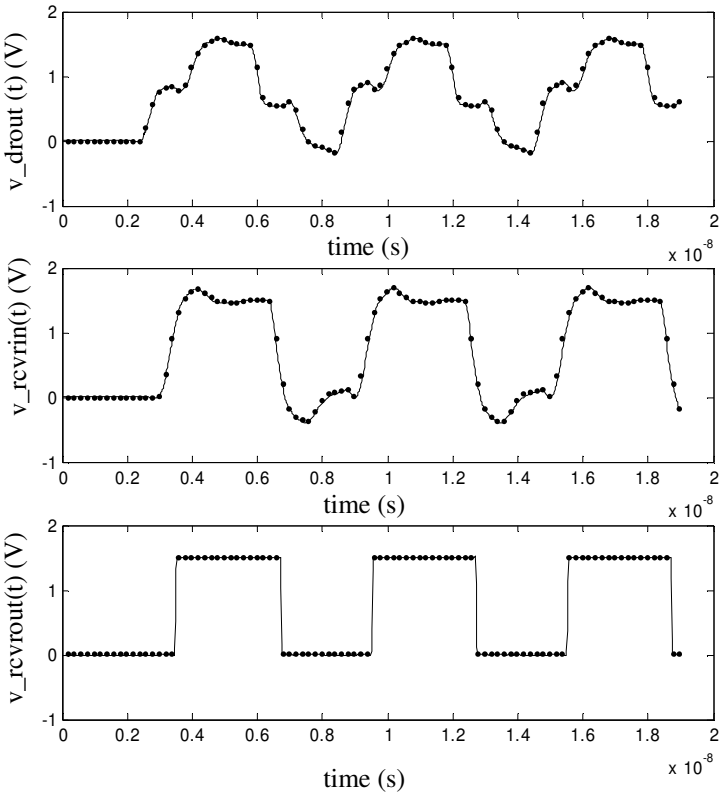


Figure 5.16 Driver output voltage, receiver input voltage, and receiver output voltage for IBM AGPV3V2 (straight line) and SFWFTD macromodel output voltage (dotted line).

It can be clearly seen that the modeled results match well with the IBM ('AGPV3V2') transistor-level driver and receiver models. The transistor-level driver and receiver circuit setup took 276 seconds for computation and the spline function with finite time difference approximation macromodel took less than 1.5 seconds for simulation. The macromodel was accurate and gave a timing error less than 20-30 ps. All the simulations were carried on an IBM 2-GHz PC with 512 MB RAM.

Test Case 3

Another realistic test case was setup where an IBM ('DDR2') driver was connected to a 75-ohm ideal transmission line with a 0.2 ns time delay. The transmission line was in

turn connected to an IBM ('DDR2') receiver. A RNN model was used to model the DDR2 driver and receiver circuits. A RNN model with 5 hidden neurons and with two previous time instances was used to model the receiver input characteristics. The driver was excited with a 2.5V pulse with 0.25 ns rise time. The voltages at the near end of the transmission line, the input of the receiver, and the output of the receiver were plotted as shown in Figure 5.17.

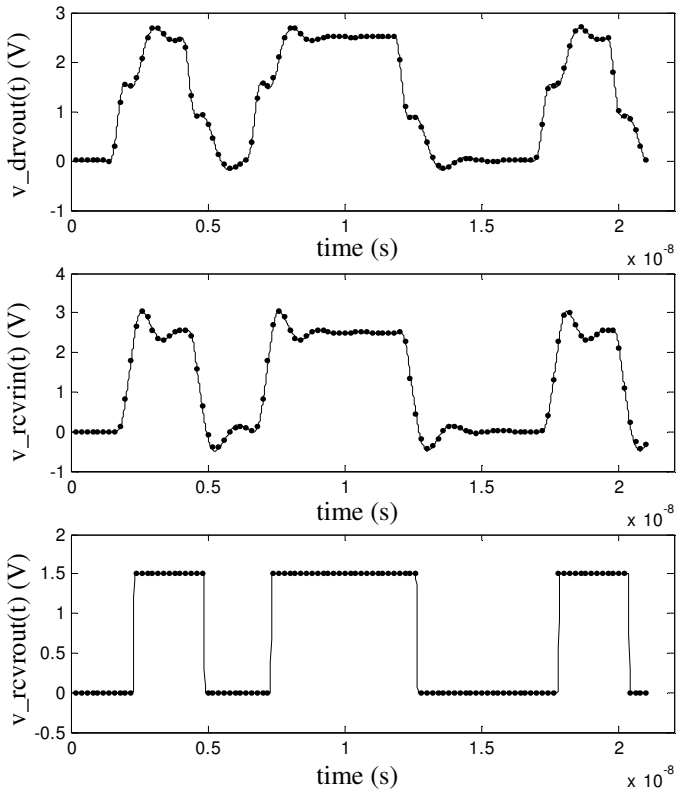


Figure 5.17 Driver output voltage, receiver input voltage, and receiver output voltage for IBM AGPV3V2 (straight line) and RNN macromodel output voltage (dotted line).

It can be clearly seen that the modeled results match well with the IBM transistor-level receiver model. The transistor-level ('DDR2') setup took 290 seconds for computation and the macromodel took less than 4 seconds for simulation. The

macromodel was accurate and gave timing error of less than 15-20 ps. All the simulations were carried on an IBM 2-GHz PC with 512 MB RAM.

5.5.2 Four-port Results

The proposed receiver modeling approach has been extended to multiple ports and the accuracy and speed-up of these macromodels is verified by generating some test cases.

Test Case 1

In this test case, an IBM receiver ('AGPV3V2') was connected to a plane pair which was modeled using the cavity resonator method [C9]. The plane pair was 10 cm × 6 cm in length and width. It had six ports on each plane, V_{dd} and *Gnd* as shown in Figure 5.18.

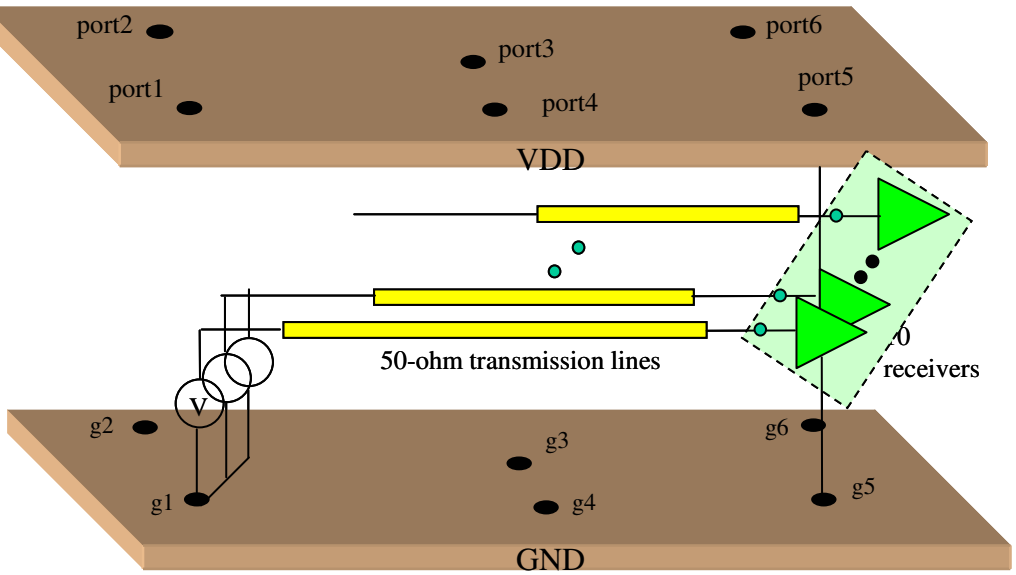


Figure 5.18 Plane pair model generated using cavity resonator method. Both planes have six ports each.

Ten receiver circuits were connected at port five and all the receivers were connected to a 50-ohm ideal transmission lines. All the transmission lines were identical and were

connected to an ideal voltage source through a 50-ohm resistance. The voltage source generated square pulses with 0.3 ns rise time and 1.5 V magnitude. The power supply (v_{dd}) was at port three. Three additional ports were used for probing. The resulting SSN was calculated using both actual transistor level receiver circuit and RNN model. A RNN model with one previous time instance and three hidden neurons was used to model sub-models f_{in} and f_{1dd}. Figure 5.19 shows the SNN at ports five, six, and four.

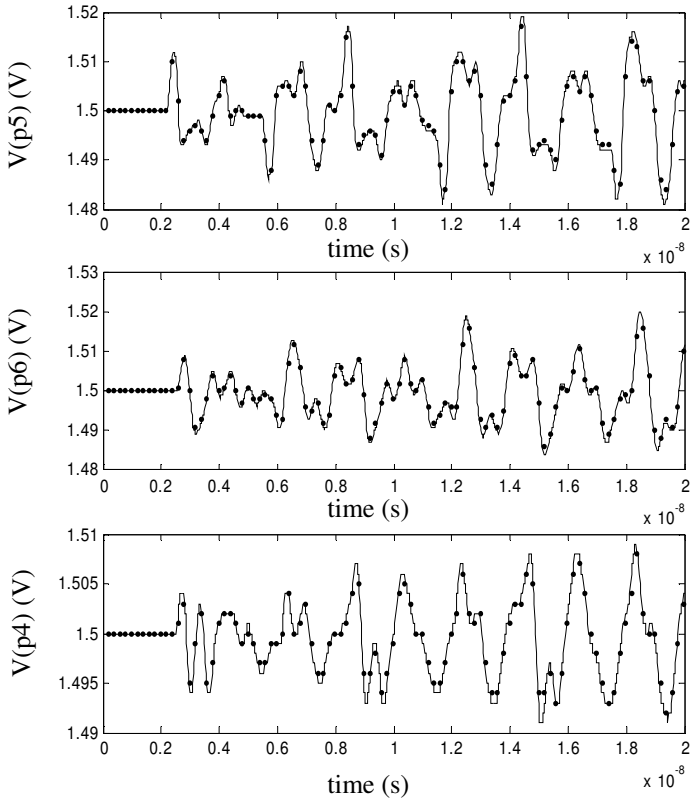


Figure 5.19 SSN at port five, port six, and port four when 10 IBM AGPV3V2 receiver circuits are switching (straight line) and 10 RNN receiver macromodels are switching (dotted line), respectively.

Figure 5.20 shows SSN at ports one and two. It can be clearly seen that the modeled and the transistor-level receiver circuit SSN waveforms match accurately. The transistor-

level receiver circuit took 1397 seconds for simulation and RNN model took 6.5 seconds for the same simulations. RNN model was 200 times faster than the IBM transistor-level receiver circuit.

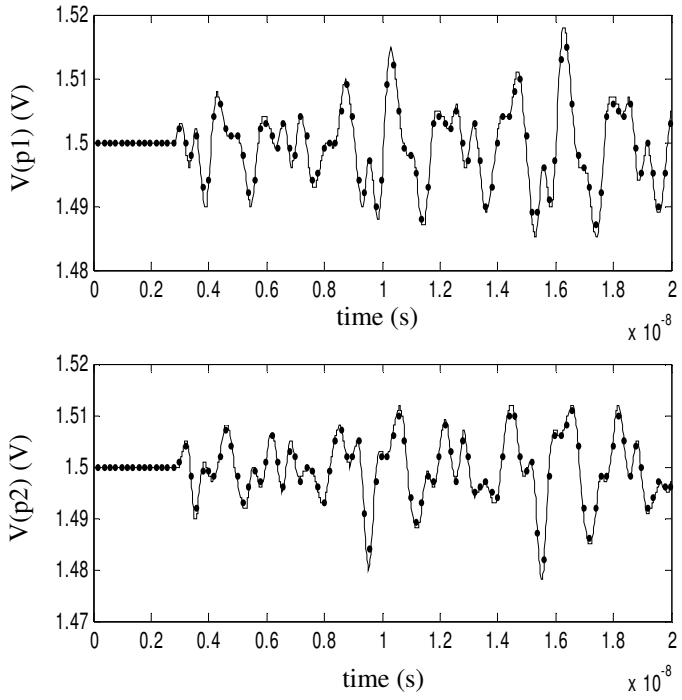


Figure 5.20 SSN at port one and port two when 10 IBM AGPV3V2 receiver circuits are switching (straight line) and 10 RNN receiver macromodels are switching (dotted line), respectively.

Test Case 2

Another more realistic test case was generated where IBM ('DDR2') driver and receiver circuits were connected to a plane pair modeled using cavity resonator method. IBM ('DDR2') driver has a power supply voltage of 2.5 volts and it is driven at 250 MHz with a rise time of 1.25 ns. The pane pair had dimensions of 6 cm X 4 cm with four ports on V_{dd} plane and four ports on *Gnd* plane. All the drivers were identical, driving ideal 50-ohm transmission lines and were connected at port one. The power supply node was at

port three. All the transmission lines were terminated at port two at the receiver. Figure 5.21 shows the plane pair used to model the power supply noise when six drivers and receivers were simultaneously switching. The IBM driver was modeled using RNN approach. The regressor vector x consists of present samples of power supply voltage and driver output voltage and past samples of driver output current, power supply current, power supply voltage, and output voltage.

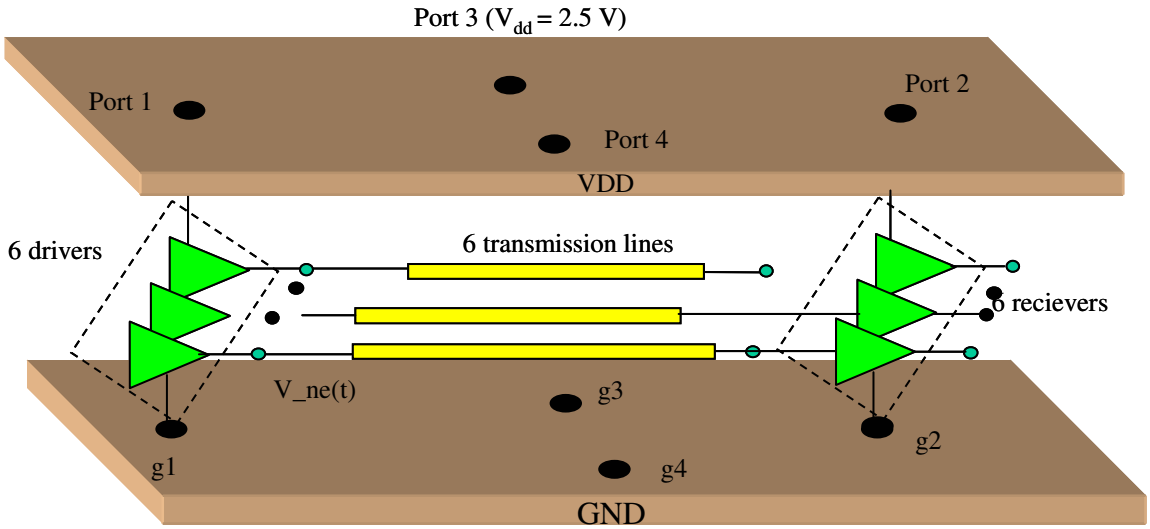


Figure 5.21 Plane pair model generated using cavity resonator method. Both planes have four ports each.

The RNN model for sub-functions $f_{1d,2d}$ required one hidden layer with two hidden neurons. Modified BPTT training algorithm was used to estimate the weights of the RNN model. A RNN model with two previous time instances and two hidden neurons was used to model sub-models f_{1dd} and f_{in}. It can be seen from Figure 5.22 that the driver output, receiver input, and receiver output voltage waveforms are accurately modeled using RNN macromodel.

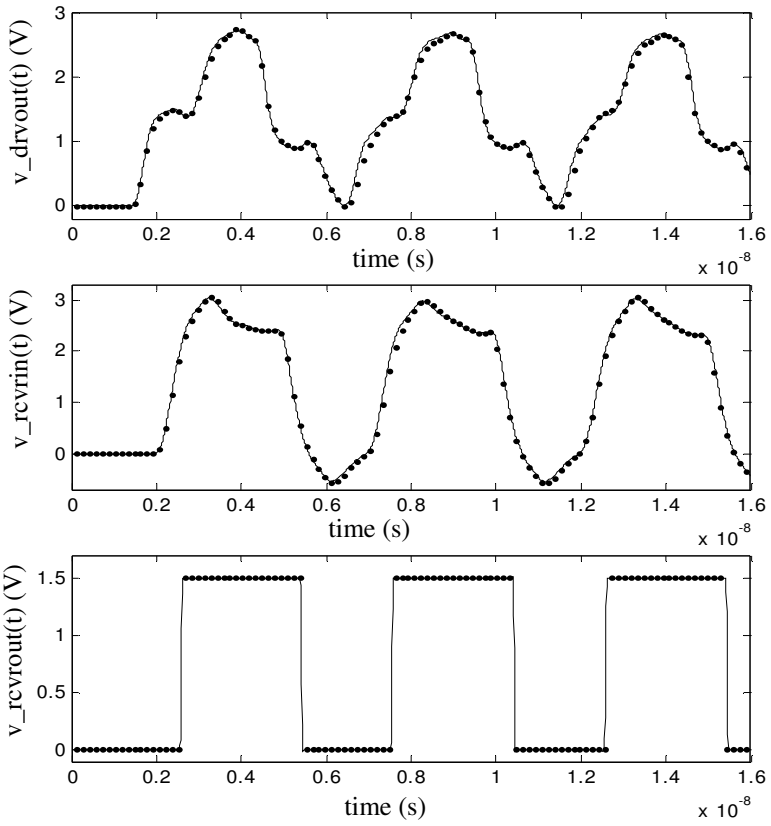


Figure 5.22 Driver output voltage, receiver input voltage, and receiver output for IBM DDR2 (straight line) and RNN driver macromodel output voltage (dotted line), respectively.

Figure 5.23 captures the SSN accurately at ports one, two, and four, respectively, when all six drivers and receivers are switching simultaneously. The actual IBM transistor driver and receiver model took 1652 seconds for the simulation and RNN model took 87 seconds for the same simulation. RNN model was at least 20 times faster than the transistor-level circuit model when the time domain simulations were carried out for three cycles. In general, the computational time speed-up between the RNN model and actual transistor-level circuit model increases with the increase in complexity of the test set-up.

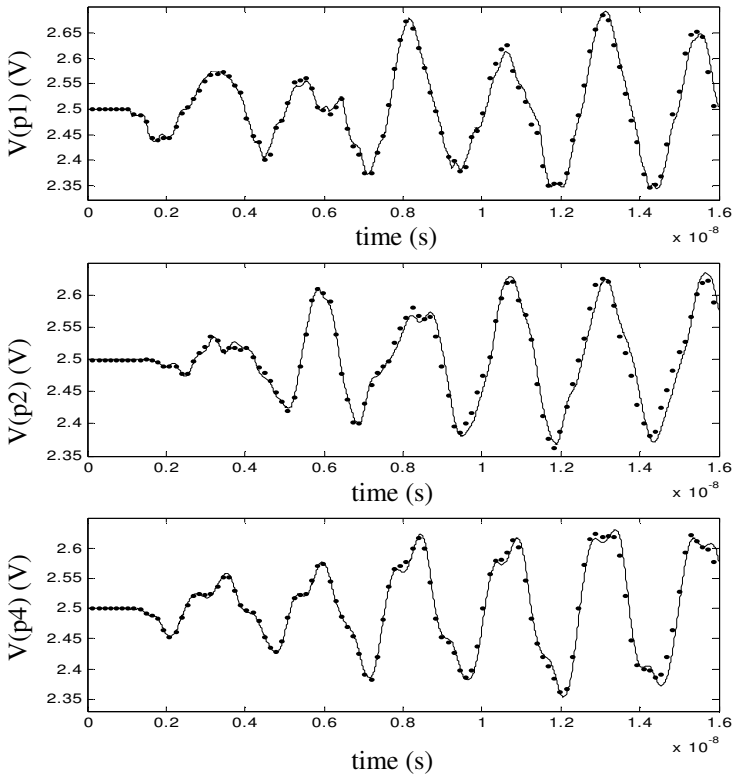


Figure 5.23 SSN at port one, port two, and port four when six IBM DDR2 driver and receiver circuits are switching (straight line) and six RNN driver and receiver macromodels are switching (dotted line), respectively.

Test Case 3

In this test case, 16 IBM ('AGPV3V2') receiver circuits were connected to 16 75-ohm ideal transmission lines. All the transmission lines were connected to an ideal voltage source through a 50 ohm resistance. The transmission lines have a delay of 0.2 ns. The voltage source generated square pulses with 0.3 ns rise time and 1.5 V magnitude. The power supply (v_{dd}) was connected through a 5 nH inductance to a 1.5 V DC ideal source. The resulting SSN was calculated using both actual transistor-level receiver circuit and RNN model. The receiver input voltage waveforms and output voltage waveforms were plotted for both the RNN model and the actual transistor-level

receiver model. It can be seen from Figure 5.24 that the SSN estimated using the RNN model matches accurately with the transistor-level circuit. Figure 5.25 shows the voltage waveforms at the receiver input and receiver output. It can be seen from Figure 5.25 that the output voltage of the receiver shows the effect of v_{dd}.

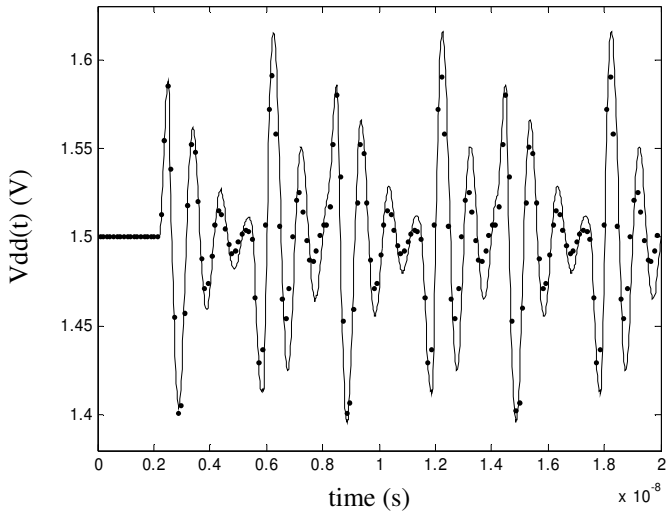


Figure 5.24 SSN when 16 IBM APV3V2 receiver circuits are switching (straight line) and 16 RNN receiver macromodels are switching (dotted line).

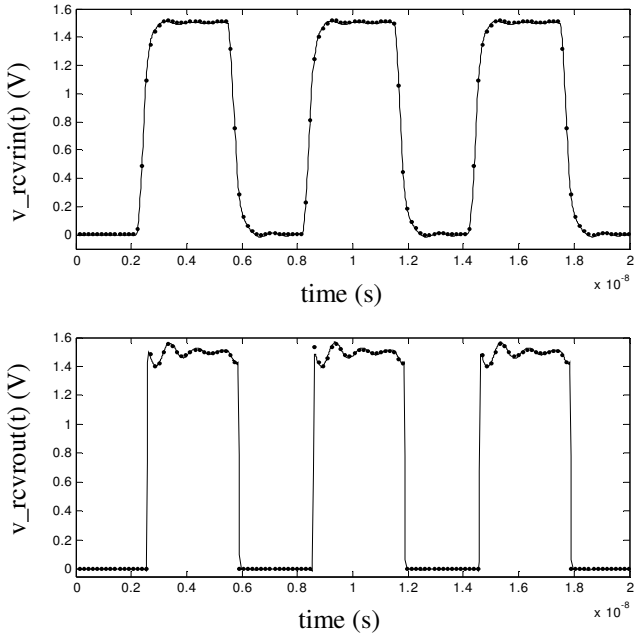


Figure 5.25 Receiver input voltage and receiver output voltage for IBM AGPV3V2 (straight line) and RNN receiver macromodel input voltage (dotted line), respectively.

The transistor level receiver circuit took 1283 seconds for simulation and RNN model took 8 seconds for the same simulations. RNN model is 150 times faster than the IBM transistor level receiver circuit.

5.6 Summary

A modeling methodology for modeling transistor-level receiver circuits has been proposed in this chapter. Receiver circuits are difficult to model as the input to the receiver is analog in nature and the output is digital. Receiver modeling has been divided into modeling receiver input characteristics (where the receiver acts like a capacitive load) and receiver output characteristics (that forms input to processors). The proposed modeling technique accurately models both the input as well as output characteristics of the receiver circuit. Spice macromodels generated using this methodology give high computational speed-up compared to transistor-level receiver circuits.

The proposed modeling technique has also been extended to multiple ports to estimate sensitive effects like SNN accurately. In this chapter, extension of the receiver modeling approach to include the effect of power supply port has been described in detail. SSN affects both the receiver input as well as output characteristics. Test cases have been generated to test the accuracy of the modeling approach. It has been shown that the effect of SSN on both the receiver input and output voltages have been captured accurately. Table 5.1 summarizes the accuracy, computational speed-up, and memory reduction for all the test cases using SFWFTD and RNN receiver macromodels. It can be seen that receiver macromodels result in huge computational speed-up and reduction in CPU memory usage.

Table 5.1 Accuracy, simulation speed-up and memory reduction for test cases.

	Simulation time Speed-up	Memory Reduction	Mean Square Error
Two port Case1	24X	200X	10^{-4} to 10^{-6}
Two port Case2	184X	180X	10^{-3}
Two port Case3	19X	185X	10^{-4}
Four port Case1	200X	320X	10^{-5}
Four port Case2	20X	130X	10^{-4}
Four port Case3	150X	580X	10^{-3} to 10^{-4}

CHAPTER VI

SCALABLE DRIVER AND RECEIVER MACROMODELS

The output voltage and current of a driver circuit is dependent on its power supply voltage, temperature, and process variation. Slight variations in the above parameters affect the output voltage and current of driver circuits. Input/Output buffer information specification (IBIS) macromodels take into account the effect of power supply voltage, temperature, and process variations. But it has been seen from chapter I that IBIS models cannot accurately model nonlinear driver and receiver circuits. It is always efficient to have a spice netlist of a driver macromodel that is scalable and can take into account the effect of temperature, power supply voltage, and process variations instead of having multiple spice netlists for each parameter variation in the driver circuit. In the development of a macromodel, scalable characteristics provide great convenience for the circuit designer and therefore greatly reduce the design cycle and time to market of product.

Increased complexity in today's high speed systems is resulting in large number of design and operational parameters, which are needed to be considered in order to perform system level signal integrity analysis. Statistical variations in transmission line geometries, temperature, and power supply voltage are resulting in degradation of system performance. Figure 6.1 shows distortion in the digital waveform in the presence of statistical variation. Traditionally, signal integrity engineers verify the worst-case

combination to finalize designs. However, worst-case design and operation scenarios may not be verified for cost-effective, high-performance designs [F1]. Scalable macromodels help in performing statistical and yield analysis on high-speed systems and deriving relationship between the variations in design and manufacturing parameters and signal integrity targets.

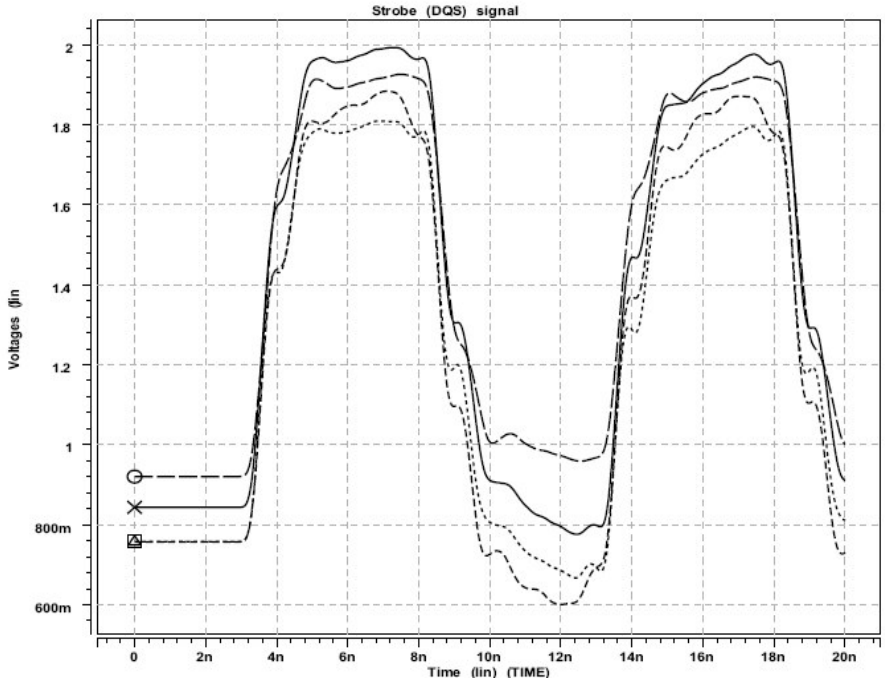


Figure 6.1 Statistical variations of signal integrity in digital systems.

In this chapter, scalable macromodels for differential driver circuits and single-ended driver and receiver circuits is discussed in detailed in section 6.1. The accuracy of scalable macromodels has been verified on various test-cases in section 6.2. Use of scalable macromodels in performing statistical and yield analysis on high-speed systems is discussed in section 6.3 along with test results. Section 6.4 summarizes the chapter.

6.1 Scalable Macromodels

6.1.1 Temperature Scalable Differential Driver Macromodels

Variation of temperature has effect on the differential driver output voltage and current. Figure 6.2 shows a scenario where the IBM differential driver ('bsdb25') temperature is varied from 0 °C to 100 °C and the driver output P and N port voltages waveforms are plotted.

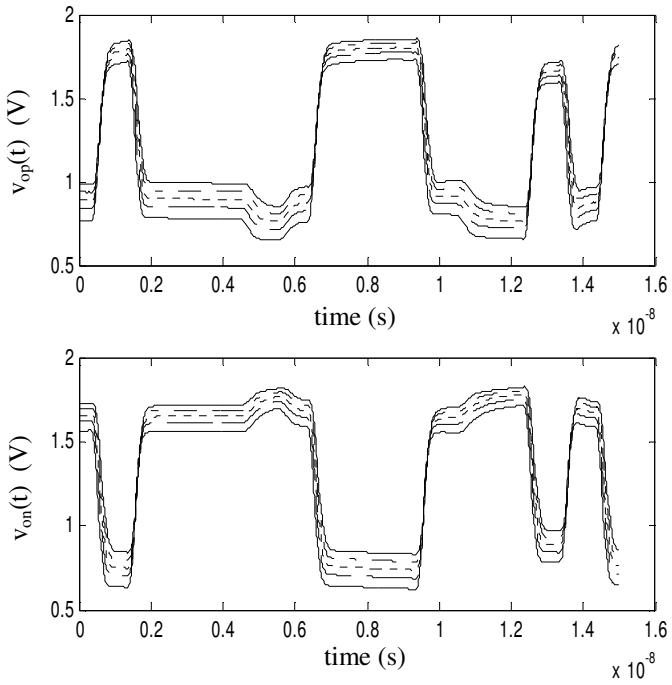


Figure 6.2 Voltage waveforms at the near-end of the transmission line for IBM transistor-level driver when temperature is varying from 0 to 100° C.

It can be seen from Figure 6.2 that the effect of temperature on the driver voltage waveforms is fairly linear. For most signal integrity and electromagnetic compatibility (EMC) simulations, the driver circuits are usually run at three different temperatures. For the best case scenario, the temperature is set around 0 °C , for typical scenario, the driver circuit temperature is set at room temperature 27 – 30 °C, and for the worst case

simulations, the temperature is set around 100 °C. In [F2], the effect of temperature on single-ended driver circuits is shown.

In order to take into account the effect of temperature, Lagrange's interpolation technique can be used. It is known that for

$$y_i = p(x_i); for\ i = 1:n \tag{6.1}$$

$$p(x) = \frac{f(x)}{(x-x_1)f'(x_1)} + \frac{f(x)}{(x-x_2)f'(x_2)} + ... + \frac{f(x)}{(x-x_{n-1})f'(x_{n-1})} + \frac{f(x)}{(x-x_n)f'(x_n)} \tag{6.2}$$

where,

$$f(x) = (x-x_1)(x-x_2)...(x-x_n) \tag{6.3}$$

In chapter IV it has been shown that a differential driver circuit can be modeled by expressing the output currents in terms of output voltages as shown below:

$$i_{op}(t) = w_{1p}(t) f_{1p}(v_{op}, v_{on}, i_{op}) + w_{2p}(t) f_{2p}(v_{op}, v_{on}, i_{op}) \tag{6.4}$$

$$i_{on}(t) = w_{1n}(t) f_{1n}(v_{op}, v_{on}, i_{on}) + w_{2n}(t) f_{2n}(v_{op}, v_{on}, i_{on}) \tag{6.5}$$

where i_{op} and i_{on} are the currents at the output ports of the differential driver. The output voltages are represented by v_{op} and v_{on}. Sub-models $f_{1p/1n}$ and $f_{2p/2n}$ capture the nonlinear relation between port P/N current and port P and N voltages when the differential driver input is set HIGH and LOW, respectively, using recurrent neural network (RNN) functions. Weighting functions $w_{1p/1n}$ and $w_{2p/2n}$ help in transitioning sub-models $f_{1p/1n}$ and $f_{2p/2n}$ from one state to another.

Since the effect of temperature is almost linear, sub-models f_{1p}, f_{2p}, f_{1n}, and f_{2n} are estimated for the best case and the worst case temperatures. Once the best case and the

worst case sub-models for f_{1p}, f_{2p}, f_{1n}, and f_{2n} are estimated, Lagrange's interpolation technique can be used to estimate sub-models at any other temperature value. For example, sub-models f_{1p} and f_{2p} can be represented as shown below:

$$f_{1p} = \frac{(temp-100)}{(-100)} f_{1p,t=0} + \frac{(temp)}{(100)} f_{1p,t=100}$$
$$f_{2p} = \frac{(temp-100)}{(-100)} f_{2p,t=0} + \frac{(temp)}{(100)} f_{2p,t=100} \quad (6.6)$$

Sub-models f_{1n} and f_{2n} can also be estimated in a similar fashion. Once sub-models are estimated as shown above, weighting functions w_{1p}, w_{2p}, w_{1n}, and w_{2n} can be expressed using Lagrange's interpolation technique as shown below:.

$$w_{1p} = \frac{(temp-100)}{(-100)} w_{1p,t=0} + \frac{(temp)}{(100)} w_{1p,t=100}$$
$$w_{2p} = \frac{(temp-100)}{(-100)} w_{2p,t=0} + \frac{(temp)}{(100)} w_{2p,t=100} \quad (6.7)$$

6.1.2 Extension to Multiple Variables

Scalability of RNN modeling technique can be extended to multiple variables. In chapter II, it has been shown that the driver output current can be modeled in terms of its output voltage as shown below:

$$i_o(t) = w_1(t) f_1(v_o, i_o) + w_2(t) f_2(v_o, i_o) \quad (6.8)$$

where i_o is the driver output current and v_o is driver output voltage. Sub-models f_1 and f_2 are represented using RNN functions. Weighting functions w_1 and w_2 help f_1 and f_2 transition from one state to another. In chapter V, it has been shown that a receiver circuit input characteristics can be modeled by expressing the receiver input current, i_{in}, in terms of receiver input voltage, v_{in}, using RNN functions as:

$$i_{in}(k) = f_{in}(i_{in}, v_{in}) \tag{6.9}$$

The same principle of scalability used for differential driver circuits can be applied to scale single-ended driver and receiver RNN models with respect to temperature, power supply voltage, and process variation. Table 6.1 provides the best case, typical case, and the worst case values of temperature, power supply voltage, and process variation for a single-ended IBM ('DDR2') driver/receiver circuit. IBM ('DDR2') driver circuit is a dual data rate (DDR) driver circuit with a power supply of 2.5 V and an operational frequency of 400 MHz.

Table 6.1 Temperature, power supply voltage and process variations for the best case, typical case and worst case for IBM DDR2 driver circuit.

	Best Case	Typical Case	Worst Case
Temperature (t)	0 °C	27 °C	100 °C
Power supply voltage (*vdd*)	2.7 V	2.5 V	2.3 V
Process variation (σ)	-1.1217	0	+1.2754

It can be seen from Table 6.1 that there are 3^3 combinations of variation of parameters. In order to do an efficient signal integrity analysis at a design stage, effect of variation of each of these parameters should be accurately captured. Assuming the effect of each of the variables is almost linear on the driver output current and voltage, Lagrange's interpolation technique can be used taking only the best case and the worst case values into consideration. Assuming all the three variables in their worst case state is represented as 000 then all the three variables in their best case is represented as 111. There are $2^3 = 8$ combinations that need to be taken into account.

In order to take into account the effect of all the three variables, sub-model f_1 can be expressed as shown below:

$$f_1 = \frac{(t)(\sigma-1.2574)(vdd-2.7)}{(100)(-2.3971)(-0.4)} f_{1,000} + \frac{(t)(\sigma-1.2754)(vdd-2.3)}{(100)(-2.3971)(0.4)} f_{1,001} + \ldots + \frac{(t-100)(\sigma+1.1217)(vdd-2.3)}{(-100)(2.3971)(0.4)} f_{1,111} \quad (6.10)$$

Sub-model f_2 and receiver sub-model f_{in} can be expressed in similar fashion. Weighting functions w_1 and w_2 are expressed as:

$$w_n = \frac{(t)(\sigma-1.2574)(vdd-2.7)}{(100)(-2.3971)(-0.4)} w_{n,000} + \frac{(t)(\sigma-1.2754)(vdd-2.3)}{(100)(-2.3971)(0.4)} w_{n,001} + \ldots + \frac{(t-100)(\sigma+1.1217)(vdd-2.3)}{(-100)(2.3971)(0.4)} w_{n,111}; n = 1,2 \quad (6.11)$$

6.2 Test Results

6.2.1 Differential Driver Results:

Two test cases have been designed to validate the accuracy of scalable differential driver RNN macromodel. These test cases use an IBM transistor-level differential driver ('bsdb25'). The IBM driver was operated at 1 GHz with a power supply voltage of 1.8V. The IBM driver was designed for a 100-ohm differential load. The first test case involves verifying the accuracy for temperature values that are with in the Lagrange's interpolation range and the second test case involves verifying the accuracy for temperature values outside of Lagrange's interpolation range.

1) Test Case 1: In first test case, the IBM driver ('bsdb25') was connected to a 100-ohm differential lossy transmission line that was 12-inches long. The transmission line was in turn terminated with a 70-ohm resistor. The voltage waveforms at the near-end of the transmission line of both the output ports were measured from the RNN macromodel and the actual transistor-level driver model for two temperature values, 27 °C and 75 °C, respectively. It can be seen from Figure 6.3 that the near-end voltage waveforms for both

the transistor-level driver model and RNN macromodel match accurately. It can be seen that the scalable RNN macromodel results in accurate results for interpolation between the best and the worst case temperatures with a speed-up of 10-15 X.

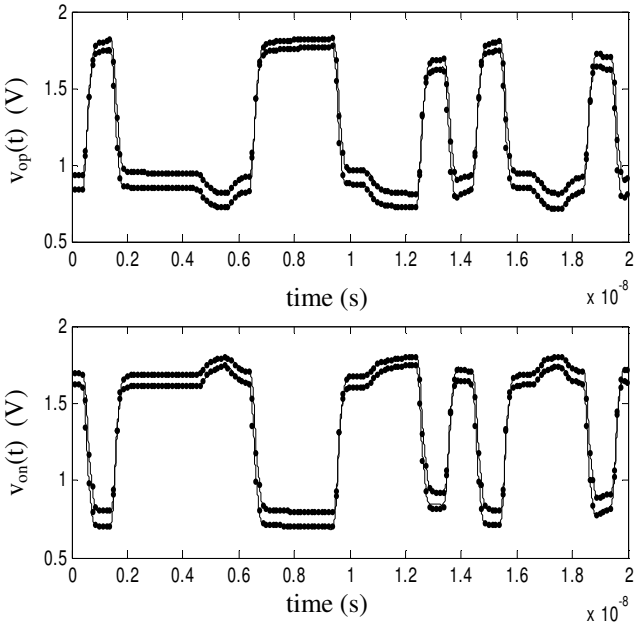


Figure 6.3 Voltage waveforms at the near-end of the transmission line for IBM transistor-level driver (straight line) and RNN macromodel (dotted line) when driver temperatures are 27° and 75° C.

2) Test Case 2: In this test case, the accuracy of scalable RNN macromodel for temperatures outside the range of Lagrange's interpolation was tested. The IBM driver ('bsdb25') was connected to a 100-ohm differential lossy transmission line that was 12-inches long. The transmission line was in turn terminated with a 130-ohm resistor. The voltage waveform at the near-end of the transmission line for both the output ports was measured from the RNN macromodel and the actual transistor-level driver model for two temperatures -25 °C and 125 °C, respectively. From Figure 6.4 it can be seen that the near voltage waveforms for both the transistor-level driver model and the RNN macromodel match accurately. The scalable macromodel not only gives accurate results

for temperature values within the Lagrange interpolation range but also outside the interpolation range. Scalable RNN macromodel was 10-15X faster than transistor-level driver circuit.

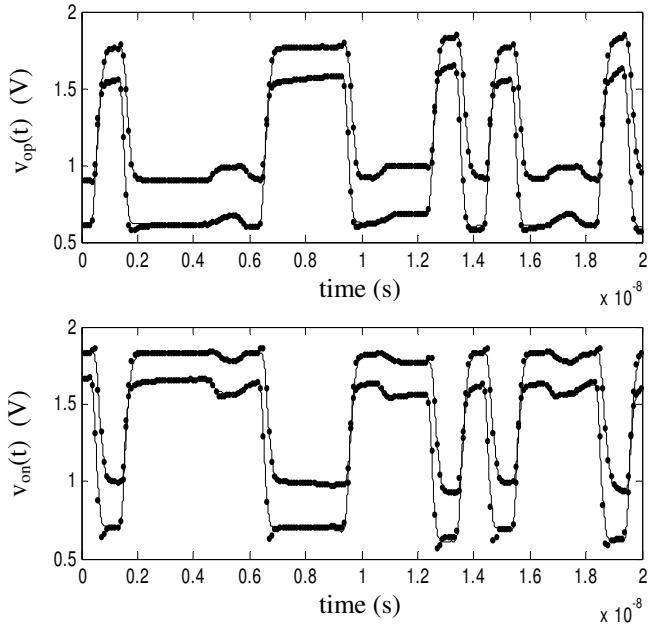


Figure 6.4 Voltage waveforms at the near-end of the transmission line for IBM transistor-level driver (straight line) and RNN macromodel (dotted line) when driver temperatures are -25° and 125° C.

6.2.2 Single-ended Driver and Receiver Results:

1) Test Case 1: A test case was designed to test the interpolation accuracy of the scalable model. In this test case, the IBM ('DDR2') driver circuit was connected to a 50-ohm ideal transmission line, which in turn was terminated with a 2 pF capacitance. This test case was generated to verify the extrapolation accuracy of the scalable macromodel. In this test case, the process variation was -1.5, the power supply voltage was 2.2 V, and the temperature was 125 °C. It can be seen from Figure 6.5 that the voltage waveforms at the near-end and the far-end of the transmission line match accurately with transistor-level driver circuit. RNN scalable macromodel was 20 times faster than transistor-level circuit

model.

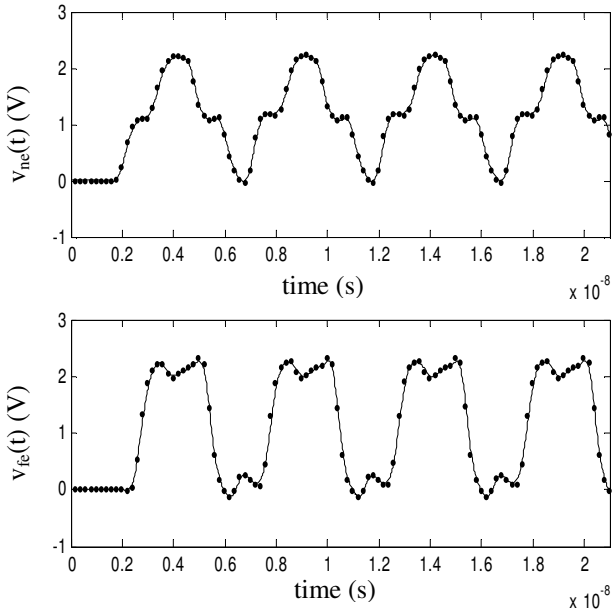


Figure 6.5 Voltage waveforms at the near-end and far-end of the transmission line for IBM DDR2 transistor-level driver (straight line) and RNN macromodel (dotted line) when driver t is 125°, σ is -1.5 and v_{dd} is 2.2 V.

2) Test Case 2: In this test case, the process variation was changed to + 1.5, temperature was made -25 °C, and the power supply voltage was made 2.8 V. The voltages at the near-end and the far-end of the transmission line from scalable RNN model and actual transistor-level driver model are shown in Figure 6.6. It has been seen that there is a good correlation between the transistor-level circuit and RNN macromodel waveforms with a speed-up of 20X. Hence, it can be seen that the scalable macromodel is accurate beyond the Lagrange's range of interpolation.

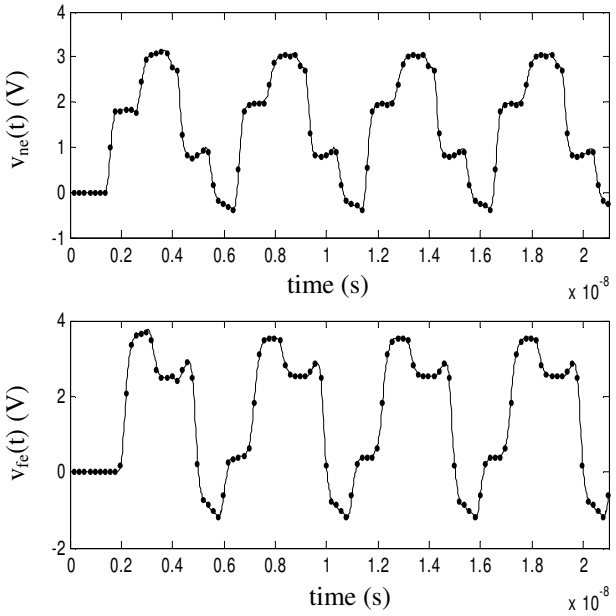


Figure 6.6 Voltage waveforms at the near-end and far-end of the transmission line for IBM DDR2 transistor-level driver (straight line) and RNN macromodel (dotted line) when driver t is -25°, σ is 1.5 and v_{dd} is 2.8 V.

3) Test Case 3: In this test case DDR2 driver and receiver circuits were connected through a 50-ohm ideal transmission line with 0.5 ns delay. The accuracy of scalable driver and receiver RNN macromodels is tested in this case. The voltage waveforms at the near-end and the far-end of the transmission line were plotted in Figure 6.7 for IBM DDR2 transistor-level driver circuit when temperature (t) is 27 °C, process variation (σ) is 0, and power supply voltage (vdd) is 2.5 V. There is good correlation between transistor-level driver and receiver circuits and scalable driver and receiver macromodels with a speed-up of 25 – 35X.

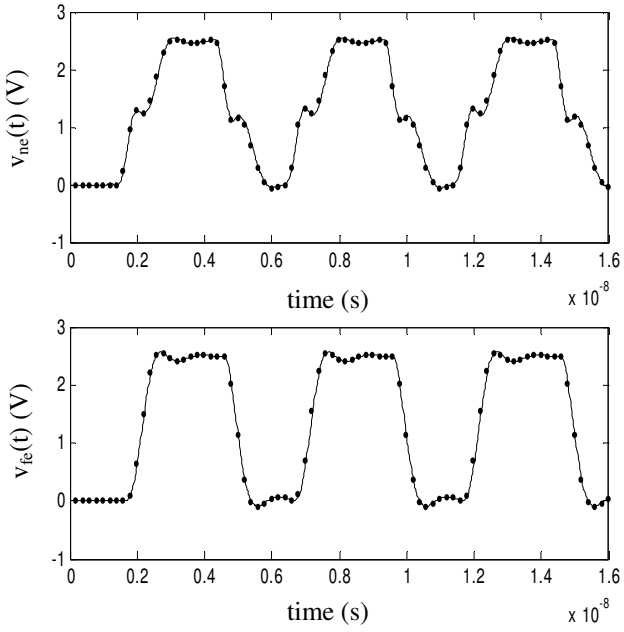


Figure 6.7 Voltage waveforms at the near-end and far-end of the transmission line for IBM DDR2 transistor-level driver-receiver circuit (straight line) and RNN macromodel (dotted line) when driver t is 27°, σ is 0, and v_{dd} is 2.5 V.

6.3 Statistical and Yield Analysis

Some of the most common signal integrity measures are voltage and timing margins, noise, transmission line properties, and crosstalk. With increased system complexity and frequency of operation, it is becoming a challenge to satisfy these signal integrity measures simultaneously. Long interconnects, short rise and fall times, and high operation frequencies distort the signal. Signal integrity measures are related to various design and operational parameters that are random variables resulting from manufacturing and operational uncertainties. Chip slew rates and transistor speed, transmission line geometries, operating temperature, and power supply voltages are considered as significant design and operational parameters. Statistical variations in these design and operational parameters may result in degradation of performance. For example, power supply variations affect voltage levels, increase signal delays, and reduce

voltage margins. To avoid such system failures, statistical distribution of design and operational parameters, and their effects on performance should be studied.

Increasing PC and server performance places severe demands for higher bandwidths on the signal busses interconnecting processors, memory units and other control chips. Manufacturing tolerance is expected to be very small at high data rates and future digital systems would be subjected to very challenging signal integrity constraints, and very narrow tolerance margins. With increased system complexity, a large number of design and operational parameters should be considered for meeting the system-level signal integrity targets. On the other hand, because of the narrowing timing and voltage margins, statistical variations in design and operational parameters are becoming more significant. In an electrical system, the classical approach to account for process and operational uncertainties is the worst-case analysis. With a large number of performance measures, it becomes very difficult to find the worst-case parameter combination for each performance measure. Second, the worst-case combination, where all design parameters are at their extremes, has very low probability of occurrence [F3]. As an example, if a normal distribution is considered as shown below:

$$f(x) = \frac{1}{\sigma\sqrt{2\Pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \quad for -\infty \le x \le \infty \tag{6.14}$$

where μ is mean and σ is standard deviation of *x*. The shape of the normal distribution curve is bell shape as shown is Figure 6.8. The curve is symmetrical about the point x = μ. The total area of the curve from $x = -\infty$ to $x = +\infty$ is 1 as shown below:

$$\int_{-\infty}^{+\infty} f(x)dx = 1 \tag{6.15}$$

It should be noted that 99.7% of the area under the probability density function of normal distribution is with $x = \mu + 3\sigma$ and $\mu - 3\sigma$. Values outside this range have little probability of occurrence.

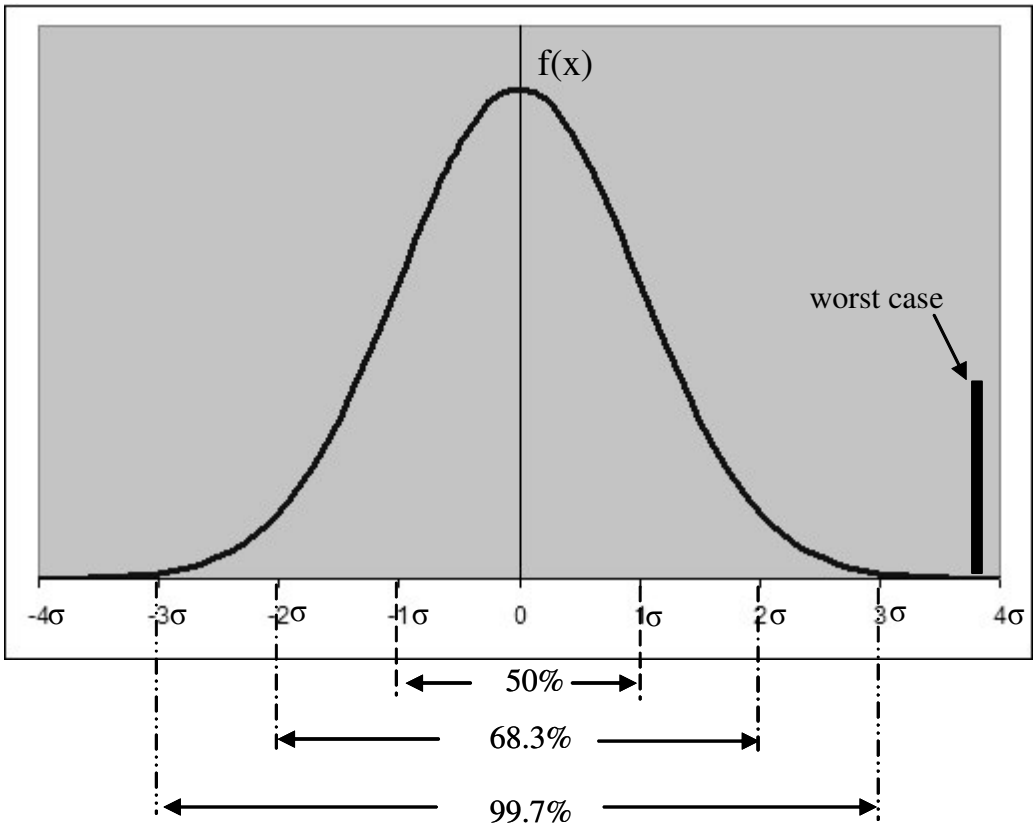


Figure 6.8 Probability density curve of Normal Distribution.

Instead of simulating the worst-case conditions, a statistical and yield analysis methodology is required to achieve cost effective high performance systems.

Parametric yield is defined as the percentage of the circuits or systems satisfying performance specifications in the presence of statistical perturbations. The most straightforward and common method to estimate parametric yield is Monte Carlo analysis [F4]. This technique depends on simulating a large number of design parameter combinations for generating the performance statistics. The values of the design

parameters are generated from random variables with associated probability distributions and correlations. Then, the yield is approximated as the ratio of the number of acceptable instances to the total number of Monte Carlo runs. This can be formalized as:

$$Y = \int_{-\infty}^{+\infty} z(x) f(x) dx \tag{6.12}$$

where $z(x)=1$ if all design values (x) satisfy the specifications, and $z(x)=0$ otherwise.

In equation (6.12), $f(x)$ is the joint probability density function of design parameters. Then yield can be estimated as:

$$\hat{Y} = \frac{1}{N} \sum_{i=1}^{N} z(xi) \tag{6.13}$$

Monte Carlo is advantageous when the statistical parameter distributions and correlations between them are too complicated to represent as analytic functions [F5]. For reasonable simulation times, Monte Carlo method results in a yield figure.

6.3.1 Test Results

Scalable macromodels consume less CPU memory and simulation time and help in performing efficient statistical analysis of today's high-speed systems that may require large number of simulations. A realistic test case was created where DDR2 driver was connected to a multi-drop interconnect line as shown in Figure 6.9. The voltage at the end of DIMM D was measured for variations in interconnect length, driver power supply voltage, and driver temperature. The temperature is varied in a Gaussian distribution with mean as 30 °C and 70 °C as its 3σ value; power supply voltage was varied with 2.5 V as mean and 0.25 V as 3σ value; and length with mean 12″ and 2″ as 3σ. Eye diagram of the voltage at the end of DIMM D was computed for a 500 sweep Monte Carlo analysis. The

height and width of the eye diagram from Figure 6.10 give an idea of the noise margin and timing jitter the receiver at the end of DIMM D can tolerate. The vertical eye opening is plotted in Figure 6.11 and the horizontal eye opening is plotted in Figure 6.12.

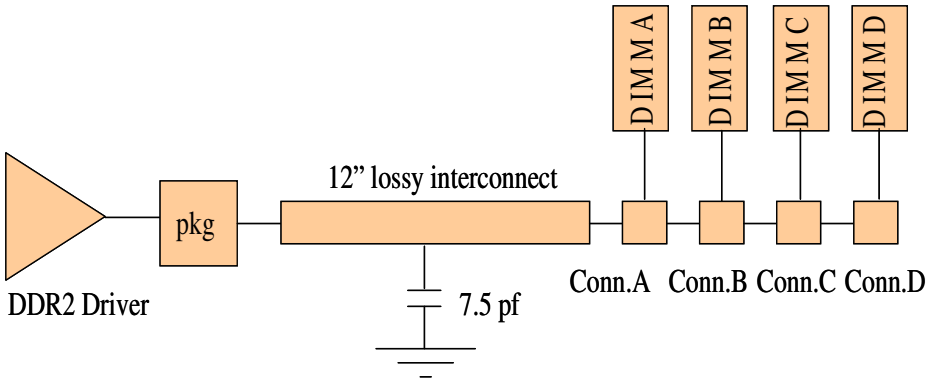


Figure 6.9 Test set-up for DDR2.

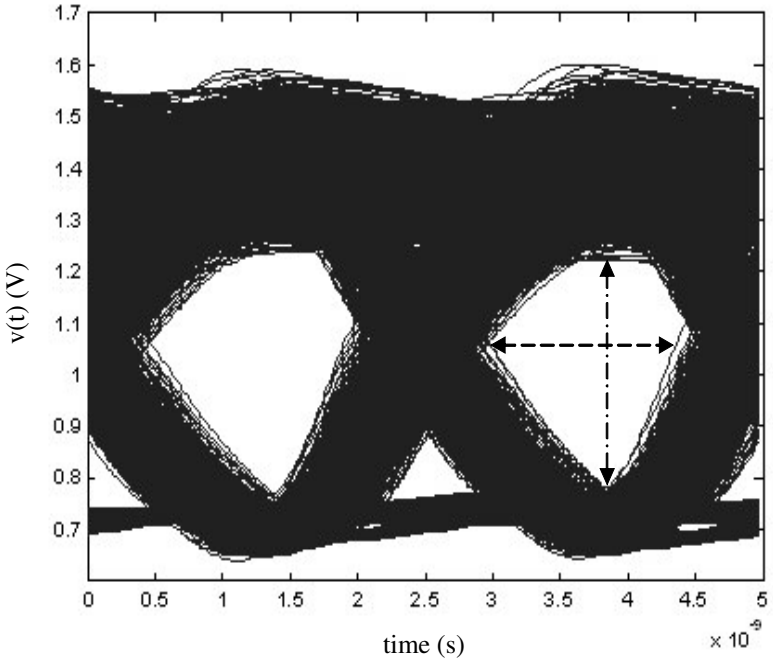


Figure 6.10 Eye diagram at the end of DIMM D.

A threshold on 550 mV was applied for the height and 1.5 ns was applied for the width and it was found that the eye passes the height threshold 85% of the times and the width threshold 95% of the times.

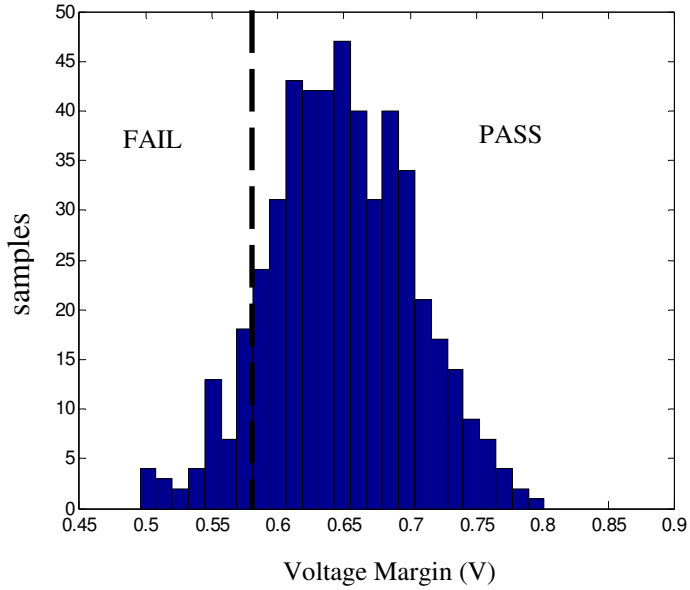


Figure 6.11 Histogram of the height of the eye opening.

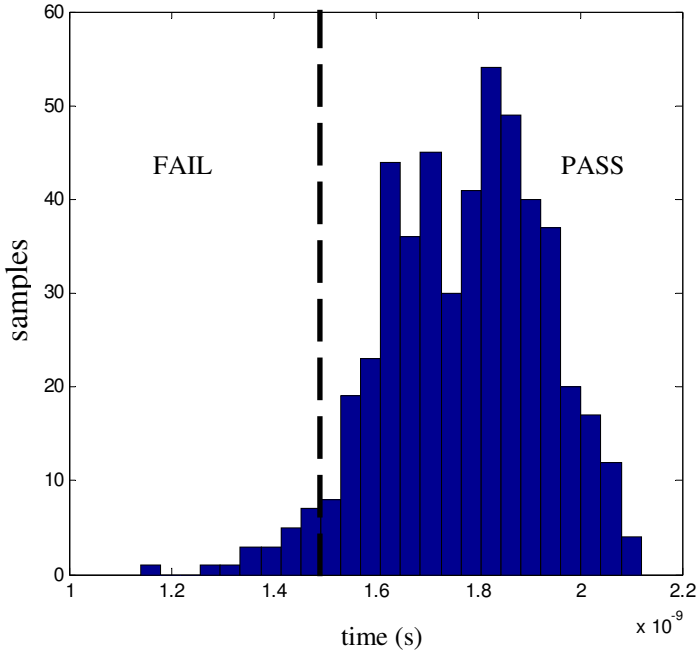
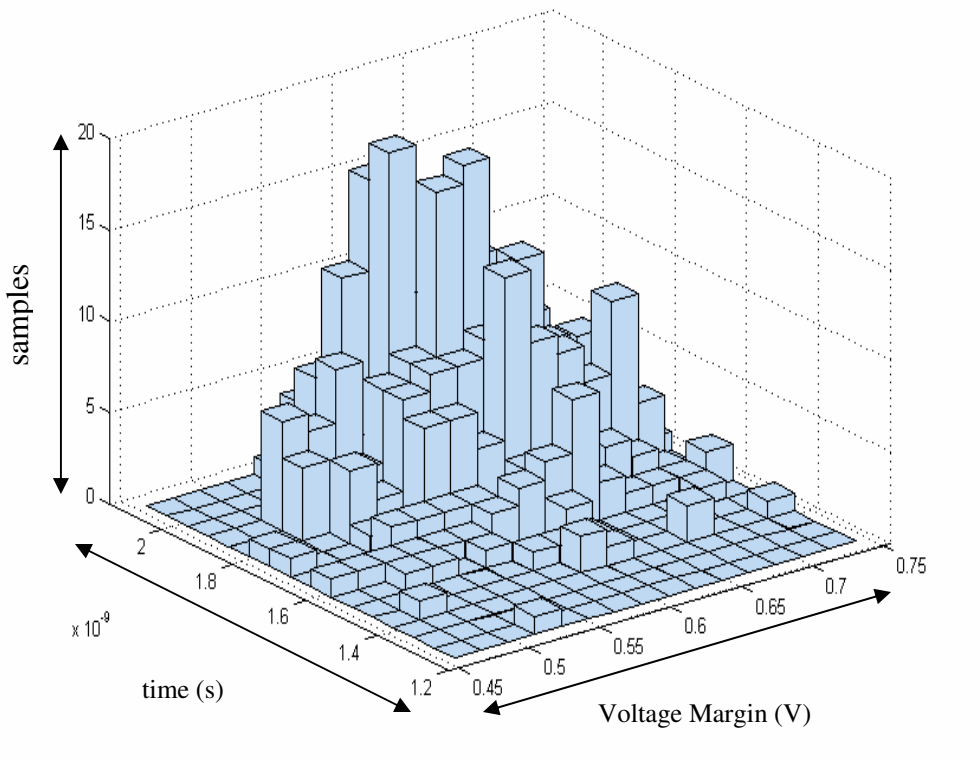


Figure 6.12 Histogram of the width of the eye opening.

Figure 6.13 helps in statistically analyzing the noise and jitter tolerance of a receiver circuit. It has been found that the eye diagram has a yield of 85% in this case. The whole simulation took around 6 hours on an IBM 1.5-GHz server.



6.4 Summary

In this chapter, scalable differential driver and single-ended driver and receiver macromodels have been discussed. It was seen that RNN differential driver macromodels can be accurately scaled with respect to temperature using Lagrange's interpolation technique. Driver and receiver macromodels have been scaled for multiple variables (power supply voltage, process variation, and temperature) using the same technique. Since the effect of these parameters on differential driver and single-ended driver and

receiver circuits has been monotonic, scalable macromodels were generated from the best and the worst case values. The accuracy of the macromodels has been tested on various test-cases both with in and outside the Lagrange's interpolation range and results yielded good correlation between macromodel and transistor-level circuit voltage waveforms. It has been shown that scalable driver circuits help in efficient statistical and yield analysis of today's high-speed digital systems which in turn help in analyzing signal integrity and EMC issues at the pre-layout stage of a design process.

CHAPTER VII

CONCLUSION AND FUTURE WORK

Since macromodeling of digital driver/receiver circuits is a new field, it is a huge challenge to come up with modeling techniques that can accurately model any kind of digital driver/receiver circuit. There are a few modeling methodologies in literature that have been proposed to model driver/receiver circuits but they have their limitations. IBIS models have become industry standard because they are commercially available, have huge design libraries, protect IP, and run faster than transistor-level driver/receiver circuits. IBIS models at the same time have limitations. IBIS models lack accuracy and cannot capture sensitive effects like crosstalk and simultaneous switching noise (SSN) accurately. IBIS models become less accurate when extended to multiple ports. IBIS receiver models do not capture the dynamic characteristics of the receiver circuits accurately and are highly based on the static characteristics. RBF modeling technique is another popular modeling approach to model driver circuits. It is a black-box modeling approach and it can be extended to multiple ports. But RBF models have convergence issues with Hspice when extended to multiple ports. Also, RBF modeling approach cannot accurately model highly nonlinear driver circuits.

The objective of this research work was to a create black-box macromodel for any transistor-level driver or receiver circuit without having any prior knowledge of the

internal logic and circuitry of the driver or receiver circuit. The goal of the dissertation was to generate macromodels of driver and receiver circuits that consume less simulation time and less CPU memory compared to transistor-level driver and receiver circuits. These macromodels at the same time should maintain high accuracy and not have any convergence issues running in SPICE.

6.1 Conclusions

Based on the work presented in chapters 2-6, the contributions of this research can be listed as follows:

1. Driver circuits have been categorized into three categories depending on their complexity. Different macromodels have been proposed based on the complexity of the driver circuits being modeled. Driver circuits where the static characteristics dominate the dynamic characteristics have been categorized as weakly nonlinear driver circuits. Static characteristic macromodels have been proposed to accurately model these driver circuits. Driver circuits where dynamic characteristics are not dominated by static characteristics have been categorized as moderately nonlinear driver circuits. Spline function with finite time difference (SFWFTD) modeling methodology has been proposed to model these driver circuits. In highly nonlinear driver circuits, the dynamic characteristics dominate the static characteristics. A modeling methodology based on recurrent neural networks (RNN) has been proposed to model these driver circuits. It has been found from various test results that all the three macromodeling techniques are black-box approaches. These macromodels are 10-40 X faster than transistor-level driver circuit and are weakly sensitive to the

external load connected to the driver circuits. Pre-compensation and pre-emphasis driver circuits are becoming popular in transmitting signal though lossy interconnects. All the above proposed macromodels can accurately model pre-emphasis driver circuits.

2. All the above proposed modeling approaches have been extended to multiple ports. These macromodels can capture the effect of power supply node and ground port on the output signal. The effect on driver output signal on the power supply node and the ground node has also been captured. These macromodels also accurately estimate sensitive effects like SSN when multiple drivers are switching. A full system level simulation taking the effect of power supply and ground node on the driver output voltage can be performed using the above macromodels.
3. Modern high-speed digital interfaces have turned to low voltage differential signaling (LVDS) because of its numerous advantages over single-ended signaling. A modeling methodology based on RNN network has been proposed to macromodel differential drivers with and without pre-emphasis effect. The macromodels based on RNN networks are 10X faster and consume 10X less CPU memory compared to transistor-level receiver circuits.
4. A modeling methodology to model receiver circuits has been proposed. Input characteristics of the receiver circuit have been modeled by expressing the receiver input current as a function of the receiver input voltage using RNN function or SFWFTD model. The output characteristics of the receiver have been captured by using a combination of voltage transfer characteristics (VTC) of the receiver and a finite time delay element.

5. Receiver circuit modeling approach has been extended to multiple ports. In reality, the receiver input current is not only dependent on the receiver input voltage but also on the receiver power supply voltage. Therefore, the input characteristics of a receiver circuit have been extended to multiple ports by taking the affect of power supply on the receiver input current into account. Similarly, the output characteristics of the receiver circuit have been extended to multiple ports. The receiver output voltage has been expressed as a function of receiver input voltage and receiver power supply voltage.
6. The accuracy of RNN modeling approach has been compared with measurement for a practical working FPGA test case. Transistor-level spice netlists of Altera ('CCT') on-chip FPGA driver and receiver circuits have been calibrated to the laboratory measurement data. RNN macromodels for the driver and receiver circuits have been generated using this transistor-level spice netlist data. Results show good correlation between the measured and modeled voltage waveforms at the receiver input. It has been shown that driver and receiver macromodels match well with measurement results if the transistor-level spice netlists are calibrated well with the measurement results.
7. Since the output voltage and current of a driver circuit is dependent on the power supply voltage, temperature, and process variation, slight variations in the above parameters affect the output voltage and current of the driver circuit. Scalable driver macromodels for driver circuits have been developed that take into account the effect of temperature, power supply voltage, and process variations. Scalable driver models help in analyzing signal integrity issues efficiently at an early design stage. Scalable

macromodels also help in performing statistical and yield analysis for the eye-opening at the receiver input.

6.2 Future Work

As an extension to the modeling methods described in the dissertation, modeling of analog and mixed-signal circuits needs to be investigated.

Mixed-signal circuit modeling: The trend towards the implementation of entire systems on chip and growing markets in mobile communications, process control, and smart sensors has accelerated the mixed signal market. Evolution of deep-submicron CMOS technology has made it possible to integrate more and more analog functionality together with large digital processing systems on single transceiver chips. The exploration and design of mixed-signal systems can be supported with accurate high-level mixed-signal simulation tools. An important building block in mixed-signal systems is an analog-to-digital converter (ADC). Figure 7.1 shows a schematic of a three-bit flash ADC. High-speed ADC models that are supplied by commercial high-level simulation tools often take into account the nominal behavior (e.g. ideal sampling and quantization). As a result, the simulation results are often inaccurate, leading to wrong conclusions/decisions at the system level. Modeling of mixed signal circuits is a challenge because the circuits constituting these circuits are both digital and analog in nature. There is a need for accurate high-level models of analog blocks that can be used in the front-end architecture simulation tool. The difficulty in modeling analog blocks at the system level is that, while the first-order, linear behavior is relatively easily modeled, the nonlinear behavior requires a careful study and even advanced mathematical methods [G1].

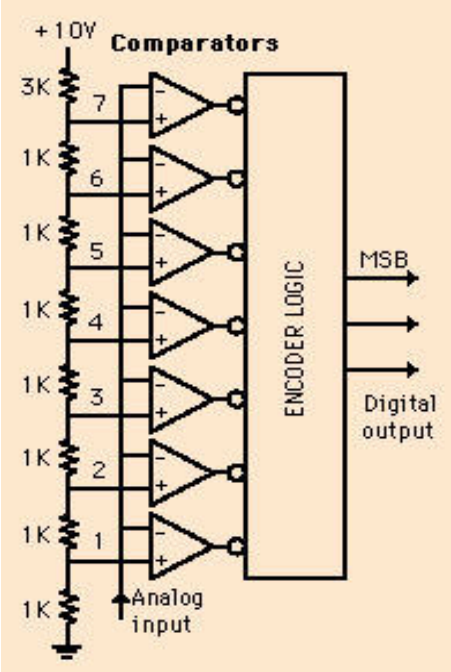


Figure 7.1 Three-bit flash ADC.

An ADC device can be approached as a nonlinear dynamic system to be characterized by means only of input– output data ('black box' behavioral approach). Traditionally, the behavioral modeling of nonlinear dynamic systems found an important theoretical basis in the Volterra series formulation [G2]–[G3]. Many examples are available in literature, which show the application of such an approach to a wide range of fields of interest [G4] – [G6] (telecommunications, microwave circuit design, image processing, robotics, physiology, physics and many others). Nevertheless, the intrinsic properties of the Volterra series formulation have some general limitations as it is not reliable for all practical applications. In presence of high nonlinearities, the number of terms in the Volterra series expansion increase in order to achieve an acceptable system

characterization. One of the future works is to find a new modeling approach to overcoming such limitations and to model ADCs accurately.

RF circuit modeling: The power amplifier (PA) is a major source of nonlinearity in a communication system. To increase their efficiency, PAs are sometimes driven into their nonlinear region, thus causing spectral regrowth (broadening) as well as inband distortion. Behavioral modeling of PAs has been the topic of significant interest over the last years [G7]-[G9]. The power series model, or the polynomial model, is widely used in the literature to describe nonlinear effects in the PA [G10]-[G12]. Most recent models target especially the dynamic behavior of PAs using different linear dynamic and nonlinear static blocks configured in different ways. Normally only the input – output relationship of the sampled complex envelope of the signal is modeled. Hence, the models are valid in a passband centered at the RF frequency. PAs that usually exhibit strong memory effect are difficult to model. Volterra series has been used to model nonlinear PAs with memory effects. The Volterra series and certain special cases of the Volterra series (the Hammerstein model and the memory polynomial model), have been proposed for predistorter design that includes memory effects [G13]-[G15]. However, the use of such series-based models is restricted to weakly nonlinear devices [G16]. Difficulties are often encountered during model identification procedure when power amplifiers are driven by wideband signals since they require high-order kernels that in its turn yield to a high computation complexity. New approaches based on Volterra-Weiner series need to be investigated to address modeling highly nonlinear PAs.

6.3 Publications

Journal Publications:

- **B. Mutnury**, M. Swaminathan and J. Libous, "Macro-Modeling of Non-Linear Digital I/O Drivers," accepted for publication in *IEEE Transactions on Advanced Packaging*, 2005.

- S. Mukherjee, S. Dalmia, **B. Mutnury** and M. Swaminathan, "Layout-level Synthesis of RF Bandpass Filter on Organic Substrates for Wi-Fi Applications," *IEEE Transactions on Microwave Theory and Techniques,* vol. 53, June 2005, pp: 2196-2210.

- **B. Mutnury**, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Macro-Modeling of Non-Linear Transistor-Level Receiver Circuits," accepted for publication in *IEEE Transactions on Advanced Packaging*, 2005.

Conference Publications:

- N. Pham, M. Cases, D. de Araujo, E. Matoglu, **B. Mutnury** and M. Swaminathan, "Design and Modeling Methodology for High-Performance Power Distribution Systems," *DesignCon 2002,* Santa Clara, CA, Jan. 2002.

- E. Matoglu, **B. Mutnury**, N. Pham, M. Cases and M. Swaminathan, "Statistical Modeling of a Multi-Drop Source Synchronous Bus," *Electronic Components & Technology Conference (ECTC),* pp. 62-69, San Diego, CA, May 2002.

- **B. Mutnury** and M. Swaminathan, "Macro-Modeling Methodology for Weakly Non-linear Drivers," *SPI workshop*, Italy, May 2003.

- **B. Mutnury**, M. Swaminathan and J. Libous, "Macro-Modeling of Non-Linear I/O Drivers using Spline Functions and Finite Time Difference Approximation," *12th Topical Meeting on Electrical Performance of Electronic Packaging (EPEP 2003)*, pp. 273-276, Princeton, NJ, Oct. 2003.

- **B. Mutnury**, M. Swaminathan and J. Libous, "Black-Box Modeling of Non-linear I/O Drivers," *The Applied Computational Electromagnetic Society (ACES)*, Syracuse, NY, Apr. 2004.

- **B. Mutnury**, M. Swaminathan and J. Libous, "Modeling of Power Supply Noise using Efficient Macro-Model of Non-Linear Driver," *IEEE International Symposium on EMC,* pp. 988-993, San Jose, CA, Aug. 2004.

- S. Mukherjee, S. Dalmia, **B. Mutnury** and M. Swaminathan, "Layout-level Synthesis of RF Bandpass Filter on Organic Substrates for Wi-Fi Applications," *European Microwave Conference (EUMC 2004)*, pp. 1377-1380, Amsterdam, Oct. 2004.

- **B. Mutnury**, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Macro-Modeling of Transistor Level Receiver Circuits," *13th Topical Meeting on Electrical Performance of Electronic Packaging (EPEP 2004)*, pp. 243-246, Portland, Or, Oct. 2004. **(Intel Best Student Paper Award)**

- S. Mukherjee, **B. Mutnury**, S. Dalmia and M. Swaminathan, "Synthesis and Optimization of Embedded Inductors on Organic Substrate using Knowledge-Based Artificial Neural Networks," *Asia Pacific Microwave Conference (APMC 2004)*, Delhi, Dec. 2004.

- **B. Mutnury**, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Macro-Modeling of Non-Linear Pre-emphasis Differential Driver Circuits," *IEEE International Microwave Symposium (IMS) 2005* , Long Beach, CA, June 2005

- **B. Mutnury**, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Modeling of Power Supply Noise with Non-Linear Drivers using Recurrent Neural Network (RNN) Models," *55th Electronic Components and Technology Conference (ECTC 2005)*, pp. 740-745, Orlando, FL, June 2005.

- M. Roumbakis, **B. Mutnury**, S. Ulrich, J. Ratcliffe, M. Cases and D. N. de Araujo, "Neuro-Genetic models in Modeling Non-Linear Digital I/O Buffer Circuits," *55th Electronic Components and Technology Conference (ECTC 2005)*, pp. 1543-1548, Orlando, FL, June 2005.

APPENDIX A

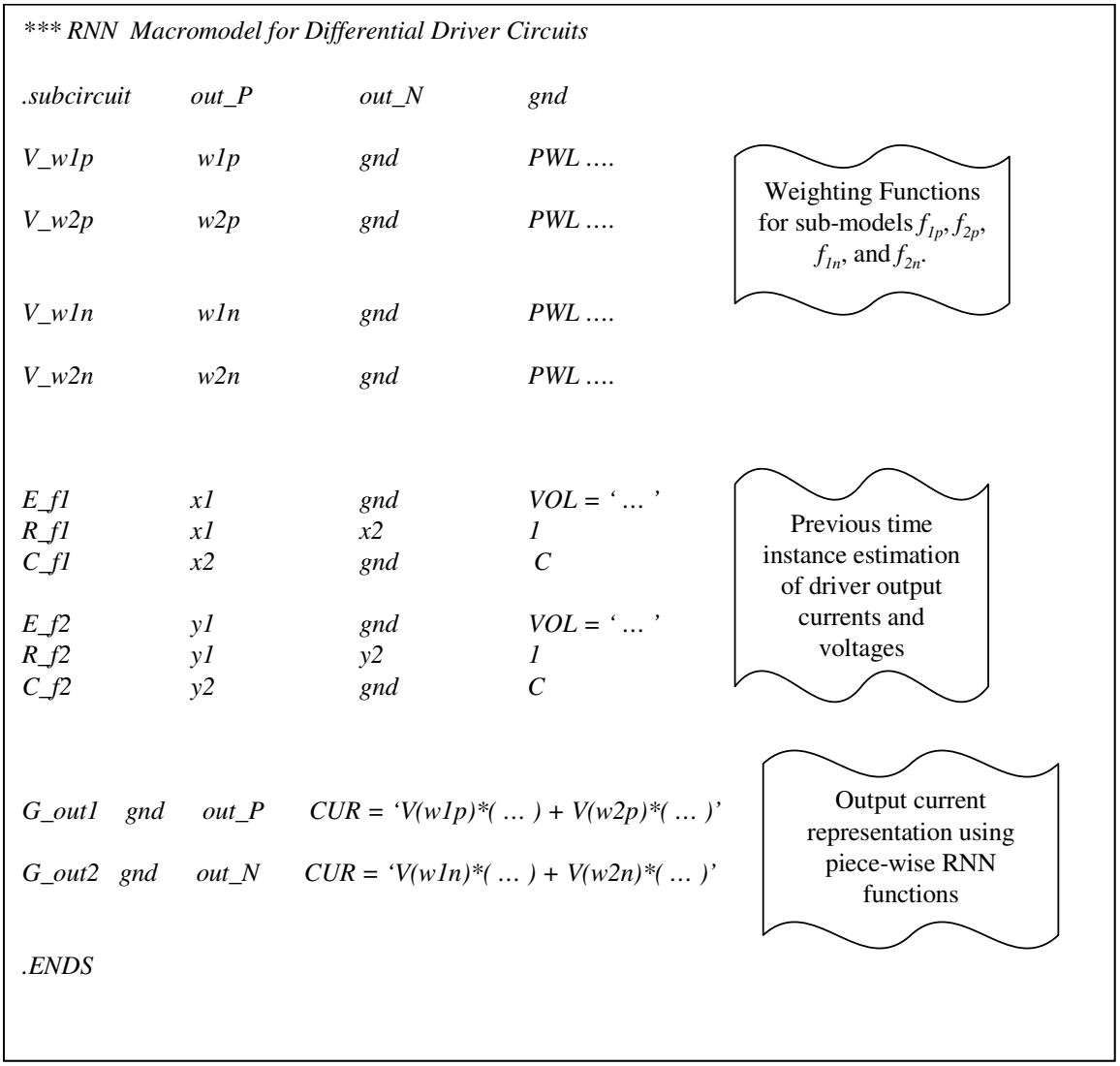


Figure A.1 Spice netlist of a differential driver model.

APPENDIX B

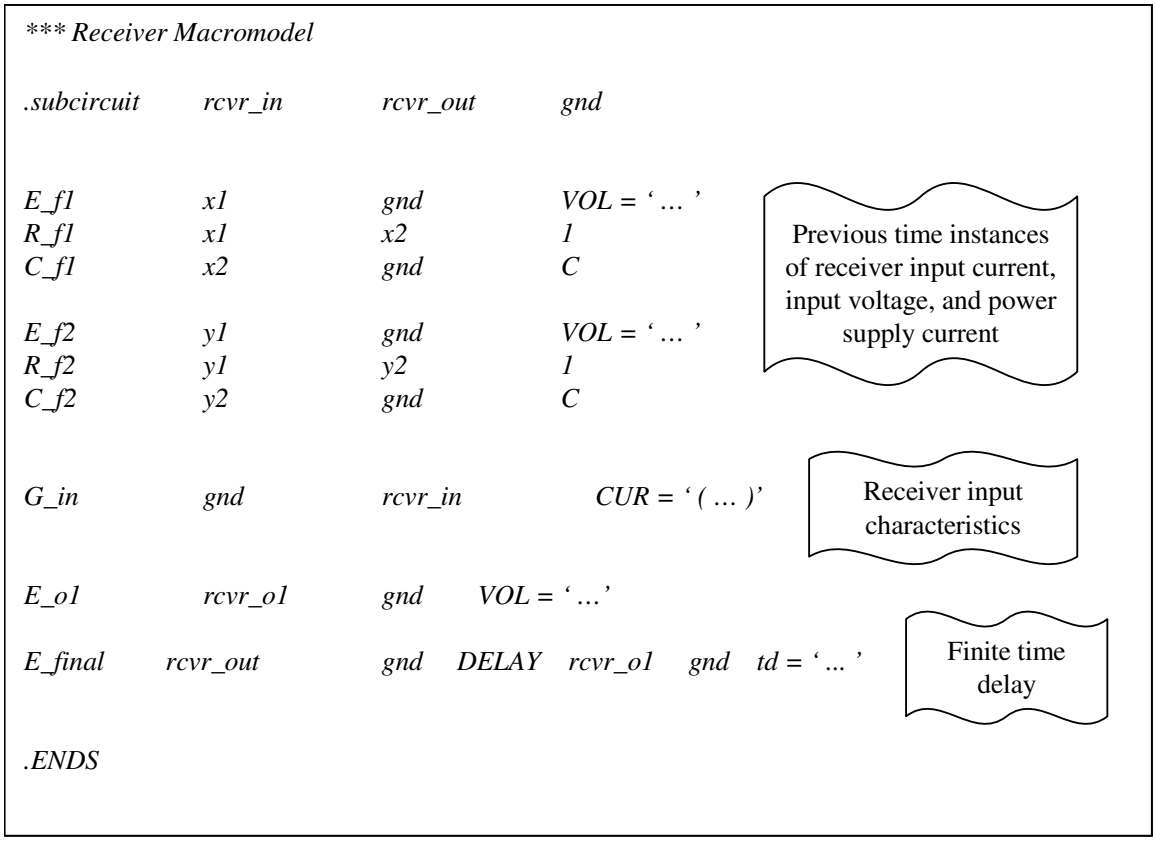


Figure B.1 Schematic of spice netlist for receiver circuits.

REFERENCES

[A1] L. Silveira, M. Kamon, and J. White, "Efficient reduced-order modeling of frequency-dependent coupling inductances associated with 3-D interconnect structures," IEEE Transactions on Components Packaging Manufacturing and Technology A, pt. B, vol. 19, no. 2, pp. 283–288, May 1996.

[A2] K. J. Kerns and A. T. Yang, "Stable and efficient reduction of large, multiport RC networks by poles analysis via congruence transformations," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 16, no.7, pp. 734–744, Jul. 1997.

[A3] L. Pillage and R. Rohrer, "Asymptotic waveform evaluation for timing analysis," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 9, no. 4, pp. 352–366, Apr. 1990.

[A4] E. Chiprout and M. Nakhla, "Analysis of interconnect network using complex frequency hopping (CFH)," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 14, no. 2, pp. 186–200, Feb. 1995.

[A5] P. Feldmann and R. Freund, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 14, no. 5, pp. 639–649, May 1995.

[A6] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling program," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 8, Aug. 1998.

[A7] D. B. Kuznetsov and J. E. Schutt-Aine, "Optimal transient simulation of transmission lines," IEEE Transactions on Circuits and Systems, vol. 43, no. 2, Feb. 1996.

[A8] S. H. Min and M. Swaminathan, "Efficient construction of two-port passive macro-models for resonant networks," IEEE 10th Topical Meeting on Electrical Performance of Electronic Packaging, pp. 229-232, Oct. 2001.

[A9] B. Gustavsen and A. Semlyen, "Simulation of transmission line transients using vector fitting and modal decomposition," IEEE Transactions on Power Delivery, vol. 13, no. 2, Apr. 1998.

[A10] B. Gustavsen and A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions," IEEE Transactions on Power Systems, vol. 16, no. 1, pp. 97-104, Feb. 2001.

[A11] D. Saraswat, R. Achar, and M. Naklar, "A fast algorithm and practical considerations for passive macromodeling of measured/simulated data," IEEE 11th Topical Meeting on Electrical Performance of Electronic Packaging, Oct. 2002.

[A12] J. Roychowdhury, "Reduced-order modeling of time-varying systems," IEEE Transactions on Circuits and Systems. II: Analog Digital Signal Process., vol. 46, no. 10, pp. 1273–1288, Oct. 1999.

[A13] J. Phillips, "Projection frameworks for model reduction of weakly nonlinear systems," in Proc. IEEE/ACM Design Automaton Conf., Jun. 2000, pp. 184–189.

[A14] P. Li, X. Li, Y. Xu, and L. Pileggi, "A hybrid approach to nonlinear macromodel generation for time-varying analog circuits," in Proc. ACM/IEEE Int. Conf. Computer-Aided Design, Nov. 2003, pp. 454–461.

[A15] P. Li and L. T. Pileggi, "Compact Reduced-Order Modeling of Weakly Nonlinear Analog and RF Circuits," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, no. 2, pp. 184-203, Feb. 2005.

[A16] J. Sj¨oberg et al., "Nonlinear black-box modeling in system identification: a unified overview," Automatica, Vol. 31, No. 12, pp. 1691–1724, 1995.

[A17] D. Wiener and J. Spina, Sinusoidal Analysis and Modeling of Weakly Nonlinear Circuits. New York: Reinhold, 1980.

[A18] S. A. Maas, Nonlinear Microwave Circuits. Norwood, MA: Artech House, 1988.

[A19] A. Dharchowdhury, R. Panda, D. Blaauw, R. Vaidyanathan, B. Tutuianu, and D. Bearden, "Design and analysis of power distribution networks in PowerPC microprocessors," in Proc. IEEE Design Automation Conf., Anaheim, CA, Jun. 1998, pp. 738–743.

[A20] M. V. Heijningen, M. Badaroglu, S. Donnay, M. Engels, and I. Bolsens, "High-level simulation of substrate noise generation including power supply noise coupling," in Proc. IEEE Design Automation Conf., Los Angeles, CA, Jun. 2000, pp. 738–743.

[A21] H. H. Chen and D. D. Ling, "Power supply noise analysis methodology for deep-submicron VLSI design," in Proc. IEEE Design Automation Conf., Anaheim, CA, Jun. 1997, pp. 638–643.

[A22] Z. Wang, R. Murgai, and J. Roychowdhury, Member, "ADAMIN: Automated, Accurate Macromodeling of Digital Aggressors for Power and Ground Supply Noise

Prediction," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 24, no. 1, Jan. 2005, pp. 56-64.

[A23] International Technology Roadmap for Semiconductors, http:/public.itrs.net.

[A24] S. Hall, G. Hall and J.A.McCall, High-Speed Digital System Design, John Wiley & Sons, 2000.

[A25] IBIS Open Forum on http://www.eia.org/eig/ibis/ibis.htm, Sep. 1997.

[A26] F. G. Canavero, I. A. Maio, and I. S. Stievano, "Black-Box Models of Digital IC ports for EMC simulations," 14th International Symposium on Electromagnetic Compatibility, Zurich, (Switzerland), pp. 679–684, Feb. 2001.

[A27] "IBIS (I/O Buffer Information Specification)," The IBIS Open Forum, Version 3.2, Aug. 1999.

[A28] R. Cuny "SPICE and IBIS Modeling Kits, The Basis for Signal Integrity Analyses," International Symposium on Electromagnetic Compatibility (EMC), pp.19-23, Santa Clara, Aug. 1996.

[A29] J.D Hayes and L. Wissel, "Behavioral modeling for timing, noise, and signal integrity analysis," IEEE Conference on Custom Integrated Circuits, 2001, pp. 353 - 356, May 2001.

[A30] P.F. Tehrani, C. Yuzbe, and J. Fang, "Extraction of transient behavioral model of digital I/O buffers from IBIS," Proceedings Electronic Components and Technology Conference, pp. 1009 - 1015, May 1996.

[A31] S. Peters, "IBIS Forum U 0 Buffer Modeling Cookbook," The IBIS Open Forum, Revision 2.0X, Sept. 1997.

[A32] T. Zak, M. Ducror, C. Xavier, and M. Drissi, "An Experimental Procedure to Derive Reliable IBIS Models," Proceedings of 3rd Electronics Packaging and Technology Conference (EPTC 2000), Dec. 2000.

[A33] A. Muranyi, "Introduction to IBIS Models," IBIS Model Training, San Jose, California – Apr. 2000.

[A34] A. Varma, A. Glaser, S. Lipa, M. Steer, and P. Franzon, "The development of a macro-modeling tool to develop IBIS models," Electrical Performance of Electronic Packaging, pp. 277 - 280, 27-29 Oct. 2003.

[A35] A. Varma, A. Glaser, S. Lipa, M. Steer, and P. Franzon, "Simultaneous switching noise in IBIS models," International Symposium on Electromagnetic Compatibility, EMC 2004, vol. 3 , pp. 1000-1004, Aug. 2004.

[A36] A. Varma, M. Steer, and P. Franzon, “SSN issues with IBIS models,” 13th Topical Meeting on Electrical Performance of Electronic Packaging, pp. 87 - 90, Oct. 2004.

[A37] http://www.national.com/appinfo/lvds/files/wp_ibis_validate.pdf.

[A38] I. S. Stievano, I. A. Maio, and F. G. Canavero, “Parametric Macromodels of Digital I/O Ports,” IEEE Transactions on Advanced Packaging, pp. 255-264, vol. 25, n. 2, May 2002.

[B1] I. S. Stievano, Z. Chen, D. Becker, F. G. Canavero, S. Grivet-Talocia, G. Katopis, and I.A. Maio, “Modeling of on-board and on-chip interconnected digital devices, ” IEEE International Symposium on Electromagnetic Compatibility (EMC EUROPE 2002), Sorrento, (Italy), pp. 27-32, Sept. 2002.

[B2] I.S. Stievano, Z. Chen, D. Becker, F. Canavero, S. Grivet-Talocia, G. Katopis, and I.A. Maio, “Macromodeling strategy for digital devices and interconnects,” 3rd International Workshop on Electromagnetic Compatibility of Integrated Circuits (EMCCompo2002), Toulouse (France), pp. 53-56, Nov. 2002.

[B3] I. S. Stievano, F. G. Canavero, Z. Chen, G. Katopis, and I. A. Maio, “High Performance RBF Models of Digital Drivers,” 5th Workshop on Signal Propagation on Interconnects (SPI), Cavallino, Venice, (Italy), pp. paper 3.1, May 2001.

[B4] S. Chen, C. F. N. Cowan and P. M. Grant, “Orthogonal least squares learning algorithm for radial basis function network,” IEEE Transactions on Neural Networks, Vol. 2, No. 2, pp. 302–309, Mar. 1991.

[B5] S.Chen, S.A. Billings, and W. Luo, “Orthogonal least squares methods and their applications to non-linear system identification,” International Journal for Controls, vol. 50, no. 5, pp. 1873-1896, 1989.

[B6] B. Mutnury and M. Swaminathan, “ Macro-modeling methodology for weakly non-linear drivers,” SPI workshop, Italy, May 2003.

[B7] B. Mutnury, M. Swaminathan and J. Libous, “Macro-Modeling of Non-Linear I/O Drivers using Spline Functions and Finite Time Difference Approximation,” 12th Tropical Meeting on Electrical Performance of Electronic Packaging (EPEP 2003), Princeton, New Jersey, pp. 273-276, Oct. 2003.

[B8] B. Mutnury, Madhavan Swaminathan and Jim Libous, “Macro-modeling of nonlinear digital I/O drivers (accepted for publication),” IEEE Transactions on Advanced Packaging, to be published.

[B9] J. Aweya, Q. J. Zhang, and D. Y. Montuno, “A direct adaptive neural controller for flow control in computer networks,” in Proc. IEEE International Neural Networks Conference, Anchorage, AK, May 1998, pp. 140–145.

[B10] K. S. Narendra and K. Parthasarathy, “Identification and control of dynamical systems using neural networks,” IEEE Transactions on Neural Networks, vol. 1, pp. 4–27, Jan. 1990.

[B11] A. U. Levin and K. S. Narendra, “Control of nonlinear dynamical systems using neural networks—Part II: Observability, identification, and control,” IEEE Transactions on Neural Networks, vol. 7, pp. 30–42, Jan. 1996.

[B12] Simon Haykin, Neural Networks – A comprehensive foundation, 2nd Edition, Prentice-Hall, 1998.

[B13] Q.J. Zhang, K.C. Gupta, Neural Networks for RF and Microwave Design, Boston: Artech House, c2000.

[B14] P. D. Wasserman, Advanced methods in neural computing, New York: Van Nostrand Reinhold, c1993.

[B15] Q.J. Zhang, K.C. Gupta and V.K. Devabhaktuni, "Artificial neural networks for RF and microwave design: from theory to practice," IEEE Transactions on Microwave Theory Tech., vol. 51, pp. 1339-1350, Mar. 2003.

[B16] B. A. Pearlmutter, “Gradient calculations for dynamic recurrent neural networks: A survey,” IEEE Transactions on Neural Networks, vol. 6, pp. 1212-1228, Sept. 1995.

[B17] V. Devabhaktuni, M.C.E. Yagoub, Y. Fang, J.Xu and Q.J. Zhang, “Neural networks for microwave modeling: model development issues and nonlinear modeling techniques,” International Journal of RF and Microwave CAE, vol. 11, pp. 4-21, 2001 (invited review).

[B18] Y.H. Fang, M. Yagoub, F. Wang and Q.J. Zhang, “A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks,” IEEE MTT International Microwave Symposium (Boston, MA), pp. 883-886, June 2000.

[B19] Y. Fang, M.C.E. Yagoub, F. Wang and Q.J. Zhang, “A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks,” IEEE Transactions on Microwave Theory Techniques, vol. 48, pp. 2335-2344, Dec. 2000.

[B20] http://www.altera.com/products/devices/stratix2/features/density/st2-vir-density-compare.html?f=tchhp&k=g1

[C1] R. Senthinathan and J. L. Prince, Simultaneous Switching Noise of CMOS Devices and Systems, Kluwer Academic Publishers, 1994.

[C2] R. R. Tummala, E. J. Rymaszewski, Alan G. Klopfenstein, Microelectronics Packaging Handbook, Technology Drivers, Part I, Chapter 3, Second Edition, Chapman & Hall, 1997.

[C3] L. Smith, "Simultaneous Switch Noise and Power Plane Bounce for CMOS Technology," IEEE 8th Topical Meeting on Electrical Performance of Electronic Packaging, pp. 163-166, San Diego, CA, Oct. 1999.

[C4] E. E. Davidson, "Electrical design of a high speed computer package," IBM J. Res.Develop., vol. 26, no. 3, pp. 349-361, May 1982.

[C5]R. Dowing, P Gebler, and G. A. Katopis, "Decoupling capacitance effects on switching noise," IEEE Trans. Comp., Hybrids, Manufact. Technol., vol. 16, pp. 484-489, Aug. 1993.

[C6]R. R. Tummala, Microsystems Packaging, New York: McGraw-Hill, 2001.

[C7] B. Mutnury, M. Swaminathan and J. Libous, "Black-Box Modeling of Non-linear I/O Drivers," Proc. 20th Annual Review Progress in Applied Computational Electromagnetics (ACES), Syracuse, NY, Apr. 2004.

[C8] B. Mutnury, M. Swaminathan and J. Libous, "Modeling of Power Supply Noise using Efficient Macro-Model of Non-Linear Driver," 2004 IEEE International Symposium on EMC, San Jose (CA), USA, pp. 988-993, Vol.3, Aug. 2004.

[C9] S. Chun, M. Swaminathan, L. D. Smith, J. Srinivasan, Z. Jin and M. K. Iyer, "Modeling of Simultaneous Switching Noise in High Speed Systems," IEEE Transactions on Advanced Packaging, vol. 24, no. 2, pp. 132-142, May 2001.

[C10] S. Chun, L. Smith, R. Anderson, M. Swaminathan, "Model to Hardware Correlation for Power Distribution Induced I/O Noise in a Functioning Computer System," To be published in Proceeding of 52th Electronic Components and Technology Conference, San Diego, California, May 2002.

[C11] B. Mutnury, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Modeling of Power Supply Noise with Non-Linear Drivers using Recurrent Neural Network (RNN) Models," accepted at 55th Electronic Components and Technology Conference (ECTC 2005), Florida, June, 2005.

[C12] S. Chun, Methodologies for Modeling Simutaneous Switching Noise in Multi-Layered Packages and Boards, Ph.D Thesis, Georgia Institute of Technology, May 2002.

[C13] J. Kim, Modeling of Package and Board Power Distribution Networks Using Transmission Matrix and Macro-modeling Methods, Ph.D Thesis, Georgia Institute of Technology, 2002.

[E1] http://www.national.com/appinfo/lvds/files/wp_ibis_validate.pdf

[E2] I. S. Stievano, Z. Chen, D. Becker, F. G. Canavero, G. Katopis, and I. A. Maio, "Behavioral Modeling of Digital IC Input and Output Ports," 10th Topical Meeting on Electrical Performance of Electronic Packaging (EPEP), Cambridge, Massachusetts (USA), pp. 331-334, Oct. 2001.

[E3] I. S. Stievano, Z. Chen, D. Becker, F. G. Canavero, G. Katopis, I. A. Maio, "Macromodeling of Digital I/O Ports for System EMC Assessment," Design, Automation and Test in Europe (DATE 02), Paris (France), pp. 1044-1048, Mar. 2002.

[E4] B. Mutnury, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Modeling of Power Supply Noise with Non-Linear Drivers using Recurrent Neural Network (RNN) Models," accepted for 55th Electronic Components and Technology Conference (ECTC 2005), FL, June, 2005.

[E5] B. Mutnury, M. Swaminathan, M. Cases, N. Pham, D. N. de Araujo and E. Matoglu, "Macro-Modeling of Transistor Level Receiver Circuits," 13th Topical Meeting on Electrical Performance of Electronic Packaging (EPEP 2004), Portland, OR, October, 2004.

[D1] Howard Johnson and Martin Graham, High-Speed Signal Propagation: Advanced Black Magic, Prentice-Hall, 2003.

[D2] J. Sjoberg et al., "Nonlinear black-box modeling in system identification: a unified overview," Automatica, Vol. 31, No. 12, pp. 1691-1724, 1995.

[D3] "I/O Buffer Information Specification (IBIS) Ver. 3.2," on the web at http://www.eigroup.org/ibis/- ibis.htm, Sep. 1999.

[D4] W. Hobbs, A. Muranyi, R. Rosenbaum and D. Telian, "IBIS: I/O buffer Information Specification, Overview," http://www.vhdl.org., January 14, 1994.

[D5] C. Sporrer, "Quality of IBIS Models - IBIS for LVDS," Design Automation and Test in Europe, DATE 03 – IBIS Summit, March 2003.

[D6] Hazem Hegazy and Mohammed Korany, "LVDS Modeling," Design Automation and Test in Europe, DATE 03 – IBIS Summit, March 2001.

[D7] I.S. Stievano, C. Siviero, I.A. Maio, F. Canavero, "Behavioral Macromodels of Differential Drivers," 8th IEEE Workshop on Signal Propagation on Interconnects (SPI), Heidelberg (Germany), pp. 131-134, May 9-12, 2004.

[D8] I.S. Stievano, C. Siviero, I.A. Maio, F.G. Canavero, "Modeling of the Static and Dynamic Behavior of Differential Drivers," 4th International Workshop on Electromagnetic Compatibility of Integrated Circuits (EMC-Compo), Angers, France, Mar. 31 - Apr. 1, 2004.

[D9] A. Deutsch, "Electrical Characteristics of Interconnections for High Performance Systems," Proceedings of IEEE, Vol 86, No. 2, pp. 315-355, 1996.

[D10] J. Broomall, H. Van Deusen, "Extending the Useful Range of Copper Interconnects for High Data Rate Signal Transmission," Electronic Components and Technology Conference, pp. 196-203, May 1997.

[D11] D. de Araujo, J. Diepenbrock, M. Cases, N. Pham, "Transmitter and Channel Equalization for High-Speed Server Interconnects," Electrical Performance of Electronic Packaging, pp. 221-224, Oct. 2003.

[F1] E. Matoglu, Statistical Design, Analysis and Diagnosis of Digital Systems and Embedded RF Circuits, Ph.D Thesis, Georgia Institute of Technology, Nov. 2003.

[F2] I.S. Stievano, I.A. Maio, F.G. Canavero, "Temperature-dependent macromodels of digital devices," in Proc. IEEE Int. Symp. on Electromagnetic Compatibility (EMC), Zurich, Switzerland, pp. 321-326, Feb. 2003.

[F3] S. G. Duvall, "Statistical circuit modeling and optimization," in Proc. International Statistical Metrology Workshop 2000, June 2000, pp. 56–63.

[F4] J. M. Hammersley and D. C. Handscomb, Monte Carlo Methods. New York: Chapman and Hall, 1964, reprinted, 1983.

[F5] T. B. Barker, Quality by Experimental Design. New York: Marcel Dekker, 1994.

[G1] S. Sengupta and A. Chatterjee, "Fast fault simulation using time convolution," 7th IEEE International Mixed-Signal Testing Workshop, pp. 181–191, Atlanta, 2001.

[G2] M. Schetzen, The Volterra and Wiener Theories of Nonlinear Systems, Wiley, New York, 1980.

[G3] J.S. Bendat, Non-linear System Analysis and Identification from Random Data, Wiley, New York, 1990.

[G4] C.H. Cheng, E.J. Powers, “Fifth-order Volterra kernel estimation for a nonlinear communication channel with PSK and QAM inputs,” 9th IEEE Signal Processing Workshop on Statistical Signal and Array Processing, New York, NY, USA, 1998, pp. 435– 438.

[G5] P. Celka, M.J. Hasler, A. Aziz, “Analysis and linearization of a broadband microwave phase modulator using Volterra system approach,” IEEE Transactions Microwave Theory Techniques, vol. 44, 1996, pp. 2246– 2255.

[G6] H. Kashiwagi, H. Harada, T. Yamaguchi, “A method for measuring nonlinear characteristics of a robot manipulator,” IMEKO ISMCR’99 Topical Workshop on Virtual Reality and Advanced Human– Robot Systems, Budapest, Hungary, pp. 101– 104, 1999.

[G7] P. M. Asbeck, H. Kobayashi, M. Iwamoto, G. Hanington, S. Nam, and L. E. Larsson, “Augmented Behavioral Characterization for Modeling the Nonlinear Response of Power Amplifiers,” presented at MTT-S, 2002.

[G8] H. Ku and J. S. Kenney, “Behavioral Modeling of Nonlinear RF Power Amplifiers Considering Memory Effects,” IEEE Transactions on Microwave Theory Techniques, vol. 51, pp. 2495-2504, 2003.

[G9] S. Boumaiza and F. M. Ghannouchi, “Thermal Memory Effects Modeling and Compensation in RF Power Amplifiers and Predistortion Linearizers,” IEEE Transactions on Microwave Theory Techniques, vol. 51, pp. 2427-2433, 2003.

[G10] S. C. Cripps, Advanced Techniques in RF Power Amplifier Design. Norwood, MA: Artech House, 2002.

[G11] P. B. Kenington, High-Linearity RF Amplifer Design. Norwood, MA: Artech House, 2000.

[G12] G. T. Zhou and J. S. Kenney, “Predicting spectral regrowth of nonlinear power amplifiers,” IEEE Transactions on Communications, vol. 50, pp. 718–722, May 2002.

[G13] C. Eun and E. Powers, “A new volterra predistorter based on the indirect learning architechture,” IEEE Transactions on Signal Processing, vol. 45, pp. 223–227, Jan. 1997.

[G14] L. Ding, R. Raich, and G. T. Zhou, “A Hammerstein linearization design based on the indirect learning architecture,” in Proceedings of IEEE Confernece on Acoustics, Speech, Signal Processing, vol. 3, Orlando, FL, May 2002, pp. 2689–2692.

[G15] L. Ding, G. T. Zhou, D. R. Morgan, Z. Ma, J. S. Kenney, J. Kim, and C. R. Giardina, “A robust digital baseband predistorter constructed using memory polynomials,” IEEE Transactions on Communications , vol. 52, pp. 159–165, Jan. 2004

[G16] R. Raich, H. Qian, G. T. Zhou, **"**Orthogonal polynomials for power amplifier modeling and predistorter design," IEEE Transactions on Vehicular Technology, vol. 53, pp.1468 – 1479, Sept. 2004.

VITA

Bhyrav Mutnury received his B.S. in Electrical Engineering from Jawaharlal Nehru Technological University (JNTU), Hyderabad in 2000. Subsequently, he joined Georgia Institute of Technology for graduate studies and started to work with Dr. Madhavan Swaminathan in the area of signal and power integrity analysis for high-speed digital and mixed-signal system modeling. He received M.S. degree in Electrical and Computer Engineering in 2002. In the summer of 2001 and 2004, he interned at IBM, Austin. His present research interests are in the field of nonlinear driver and receiver modeling for high-speed system level analysis. He has contributed to several publications in the signal and power integrity area. He is the recipient of the Intel best paper award at the Electrical Performance for Electronic Packaging (EPEP) conference in 2004.