CODECS: A MIXED-LEVEL CIRCUIT AND DEVICE SIMULATOR

by

Kartikeya Mayaram

Memorandum No. UCB/ERL M88/71

21 November 1988

CODECS: A MIXED-LEVEL CIRCUIT AND DEVICE SIMULATOR

by

Kartikeya Mayaram

Memorandum No. UCB/ERL M88/71

21 November 1988

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

CODECS: A MIXED-LEVEL CIRCUIT AND DEVICE SIMULATOR

by

Kartikeya Mayaram

Memorandum No. UCB/ERL M88/71

21 November 1988

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

CODECS: A Mixed-Level Circuit and Device Simulator

by

Kartikeya Mayaram

Abstract

CODECS is a mixed-level circuit and device simulator that provides a direct link between technology parameters and circuit performance. Detailed and accurate analyses of semiconductor circuits are possible by use of physical (numerical) models for critical devices. The numerical models are based upon solution of Poisson's equation and the current-continuity equations. Analytical models can be used for the noncritical devices.

The goal of this research has been to develop a general framework for mixed-level circuit and device simulation that supports a wide variety of analyses capabilities and numerical models. Emphasis has been on algorithms to couple the device simulator with the circuit simulator and an evaluation of the convergence properties of the coupled simulator. Different algorithms have been implemented and evaluated in CODECS.

Another aspect of this research has been to investigate critical applications of mixed-level circuit and device simulation. Typical examples include simulation of high-level injection effects in BiCMOS driver circuits, non-quasi-static MOS operation, switch-induced error in MOS switched-capacitor circuits, and inductive turn off of pin rectifiers. For these examples conventional circuit simulation with analytical models gives inaccurate results.

CODECS incorporates SPICE3 for the circuit-simulation capability and for analytical models. A new one- and two-dimensional device simulator has been developed. CODECS supports dc, transient, small-signal ac, and pole-zero analyses of circuits

containing one- and two-dimensional numerical models for diodes and bipolar transistors and two-dimensional numerical models for MOSFETs. The numerical models in CODECS include physical effects such as bandgap narrowing, Shockley-Hall-Read and Auger recombinations, concentration and field-dependent mobilities, concentration-dependent lifetimes, and avalanche generation.

Donald O. Pederson

Committee Chairman

Acknowledgements

This research has been made possible by a grant from the Semiconductor Research Corporation. I am pleased to acknowledge their financial support.

I thank Professor Donald Pederson, my research advisor, for having given me the opportunity to work on interesting projects during my graduate studies at Berkeley. He has been an excellent mentor and I am grateful to him for his advice and guidance, on matters both technical and nontechnical. His continuous support and encouragement have cheered me up when I was in low spirits and inspired me to perform my best. From him I have also learned presentation skills, writing skills, and teaching skills.

Professor Chenming Hu's courses on semiconductor device physics gave me a strong foundation in device physics. He has provided me with excellent guidance during the course of my studies and I thank him for his support.

I have also received valuable guidance from Professor Robert Dutton of Stanford University. He gave me a special treatment (his students tell me that I saw more of him than they did) and made significant suggestions on critical applications of CODECS.

Professors Paul Gray, David Hodges, Ping Ko, Richard Newton, and Alberto Sangiovanni-Vincentelli also gave me valuable advice. I take this opportunity to thank them for teaching state-of-the-art courses and for broadening my horizons.

Dr. Roberto Guerrieri of University of Bologna, Italy, provided insight in device simulation and my discussions with him were extremely helpful.

I am grateful to Dr. James Dyer, now at Polaroid Corporation, for his help and for encouraging me to pursue graduate studies at UC Berkeley. I thank Dr. William McCalla of Hewlett Packard and Dr. Andrei Vladimirescu of Analog Design Tools for their continued interest and support, and Mr. Ted Vucurevich of Analog Devices for

supporting the Lisp-based simulation work. The loan of a Symbolics Lisp machine from Analog Devices and Symbolics Inc. is gratefully acknowledged.

During the course of my research work I have benefited from discussions with Drs. Herman Gummel, Peter Llyod, Jim Prendergast, and Kishore Singhal of AT&T Bell Laboratories.

I have been privileged to work with many talented students during my stay at Berkeley. Jeff Burns and Jack Lee gave me critical advice and support during difficult times. Conversations with Theo Kelessoglou were most beneficial. Cormac Conroy and David Gates gave valuable comments on initial drafts of the thesis. Tom Quarles was most helpful with SPICE3. Without his help CODECS could not be in its present form. Discussions with Ken Kundert were also beneficial. Geert Rosseel of Stanford University was the first outside user of CODECS. I thank him for his patience in using CODECS and for applications of CODECS to BiCMOS drivers. My silent thanks to others whose names I cannot mention because of space constraints.

Most of all, I thank my parents, sisters, and my wife, Namita, for their constant love and support and for their patience during the course of this work. I am grateful to Namita for keeping me in good cheer and for all the sacrifices that she has made for me.

TABLE OF CONTENTS

CHAPTER 1

Introduction

Simulation plays an important role in the design of present day integrated circuits (ICs), devices, and processes. Alternative design techniques can be quickly evaluated and various tradeoffs can be determined before a circuit or device is fabricated. Since fabrication of circuits and devices is time consuming and costly, simulation provides an efficient and attractive way to explore the design space [1.1].

Various types of simulations currently in use, from the process level to the circuit level, are depicted in Figure 1.1. Also shown is the relationship between fabrication and simulation. Two branches are shown, one the experimental branch and the other the computational or simulation branch. Corresponding to each step in the experimental branch there is a simulation step that serves a similar purpose. The eventual goal is to produce circuits that are functionally correct and meet the desired specifications. This is achieved in an iterative manner until the design converges to the specifications. Simulations may not completely eliminate repeated fabrications but they significantly reduce the number of iterations that are required. As an example, there is a thousand-to-one cost savings in simulating a process step compared to laboratory costs [1.1].

A simulator that combines two or more levels of simulation is called a *mixed-level simulator*. The key idea behind a mixed-level simulator is to use detailed forms of simulation on the critical parts of a circuit, to get precise waveform information, and less accurate but faster forms of simulations for the rest of the circuit. There is a definite tradeoff between the accuracy of the simulations and the runtime. The mixed-level circuit

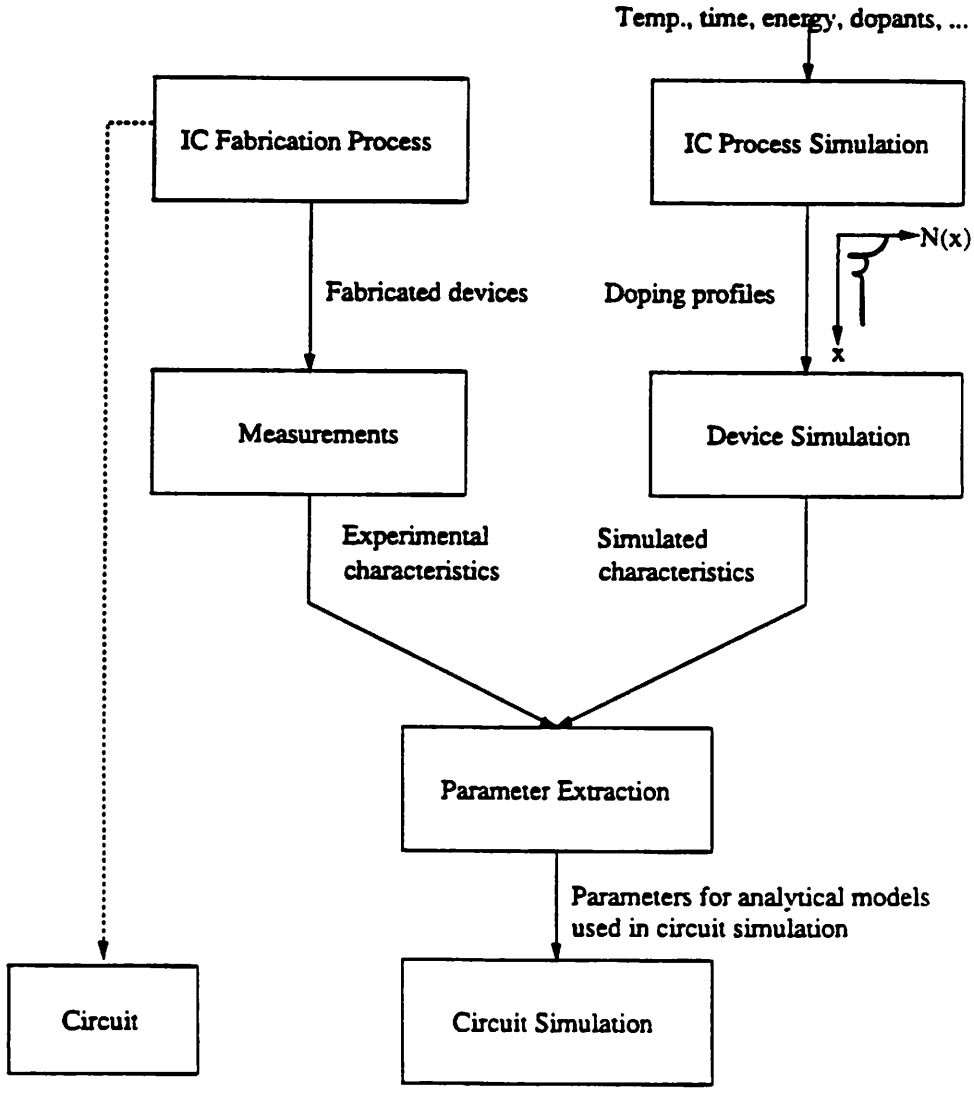


Figure 1.1: Relationship between fabrication and simulation

and device simulator CODECS is described in this dissertation. Some critical applications of mixed-level circuit and device simulation are also presented.

Early work in mixed-level simulation was devoted to combining circuit, timing and logic-level simulations SPLICE1 [1.2], DIANA [1.3]. In later work mixed-level simulation was used to combine circuit and logic simulations in SAMSON [1.4], and circuit, logic and register-transfer-level simulations in SPLICE2 [1.5].

In general, mixed-level simulation has been used extensively to combine the circuit level of simulation with higher levels of simulation. Another direction in which mixed-level simulation can be used is to combine circuit simulation with device simulation as has been done in MEDUSA [1.6] and SIFCOD [1.7]. This approach has several advantages. Devices can be simulated under realistic dc and time-dependent boundary conditions imposed by the circuit in which they are embedded. Conventional device-level simulation typically allows only voltage or current boundary conditions for a device; and, hence, cannot account for circuit embedding. Furthermore, with a mixed-level circuit and device simulator one can simulate circuits even in the absence of analytical models* for semiconductor devices. If the doping profiles and the geometry of the device are available, then circuits can be simulated. This provides a predictive capability at the circuit level, since the impact of process changes and alternate device designs on circuit performance can be determined. Such a simulator provides a direct link between technology and circuit performance. In addition, developers of analytical models for semiconductor devices can use the mixed-level circuit and device simulator as a means for verifying the models for the circuit simulator. Physical effects, that are important and must be incorporated, can be determined and their effect on circuit performance can be evaluated.

* Analytical models refer to mathematical expressions that relate the terminal currents to the terminal voltages of a device.

There is, however, one disadvantage to this approach. Device-level simulation requires the solution of partial-differential equations (Poisson's equation and the current-continuity equations) and is computationally expensive. The tradeoff between accuracy and runtime can be significant. Mixed-level circuit and device simulation is suitable for leaf-cell-based designs, where the leaf cells are made up of a small number of transistors. Alternatively, critical devices can be identified and simulated at the device level with the rest of the circuit being simulated at the circuit level.

MEDUSA provides a general-purpose circuit-simulation capability with a one-dimensional device simulator for bipolar transistors, and an approximate quasi-two-dimensional simulator for MOSFET's. SIFCOD, on the other hand, provides one-dimensional and two-dimensional numerical models but has a very simple circuit-simulation capability. Recently a coupled device and circuit simulator has been reported [1.8] which is based on the two-dimensional device-level simulator PISCES [1.9]. The simulator is a modified version of PISCES that incorporates a circuit-simulation capability. All of these simulators provide only dc and transient simulations of circuits with numerical devices. Although these simulators could be extremely useful to process, device, and circuit designers as well as the device modeling community, they are unfortunately not available in the public domain.

The goal of this research has been to develop a general framework for mixed-level circuit and device simulation that supports a wide variety of analyses and numerical models. A coupled device and circuit simulator, called CODECS, is the result of this research. The simulation environment of CODECS enables one to model critical devices within a circuit by physical (numerical) models based upon the solution of Poisson's equation and the current-continuity equations. Analytical models can be used for the noncritical devices. CODECS supports dc, transient, small-signal ac, and pole/zero analyses of circuits containing one- and two-dimensional numerical models for diodes and

bipolar transistors, and two-dimensional numerical models for MOSFETs. In addition dc and transient sensitivities to doping profiles can be computed at the device level. The numerical models in CODECS include physical effects such as bandgap narrowing, Shockley-Hall-Read and Auger recombinations, concentration and field-dependent mobilities, concentration-dependent lifetimes and avalanche generation.

An effective coupling of the device and circuit-simulation capabilities is achieved by a proper choice of algorithms and architecture. Various algorithms to couple the device-level and circuit-level simulators have been implemented in CODECS. These algorithms are evaluated based on the convergence properties and runtime performance of the coupled simulator. This study also provides guidelines for choice of a particular algorithm.

Another aspect of this research has been to investigate critical applications of mixed-level circuit and device simulation. Typical examples include simulation of high-level injection effects in BiCMOS buffer circuits, non-quasi-static MOS operation, switch-induced error in MOS switched-capacitor circuits, and inductive turn off of pin rectifiers. For these examples conventional circuit simulation with analytical models gives inaccurate results.

This report is organized in the following manner. Chapter 2 provides an overview of circuit simulation and the various techniques used for modeling of semiconductor devices for use in circuit simulators. In Chapter 3 an overview of the physical models used at the device level of simulation is provided. This is followed by a description of the space-discretization techniques and the solution algorithms. The problem of mixed-level circuit and device simulation is addressed in Chapter 4. A framework is proposed for the mixed-level simulator and algorithms to couple the device and circuit simulators are described. The convergence properties of the coupled simulator are also investigated. Chapter 5 describes in detail the algorithms used in CODECS and the techniques used

for enhancing dc convergence. An initial prototype of CODECS was developed in LISP and Chapter 6 provides a speed performance comparison of the LISP-based version with the new version in the C language. A comparison of the tradeoffs involved in use of analytical models is described in Chapter 7 and several applications of CODECS are illustrated in Chapter 8. Finally, the major contributions of this research are summarized in Chapter 9 and possible directions for future research are also listed.

CHAPTER 2

An Overview of Circuit Simulation and Semiconductor Device Modeling

2.1. Introduction

Circuit-level simulation is one major component of a mixed-level circuit and device simulator. This chapter provides an overview of the circuit-simulation problem. First, the nonlinear dc and transient analyses are introduced and the solution techniques currently in use, namely, the direct and relaxation-based methods are described. This is followed by a description of the small-signal ac and pole-zero analyses.

Modeling of semiconductor devices plays an important role in circuit simulation. The simulation results are reliable only if accurate models are used for the devices. Accuracy of a model is related to the application. For some simulations, such as those of digital circuits, precise models are not necessary. On the other hand, simulation of analog and other high-performance circuits mandates the use of "exact" models. A model that is suited to the simulation of digital circuits may be inappropriate for simulating analog circuits. There is a tradeoff between the accuracy of a model and the simulation runtime. Therefore, different models are used depending on the speed and accuracy requirements of a simulation. Commonly used approaches to modeling of semiconductor devices can be classified as: analytical models, empirical/table models, quasi-numerical models, and numerical models.

The modeling task involves use of a particular technique to model accurately the static (dc) and dynamic (transient) operation of the device. First, a model is developed for the dc operation. Then, the dc model is enhanced for use in transient analysis by incorporating charge-storage effects.

Analytical models are by far the most popular and are addressed in detail in this chapter. These models relate the terminal currents and voltages of a device by closed-form expressions. An understanding of the device physics provides a first-order relationship between the terminal currents and voltages. Second-order physical effects are incorporated by empirical factors. Nonetheless, several deficiencies exist in analytical models and these are also described.

A brief description of the other modeling techniques is also provided along with their advantages and limitations.

2.2. The Circuit-Simulation Problem

Circuit simulation consists of two distinct phases, an equation-assembly phase followed by an equation-solution phase. Circuit-level equations can be assembled by the combined use of the Kirchoff's current and voltage laws (KCL and KVL) along with the branch-constitutive relations for each element in the circuit [2.1]. Such an approach to formulating the circuit equations is called the Sparse-Tableau Approach (STA) and is used in the simulation program ASTAP [2.2]. However, the number of equations describing the circuit is large; $n + 2b$ equations for a circuit with $n + 1$ nodes and b branches.

Nodal Analysis (NA) [2.1] requires only n equations, for a circuit with $n + 1$ nodes. However, NA is restrictive in that it allows only elements which are voltage controlled, i.e., elements for which the terminal currents can be expressed as a function of the terminal voltages. Modified Nodal Analysis (MNA), first used by [2.3] and later

formalized by [2.4], accommodates *nonnodal* elements and for this reason is preferred and has been used in SPICE [2.3, 2.5]. MNA allows all types of circuit elements and results in a system of equations that, although larger in size compared to NA, is still considerably smaller than STA.

2.2.1. Dc and Transient Analyses

Dc and transient analyses are presented in this section. The transient simulation problem is introduced and the dc operating point analysis is shown to be a special case of the general transient problem. The dc operating point solution provides the initial condition for transient analysis and is computed before starting the transient simulation.

Transient simulations are by far the most frequently used analyses in circuit design. The dynamic response of a circuit is described by a system of nonlinear differential-algebraic equations obtained from the equation-assembly phase. These equations can be written in a general form as

$$\mathbf{f}(\dot{\mathbf{z}}(t), \mathbf{x}(t), \mathbf{u}(t)) = \mathbf{O}$$

$$\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t)) \tag{2.1}$$

where $\mathbf{x}$ is the vector of unknowns, i.e., the node voltages and the currents through nonnodal elements, $\mathbf{u}$ is the excitation vector, $\mathbf{z}$ is the vector of capacitor charges and inductor fluxes, $\mathbf{f}$ is a nonlinear vector-valued function obtained from an MNA formulation of the circuit equations, and $\mathbf{g}$ is a nonlinear vector-valued function that relates the capacitor charges and inductor fluxes to the capacitor voltages and inductor currents, respectively. The initial conditions are obtained by setting $\dot{\mathbf{z}}(0) = 0$ and solving the nonlinear algebraic equations

$$\mathbf{f}(\mathbf{O}, \mathbf{x}(0), \mathbf{u}(0)) = \mathbf{O} \tag{2.2}$$

The solution vector $\mathbf{x}_0 = \mathbf{x}(0)$ is called the dc operating point solution of the circuit. The

capacitor charges and inductor fluxes at time $t = 0$ are given by the algebraic relationship

$$\mathbf{z}(0) = \mathbf{g}(\mathbf{x}_0) \tag{2.3}$$

For a dc operating point analysis the nonlinear Equation (2.2) is solved. This is done by an iterative approach and the Newton-Raphson method is frequently used since it has local quadratic convergence [2.3, 2.6]. Use of Newton's method results in the solution of a linear system of equations for each iteration. This system of equations is given by

$$\mathbf{J}(\mathbf{x}^k)\Delta\mathbf{x}^{k+1} = -\mathbf{f}(\mathbf{x}^k) \tag{2.4}$$

where $\mathbf{J}(\mathbf{x}^k) = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right]^k$ is the Jacobian matrix of the circuit-level equations, and k is the iteration number. Equation (2.4) is solved at each iteration for $\Delta\mathbf{x}^{k+1}$, which is used to calculate the new iterate, $\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}^{k+1}$. This is done until convergence is reached ($\| \Delta\mathbf{x}^{k+1} \| < \varepsilon$, where ε is a user-specified error tolerance). Alternatively, Equation (2.4) can be written as

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \left[\mathbf{J}(\mathbf{x}^k)\right]^{-1}\mathbf{f}(\mathbf{x}^k) \tag{2.5}$$

This system of equations is obtained from the companion circuit [2.7] of the circuit under consideration. The companion circuit is a linear circuit obtained from the original nonlinear circuit by replacing all nonlinear elements by their linear companion models [2.7]. SPICE makes use of the companion circuit and the equations are assembled in the form of Equation (2.5). The linear equations are solved by Gaussian elimination or LU decomposition. In general, the system of circuit equations is very sparse; hence, sparse-matrix techniques are used [2.6, 2.8].

For transient analysis, the equations can be solved in two different ways, the standard approach being the *direct method* and the other techniques being *relaxation-based methods*. The flow-chart for the direct method is shown in Figure 2.1. The simulation interval is divided into time points, and Equation (2.1) is solved at each of these timepoints. At timepoint t_{n+1} Equation (2.1) can be expressed as

$$\mathbf{f}(\dot{\mathbf{z}}_{n+1}, \mathbf{x}_{n+1}, \mathbf{u}_{n+1}) = \mathbf{O}$$

$$\mathbf{z}_{n+1} = \mathbf{g}(\mathbf{x}_{n+1}) \tag{2.6}$$

where the computed values $\mathbf{x}_{n+1}$ are used. These are the approximate values of the exact solution $\mathbf{x}(t_{n+1})$ that are obtained from the numerical solution. The above equation can be solved for the unknowns $\mathbf{x}_{n+1}$, once $\dot{\mathbf{z}}_{n+1}$ is expressed in terms of the values of the unknowns $\mathbf{x}$ at the present and previous timepoints. This is done by use of an integration formula where all time derivatives are replaced by discretized approximations [2.3, 2.6, 2.8]. This step is know as time discretization. For the backward-differentiation formulae (BDF) [2.9] of order k, $1 \le k \le 6$, $\dot{\mathbf{z}}_{n+1}$ is given by

$$\dot{\mathbf{z}}_{n+1} = \frac{1}{h_n} \sum_{i=0}^{i=k} \mathbf{z}_{n+1-i} = \frac{1}{h_n} \sum_{i=0}^{i=k} \mathbf{g}(\mathbf{x}_{n+1-i}) \tag{2.7}$$

Use of $\dot{\mathbf{z}}_{n+1}$ from Equation (2.7) in Equation (2.6), results in a system of nonlinear algebraic equations that can be expressed in a general form as,

$$\mathbf{F}(\mathbf{x}_{n+1}) = \mathbf{O} \tag{2.8}$$

The solution of Equation (2.8) provides the solution at timepoint t_{n+1}. The nonlinear equations are solved in a manner similar to that used for the dc operating point analysis. Newton's method is used whereby linear equations are assembled for the companion circuit at each iteration and these are then solved by sparse Gaussian-elimination or LU-decomposition techniques. Once convergence is achieved, the solution at the present timepoint is available. A new timestep is then selected as described below, and the

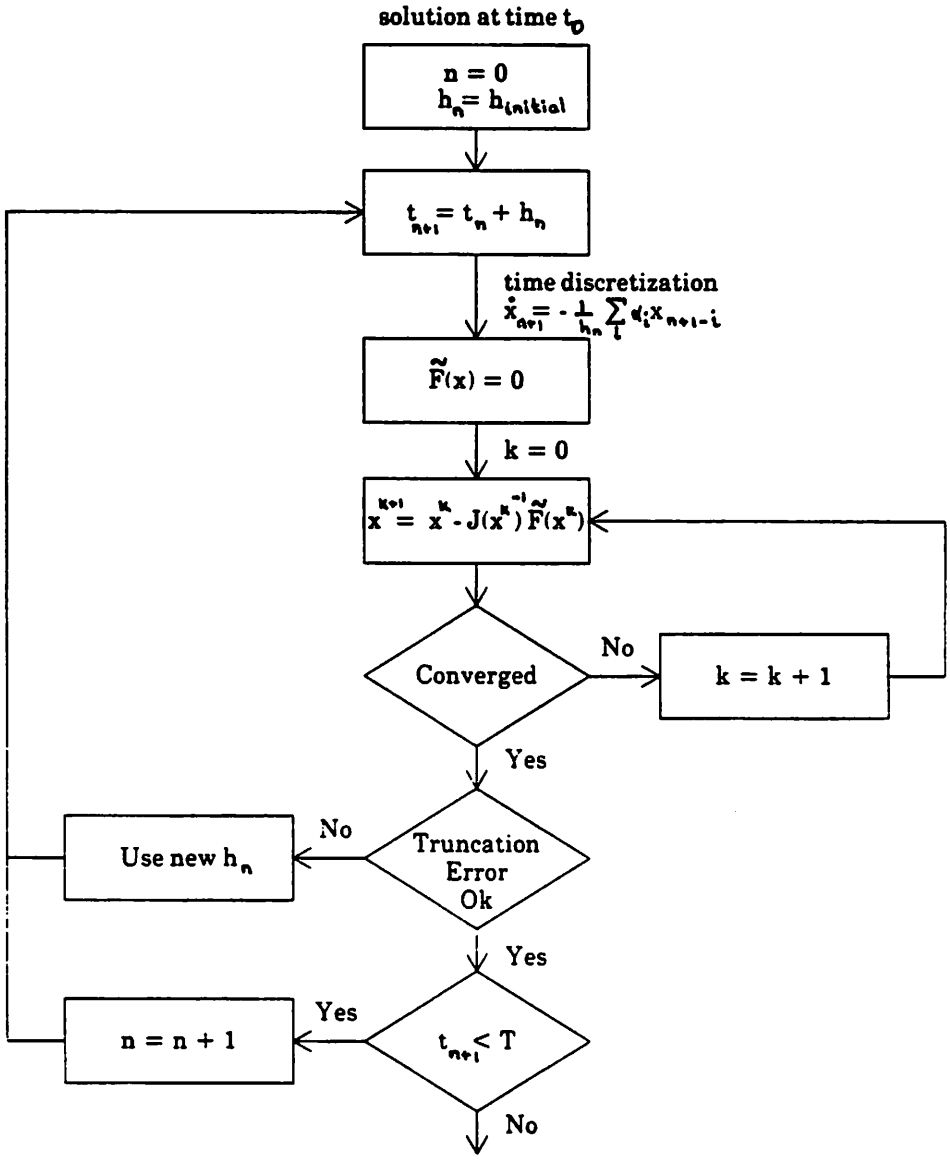


Figure 2.1: Flowchart for transient analysis

equations are solved at the new timepoint in a similar manner. The timestep is selected such that error in time discretization due to the integration formula is less than a user-specified error tolerance [2.3, 2.9].

The relaxation-based methods differ from the direct method in that all equations are not solved simultaneously; rather they are solved in a decoupled manner. Thus Gaussian elimination or LU decomposition is not required, and a substantial improvement in run-time performance can be achieved, particularly for large circuits where a large system of equations has to be solved. These methods work extremely well when there is no feedback or strong coupling between nodes; the equations are effectively decoupled. With feedback some of the equations are coupled; hence, a relaxation approach may converge very slowly, if at all. Two relaxation-based methods that have been successfully used are *Iterated Timing Analysis* (ITA) [2.10] and *Waveform Relaxation* (WR) [2.11, 2.12]. These methods differ in the manner in which the equations are decoupled.

In ITA, the nonlinear differential-algebraic equations are discretized in time as in the direct method. The nonlinear algebraic equations at each timepoint are then solved by use of a Gauss-Seidel or a Gauss-Jacobi approach [2.10] instead of a direct solution. The relaxation is at the nonlinear algebraic-equation level. For a Gauss-Jacobi algorithm, the n-th equation is solved for the n-th unknown, with an estimate for the values of all the other unknowns at the first iteration. At other iterations the values from the previous iteration are used. This process is repeated until convergence is reached on the whole system of equations.

In general, each equation is a nonlinear equation, and the nonlinear equations have to be solved by use of an iterative method. In ITA Newton's method is used. However, the equations are not solved to convergence, instead only one iteration of Newton's method is used. It can be shown that both the Gauss-Jacobi-Newton and Gauss-Seidel-Newton loops have linear convergence [2.10]. Once the solution has been obtained at a

timepoint, the next timepoint is selected based on an error-control criteria, and the whole process is repeated. It can be shown that if there is a grounded capacitor at each node, ITA will converge. This assumption is quite realistic for many digital-MOS circuits. With ITA the multirate and latency present in the circuit can also be exploited using an event-driven simulation technique [2.10].

With WR the relaxation is applied at the nonlinear differential-equation level [2.11]. The unknowns are the node-voltage waveforms. For each relaxation iteration, new waveforms are computed for the node voltages. The differential equations can be decoupled by use of a Gauss-Seidel or a Gauss-Jacobi method [2.12]. For the Gauss-Jacobi algorithm, the n-th differential equation is solved for the voltage waveform at node n, with an estimate for the voltage waveforms at the other nodes of the circuit at the first iteration. For subsequent iterations, the waveforms from the previous iteration are used. Convergence is checked by comparing waveforms from the previous iteration to the waveforms at the present iteration, for each node. As with ITA, convergence is guaranteed if there exists a grounded capacitor at each node [2.11].

As mentioned earlier, relaxation-based methods provide no advantage in simulating tightly coupled circuits. They work well for many digital-MOS circuits where the signal flow is essentially in one direction and there is no feedback. Whenever feedback exists, some of the nodes are strongly coupled to one another and the equations must be solved simultaneously as in the direct method. Thus, another approach combines the advantages of the direct method with that of the relaxation-based approaches. Partitioning algorithms [2.12, 2.13] are used to identify the strongly coupled blocks within a circuit. Direct methods are used on these subcircuits, and a relaxation-based method is used between the subcircuits. In this manner the relaxation method is used only on subcircuits which are decoupled and works extremely well. This combination of direct and relaxation-based methods takes advantage of both solution techniques.

2.2.2. Small-Signal Ac Analysis

Small-signal ac analysis is of interest in the simulation of analog circuits. It involves determining the response of a circuit to a "small" sinusoidal excitation after a dc operating point has been established. The magnitude of the excitation is such that the operation of the nonlinear circuit is linear around the operating point and no significant harmonics are generated.

Consider the circuit equations given by Equation (2.1). Assume the dc operating point is given by $\mathbf{x}_0$ and $\mathbf{u}_0$. A small perturbation is assumed in the excitation vector and the new excitation, represented as a phasor, is given by $\mathbf{u} = \mathbf{u}_0 + \tilde{\mathbf{u}}e^{j\omega t}$, where $\tilde{\mathbf{u}}$ is a complex quantity and ω is the frequency of the sinusoidal sources. All input sources are assumed to be of the same frequency. In response to this perturbation the solution vector $\mathbf{x}$ is given by $\mathbf{x} = \mathbf{x}_0 + \tilde{\mathbf{x}}e^{j\omega t}$ such that the circuit constraints (KCL and KVL) are still satisfied. The small-signal ac response is obtained by a truncated Taylor series expansion of Equation (2.1) around the operating point; the linear terms are retained, and

$$\mathbf{f}(\dot{\mathbf{z}}, \mathbf{x}, \mathbf{u}) = \mathbf{f}(0, \mathbf{x}_0, \mathbf{u}_0) + \frac{\partial \mathbf{f}}{\partial \dot{\mathbf{z}}}\dot{\mathbf{z}} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x} - \mathbf{x}_0) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{u} - \mathbf{u}_0) \tag{2.9}$$

In Equation (2.9), $\mathbf{f}(0, \mathbf{x}_0, \mathbf{u}_0)$ is zero since $\mathbf{x}_0$ is the dc operating point solution. The left-hand-side is also zero, since the perturbed response does not violate the circuit constraints. Therefore, the linearized equations are given by

$$\frac{\partial \mathbf{f}}{\partial \dot{\mathbf{z}}}\mathbf{g}'(\mathbf{x}_0) j\omega\tilde{\mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\tilde{\mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\tilde{\mathbf{u}} = \mathbf{O} \tag{2.10}$$

From Equation (2.10) the small-signal ac solution $\tilde{\mathbf{x}}$ is given by

$$\left[j\omega\frac{\partial \mathbf{f}}{\partial \dot{\mathbf{z}}}\mathbf{g}'(\mathbf{x}_0) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right]\tilde{\mathbf{x}} = -\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\tilde{\mathbf{u}} \tag{2.11}$$

The above equations are a function of the frequency, ω; hence, the solution $\tilde{\mathbf{x}}$ is also a function of the same frequency. At a particular frequency, the linearized equations are assembled with MNA or NA as in dc or transient analyses. All nonlinear devices are represented by linearized admittances which are used for assembling the equations. The complex linear system of equations is then solved by sparse Gaussian-elimination or LU-decomposition techniques. With different values of ω the solution $\tilde{\mathbf{x}}$ can be obtained over a range of frequencies.

2.2.3. Pole-zero Analysis

Pole-zero analysis is useful when a designer is interested in determining the poles and zeros of a transfer function, i.e., the natural frequencies of the circuit and the transmission zeros of the transfer function. The transfer function, $T(s)$, of a circuit linearized at an operating point can be represented as,

$$T(s) = \frac{N(s)}{D(s)} \tag{2.12}$$

where $s = \sigma + j\omega$ is the complex frequency. The zeros of $N(s)$ are the zeros of $T(s)$ and the zeros of $D(s)$ are the poles of $T(s)$. Therefore, the poles and zeros of $T(s)$ can be obtained from the zeros of the polynomials $N(s)$ and $D(s)$. These polynomials are determinants of admittance matrices; the determinant can be obtained by decomposing the matrix in LU factors and taking the product of the diagonal terms.

The zeros of the polynomials are found by an iterative root-finding method such as Muller's method [2.8]. Since the number of roots are not known *a priori*, a termination procedure [2.8] must also be used to terminate the root-finding algorithm. Alternatively, other techniques can be used [2.14].

2.3. Semiconductor Device Modeling for Circuit Simulation

Regardless of the circuit-level analysis and the solution methods used to solve the circuit-level equations, branch-constitutive relations are required that model the semiconductor devices. Device models describe the physical operation of a device by providing a relationship between the terminal currents and the terminal voltages. Once this relationship is known, the companion circuit for the nonlinear semiconductor device can be obtained given the terminal voltages.

Several approaches are currently in use for modeling of the semiconductor devices. These can be classified under four major categories: analytical, empirical/table, semi-numerical, and numerical. There is always a tradeoff between the accuracy obtained with a model and the computation time required to calculate the equivalent conductances and currents, to be used in the companion circuit, for given terminal voltages. A simulation is only as good as the models used for the semiconductor devices. An inaccurate model results in erroneous simulation results. Therefore, modeling is very critical for circuit simulation and this section addresses the different approaches used in modeling. Various approaches to modeling are described along with their advantages and disadvantages.

2.3.1. Analytical Models

Analytical models are generally derived from an understanding of the physics of device operation under some restricted conditions such that closed-form expressions can be obtained for the terminal characteristics. Typically, a piecewise-uniform doping is assumed within the device, and use is made of the drift-diffusion equations to obtain the current-voltage characteristics for the semiconductor devices under dc conditions. Empirical parameters are introduced to model higher-order physical effects. Some simple examples of this approach are the Ebers-Moll model [2.15] for the bipolar transistor and the Shichman-Hodges model [2.16] for MOSFETs. These models are often inadequate

for modeling present day integrated devices; hence, there is a significant research effort for obtaining better models that account for all the important physical phenomena within the device. The Gummel-Poon model [2.15, 2.17] has been the workhorse for simulation of bipolar integrated circuits and has proved to be extremely useful. However, present day bipolar devices cannot be adequately modeled by this model and once again there is need for a better model to replace the Gummel-Poon model [2.18, 2.19]. For MOSFETs a unified model has not emerged [2.20-2.31] and even today new models are constantly being developed [2.32-2.36].

A model that is used for a particular device is characterized by a set of parameters called the model parameters. These parameters are the various constants that appear in the closed-form expressions relating the terminal currents to the terminal voltages. Their values are determined from the measured characteristics of a device. Typically, parameter extraction relies on curve fitting such that the data from the analytical model is in reasonable agreement with the measured data. Since the number of parameters to be determined is usually quite large, parameter extraction makes use of optimization techniques such as those used in TECAP [2.37] or SUXES [2.38].

Optimization is used to minimize the error between the simulated and measured current-voltage characteristics and the model parameters are chosen to minimize this error. Even though the simulated current-voltage characteristics may be in good agreement with experimental data, the values of the terminal conductances could be inaccurate. Therefore, the optimization procedure should also minimize the error of the simulated and measured conductance values [2.39]. In the parameter extraction phase several model parameters are used as curve-fitting parameters and loose their physical significance. Although the model was originally based on the physics of device operation, the parameters are not. Therefore, any correlation that existed between the process parameters, such as the doping levels, may disappear, and the model cannot be used to

predict the effect of process variations on circuit performance.

Analytical models are based on the physics and the structure of the device; hence, a new model is required for each device type and structure. The model parameters can only be determined after actual devices have been fabricated, and their characteristics have been measured. In fact, the model may have to be modified significantly to obtain a good agreement with measured data, if some physical effect that was not incorporated in the model is important in the fabricated devices, e.g., velocity saturation in short-channel MOSFETs [2.40]. Good analytical models, therefore, become available only after an IC fabrication process is in production. Circuit designers are often faced with the problem of designing and simulating circuits without a good set of model parameters; therefore, the circuit design may be quite conservative and it is difficult to push a technology to its limits. Furthermore, even if the device structure does not change significantly, the operation of the device may be altered considerably, e.g., a lightly doped drain MOSFET [2.41] operates differently from a conventional MOSFET [2.42, 2.43]. In this case a new analytical model is required for the modified structure, which again necessitates fabrication of devices and determination of the experimental characteristics. Analytical models do not possess a predictive capability, and cannot be used to evaluate the impact of process changes on circuit performance.

Once the dc model has been obtained for the device it is extended to account for the dynamic or transient operation. This requires identifying the depletion regions and the charge-storage regions within the device and modeling them as capacitors attached between the various internal and external terminals of the device. The capacitance models are derived under the assumption that the terminal voltages vary slowly during transient analysis, i.e., quasi-static operation is assumed. Therefore, expressions for the dc charges can be used to model the incremental capacitances. Examples of such a modeling approach are the MOS-capacitance models [2.44] and the depletion-region

capacitance of pn junctions.

The capacitance models are based on the concept of incremental-charge partitioning [2.45]. The incremental capacitances are defined by

$$C_{ij} = \frac{\Delta Q_i}{\Delta V_j} \tag{2.13}$$

where ΔQ_i is an incremental charge associated with terminal i due to a change in voltage ΔV_j at terminal j. Physical insight into device operation is necessary to determine the charge partitioning.

As an example, consider the depletion-region capacitance of a pn-junction obtained by use of incremental-charge partitioning. This capacitance is given by

$$C_j(V) = \frac{C_{j0}}{\left[1 - \frac{V}{V_{bi}}\right]^m} \tag{2.14}$$

where C_{j0} is the zero-bias junction capacitance, V is the forward voltage across the junction, and V_{bi} is the built-in potential of the pn junction and m is the junction-grading coefficient. Under reverse-bias conditions this capacitance model is adequate. For forward-bias conditions the depletion region decreases and the capacitance increases. The capacitance model of Equation (2.14) suggests that the depletion-region capacitance becomes very large, as shown in Figure 2.2, when the junction is biased such that operation is in high-level-injection conditions, i.e., V is close to V_{bi}. This is incorrect as has been demonstrated in [2.46, 2.47]. The capacitance model fails to predict the correct trend since the incremental charge cannot be uniquely associated with the contacts [2.45]. Under reverse-bias conditions, the incremental charge can be uniquely assigned to each contact and the assumption of incremental-charge partitioning is a good one. Circuit simulators use a modification of the Equation (2.14) under forward-bias conditions. The nonlinear capacitance model is linearized at a voltage V_x, where, $0 < V_x < V_{bi}$, and use

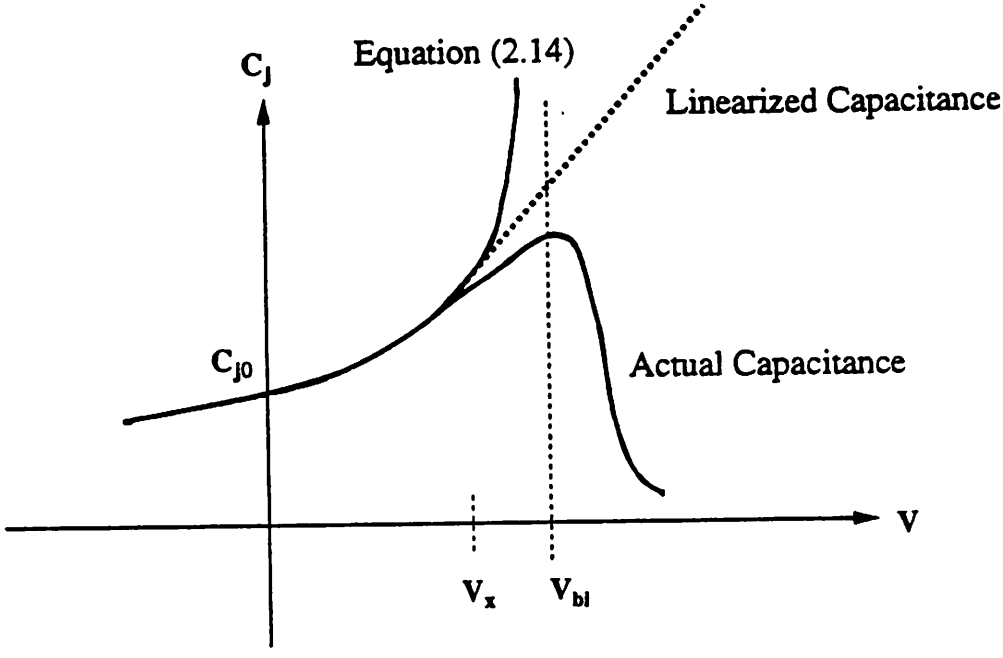


Figure 2.2: Junction capacitance-voltage characteristics

is made of the linearized characteristics. However, this still predicts an increase in the depletion-region capacitance whereas the capacitance decreases for higher forward biases as shown in Figure 2.2 [2.15].

For MOSFET-capacitance models, the application of incremental-charge partitioning provides correct results only for capacitances connected between the gate contact and any of the other contacts [2.45]. However, there is no rigorous basis for partitioning the channel charge between the drain and source terminals. The partitioning that is usually performed is arbitrary and is only motivated by physical operation of the device. There is no theoretical justification that one particular charge-partitioning scheme is better than another [2.32]. Ward [2.44] has developed a channel-charge partitioning scheme that is used in several MOSFET models. This partitioning scheme leads to nonreciprocal capacitances and the source-drain and drain-source capacitances are negative [2.32].

The capacitance models are based on a low-frequency analysis or the quasi-static assumption. A mathematical analysis in [2.45] indicates that this approach is not valid at high frequencies. Thus, low-frequency models do not suffice for high-frequency operation or operation under fast transient conditions.

If a charge-storage model is based on capacitances, it can lead to nonconservation of charge during a transient simulation. It has been shown that a charge-based model is essential for conserving charge [2.12, 2.31]. Therefore, analytical models frequently use closed-form expressions to relate the charges to the terminal voltages. The incremental capacitances are derived from the charge expressions for experimental verification. All experimental device data are based on capacitance measurements since measurement of charges is difficult. However, the capacitance data are obtained from low-frequency measurements rendering them useless for high-frequency operation of the device.

Charge storage of minority carriers is important in diodes and bipolar transistors. The removal of the stored charge results in a delay when the device is turned off and is

of concern in switching circuits. Minority carrier charge storage is modeled by means of an effective transit time in a charge-control representation of the diode or bipolar transistor [2.48]. For the low-frequency or quasi-static model the charge storage is given by the product of the transit time and the current, and the associated capacitance is called the *diffusion capacitance* [2.48]. At frequencies higher than the reciprocal of the transit time, the frequency dependence of the diffusion capacitance is important; the diffusion capacitance decreases as the frequency increases [2.49]. Commonly available SPICE analytical models use a constant value of the diffusion capacitance for all frequencies and are inaccurate at high frequencies.

Even if the analytical model works well under dc conditions, the transient response may be questionable for some regions of device operation. The problems with some of the existing analytical models are now summarized. A detailed comparison of analytical and numerical models is provided in Chapter 7 and substantiates some of the deficiencies outlined here.

The diode models do not account for conductivity modulation [2.50] under high-level-injection conditions. Furthermore, the charge-storage model is inadequate to study the turn off of diodes which is important for the study of power-diode circuits.

Bipolar transistor models are inaccurate for operation under high-level-injection conditions in the collector region, i.e., when base pushout occurs [2.49, 2.50]. Since the charge stored in the collector region is not properly modeled the delay predicted during the turn off of the transistor would be inaccurate. Attempts have been made to modify the Gummel-Poon bipolar model to take into account the base pushout in [2.18, 2.19]. All these models, however, make use of low-frequency capacitance models.

In MOSFET modeling the capacitance models are also based on a quasi-static analysis. Recent attempts in modeling have addressed this problem [2.34, 2.35]; these models effectively solve a one-dimensional electron current-continuity equation for n-

channel MOSFETs. There appears to be no simpler way to model the non-quasi-static operation and hence quasi-numerical approaches are used. Most MOS models make use of a scheme for channel-charge partitioning which is somewhat arbitrary and has no sound theoretical basis. This can lead to errors particularly in circuits in which the charge that flows out of each terminal must be accurately modeled. In spite of some of these shortcomings analytical models have been and are extremely useful for circuit design. If a circuit designer is aware of the limitations of the model, he/she can design a circuit ensuring that devices never operate in regions which are not adequately modeled by the model in use with the circuit simulator.

2.3.2. Empirical or Table Models

As the name suggests empirical models are not based on the physical operation of the device. These models are based on the terminal characteristics of the device; and, therefore, they can be expected to be technology independent when compared with analytical models. The basic idea in a table model is to store a set of discrete data, for the current-voltage and charge-voltage characteristics, in multi-dimensional arrays or tables. The model-evaluation subroutine of the circuit simulator then interpolates between the discrete values that are stored to obtain the current and conductance values for a given set of terminal voltages. Table models have been commonly used for modeling the dc characteristics of a device. These models ensure accuracy in the current-voltage characteristics, but in general do not address the accuracy in conductances which is of concern for analog circuits. The models are very efficient since no complex function evaluations are required; only simple arithmetic operations have to be performed to compute the conductance and current values.

Table models have been successfully used for the simulation of MOS circuits [2.51, 2.52, 2.53]. However, they have not been used for simulation of bipolar transistor

circuits because of the exponential nonlinearity in bipolar devices.

The accuracy of a table model depends upon the number of data points stored in the tables. If a large number of discrete data points are stored, the table model would be very accurate. Various approaches have tried to reduce the dimensions of the problem by use of a smaller number of tables and by modeling some dependencies with analytical expressions [2.51, 2.52]. The dependency on device sizes is incorporated in the tables. Therefore, different tables have to be used for different device sizes. This places a restriction on the device sizes that can be used in simulating a circuit. The various table models differ in the way in which they interpolate data; some models just make use of linear interpolation, whereas others use splines [2.53], or piecewise cubic interpolation [2.54, 2.55]. Alternatively, B-splines can be used for device modeling [2.56].

Table models rely on measured data to generate the multi-dimensional tables of the current-voltage characteristics; therefore, they can be used only after devices have been fabricated with a process. In this mode, table models do not posses any predictive capability. If use is made of data obtained from device-level simulations, table models can also be used in a predictive manner. Since table models only need experimental data, they are insensitive to changes in technology. In fact, the models do not change when the underlying process is modified. This is one of the reasons for using a table-based approach for hardware implementation [2.55].

The use of table models for modeling the intrinsic capacitances of a MOSFET has been very limited. In [2.57] the capacitance model is based on analytical models, and a table model is presented in a later work [2.58]. This table model makes use of measured capacitance data for generating the charge-based tables. An elaborate interpolation scheme and a fixed-charge partition scheme are used for calculating the capacitances. However, no circuit simulation examples have been presented with this model, and it is not clear if the model conserves charge during transient simulations. An alternate

charge-based formulation has been used in [2.59]. A channel-charge partitioning scheme has to be incorporated in the table model similar to that for analytical models. The capacitance/charge tables are again based on data from dc characteristics, and therefore are accurate only for low-frequency operation.

Table models are simple, efficient, and technology independent. They are not very general and have been used only for MOS devices. Furthermore, a good table representation for charge is not available from experimental data. Thus, generation of charge-based table models from experimental data is difficult, and device simulations must be used. The existing charge-based table models are inadequate for high-frequency operation.

2.3.3. Quasi-Numerical Models

These models make use of the basic physical laws that govern the device operation. Simplifications are introduced such that the equations can be solved efficiently by numerical techniques. The models are referred as quasi-numerical models since complete numerical solutions are not used.

In [2.60] a quasi-numerical model is presented for the bipolar transistor in which one-dimensional expressions for device characteristics are integrated across a device to yield two-dimensional characteristics. This approach has been successful in modeling the two-dimensional operation of the device and is at least an order of magnitude faster than conventional device-level simulation [2.60]. However, the model is a dc model and no simple extensions are possible for modeling the transient operation. In addition, the model has been derived for a particular type of bipolar device structure, namely, oxide-isolated walled-emitter npn transistors.

The simulation program BIPOLE [2.61] uses one-dimensional transport equations with a variable boundary regional approach. Two-dimensional effects are handled by

combining one-dimensional analyses in the lateral and vertical dimensions. However, this approach is also restricted to a steady-state solution of the device-level equations and no time-dependent analysis can be performed.

MOS modeling has been addressed by [2.62] and [2.63]. In [2.62] the emphasis has been on an environment for developing quasi-numerical models. A model has been developed only for the threshold voltage of the device. The threshold-voltage model alone is not sufficient for circuit simulation; and, hence, this model is not suitable for use in a circuit simulator.

A quasi-numerical model for the drain current is presented in [2.63] along with a threshold-voltage model. The drain current is obtained by a numerical solution of the one-dimensional current-continuity equation. However, both of these approaches are limited to dc conditions and are inadequate for dynamic operation of the device.

The non-quasi-static MOS models of [2.34] and [2.35] address the problem of dynamic operation of MOSFETs. The one-dimensional current-continuity equation is solved assuming analytical expressions for the various charges. In this sense these models are quasi-numerical instead of being pure analytical models. The model in [2.34] employs a charge-partitioning scheme, whereas the model in [2.35] makes use of quasi-static-charge models for the gate and substrate currents. Physical effects such as velocity saturation are difficult to account for in the models of [2.34] and [2.35].

2.3.4. Numerical Models

Numerical models employ the solution of the basic physical laws governing device operation for determining the characteristics of a device. This involves a numerical solution of Poisson's equation and the current-continuity equations; hence, the name numerical models. These models are accurate and useful for detailed simulations. They also provide a means for predicting the impact of process variations on circuit performance.

However, the CPU time required with numerical models is prohibitive for use in large circuits. They can be used to advantage in the simulation of leaf cells or small circuits.

Commonly, numerical models are used to study internal device operation and the current-voltage characteristics at the device terminals. One of the first application towards circuit simulation was in the SITCAP [2.64] program, where a one-dimensional numerical model for the bipolar transistor is used to generate model parameters for the analytical Gummel-Poon model. The use of numerical models in circuit simulation has been limited to the circuit simulator MEDUSA [2.65]. However, this simulator is not available in the public domain; therefore, numerical models have not found widespread use in circuit simulation in spite of the accuracy that they can provide.

CHAPTER 3

An Overview of Device Simulation

3.1. Introduction

The second component of a mixed-level circuit and device simulator is the device-simulation capability. The semiconductor device-simulation problem can be formulated as a set of three coupled nonlinear partial-differential equations (PDEs) in space and time, which are obtained from the underlying physics. These equations are solved for applied terminal voltages which constitute the boundary conditions for the PDEs. Device simulation also relies on physical models for several phenomena and the models in common use are described. However, an exhaustive survey of models [3.1, 3.2] is beyond the scope of this chapter. The device-simulation problem can be reformulated in different ways by use of transformations that lead to a new set of basic variables. These choices are surveyed along with various techniques for scaling the semiconductor device equations.

The continuous device-simulation problem is discretized both in space and time. Space discretization plays an important role in the overall accuracy of a simulation. Different techniques are available for performing the discretization. Of these, the finite-difference scheme is described in detail, as this technique has been used in CODECS. The finite-box and finite-element formulations are also presented for completeness. The problem of automatic mesh generation and grid refinement is then described.

After space discretization the device equations result in a system of nonlinear differential algebraic equations. For time-domain transient analysis application of a suitable integration scheme leads to a system of nonlinear algebraic equations. These equations are linearized using the Newton-Raphson technique and solved by means of relaxation or direct methods. Some of the main solution techniques currently in use are reviewed.

Small-signal ac analysis is performed on a device by linearizing the device equations at a dc operating point. The problem is described and the approaches to solve it are presented.

3.2. The Device-Simulation Problem

The operation of a semiconductor device can be described by three fundamental equations obtained from the Boltzmann transport equation [3.1]. These are Poisson's equation and the electron and hole-current continuity equations.

$$\nabla.\varepsilon\mathbf{E} = q\,(N_D - N_A + p - n) \tag{3.1a}$$

$$\frac{1}{q}\nabla.\mathbf{J}_n = \frac{\partial n}{\partial t} - (G - R) \tag{3.1b}$$

$$\frac{1}{q}\nabla.\mathbf{J}_p = -\frac{\partial p}{\partial t} + (G - R) \tag{3.1c}$$

where

$$\mathbf{E} = -\nabla\psi \tag{3.2a}$$

$$\mathbf{J}_n = -q\,\mu_n n\,\nabla\psi + qD_n\,\nabla n \tag{3.2b}$$

$$\mathbf{J}_p = -q\,\mu_p p\,\nabla\psi - qD_p\,\nabla p \tag{3.2c}$$

and

ε	=	dielectric constant of the material ($F\,cm^{-1}$)
q	=	electronic charge (C)
ψ	=	electrostatic potential (V)
n (p)	=	electron (hole) concentration (cm^{-3})
E	=	electric field ($V\,cm^{-1}$)
J_n (J_p)	=	electron (hole) current density ($A\,cm^{-2}$)
μ_n (μ_p)	=	electron (hole) mobility ($cm^2\,V^{-1}\,s^{-1}$)
D_n (D_p)	=	electron (hole) diffusivity ($cm^2\,s^{-1}$)
N_D (N_A)	=	donor (acceptor) concentration (cm^{-3})
G	=	net generation rate ($cm^{-3}\,s^{-1}$)
R	=	net recombination rate ($cm^{-3}\,s^{-1}$)

The solution of the above system of equations provides the internal distribution of electrostatic potential and the carrier densities, and the external terminal currents. These equations cannot be solved analytically; therefore, numerical methods have to be used. The first efforts in numerical solution of these semiconductor device equations date back to 1964 when Gummel [3.3] solved the steady-state equations in one dimension for the bipolar transistor. The solution of the time-dependent problem was presented in [3.4]. Solution of the one-dimensional problem was also employed by DeMari for the steady-state [3.5] and transient analysis [3.6] of pn-junction diodes. The steady-state and time-dependent problems have been solved in two-dimensions by several researchers. Some examples of these efforts are the simulation programs PISCES [3.7], HFIELDS[3.8], BAMBI [3.9], GALENE [3.10], FIELDAY [3.11], and CADDET [3.12]. Recent work has been in the solution of the basic equations in three dimensions. As device dimensions are scaled down three-dimensional effects become important. Device-simulation programs that provide three-dimensional analysis are FIELDAY [3.13], TRANAL [3.14],

WATMOS [3.15], and CADDETH [3.16].

3.3. Physical Models for Device Simulation

The basic semiconductor device equations rely on physical models for several different quantities, such as the mobilities and the recombination and generation rates. The models are used to provide a relationship between these quantities and the doping levels, the carrier densities, current densities, and the electric fields. A brief survey of the various models currently in use is provided here.

3.3.1. Carrier-Mobility Models

The modeling of carrier mobilities is important for predicting accurate values of the current densities. The mobility model is used to account for different scattering mechanisms. Carriers are scattered by phonons and defects within the material and this scattering mechanism is called *lattice scattering*. Scattering due to the ionized impurities results in a reduction of the mobility. Different models have been proposed to model the mobility reduction due to ionized impurities. The first model was proposed by Caughey and Thomas [3.17], where an empirical expression is used that fits the experimental data, and is given by

$$\mu(N_T) = \mu_{\min} + \frac{\Delta\mu}{1 + \left[\frac{N_T}{N_{ref}}\right]^{\alpha}} \tag{3.3}$$

where $N_T = N_D + N_A$ is the total doping concentration. Equation (3.3) is used to model the mobilities for both the majority and the minority carriers, although the expression was originally proposed for majority-carrier mobilities. Data for the various constants appearing in Equation (3.3) is available in the literature for silicon at a temperature of 300 K; there is no agreement in the literature as to a consistent set of parameters. This is

possibly due to the differences in processes on which the experimental devices are fabricated and in reality the mobility models have to be "tuned" to a process.

Two other models that are in use in device simulators are the Scharfetter-Gummel model [3.4] and a model due to Arora et al., [3.18]. For the Scharfetter-Gummel mobility model

$$\mu(N_T) = \frac{\mu_0}{\left[1 + \frac{N_T}{N_{ref} + N_T/S}\right]^{1/2}} \tag{3.4}$$

However, this model is valid only at a temperature of 300 K. The model of Arora et al., is similar in form to the model of Caughey and Thomas but has a built-in temperature dependence

$$\mu(N_T) = \mu_{min}\left[\frac{T}{T_0}\right]^{-0.57} + \frac{\Delta\mu\left[\frac{T}{T_0}\right]^{-2.33}}{1 + \frac{N_T}{N_{ref}\left[\frac{T}{T_0}\right]^{2.546}}} \tag{3.5}$$

where $T_0 = 300K$.

The scattering due to electrons and holes becomes important when the carrier densities are large. One approach to account for carrier-carrier scattering is to simply use an alternate expression for N_T of Equation (3.3), namely,

$$N_T = \alpha(N_A + N_D) + (1 - \alpha)(n + p) \tag{3.6}$$

where α is assumed to be 0.34 [3.10].

Mobility is also degraded by the electric field. In fact, for large parallel electric fields the drift velocity of the carriers saturates and the mobility model must account for the phenomenon of velocity saturation. Two frequently used models are the Caughey-Thomas model [3.17] and the Scharfetter-Gummel mobility model [3.4]. In the former

the mobility is expressed as

$$\mu(N, E_l) = \frac{\mu(N)}{\left[1 + \left[\frac{\mu(N)E_l}{v_{max}}\right]^{\beta}\right]^{-1/\beta}} \tag{3.7}$$

where β is 1 for holes and 2 for electrons and v_{max} is 1.1×10^7 cm/sec for electrons and 9.5×10^6 cm/sec for holes [3.10]. E_l is the component of electric field parallel to the current-flow direction. A more physical viewpoint requires the use of the quasi-Fermi level instead of the electric field. However, the differences between use of the electric field and quasi-Fermi levels is small and the use of E_l has been preferred [3.1, 3.10]. For the Scharfetter-Gummel mobility model the electric field dependence is given by

$$\mu(N, E_l) = \frac{\mu_0}{\left[\left[\frac{\mu_0}{\mu(N)}\right]^2 + \frac{(E_l/A)^2}{E_l/A + F} + \left[\frac{E_l}{B}\right]^2\right]^{1/2}} \tag{3.8}$$

Another scattering mechanism, that is important in MOSFETs, is surface scattering due to roughness at the silicon-oxide interface and due to interface states. The mobility models for this scattering phenomenon are empirical in nature. The first mobility model to account for surface scattering was proposed by Yamaguchi [3.19], where the electric field is split into two components E_l, the component parallel to the current (lateral electric field), and E_t, the component of electric field perpendicular to current-flow direction (transverse electric field). The mobility is given by

$$\mu(N, E_l, E_t) = \frac{\mu(N, E_l)}{\left[1 + \alpha E_t\right]^{1/2}} \tag{3.9}$$

This model was later revised by Yamaguchi [3.20] to a physically acceptable empirical model. In this case the mobility is first calculated as

$$\mu(N, E_t) = \frac{\mu(N)}{\left[1 + \alpha E_t\right]^{1/2}} \tag{3.10}$$

The above value of $\mu(N, E_t)$ is used in Equation (3.7) or (3.8) instead of $\mu(N)$.

Selberherr [3.21] has proposed a mobility model for surface scattering that also depends on the distance perpendicular to the interface. In PISCES [3.22] use is made of the Watt-Plummer [3.23] model for surface mobility.

3.3.2. Carrier Generation and Recombination Models

The important recombination and generation mechanisms for silicon are Shockley-Read-Hall (SRH) recombination, Auger recombination, and avalanche generation. The recombination rate for the SRH process is given by

$$R_{SRH} = \frac{np - n_{ie}^2}{\tau_p(n + n_t) + \tau_n(p + p_t)} \tag{3.11}$$

where n, p are the electron and hole concentrations, n_{ie} is the effective intrinsic carrier concentration, τ_n (τ_p) the electron (hole) lifetime and n_t (p_t) is the concentration of traps at an energy level E_t. In the SRH recombination process, transitions between the conduction and valence bands are assisted by traps in the band gap. Since the energy levels of the traps are not known, the most effective trap level is assumed, i.e., at the center of the bandgap [3.24]. Therefore, $n_t = p_t = n_{ie}$.

At higher concentrations the lifetimes decrease due to the creation of additional recombination centers. The dependence of lifetimes on doping is given by empirical relations that fit experimental data. A commonly used expression given in [3.25] is

$$\tau = \frac{\tau_0}{1 + \frac{N_T}{N_{ref}}} \tag{3.12}$$

where $N_T = N_D + N_A$ is the total doping concentration. The experimental data again shows a wide scatter in the various parameters, τ_0 and N_{ref}.

Another recombination process that involves direct band-to-band transitions is Auger recombination. The model for Auger recombination is given by

$$R_{Aug} = (c_n n + c_p p)(np - n_{ie}^2) \tag{3.13}$$

where c_n, c_p are the Auger-capture coefficients. The typical values of these constants at $300K$ are $2.8\times10^{-31}\ cm^6/s$ and $9.9\times10^{-32}\ cm^6/s$, respectively. The Auger coefficients have a very weak dependence on the doping levels, carrier densities, and temperature.

Carrier generation due to impact ionization (avalanche generation) is modeled by

$$G_{Av} = \alpha_n |J_n| + \alpha_p |J_p| \tag{3.14}$$

The ionization coefficients α_n (α_p) depend on the electric field according to Chynoweth's law [3.26],

$$\alpha = \alpha_\infty e^{-B/E_l} \tag{3.15}$$

where E_l is the component of electric field parallel to the current density. The electric field component perpendicular to the current flow does not cause ionization [3.27]. The values of impact ionization parameters α_∞ and B are available in the literature [3.1].

3.3.3. Heavy-Doping Effects

In heavily doped regions of silicon the material becomes degenerate and the bandgap narrows and becomes position dependent. A simple approach to model bandgap narrowing and degeneration is by the use of an effective intrinsic carrier concentration n_{ie}, related to the intrinsic concentration for low doping n_{i0}.

$$n_{ie}^2 = n_{i0}^2\, e^{\frac{\Delta E_g}{kT}} \tag{3.16}$$

where ΔE_g is the effective bandgap narrowing and is determined by electrical measurements. A frequently used model for ΔE_g is due to Slotboom [3.28],

$$\Delta E_g = E_{BGN}\left[\ln\left[\frac{N_T}{N_{BGN}}\right] + \left[\left[\ln\left[\frac{N_T}{N_{BGN}}\right]\right]^2 + 0.5\right]^{1/2}\right] \tag{3.17}$$

where $E_{BGN} = 9\text{mV}$, and $N_{BGN} = 1\times10^{17}\ cm^{-3}$. Since only the total bandgap narrowing can be determined from measurements it is assumed that the shift in conduction and valence band edges is identical, i.e,

$$\Delta E_c = \Delta E_v = \frac{\Delta E_g}{2} = kT\ln\left[\frac{n_{ie}}{n_{i0}}\right] \tag{3.18}$$

The electron and hole concentrations are given by

$$n = n_{ie}\,e^{\frac{\psi - \phi_n}{V_T}} \tag{3.19a}$$

$$p = n_{ie}\,e^{\frac{\phi_p - \psi}{V_T}} \tag{3.19b}$$

where ϕ_n (ϕ_p) is the quasi-Fermi level for electrons (holes). Due to the position-dependent bandgap, an additional electric field is created and the expressions for current densities are now given by [3.29]

$$\mathbf{J}_n = -q\mu_n n\nabla\left[\psi + \frac{\Delta E_c}{q}\right] + qD_n\nabla n \tag{3.20a}$$

$$\mathbf{J}_p = -q\mu_p p\nabla\left[\psi - \frac{\Delta E_v}{q}\right] - qD_p\nabla p \tag{3.20b}$$

Once the above physical phenomena have been modeled the basic semiconductor equations can be solved; the device terminal voltages establish the boundary conditions for the PDEs.

3.4. Boundary Conditions

The boundary conditions can be classified as Dirichlet or Neumann [3.30], or a combination of the two called a mixed-boundary condition. The boundary conditions used in semiconductor device simulation can be classified as physical and artificial boundary conditions [3.1]. The latter boundary conditions have to be imposed to simulate only the intrinsic device.

The physical boundaries are those due to the contacts and interfaces with the oxide or other insulators. The simplest type of contact is an ohmic contact which can be either voltage controlled or current controlled. For a voltage-controlled contact the boundary condition for the electrostatic potential is given by

$$\psi_C = \psi_0 + V_{app} \tag{3.21}$$

where ψ_C is the electrostatic potential at the contact for an applied voltage V_{app} and ψ_0 is the equilibrium potential. If the ohmic contact is to a silicon region the boundary values of the carrier concentrations are obtained by assuming thermal equilibrium and charge neutrality at the contact. These conditions are given by

$$np - n_{ie}^2 = 0 \tag{3.22a}$$

$$N_D - N_A + p - n = 0 \tag{3.22b}$$

From these two conditions it can be shown that

$$\psi_0 = \mathrm{sgn}(N_D - N_A)\ln\left[\frac{|N_D - N_A|}{2n_{ie}} + \left[1 + \frac{|N_D - N_A|^2}{4n_{ie}^2}\right]^{1/2}\right] \tag{3.23a}$$

$$n_C = n_0 = n_{ie}e^{\frac{\psi_0}{V_T}} \tag{3.23b}$$

$$p_C = p_0 = n_{ie}e^{-\frac{\psi_0}{V_T}} \tag{3.23b}$$

The function $\mathrm{sgn}(x)$ is -1 if $x < 0$ and +1 if $x > 0$. Equations (3.21), (3.23b) and

(3.23c) are the Dirichlet boundary conditions for the potential, electron concentration, and hole concentration, respectively.

For the gate contact of a MOSFET, ψ_0 has to be evaluated in a different manner. To derive the boundary condition, consider the energy-band diagram under equilibrium for a MOS-capacitor as shown (the substrate is p-type) in Figure 3.1. With the quasi-Fermi level of the substrate as the reference, the gate potential at equilibrium is given by,

$$\begin{aligned}\psi_{g0} &= |\Phi_{MS}| - |\phi_p| \\ &= \Phi_{SM} - |\phi_p| \end{aligned} \tag{3.24}$$

where Φ_{MS} is the metal-semiconductor work-function difference, and ϕ_p is the hole quasi-Fermi level. With Φ_S expressed in terms of the electron affinity χ, Equation (3.24) can be written as

$$\psi_{g0} = \chi + \frac{1}{2q}\left[E_g + kT\ln\left(\frac{N_C}{N_V}\right)\right] + |\phi_p| - \Phi_M - |\phi_p| \tag{3.25}$$

which can be rearranged to,

$$\psi_{g0} = \chi + \frac{1}{2q}\left[E_g + kT\ln\left(\frac{N_C}{N_V}\right)\right] - \Phi_M \tag{3.26}$$

where E_g is the bandgap, N_C (N_V) is the density of states at the edge of the conduction (valence) band. It should be noted that ψ_{g0} from Equation (3.26) can be calculated independent of the doping level in the silicon region, whereas Equation (3.24) requires information of the doping level.

The interface boundary conditions make use of Gauss's Law in differential form. At the interface,

$$\varepsilon_{Si}E_{Si} - \varepsilon_{ox}E_{ox} = Q_i \tag{3.27a}$$

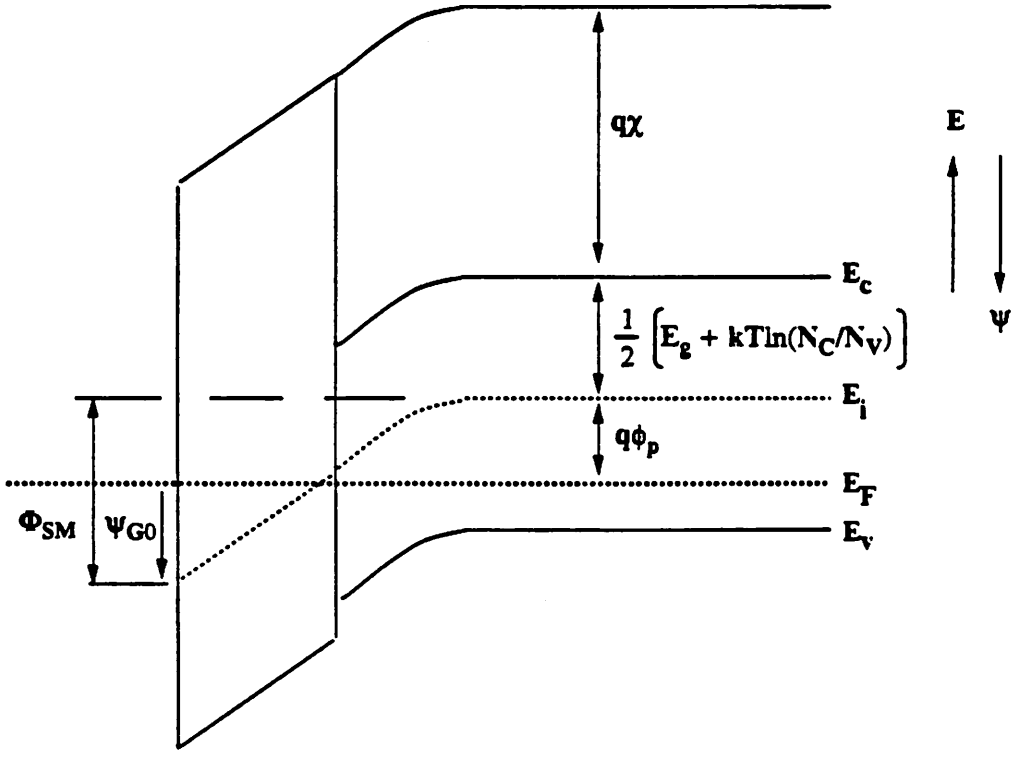


Figure 3.1: Energy band diagram for MOS capacitor at equilibrium.

where Q_i is the interface-charge density. The implementation of this boundary condition is described in Chapter 5. The components of current perpendicular to the interface are given by

$$\mathbf{J}_n . \hat{\mathbf{n}} = -qR_{SURF} \tag{3.27b}$$

$$\mathbf{J}_p . \hat{\mathbf{n}} = qR_{SURF} \tag{3.27c}$$

where $\hat{\mathbf{n}}$ is a unit normal vector, and R_{SURF} is the surface recombination rate. If R_{SURF} is assumed to be zero the boundary conditions (3.27b) and (3.27c) reduce to homogeneous Neumann boundary conditions.

Finally, the artificial boundaries are inserted so that only the intrinsic portion of the device is simulated. These boundaries have to be determined by the physical operation of the device. For a MOSFET as shown in Figure 3.2 these boundaries correspond to the edges A-B and C-D. At such boundaries it is assumed that the components of electric field and current densities normal to the boundary are zero. Thus

$$\mathbf{E}_n . \hat{\mathbf{n}} = 0 \tag{3.28a}$$

$$\mathbf{J}_n . \hat{\mathbf{n}} = 0 \tag{3.28b}$$

$$\mathbf{J}_p . \hat{\mathbf{n}} = 0 \tag{3.28c}$$

Naturally, the boundaries A-B, C-D should be placed such that the above conditions hold good. Otherwise the simulation results would be questionable.

3.5. Choice of Independent Variables

The semiconductor device-simulation problem can be solved as given by Equation (3.1) with ψ, n and p being the basic set of independent variables. One problem with this set of variables is that it can lead to negative, unphysical, values for carrier concentrations due to roundoff errors. Thus, care must be taken to ensure that the carrier

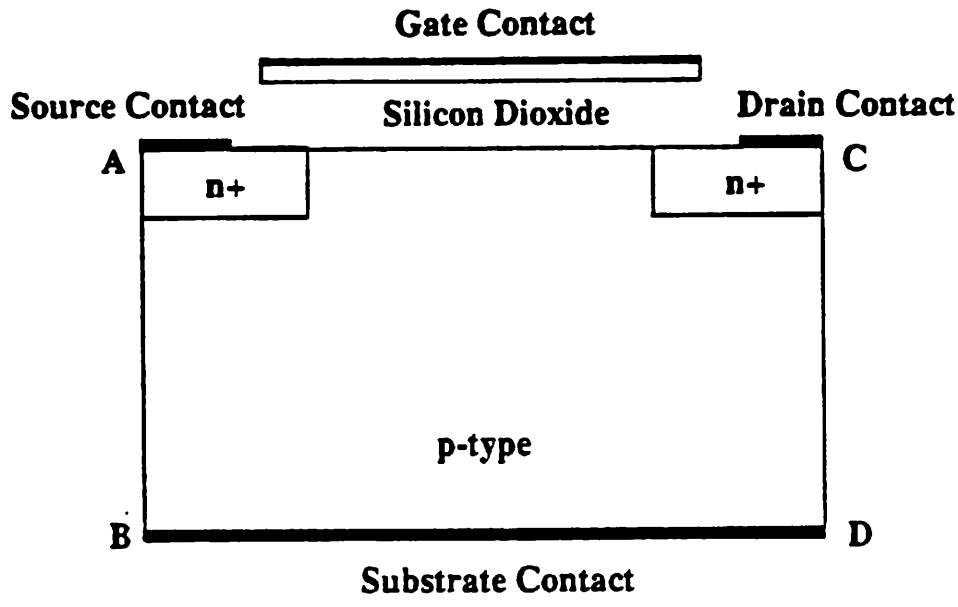


Figure 3.2: Cross-section of a MOSFET.

concentrations are not negative. However, this set of variables is preferred by [3.1], and experiments have indicated that faster convergence is achieved by the use of ψ, n, and p [3.31]. For this reason CODECS makes use of ψ, n, and p as the independent variables. A brief summary of the other variables is included for completeness.

An attractive set is ψ, ϕ_n, and ϕ_p, i.e., the electrostatic potential and the electron and hole quasi-Fermi levels. The advantages are that all the variables are of the same order of magnitude, and carrier concentrations can never become negative. However, the continuity equations become exponentially nonlinear in ϕ_n, ϕ_p. This choice of variables has been used in [3.32]. A comparison in [3.31] indicates that a larger number of iterations is required to reach convergence, possibly due to the severe nonlinearity of the problem.

Slotboom [3.33] has used another set of variables denoted by ψ, u, and v, where

$$u = e^{-\frac{\phi_n}{V_T}} \tag{3.29a}$$

$$v = e^{\frac{\phi_p}{V_T}} \tag{3.29b}$$

The disadvantage is the large range of values that u and v can take, and this set of variables can also lead to negative values of n and p due to roundoff errors.

3.6. Scaling of Semiconductor Equations

The variables ψ, n, p are of widely different orders of magnitude, hence the device equations have to be scaled in an appropriate manner. The classical scaling approach was developed by [3.4] and has been widely used. Another scaling that is rigorous from a mathematical point of view is that used by Markowich [3.34]. Finally, a third scaling scheme is that used in SEDAN [3.35]. The scaling factors for the three different approaches to scaling are presented in Table 3.1.

Variables	Scale Factors	DeMari's	Singular Pert.	SEDAN
potential	V_0	kT/q	kT/q	kT/q
concentration	N_0	n_i	max $\lvert N(x,y)\rvert$	$\sqrt{N_c N_v}$
length	L_0	$\sqrt{\varepsilon V_0/qN_0}$	max (x,y)	$\sqrt{\varepsilon V_0/qN_0}$
mobility	μ_0	q/kT	$q/kT \times \max(D_n, D_p)$	1

Table 3.1: Various scaling techniques

Based on the above scale factors, the scale factors for the other variables can be obtained as in Table 3.2

Variables	Scale Factors	Relation
Electric Field	E_0	V_0/L_0
Current Density	J_0	$q\mu_0 N_0 E_0$
Time	T_0	$L_0^2/\mu_0 V_0$
Frequency	F_0	$1/T_0$
Recombination Rate	R_0	N_0/T_0

Table 3.2: Dependent scale factors

If the scale factors of DeMari or SEDAN are used with a constant ε, the semiconductor equations can be expressed as

$$\nabla.\mathbf{E} = (N_D - N_A + p - n) \tag{3.30a}$$

$$\nabla.\mathbf{J}_n = \frac{\partial n}{\partial t} - (G - R) \tag{3.30b}$$

$$\nabla.\mathbf{J}_p = -\frac{\partial p}{\partial t} + (G - R) \tag{3.30c}$$

The electric fields and current densities are given by

$$\mathbf{E} = -\nabla\psi \tag{3.31a}$$

$$\mathbf{J}_n = -\mu_n \left[n\nabla\psi - \nabla n \right] \tag{3.31b}$$

$$\mathbf{J}_p = -\mu_p \left[p\nabla\psi + \nabla p \right] \tag{3.31c}$$

3.7. Space-Discretization Techniques

The numerical solution of the basic semiconductor device equations involves solution of a discrete problem over a simulation domain. The domain is divided into smaller regions, and the discrete problem is solved for each of these regions with the applied boundary conditions. A nonlinear algebraic system of equations is obtained under

steady-state conditions and the unknowns are the discrete approximations to the continuous variables. The space discretization can be performed by either a finite-difference [3.36] or a finite-element formulation [3.37] and these discretization techniques are now described.

3.7.1. Finite-Difference Discretization

The classical finite-difference scheme can be easily applied only to a rectangular domain. The simulation domain is divided into nonoverlapping rectangular regions by grid lines parallel to the x and y axes. The discretized equations are assembled at each grid node by approximating the spatial derivatives with difference expressions. For this reason the grid lines should be carefully chosen such that the discrete problem is a good approximation to the continuous problem.

For a rectangular mesh with grid spacings $h_i = x_{i+1} - x_i$ and $k_j = y_{j+1} - y_j$, the derivatives of a function $f(x, y)$ are approximated at grid node (i, j) shown in Figure 3.3 by

$$\left[\frac{\partial f}{\partial x}\right]_{i,j} = \frac{f(x_{i+1/2}, y_j) - f(x_{i-1/2}, y_j)}{\frac{h_i + h_{i-1}}{2}} \tag{3.32a}$$

$$\left[\frac{\partial f}{\partial y}\right]_{i,j} = \frac{f(x_i, y_{j+1/2}) - f(x_i, y_{j-1/2})}{\frac{k_j + k_{j-1}}{2}} \tag{3.32b}$$

where $x_{i-1/2} = x_i - \frac{h_{i-1}}{2}$, $x_{i+1/2} = x_i + \frac{h_i}{2}$, $y_{j-1/2} = y_j - \frac{k_{j-1}}{2}$, and $y_{j+1/2} = k_j + \frac{k_j}{2}$.

Use of this approximation results in the following discretizations for Poisson's and the current-continuity equations

$$\frac{E_x|_{i+1/2,j} - E_x|_{i-1/2,j}}{\frac{h_i + h_{i-1}}{2}} + \frac{E_y|_{i,j+1/2} - E_y|_{i,j-1/2}}{\frac{k_j + k_{j-1}}{2}} = \left[N + p - n\right]_{i,j} \tag{3.33a}$$

$$\frac{J_{nx}|_{i+1/2,j} - J_{nx}|_{i-1/2,j}}{\frac{h_i + h_{i-1}}{2}} + \frac{J_{ny}|_{i,j+1/2} - J_{ny}|_{i,j-1/2}}{\frac{k_j + k_{j-1}}{2}} = \left[\frac{\partial n}{\partial t}\right]_{ij} - \left[G - R\right]_{i,j} \qquad (3.33b)$$

$$\frac{J_{px}|_{i+1/2,j} - J_{px}|_{i-1/2,j}}{\frac{h_i + h_{i-1}}{2}} + \frac{J_{py}|_{i,j+1/2} - J_{py}|_{i,j-1/2}}{\frac{k_j + k_{j-1}}{2}} = -\left[\frac{\partial p}{\partial t}\right]_{ij} + \left[G - R\right]_{i,j} \qquad (3.33c)$$

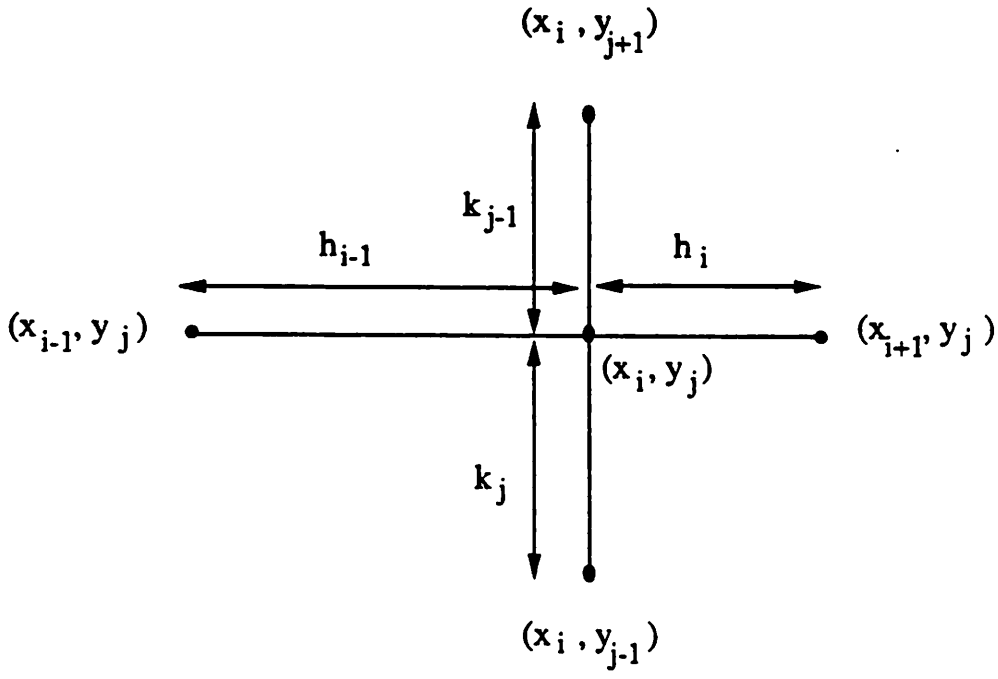


Figure 3.3: Schematic of grid for finite-difference discretization.

In the above equations the discrete values of $E_{x,y}$, $J_{nx,y}$ and $J_{px,y}$ are required at the midpoints of each interval $\left[x_i, x_{i+1}\right]$ or $\left[y_j, y_{j+1}\right]$. These values are approximated from the nodal values of the electrostatic potential and carrier concentrations. For the electric field use is made of a simple difference scheme, whereby

$$E_x \mid_{i+1/2,j} = - \frac{\Psi_{i+1,j} - \Psi_{i,j}}{h_i} \tag{3.34}$$

This approximation corresponds to the assumption of a constant electric field between two adjacent grid points. However, a similar difference scheme cannot be used for the current densities at the midpoint of an edge, as this scheme is unstable for a mesh in which the electrostatic potential between two grid points is larger than twice the thermal voltage [3.38]. Scharfetter and Gummel [3.4] found this instability in the simple discretization scheme and proposed an alternate way for computing the current densities. This scheme can be illustrated by considering the electron current density $J_{nx} \mid_{i+1/2,j}$, that has to be evaluated between grid points (x_i, y_j) and (x_{i+1}, y_j). From the normalized drift-diffusion equation for electrons

$$J_{nx} = \mu_n \left[nE_x + \frac{dn}{dx} \right] \tag{3.35}$$

If the electric field, current density, and mobility are assumed to be constant between the grid points, at their midpoint values, the above expression can be written as

$$J_{nx} \mid_{i+1/2,j} = \mu_n \mid_{i+1/2,j} \left[nE_x \mid_{i+1/2,j} + \frac{dn}{dx} \right] \tag{3.36}$$

which is an ordinary differential equation in the electron concentration n. Equation (3.36) can be integrated to get an expression for $n(x, y_j)$ for the interval $\left[x_i, x_{i+1}\right]$. Using the value of $n(x, y_j)$ at (x_{i+1}, y_j), one can express $J_{nx} \mid_{i+1/2,j}$ as

$$J_{nx} \mid_{i+1/2,j} = \frac{\mu_n \mid_{i+1/2,j}}{h_i} \left[n_{i+1,j} B(\Psi_{i+1,j} - \Psi_{i,j}) - n_{i,j} B(-(\Psi_{i+1,j} - \Psi_{i,j})) \right] \tag{3.37}$$

where $B(x) = \dfrac{x}{e^x - 1}$ is the Bernoulli's function. The Scharfetter-Gummel discretization can be used with coarse grids to obtain stable estimates for the current densities. This completes the space discretization of the device equations.

As described, the finite-difference scheme can be applied only to a rectangular grid and appears to be very limited, since it cannot be used for an arbitrary simulation domain. Finite-element schemes allow the use of different types of elements for space discretization which can be used for arbitrary domains. Extensions of the finite-difference scheme have been made such that it can be used with triangular grids. The discretization is performed by the box integration method. The finite-difference scheme on triangular grids is used in several device simulators [3.7, 3.8, 3.11]. Triangular grids make it possible to discretize the device equations over an arbitrary simulation domain. In addition grid nodes can be selectively placed in active regions of the device. The box-integration method for a rectangular grid is described in Chapter 5, along with the discretization of the boundary conditions.

An extension of the finite-difference approach is the method of finite boxes. In this method grid points in superfluous or inactive regions of the simulation domain can be eliminated by the use of grid lines terminating in the interior of the simulation region. Adler [3.39] has used the approach of terminating grid lines in only one direction. A generalization of this approach, wherein grid lines can terminate in both directions, is the method of finite boxes as used by [3.40]. The advantage of the finite-box method is the reduction in the number of grid points. However, this is achieved at the expense of additional storage and elaborate data structures [3.40].

3.7.2. The Finite-Element Method

The method of finite elements has been extensively applied to problems of structural engineering for several years. However, its application to the problem of semiconductor device simulation is fairly recent [3.41].

In this method, analogous to the finite-difference method, the domain of simulation is partitioned into a finite number of nonoverlapping subdomains or elements. These

elements are taken to be triangular or rectangular in shape. Once the elements have been selected, an approximation $\hat{f}$ of the exact solution f is found for each element. The partial solution for each element is such that its contribution to the complete approximate solution is zero outside the element. Then, the complete solution is the sum of partial solutions over each of the elements,

$$\hat{f} = \sum_{i=1}^{N} \hat{f}_i \tag{3.38}$$

where N is the number of elements and $\hat{f}_i$ is the partial solution for element i.

The commonly used approximation for $\hat{f}_i$ is a low-order polynomial which can be formulated in terms of basis or shape functions as

$$\hat{f}_i = \sum_{j=1}^{k} f_j \xi_j \tag{3.39}$$

where k is the number of nodes in the element ($k = 3$ for a triangular element and $k = 4$ for a rectangular element), f_j are the values of f at these nodes, that are to be determined by the finite-element method, and ξ_j are the shape functions. The shape functions are nonzero over the specified element and zero outside the element. The complete solution is then given by

$$\hat{f} = \sum_{i=1}^{N} \left[\sum_{j=1}^{k} f_j \xi_j \right]_i \tag{3.40}$$

Let $\mathbf{F}(\mathbf{w}) = \mathbf{O}$ denote the basic partial-differential equations for the semiconductor device and $\mathbf{G}(\mathbf{w}) = \mathbf{O}$ denote the boundary conditions. Then $\mathbf{F}(\hat{\mathbf{f}})$ and $\mathbf{G}(\hat{\mathbf{f}})$ are the residuals obtained by substituting the approximate solution into the differential equations and the boundary conditions, respectively. The next step is to use a sum of integrated weighted residuals of the differential equation and the boundary conditions, and equate the sum to zero for a given set of weight functions. This system of equations is then

solved for f_i. If the weighting functions are chosen to be the same as the shape functions the method of weighted residuals is called a Galerkin method [3.42].

One problem with the use of the classical finite-element method when applied to semiconductor device problems is the use of low-order polynomials for approximating the carrier concentrations. The carrier concentrations are exponential functions and this results in poor performance of the finite-element method compared to the finite-difference method in which Scharfetter-Gummel discretization is used for the current continuity equations.

3.7.3. Grid/Mesh Generation

In the previous discussion of finite-difference and finite-element methods, a grid is assumed and the discretization is performed with this grid in mind. The grid spacings were assumed to be such that the electrostatic potential and the carrier densities could be computed accurately from the discretized equations. However, the problem of grid generation, which involves use of a suitable mesh with the smallest number of grid points, was not addressed. With a very large number of grid points a solution with greater accuracy can be computed but it requires a significant amount of computational effort. It is for this reason that an efficient mesh is required, which provides accuracy without excessive computational burden. Heuristics are necessary for the design of a good mesh since an appropriate mesh cannot be designed a priori.

In general, adaptive-grid refinement is used to generate an efficient grid for the structure to be simulated. The starting point is a grid that resolves the geometry and the doping levels in the device. An initial solution is computed on this grid and then error estimates are computed from the solution. Grid points are inserted in regions where the solution does not meet the user-specified accuracy criterion. The solution has to be interpolated for the new grid points that have been inserted. Then the solution is recomputed

on this new grid and the grid is readapted if the desired error tolerances are not achieved or a minimum grid spacing is reached. For certain devices, such as the MOSFET and the bipolar transistor, a grid can be designed a priori by a knowledge of the operation of the device. In other situations, such as latchup, adaptive-grid refinement is a must. Some adaptive-grid generation strategies tend to align the grids with the current-flow direction [3.43].

3.8. Solution Methods for Device Equations

The techniques used to solve the device-level equations for dc, transient and small-signal ac analysis are described in this section. After space discretization, the equations at a grid node i can be expressed in symbolic form as

$$\text{Poisson's equation:} \quad F_{\psi i} = 0 \tag{3.41a}$$

$$\text{n–continuity equation:} \quad F_{ni} - \left[\frac{\partial n}{\partial t}\right]_i = 0 \tag{3.41b}$$

$$\text{p–continuity equation:} \quad F_{pi} - \left[\frac{\partial p}{\partial t}\right]_i = 0 \tag{3.41c}$$

The complete system of equations when assembled can be represented in a general form as

$$\mathbf{F}(\dot{\mathbf{w}}(t), \mathbf{w}(t)) = 0 \tag{3.42}$$

where $\mathbf{w}$ is the vector of electrostatic potential, electron concentration, and hole concentration at each grid node.

3.8.1. Dc and Transient Analyses

After space discretization, a system of nonlinear differential-algebraic equations is obtained for transient analysis; for dc analysis this reduces to a system of nonlinear algebraic equations. The system of nonlinear equations obtained from dc analysis can be solved by a direct method using a Newton scheme. This method is the best from a performance and convergence point of view [3.31]. However, it requires a large amount of storage and computational effort, since a large system of equations has to be solved. Thus, efficient ways of solving the above nonlinear problem are currently under investigation.

The very first numerical solution of the semiconductor device equations due to Gummel [3.3] made use of a relaxation-based approach. The equations are solved by use of nonlinear relaxation. This method can be explained by assuming the variables to be ψ, ϕ_n, ϕ_p. Let the nonlinear system of equations be expressed as

$$F_{\psi}(\psi, \phi_n, \phi_p) = 0 \tag{3.43a}$$

$$F_{\phi_n}(\psi, \phi_n, \phi_p) = 0 \tag{3.43b}$$

$$F_{\phi_p}(\psi, \phi_n, \phi_p) = 0 \tag{3.43c}$$

then Gummel's algorithm can be described as follows. Equation (3.43a) is solved for ψ^{k+1} with the values of ϕ_n and ϕ_p from the previous iteration. The computed value of ψ^{k+1} is used in Equations (3.43b) and (3.43c) to solve for ϕ_n^{k+1} and ϕ_p^{k+1}, respectively. The whole process is repeated until convergence is achieved, whereby a consistent value has been obtained for ψ, ϕ_n, and ϕ_p. Each equation is nonlinear and Newton's method is used to solve the nonlinear system of equations.

Gummel's method works well under low-level injection conditions when Poisson's equation is effectively decoupled from the current-continuity equations. However, under high-level injection conditions Poisson's equation becomes strongly coupled with the

continuity equations; the mobile charges affect the electric field distribution. Under such conditions the relaxation method has extremely slow convergence and may require a very large number of iterations to reach convergence. From a performance point of view, this method loses its advantage under high-level injection conditions and cannot be used for device-level simulations employing arbitrary bias conditions.

Two modifications have been proposed for Gummel's method which ensure better convergence under high-level injection conditions. The first method referred to as quasi-simultaneous relaxation (QSR) [3.44] has been applied to the simulation of MOSFETs operating under strong inversion. This is again a condition under which Gummel's method has very slow convergence. In QSR, Poisson's equation and the one-dimensional current-continuity equation for the carriers in the channel are solved in a coupled manner. Then the electron and hole continuity equations are solved as in the regular Gummel's method. This method inherently takes into account the coupling of Poisson's equation with one continuity equation; thus, the nonlinear relaxation is accelerated.

The other approach [3.45] considers a partial coupling between Poisson's equation and the current-continuity equation for the carrier that dominates the current flow. The coupling of Poisson's equation to the other continuity equation is ignored as in Gummel's method. As a consequence of the assumption that one carrier contributes to the total current flow, the above approach is useful for one-carrier simulation of MOS-FETs or of highly asymmetric pn-junctions. It cannot be applied for the simulation of bipolar transistors, where current flow is due to both carriers under high-level injection conditions.

For transient analysis, the terms $\frac{\partial n}{\partial t}$ and $\frac{\partial p}{\partial t}$ have to be discretized in time. This is done by use of an integration formula as in the circuit-simulation problem described in Chapter 2. Some device simulators use semi-explicit integration methods so that the

equations can be solved in a decoupled manner. However, these decoupled schemes are restrictive in the allowable timesteps. A stable decoupled scheme due to [3.46] is stable independent of the timestep. In [3.1, 3.47] the use of an implicit integration method has been recommended to obtain reliable results.

3.8.2. Small-Signal Ac Analysis

As in the small-signal ac analysis at the circuit level, small-signal analysis at the device level involves determining the terminal currents of a device in response to "small" sinusoidal voltages applied to its terminals, after a dc operating point has been established. The magnitude of the excitation is such that the operation of the nonlinear device is linear around the operating point and negligible harmonics are generated.

Consider the device equations to be written in a general form as

$$\mathbf{F}_{\psi}(\psi, \mathbf{n}, \mathbf{p}) = \mathbf{O} \tag{3.46a}$$

$$\mathbf{F}_{n}(\psi, \mathbf{n}, \mathbf{p}) - \frac{\partial \mathbf{n}}{\partial t} = \mathbf{O} \tag{3.46b}$$

$$\mathbf{F}_{p}(\psi, \mathbf{n}, \mathbf{p}) + \frac{\partial \mathbf{p}}{\partial t} = \mathbf{O} \tag{3.46c}$$

where the time-derivative terms have been specified explicitly. Small-signal ac analysis involves finding the ac response at an established dc operating point (ψ_0, $\mathbf{n}_0$, $\mathbf{p}_0$, V_0) where V_0 is the applied bias. The dc quantities are perturbed by an ac signal of a small magnitude, i.e., all quantities, u, are expressed as the sum of a dc value and an ac value

$$u = u_0 + \tilde{u}e^{j\omega t} \tag{3.47}$$

where $\tilde{u}$ is a complex quantity. Substituting these expressions for the internal variables in Equation (3.46) and retaining the linear terms from a Taylor series expansion around the operating point, one can write the device-level equations as

$$\begin{bmatrix} \frac{\partial F_\psi}{\partial \psi} & \frac{\partial F_\psi}{\partial n} & \frac{\partial F_\psi}{\partial p} \\ \frac{\partial F_n}{\partial \psi} & \frac{\partial F_n}{\partial n} - j\omega \mathbf{I} & \frac{\partial F_n}{\partial p} \\ \frac{\partial F_p}{\partial \psi} & \frac{\partial F_p}{\partial n} & \frac{\partial F_p}{\partial p} + j\omega \mathbf{I} \end{bmatrix} \begin{bmatrix} \tilde{\psi} \\ \tilde{n} \\ \tilde{p} \end{bmatrix} = - \begin{bmatrix} \frac{\partial F_\psi}{\partial V} \\ \frac{\partial F_n}{\partial V} \\ \frac{\partial F_p}{\partial V} \end{bmatrix} \tilde{V} \tag{3.48}$$

where all the derivatives are evaluated at the dc operating point. The above system of equations can be written in a compact form as

$$\left[\mathbf{J} + j\mathbf{D}\right]\tilde{\mathbf{w}} = \mathbf{B} \tag{3.49}$$

where **J** is the dc Jacobian matrix of the device-level equations, **D** is a diagonal matrix with entries 0, $-\omega$, ω, corresponding to Poisson's equation and the electron and hole current-continuity equations, respectively, $\tilde{\mathbf{w}}$ is the vector of small-signal values of the electrostatic potential, and electron and hole concentrations, and **B** is the right-hand-side vector that accounts for the boundary conditions. This system of equations can be solved using direct-solution techniques to compute the small-signal quantities. A faster solution technique due to Laux [3.48] can be used at low frequencies. This technique is based on a splitting of the original system of equations into its real and imaginary parts. Let $\mathbf{w} = \mathbf{w}_R + j\mathbf{w}_I$ then

$$\begin{bmatrix} \mathbf{J} & -\mathbf{D} \\ \mathbf{D} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{w}_R \\ \mathbf{w}_I \end{bmatrix} = \begin{bmatrix} \mathbf{B} \\ \mathbf{O} \end{bmatrix} \tag{3.50}$$

This system of equations can be solved by a Block-SOR technique [3.48] as

$$\mathbf{w}_R^{k+1} = \left(1 - \omega_R\right)\mathbf{w}_R^{k+1} + \omega_R \mathbf{J}^{-1}\left[\mathbf{D}\mathbf{w}_I^k + \mathbf{B}\right] \tag{3.51a}$$

$$\mathbf{w}_I^{k+1} = \left(1 - \omega_R\right)\mathbf{w}_I^{k+1} + \omega_R \mathbf{J}^{-1}\mathbf{D}\mathbf{w}_R^k \tag{3.51b}$$

where ω_R is the SOR relaxation parameter. The advantage with the above scheme is that **J** is available in its LU factors from the dc operating point solution and calculation of $\mathbf{J}^{-1}$ requires only forward and back substitutions. Since the LU factors from the dc

operating point analysis can be used to calculate the solution at other frequencies, this method is considerably faster than a direct solution of the small-signal equations. However, at high frequencies $f > f_T/10$, where f_T is the unity-gain frequency of the device, the convergence of the block-relaxation method degrades and at very high frequencies a direct-solution method is favored [3.48].

Ac analysis at the device level as described above is quite useful. It can be used to verify compact circuit-level analytical models. In addition, it provides information on the unity-gain frequency of the device.

CHAPTER 4

Coupled Circuit and Device Simulation

4.1. Introduction

A mixed-level circuit and device simulator should support a variety of analyses and numerical models. It should provide capabilities for dc, transient, small-signal ac, and pole-zero analyses. These analyses must be available for one- and two-dimensional numerical devices and, if computational resources permit, even for three-dimensional numerical devices. In this chapter the problem of coupled circuit and device simulation is addressed. The techniques are applicable to one-, two-, or three-dimensional numerical devices but have been implemented for only one- and two-dimensional devices. The emphasis is on algorithms to couple the device-simulation capability with the circuit simulator for dc, transient, small-signal ac, and pole-zero analyses.

The problem of coupled circuit and device simulation is illustrated by an example of a dc problem. A circuit-level interpretation of the simulation problem provides a motivation for the two-level Newton algorithm. This scheme requires calculation of linearized equivalent conductances for the numerical devices which are then used in the linearized circuit-level equations. It is shown that the linearized conductances can be obtained by forward and back substitutions and do not require a large computational effort.

The two-level Newton algorithm is used to propose an architecture for coupling the device simulator to the circuit simulator. The proposed architecture allows complete

decoupling between the circuit and device simulators; the numerical models are viewed as another model type from the circuit simulation point of view. The interface between the two simulators is well defined and the interface routines necessary for this coupling are described.

A general formulation for the coupled problem under dc conditions is then provided, by means of an example. This leads to other approaches for coupling the device and circuit simulators. In particular the full LU-decomposition technique and the block-iterative algorithm used in MEDUSA [4.1] are described. A modification of the two-level Newton algorithm is also proposed wherein a linear prediction is used to provide a better initial guess. The implementation of all four algorithms: (1) the two-level Newton algorithm, (2) the full LU-decomposition algorithm, (3) MEDUSA's block-iterative algorithm, and (4) the modified two-level Newton algorithm, within the framework of CODECS is presented. It is shown that all of these algorithms can be implemented in the proposed framework in a decoupled manner.

The convergence properties of these algorithms are then examined for dc and transient analyses. A performance analysis indicates that the modified two-level Newton method is the best for dc analysis, since the other three schemes do not converge on some benchmark circuits. For transient analysis the full LU-decomposition scheme is better from a convergence and performance point of view. The MEDUSA algorithm is found to be unsuitable for both dc and transient analysis. Some parallelization issues of the four algorithms are also presented.

The last part of this chapter is devoted to coupling the two simulators for small-signal ac and pole-zero analyses. Admittances for the numerical devices are calculated by a numerical small-signal analysis of the device and used in the circuit-level equations.

4.2. DC and Transient Analyses

Since dc and transient simulations are by far the most useful, a major part of this chapter addresses algorithms for dc and transient analyses in a mixed-level circuit and device-simulation environment. A dc operating point analysis is required before a transient run is initiated or for small-signal ac and pole-zero analyses. For this reason convergence under dc conditions is extremely important; hence, the algorithms used for dc analysis must exhibit good convergence properties. The transient analysis problem is better conditioned than the dc problem and solutions from the previous timepoints provide a good initial prediction for the solution at the present timepoint. It can be anticipated that an algorithm different from the one used in dc analysis may perform better. This section investigates different ways to couple the device and circuit simulation capabilities for dc and transient analyses. An evaluation is then made on their convergence properties and runtime performance using a variety of benchmark examples.

4.2.1. The Two-Level Newton Algorithm

The problem of mixed-level device and circuit simulation is best illustrated by an example of dc operating point analysis. Consider the simple circuit shown in Figure 4.1. G is a linear conductance and E_s is a dc voltage source. The nonlinear device is a diode for which the characteristics are specified by a doping profile $N(x)$. The simulation problem can be stated as:

Given E_s, G, and $N(x)$

Find V.

One solution for this problem is motivated by examining the problem from a circuit simulation point of view. In this case, analytical models are used for the nonlinear devices; the diode terminal characteristics are described by a closed-form expression, $i = I(V)$. When Newton's method is used to solve the nonlinear circuit equations, a

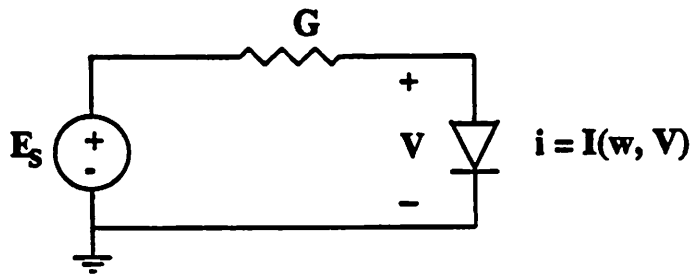


Figure 4.1: Circuit used as an example.

linear circuit (the companion circuit) is solved at each iteration until convergence is achieved. The companion circuit for the example under consideration is shown in Figure 4.2.

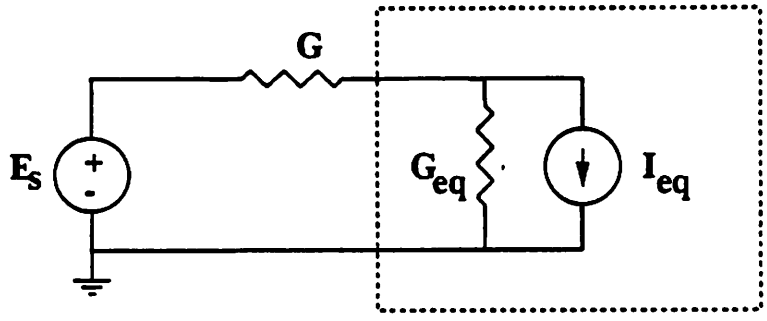


Figure 4.2: Linearized companion circuit.

The linear conductance, G_{eq}, and the current source, I_{eq}, are obtained from the nonlinear characteristics as depicted in Figure 4.3, and can be expressed as

$$G_{eq}^{k+1} = \left[\frac{\partial i}{\partial V}\right]_k \tag{4.1}$$

$$I_{eq}^{k+1} = i^k - \left[\frac{\partial i}{\partial V}\right]_k V^k \tag{4.2}$$

where k is the iteration number.

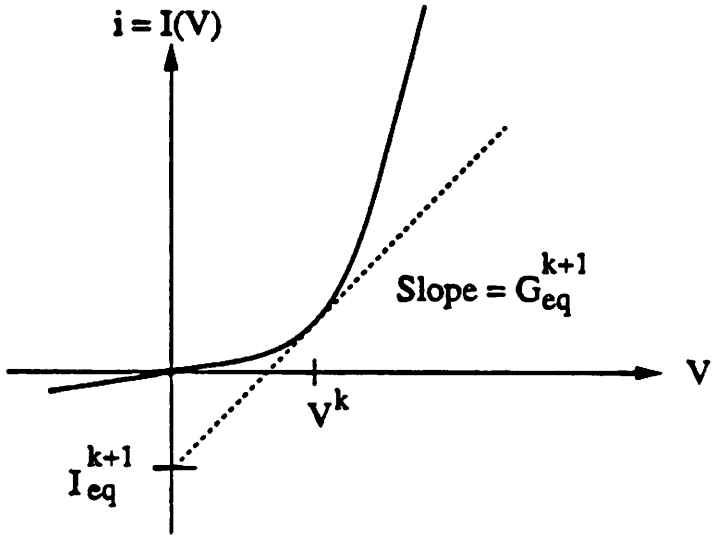


Figure 4.3: Calculation of linearized conductance and current.

Once G_{eq} and I_{eq} are known, the circuit-level iteration can be performed. With numerical devices, the current-voltage characteristics are not known as closed-form expressions. Thus, G_{eq} and I_{eq} cannot be calculated by function evaluations and numerical techniques must be used. The partial-differential equations describing a device have to be solved for each operating point before I_{eq} and G_{eq} can be calculated.

As shown in Chapter 3 the steady-state device-level equations result in a system of nonlinear algebraic equations after a space discretization. These nonlinear equations are also solved by a Newton method. Once the equations have been solved for an applied bias V, the equivalent currents and conductances can be calculated as described in

Section 4.2.2. The overall solution technique is a two-level Newton scheme wherein Newton's method is used at the device level and also at the circuit level. This is a special case of the multi-level Newton algorithm proposed in [4.2] for circuit simulation.

The flowchart of the two-level Newton algorithm is illustrated in Figure 4.4. First, the contributions of all circuit elements that are represented by analytical models are entered into the circuit-level Jacobian matrix and the right-hand-side vector. The partial-differential equations are solved for each numerical device, with the terminal voltages establishing the boundary conditions, until convergence is achieved at the device level. Once the solution at the device level has been obtained, G_{eq} and I_{eq} are calculated and assembled in the circuit-level equations. The linearized circuit-level equations are then solved and convergence is checked at the circuit level. If convergence is achieved, the solution has been obtained; otherwise the outer circuit-level loop is repeated.

4.2.2. Calculation of Conductances

This section describes the technique used for calculating the equivalent conductances and is applicable to all types of numerical models. After space and time discretization the device-level equations can be represented as a set of nonlinear algebraic equations,

$$\mathbf{F}(\mathbf{w}, V) = \mathbf{O} \tag{4.3}$$

where $\mathbf{w}$ is the vector of internal variables, i.e., the electrostatic potential, and the electron and hole concentrations at each spatial grid point. If there are n spatial grid points in the silicon region and m grid points in the oxide region within the device, then the dimension of $\mathbf{w}$ is $3n+m$. The dependence on the boundary condition V is explicitly written in the above equation. For a two-terminal device, let $i = I(\mathbf{w}, V)$ represent the terminal current as a function of $\mathbf{w}$ and V; i is calculated by summing the current

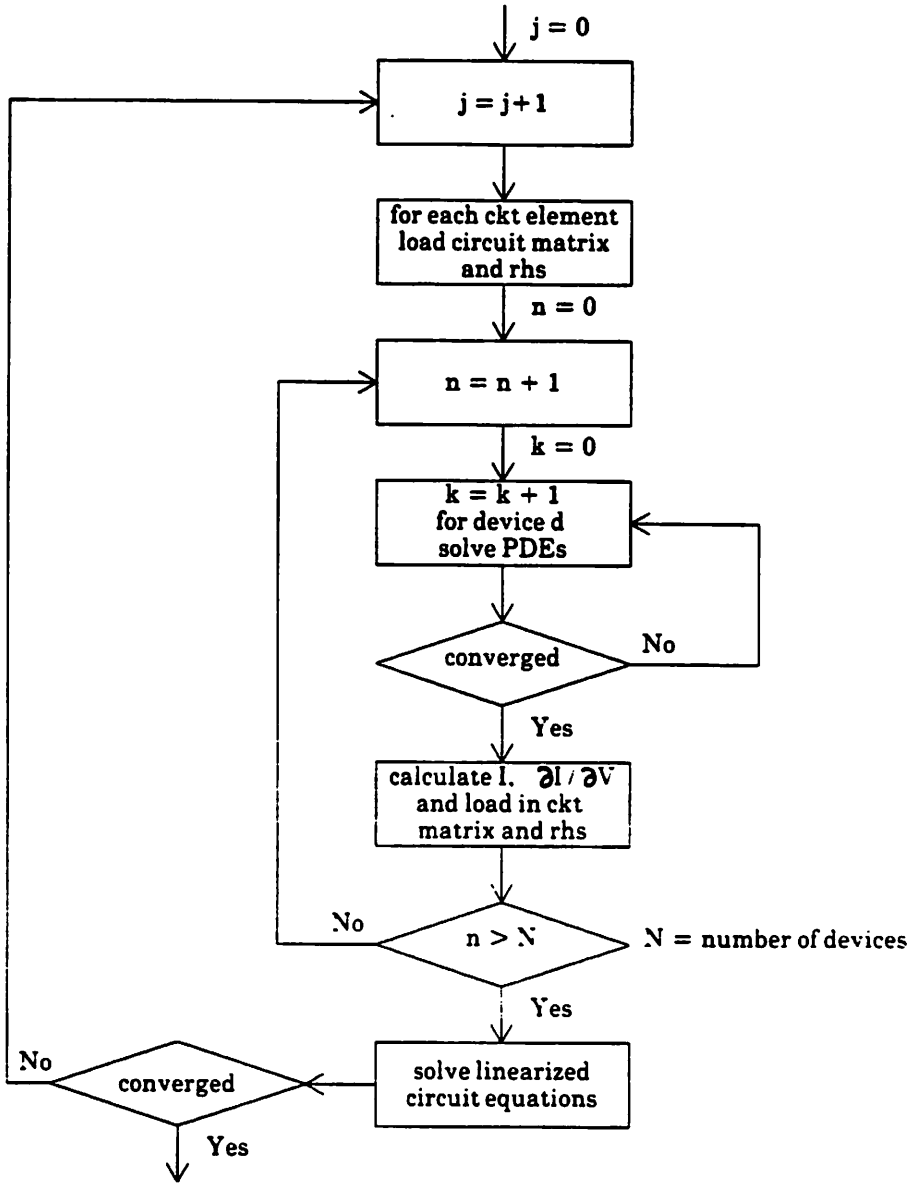


Figure 4.4: Flowchart of the two-level Newton scheme.

density components around a contact. It should be noted that w is also an implicit function of V, since the value of w depends on the applied voltage. The system of Equation (4.3) is solved for an applied voltage V_o. This is done by Newton's method whereby

$$\Delta \mathbf{w} = -\mathbf{J}_w^{-1} F(\mathbf{w}, V_o) \tag{4.4}$$

is solved at each iteration; $\mathbf{J}_w = \frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ is the Jacobian matrix of the device-level equations. $\mathbf{J}_w$ is decomposed into its LU factors at each iteration and $\Delta \mathbf{w}$ is obtained by forward and back substitutions. The iterations terminate when $\Delta \mathbf{w}$ satisfies the convergence tolerance and $|F(\mathbf{w}, V_o)|$ is sufficiently small. At this stage w is the solution of $F(\mathbf{w}, V_o) = \mathbf{O}$ and $I(\mathbf{w}, V_o)$ can be calculated, since $\mathbf{w}(V_o)$ is known. To calculate the linearized conductance $G_{eq} = \frac{\partial i}{\partial V}$, use is made of the chain rule which gives

$$G_{eq} = \frac{\partial i}{\partial V} = \frac{\partial I}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial V} + \frac{\partial I}{\partial V} \tag{4.5}$$

where $\frac{\partial I}{\partial \mathbf{w}}$ and $\frac{\partial I}{\partial V}$ are obtained by symbolic differentiation of the function $I(\mathbf{w}, V)$. $\frac{\partial \mathbf{w}}{\partial V}$ is determined in the following manner. The partial derivative of Equation (4.3) with respect to V is

$$\mathbf{J}_w \frac{\partial \mathbf{w}}{\partial V} + \mathbf{J}_V = \mathbf{O} \tag{4.6}$$

with $\mathbf{J}_V = \frac{\partial \mathbf{F}}{\partial V}$. From Equation (4.6) one can solve for $\frac{\partial \mathbf{w}}{\partial V}$ as

$$\frac{\partial \mathbf{w}}{\partial V} = -\mathbf{J}_w^{-1} \mathbf{J}_V \tag{4.7}$$

Since $\mathbf{J}_w$ is available in its LU factors that were calculated during the solution of Equation (4.3) by use of Equation (4.4), only forward and back substitutions are required in calculating $\frac{\partial \mathbf{w}}{\partial V}$ which is computationally inexpensive. Then G_{eq} can be calculated from

Equation (4.5).

An alternative way to compute G_{eq} is to perturb the terminal voltage V_o by a small amount δV and then calculate the current for the new voltage $V_o + \delta V$. G_{eq} is computed as

$$G_{eq} = \frac{i(V_o + \delta V) - i(V_o)}{\delta V}$$

which is an approximation to $\frac{\partial i}{\partial V}$. This approach is computationally expensive requiring two device-level solutions for each bias point. Furthermore, the difference $i(V_o + \delta V) - i(V_o)$ is prone to errors especially when $i(V_o + \delta V)$ and $i(V_o)$ are close to one another, since it involves the difference of two nearly equal quantities. It is also difficult to choose an appropriate value of δV such that the divided-difference scheme provides accurate results [4.3]. Therefore, the first scheme to calculate conductances is used.

4.2.3. Architecture of CODECS

It is clear from the previous sections that a numerical device model is similar to an analytical device model in several respects for circuit simulation. Given the terminal voltages the equivalent currents and conductances have to be calculated and used in the circuit-level equations. For an analytical model this task involves function evaluations, whereas for a numerical device the three PDEs have to be solved. The interface to a circuit simulator can be identical for the two types of models as shown in Figure 4.5, where the task of model evaluation is illustrated. The interface to the circuit simulator is through routines for model evaluation, and for loading the equivalent current and conductances in the circuit-level Jacobian and right-hand-side vector.

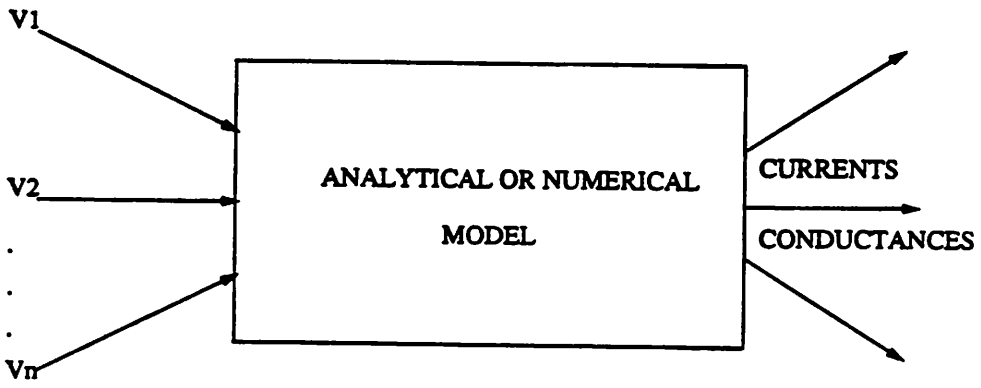


Figure 4.5: The task of model evaluation.

The overall framework of CODECS is shown in Figure 4.6. The circuit simulator is the controlling program. It supports analytical models for the circuit elements and also stores the vector of node voltages. These voltages are available to the model-evaluation subroutines that calculate the equivalent conductances and currents for a device. The numerical devices are simulated by the device simulator of CODECS, and the interface to the circuit simulator is identical to that for analytical models. Device-level simulation is used to solve the PDEs for a numerical device for given terminal voltages. Then the terminal conductances and currents are calculated at the operating point, and assembled in the circuit-level Jacobian matrix and right-hand-side vector.

4.2.4. The Full-Newton Algorithm

With this method the problem is formulated in an alternate manner. The device-level and circuit-level equations are combined and expressed as one system of equations. Newton's method is then applied to the complete system of equations. Unlike the two-

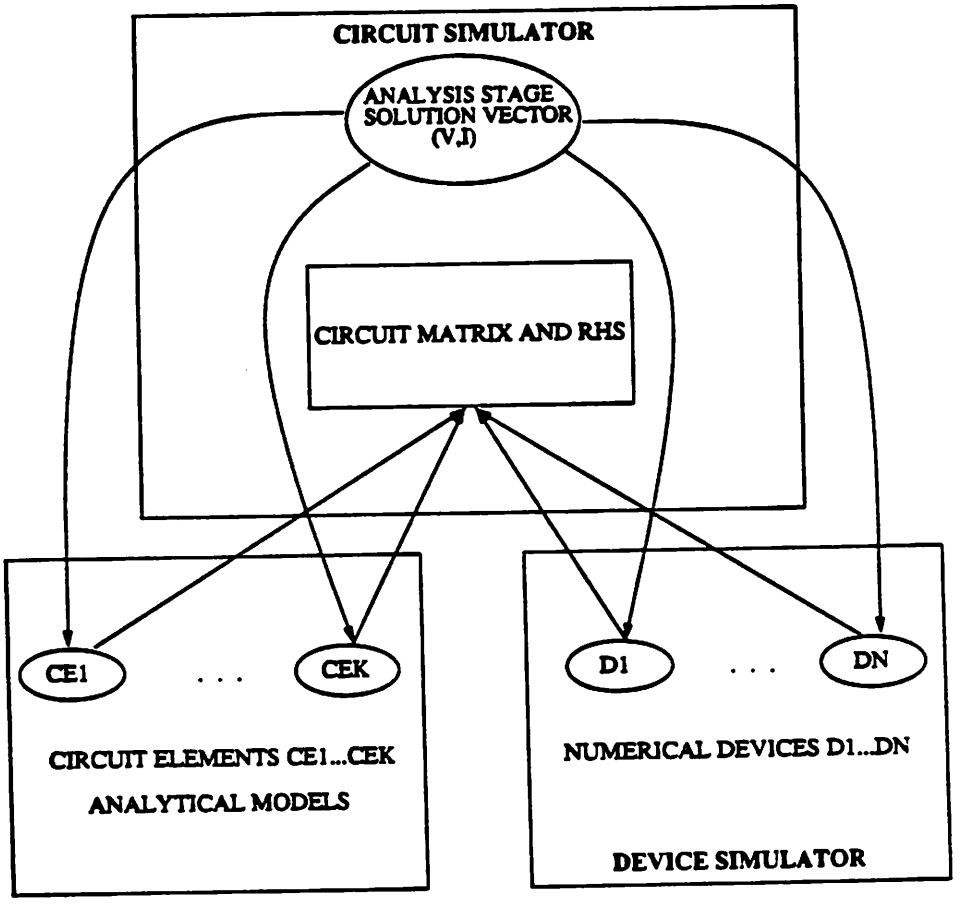


Figure 4.6: Architecture of CODECS. Numerical devices are interfaced with the circuit simulator in a manner similar to that for analytical devices. The circuit node voltages establish the boundary conditions for the numerical devices. The PDEs are solved by the device-level simulator of CODECS.

level Newton algorithm where the device and circuit-level unknowns are solved separately in a decoupled manner, the complete set of unknowns is solved simultaneously. For the circuit of Figure 4.1, the device-level equations are $F(\mathbf{w}, V) = \mathbf{O}$ and these are combined with KCL at the circuit level to get

$$\mathbf{F}(\mathbf{w}, V) = \mathbf{O} \tag{4.8}$$

$$I(\mathbf{w}, V) + G(V - E_s) = 0 \tag{4.9}$$

Equations (4.8) and (4.9) are solved using the Newton-Raphson method. The equations to be solved at each iteration of the Newton's method are then,

$$\mathbf{J}_w \Delta\mathbf{w} + \mathbf{J}_V \Delta V = -\mathbf{F}(\mathbf{w}, V) \tag{4.10}$$

$$\frac{\partial I}{\partial \mathbf{w}} \Delta\mathbf{w} + \frac{\partial I}{\partial V} \Delta V + G\Delta V = -I(\mathbf{w}, V) - G(V - E_s) \tag{4.11}$$

From Equation (4.10), $\Delta\mathbf{w}$ can be expressed as

$$\Delta\mathbf{w} = \mathbf{J}_w^{-1}(-\mathbf{F}(\mathbf{w}, V) - \mathbf{J}_V \Delta V) \tag{4.12}$$

Equation (4.12) can be rewritten as

$$\Delta\mathbf{w} = \Delta\hat{\mathbf{w}} - \mathbf{J}_w^{-1}\mathbf{J}_V \Delta V \tag{4.13}$$

where $\Delta\hat{\mathbf{w}} = \mathbf{J}_w^{-1}(-F(\mathbf{w}, V))$. Substituting $\Delta\mathbf{w}$ from Equation (4.13) into Equation (4.11), one obtains

$$\left[-\frac{\partial I}{\partial \mathbf{w}}\mathbf{J}_w^{-1}\mathbf{J}_V + \frac{\partial I}{\partial V} + G\right]\Delta V = -I(\mathbf{w}, V) - \frac{\partial I}{\partial \mathbf{w}}\Delta\hat{\mathbf{w}} - G(V - E_s) \tag{4.14}$$

This equation can be rewritten as

$$\left[G_{eq} + G\right]\Delta V = -I_T - G(V - E_s) \tag{4.15}$$

with $G_{eq} = -\frac{\partial I}{\partial \mathbf{w}}\mathbf{J}_w^{-1}\mathbf{J}_V + \frac{\partial I}{\partial V}$ and $I_T = I(\mathbf{w}, V) + \frac{\partial I}{\partial \mathbf{w}}\Delta\hat{\mathbf{w}}$. Equation (4.15) is similar in form to that obtained with an analytical model for the diode or by use of the two-

level Newton algorithm. Thus the above technique can also be used to embed numerical models within a circuit-simulation program. The full-Newton scheme can be implemented in two different ways.

4.2.4.1. Full LU-Decomposition Technique

$\mathbf{J}_w$ is decomposed into its LU factors and used to calculate $\Delta\hat{\mathbf{w}}$ and $\mathbf{J}_w^{-1}\mathbf{J}_V$ of Equation (4.13) by forward and back substitutions. Then G_{eq} and I_T are computed by matrix multiplications. Equation (4.15) is solved, whereby ΔV is obtained and $\Delta\mathbf{w}$ is calculated from Equation (4.13), using the previously computed values of $\Delta\hat{\mathbf{w}}$ and $\mathbf{J}_w^{-1}\mathbf{J}_V$. The equations are solved to convergence. This technique is similar to the use of block-LU decomposition with bordered-block-diagonal matrices in circuit simulation [4.4, 4.5].

4.2.4.2. MEDUSA's Block-Iterative Technique

$\mathbf{J}_w$ is decomposed into its LU factors; $\Delta\hat{\mathbf{w}}$, G_{eq} and I_T are calculated as above. Then ΔV is obtained from Equation (4.15), and $\Delta\mathbf{w}$ is assigned the value of $\Delta\hat{\mathbf{w}}$. This is equivalent to assuming $\Delta V = 0$ in equation Equation (4.13) and ignoring the coupling term due to ΔV. The equations are solved to convergence. This algorithm is used in MEDUSA [4.1].

4.2.5. Implementation Issues

The three algorithms described above have been implemented in the framework of CODECS shown in Figure 4.5. The interface to the circuit simulator is through a model-evaluation subroutine which calculates and loads the device contributions in the circuit-level equations. The subroutines differ according to the algorithm used but the essential features are identical. First, the new voltages are calculated and used to establish the boundary conditions for the device-level equations; then the device equations are

solved. This is followed by calculation of the terminal currents and conductances which are then loaded in the circuit Jacobian matrix and right-hand-side vector. The pseudo-C code for the three algorithms is shown below and illustrates the similarity between them. Furthermore, no algorithm has any significant advantage from an implementation point of view and all three techniques effectively decouple the device-level equations from the circuit-level equations. The function *setBoundaryConditions* is used to establish the boundary conditions for the device, the function *biasSolution* solves the device equations to convergence or the iteration limit *iterLimit*, whichever is reached first. An *iterLimit* value of one allows calculation of $\Delta\hat{\mathbf{w}}$ of Equation (4.13). The function *updateSolution* is used to calculate $\Delta\mathbf{w}$ from Equation (4.13) given $\Delta\hat{\mathbf{w}}$ and ΔV.

4.2.5.1. The Two-level Newton Algorithm

For the two-level Newton scheme, at each operating point the new terminal voltages (boundary conditions) are imposed on the device and a solution is obtained for the new bias conditions.

```
setBoundaryConditions( device );
biasSolution( device, iterLimit );
```

4.2.5.2. The Full LU-Decomposition Algorithm

For the full LU-decomposition scheme $\mathbf{J}_w^{-1}$ and $\Delta\hat{\mathbf{w}}$ are calculated by the device simulator. Then the circuit node voltages are calculated. $\Delta\mathbf{w}$ can be obtained only after the circuit-level equations have been solved since ΔV is required for its calculation. Thus, the following approach is used. At the completion of the device-level solution only $\Delta\hat{\mathbf{w}}$ is calculated and stored. Before starting the next device-level iteration, $\Delta\mathbf{w}$ is calculated from $\Delta\hat{\mathbf{w}}$ and ΔV, and $\mathbf{w}$ is updated. The new updated value of $\mathbf{w}$ is used for the next iteration. This sequence of operations allows a decoupling between the circuit and device simulators. At the first iteration of an operating point the terminal voltages are

imposed on the device. However, in the subsequent iterations the subroutine *updateSolution* is used to establish the new boundary conditions and to calculate Δw and hence the correct value of w to be used for the new iteration.

```
if( predictionStep ) {
   setBoundaryConditions( device );
} else {
   updateSolution( device );
}
biasSolution( device, 1 );
```

4.2.5.3. Medusa's Algorithm

Algorithm Med is similar to the two-level Newton scheme except that only one pass is made through the device-level equations for each circuit-level iteration and the calculation of conductances and currents is done in a different manner.

```
setBoundaryConditions( device );
biasSolution( device, 1 );
```

4.2.5.4. Modified Two-Level Newton Algorithm

From the above pseudo-C code it is seen that the two-level Newton algorithm and algorithm Med are similar. The difference is that in the two-level Newton scheme the device equations are solved to convergence with a Newton method. A modified 2-level Newton algorithm can also be used similar to the full LU-decomposition algorithm in which the device-level equations are solved to convergence. The pseudo-C code for this algorithm is given as

```
if( predictionStep ) {
   setBoundaryConditions( device );
} else {
   updateSolution( device );
}
biasSolution( device, iterLimit );
```

It is shown in Chapter 5 that the *updateSolution* step with a two-level Newton method provides a linear prediction of the solution at the new operating point. Some modifications described therein are necessary for the algorithm to work under dc conditions.

Four possible techniques to couple the device simulator to the circuit simulator have been described. These are the

(1) modified two-level Newton algorithm (M2lev),

(4) two-level Newton algorithm (2lev),

(3) full LU decomposition technique (FullLU), and

(4) block-iterative technique of MEDUSA (Med).

The algorithms are now evaluated on the basis of their convergence properties and run-time performance.

4.2.6. Convergence Properties for Dc Analysis.

The convergence properties of these algorithms has been examined by evaluating their performance on several benchmark circuits. A short summary of the circuits and the numerical models which are used is given in Table 4.1. CODECS input listings and the details of the numerical models are given in Appendix B.

In Table 4.2 the results for the dc operating point analysis of circuits with one-dimensional numerical models for the bipolar transistor are given. The results are given as the number of circuit-level iterations followed by the total simulation time. A '-' indicates that convergence was not achieved in 100 iterations.

As can be seen from Table 4.2 the modified two-level Newton scheme (M2lev) was always successful in finding an operating point, whereas the two-level Newton and full LU-decomposition schemes were only partially successful, and the MEDUSA

Circuit	# Nodes	# Circuit Elements	# Numerical Devices	Model Type	# Grid Points
RTLinv	4	4	1 BJT	1D	61
Oscillator	5	8	1 BJT	1D	61
VCO	7	10	6 BJT	1D	61
Invchain	10	10	4 BJT	1D	61
Astable	6	8	2 BJT	1D	61
MECLgate	26	24	11 BJT	1D	61
Pass	6	7	1 MOS	2D	31 x 19
MOSinv	5	5	1 MOS	2D	31 x 19
ChargePump	7	7	1 MOS	2D	21 x 21

Table 4.1: Description of benchmark circuits

Circuit	M2lev	2lev	FullLU	Med
RTLinv	8/5.0s	8/5.5s	8/2.42s	-
Oscillator	8/4.5s	8/4.9s	9/2.6s	-
VCO	8/25s	-	10/16s	-
Invchain	9/22s	-	-	-
Astable	9/11s	-	-	-
MECLgate	51/81s	51/94s	-	-

Table 4.2: Comparison of iterations and runtimes

algorithm failed the test in all of these cases. This has been found to be true on other examples as well. For this reason CODECS uses the modified two-level Newton scheme for dc operating point analysis. The modified two-level Newton scheme is computationally expensive compared to full-LU decomposition but is preferred for dc analysis since it has worked well over a wide variety of examples.

4.2.7. Transient Analysis Comparisons

The transient simulation problem is better conditioned than the problem of simulating the dc operating point; hence, the algorithm that works best for simulation of the dc operating point may not be optimal for transient analysis. In this section the four algorithms are compared on the basis of their performance for transient simulations. The simulations are started with the dc operating point of the circuit being obtained by the modified two-level Newton algorithm. All simulations have been run with a latency check that is described in Chapter 5. The second order backward-differentiation formula [4.6] is used for time discretization. A starred entry indicates that the simulation did not complete successfully due to a "timestep too small" error, and the result is reported with the latency check turned off. In Table 4.3, the number of circuit iterations are presented for transient analysis of the benchmark circuits.

Circuit	M2lev	2lev	FullLU	Med
Oscillator	16916	16916	18333	23836
VCO	5093	5109	5864	7028
Invchain	1563	1578	1716	2324
Astable	5930	6305	6369	9087
MECLgate	2450	2450	2609	3236*
Pass	236	236	295	338
MOSinv	287	313	336	533
ChargePump	1644	1661	1850	2661

Table 4.3: Comparison of number of circuit iterations

The modified two-level Newton algorithm requires the smallest number of circuit-level iterations. The two-level Newton algorithm requires ten percent more iterations, in some examples, and the full-LU technique takes approximately twenty-five percent more iterations compared to the modified two-level Newton method. In all examples

algorithm Med takes the largest number of circuit iterations and in the MECL-gate example the simulation could only be performed by turning off the latency check.

In Table 4.4 are presented the number of timepoints that were accepted and rejected during the transient analysis. It is seen that the timepoints accepted and rejected are of the same order.

Circuit	M2lev	2lev	FullLU	Med
Oscillator	5274 / 366	5274 / 366	5274 / 361	5274 / 361
VCO	1099 / 161	1106 / 161	1125 / 176	1093 / 164
Invchain	401 / 17	404 / 18	401 / 17	410 /20
Astable	1473 / 198	1583 / 234	1465 / 181	1554 / 219
MECLgate	619 / 30	619 / 30	619 / 31	619 / 31*
Pass	82 / 8	82 / 8	82 / 8	82 / 8
MOSinv	95 / 4	104 / 8	95 / 4	104 / 8
ChargePump	497 / 74	497 / 74	493 / 73	492 / 75

Table 4.4: Comparison of timepoints accepted and rejected

The simulation runtimes are presented in Table 4.5. The full LU-decomposition scheme takes the smallest amount of time. The modified two-level Newton scheme, on an average, is a factor of 1.7 slower than the full LU-decomposition scheme and in some cases does even better than algorithm Med. This might appear surprising at first because the modified two-level Newton scheme requires more CPU time for each circuit-level iteration. However, as seen from Table 4.6, a fewer number of circuit-level iterations are required at each timepoint. Algorithm Med has no apparent advantage; it requires more computational effort and does not work well with the latency check.

Circuit	M2lev	2lev	FullLU	Med
Oscillator	3126	3636	2352	3123
VCO	4805	5440	2911	3901
Invchain	890	965	514	806
Astable	2085	2538	1230	2031
MECLgate	3629	3931	2121	4577*
Pass	1786	1955	1059	1235
MOSinv	1626	2194	1155	1800
ChargePump	7172	8045	4039	5910

Table 4.5: Comparison of total analysis time

Circuit	M2lev	2lev	FullLU	Med
Oscillator	3.0	3.0	3.3	4.2
VCO	4.0	4.0	4.5	5.6
Invchain	3.7	3.7	4.1	5.4
Astable	3.5	3.5	3.9	5.1
MECLgate	3.8	3.8	4.0	5.0*
Pass	2.6	2.6	3.3	3.8
MOSinv	2.9	2.8	3.4	4.8
ChargePump	2.9	2.9	3.3	4.7

Table 4.6: Comparison of iterations per timepoint

The four algorithms can also be compared on a time per iteration basis. This gives the raw speed of each algorithm. It is seen from Table 4.7 that on a time per iteration basis the modified two-level Newton scheme is marginally better than the two-level Newton scheme in all examples, and it is significantly better in the MOSinv example. However, the two-level Newton schemes are approximately a factor of two slower than the full LU scheme and algorithm Med. On a time per iteration basis, algorithm Med

and the full LU-decomposition scheme are almost identical except in the MECL gate example. The full-LU scheme requires fewer iterations per timepoint than algorithm Med and is preferable since it results in an overall smaller simulation time.

Circuit	M2lev	2lev	FullLU	Med
Oscillator	0.18	0.21	0.13	0.13
VCO	0.94	1.06	0.50	0.56
Invchain	0.57	0.61	0.30	0.35
Astable	0.35	0.40	0.19	0.22
MECLgate	1.48	1.60	0.81	1.41
Pass	7.60	8.30	3.60	3.65
MOSinv	5.67	7.00	3.49	3.38
ChargePump	4.36	4.84	2.18	2.22

Table 4.7: Comparison of time per iteration

4.2.8. Conclusions on Performance, Convergence, and Memory Requirements

Theoretical results suggest that the two-level Newton and the full-Newton schemes have local quadratic convergence. Algorithm Med has been shown to have superlinear convergence [4.1]. The practical implementations are in agreement with the theoretical results; the previous results indicate the higher rate of convergence for the two-level and full-Newton schemes. However, for dc analysis the modified two-level Newton algorithm is better in a practical implementation. It has better convergence than the full-Newton scheme.

A possible explanation for this is the strong decoupling between the circuit and the device-level equations provided by the two-level Newton scheme. In transient analysis the matrices are better conditioned and all schemes can be used. However, the full LU-decomposition algorithm requires the smallest amount of simulation time and is,

therefore, most suitable for transient analysis. The modified two-level Newton scheme could also be used for transient analysis but there is a speed penalty of a factor of two compared to the full LU-decomposition technique. From a performance and convergence point of view, MEDUSA's algorithm has no advantage.

An argument in favor of the MEDUSA algorithm is a smaller memory requirement [4.1] since the coupling terms in Equation (4.13) are not stored. This, however, leads to a slower convergence. For a mixed-level device and circuit-simulation environment with capabilities to simulate devices with at most four terminals (this is typical of semiconductor devices), the coupling matrix can be stored as three vectors. In CODECS this has been done by temporarily using memory locations that have been allocated for the right-hand-side vector, the solution vector, and a scratch vector at the device level. Thus, there is no additional memory requirement in CODECS when implementing the full LU-decomposition scheme. Furthermore, a significant improvement in performance is also achieved. It should be noted that the two-level Newton algorithm requires the same amount of memory as the MEDUSA algorithm, whereas the modified two-level Newton method and the full-Newton method are identical in terms of memory requirements.

4.2.9. Parallelization Issues

All four algorithms provide a decoupling between the circuit equations and the device equations. This decoupling can be used to advantage in a multiprocessing computing environment. Once again no particular algorithm has an edge over the others in terms of having a higher degree of parallelism.

The task of model evaluation with analytical models has been found to be highly parallelizable [4.7]. In parallel model evaluation each device is spawned off to a different processor and the calculation of conductances and currents takes place in parallel. The same approach can be extended to numerical devices. Here the model evaluation task

involves solution of the PDEs, thus each processor would solve the device equations in parallel with other processors. One could then expect an almost linear speedup with the number of processors, similar to what has been achieved with the use of analytical models [4.7].

4.3. Small-Signal Ac Analysis

Ac analysis is useful for analog circuit simulations. Since CODECS is intended to be a general-purpose coupled device and circuit simulator, it also provides a capability for small-signal ac analysis. Alternatively, one could run a transient simulation and extract the frequency-domain response using Fourier transform techniques. However, this approach is computationally expensive and ac analysis provides a good way of obtaining the small-signal frequency-domain response.

The ac admittances for each device have to be computed and loaded in the linear circuit-level equations. The admittances are functions of frequency, and at a particular frequency, ω, the solution of the algebraic circuit-level equations gives the small-signal circuit node voltages and voltage source currents. For analytical device models the admittances are calculated around an operating point by function evaluations. For a numerical device the admittances can be calculated at the frequency ω by solving the small-signal device-level equations as described in Chapter 3. The solution of the device-level equations gives the small-signal ac values of the internal variables, the electrostatic potential, and the carrier concentrations at each spatial grid point. From this information the ac admittances for a numerical device can be calculated and then used in the circuit-level equations.

4.3.1. Calculation of Ac Admittances

The small-signal ac terminal current $\tilde{I}$ for a device can be expressed as

$$\tilde{I}(\omega) = \tilde{I}(\tilde{\mathbf{w}}(\omega), \tilde{V}, \omega) \tag{4.16}$$

where $\tilde{\mathbf{w}}$ is the vector of small-signal values of the electrostatic potential and electron and hole concentrations, $\tilde{V}$ is the applied small-signal voltage, and ω is the radian frequency. The explicit dependence on ω, in Equation (4.16), is through the displacement current component of the total current. The small-signal ac admittance is given by

$$Y(\omega) = \frac{\tilde{I}(\omega)}{\tilde{V}} \tag{4.17}$$

If $\tilde{V}$ is taken to be unity

$$Y(\omega) = \tilde{I}(\omega) \tag{4.18}$$

The small-signal ac current is given by the linear term of the Taylor series expansion of $I = I(\mathbf{w}, V)$ around the operating point $(\mathbf{w}_o, V_o)$. Therefore,

$$Y(\omega) = \tilde{I}(\omega) = \frac{\partial I}{\partial \mathbf{w}}\tilde{\mathbf{w}} + \frac{\partial I}{\partial V} \tag{4.19}$$

where $\frac{\partial I}{\partial \mathbf{w}}$ and $\frac{\partial I}{\partial V}$ are evaluated at the dc operating point by use of symbolic differentiation, and $\tilde{\mathbf{w}}$ is calculated as described in Chapter 3.

4.4. Pole-Zero analysis

Pole-zero analysis requires computation of admittances for a device as a function of the complex frequency $s = \sigma + j\omega$. The technique used to calculate the admittance for a numerical device is an extension of the method used for small-signal ac analysis. Instead of using a frequency ω, the complex frequency s is used and the admittances are expressed as $Y(s)$. Given a value for s, $Y(s)$ can be calculated and used in the circuit-level equations. The circuit-level transfer function can then be computed and its poles

and zeros can be determined.

For numerical devices $Y(s)$ is computed starting from the basic device equations. As in Chapter 3, the unknowns are assumed to be of the form

$$\mathbf{w} = \mathbf{w}_o + \tilde{\mathbf{w}}e^{st} \tag{4.20}$$

Using a Taylor series expansion around an operating point and retaining the linear terms, one can assemble the device-level equations in the form (similar to the equations for small-signal ac analysis given in Chapter 3)

$$\left[\mathbf{J}_w + D\right]\tilde{\mathbf{w}} = \mathbf{B} \tag{4.21}$$

where $\mathbf{J}_w$ is the dc Jacobian matrix of the device-level equations, **D** is a diagonal matrix with entries 0 corresponding to Poisson's equation, $-s$ corresponding to the electron current-continuity equation, and s corresponding to the hole current-continuity equation, $\tilde{\mathbf{w}}$ is the vector of small-signal values of the electrostatic potential, and electron and hole concentrations, and **B** is the right-hand-side vector that accounts for the boundary conditions.

The above equations are solved for $\tilde{\mathbf{w}}(s)$ by a direct-solution method and then $Y(s)$ is computed by

$$Y(s) = \frac{\partial I}{\partial \mathbf{w}}\tilde{\mathbf{w}}(s) + \frac{\partial I}{\partial V} \tag{4.22}$$

where $\frac{\partial I}{\partial \mathbf{w}}$ and $\frac{\partial I}{\partial V}$ are computed by symbolic differentiation.

4.5. Requirements on Circuit and Device Simulators

The architecture described earlier mandates that the circuit and device simulators possess several basic characteristics. The circuit simulator should allow easy incorporation of new device models since this capability is used to embed the numerical devices

within the circuit simulator. In addition all the necessary analysis capabilities must be supported at the circuit level of simulation. SPICE3 [4.8] is modular and allows incorporation of new devices and models. Furthermore, it supports all the important analysis capabilities at the circuit level. This makes SPICE3 attractive for developing a mixed-level circuit and device-simulation environment.

The device simulator must be quite general. It should have capabilities for simulating one- and two-dimensional structures and support the various analyses at the device level of simulation. Furthermore, it should be able to simulate multiple devices as opposed to simulating a single device. Most present day device simulators are structured to simulate only one device which makes them unsuitable for use in a mixed-level simulation framework as shown in Fig 4.6. The device simulator should also provide subroutines that compute the terminal currents and conductances of a numerical device for given terminal voltages. Since many of the above features are not available in existing device simulators, a new device-level simulator has been developed for CODECS.

CHAPTER 5

Device-Level Algorithms of CODECS

5.1. Introduction

The algorithms used for one- and two-dimensional numerical device simulation for dc, transient, small-signal ac and pole-zero, and sensitivity analyses are described in this chapter. The algorithms at the circuit level are those of SPICE and are not included; they can be found in [5.1, 5.2].

The first part of this chapter describes the space discretization used in CODECS for one- and two-dimensional devices. This is followed by a description of the base boundary condition for a one-dimensional bipolar transistor. The traditional way of implementing the base boundary condition in a one-dimensional bipolar transistor leads to nonconvergence under negative base-emitter voltages. For this reason a different formulation has been used which overcomes the problem and in addition provides a more physical representation of the one-dimensional bipolar transistor.

Dc analysis is important for finding the steady-state solution of a circuit. The dc operating point information is necessary for small-signal ac analysis and frequently used as the starting point for transient analysis. However, in the mixed-level environment dc convergence is a serious problem. The dc convergence problem is examined and approaches to improve convergence in CODECS are described. These include use of a modified two-level Newton scheme with a norm-reducing Newton's method at the device level. Device-based limiting along with a voltage-step backtracking scheme have been

found to give reasonable convergence.

Transient analysis is extremely important in determining the dynamic response of a circuit. Section 5.4 describes the integration formulae used and the calculation of local error for control of timesteps and integration order. A comparison is made between two different schemes for timestep and order control based on accuracy and performance. Latency at the device level should be exploited since the device-level simulation requires a large amount of computational time. As seen in Chapter 4 a full-Newton algorithm performs the best for transient analysis; hence, the implementation of latency check for the full-Newton algorithm is described. The performance with the latency check is compared with the run times without latency exploitation. Finally, the current-conservation property of the integration formulae has also been examined.

The algorithms for ac and pole-zero analyses are an extension of the technique for small-signal ac analysis described in Chapter 3. Some implementation issues are presented.

The last part of this chapter is devoted to the calculation of sensitivities at the device level. This results in a capability whereby the sensitivity of device performance to process parameters can be determined. Although the effect of process variations can be determined by repeated simulations, they are computationally extremely expensive. Sensitivity calculations are a good alternative. The sensitivity problem is formulated for both dc and transient analyses and the implementation details are described. Examples of sensitivities to doping profile variations are also presented.

5.2. Space Discretization in Two Dimensions

The fundamental semiconductor equations are Poisson's equation and the current-continuity equations. These are given in normalized form (Chapter 3) by

$$\nabla.\mathbf{E} = (N_D - N_A + p - n) \tag{5.1a}$$

$$\nabla.\mathbf{J}_n = \frac{\partial n}{\partial t} - (G - R) \tag{5.1b}$$

$$\nabla.\mathbf{J}_p = -\frac{\partial p}{\partial t} + (G - R) \tag{5.1c}$$

where

$$\mathbf{E} = -\nabla\psi \tag{5.2a}$$

$$\mathbf{J}_n = -\mu_n n \nabla\psi + D_n \nabla n \tag{5.2b}$$

$$\mathbf{J}_p = -\mu_p p \nabla\psi - D_p \nabla p \tag{5.2c}$$

In CODECS the above system of equations is discretized in space by the *Box Integration Method* [5.3]. For a rectangular simulation domain, a rectangular mesh M is defined as

$$M = \left\{ (x_i, y_j) \mid 1 \le i \le N_x + 1,\ 1 \le j \le N_y + 1 \right\} \tag{5.3}$$

where there are $N_x + 1$ grid lines in the x direction and $N_y + 1$ grid lines in the y direction resulting in a total of $(N_x + 1)\times(N_y + 1)$ grid points in the simulation domain. The device equations can be expressed in a general form as

$$\nabla.\mathbf{F} = r \tag{5.4}$$

where $\mathbf{F}$ is a vector-valued function, $\mathbf{F} = (f_x, f_y)$, and r is the right-hand side of the equation. From Green's theorem [5.4]

$$\iint_R \nabla.\mathbf{F}\,dxdy = \int_C f_x dy - f_y dx \tag{5.5}$$

where R is a region in the x–y plane and C is the contour that encloses the region R. The mesh spacings in the x and y directions are defined to be $h_i = x_{i+1} - x_i$ and $k_j = y_{j+1} - y_j$, respectively.

Consider an internal grid point (i, j) located at (x_i, y_j) such that $1 < i < N_x + 1$ and $1 < j < N_y + 1$. For such a grid point there are four neighboring grid points as shown in Figure 5.1. Region R is defined by the rectangle shown in dashed

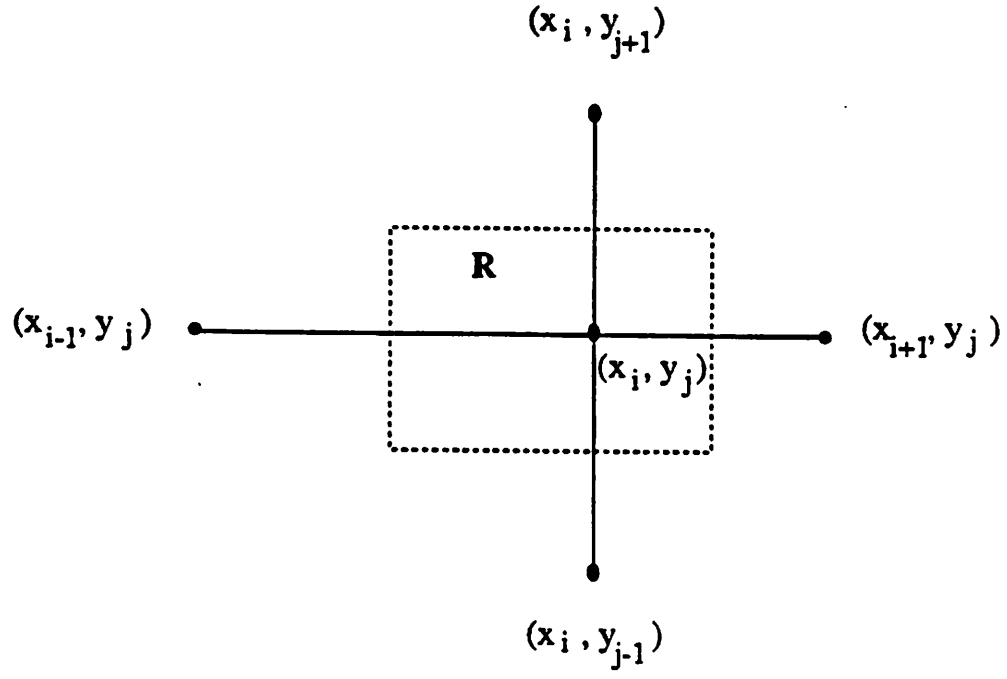


Figure 5.1: An internal grid point with four neighboring nodes.

lines; the edges of the rectangle intersect the respective grid lines at their mid points. This choice of R allows the complete rectangular domain to be covered by non-overlapping rectangles. For grid point (i, j) Equation (5.5) can be written as

$$\int_C f_x dy - f_y dx = \int_{x_{i-1/2}}^{x_{i+1/2}} -f_y(x, y_{j-1/2})dx + \int_{y_{j-1/2}}^{y_{j+1/2}} f_x(x_{i+1/2}, y)dy +$$

$$\int_{x_{i+1/2}}^{x_{i-1/2}} -f_y(x, y_{j+1/2})dx + \int_{y_{j+1/2}}^{y_{j-1/2}} f_x(x_{i-1/2}, y)dy \qquad (5.6)$$

The four integrals are approximated in the following manner:

$$\int_{x_{i-1/2}}^{x_{i+1/2}} -f_y(x, y_{j-1/2})dx = -f_y(x_i, y_{j-1/2})\frac{h_i + h_{i-1}}{2} \tag{5.7a}$$

$$\int_{y_{j-1/2}}^{y_{j+1/2}} f_x(x_{i+1/2}, y)dy = f_x(x_{i+1/2}, y_j)\frac{k_j + k_{j-1}}{2} \tag{5.7b}$$

$$\int_{x_{i+1/2}}^{x_{i-1/2}} -f_y(x, y_{j+1/2})dx = f_y(x_i, y_{j+1/2})\frac{h_i+h_{i-1}}{2} \tag{5.7c}$$

$$\int_{y_{j+1/2}}^{y_{j-1/2}} f_x(x_{i-1/2}, y)dy = -f_x(x_{i-1/2}, y_j)\frac{k_j + k_{j-1}}{2} \tag{5.7d}$$

The right-hand side is approximated by assuming that r is a constant over the rectangle at the value r_{ij}, then

$$\iint_R r dx dy = r_{ij}\iint_R dx dy = r_{ij} \times \text{Area of rectangle}$$

$$= r_{ij}\left[\frac{h_i + h_{i-1}}{2}\right]\left[\frac{k_j + k_{j-1}}{2}\right] \tag{5.8}$$

With the above approximations the space discretization of Equation (5.4) can be expressed as

$$-f_y(x_i, y_{j-1/2})\frac{h_i + h_{i-1}}{2} + f_x(x_{i+1/2}, y_j)\frac{k_j + k_{j-1}}{2} + f_y(x_i, y_{j+1/2})\frac{h_i + h_{i-1}}{2}$$

$$- f_x(x_{i-1/2}, y_j)\frac{k_j + k_{j-1}}{2} = r_{ij}\left[\frac{h_i + h_{i-1}}{2}\right]\left[\frac{k_j + k_{j-1}}{2}\right] \tag{5.9}$$

Equation (5.9) can be used to write the discretized semiconductor equations,

$$-E_y|_{i,j-1/2}\frac{h_i + h_{i-1}}{2} + E_x|_{i+1/2,j}\frac{k_j + k_{j-1}}{2} + E_y|_{i,j+1/2}\frac{h_i + h_{i-1}}{2}$$

$$- E_x|_{i-1/2,j}\frac{k_j + k_{j-1}}{2} = (N_{ij} + p_{ij} - n_{ij})\left[\frac{h_i + h_{i-1}}{2}\right]\left[\frac{k_j + k_{j-1}}{2}\right] \tag{5.10a}$$

$$-J_{ny}|_{i,j-1/2}\frac{h_i + h_{i-1}}{2} + J_{nx}|_{i+1/2,j}\frac{k_j + k_{j-1}}{2} + J_{ny}|_{i,j+1/2}\frac{h_i + h_{i-1}}{2}$$

$$- J_{nx}|_{i-1/2,j}\frac{k_j + k_{j-1}}{2} = \left[\left[\frac{\partial n}{\partial t}\right]_{ij} - (G - R)_{ij}\right]\left[\frac{h_i + h_{i-1}}{2}\right]\left[\frac{k_j + k_{j-1}}{2}\right] \qquad (5.10b)$$

$$-J_{py}|_{i,j-1/2}\frac{h_i + h_{i-1}}{2} + J_{px}|_{i+1/2,j}\frac{k_j + k_{j-1}}{2} + J_{py}|_{i,j+1/2}\frac{h_i + h_{i-1}}{2}$$

$$- J_{px}|_{i-1/2,j}\frac{k_j + k_{j-1}}{2} = \left[\left[-\frac{\partial p}{\partial t}\right]_{ij} + (G - R)_{ij}\right]\left[\frac{h_i + h_{i-1}}{2}\right]\left[\frac{k_j + k_{j-1}}{2}\right] \qquad (5.10c)$$

For a grid node where a homogeneous Neumann boundary condition is applied either $f_x = 0$ or $f_y = 0$ on the boundary. In such a situation one edge of the rectangle R is the boundary segment as shown in Figure 5.2 and the discretized equation is given by

$$-f_y(x_i, y_{j-1/2})\frac{h_{i-1}}{2} + f_y(x_i, y_{j+1/2})\frac{h_{i-1}}{2}$$

$$- f_x(x_{i-1/2}, y_j)\frac{k_j + k_{j-1}}{2} = r_{ij}\frac{h_{i-1}}{2}\left[\frac{k_j + k_{j-1}}{2}\right] \qquad (5.11)$$

where the right-hand side is given by $r_{ij} \times$ Area of rectangle.

For a node with Dirichlet boundary conditions, an ohmic-contact node, no equations have to be solved since all the quantities ψ, n, and p at that grid node are known from the equilibrium solution and the applied voltage.

Grid points that are on the silicon-oxide interface have to be treated in a different manner. Such a situation is shown in Figure 5.3; in this case two rectangles are considered, one above and and the other below the interface line. The box integration method is then used on each of these rectangles. The contributions along the common edge of the two rectangles cancel and the discretized equation is

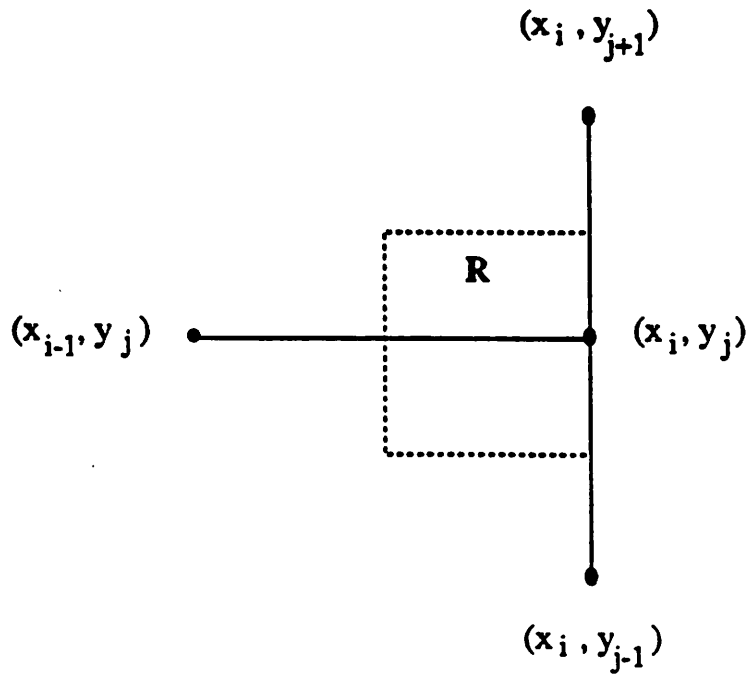


Figure 5.2: A grid node on the boundary segment.

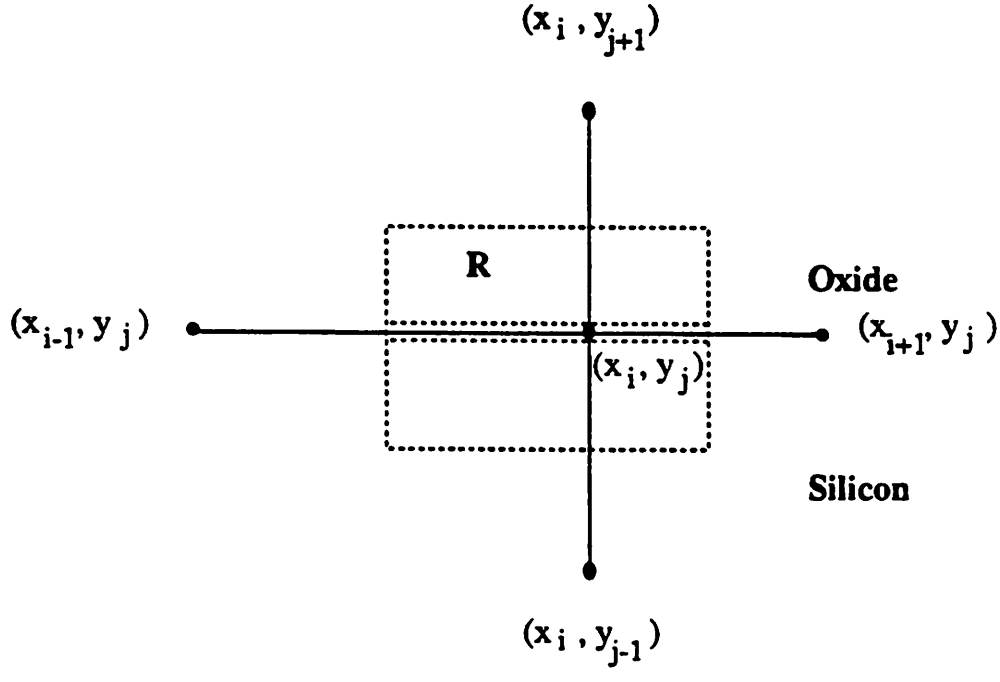


Figure 5.3: A grid node on the silicon-oxide interface.

$$\left[-f_y(x_i, y_{j-1/2})\right]_{silicon}\frac{h_i + h_{i-1}}{2} + \left[f_x(x_{i+1/2}, y_j)\right]_{silicon}\frac{k_{j-1}}{2} + \left[f_x(x_{i+1/2}, y_j)\right]_{oxide}\frac{k_j}{2}$$

$$+ \left[f_y(x_i, y_{j+1/2})\right]_{oxide}\frac{h_i + h_{i-1}}{2} - \left[f_x(x_{i-1/2}, y_j)\right]_{oxide}\frac{k_j}{2} - \left[f_x(x_{i-1/2}, y_j)\right]_{silicon}\frac{k_{j-1}}{2}$$

$$= \frac{k_{j-1}}{2}\left[\frac{h_i + h_{i-1}}{2}\right]\left[r_{ij}\right]_{silicon} + \frac{k_j}{2}\left[\frac{h_i + h_{i-1}}{2}\right]\left[r_{ij}\right]_{oxide} \tag{5.12}$$

The above scheme is used for discretizing the Poisson's equation at the interface grid nodes. Since there is no charge in the oxide region, $\left[r_{ij}\right]_{oxide} = 0$. Furthermore, f_y is continuous at the interface if there are no interface charges. Therefore,

$$\left[\frac{\partial \psi}{\partial y}\right]_{ij,silicon} = \varepsilon_r \left[\frac{\partial \psi}{\partial y}\right]_{ij,oxide}$$

where $\varepsilon_r = \frac{\varepsilon_{ox}}{\varepsilon_s}$.

As seen from Equation (5.10) $E_{x,y}$, $J_{nx,y}$ and $J_{px,y}$ involve derivatives of the variables ψ, n and p. Approximations have to be made to obtain values of the electric fields and current densities at the midpoints of each edge since they are required in Equation (5.10). To calculate the electric field values at the midpoint of each grid line, the potential is assumed to vary linearly between grid points, i.e., the electric field is a constant between two grid points and is given by

$$E_x\,|_{i+1/2,j} = -\frac{\psi_{i+1,j} - \psi_{i,j}}{h_i} \tag{5.13a}$$

For the current densities at the midpoints use is made of the Scharfetter-Gummel discretization, which also makes use of the fact that the electric field is constant between two grid points. Therefore,

$$J_{nx}\,|_{i+1/2,j} = \frac{\mu_n\,|_{i+1/2,j}}{h_i}\left[n_{i+1,j}B(\psi_{i+1,j} - \psi_{i,j}) - n_{i,j}B(-(\psi_{i+1,j} - \psi_{i,j}))\right] \tag{5.13b}$$

$$J_{px}|_{i+1/2,j} = \frac{\mu_p|_{i+1/2,j}}{h_i}\left[p_{i,j}B(\psi_{i+1,j} - \psi_{i,j}) - p_{i+1,j}B(-(\psi_{i+1,j} - \psi_{i,j}))\right] \tag{5.13c}$$

where $B(x) = \dfrac{x}{e^x - 1}$ is the Bernoulli's function. As indicated in [5.5], Bernoulli's function has to be evaluated with care and CODECS makes use of the method suggested in [5.5].

5.3. Space Discretization in One Dimension

Analogous to the space discretization for two dimensions, the following discretizations can be obtained for the semiconductor equations in one space dimension. For grid node i located at x_i, the equations are

$$\frac{\psi_{i+1} - \psi_i}{h_i} - \frac{\psi_i - \psi_{i-1}}{h_{i-1}} = -\left[N + p - n\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.14a}$$

$$J_{n,i+1/2} - J_{n,i-1/2} = \left[\frac{\partial n}{\partial t} - (G - R)\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.14b}$$

$$J_{p,i+1/2} - J_{p,i-1/2} = \left[-\frac{\partial p}{\partial t} + (G - R)\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.14c}$$

5.3.1. Base Boundary Condition for One-Dimensional Bipolar Transistor

The traditional way of setting the base boundary conditions for a one-dimensional bipolar transistor treats the base node as a point contact. To account for the lateral base current flow in a bipolar transistor a two-dimensional structure must be simulated. However, it is also possible to do one-dimensional simulations by using a special boundary condition for the base node. The quasi-Fermi level of the the majority carriers at the base contact is assumed to be equal to the applied base-emitter voltage [5.6]. This condition results in a discontinuity in the current at the base node and the discontinuity equals

the base current [5.7]. The discontinuity in current is inevitable for a one-dimensional model since lateral and vertical current flows cannot be modeled simultaneously in one dimension.

With the use of the above boundary condition for an *npn* transistor, the hole current-continuity equation (holes are the majority carriers) is replaced by

$$p = n_{ie} \exp\left[\frac{q(V_{BE} - \psi)}{kT}\right] \tag{5.15}$$

where V_{BE} is the base-emitter voltage. This approach leads to nonconvergence during transient analysis, particularly so for negative values of V_{BE}. The hole-current continuity equation is not solved; hence, the rate of change of holes with time is not monitored by the timestep-control scheme described later in this chapter. For this reason CODECS makes use of a different formulation for the base boundary condition. The hole current-continuity equation is used, thereby providing a more physical representation of the one-dimensional bipolar transistor.

The discretized equations at the base node B, located at $x = x_B$, for an *npn* transistor can be expressed as

$$\frac{\psi_{B+1} - \psi_B}{h_B} - \frac{\psi_B - \psi_{B-1}}{h_{B-1}} = -\left[N + p - n\right]_B \frac{h_B + h_{B-1}}{2} \tag{5.16a}$$

$$J_{n,B+1/2} - J_{n,B-1/2} = \left[\frac{\partial n}{\partial t} - (G - R)\right]_B \frac{h_B + h_{B-1}}{2} \tag{5.16b}$$

$$J_{p,B+1/2} - J_{p,B-1/2} = \left[-\frac{\partial p}{\partial t} + (G - R)\right]_B \frac{h_B + h_{B-1}}{2} + J_B \tag{5.16c}$$

where J_B is the lateral base-current density. Equation (5.16c) states that the lateral current injected into the base contact is like a generation term. This is similar to the approach of [5.8] where the base current is taken to be a generation term over the whole base region. The hole current-continuity equation expresses the boundary condition at the

base node in terms of an injected current. The above system of equations could be used except that with a current boundary condition convergence is harder to achieve [5.9]. For this reason J_B has to be modeled in terms of the applied base-emitter voltage and this is done by a physical understanding of the operation of a bipolar transistor.

Consider the T-structure shown in Figure 5.4, which represents

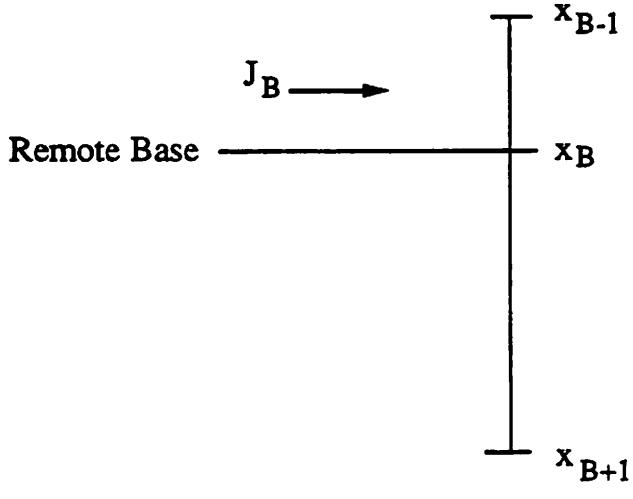


Figure 5.4: A T-representation for the one-dimensional bipolar transistor.

the simplest approximation to the two-dimensional nature of the bipolar transistor. The base current is injected at a remote base contact instead of x_B. Since the base current is due to the flow of majority carriers (holes for an *npn* transistor), J_B (in normalized form) can be expressed as [5.10]

$$J_B = J_{py} = -p\mu_p \frac{\partial \phi_p(x, y)}{\partial y} \tag{5.17}$$

where $\phi_p(x, y)$ is the quasi-Fermi level of the majority carriers. Assume that $\phi_p(x, y)$ is fairly constant with x in the base region. This can be expected since the majority carriers

in the base flow laterally. Furthermore, this has been shown to be a reasonable assumption by use of two-dimensional numerical simulations [5.11]. Thus,

$$\phi_p(x, y) \approx \phi_p(y) \tag{5.18}$$

and

$$J_B = -p\mu_p \frac{\partial \phi_p(y)}{\partial y} \tag{5.19}$$

Equation (5.19) is now discretized as

$$J_B = -p_{j+1/2}\mu_p \mid_{j+1/2} \frac{\phi_{p,j+1} - \phi_{p,j}}{\Delta y_j} \tag{5.20}$$

With the assumption $p_{j+1/2}\mu_p \mid_{j+1/2} \approx p_B \mu_{p,B}$, the base current can be expressed as

$$J_B = -p_B \mu_{p,B} \frac{\phi_{p,B} - \phi_{p,j}}{\Delta y_j} \tag{5.21}$$

At the remote base contact $\phi_p = V_{BE}$, then

$$J_B = p_B \mu_{p,B} \frac{V_{BE} - \phi_{p,B}}{\Delta y_j} \tag{5.22}$$

In normalized form $p = n_{ie} e^{\phi_{p,B} - \psi_B}$; therefore, J_B can be written as

$$J_B = p_B \mu_{p,B} \frac{V_{BE} - \psi_{p,B} - \ln(p_B / n_{ie})}{\Delta y} \tag{5.23}$$

where Δy is the distance of the internal base node from the remote base contact and can be used to model the base resistance of the bipolar transistor. It should be noted that the base resistance is conductivity modulated since p_B appears in the expression for J_B. Thus Equation (5.23) provides a physical representation of the bipolar transistor and is used in CODECS. Δy is a user specified parameter and has a default value of Δx_B. The above scheme exhibits no convergence problems in all transient one-dimensional bipolar transistor simulations that have been performed with CODECS.

5.4. Dc Analysis

At the device level dc analysis involves the solution of the device equations for an applied terminal voltage V. The device equations after space discretization can be expressed as

$$F(\mathbf{w}(V), V) = \mathbf{O} \tag{5.24}$$

where $\mathbf{w}(V)$ is the vector of internal variables corresponding to the applied bias. A solution of these highly nonlinear equations by iterative methods is prone to severe convergence problems. Newton's method is guaranteed to converge only when the initial guess is close to the solution. However, if the initial guess is far from the final solution then nothing can be said about the convergence of Newton's method. In order to alleviate the convergence problem, several modifications and heuristics have to be used along with the classical Newton's method. This section describes the enhancements used in CODECS to obtain reasonable convergence under dc conditions.

5.4.1. Device-Based Limiting Scheme

Consider the problem of tracing out the dc current-voltage characteristics of a semiconductor device by use of device-level simulation. The change in the terminal voltages from one bias point to the next must be small, otherwise convergence is not possible in the device simulation. In typical device-level simulations the user specifies the voltage increments to be used for generating the dc characteristics. In case of nonconvergence the user refines the voltage increments to be smaller than those used previously. Thus, by trial and error and by the use of small increments the user can obtain the solution. This corresponds to a source-stepping scheme [5.1], the stepping increments for which are provided by the user.

In a mixed-level circuit and device simulator, nonconvergence at the device level manifests itself as a serious problem. When the devices are part of a circuit iterative loop, the terminal voltages may change by quite a significant amount from iteration to iteration, particularly so in the first few circuit-level iterations when the iterates are far from the solution. Applying these circuit node voltages as boundary conditions to a device will cause nonconvergence in almost all cases. Thus it is essential to limit the per iteration change of the terminal voltages. The limiting schemes are based on physical operation of a device and are therefore referred to as device-based limiting techniques.

Device-based limiting is also required with analytical models [5.1] to restrict the per iteration change in terminal voltages, to avoid overflow problems, and to guide Newton's method into its region of convergence. These limiting schemes are derived from the current-voltage characteristics of the device, such as *pnjlim* the subroutine for limiting pn-junction voltages in SPICE. However, a similar voltage-limiting scheme cannot be used for limiting the voltage change across pn-junctions modeled numerically, since the current-voltage characteristics are not known a priori. CODECS relies on a simple limiting scheme similar in flavor to that used in [5.12].

For the pn-junction it is known that under low-level-injection conditions the voltage increments can be larger than those used under high-level-injection conditions. Let the built-in potential of a pn-junction be V_{bi}, then the limiting is performed as shown in Figure 5.5 for positive values of the diode voltage. At present CODECS uses $\Delta V_1 = 4V_t$ and $\Delta V_2 = 2V_t$. The values of ΔV_1 and ΔV_2 may appear to be conservative, but this choice has performed well over a wide variety of examples.

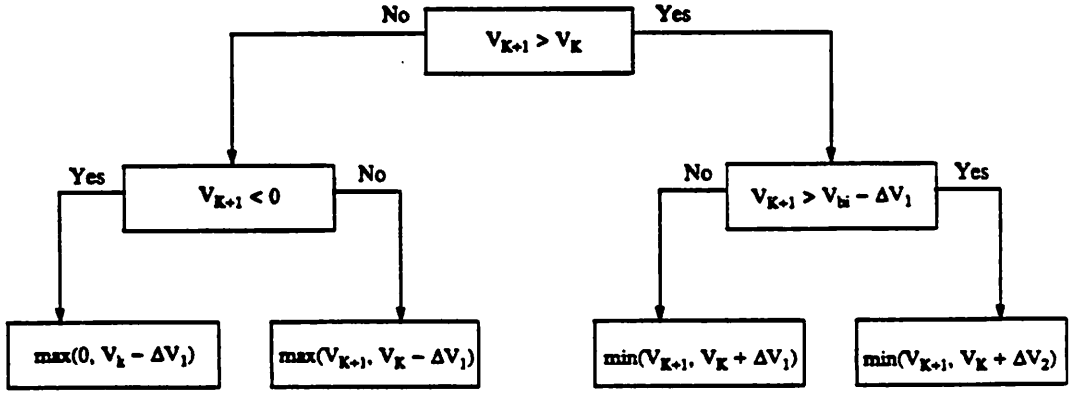


Figure 5.5: Limiting scheme for forward-biased pn junctions.

5.4.2. Voltage-Step Backtracking

The above limiting scheme can lead to nonconvergence under some operating conditions. As an aid to convergence a voltage-step backtracking scheme has also been implemented in CODECS. If convergence is not achieved at the device level for a certain increment in the terminal voltage ΔV, the device internal state is restored to the previous solution, and a new voltage increment of $\Delta V/2$ is used. This process may be repeated until a voltage increment is found for which the device-level equations converge. Without exception, convergence can be achieved at the device level with a sufficiently small increment in the terminal voltage. Device-based limiting combined with the backtracking scheme exhibits good convergence for dc analysis.

5.4.3. Linear Prediction of Initial Guess

A linear-projection scheme as used in [5.13] is also essential for faster convergence. This scheme is derived in an alternate manner here. Consider the device-level equations as in Equation (5.24). These equations are solved by use of Newton's method whereby,

$$\Delta \mathbf{w} = -\mathbf{J}_w^{-1}\mathbf{F}(\mathbf{w}, V) \tag{5.25}$$

is solved at each iteration until convergence is achieved; $\mathbf{J}_w = \frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ is the Jacobian of the device-level equations. Newton's method requires a good initial guess to ensure convergence and the linear prediction step attempts to provide such a guess. The first-order prediction is made by use of a forward-Euler scheme

$$\mathbf{w}^{k+1} = \mathbf{w}^k + \left[\frac{\partial \mathbf{w}}{\partial V}\right]_k \Delta V \tag{5.26}$$

where ΔV is the change in voltage from bias point k to k+1. To obtain $\frac{\partial \mathbf{w}}{\partial V}$ differentiate Equation (5.24) with respect to V, then

$$\frac{\partial \mathbf{F}}{\partial \mathbf{w}}\frac{\partial \mathbf{w}}{\partial V} + \frac{\partial \mathbf{F}}{\partial V} = \mathbf{O} \tag{5.27}$$

from which

$$\frac{\partial \mathbf{w}}{\partial V} = -\mathbf{J}_w^{-1}\left[\frac{\partial \mathbf{F}}{\partial V}\right] \tag{5.28}$$

$\frac{\partial \mathbf{F}}{\partial V}$ has nonzero terms corresponding only to the contact nodes and can be easily assembled. $\mathbf{J}_w$ is available in its LU-factors from the solution of Equation (5.24) by use of Equation (5.25), hence calculation of $\frac{\partial \mathbf{w}}{\partial V}$ requires only forward and back substitutions.

The above scheme does provide a good initial guess as demonstrated in [5.13]. However, it has also been suggested that convergence is faster if a fraction of ΔV is used for predicting the electron and hole concentrations. A typical value suggested was $0.7\Delta V$. This indicates that a straight-forward implementation of the above scheme may not provide a speed up.

A careful analysis of the linear prediction scheme has shown that nonconvergence may occur when the prediction step results in a negative value for the carrier concentrations. This is unphysical; therefore, the prediction step has to be modified to predict physically acceptable values. The improved linear prediction scheme in CODECS uses a Fibonacci search sequence to determine acceptable values of Δn and Δp whenever the initial values of Δn or Δp result in negative carrier concentrations.

The prediction schemes are now examined based on their convergence properties. Since the application is for a mixed-level circuit- and device-simulation environment, these schemes have been evaluated for a dc operating point analysis using the two-level Newton algorithm on circuit examples with numerical models for the semiconductor devices. In Table 5.1 are summarized the results for the classical Newton's method (NewtonA), Newton's method with linear prediction (NewtonB), and Newton's method with improved linear prediction (NewtonC). The numbers are given in the sequence: number of device iterations, number of circuit iterations, and the runtime in seconds on a VAX 8650.

The schemes without prediction and with linear prediction have convergence problems. The Newton scheme with prediction requires a smaller number of device-level iterations on the examples in which convergence is achieved; 36 and 30, respectively, for the *Invchain* example and 34 and 28, respectively, for the *Oscillator* example. The Newton scheme without prediction converges in the *MECLgate* circuit but does not when prediction is used. This depicts that if the prediction step is implemented without any

Circuit	NewtonA	NewtonB	NewtonC
RTLinv	36/8/5.4s	30/8/4.8s	31/8/5s
Oscillator	34/8/4.5s	28/8/4.5s	28/8/4.5s
VCO	-	147/8/24s	152/8/25s
Invchain	-	-	150/9/22s
Astable	-	-	76/9/11s
MECLgate	668/51/94s	-	521/51/81s

Table 5.1: Comparison of iterations and runtimes

modifications it could also lead to nonconvergence. The modified prediction scheme does the best; it converges on all examples. Furthermore, for the examples in which the prediction step without modifications converges the iteration counts are similar. The modified prediction scheme requires twenty-two percent fewer device-level iterations compared with Newton's method for the *MECLgate* example. From the above results it is borne out that the prediction step must ensure that carrier concentrations are never negative; this results in an overall better convergence. The modified two-level Newton algorithm described in Chapter 4 makes use of the modified linear prediction scheme for dc analysis and a simple prediction during transient analysis.

The above analysis may suggest that use of a modified updating scheme for the full-Newton algorithm of Chapter 4 may give better dc convergence. However, this is not the case, since with the full-Newton algorithm the update scheme is part of a Newton step and is not used to predict an initial guess for the solution as in the two-level Newton algorithm. In the full-Newton scheme any modification in the updating scheme corrupts the Newton direction and can actually degrade the convergence. The results obtained by use of a simple and modified updating for the full-Newton method are compared in Table 5.2. The numbers are in the sequence of number of device iterations, circuit iterations, and the runtime in seconds on a VAX 8650. For the full-Newton

algorithm the number of device-level iterations is the product of the number of circuit-level iterations and the number of numerical devices. In addition some iterations are required to obtain an initial solution for the numerical devices, typically 8 iterations per device. For the VCO example with 10 circuit-level iterations and 6 numerical devices the number of iterations is 60. An additional 6×8 iterations are required for generating the initial solution giving a total of 108 device-level iterations.

Circuit	Simple Updating	Modified Updating
RTLinv	16/8/2.4s	17/9/2.7
Oscillator	17/9/2.6s	17/9/2.7
VCO	108/10/16s	114/11/18s
Invchain	-	-
Astable	-	-
MECLgate	-	-

Table 5.2: Comparison of simple and modified updating schemes

5.4.4. Norm-Reducing Newton's Method

In addition to the above schemes, Newton's method at the device level has to be guided into its region of convergence. The classical Newton method tends to overshoot, particularly so if the initial guess is a poor one. It may so happen that Newton's method may never enter its region of convergence and may eventually result in nonconvergence. A typical way of avoiding overshoot in Newton's method is to use a norm-reducing scheme in which the Newton update is chosen such that the norm of the right-hand-side vector reduces from iteration to iteration. Such schemes have been proposed by [5.14] and [5.15].

The norm-reducing Newton method is now described. Let $\Delta \mathbf{w}^k$ be the Newton update at iteration k, then $\mathbf{w}^{k+1}$ is calculated as

$$\mathbf{w}^{k+1} = \mathbf{w}^k + \lambda \Delta \mathbf{w}^k \tag{5.29}$$

where λ is chosen such that

$$||F(\mathbf{w}^{k+1})|| < ||F(\mathbf{w}^k)|| \tag{5.30}$$

i.e., the norm of the right-hand-side vector decreases monotonically. The calculation of λ can be done in several ways as in [5.15-5.17]. The Armijo step-size rule of [5.17] has been used in CODECS, in which the search technique instead of being exponential in nature is based on the Fibonacci sequence. This technique has been previously implemented for circuit-simulation problems [5.18] and gave good results. When $\mathbf{w}^k$ is close to the solution, the rate of convergence of Newton's method is quadratic, whereby Equation (5.30) would be satisfied from one iteration to the next with $\lambda = 1$. Thus, near the solution the norm-reducing Newton method reduces to the classical Newton method.

An appropriate norm must be used in Equation (5.30) otherwise the convergence may be extremely slow. The L2 norm has been frequently used [5.15, 5.16]. However, as shown in [5.19], the steepest-descent direction of the L2 norm may not coincide with the Newton-update direction and results in slow convergence of the norm-reducing Newton method. An alternate norm, the 'Nu' norm, has been proposed in [5.19] which ensures that the direction of steepest descent of the norm coincides with the update direction. In CODECS the max norm is used; it gives good performance and can be easily calculated. The 'Nu' norm requires forward and back substitutions with the LU-factored device-level Jacobian matrix and hence is more expensive to evaluate. A comparison of the convergence in the max and 'Nu' norms when used at the device level with the two-level Newton scheme is given in Table 5.3. Device-based limiting and the modified linear prediction are used in both cases. The numbers are in the sequence: device-level iterations, circuit-level iterations, and runtime in seconds on a VAX 8650.

Circuit	Max Norm	Nu Norm
RTLinv	31/8/5s	32/8/6s
Oscillator	28/8/4.5s	28/8/4.7s
VCO	152/8/25s	158/8/28s
Invchain	150/9/22s	152/9/25s
Astable	76/9/11s	77/9/12.5s
MECLgate	521/51/81s	583/51/97s

Table 5.3: Comparison of Max and 'Nu' norms

It is seen from Table 5.3 that for all the examples the convergence with the max norm is similar to that with the 'Nu' norm. For the *MECLgate* example, however, a much larger number of device-level iterations are required to achieve convergence with the 'Nu' norm. Based on the results of these experiments, the max norm is used in CODECS for the norm-reducing Newton method.

5.5. Transient Analysis

Nonlinear time-domain simulation is an extremely useful analysis, particularly so in a mixed-level circuit and device simulator since it provides a way of evaluating the dynamic performance of devices within a circuit environment. The transient behavior of a device can vary depending upon its region of operation and the applied terminal voltages. Even with good analytical models for the dc characteristics of a device it is difficult to predict the dynamic operation of the device.

In Chapter 4 it has been shown that convergence is not as serious a problem in transient analysis as it is for dc analysis. The full-Newton scheme was found to have the best runtime performance and is therefore used in CODECS. This section describes the integration formulae and timestep-control schemes used at the device level of CODECS.

The nonlinear device-level equations after space discretization can be expressed in the form

$$F(\dot{\mathbf{w}}(t), \mathbf{w}(t), V(t)) = \mathbf{O} \tag{5.31}$$

where $\mathbf{w}$ is the vector of internal variables and $V(t)$ are the applied terminal voltages. This system of differential-algebraic equations is transformed to a system of nonlinear algebraic equations after discretization in time, i.e., at timepoint t_{n+1},

$$\tilde{F}(\mathbf{w}_{n+1}) = \mathbf{O} \tag{5.32}$$

These nonlinear equations can be solved by Newton's method. The discretization in time is done by use of an integration formula which provides an approximation for the time derivative of a variable $x(t)$ at time t_{n+1} in terms of the values of the variable at t_{n+1} and values of the variable and its derivatives at the previous timepoints. *Linear multistep* (p-step) integration formulae of order k, that are in common use in computer-aided circuit and device analysis, are given by

$$\sum_{i=0}^{p} a_i x_{n+1-i} - h_n \sum_{i=0}^{p} b_i \dot{x}_{n+1-i} = 0 \tag{5.33}$$

where $2p+1-k$ of the a_i and b_i are assigned arbitrarily and $h_n = t_{n+1} - t_n$. The rest of the coefficients are determined from *exactness constraints*, i.e., a kth-order integration formula is exact for polynomials in t of degree less than or equal to k. An integration method is an implicit method when $a_0 \neq 0$. Implicit methods require an iterative solution but have larger regions of stability and hence are preferred. The integration methods must be stiffly stable to be of practical use since the time constants of the equations may vary by several orders of magnitude [5.1, 5.20].

Trapezoidal integration has been widely used in circuit simulation [5.1]. It is an A-stable method that has the smallest local error [5.21]. However, it is not suitable for device-level simulation because it is not L-stable [5.22]. Device-simulation problems are

extremely stiff and if the local error is not monitored carefully the results exhibit ringing, an artifact of the integration method. With a proper control on the error, the timesteps taken may be small whereby there is a restriction on the timestep and no advantage in using the trapezoidal method [5.23]. The backward-Euler method has been used in several device simulators [5.4, 5.24], even though it is a first-order method and has a large local error. A second-order method that is both A-stable and L-stable is the backward-differentiation formula of second order, BDF2. Thus, it is well suited to device-level simulation problems. However, it also has a local error that is larger than the trapezoidal method and an alternate integration formula is the TR-BDF2 scheme proposed in [5.23]. The TR-BDF2 method requires two solutions for each timepoint, the first makes use of the trapezoidal integration for a fraction of the timestep and then BDF2 is used for the rest of the interval. Implementation of this integration scheme in CODECS would require modifications to the core of the circuit simulator, which controls the simulation. Since the intent has been to keep the circuit-simulation algorithms unaltered, the algorithms have to be chosen to ensure decoupling between the circuit and device simulators. For this reason the TR-BDF2 has not been considered for time-domain transient analysis in CODECS.

5.5.1. Local Error and Error Estimates

Assume the linear multistep integration method is given by Equation (5.33). Furthermore, assume that this equation has been normalized such that $\sum_{i=0}^{p} b_i = 1$. The local error of the integration method at time point t_{n+1} is then defined as

$$LE_{n+1} = \sum_{i=0}^{p} a_i x(t_{n+1-i}) - h_n \sum_{i=0}^{p} b_i \dot{x}(t_{n+1-i}) \tag{5.34}$$

where $x(t_{n+1})$ denotes the exact solution to the differential equation at time t_{n+1}. Thus,

local error is the amount by which the exact solution fails to satisfy the difference equation, Equation (5.33). The above equation can be expanded around t_n by use of Taylor series. For a kth-order integration method, with the assumption of uniform stepsizes, the local error [5.25] is

$$LE_{n+1} = C_{k+1} h^{k+1} x^{k+1}(t_n) \tag{5.35}$$

where C_{k+1} is called the error constant of the integration method. A method of order k that has the smallest value of C_{k+1} will have the smallest local error and will therefore be more accurate. From Equation (5.35) it might appear that by scaling Equation (5.33), C_{k+1} could be reduced thereby reducing the error of the method. However, the values of x computed from Equation (5.33) will not change; therefore, the accuracy of the method will not be affected. It is for this reason that the normalization $\sum_{i=0}^{p} b_i = 1$ was introduced; after this normalization the constant C_{k+1} is invariant to such scalings [5.25] and hence can be used as a measure of the accuracy. The local error for two second-order methods, BDF2 (second-order BDF) and trapezoidal method, is now derived for the case of nonuniform step sizes.

5.5.1.1. Local Error for the BDF2 Integration Method

The BDF2 integration method is given by

$$\sum_{i=0}^{2} a_i x_{n+1-i} - h_n \dot{x}_{n+1-i} = 0 \tag{5.36}$$

where a_0, a_1 and a_2 are obtained from the exactness constraints. Their values are

$$a_0 = \frac{2h_n + h_{n-1}}{h_n + h_{n-1}}$$

$$a_1 = -\frac{h_n + h_{n-1}}{h_{n-1}}$$

$$a_2 = \frac{h_n^2}{h_{n-1}(h_n + h_{n-1})}$$

where $h_n = t_{n+1} - t_n$. To obtain the error, the exact solution $x(t)$ is substituted in Equation (5.36)

$$LE_{n+1} = \sum_{i=0}^{2} a_i x(t_{n+1-i}) - h_n \dot{x}(t_{n+1-i}) \tag{5.37}$$

From a Taylor series expansion around t_n, it can be shown that

$$LE_{n+1} \approx -\frac{h_n^2(h_n + h_{n-1})}{6} \frac{d^3x(t_n)}{dt^3} \tag{5.38}$$

5.5.1.2. Local Error for the Trapezoidal Rule Integration Method

The trapezoidal method when expressed in the form of Equation (5.33) with the normalization $\sum_{i=0}^{P} b_i = 1$ is given by

$$x_{n+1} - x_n - h_n \left[\frac{\dot{x}_{n+1}}{2} - \frac{\dot{x}_n}{2} \right] = 0 \tag{5.39}$$

Substituting the exact solution $x(t)$ in Equation (5.39), one obtains

$$LE_{n+1} = x(t_{n+1}) - x(t_n) - h_n \left[\frac{\dot{x}(t_{n+1})}{2} - \frac{\dot{x}(t_n)}{2} \right] \tag{5.40}$$

From a Taylor series expansion around t_n, it can be shown that

$$LE_{n+1} \approx -\frac{h_n}{12} \frac{d^3x(t_n)}{dt^3} \tag{5.41}$$

5.5.2. Estimation of Local Error

The calculation of local error for a second-order integration method requires an estimation of $\frac{d^3x(t_n)}{dt^3}$, i.e., the third derivative of the solution. One way to do this is to make use of divided differences as in SPICE [5.1]. However, divided differences are prone to errors and may not provide a good estimate of the local error. An alternative approach is the use of a predictor-corrector technique which is illustrated for a second-order method.

Consider a second-order polynomial predictor to be used with a second-order integration method. Let $x(t)$ be the actual solution, then a predicted value for x at time t_{n+1} is given by

$$x^P(t_{n+1}) = c_1 x(t_n) + c_2 x(t_{n-1}) + c_3 x(t_{n-2}) \tag{5.42}$$

where the coefficients c_1, c_2, and c_3 are determined by exactness constraints.

$$c_1 = 1 + \frac{h_n(h_n + 2h_{n-1} + h_{n-2})}{h_{n-1}(h_{n-1} + h_{n-2})}$$

$$c_2 = -\frac{h_n(h_n + h_{n-1} + h_{n-2})}{h_{n-1}h_{n-2}}$$

$$c_3 = \frac{h_n(h_n + h_{n-1})}{h_{n-2}(h_{n-1} + h_{n-2})}$$

Consider the difference between the exact solution $x(t_{n+1})$ and the predicted value $x^P(t_{n+1})$. From a Taylor series expansion around t_n, it can be shown that [5.26]

$$x(t_{n+1}) - x^P(t_{n+1}) \approx \frac{h_n}{6}(h_n + h_{n-1})(h_n + h_{n-1} + h_{n-2})\frac{d^3x(t_n)}{dt^3} \tag{5.43}$$

Thus, $\frac{d^3x(t_n)}{dt^3}$ can be approximated by

$$\frac{d^3x(t_n)}{dt^3} \approx \frac{6}{h_n(h_n + h_{n-1})(h_n + h_{n-1} + h_{n-2})}(x_{n+1} - x^P_{n+1}) \tag{5.44}$$

where x_{n+1} and x_{n+1}^P the computed values of $x(t_{n+1})$ and $x^P(t_{n+1})$, respectively, have been used. The above formula is quite useful since it relates the local error to the difference between the corrected and predicted values. Furthermore, the predicted value provides a good initial solution for the new time point.

Based on the above approximation for the third derivative of $x(t)$ the local errors for the BDF2 and trapezoidal integration methods can be expressed as

$$\text{BDF2:} \quad LE_{n+1} = \frac{h_n}{h_n + h_{n-1} + h_{n-2}}(x_{n+1} - x_{n+1}^P) \tag{5.45}$$

$$\text{TR:} \quad LE_{n+1} = \frac{h_n^2}{2(h_n + h_{n-1})(h_n + h_{n-1} + h_{n-2})}(x_{n+1} - x_{n+1}^P) \tag{5.46}$$

A linear test problem has been used to evaluate these error estimates. The linear problem is

$$\dot{x} = \lambda x \ , \quad \lambda = -1 \ , \quad x(0) = 5$$

The computed local-error estimate from Equations (5.45) and (5.46) is compared with the actual local error, using the definition of the local error. The solution is started with an $h_{\min} = 1\mu\text{sec}$. The timestep is doubled every timepoint until it reaches the maximum allowed timestep of 0.25sec. The results for BDF2 and TR are shown in Figure 5.6. It is seen that BDF2 has a higher local error compared to TR as expected. Furthermore, the agreement between the calculated value of local error and the actual local error is good. One can only expect the above estimates to be reasonable for a nonlinear problem, such as the device-simulation problem.

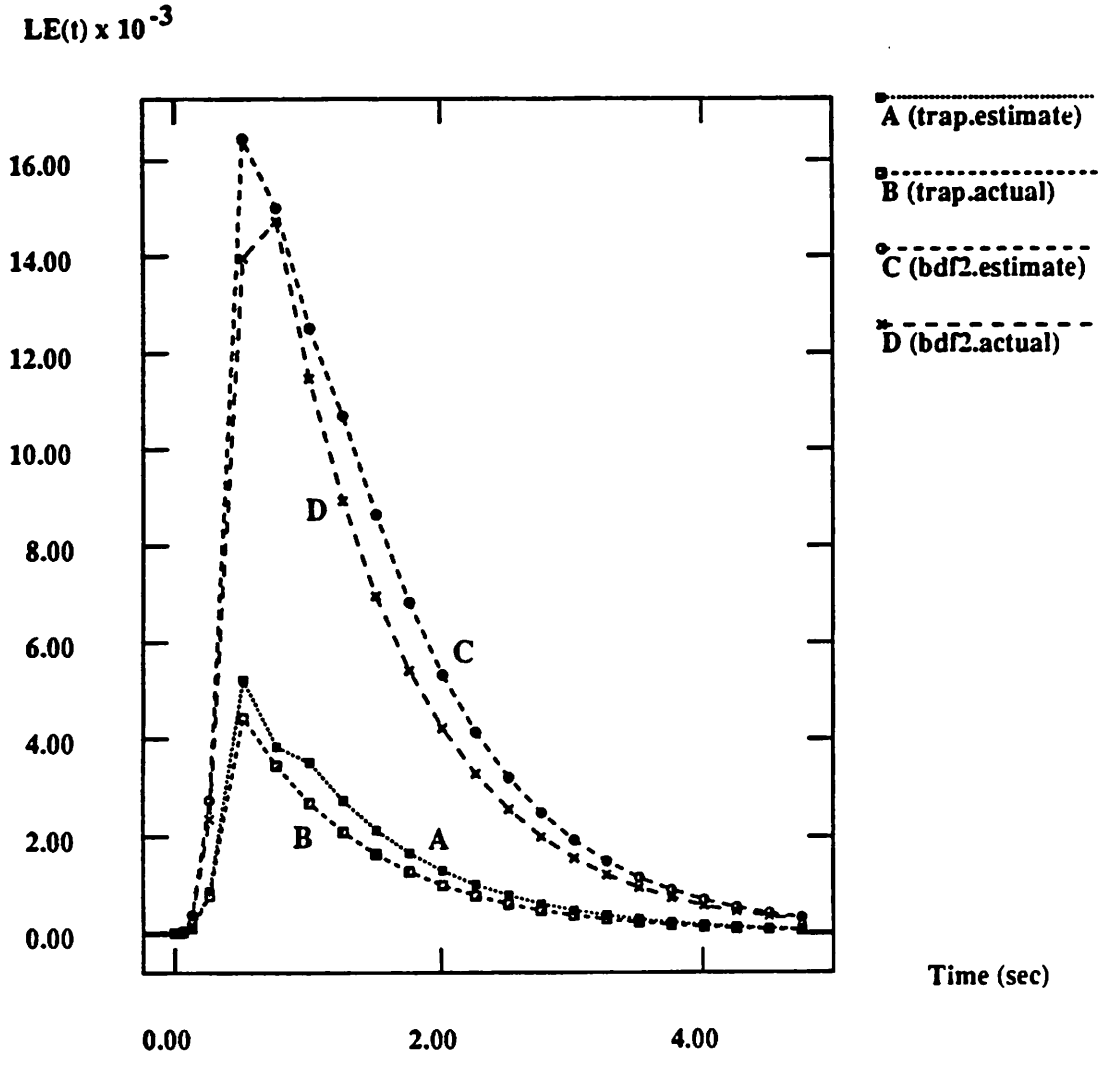


Figure 5.6: Estimated and actual values of local error for the trapezoidal and BDF2 integration methods.

5.5.3. Time-Step Control in CODECS

Timesteps used during transient analysis must ensure accuracy of a stiffly-stable integration scheme which is used for the solution of stiff systems. The simplest scheme is to use a fixed timestep over the whole time interval. However, this is not appropriate for stiff problems, such as the circuit- and device-simulation problems, because the time constants of the equations may vary by several orders of magnitude. The timestep must be small enough to accurately resolve the fast transients. This results in the use of an extremely large number of timepoints when computing a solution with fast and slow transients. Varying the timestep during the simulation can provide speed as well as accuracy. Thus, fast transients would be resolved with small timesteps whereas during the slow transient larger timesteps are used. This section describes the timestep control used at the device level in CODECS. The allowed timesteps are determined from the local-error estimates derived earlier.

A solution computed at timepoint t_{n+1} by use of an integration method is considered to be acceptable if the local error E_{n+1} is less than a user-specified error tolerance, i.e.,

$$E_{n+1} < E_{user} \tag{5.47}$$

where E_{user} can be expressed in terms of a relative error tolerance and an absolute error-tolerance parameter ε_r and ε_a, respectively. These parameters are used to control the accuracy,

$$E_{user} = \varepsilon_r \, |x_{n+1}| + \varepsilon_a \tag{5.48}$$

The timesteps can be controlled by considering the relative error of the allowable local error and the actual local error.

$$r = \frac{E_{n+1}}{E_{user}} = \frac{|C_{k+1} h_n^{k+1} x^{(k+1)}(t_n)|}{|C_{k+1} h_{allowable}^{k+1} x^{(k+1)}(t_n)|} = \left[\frac{h_n}{h_{allowable}} \right]^{k+1} \tag{5.49}$$

Thus, the allowable timestep for the kth-order method is given by

$$h_{allowable} = h_n \times r_{LE} \tag{5.50}$$

where

$$r_{LE} = r^{\frac{-1}{k+1}} = \left[\frac{E_{user}}{E_{n+1}}\right]^{\frac{1}{k+1}} \tag{5.51}$$

The value of r_{LE} is used to select or reject the new timestep based on an a posteriori analysis. For device-level equations it is appropriate to use a root-mean-squared norm for calculating the relative error [5.23]. Thus r is calculated as

$$r = \left[\frac{1}{N}\sum_{i=1}^{N}\left[\frac{E_{user,i}}{E_{n+1,i}}\right]^2\right]^{\frac{1}{2}} \tag{5.52}$$

where N is the number of equations.

The timestep selection criterion is now described. Consider the solution at time t_{n+1} which was computed using a timestep h_n. If $r_{LE} < 0.9$ then timestep h_n is rejected and the solution for t_{n+1} is attempted with a new timestep $h^* = h_n \times r_{LE}$. For values of $r_{LE} \geq 0.9$, the timestep is considered acceptable and the new timestep is taken to be $MIN(r_{LE} \times h_n, 2.0 \times h_n, h_{print})$, where h_{print} is the print-time interval. At each timepoint the timestep is allowed to increase by at most a factor of two and never allowed to exceed the print-time interval. In addition to the relative error criterion a timepoint is also rejected if too many iterations are required for convergence at that timepoint. The timestep is rejected whenever convergence is not achieved in ten iterations and a new solution is attempted with a timestep of $h^* = \frac{h_n}{8}$.

It is well known that stability results for the various integration methods are applicable only when uniform timesteps are used. Nothing can be said about the stability of an integration method when the timesteps are allowed to vary. It can only be hoped that

the local error will lead to small enough timesteps such that the integration method remains stable. However, most linear multistep methods (except for one-step methods) can be made unstable by a suitable variation in the timesteps [5.27]. It can be shown that the second-order BDF is stable with nonuniform step sizes when the ratio of timesteps $\frac{h_{n+1}}{h_n}$ is kept smaller than 1.2 [5.28]. Intuitively, it appears that if the variations in timesteps are made gradual, stability problems can be avoided.

An alternate scheme for timestep control is described that provides for a gradual change in the timesteps. For $0.9 \leq r_{LE} \leq 1.2$ the step size is left unchanged, $h_{n+1} = h_n$. With $r_{LE} > 1.2$ a linear increase in timestep is allowed whereby

$$h_{n+1} = MIN(h_n(1 + 0.9(r_{LE} - 1.2)), 2h_n, h_{print})$$

When $0.5 \leq r_{LE} < 0.9$, the timepoint is rejected and the step size is reduced by a factor of two. For $r_{LE} < 0.5$ the timepoint is rejected and a new solution is computed by using the first-order BDF with a timestep of $r_{LE} \times h_n$. This scheme also keeps a check on the number of iterations required for convergence at a particular timepoint. Whenever the number of iteration exceeds ten, the timepoint is rejected and a new solution is attempted with a step size of $\frac{h_n}{8}$.

The two schemes for timestep control are compared for the BDF2 integration method in Table 5.4. The data is given for the benchmark examples using the full-Newton algorithm. The numbers for the various examples are in the following sequence: number of iterations, number of timepoints accepted, number of timepoints rejected, total number of timepoints, and the run times in seconds on a VAX 8800.

The second scheme for timestep control does better in only one example, the *VCO* circuit; the number of iterations and hence the runtime is smaller. In all the other examples the first scheme does almost the same or better. Scheme 2 requires a larger number

Circuit	Scheme 1	Scheme 2
Oscillator	18333	21147
	5274	6188
	361	426
	5635	6614
	2352	2760
VCO	5864	5797
	1125	1200
	176	133
	1301	1333
	2911	2760
Invchain	1716	1915
	401	453
	17	23
	418	476
	514	585
Astable	6369	6826
	1465	1675
	181	134
	1546	1809
	1230	1326
MECLgate	2609	2920
	619	691
	31	41
	650	732
	2121	2345
Pass	295	292
	82	89
	8	4
	90	93
	1059	1014
MOSinv	336	340
	95	103
	4	2
	99	105
	1155	1200
ChargePump	1850	1906
	493	556
	73	55
	566	611
	4039	4301

Table 5.4: Comparison of two timestep control schemes

of timepoints and this is due to the restrictions imposed on the timesteps. Since the first scheme does not show any instability in these examples, there may be no apparent advantage in using a timestep-control strategy that is as restrictive as the second scheme. Therefore, in CODECS, the first scheme is used for controlling the timesteps.

For the trapezoidal integration method, the two schemes are compared in Table 5.5. The conclusions again are similar to those obtained with the BDF2 integration. The first scheme does better in almost all the examples.

Finally, the results for the BDF2 and trapezoidal methods are compared for the first timestep control scheme in Table 5.6. This comparison allows an evaluation of which integration method is more suitable for use in a mixed-level circuit and device simulator.

The TR integration performs remarkedly better on the *Oscillator*, *MECLgate* and *Pass* circuits. For the other examples it performs almost similar to BDF2 or slightly worse than it. From this data it is clear that TR integration does not provide as significant an improvement in performance as it did for circuit simulation [5.1]. The total number of timepoints (accepted and rejected) are smaller with the TR algorithm but the number of rejected timepoints is significantly larger when compared with BDF2. This indicates that computational effort is wasted in determining the solution at timepoints that are eventually rejected. The timepoints are rejected whenever the timestep fails to meet the error criterion or when convergence is not achieved in ten iterations. A breakup of the timepoint rejections is presented in Table 5.7, where the statistics of timepoints rejected by the error criteria (Local Error Reject) and due to nonconvergence (Noncon Reject) are given for the BDF2 and TR integration methods. The number of iterations required per timepoint are also included.

For the BDF2 method, a significant number of timepoints are rejected due to non-convergence only in the *VCO* and *Astable* examples. This is to be expected since the circuit waveforms have sudden transitions which result in a large change in the terminal

Circuit	Scheme 1	Scheme 2
Oscillator	16127	16712
	3491	3939
	705	539
	4196	4478
	2063	2103
VCO	5890	5923
	1003	1049
	209	186
	1212	1235
	3000	3029
Invchain	1738	1715
	322	346
	66	45
	388	391
	545	514
Astable	5744	6406
	1107	1331
	216	219
	1323	1550
	1139	1232
MECLgate	1966	2138
	429	476
	55	53
	484	529
	1680	1806
Pass	240	232
	67	71
	4	2
	71	73
	865	858
MOSinv	326	317
	86	85
	7	3
	93	88
	1120	1108
ChargePump	1853	1887
	399	441
	81	73
	480	514
	4124	4396

Table 5.5: Comparison of two timestep control schemes

Circuit	BDF2	Trapezoidal
Oscillator	18333	16127
	5274	3491
	361	705
	5635	4196
	2352	2063
VCO	5864	5890
	1125	1003
	176	209
	1301	1212
	2911	3000
Invchain	1716	1738
	401	322
	17	66
	418	388
	514	545
Astable	6369	5744
	1465	1107
	181	216
	1546	1323
	1230	1139
MECLgate	2609	1966
	619	429
	31	55
	650	484
	2121	1680
Pass	295	240
	82	67
	8	4
	90	71
	1059	855
MOSinv	336	326
	95	86
	4	7
	99	93
	1154	1120
ChargePump	1850	1853
	493	399
	73	81
	566	480
	4039	4124

Table 5.6: Comparison of BDF2 and Trapezoidal methods

Circuit	BDF2			Trapezoidal		
	Local Error Rejects	Noncon Rejects	Iter per Timepoint	Local Error Rejects	Noncon Rejects	Iter per Timepoint
Oscillator	361	0	3.3	504	201	3.8
VCO	103	73	4.5	91	118	4.9
Invchain	15	2	4.1	52	14	4.5
Astable	148	33	4.1	139	77	4.3
MECLgate	30	1	4.0	53	2	4.1
Pass	8	0	3.3	4	0	3.4
MOSinv	4	0	3.4	4	3	3.5
ChargePump	62	11	3.3	42	39	3.9

Table 5.7: Comparison of rejected timesteps

voltages of the device. Consequently a large number of iterations are required for convergence at such timepoints. Whenever, the number of iterations exceeds ten the timepoint is rejected and the timestep is reduced by a factor of eight. For the TR method a large number of timepoints are rejected due to nonconvergence in all but three examples. The larger number of nonconvergent timepoints with the trapezoidal integration can be explained in the following manner. Since TR allows larger timesteps at each timepoint, due to the smaller local error, the terminal voltages of the device will change by larger amounts. As a result a larger number of iterations will be required for convergence, increasing the occurrence of nonconvergent timepoints.

The waveforms obtained by the two integration methods for different circuits are almost identical and indicate that the trapezoidal integration method showed no ringing. This was achieved by properly monitoring and controlling the local error. However, the advantage gained by use of the trapezoidal method is not as significant as it is for circuit simulation [5.1]. For this reason both the BDF2 and TR algorithms are available in

CODECS.

5.5.4. Higher-Order Integration Methods

In mixed-level circuit and device simulation each timepoint requires the solution of device-level equations. If the number of timepoints can be reduced without substantially increasing the number of iterations per timepoint then significant improvement in speed performance is possible. One way of achieving a smaller number of timepoints is by the use of higher-order integration methods which have smaller local errors. Backward-differentiation formulae of order k, $1 \le k \le 6$, have been used for circuit-simulation problems and are now evaluated for CODECS. A kth-order BDF [5.27] is given by

$$\sum_{i=0}^{k} a_i x_{n+1-i} - h_n \dot{x}_{n+1} = 0 \tag{5.53}$$

The first- and second-order BDF are astable; and, hence, stiffly stable. Methods of order k, $3 \le k \le 6$, are stiffly stable and methods of order greater than seven are unstable [5.27]. The coefficients a_i are functions of the step sizes and are determined by the exactness constraints whereby a system of $k+1$ linear equations is solved at each timepoint. In SPICE, full-matrix solution of the linear equations is performed to determine the a_i's [5.1]. However, CODECS uses closed-form expressions for a_i as given in [5.26].

$$a_0 = h_n \sum_{j=1}^{k} \frac{1}{(t_n - t_{n-j})}$$

$$a_i = -\frac{h_n}{(t_n - t_{n-i})} \prod_{j=1, j\neq i}^{k} \frac{(t_n - t_{n-j})}{(t_{n-i} - t_{n-j})}, \qquad i = 1, 2, \cdots, k \tag{5.54}$$

A kth-order polynomial predictor is given by

$$x_{n+1}^{P} = \sum_{i=1}^{k+1} c_i x_{n+1-i} \tag{5.55}$$

where c_i can be determined by exactness constraints. Closed-form expressions for c_i are also available [5.26] and are given by

$$c_i = \prod_{j=1, j\neq i}^{k+1} \frac{(t_n - t_{n-j})}{(t_{n-i} - t_{n-j})} \tag{5.56}$$

It can be shown that the local error at time t_{n+1} using a kth-order integration method is [5.27]

$$E_{n+1}^k = \frac{h_n}{h_n + h_{n-1} + \cdots + h_{n-k}} (x_{n+1}^k - x_{n+1}^{k,P}) \tag{5.57}$$

where x_{n+1}^k is the computed value of the solution $x(t_{n+1})$ and $x_{n+1}^{k,P}$ is the predicted value.

The above error estimates are examined for the linear test problem

$$\dot{x} = \lambda x, \quad \lambda = -1.0, \quad x(0) = 5$$

This differential equation is integrated by use of higher-order BDF methods and the computed local error is compared to the actual error. The integration is started with a minimum timestep of 1μsec, and the timestep is doubled every timepoint until it reaches a maximum value of 0.25sec. The integration order is raised from order k to k+1 after k+1 steps of the kth-order method, which is similar to the scheme of [5.27] for increasing the integration order. The estimates of local error from Equation (5.57) and the actual error are compared in Figures 5.7(a) and 5.7(b). The error estimates are very close to the actual error for integration orders k, $2 \leq k \leq 5$. For the sixth-order BDF, the local error estimate shows some ringing and overestimates the error, but the error is quite small; less than 10^{-4}.

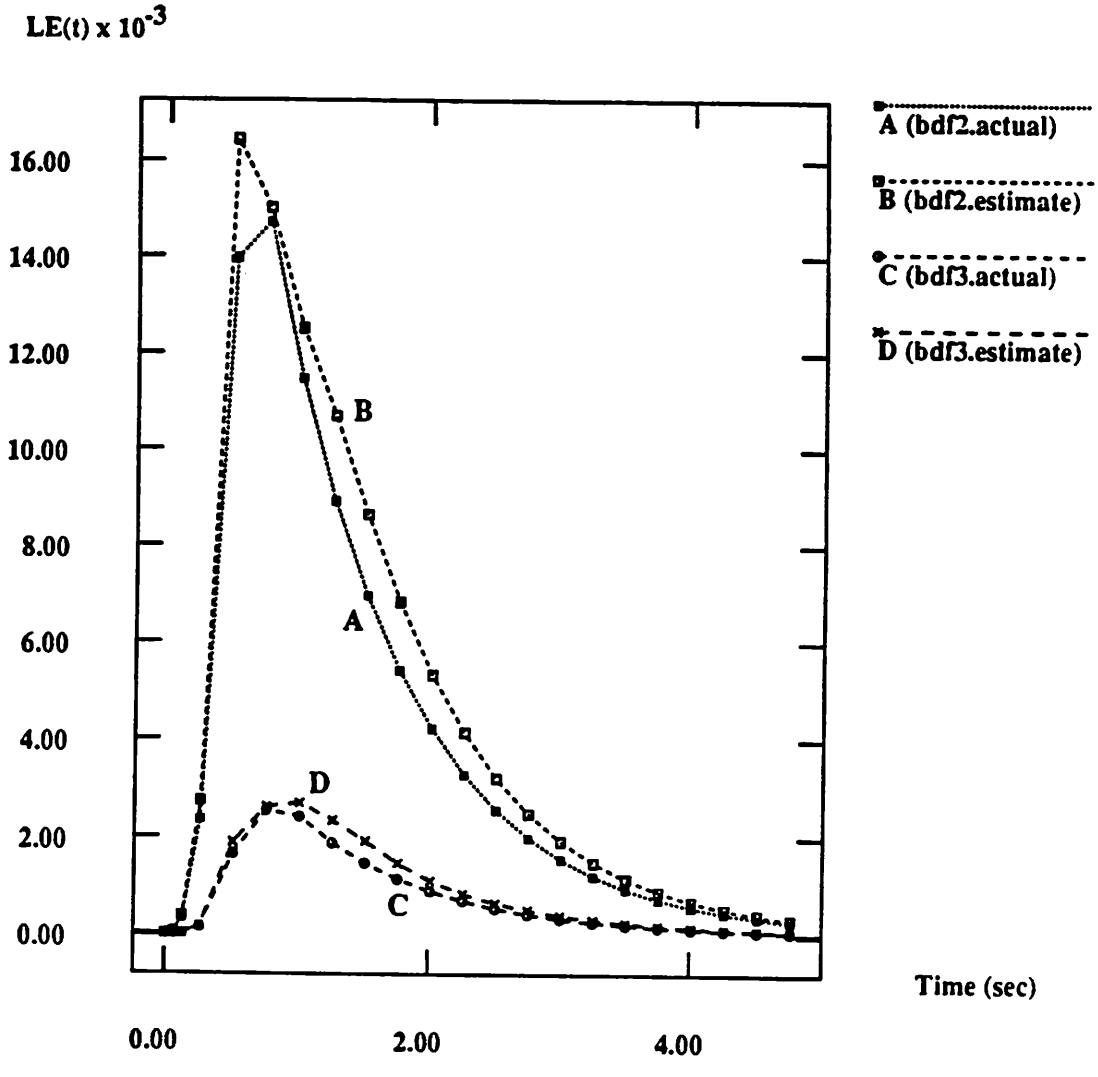


Figure 5.7(a): Estimated and actual values of local error for BDF2 and BDF3 integration methods.

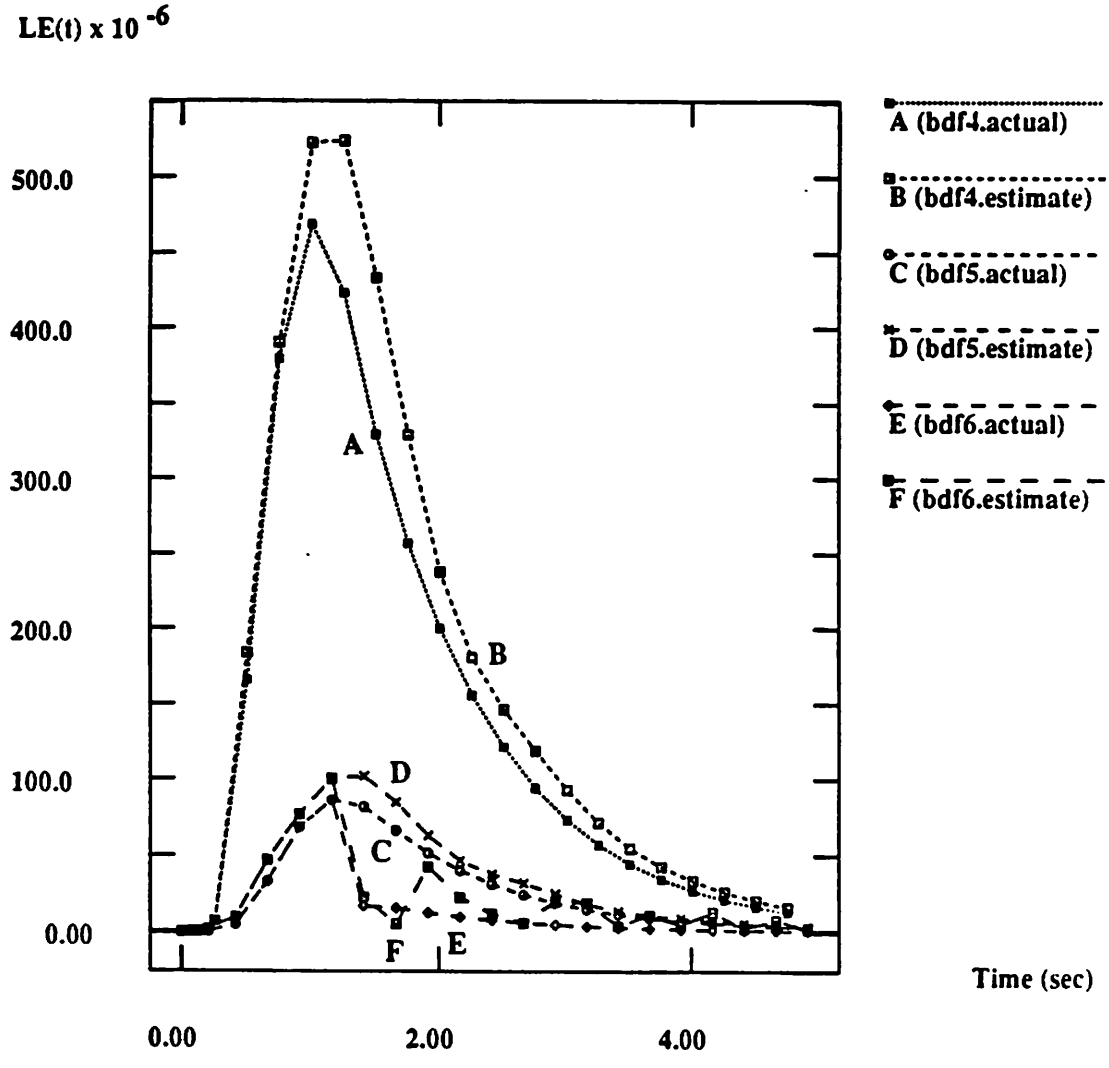


Figure 5.7(b): Estimated and actual values of local error for for the fourth, fifth, and sixth order BDF integration methods.

5.5.5. Timestep and Order Control for Higher-Order BDF

Higher-order BDF methods allow the choice of an integration order along with a new timestep. Thus, both timesteps and integration order can be varied and this must be done in a manner that ensures accuracy and provides an improvement in performance. A method of order k is used at a particular timepoint whenever it allows the largest timestep. The order-selection criterion is based on [5.27].

Assume that the solution at a particular timepoint t_{n+1} has been computed by use of a kth-order integration method, $2 \leq k \leq 5$. If the local-error criterion accepts the timepoint, the local error for the $(k-1)$th-order method is compared to that of the kth-order method. If the lower-order method allows a timestep greater than that of the kth-order method by a factor of *orderDownFactor*, then the $(k-1)$th-order integration method is used for calculating the solution at the next timepoint. When there is no advantage in using the low-order method, the possibility of increasing the integration order is evaluated. If the $(k+1)$th-order method allows a timestep greater than that for the kth-order method by a factor *orderUpFactor* then order $k+1$ is selected for computing the solution at the next timepoint. Obviously for the first-order method only an increase in order is attempted, whereas for the sixth-order method only a decrease in order is possible. The condition for changing orders is evaluated after every $k+1$ timepoints have been computed with the kth-order integration method.

The timestep-control scheme is the same as that used for the second-order BDF. To compare the performance of variable timestep, variable integration order schemes a non-linear device-level problem has been solved. A transient simulation is performed on a MOSFET connected as a capacitor, the source, drain, and substrate terminals are grounded and the gate voltage is ramped from +5V to -5V in 10nsec; the MOS-gate capacitor-voltage characteristics can be determined. In Table 5.8, the results obtained for various values of the relative tolerance parameter ε_r for an *orderUpFactor* of 1.2 and an

orderDownFactor of unity are compared. The latter condition ensures that a lower order will be used whenever two integration orders allow the same timestep. The nonlinear equations are solved with a relative error tolerance of ε_r, whereas the local-error check uses a relative error tolerance of $10\varepsilon_r$. The data is given in the sequence: number of accepted timepoints, number of rejected timepoints, number of iterations, and analysis time in seconds on a VAX 8800. No statistics are given for a maximum order of 6 since the highest order that was used was 5. From Table 5.8 it is seen that for tighter error tolerances the higher-order methods provide some speedup. However, going beyond the third-order method does not provide significant improvement. For an $\varepsilon_r = 1 \times 10^{-5}$, a maximum order of five results in a poorer performance compared with a maximum order of three. Thus, methods higher than three do not provide any significant improvement in performance. For looser tolerances, the second-order method is quite adequate.

As additional data, consider the data of Table 5.9 obtained by using an *orderUpFactor* of 2.0 and an *orderDownFactor* of unity. This choice of parameter allows a higher-order method to be used only if the timestep is larger by a factor of two compared with the lower-order integration method. With this choice of parameters, it is seen that the highest order used was three. Furthermore, it does not provide any additional speedup compared with the second-order method. In conclusion, a second-order method provides reasonable performance for transient analysis.

5.5.6. Iteration-Domain Latency

At each timepoint the numerical devices converge in a different number of iterations depending on how their terminal voltages vary. If a device converges at a particular operating point and its terminal voltages do not change then there is no need to evaluate the device. This form of latency, called iteration-domain latency, has been found to be useful in event-driven simulation [5.29]. The latency check in CODECS is

Method/Order	$\varepsilon_r = 1\times10^{-3}$	$\varepsilon_r = 1\times10^{-4}$	$\varepsilon_r = 1\times10^{-5}$
Trapezoidal	114	143	218
	22	25	21
	423	530	761
	5948	7882	11155
BDF/Max Order 1	147	334	860
	12	31	50
	444	1081	2714
	6210	15090	37758
BDF/Max Order 2	111	171	291
	11	15	31
	340	551	959
	4656	7950	13178
BDF/Max Order 3	105	137	201
	11	15	29
	324	453	691
	4492	6384	9250
BDF/Max Order 4	105	126	176
	11	16	29
	324	424	619
	4500	5887	8566
BDF/Max Order 5	105	126	199
	11	16	34
	324	424	710
	4500	5887	9768

Table 5.8: Comparison of higher order methods

Method/Order	$\varepsilon_r = 1\times10^{-3}$	$\varepsilon_r = 1\times10^{-4}$	$\varepsilon_r = 1\times10^{-5}$
Trapezoidal	135	153	256
	22	36	18
	452	581	839
	6664	8601	12484
BDF/Max Order 1	147	334	860
	12	31	50
	444	1081	2714
	6210	15090	37758
BDF/Max Order 2	142	185	301
	13	25	21
	433	620	955
	6042	8541	13170
BDF/Max Order 3	142	181	285
	13	27	23
	433	614	916
	6100	8305	12910

Table 5.9: Comparison of higher order methods

based on a similar idea and is also similar to the bypass scheme of SPICE [5.1]. A numerical device is considered to be latent when all the following conditions are met,

(a) the device-level equations have converged

(b) the terminal voltages meet the convergence criterion, i.e., they are within the error tolerances, and

(c) the change of current is also below the tolerance for device currents

The last two conditions can be described by the following equations.

$$|V_k - V_{k-1}| < \varepsilon_r MAX(|V_k|, |V_{k-1}|) + \varepsilon_a$$

$$|I_k - \hat{I}_k| < \varepsilon_r MAX(|I_k|, |\hat{I}_k|) + \varepsilon_a$$

where $\hat{I}_k$ is the the linearized terminal current that corresponds to the voltage V_k. A comparison of the use of the above latency scheme in CODECS is given in Table 5.10. The data is given in the following sequence: total number of iterations, number of timepoints accepted, number of timepoints rejected, total number of timepoints, and the total analysis time in seconds on a VAX 8800.

The improvement in speed obtained by the use of the latency check is summarized as a ratio in Table 5.11.

It is seen that the number of timepoints remain almost the same. An improvement in speed is achieved since the devices that have converged need not be re-evaluated. The circuits *Oscillator*, *Pass*, *MOSinv*, *ChargePump*, have only one numerical device; therefore, there is no improvement in performance. Latency check in the iteration domain is appropriate for circuits in which there are multiple numerical devices. On average a fifty-percent speedup is obtained for these test examples.

Circuit	With Latency Check	No latency Check
Oscillator	18333	18333
	5274	5274
	361	361
	5635	5635
	2352	2353
VCO	5864	5865
	1125	1113
	176	172
	1301	1285
	2911	4467
Invchain	1716	1749
	401	401
	17	17
	418	418
	514	908
Astable	6369	6762
	1465	1473
	181	177
	1546	1650
	1230	1713
MECLgate	2609	2602
	619	619
	31	31
	650	650
	2121	3893
Pass	295	296
	82	82
	8	8
	90	90
	1059	1160
MOSinv	336	327
	95	95
	4	4
	99	99
	1155	1239
ChargePump	1850	1853
	493	495
	73	73
	566	468
	4039	4356

Table 5.10: Statistics with and without latency check

Circuit	Latency Check	No Latency Check	Ratio
Oscillator	2352	2353	1.0
VCO	2911	4467	1.5
Invchain	514	908	1.8
astable	1230	1713	1.4
MECLgate	2121	3893	1.8
Pass	1059	1160	1.1
MOSinv	1155	1239	1.0
ChargePump	4039	4356	1.1

Table 5.11: Comparison of total analysis time

5.5.7. Current-Conservation Property of BDF

In this section it is shown that the space-discretized equations conserve total current when discretized in time by use of the backward-differentiation formula. It is shown in [5.30] that the first-order BDF conserves current. Here the result is generalized to the higher-order BDF. In fact, it can be shown that any linear-multistep integration method will possess this property.

Current conservation is easily demonstrated for a one-dimensional diode, and can be extended to two-dimensional devices. The device equations for grid node i located at x_i can be expressed as

$$\frac{\psi_{i+1} - \psi_i}{h_i} = \frac{\psi_i - \psi_{i-1}}{h_{i-1}} - \left[N + p - n\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.58a}$$

$$J_{n,i+1/2} = J_{n,i-1/2} + \left[\frac{\partial n}{\partial t} - (G - R)\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.58b}$$

$$J_{p,i+1/2} = J_{p,i-1/2} + \left[-\frac{\partial p}{\partial t} + (G - R)\right]_i \frac{h_i + h_{i-1}}{2} \tag{5.58c}$$

The total current for the interval (x_i, x_{i+1}) is the sum of the electron current, the hole current, and the displacement current.

$$J_{T,i+1/2} = J_{n,i+1/2} + J_{p,i+1/2} + J_{d,i+1/2} \tag{5.59}$$

To show current conservation it suffices to show that

$$J_{T,i+1/2} = J_{T,i-1/2} \tag{5.60}$$

since current will be a constant in a one-dimensional diode when the total current is conserved. Recall that the displacement current is related to the electric field. In normalized form

$$J_{d,i+1/2} = \frac{\partial E_{i+1/2}}{\partial t} \tag{5.61a}$$

With the finite-difference scheme for space discretization, described earlier,

$$J_{d,i+1/2} = -\frac{\partial}{\partial t}\left[\frac{\psi_{i+1} - \psi_i}{h_i}\right] \tag{5.61b}$$

From Equation (5.58a) $J_{d,i+1/2}$ can be related to $J_{d,i-1/2}$ as

$$J_{d,i+1/2} = J_{d,i-1/2} + \frac{\partial}{\partial t}\left[N_i + p_i - n_i\right]\frac{h_i + h_{i-1}}{2} \tag{5.62}$$

Now discretizing the time derivatives by use of BDF of order k, $1 \le k \le 6$, at time $t = t_m$, and evaluating $J_{T,i+1/2}$, one obtains

$$J_{T,i+1/2} = J_{T,i-1/2} + \sum_{j=0}^{k}\alpha_j \left[n_i\right]_{m+1-j}\left[\frac{h_i + h_{i-1}}{2}\right] - \sum_{j=0}^{k}\alpha_j \left[p_i\right]_{m+1-j}\left[\frac{h_i + h_{i-1}}{2}\right] +$$

$$\sum_{j=0}^{k}\alpha_j \left[N_i + p_i - n_i\right]_{m+1-j}\left[\frac{h_i + h_{i-1}}{2}\right] \tag{5.63}$$

which can be simplified to

$$J_{T,i+1/2} = J_{T,i-1/2} + \sum_{j=0}^{k}\alpha_j \left[N_i\right]_{m+1-j}\left[\frac{h_i + h_{i-1}}{2}\right] \tag{5.64}$$

The term in the summation is zero, since N_i is a constant for all time points and the integration method satisfies the exactness constraint, i.e., $\sum_{j=0}^{k}\alpha_j = 0$. This gives the desired result; the space-discretized equations exhibit continuity of current when time discretization is performed by the kth-order backward-differentiation formula.

5.6. Small-Signal Ac and Pole-Zero Analyses

Steady-state sinusoidal analysis at the device level was described in Chapter 3. There it was shown that the device-level equations can be assembled in the form

$$\left[\frac{\partial \mathbf{F}}{\partial \mathbf{w}} + j\mathbf{D}(\omega)\right]\tilde{\mathbf{w}}(\omega) = \mathbf{b} \tag{5.65}$$

where $\frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ is the Jacobian matrix of the device-level equations at the dc operating point, and $\mathbf{D}$ is a diagonal matrix with diagonal entries 0, $-\omega A_i$, and ωA_i, corresponding to the Poisson equation, the electron current-continuity equation, and the hole current-continuity equation, respectively. A_i is the area of the region associated with grid node i, and $\tilde{\mathbf{w}} = \mathbf{w}_r + j\mathbf{w}_i$ is the vector of the small-signal internal device variables. The above system of equations can be solved by a SOR-technique as described in Chapter 3, which works well at low frequencies. At higher frequencies it can be shown that the relaxation method will have very slow convergence. For this reason, CODECS uses a Gauss-Siedel relaxation at low frequencies and switches to a direct method whenever the relaxation method fails to converge in five iterations. The direct method solves the complete system of linear coupled equations simultaneously and always obtains the solution. This process is explained in detail below.

At low frequencies, CODECS solves the decoupled system of equations obtained by Gauss-Siedel (GS) relaxation as

$$\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \mathbf{w}_r^{k+1} = \mathbf{b}_r + \mathbf{D}\mathbf{w}_i^k \tag{5.66a}$$

$$\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \mathbf{w}_i^{k+1} = -\mathbf{D}\mathbf{w}_r^{k+1} \tag{5.66b}$$

where k is the iteration count for the relaxation method. The above decoupling has the advantage that $\frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ is available in its LU-factors from the dc operating point solution of the device-level equations; only forward and back substitutions are required in solving $\mathbf{w}_r$ and $\mathbf{w}_i$, which makes the computations extremely efficient. As an example, a fifty-point ac analysis of a single MOSFET circuit took $315sec$ with the GS method, whereas the direct method required $2608sec$. Clearly the GS approach is more efficient and can result in significant savings of CPU time, particularly so for low frequencies and larger circuit examples.

Once the small-signal values of the internal variables have been obtained at a particular frequency ω, the admittances of the device can be calculated at ω. The pseudo-C code below illustrates calculation of the admittance for a numerical one-dimensional diode.

```
computeDiodeAdmittance( pDevice, omega, area, yd )
 /* pDevice is the device pointer */
/* omega is the analysis frequency */
/* area is the diode area */
/* yd is the calculated admittance value */
{
   if( AcAnalysisMethod IS SOR ) {
      zeroRhsVector( pDevice );
       /* store contribution of boundary terms in rhs vector */
      assembleBoundaryRhsTerms( pDevice );
      /* compute SOR solution. if converged OK */
      SORfailed = getSORSolution( pDevice, omega );
      if( SORfailed ) {
         /* SOR didn't converge, switch to direct method */
         AcAnalysisMethod = DIRECT;
      }
   }
   if( AcAnalysisMethod IS DIRECT ) {
      zeroRhsVector( pDevice );
      /* assemble the dc jacobian matrix */
      loadJacobian( pDevice );
      /* store contribution of boundary nodes in rhs */
      assembleBoundaryRhsTerms( pDevice );
      /* assemble the complex diagonal terms for Silicon nodes */
      foreach( node ) {
         if( nodeType IS SILICON ) {
            addComplexTerms( node );
         }
      }
      /* factor and solve matrix */
      spFactor( pDevice->matrix );
      spSolve( pDevice->matrix );
   }
   yd = computeAdmittance( pDevice, omega, area );
}
```

The function *assembleBoundaryRhsTerms* needs further explanation. The right-hand-side vector corresponds to the terms $\frac{\partial \mathbf{F}}{\partial V}$, and an example illustrates the calculation of $\frac{\partial \mathbf{F}}{\partial V}$ for a one-dimensional numerical diode. The diode is discretized in space as shown in Figure 5.8 using $L+1$ grid points. For an ohmic contact, the potential

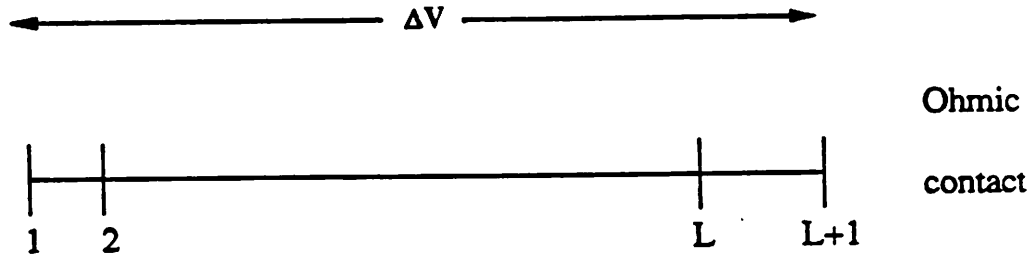


Figure 5.8: One-dimensional space discretization for a diode.

at node $L+1$ is given by

$$\psi_{L+1} = \psi_{0,L+1} + V \tag{5.67}$$

where $\psi_{0,L+1}$ is the equilibrium potential and V is the applied voltage. Only the equations at node L have a dependence on V, through the dependence of ψ_{L+1} on V; therefore, only the entries corresponding to node L are nonzero in the right-hand-side vector. The equations at grid node L are given by

$$F_\psi = \frac{\psi_{L+1} - \psi_L}{h_L} - \frac{\psi_L - \psi_{L-1}}{h_{L-1}} + (N_L + p_L - n_L)\frac{h_L + h_{L-1}}{2} = 0 \tag{5.68a}$$

$$F_n = J_{n,L+1/2} - J_{n,L-1/2} + \left[-\frac{\partial n}{\partial t} + (G - R) \right]_L \frac{h_L + h_{L-1}}{2} = 0 \tag{5.68b}$$

$$F_p = J_{p,L+1/2} - J_{p,L-1/2} + \left[\frac{\partial p}{\partial t} - (G - R) \right]_L \frac{h_L + h_{L-1}}{2} = 0 \tag{5.68c}$$

From these equations one obtains,

$$\frac{\partial F_\psi}{\partial V} = \frac{\partial F_\psi}{\partial \psi_{L+1}} = \frac{1}{h_L} \tag{5.69a}$$

$$\frac{\partial F_n}{\partial V} = \frac{\partial F_n}{\partial \psi_{L+1}} = \frac{\partial J_{n,L+1/2}}{\partial \psi_{L+1}} \tag{5.69b}$$

$$\frac{\partial F_p}{\partial V} = \frac{\partial F_p}{\partial \psi_{L+1}} = \frac{\partial J_{p,L+1/2}}{\partial \psi_{L+1}} \tag{5.69c}$$

These derivative terms are used to assemble the right-hand-side vector. Recall, that the boundary terms are also assembled in the right-hand-side vector for calculating conductances for dc and transient analyses and this is done in an identical manner.

The pole-zero analysis is simply an extension of the small-signal ac analysis technique. Instead of using a frequency ω, a complex frequency $s = \sigma + j\omega$ is used. Small-signal analysis then results in a linear system of equations that can be expressed as

$$\left[\frac{\partial \mathbf{F}}{\partial \mathbf{w}} + j\mathbf{D}(s)\right]\tilde{\mathbf{w}} = \mathbf{b} \tag{5.64}$$

where $\mathbf{D}(s)$ is a diagonal matrix with 0, $-sA_i$, and sA_i as the diagonal entries for grid node i. This system of equations is assembled, and solved by a direct method for a given frequency s. Once $\tilde{\mathbf{w}}(s)$ has been obtained, the admittance $Y(s)$ is calculated in a manner similar to the calculation of admittances described earlier for $s = j\omega$. A SOR-solution is not attempted in this case, since the system of equations cannot be partitioned as with $s = j\omega$. These admittances are then used in the circuit-level equations to determine the poles and zeros of the circuit transfer function.

5.7. Calculation of Sensitivity to Doping Profiles

Sensitivity calculations have been found to be extremely useful at the circuit level of simulation where they allow a designer to determine the elements of the circuit that are critical to the performance. By estimating the sensitivity of the response to all the elements in the circuit, critical components can be identified and tuned to achieve better performance. In addition, sensitivity calculations provide a basis for gradient-based optimizations [5.31]. A similar capability would be useful at the device level as has been indicated in [5.32] where device-level sensitivities have been implemented under dc conditions. Differential-sensitivity calculations are particularly useful for device simulation because repeated simulation of a device with different doping profiles requires significant computational effort, particularly for two and three-dimensional devices. On the other hand, as illustrated later in this section, sensitivities are quite inexpensive to evaluate and provide a first-order analysis of the tradeoffs involved in device design.

The sensitivity problem is formulated as a problem for calculation of transient sensitivities. The dc sensitivities are then obtained as a special case of the transient problem by setting the time-derivative terms to zero. Calculation of sensitivities is done by a direct-differentiation approach which has been preferred for use in circuit simulators [5.33, 5.34].

At the device-simulation level, the system of equations that is solved can be represented as

$$\mathbf{F}(\dot{\mathbf{w}}(t), \mathbf{w}(t), V(t)) = \mathbf{O} \tag{5.71}$$

where $\mathbf{w}(t)$ is the vector of internal variables and $V(t)$ are the applied terminal voltages. For assumed analytical doping profiles, let u be a doping-profile parameter; u is the peak concentration or characteristic length for a Gaussian profile, or the concentration for a uniform doping. The implicit dependence of $\mathbf{w}$ on u can be incorporated in the above

equation, which can then be expressed as

$$\mathbf{F}(\dot{\mathbf{w}}(t, u), \mathbf{w}(t, u), u, V(t)) = \mathbf{O} \tag{5.72}$$

To compute differential sensitivities this equation is differentiated with respect to the doping-profile parameter u to obtain

$$\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}} \frac{\partial \dot{\mathbf{w}}}{\partial u} + \frac{\partial \mathbf{F}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial u} + \frac{\partial \mathbf{F}}{\partial u} = \mathbf{O} \tag{5.73}$$

Since

$$\frac{\partial \dot{\mathbf{w}}}{\partial u} = \frac{\partial}{\partial u} \frac{\partial \mathbf{w}}{\partial t} = \frac{\partial}{\partial t} \frac{\partial \mathbf{w}}{\partial u} \tag{5.74}$$

Equation (5.73) can be rewritten as

$$\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}} \frac{\partial}{\partial t} \frac{\partial \mathbf{w}}{\partial u} + \frac{\partial \mathbf{F}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial u} + \frac{\partial \mathbf{F}}{\partial u} = \mathbf{O} \tag{5.75}$$

which is a system of linear differential equations in $\frac{\partial \mathbf{w}}{\partial u}$, the sensitivities of the internal variables to the parameter u. The initial condition for this system of differential equations is obtained by setting the time derivative term to zero, i.e.,

$$\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \left[\frac{\partial \mathbf{w}}{\partial u} \right]_0 = - \frac{\partial \mathbf{F}}{\partial u} \tag{5.76}$$

The above equation also provides the dc sensitivities $\left[\frac{\partial \mathbf{w}}{\partial u} \right]_0$. $\frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ is available in LU-factors from the solution of the device-level equations under dc conditions and only forward and back substitutions are required to calculate the dc differential sensitivities.

When a kth-order backward-differentiation formula ($1 \le k \le 2$) is used for time discretization at time $t = t_{n+1}$, $\frac{\partial}{\partial t} \frac{\partial \mathbf{w}}{\partial u}$ can be expressed as

$$\frac{\partial}{\partial t}\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} = \frac{1}{h_n}\sum_{i=0}^{k} a_i \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1-i} = \sum_{i=0}^{k} \alpha_i \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1-i} \tag{5.77}$$

or,

$$\frac{\partial}{\partial t}\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} = \alpha_0 \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} + \sum_{i=1}^{k} \alpha_i \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1-i} \tag{5.78}$$

Substitution of the discrete approximation for the time derivative terms in Equation (5.75) results in

$$\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}}\left[\alpha_0 \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} + \sum_{i=1}^{k} \alpha_i \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1-i}\right] + \frac{\partial \mathbf{F}}{\partial \mathbf{w}}\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} + \frac{\partial \mathbf{F}}{\partial u} = \mathbf{O} \tag{5.79}$$

Assembling the terms in $\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1}$, one obtains

$$\left[\alpha_0 \frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}} + \frac{\partial \mathbf{F}}{\partial \mathbf{w}}\right]\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1} = -\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}}\sum_{i=1}^{k} \alpha_i \left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1-i} - \frac{\partial \mathbf{F}}{\partial u} \tag{5.80}$$

The above equation is a system of linear equations that is solved once the device-level equations have converged at time point t_{n+1}. $\left[\alpha_0 \frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}} + \frac{\partial \mathbf{F}}{\partial \mathbf{w}}\right]$ is the Jacobian used for the solution of the device-level equations at time point t_{n+1}. Hence, it is available in its LU factors and $\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n+1}$ can be obtained by forward and back substitutions, which are not computationally expensive.

It should be noted that $\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{w}}}$ is a diagonal matrix with diagonal entries 0, $-A_i$, and A_i corresponding to Poisson's equation and the electron and hole current-continuity equations, respectively. Equation (5.80) requires values of the differential sensitivities from the previous timepoints, i.e., values of $\left[\frac{\partial \mathbf{w}}{\partial u}\right]_n$ and $\left[\frac{\partial \mathbf{w}}{\partial u}\right]_{n-1}$; therefore, these quantities must be stored.

5.7.1. Implementation of Sensitivity Calculations

As seen from the previous derivation the sensitivities of all internal variables at previous timepoints have to be stored; they are required for integrating the sensitivity differential equation. Thus, in addition to storing the values of ψ_i, n_i and p_i at grid node i, the quantities, $\frac{\partial \psi_i}{\partial u}$, $\frac{\partial n_i}{\partial u}$, and $\frac{\partial p_i}{\partial u}$ have to be stored at previous timepoints in the state vector. This leads to an increase in the size of the state vector by a factor of two.

Once the right-hand-side vector for the sensitivity equation has been assembled the sensitivities can be obtained by forward and back substitutions. The right-hand-side vector is made up of two terms: the first one is due to the time discretization of the sensitivity equation and the other term is due to $\frac{\partial \mathbf{F}}{\partial u}$. The second term requires computing the derivatives of the equations at each grid node with respect to the doping-profile parameter u. Poisson's equation has a direct dependence on doping through the term $N_A - N_D$. The dependence on doping enters the current-continuity equations through the carrier mobility and carrier lifetime dependencies on doping. The band-gap narrowing term is also doping dependent and should be taken into account. However, for the present implementation, this dependence has been ignored and only the doping dependent mobilities and lifetimes are accounted for.

Variations in the doping also change the electrostatic potential and electron and hole concentrations at the contact nodes. For ohmic-contact nodes the sensitivities can be easily calculated from equilibrium conditions and the charge-neutrality condition. It can be shown for the ohmic contact that,

$$\frac{\partial \psi}{\partial N} = \frac{1}{n_0 + p_0} \tag{5.81a}$$

$$\frac{\partial n}{\partial N} = \frac{n_0}{n_0 + p_0} \tag{5.81b}$$

and

$$\frac{\partial p}{\partial N} = \frac{p_0}{n_0 + p_0} \tag{5.81c}$$

where N is the net doping, and n_0 and p_0 are the equilibrium electron and hole concentrations at the contact.

Once the sensitivity equations have been solved $\frac{\partial \psi_i}{\partial u}$, $\frac{\partial n_i}{\partial u}$, and $\frac{\partial p_i}{\partial u}$ are known at each grid point. From this information the sensitivities of terminal currents to the doping can be determined. Thus, for given terminal voltages one can determine the change in terminal current for a change in the doping within some region of the device.

5.7.2. Sensitivity Simulation Examples

A pn-junction diode with the doping profile shown in Figure 5.9 is used as the example. The dc sensitivity of diode current to the doping level in the lightly doped region, N_D, as a function of voltage is plotted in Figure 5.10. The sensitivity plot indicates that initially the current decreases with an increase in N_D, but at higher voltages an increase in N_D will result in an increase of the current. This behavior can be explained by examining the operation of the diode under low-level and high-level-injection conditions. Under low-level injection an increase in N_D reduces the amount of injected carriers and hence one can expect a reduction in the current. However, under high-level-injection conditions, which occur at higher voltages, the current increases with an increase in N_D; the current roll-off occurs at a higher voltage. In Figure 5.10 are also shown the sensitivities of diode current to N_D obtained by use of repeated simulations for 1%, 10%, and 100% variations in the doping level. The agreement with the 1% and the 10% deviations is extremely good, showing the accuracy of the sensitivity calculations. The 100% deviations show a trend similar to that obtained by the differential-

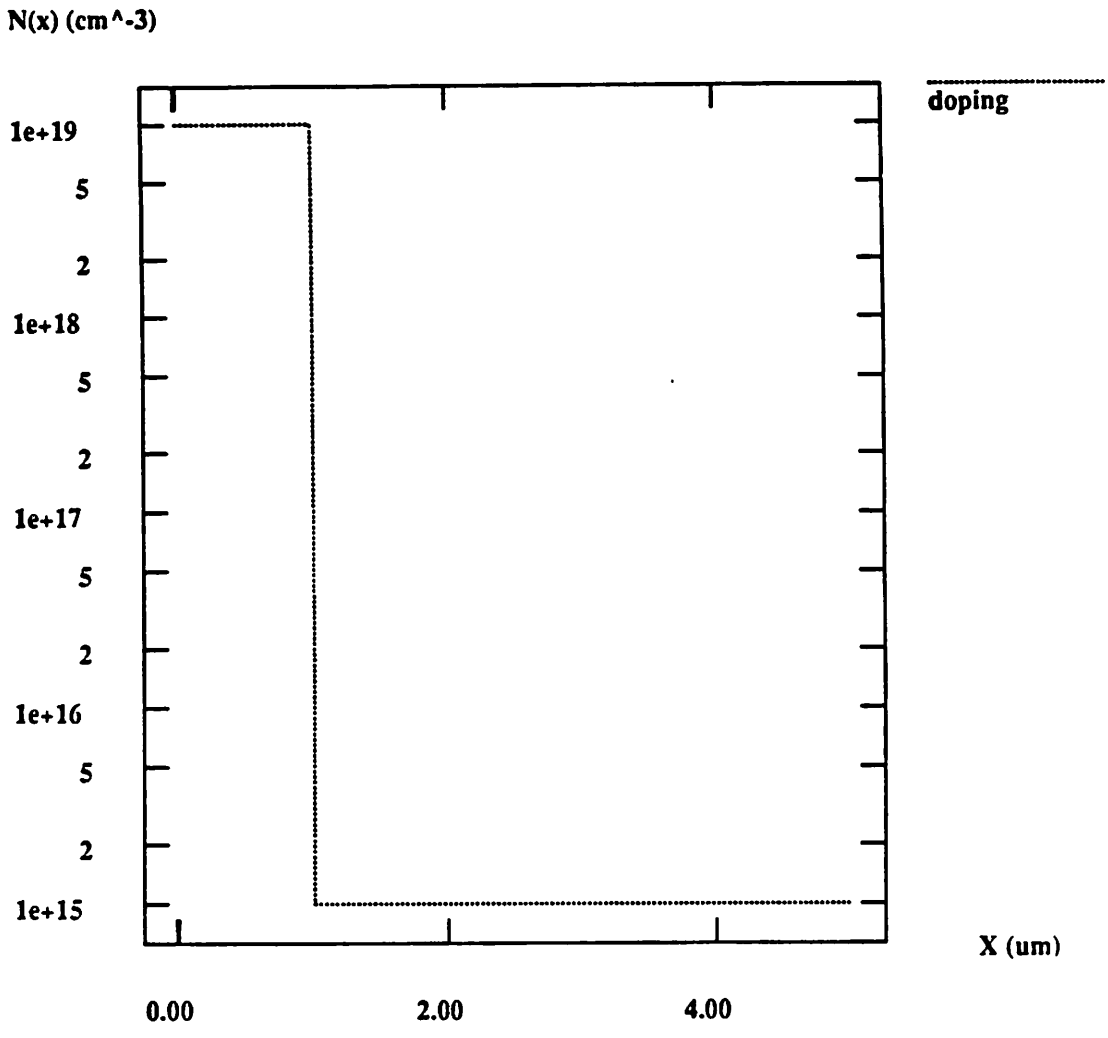


Figure 5.9: Doping profile for one-dimensional diode.

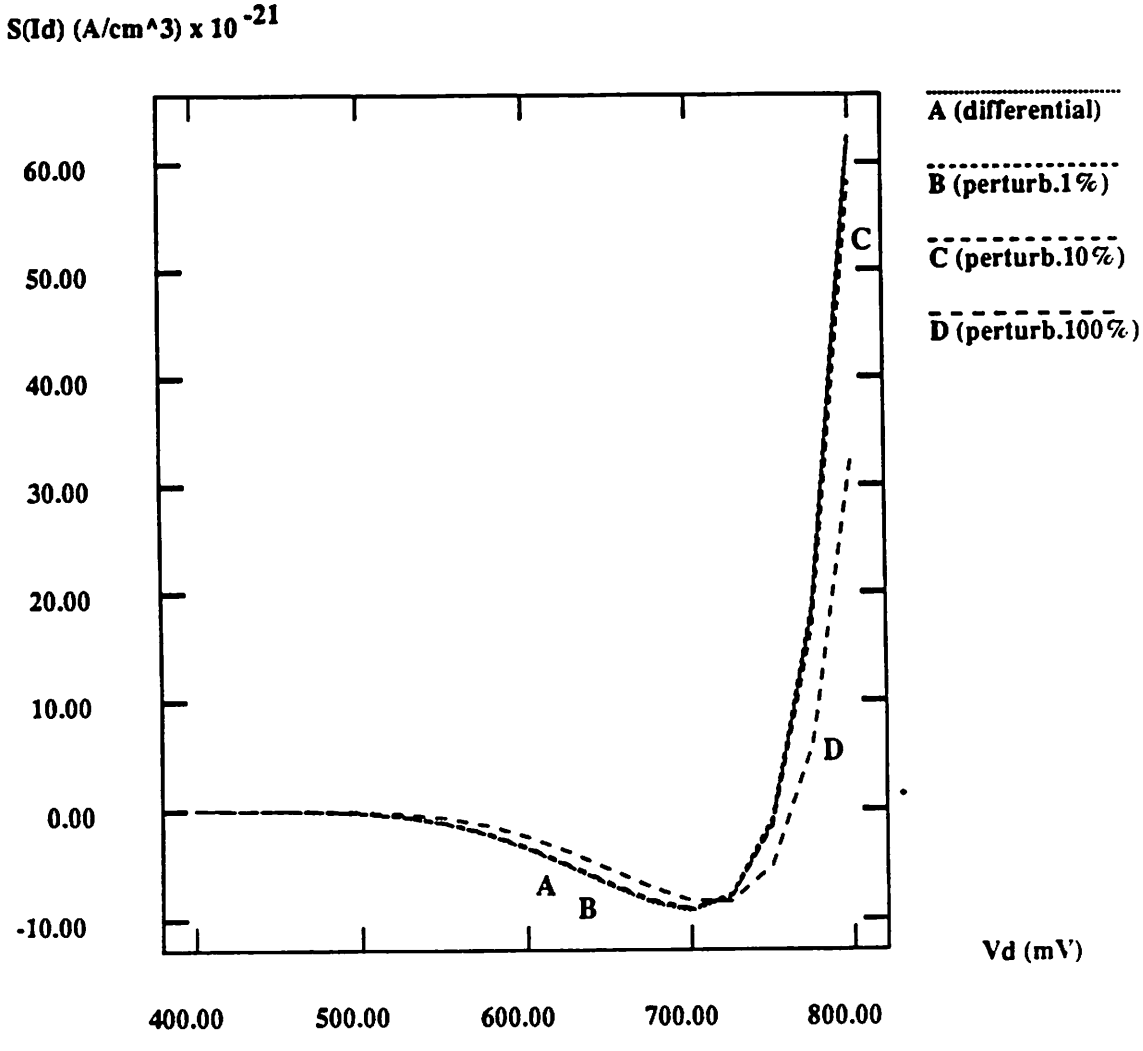


Figure 5.10: Dc sensitivities of diode current to the doping level obtianed from sensitivity computations and solutions with 1%, 10%, and 100% perturbations in the doping level.

sensitivity calculations; however, the values are quite different.

For transient analysis, the error made in time discretization of the sensitivity equations should be taken into account while calculating an acceptable timestep. In circuit simulation, the error control on circuit equations provides reasonable accuracy for the sensitivity equations and the timestep selection is done based only on the circuit-level equations [5.33]. It is not clear as to what the behavior would be for device-level equations and this is investigated. A error check has also been implemented for the sensitivity equations. In Figure 5.11 the transient sensitivity of diode current is plotted as a function of time, after the diode is turned off, by monitoring the error in the time discretization of the sensitivity equations. For comparison the result obtained with only a check on the device-level equations is also shown. It is seen that the sensitivity calculations with timestep control are different from those without the timestep control for the sensitivity equations. However, the difference is not that significant and it is expected that timestep control of the device-level equations should provide reasonably accurate sensitivity calculations. This results in smaller runtimes as illustrated in Table 5.12, where the runtime statistics for the two cases are compared.

Statistics	without LE check	with LE check
Number of iterations	154	316
Timepoints accepted	75	119
Timepoints rejected	3	40
Analysis time (s)	42	87

Table 5.12: Statistics for local-error control

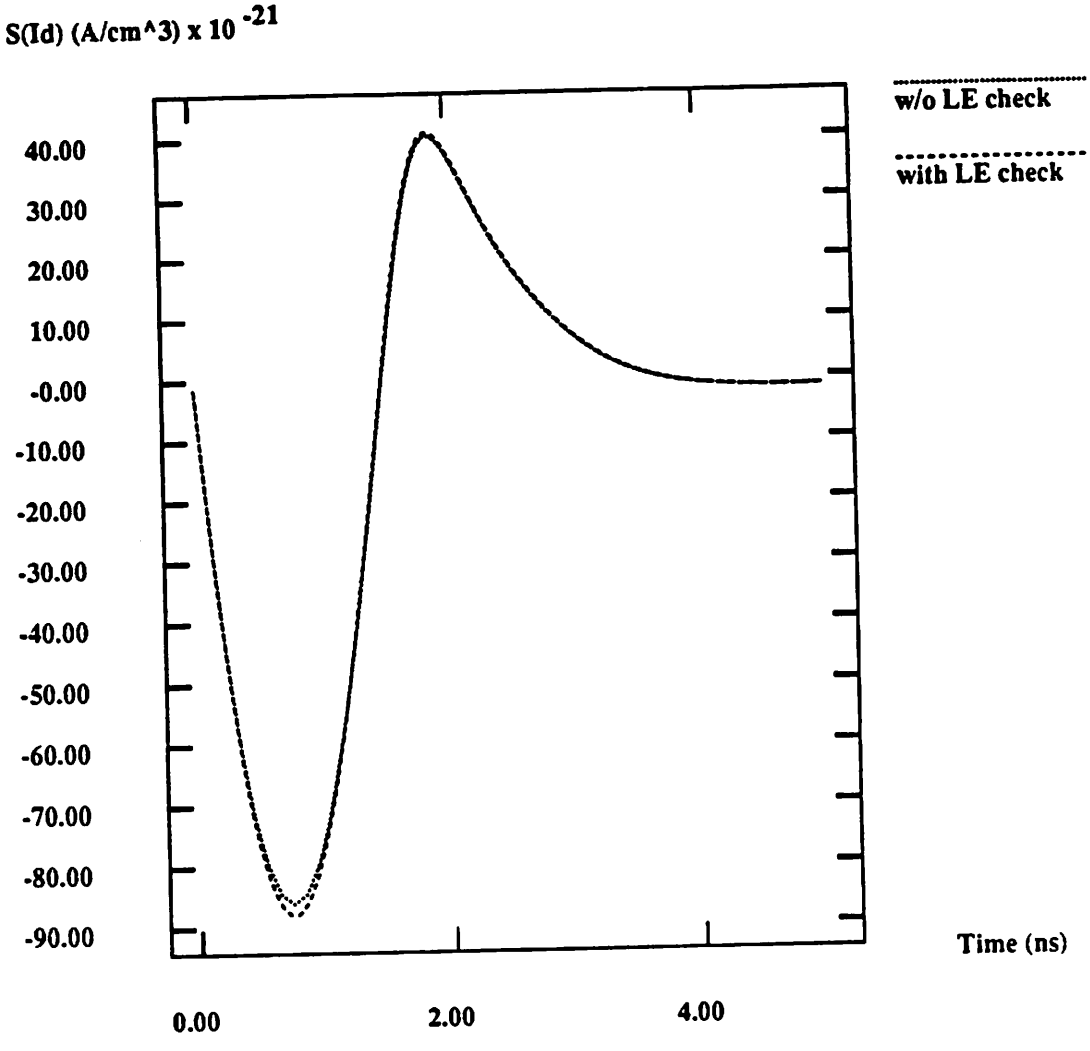


Figure 5.11: Differential sensitivity for transient simulations obtained by use of a local-error control on the sensitivity equations compared to that without the local-error control.

Finally, a comparison is made with the sensitivities calculated by use of repeated simulations with 1%, 10%, and 100% variations in the doping profiles in Figure 5.12. Good agreement is achieved with the 1% and 10% variation results. The 100% variation result differs by about 50% from the differential-sensitivity calculation.

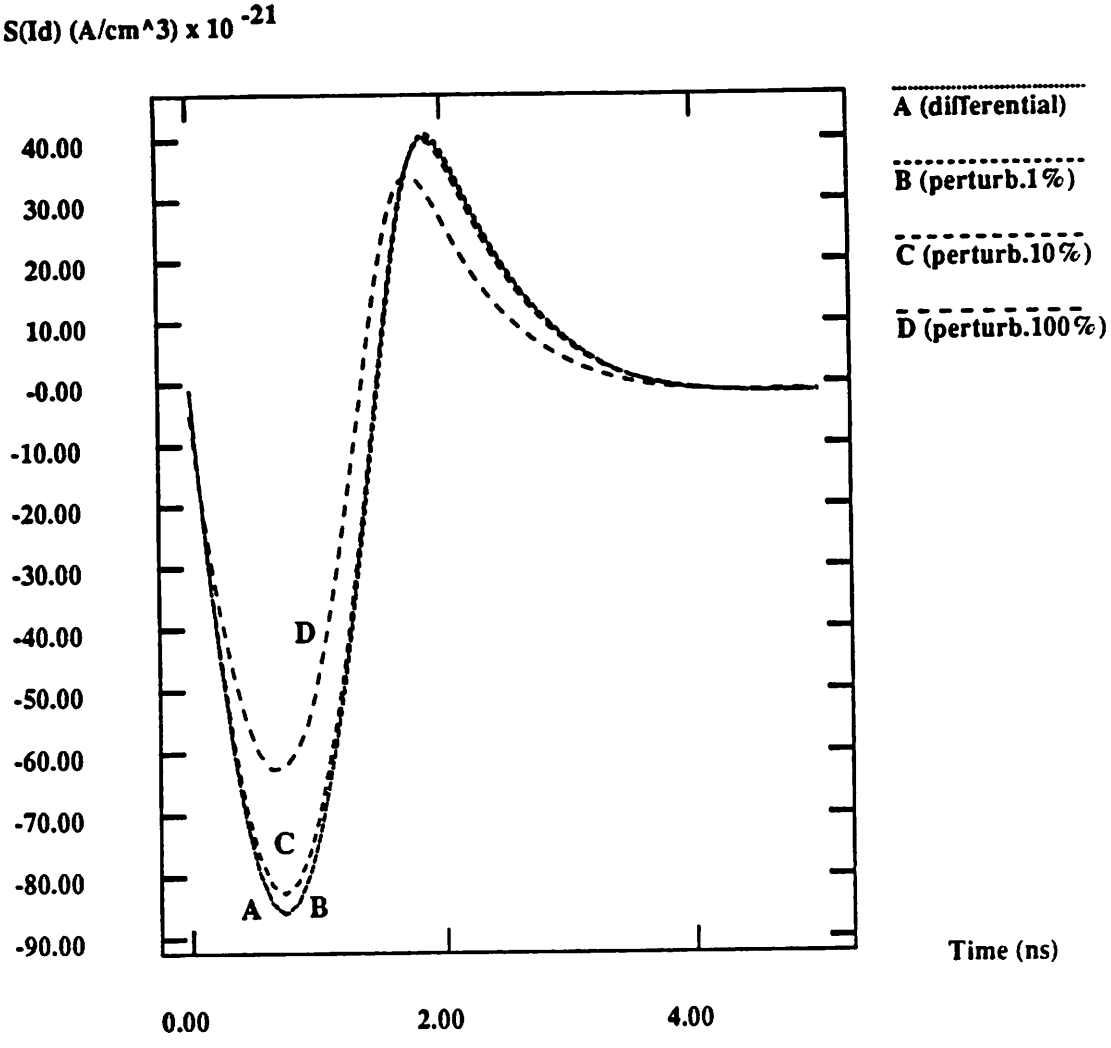


Figure 5.12: Transient sensitivities of diode current to the doping level obtianed from sensitivity computations and solutions with 1%, 10%, and 100% perturbations in the doping level.

CHAPTER 6

CODECS Design Considerations

6.1. Introduction

An initial prototype of CODECS was implemented in Lisp using object-oriented programming. The Lisp programming language was used because of enhanced software productivity. Furthermore, object-oriented programming allowed easy coupling between the circuit and device simulators. Previous experience with Lisp-based circuit simulation [6.1-6.3] indicated that there would not be a significant performance penalty on a Lisp machine.

Consequently, a one- and two-dimensional device-simulation capability was developed in Lisp. Object-oriented programming provides a simple interface between the device simulator and the circuit simulator. The runtime performance of the mixed-level circuit and device simulator is extremely poor, even on a Symbolics 3600 Lisp machine [6.4], as illustrated in this chapter. Benchmark examples are used to evaluate the runtime performance. Detailed profiling of one- and two-dimensional simulation examples is provided to identify the bottlenecks in the program. It is shown that adequate performance cannot be achieved on the Lisp machine.

As a result of the above study, a new version of CODECS has been developed in the C programming language. A new one- and two-dimensional device-simulation capability has been written in C. SPICE3 [6.5] has been incorporated for the circuit-simulation capability and for analytical models of semiconductor devices. A comparison

is made between the runtime performances of the Lisp- and C-based versions of CODECS. The C-based version has a significantly improved performance, particularly for two-dimensional simulation examples.

6.2. Lisp-based implementation - CODECSlisp

A prototype of CODECS was developed using object-oriented programming in Lisp. The program is written in Zetalisp; object-oriented programming is available through the *Flavors* system [6.6]. The choices that led to the prototype being developed in Lisp were: high software productivity, rapid prototyping, availability of object-oriented programming, and adequate performance for circuit simulation on the Symbolics 3600 Lisp machine [6.1-6.3].

The two main components of a mixed-level circuit and device simulation environment are a circuit simulator and a device simulator. For the circuit-simulation component CODECSlisp makes use of BIASlisp, an object-oriented circuit simulator [6.3]. A new one- and two-dimensional device simulation capability DSIM has been developed in Lisp using object-oriented programming. The features of CODECSlisp, the choice of objects, and the coupling between BIASlisp and DSIM are described in [6.7].

6.2.1. Runtime Performance

This section examines the runtime performance of CODECSlisp for transient analysis using one- and two-dimensional numerical models. Detailed profiling is provided to identify the portions where the program spends most of its time.

The first set of examples make use of one-dimensional numerical models for bipolar transistors and are the *Invchain* and *VCO* examples introduced in Chapter 4. Results for transient analysis of these two circuits are presented in Table 6.1 for the Symbolics 3600 Lisp machine without hardware floating-point support.

Description	Invchain	VCO
# iterations	2211	6284
# timepoints accepted	446	1108
# timepoints rejected	43	265
Analysis time (hrs)	31.5	152.9

Table 6.1: Runtime performance of CODECSlisp

It is seen that even for the small one-dimensional examples the simulation times are in the order of days (1.3 and 6.4 days, respectively). This is a significant amount of time and makes CODECSlisp unsuitable for practical use.

A breakup of the total analysis time is given in Table 6.2 for the device-level computations during the transient analysis. *LUDecomp* is the time taken to perform the LU decomposition of the device-level matrices. *Solve* is the time required for forward and back substitutions which are used in obtaining the solution of the device-level equations and for calculating the terminal conductances. *LoadJacobianRhs* is the time taken for loading the device-level Jacobian matrices and the right-hand-side vectors. *CalcCurrDeriv* is the time required for calculating the current densities and recombination rates and their derivatives. These tasks are performed by calls to three other functions which are *Bernoulli* to evaluate the Bernoulli function and its derivatives, *Mobility* to calculate the field-dependent mobility values and their derivatives, and *Recomb* to compute the recombination rates and their derivatives. *LocalError* is the time spent in calculating the local error during transient analysis and in estimating a new timestep.

It is seen that a major portion of the total time is spent in the equation assembly phase, i.e., in *CalcCurrDeriv* and *LoadJacobianRhs*. This phase takes about 57% of the total simulation time with calculation of the currents and derivatives taking approximately 43% of the total time. In the process of computing the current densities and their

Description	Invchain Time (hrs)	VCO Time (hrs)
LUDecomp	5.0 (15.7%)	24.25 (15.9%)
Solve	5.9 (18.7%)	28.85 (18.9%)
LoadJacobianRhs	4.54 (14.4%)	22.2 (14.5%)
CalcCurrDeriv	13.5 (42.9%)	65.56 (42.9%)
Bernoulli	3.67	17.8
Mobility	3.32	15.9
Recomb	2.43	11.8
LocalError	0.14 (0.4%)	0.63 (0.4%)

Table 6.2: Detailed runtime profile of CODECSlisp

derivatives, 70% of the time is spent in evaluating the Bernoulli function, the carrier mobilities, and the recombination rates. The remaining 30% is spent in computing the time-dependent terms and in the computation of all current densities and their derivatives. LU decomposition takes up 16% of the total time with forward and back substitutions taking up another 19%. The remaining 8% of the total simulation time is spent in solving the circuit-level equations, calculating the integration and prediction coefficients, and in performing convergence checks. All simulation times are reported with the garbage collection [6.1] turned on and include the overhead of automatic reclamation of unused memory space. It is impossible to run any of these examples with garbage collection turned off.

The other example is a series resistor diode circuit; the diode is modeled by a two-dimensional numerical device. Three transient simulations have been performed with the diode modeled by 21x3, 51x3, and 101x3 grid points, respectively. The results are reported in Table 6.3. It is seen that as the equations increase in the ratio 1 : 2.5 : 5, the runtimes increase in the ratio 1 : 3.1 : 8.9, i.e., a super-linear increase with the number

Description	21x3	51x3	101x3
# iterations	412	436	444
# timepoints accepted	113	116	107
# timepoints rejected	14	18	30
Analysis time (hrs)	3.7	11.56	32.8

Table 6.3: Runtime performance for diode example

of equations. The detailed breakup of the total simulation time is presented in Table 6.4.

Description	21x3 Time (sec)	51x3 Time (sec)	101x3 Time (sec)
LUDecomp	6385 (47.7%)	20260 (48.7%)	66077 (56%)
Solve	2020 (15.1%)	6950 (16.7%)	17148 (14.5%)
LoadJacobianRhs	1243 (9.3%)	3720 (8.9%)	9456 (8.0%)
CalcCurrDeriv	3223 (24.0%)	9664 (23.0%)	22996 (19.5%)
Bernoulli	802	2423	5694
Mobility	917	2676	6369
Recomb	27	1278	3113
LocalError	27 (0.2%)	79 (0.2%)	220 (0.2%)

Table 6.4: Detailed runtime profile of CODECSlisp

For the above example, LU decomposition takes almost 50% of the total simulation time. The matrices are larger in size and hence the matrix-decomposition time becomes significant. For the one-dimensional examples, assembly of the device-level equations requires a larger amount of CPU time. For two-dimensional examples one can expect the LU-decomposition time to be important since the matrices are significantly larger in size.

An idea of the penalty in performance due to the use of Lisp can be obtained by translating the functions *Bernoulli*, *Mobility*, and *Recomb* into C. As an additional data

point, the above functions are also compiled for the VAX Common Lisp on a VAX 8650. The numbers in parenthesis for VAX Common Lisp are obtained when the data types of all variables are declared explicitly and all possible compiler optimizations are used. A comparison of the time per call, for the three cases, is given in Table 6.5.

Function	Zetalisp (Symbolics 3600) (msec)	Common Lisp (VAX 8650) (msec)	C (VAX 8650) (msec)
Bernoulli	3.3	0.82 (0.34)	0.08
Mobility	5.7	0.9 (0.6)	0.12
Recomb	2.6	0.68 (0.3)	0.035

Table 6.5: Comparison of Zetalisp, Common Lisp and C

The same data is presented in Table 6.6 with the results normalized to the runtimes for the C code. It is seen that the VAX Common Lisp with all possible optimizations is significantly better than Zetalisp on the Symbolics 3600, but even with carefully optimized code one cannot get the same performance as with C. The Symbolics 3600 is roughly a 1.5 MIPS machine whereas the VAX 8650 is a 6 MIPS machine. If one takes into account the difference in the MIPS rating of the machines, the results for the Symbolics 3600 should be scaled down by a factor of four. Then, the Symbolics 3600 without floating-point hardware is about 10-20 times slower than an equivalent C code. The results for VAX Common Lisp indicate that the Lisp code is a factor of 5-10 slower than an equivalent code in C even after extensive optimizations.

From the above performance analysis it appears that Common Lisp on the VAX 8650 may perform better compared to Zetalisp on the Symbolics 3600 Lisp machine on all examples. However, this is not the case when object-oriented programming is used. The object-oriented system for Common Lisp is Portable Common Loops [6.8] and has

Function	Zetalisp (Symbolics 3600) / C (VAX 8650)	Common Lisp (VAX 8650) / C (VAX 8650)
Bernoulli	41.5	10.3 (4.3)
Mobility	47.5	7.5 (5.0)
Recomb	74	19.5 (8.6)

Table 6.6: Normalized Performance

been used to evaluate the performance of BIASlisp on one example. The example is the dc operating point analysis of an NMOS depletion-load inverter chain, with twenty-five inverters. The runtime statistics are presented in Table 6.7. *MatrixLoad* is the time required for assembling the modified-nodal admittance (MNA) matrix. *LUDecomp* is the time required to decompose the MNA matrix into LU factors and *Solve* is the time for obtaining the solution by forward and back substitutions.

Description	BIASlisp (Symbolics 3600)	Common Lisp (VAX 8650)
# iterations	40	40
Analysis time (sec)	65	235
MatrixLoad (sec)	60	229
LUDecomp (sec)	0.8	0.4
Solve (sec)	2.2	2.9

Table 6.6: Comparison of BIASlisp under Zetalisp and Common Lisp

Matrix-assembly time is the most significant in this example. This is the time spent in calculating and loading the MOSFET conductances and currents. The calculation of these quantities is done by message passing; the overhead in Common Lisp on a VAX 8650 is quite significant. The times for LU decomposition and solve are comparable since these parts of the program are not coded using object-oriented programming. Thus, adequate

performance is not obtained for Lisp on a VAX 8650. There is a definite penalty in performance by use of Lisp and for this reason CODECS is written in the C programming language.

6.3. C-based implementation of CODECS

As shown in the previous sections the performance of CODECSlisp is extremely poor which makes it unsuitable for any practical applications. In particular, it is impossible to simulate circuits with two-dimensional numerical devices. For this reason, CODECS has been re-written in the C programming language. The new version of CODECS has been used as the test bed for evaluating the algorithms described in Chapters 4 and 5. SPICE3, also written in the C language, provides the circuit-simulation capability for CODECS. The advantages in using SPICE3 are: modular structure, the capability to add new device models, and the availability of a wide variety of analytical models and analysis capabilities. A new one- and two-dimensional device simulator has been developed to provide the capability for simulating numerical devices. The numerical devices are embedded as additional devices and models within SPICE3. In addition to using SPICE3, CODECS also makes use of the sparse-matrix package Sparse1.3 [6.9]. Since Sparse1.3 is a highly optimized package for sparse-matrix manipulations, additional improvement in performance has been obtained.

The architecture and algorithms of CODECS have been presented in Chapters 4 and 5. One major difference between CODECS and CODECSlisp has been the use of an element-based assembly of equations. The partial-differential equations are discretized on a rectangular grid using the box-integration method. Assembling the equations on a rectangle by rectangle basis allows simulation of arbitrary two-dimensional rectangular structures. CODECSlisp was fairly limited in terms of its capabilities of simulating rectangular geometries. The new C-based two-dimensional simulation capability also allows

one-carrier simulation, thereby, providing an efficient technique for simulation of MOS devices.

6.3.1. Runtime Performance and Comparisons to CODECSlisp

The performance of the C-based version of CODECS is examined in this section. The examples are the same as those used in the evaluation of CODECSlisp. In Table 6.7 are presented the results for the *Invchain* and *VCO* examples. The times reported are for the two-level Newton algorithm without latency checking so that comparisons can be made with the runtime statistics of CODECSlisp. The hardware used is a VAX 8650 computer with hardware floating-point arithmetic.

Description	Invchain	VCO
# iterations	1611	5220
# timepoints accepted	404	1091
# timepoints rejected	18	160
Analysis time (sec)	1904	9617

Table 6.7: Runtime performance of CODECS

The above runtimes indicate significant improvement in performance over CODECSlisp, approximately a factor of sixty. The number of iterations are different and this is possible due to the differences in equation assembly and the sparse-matrix package. Furthermore, the base boundary conditions in the new version of CODECS have been implemented in the manner described in Chapter 5 and this could also result in a different number of iterations. If one considers the time per iteration as a measure of the raw speed of the two programs, the improvement in performance is approximately a factor of forty-five. The above runtimes were obtained with profiling turned on. However, with profiling turned off the runtimes are 1655 seconds and 8210 seconds, respectively;

the overheard of profiling is approximately twelve percent.

A breakup of the total simulation time is shown in Table 6.8. It is seen that the major part of the simulation time is spent in assembling the device-level matrix; this is similar to the conclusion derived from CODECSlisp. However, the matrix-decomposition time and the time for forward and back substitutions is smaller due to the use of the optimized sparse-matrix package Sparse1.3.

Description	Invchain Time (sec)	VCO Time (sec)
LUDecomp	187 (9.8%)	983 (10.2%)
Solve	148 (7.7%)	750 (7.8%)
LoadJacobianRhs	142 (7.5%)	740 (7.7%)
CalcCurrDeriv	897.5 (47%)	4578 (47.6%)
Bernoulli	223	1164
Mobility	318	1579
Recomb	68	345
LocalError	15.9 (0.8%)	66.5 (0.7%)

Table 6.8: Detailed runtime profile of CODECS

The detailed runtime profile is compared to CODECSlisp in Table 6.9 where the ratio of improvement in performance is given for each function. There is a significant improvement in the runtimes of *LUDecomp*, *Solve*, *LoadJacobianRhs*, and *Recomb*. The other functions also show an improvement, although not as significant.

Description	Invchain	VCO
LUDecomp	96	89
Solve	144	139
LoadJacobianRhs	115	108
CalcCurrDeriv	54	52
Bernoulli	59	58
Mobility	38	36
Recomb	128	123
LocalError	32	34

Table 6.9: Improvement in runtime performance

The other example is the transient analysis of a series resistor diode circuit in which the diode is modeled as a two-dimensional numerical device with 21x3, 51x3, and 101x3 grid points, respectively. The runtimes for this set of examples is presented in Table 6.10. Compared to CODECSlisp the improvement in performance is by factors of 66, 72, and 99, respectively. As the problem size increases the improvement in performance is larger mainly due to the use of a good sparse-matrix package.

Description	21x3	51x3	101x3
# iterations	362	363	360
# timepoints accepted	123	126	126
# timepoints rejected	13	18	32
Analysis time (sec)	202	575	1195

Table 6.10: Runtime performance for diode example

A detailed breakup of the simulation time for these examples is provided in Table 6.11. The time for LU decomposition and matrix assembly is a major part of the total time. For larger examples the time for LU decomposition is dominant.

Description	21x3 Time (sec)	51x3 Time (sec)	101x3 Time (sec)
LUDecomp	58 (29%)	184 (32%)	393 (33%)
Solve	16 (7.9%)	52 (9%)	108 (9%)
LoadJacobianRhs	19.3 (9.6%)	53.7 (9.3%)	114 (9.5%)
CalcCurrDeriv	73.8 (36.5%)	194.6 (33.8%)	395.6 (33%)
Bernoulli	20.3	53.0	108.7
Mobility	22.8	58.9	118.3
Recomb	4.4	11.8	25.0
LocalError	1.3 (0.6%)	3,1 (0.5%)	6.2 (0.5%)

Table 6.11: Detailed runtime profile of CODECS

Finally, to compare the improvement in performance on a per component basis, data is given in Table 6.12 as a ratio of CODECSlisp to CODECS. For all cases it is seen that as the number of equations increases CODECSlisp tends to be slower in all the functions. Hence, for larger examples, in particular two-dimensional examples CODECS in C provides significant speed up.

Description	21x3	51x3	101x3
LUDecomp	110	110	168
Solve	126	134	159
LoadJacobianRhs	64	69	83
CalcCurrDeriv	44	50	60
Bernoulli	40	46	52
Mobility	40	45	54
Recomb	66	108	125
LocalError	21	25	35

Table 6.12: Improvement in runtime performance

CHAPTER 7

Comparison of Analytical and Numerical Models

7.1. Introduction

In Chapter 2 it was indicated that analytical models are inadequate under certain regimes of device operation. This problem is addressed in detail in the present chapter such that the tradeoffs between accuracy and speed performance can be made. Several one- and two-dimensional examples are used and the emphasis is on transient and small-signal ac response since the deficiencies of analytical models show up significantly in these analyses. The benchmark circuits that have been used in the experiments are also described.

The starting point for all comparisons is a set of model parameters for the analytical SPICE model that gives a good agreement in the dc characteristics of the numerical and analytical models. It is shown for both bipolar and MOS examples that even though the agreement between dc characteristics is good and the charge-storage effects are modeled as accurately as possible, the dynamic response can be significantly different. The reasons for these differences are examined and provide a guideline as to when numerical models must be used.

A comparison is made in the runtime performance of the analytical and numerical models. Analytical models are naturally faster in performance but do not always provide accurate results. The comparison only indicates the differences in runtime performance between analytical and numerical models. This then motivates the use of a mixed-level

simulation wherein only the critical devices should be evaluated numerically.

7.2. One-dimensional Diode Example

The simplest example using one-dimensional numerical models is the diode circuit as shown in Figure 7.1(a). This example is used to study the transient voltage across the diode when the source voltage is stepped from $+V_f$ to $-V_r$. An analytical model does not accurately predict the voltage as illustrated with this example.

The diode doping profile and dimensions used for the simulation are shown in Figure 7.1(b); the diode is a p^+nn^+ structure. The dc current-voltage characteristics of the diode obtained from the numerical model are shown in Figure 7.1(c). Included in this figure are also the dc characteristics for an analytical model (the SPICE diode model), that best fits the numerical results. For the analytical model the characteristics are plotted for different values of the diode series resistance, the only parameter for the analytical model with significant impact on the current-voltage characteristics in the high-current region. There is a good agreement for diode voltages up to 0.6V; at voltages larger than 0.6V the differences in the two characteristics becomes significant due to the onset of high-level injection. CODECS, as other numerical simulators, predicts the correct result since conductivity modulation [7.1] under high-level-injection conditions and other physical phenomena are modeled.

The internal carrier distribution for a diode voltage of 0.8V are shown in Figure 7.2 and indicates high-level injection. Because of conductivity modulation the resistance of the diode decreases with an increase in its voltage. The analytical model uses a fixed value for the diode series resistance and there is no bias dependence; thus, conductivity modulation cannot be modeled. As a consequence the diode resistance does not change under high-level-injection conditions and an exact match cannot be obtained with the numerical model or with experimental device characteristics at large bias voltages.

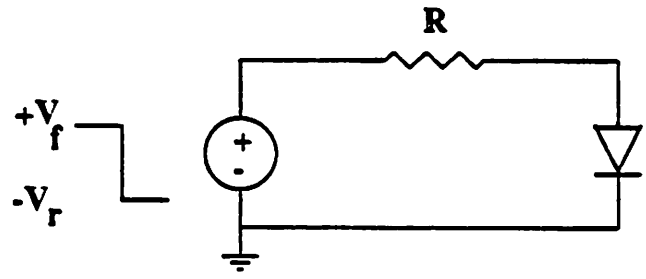


Figure 7.1(a): Resistor diode circuit.

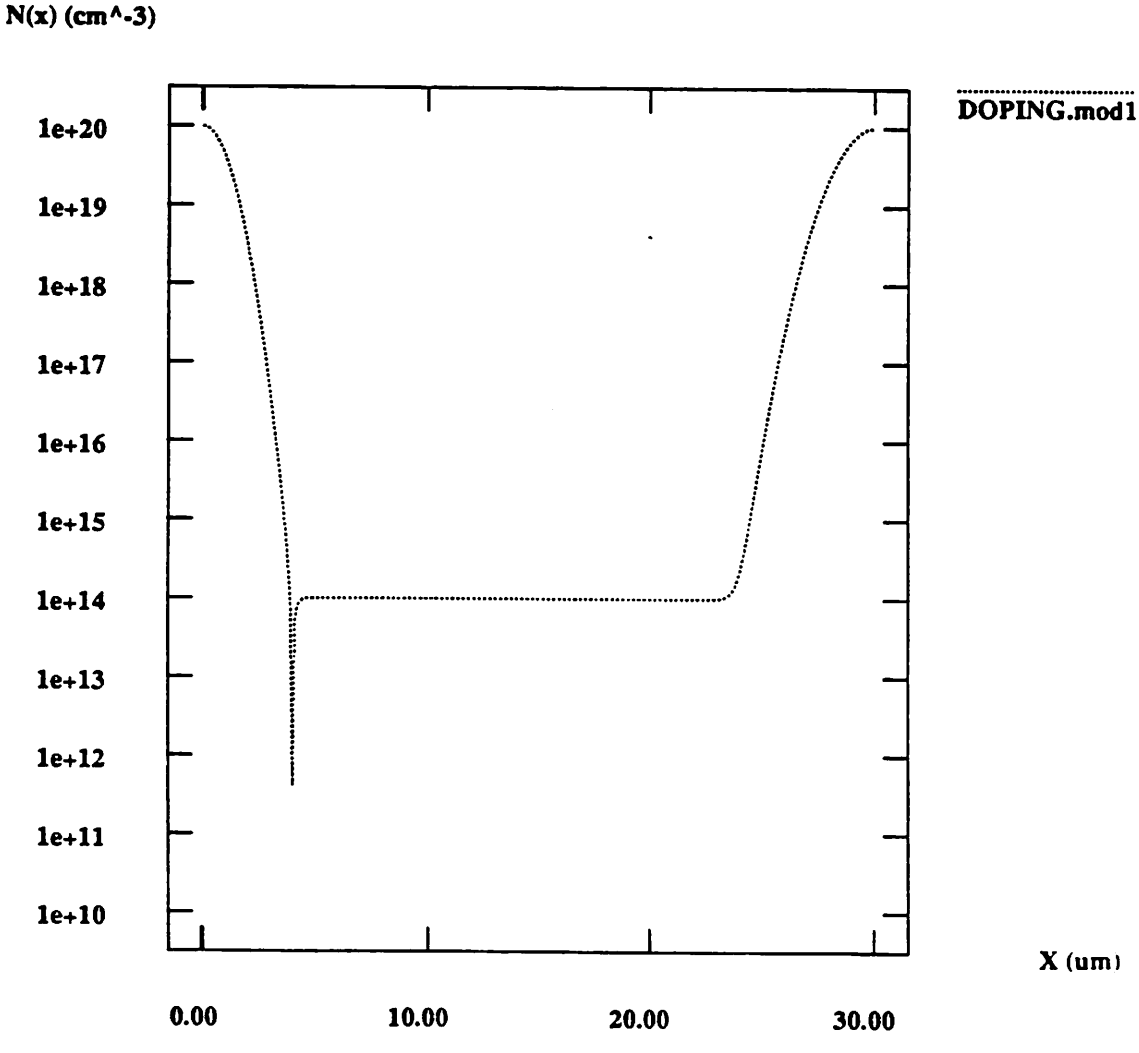


Figure 7.1(b): Doping profile and dimension of diode.

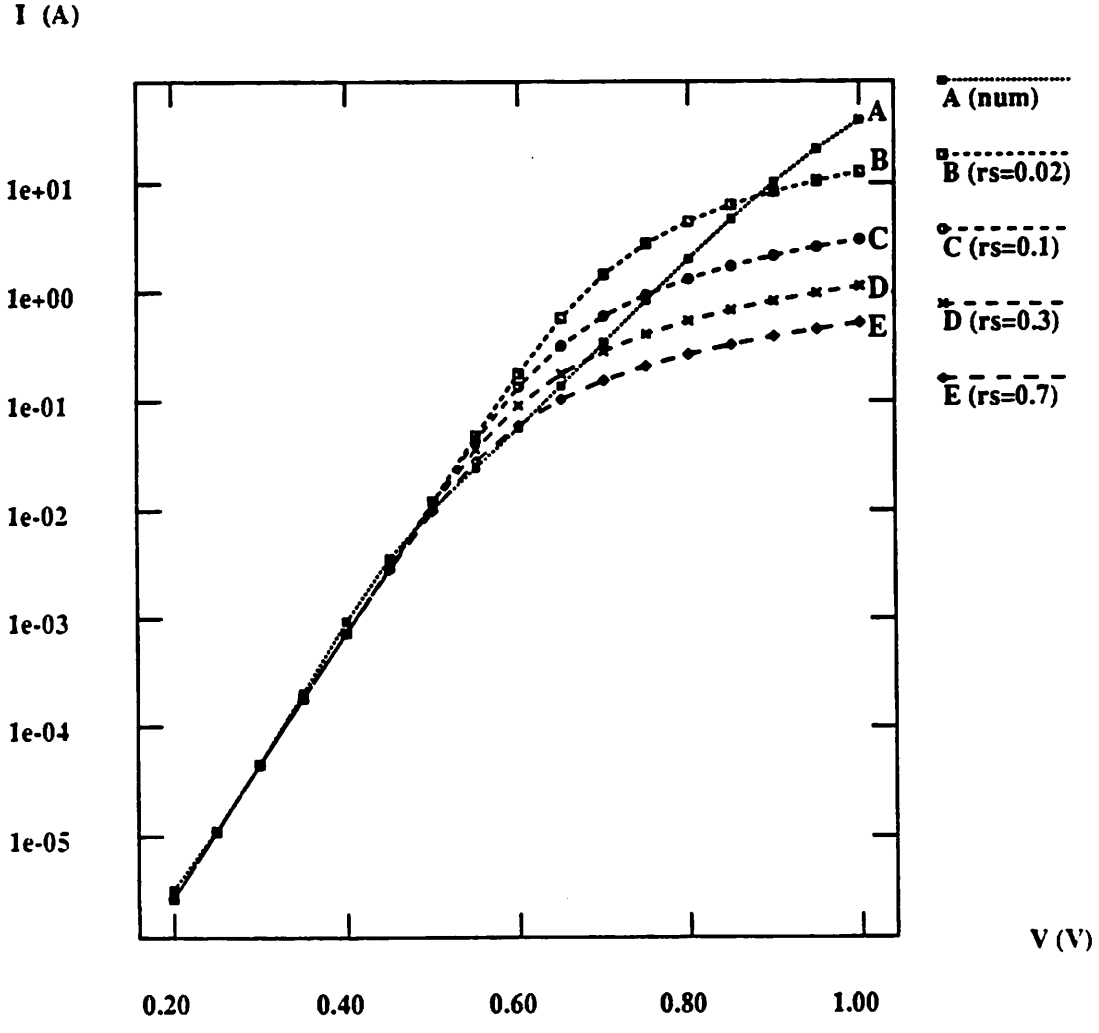


Figure 7.1(c): Dc current-voltage characteristics of diode. The results for the analytical model are for different values of the series resistance.

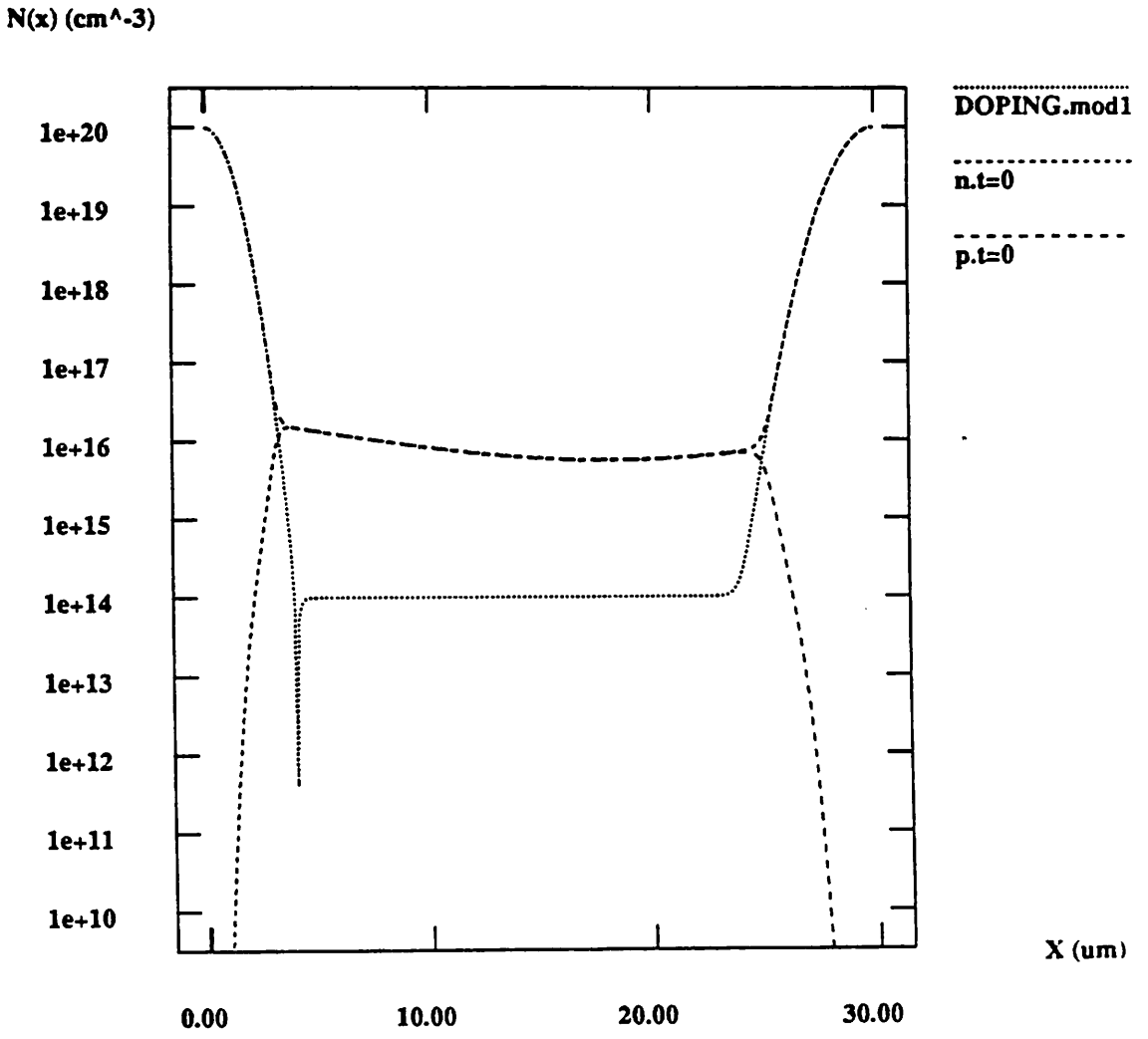


Figure 7.2: Internal electron and hole distribution for a forward voltage of 0.8V. Notice that the *n*-region is conductivity modulated.

The dynamic response of the device is determined by the regions of carrier charge storage and the depletion regions within the device. Thus, for transient simulations the depletion-region capacitance and the transit-time parameter have to be extracted for the analytical model. The depletion-region capacitance is given by the capacitance-voltage (C-V) characteristics of the diode under reverse-bias conditions. For the diode of this example the C-V characteristics are obtained by use of small-signal ac analysis on the numerical device under reverse-bias conditions. Small-signal ac analysis is performed on the diode for different values of the reverse voltage and the imaginary part of the small-signal ac current is used to calculate the capacitance value. The capacitance is then plotted as a function of the reverse voltage as shown in Figure 7.3. The C-V characteristics of the analytical diode model are also shown and a good fit is possible with the numerical data. It is, however, difficult to extract the transit-time parameter, TT; therefore, simulations with the analytical model have been performed for different values of TT.

The simulated diode voltage as a function of time is shown in Figure 7.4(a). The results from the analytical model are plotted for different values of the parameter TT. A variation in TT only changes the time of zero crossing of the diode voltage and does not alter the dynamic behavior otherwise. The diode voltage switches from a positive value to a negative value over a very small time interval and the diode voltage obtained from the analytical model does not emulate the behavior of the diode voltage obtained from numerical simulations. The simulations from CODECS are physically correct [7.1] and can be explained by examining the charge storage within the device. At the time at which the diode voltage becomes zero there is a large number of carriers stored in the n-region of the diode. The distribution of the electrons and holes within the diode are shown for two instances of time in Figure 7.4(b). The two instances correspond to the time at which the diode voltage is zero and at a time t = 70nsec, when the diode voltage

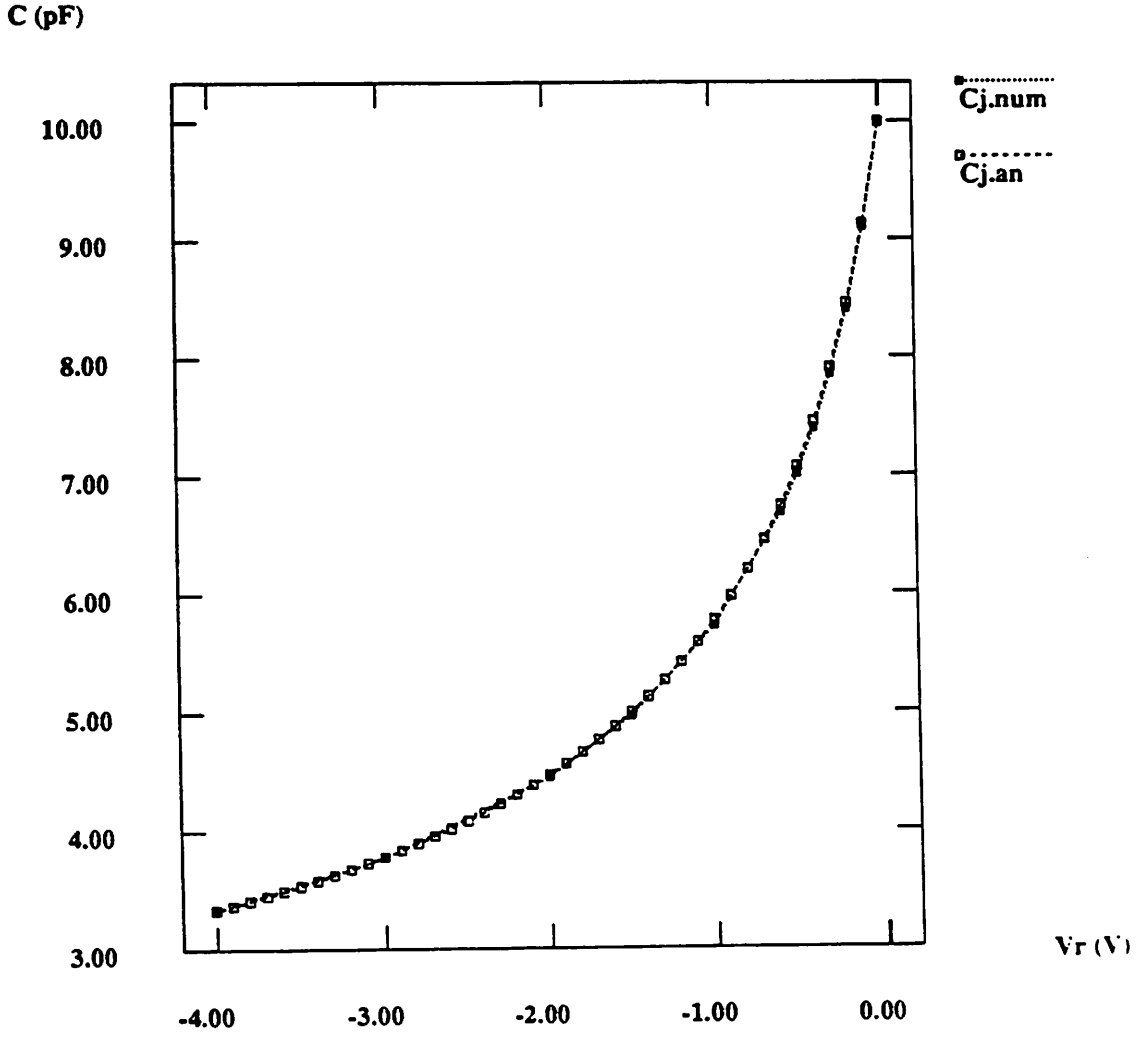


Figure 7.3: Capacitance-voltage characteristics of diode for reverse-bias conditions. Analytical and numerical models show good agreement.

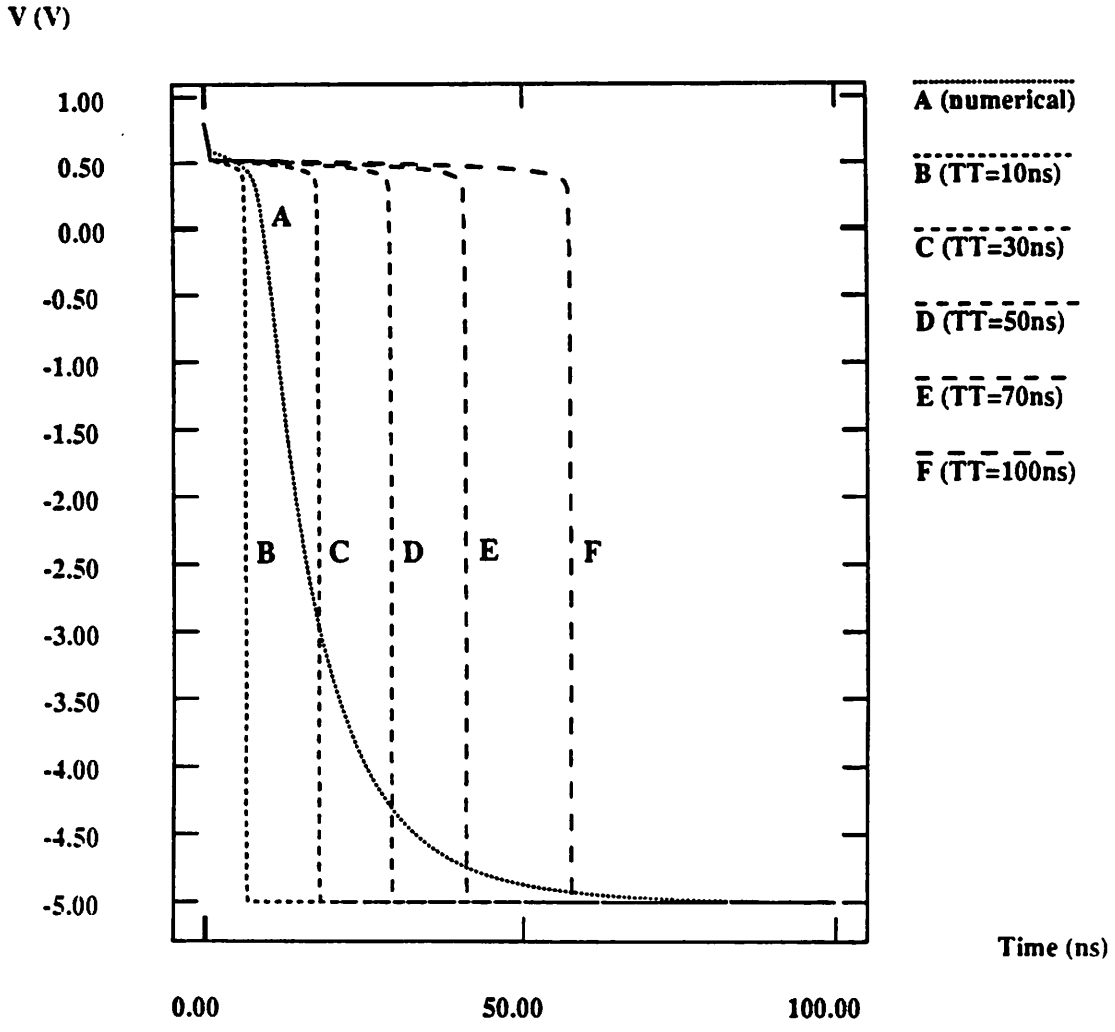


Figure 7.4(a): Diode voltage as a function of time for the turn-off transient. The simulations with the analytical model are for different values of the model parameter *TT*.

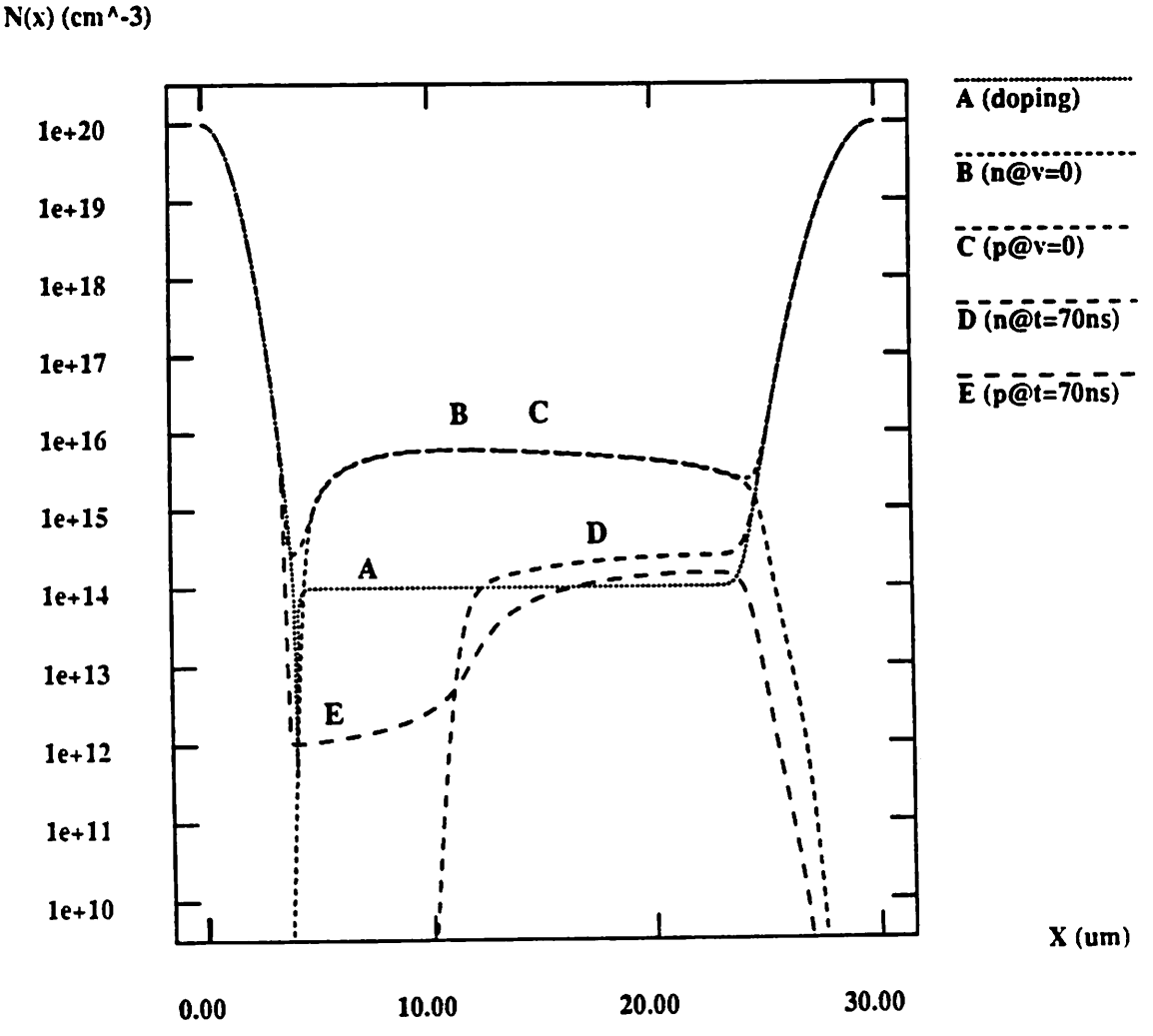


Figure 7.4(b): Electron and hole concentrations as a function of position for two time instants.

is near its steady-state value of -5V. It is clear from Figure 7.4(b) that a large amount of stored charge has to be removed from the n-region in addition to the charging up of the depletion-region capacitance. This additional charge is responsible for the slow decay of the diode voltage in Figure 7.4(a). The analytical model does not predict the correct nature of the transient voltage across the diode.

7.3. One-Dimensional Bipolar Transistor Examples

For the simulation examples of this section numerical one-dimensional models are used for the bipolar transistors. The bipolar transistor is assumed to have a uniform doping profile as shown in Figure 7.5(a). Uniform doping is assumed only for convenience, since depletion-region capacitances can be easily calculated. In particular the expression for abrupt-junction capacitance [7.2] can be used, and the forward transit time can also be calculated. Although the doping profiles are not realistic they suffice to illustrate the deficiencies of the analytical bipolar transistor model of SPICE.

The dc current-voltage characteristics obtained from numerical simulation are shown in Figure 7.5(b). Also shown are the characteristics of an analytical Gummel-Poon model [7.3] that best fits the results obtained from physical simulations. A good agreement exists between the dc characteristics of the analytical and numerical models.

To verify that the values of junction capacitances and the transit-time parameters for the analytical model are indeed accurate, transient simulations have been performed on the *Oscillator* and *VCO* examples introduced in Chapter 4. The transient responses of the two circuits simulated by use of analytical and numerical models are shown in Figures 7.6 and 7.7. It is seen that the analytical model predicts the frequencies of oscillation and the voltage magnitudes in agreement with the numerical model. For these examples the transistors are never driven into quasi-saturation or saturation [7.1, 7.2] and the analytical model can be used without loss of accuracy. However, whenever the transistor

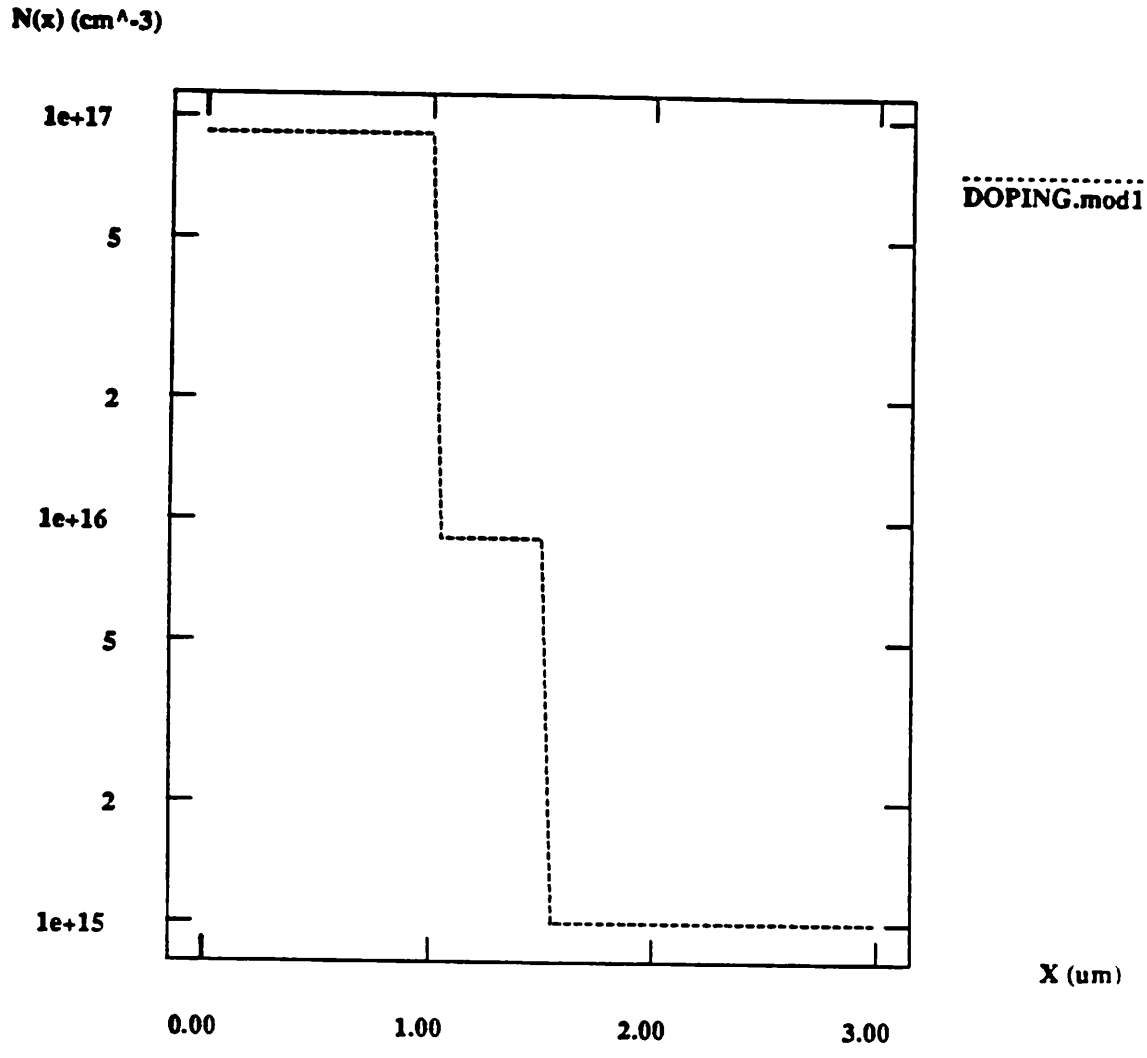


Figure 7.5(a): Doping profile and dimension of *npn* bipolar transistor.

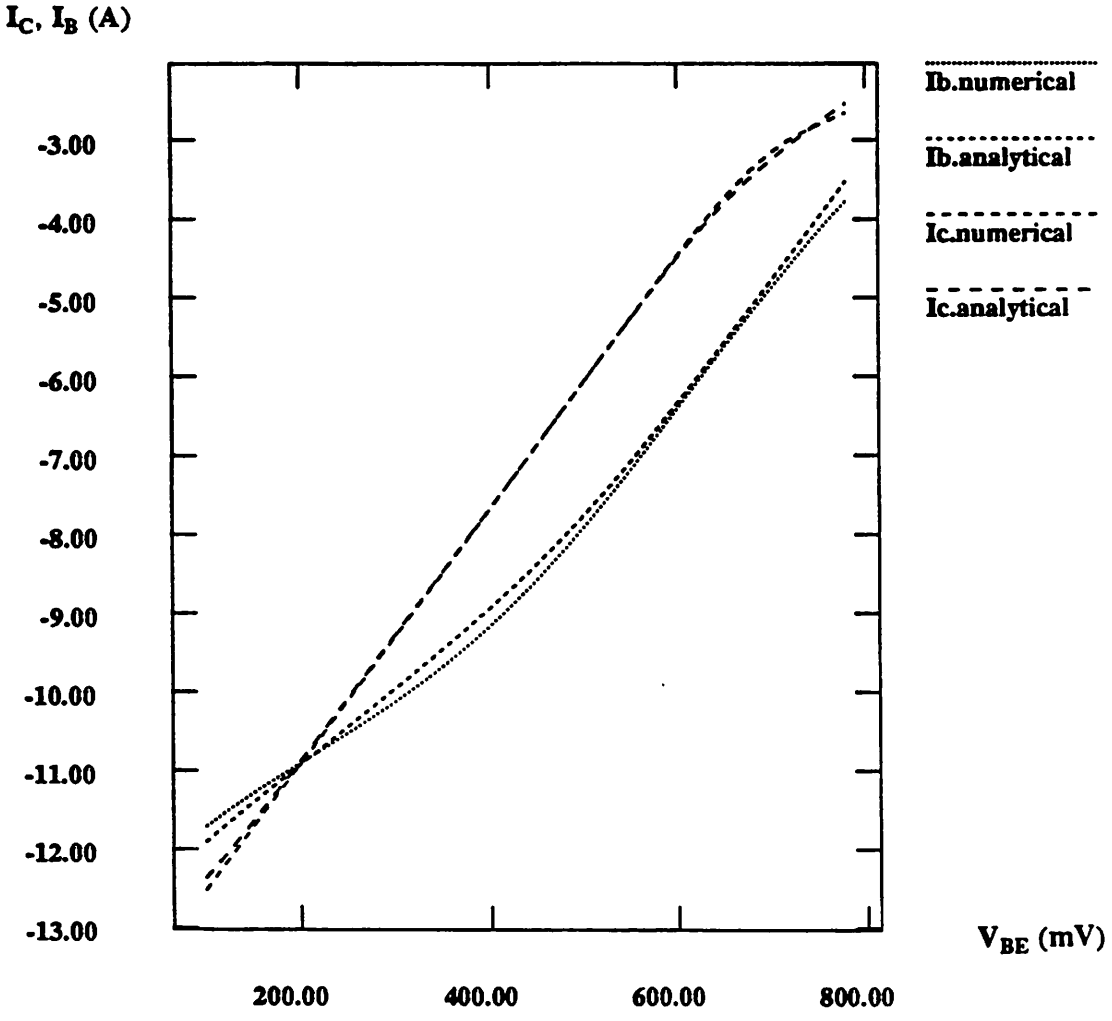


Figure 7.5(b): Dc current-voltage characteristics of the bipolar transistor. The numerical and analytical models show good agreement.

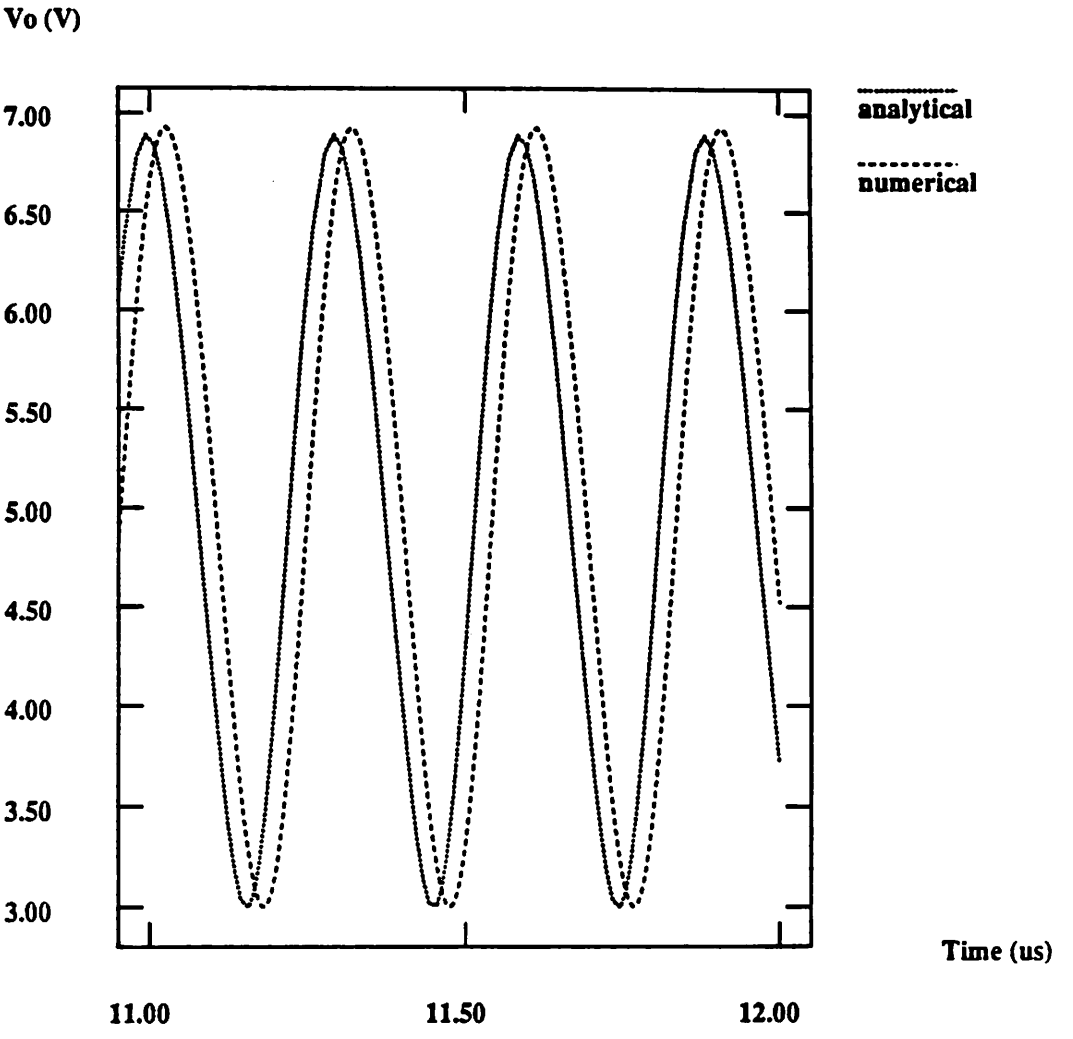


Figure 7.6: Simulated transient response for the oscillator circuit. Numerical and analytical models predict an agreement in the voltage magnitude and frequency of oscillation.

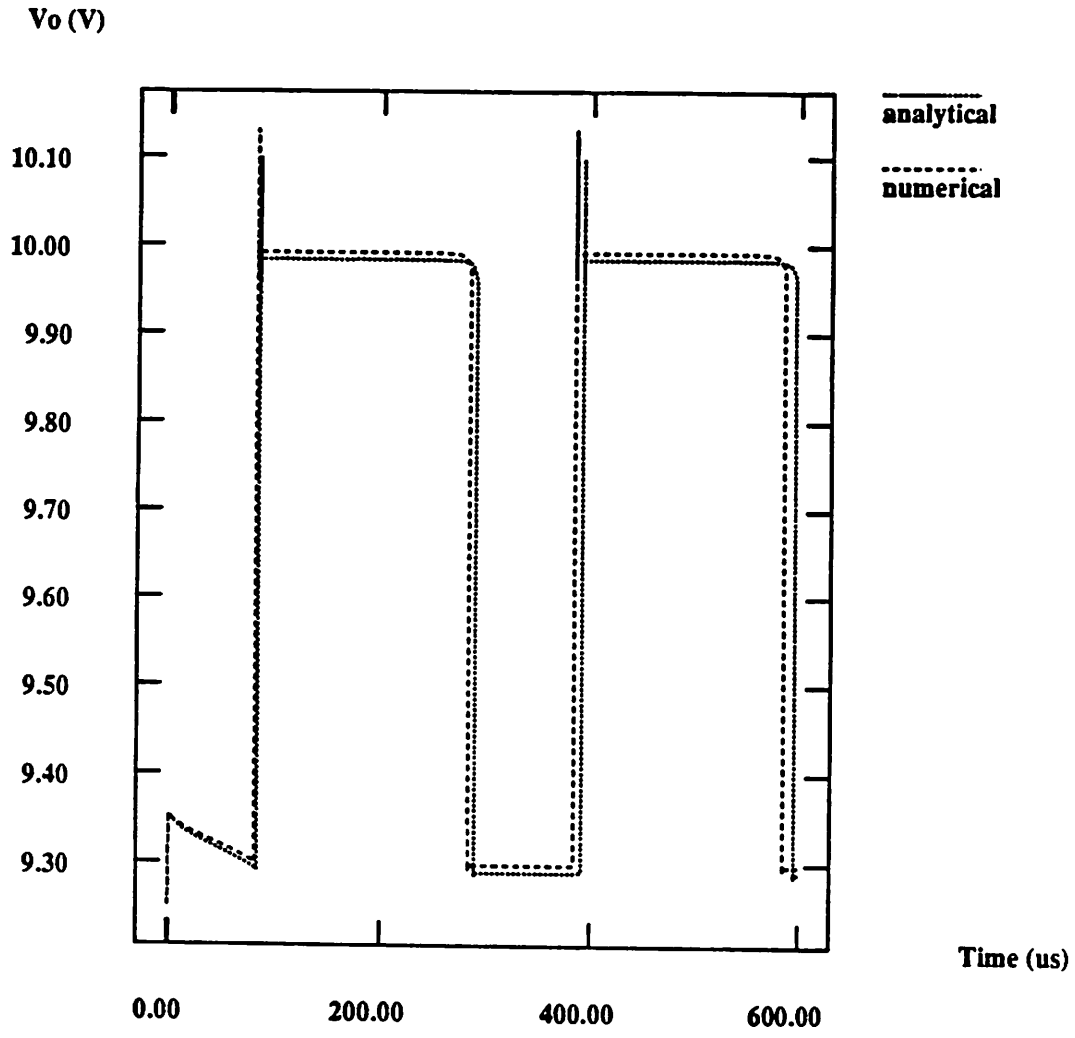


Figure 7.7: Simulated transient response for the VCO circuit. Results from the analytical and numerical model show good agreement.

operates under high-level-injection conditions the analytical model fails to produce the correct results.

Consider the simple example of the RTL-inverter circuit shown in Figure 7.8; the transient response of the output voltage to a step input voltage is shown in Figure 7.9. For the analytical model the parameter *TR*, the reverse transit time, is required when the transistor is operating in bipolar saturation. Since this parameter depends on bias conditions it is again difficult to use one single value of *TR* for the complete analysis. A comparison of the responses for different values of *TR* is provided in Figure 7.9, where it is seen that no value of *TR* provides a good agreement with the physical simulations. The lack of a proper value of *TR* can produce significantly different characteristics in other circuits as well.

Consider the four-transistor inverter chain of Figure 7.10. The simulations with the analytical model are performed with *TR* = 3.5ns, the value that appears to be most reasonable from Figure 7.9. The voltages at nodes 3 and 9 are plotted in Figure 7.11, and the differences in the response are obvious.

The f_T* of the bipolar transistor is a function of collector current and is plotted in Figure 7.12 for both the numerical and analytical models. From the numerical model f_T reaches a peak value of 5.0 GHz at a collector current of approximately 1mA; the analytical model predicts a peak f_T of 3.8 GHz. At large collector currents the f_T degrades significantly as seen with the numerical simulations. However, the analytical model predicts substantially higher values of f_T at large collector currents. Since the numerical models are based on the physics it is expected that the results from the numerical simulations are correct, and that the analytical model does not provide a good estimate of f_T for large collector currents.

* f_T is the frequency at which the short-circuit, common-emitter current gain falls to unity.

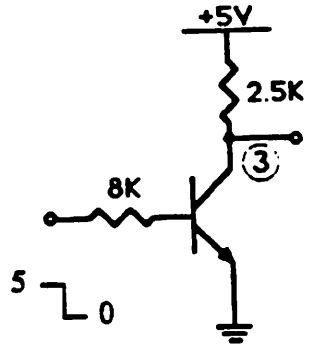


Figure 7.8: An RTL-inverter circuit.

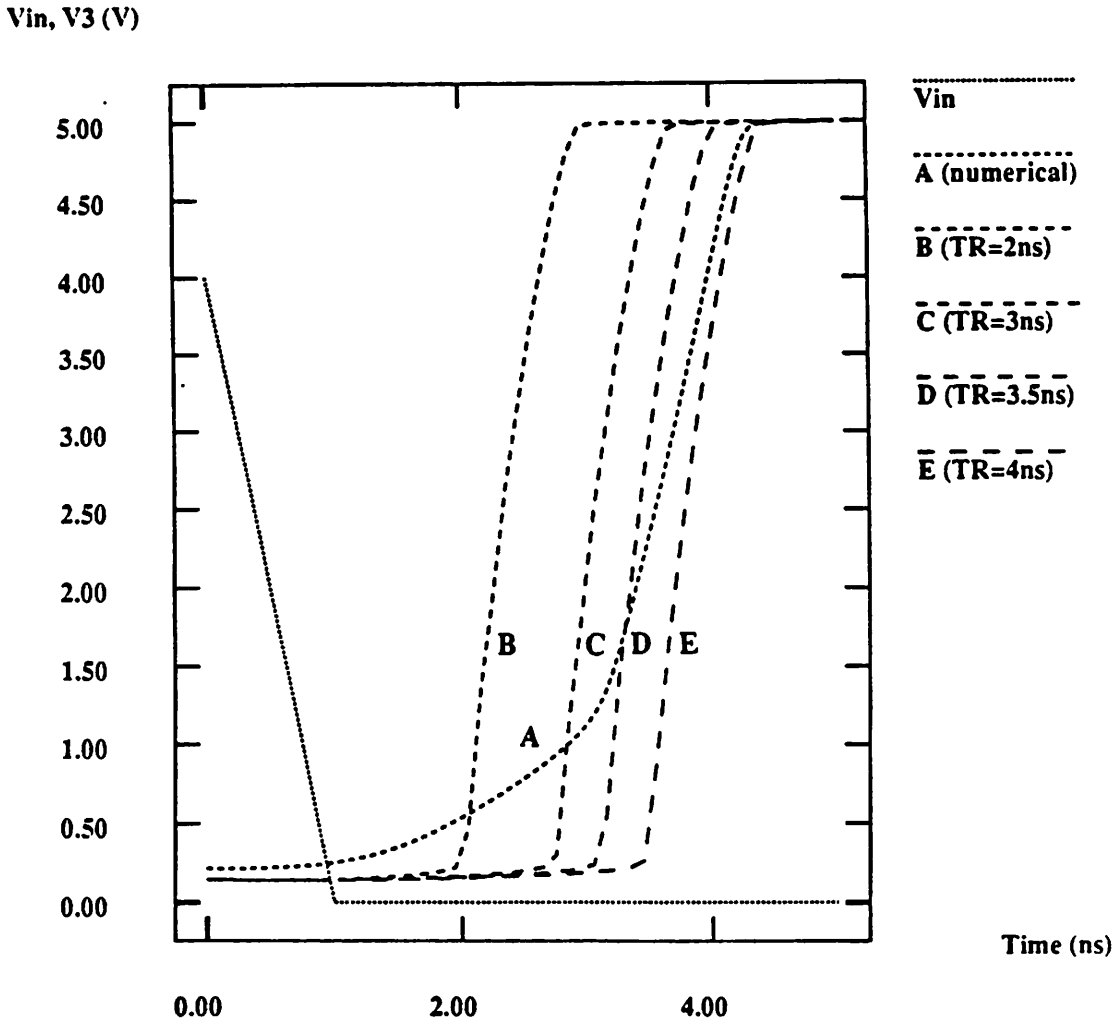


Figure 7.9: Simulated transient output voltage for the RTL inverter. Simulations with the analytical model are for different values of the model parameter *TR*.

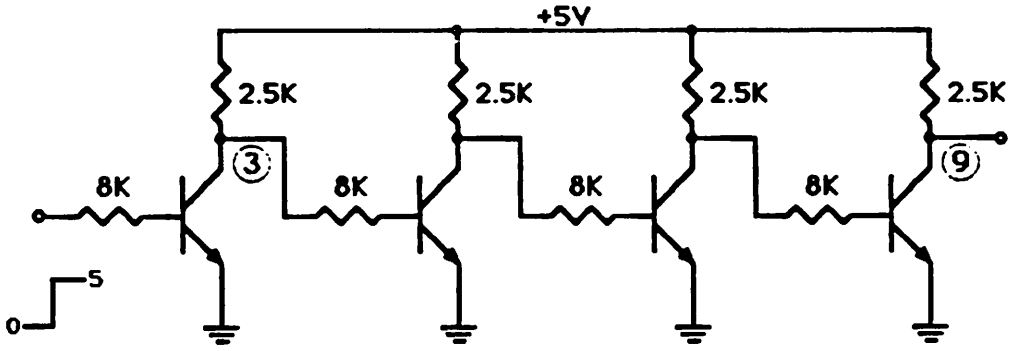


Figure 7.10: RTL-inverter chain with four inverters.

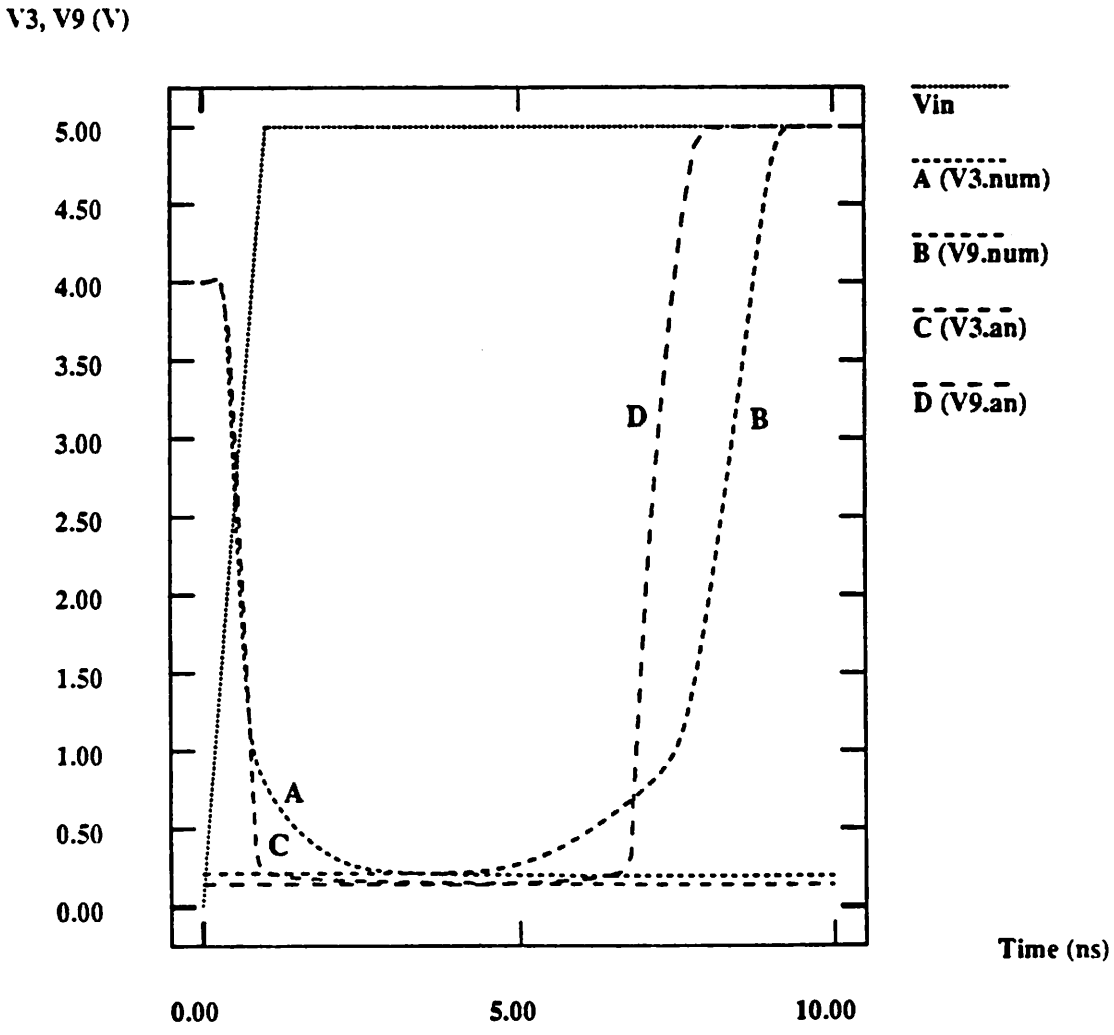


Figure 7.11: Simulated transient voltage at nodes 3 and 9 comparing numerical and analytical models.

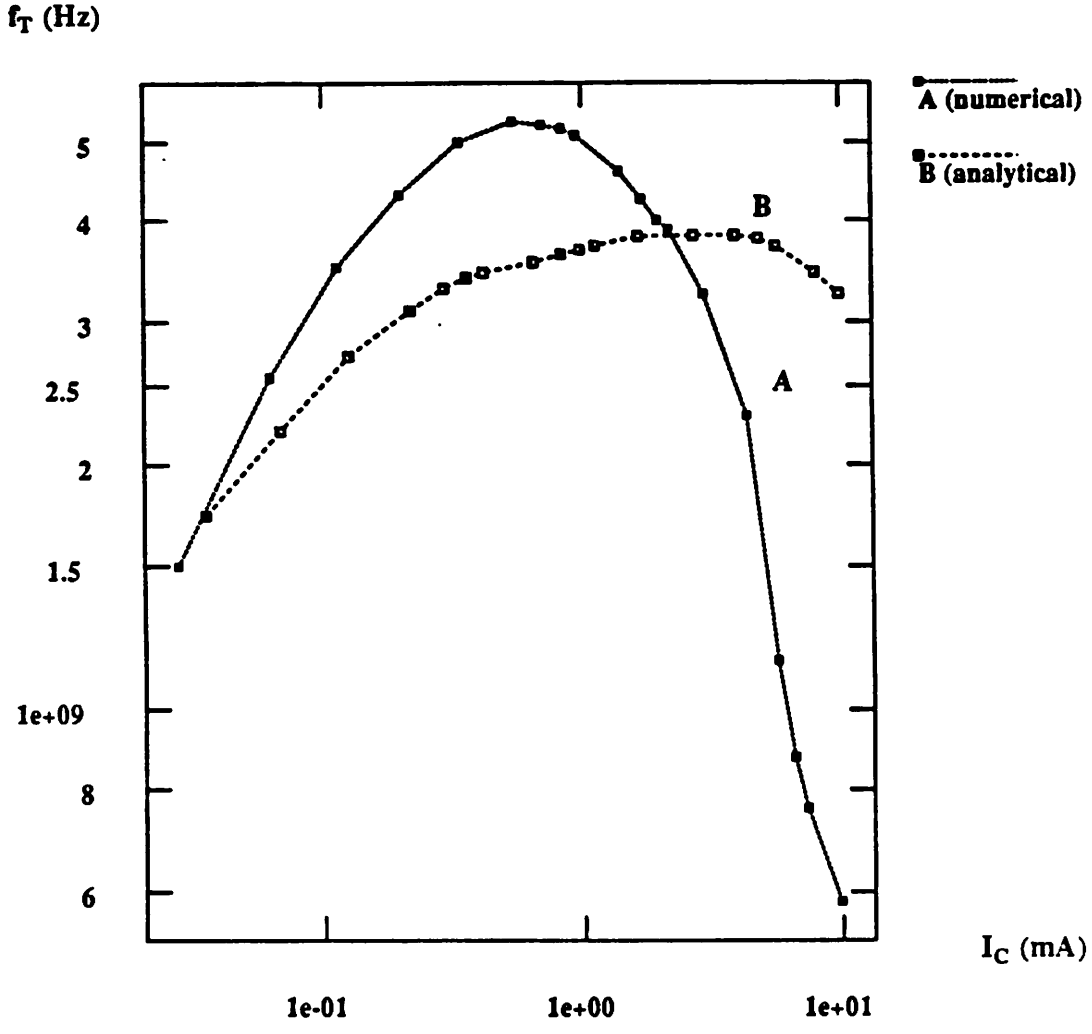


Figure 7.12: Simulated f_T as a function of collector current using analytical and numerical models.

Another example in which the analytical model will not give accurate results is the operation of the transistor under quasi-saturation conditions. The GP-model does not accurately model the base pushout; and, hence, the charge stored in the collector region. Therefore, it does not provide physically accurate results whenever base pushout occurs. High-level-injection conditions are critical to the performance of bipolar transistors that are used to drive large capacitive loads such as the output transistors in BiCMOS or ECL driver circuits. Shown in Figure 7.13(a) is a BiCMOS driver circuit during the pull-up transient; the bipolar transistor doping profile is again assumed to be uniform as shown in Figure 7.13(b). The bipolar transistor operates under high-level injection in the collector for large capacitive loads. Base pushout occurs and as a consequence the current gain decreases and the base transit time increases. The following example once again illustrates that analytical models that provide a good match in the dc current-voltage characteristics with numerical models may give considerably different results under transient simulations. In addition, this example illustrates how numerical models can be used to evaluate the impact of technological changes on circuit performance, a task that is very difficult if not impossible to do with analytical models.

The dc current-voltage characteristics for the bipolar transistor, obtained from numerical and analytical models are shown in Figure 7.14(a). In Figure 7.14(b) the dc current-voltage characteristics of two different transistors are compared; the first transistor has the doping profile of Figure 7.13(b) and the doping profile of the second transistor is given in Figure 7.15. As seen from Figure 7.14(b), the addition of the buried layer to the original transistor does not alter the dc characteristics and hence the same analytical model can be used. The transient response for the two different profiles would be the same when computed using the analytical model, whereas the numerical simulations indicate significantly different operation.

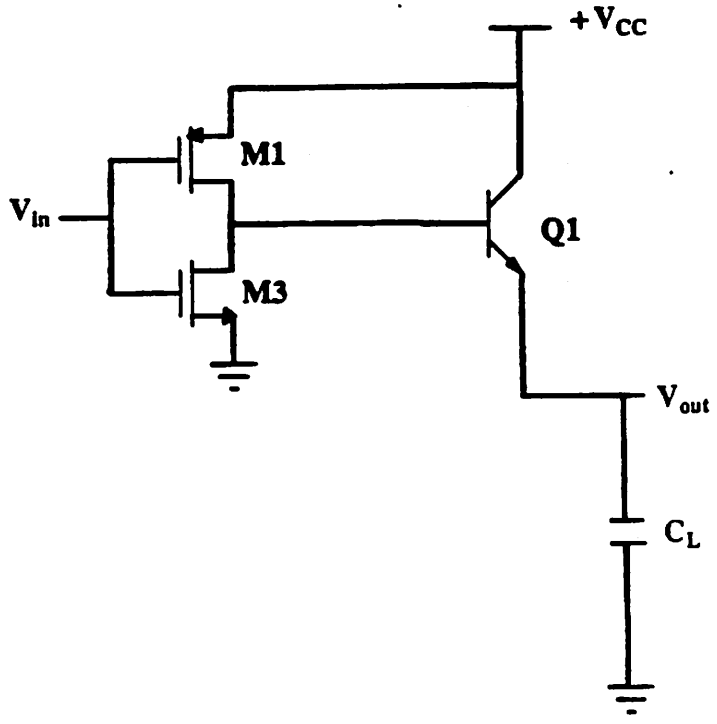


Figure 7.13(a): BiCMOS driver circuit for pull-up transient.

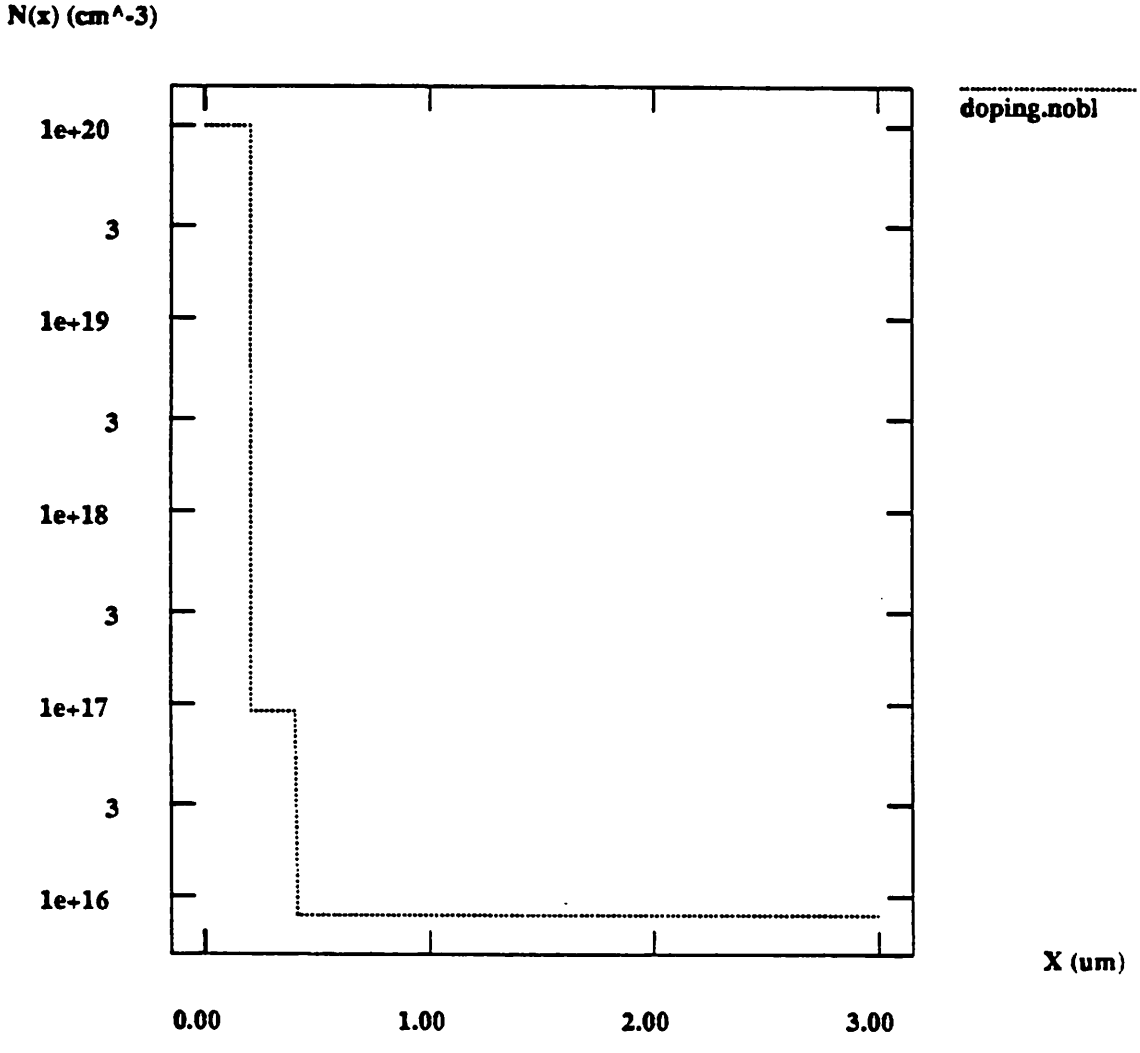


Figure 7.13(b): Bipolar transistor doping profile and dimension.

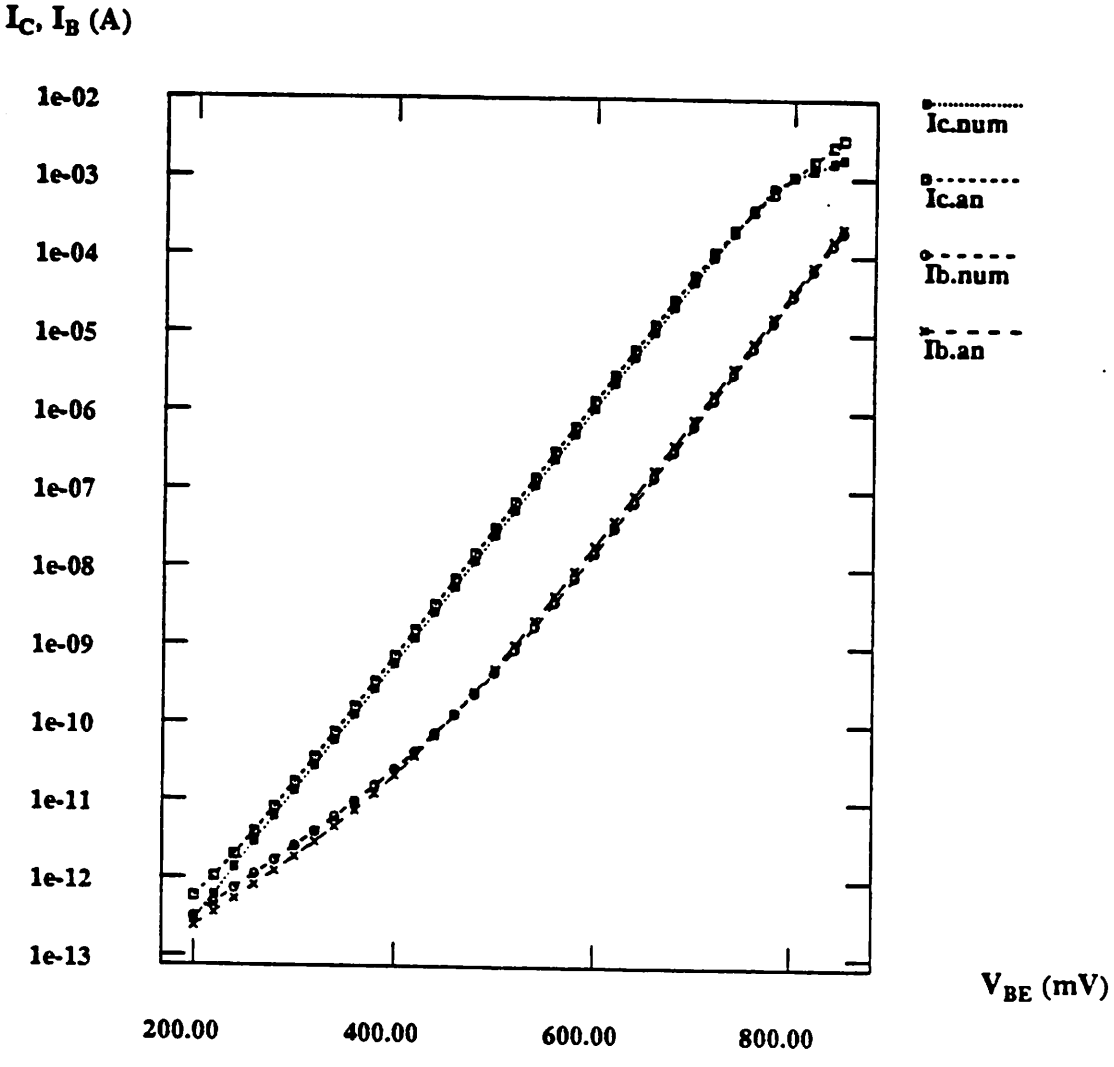


Figure 7.14(a): Simulated dc current-voltage characteristics from numerical and analytical models.

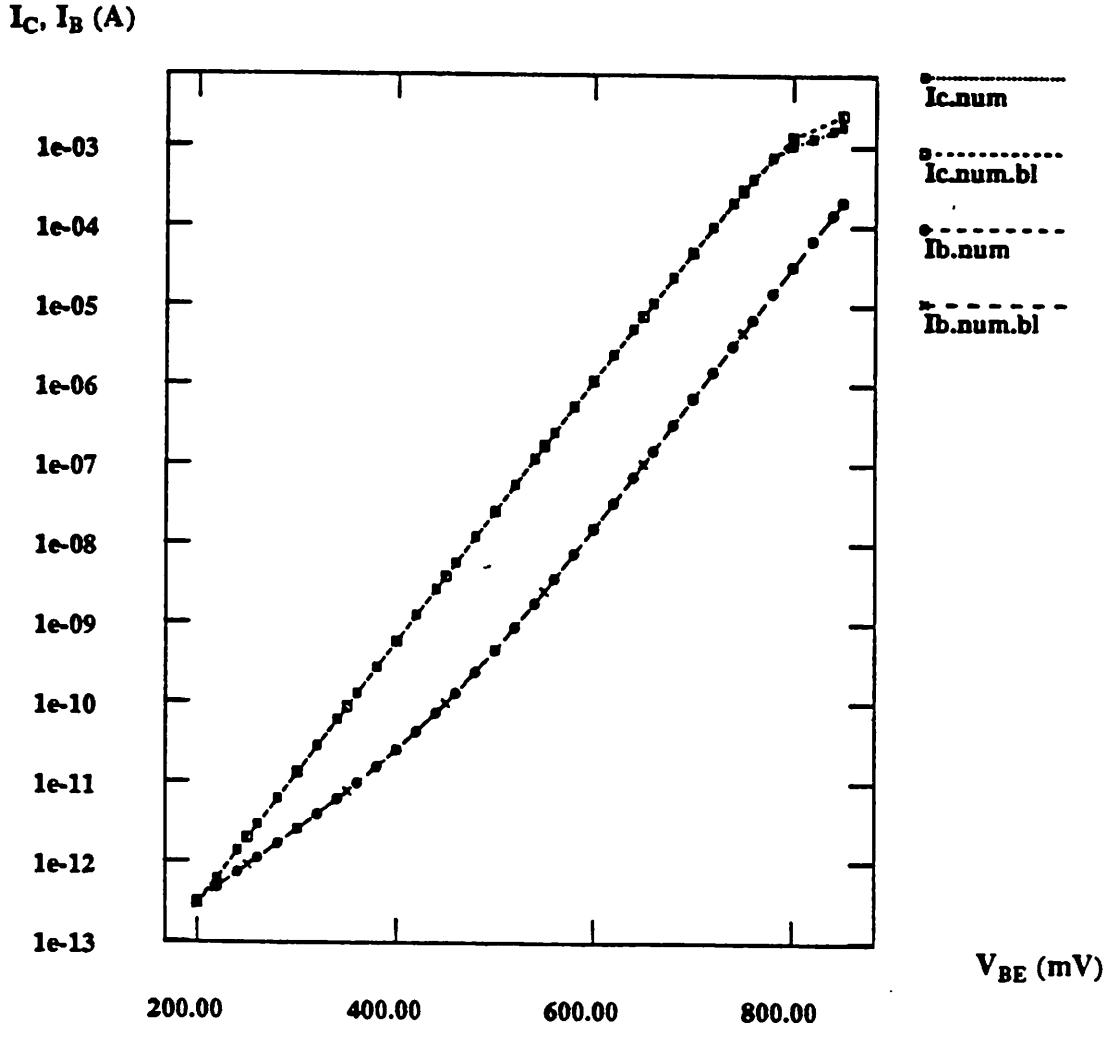


Figure 7.14(b): Simulated dc current-voltage characteristics for transistors with and without a buried layer.

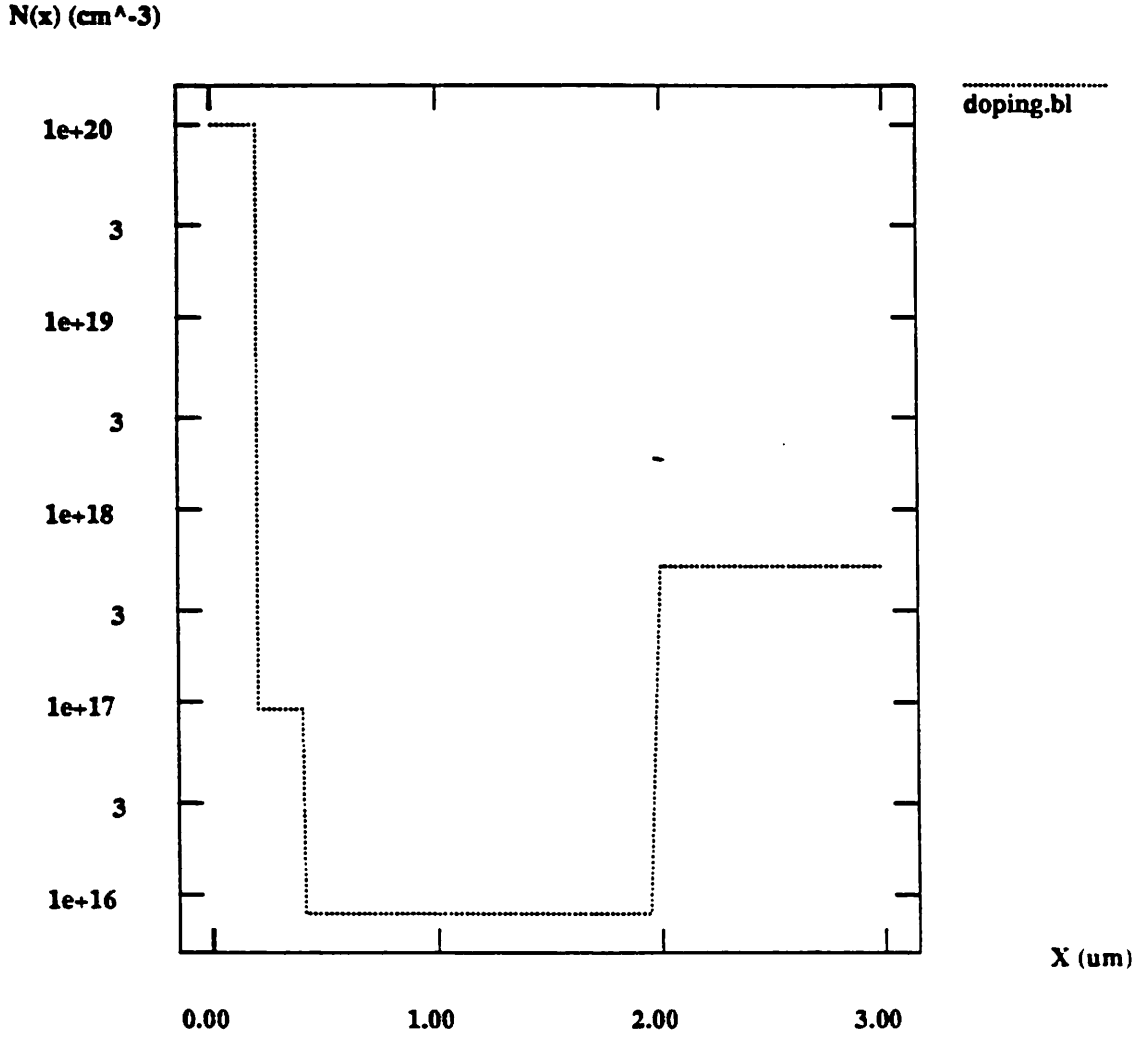


Figure 7.15: Doping profile and dimension of the bipolar transistor with a buried layer.

In Figures 7.16(a) and 7.16(b) are shown the simulated collector current waveforms as a function of time for the pull-up transient. The simulations have been performed for two different values of load capacitance, 0.1pF and 10pF, respectively. The SPICE Level-2 MOS model is used for the MOSFET, and a numerical one-dimensional model or the analytical Gummel-Poon model is used for the bipolar transistor. For small values of load capacitance the bipolar transistor operates under low-level-injection conditions and the simulated collector currents with the three different models are in agreement as seen in Figure 7.16(a). However, for large values of load capacitance base-pushout effects become important. The collector currents for the three different models are substantially different in Figure 7.16(b). The numerical models show a distinct roll-off of collector current at the onset of high-level injection in the collector. The transistor with the buried layer goes into high-level injection at a larger collector current as seen in Figure 7.16(b). The simulated results from the analytical model do not depict the same behavior in collector current, and it is clear that the analytical model cannot be used for simulation of base-pushout effects. The internal carrier distributions for the two bipolar transistor profiles are shown in Figure 7.17. The transistor with a buried layer has less base pushout compared with the transistor without the buried layer. Therefore, a larger current is available to charge up the load capacitance leading to a smaller delay. The analytical model cannot be used to evaluate the effect of the buried layer, since the dc characteristics do not change and the same analytical model is applicable to both types of transistor structures.

7.4. Two-Dimensional MOS Examples

CODECS simulation of MOS circuits is presented in this section. Consider the MOS transistor geometry shown in Figure 7.18(a) and the two-dimensional doping profile shown in Figure 7.18(b). Again for simplicity uniform doping profiles are

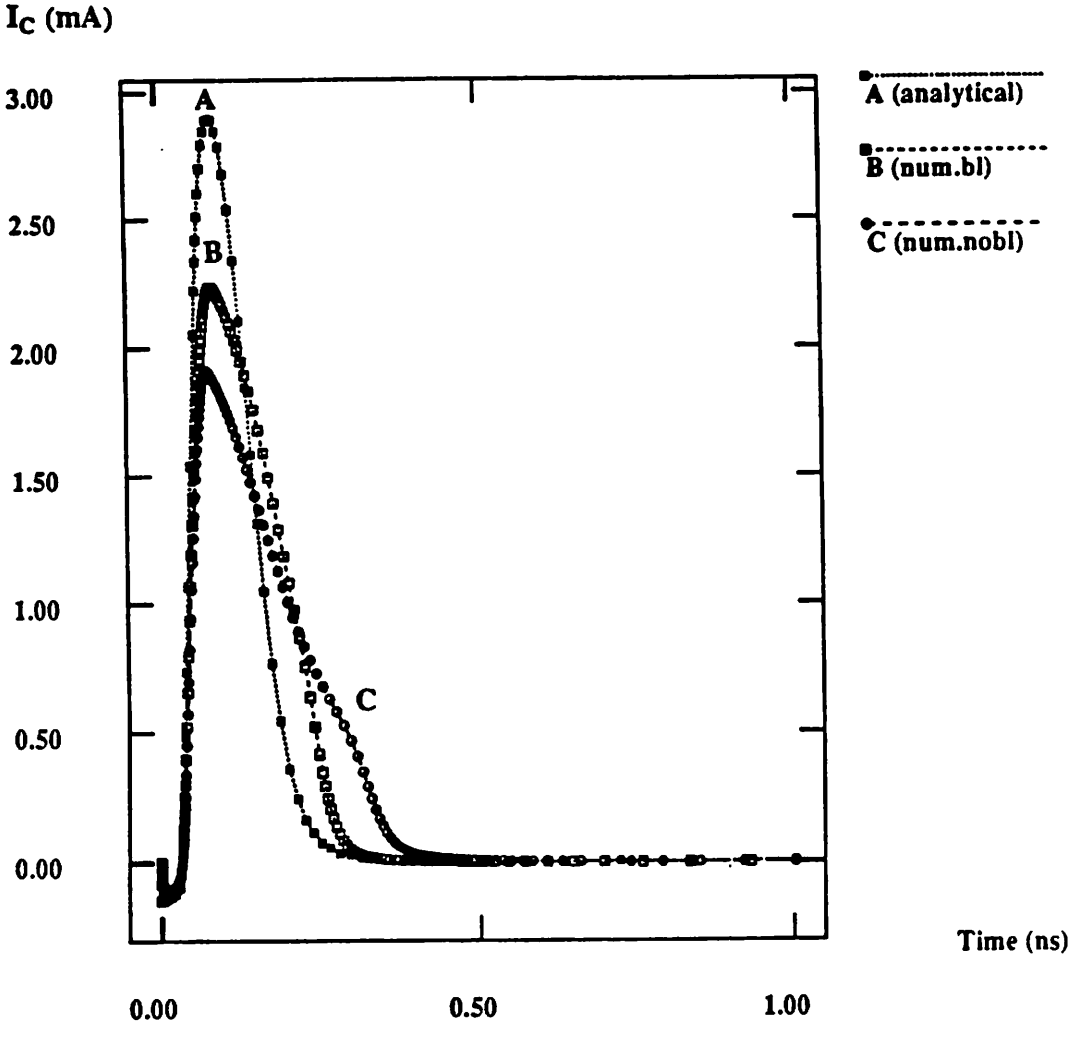


Figure 7.16(a): Simulated collector currents for the pull-up transient for C_L = 0.1pF. The results are shown for the analytical model and for the numerical models with and without the buried layer.

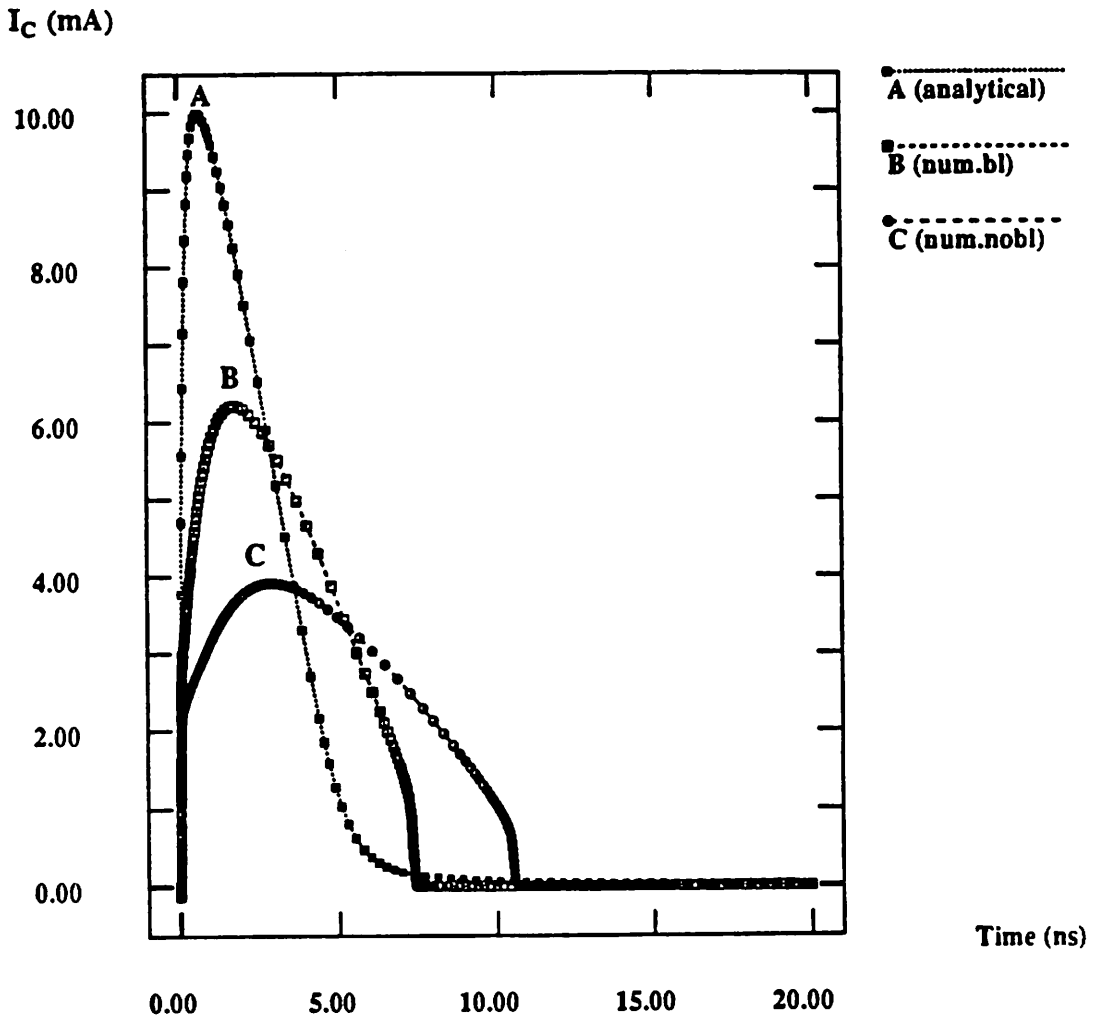


Figure 7.16(b): Simulated collector currents for the pull-up transient for C_L = 10pF. The results are shown for the analytical model and for the numerical models with and without the buried layer. The base-pushout effect is seen with the numerical models.

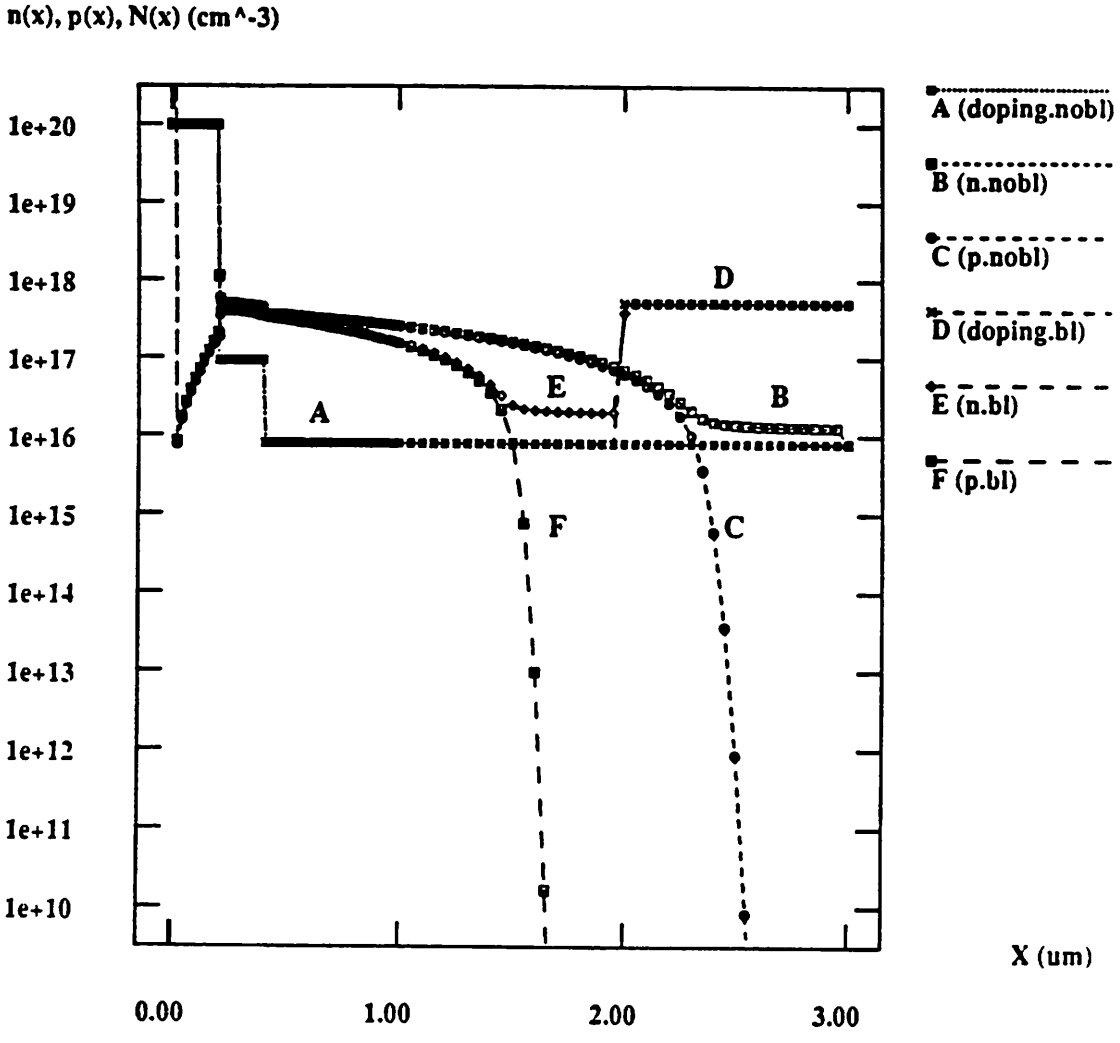


Figure 7.17: Internal carrier distributions showing the base-pushout effect in the transistors with and without the buried layer.

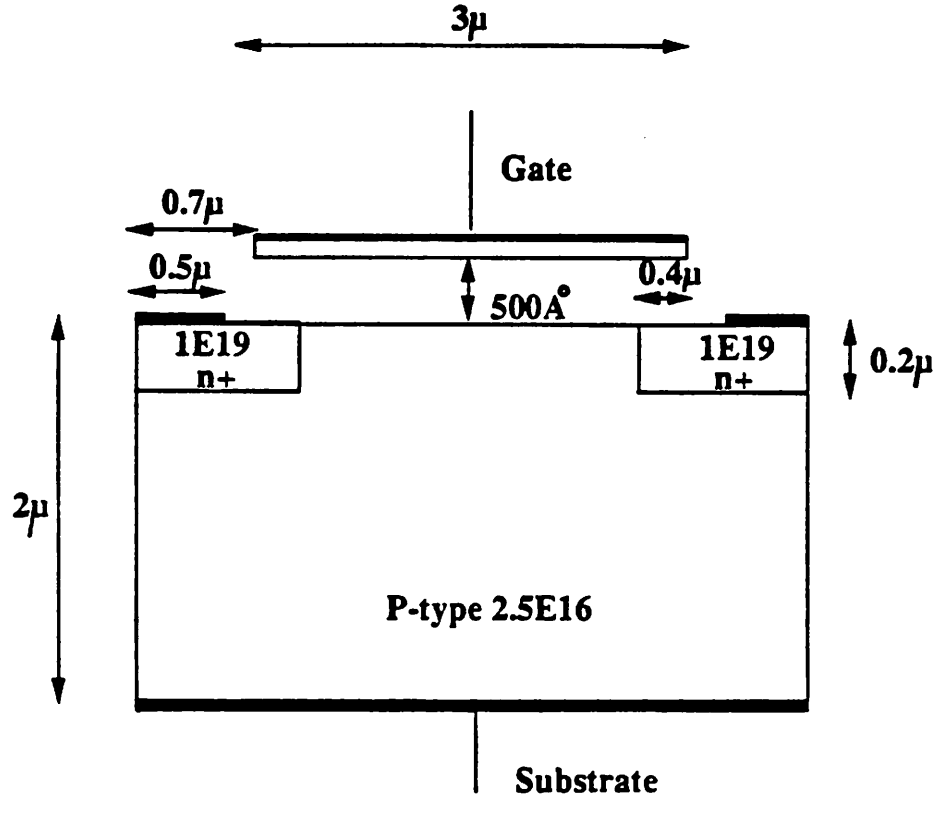


Figure 7.18(a): Geometry of the MOS transistor used for simulations.

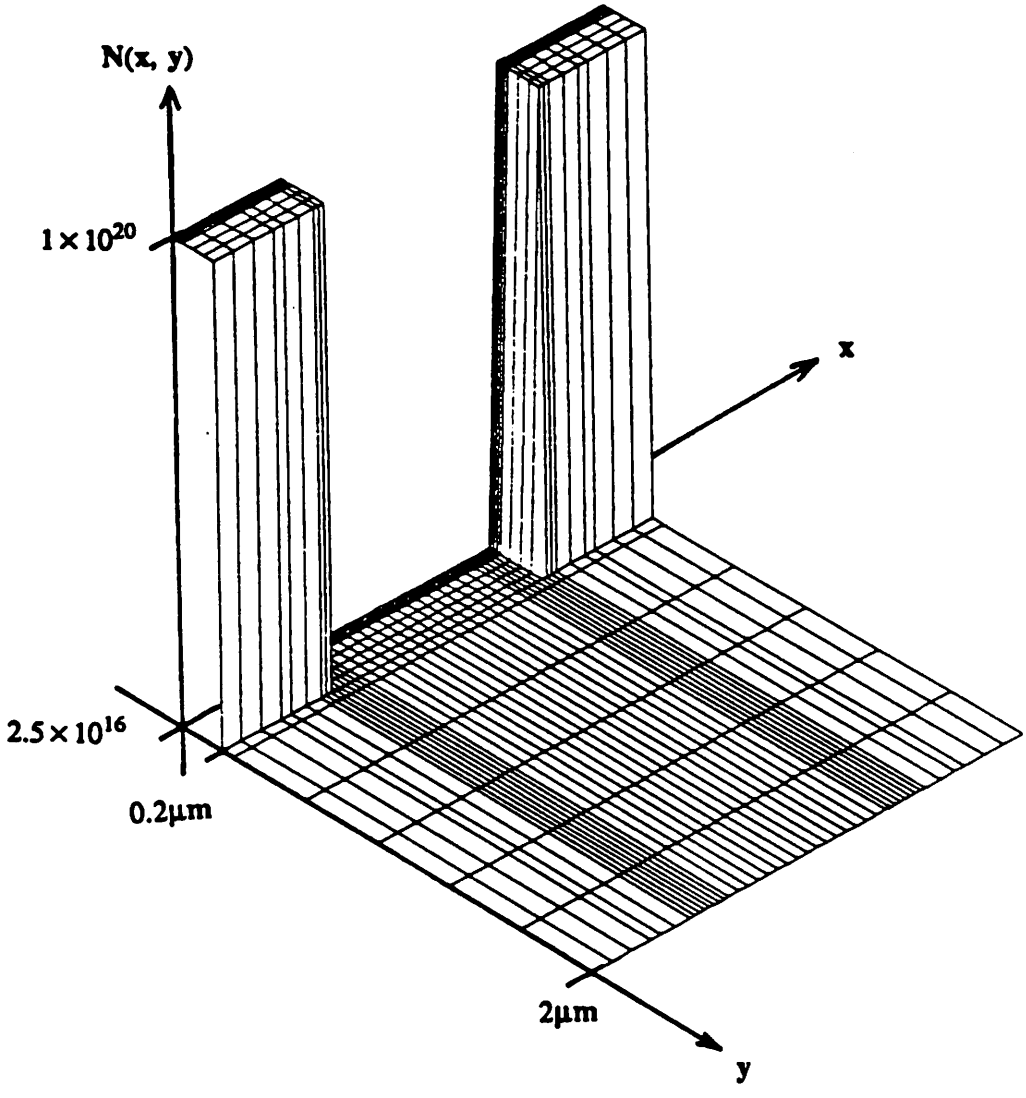


Figure 7.18(b): Doping profile of MOSFET.

assumed as shown. The dc characteristics of the MOS device are shown in Figure 7.19 where the I_D–V_{GS} characteristics with the substrate bias as a parameter are given in Figure 7.19(a) and the I_D–V_{DS} characteristics for different gate voltages are given in Figure 7.19(b). Also shown in these figures are the characteristics of the SPICE Level-2 MOS model that best fits the numerical results. Reasonable agreement is seen between the characteristics generated from the two models.

Charge nonconservation has been of serious concern in MOSFET circuits. Some analytical MOS models do not conserve charge and are not suited to simulation of switched-capacitor circuits, where charge transfers are important. Any MOSFET model must possess the property of charge conservation. In analytical models charge conservation can be achieved by use of charge as the state variable [7.4, 7.5]. Physical models should conserve charge since charge conservation is implied in the basic equations, and this is indeed the case. Two examples are used to illustrate charge conservation in the numerical model. They are the bootstrap circuit of Figure 7.20(a) [7.6] and the charge pump circuit of Figure 7.20(b) [7.4]. The first circuit is a simple circuit to study charge conservation, whereas the second circuit provides a better test for charge conservation since the MOS transistor goes through all its regions of operation. The capacitor voltages as a function of time are shown in Figures 7.21(a) and 7.21(b). The capacitor voltage from the numerical model is maintained at its constant value from cycle to cycle, thereby depicting that charge is conserved. The SPICE Level-2 MOS model, with the Meyer capacitance model [7.7], is known to have charge conservation problems [7.4]. This is seen to be the case in Figures 7.21(a) and 7.21(b), where the results from the Level-2 model are also included; a shift in the capacitor voltages can be seen due to nonconservation of charge.

Next consider the turn-on and turn-off transients of a MOS transistor; the simulations are performed by use of the simple circuit of Figure 7.22. The gate voltage is

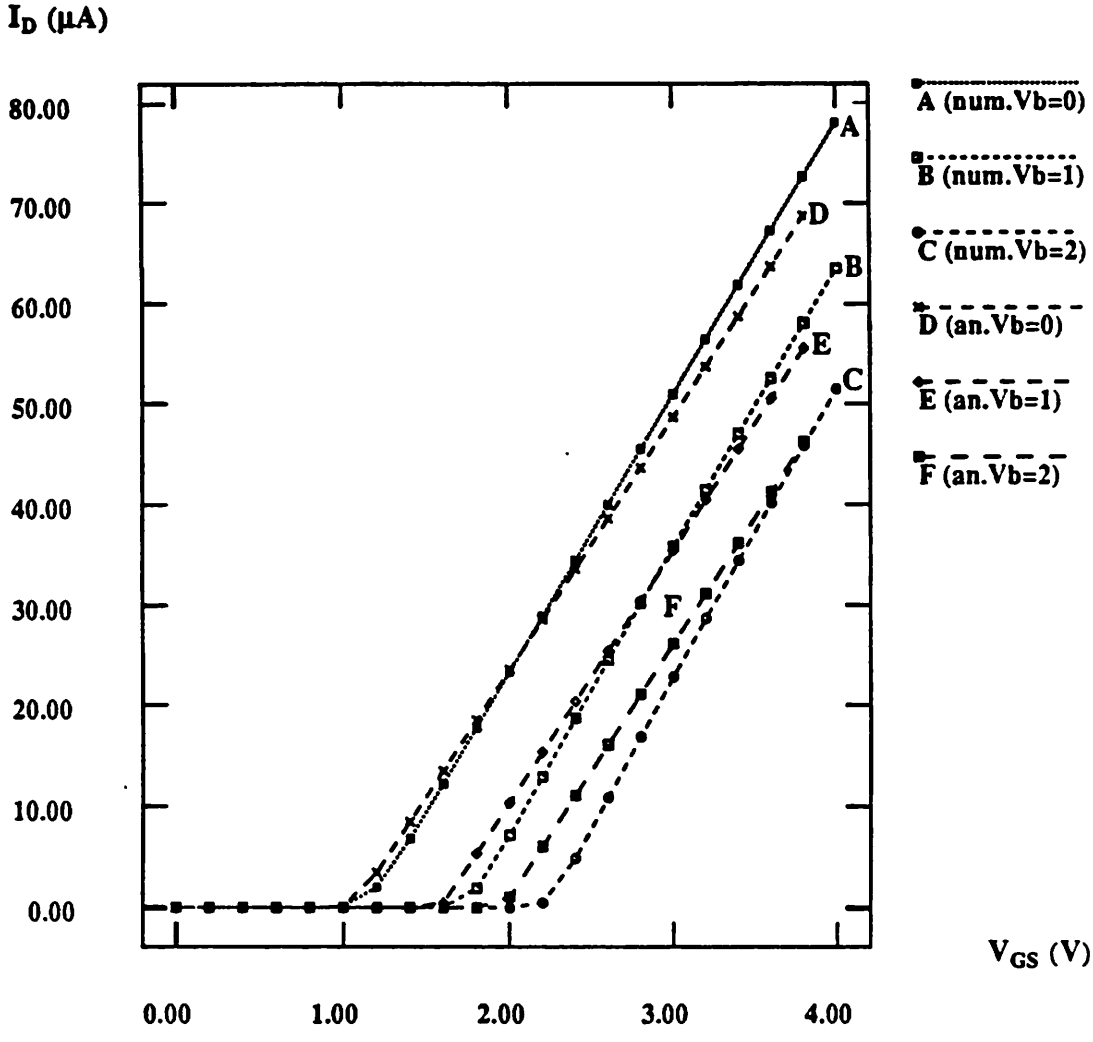


Figure 7.19(a): I_D–V_{GS} characteristics for the MOS transistor (W/L = 100μ/3μ). Simulated results from analytical and numerical models are shown for substrate bias voltages of 0V, -1V, and -2V, respectively.

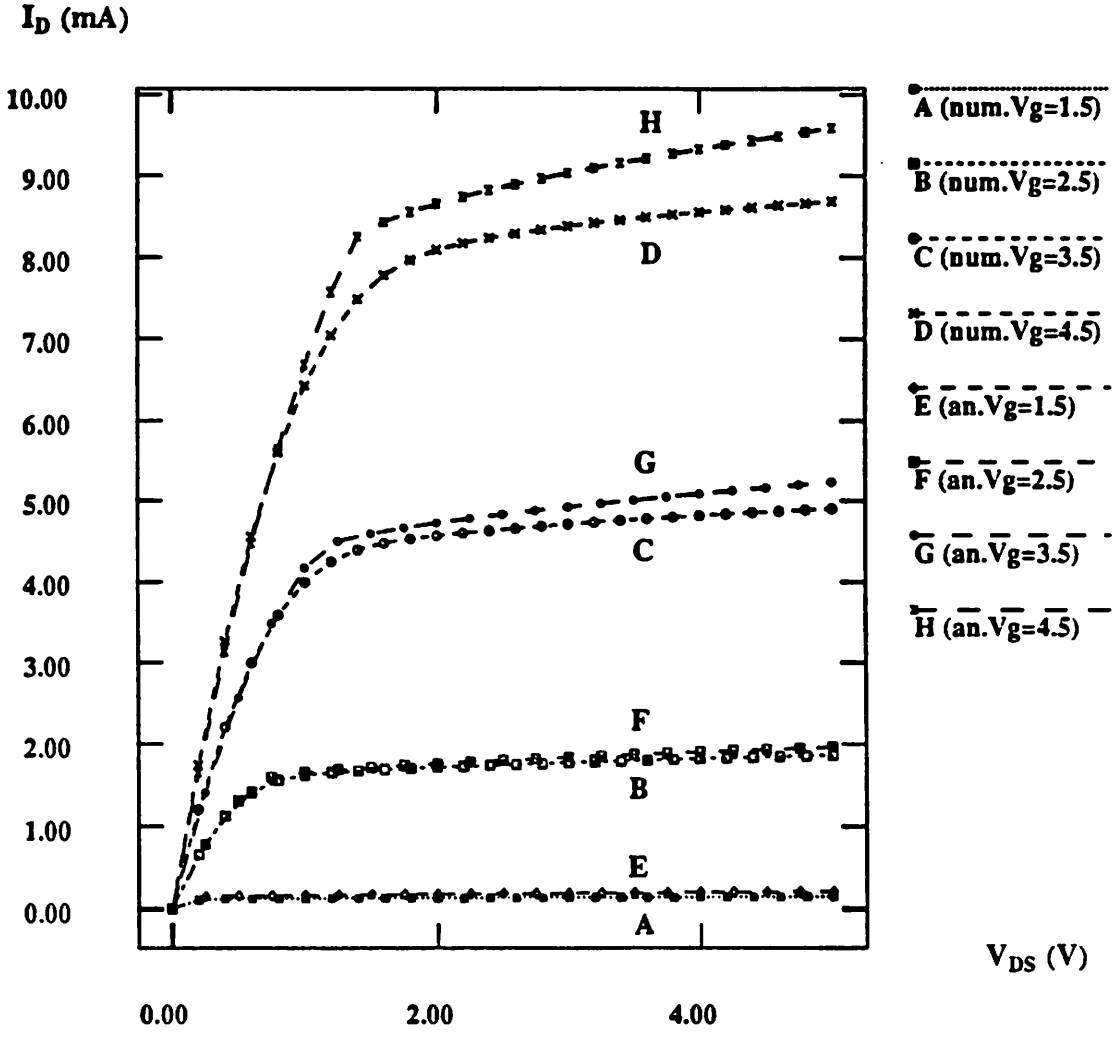
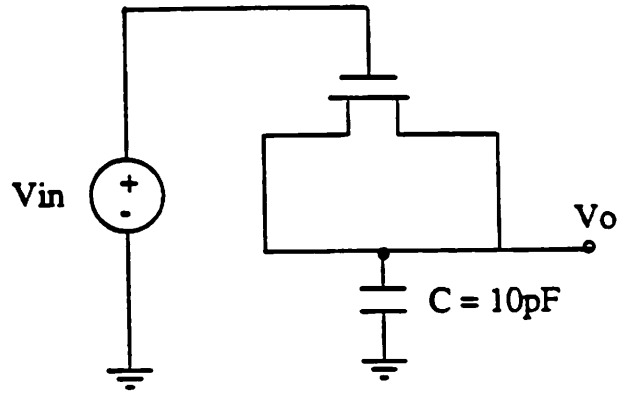


Figure 7.19(b): I_D–V_{DS} characteristics for the MOS transistor (W/L = 100μ/3μ). Simulated results from analytical and numerical models are shown for V_{GS} of 1.5V, 2.5V, 3.5V, and 4.5V, respectively.



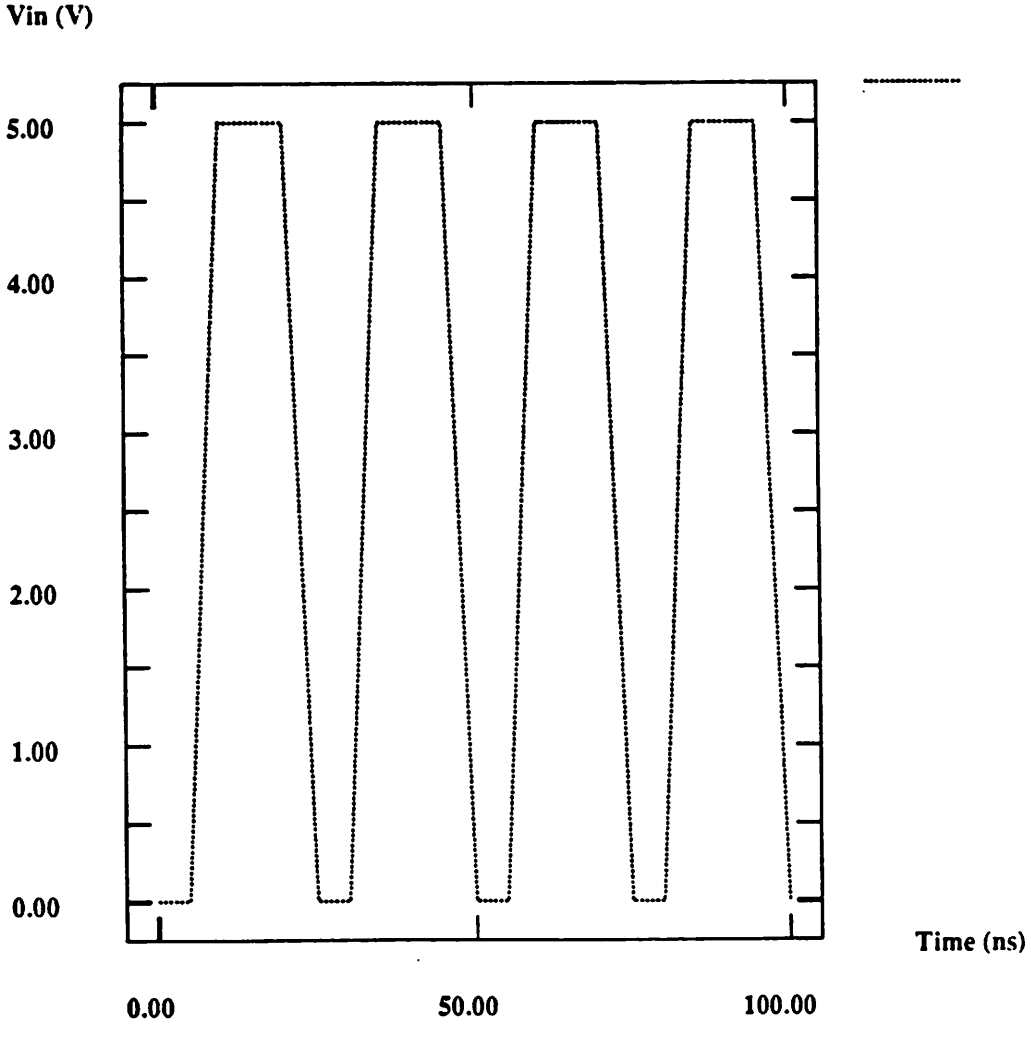
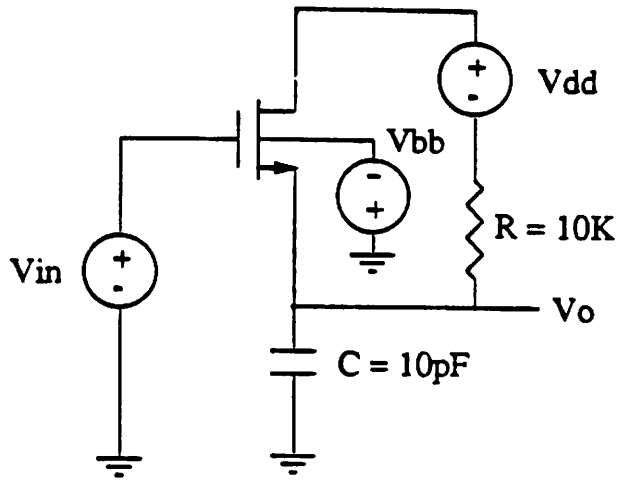


Figure 7.20(a): Bootstrap circuit of [7.6]



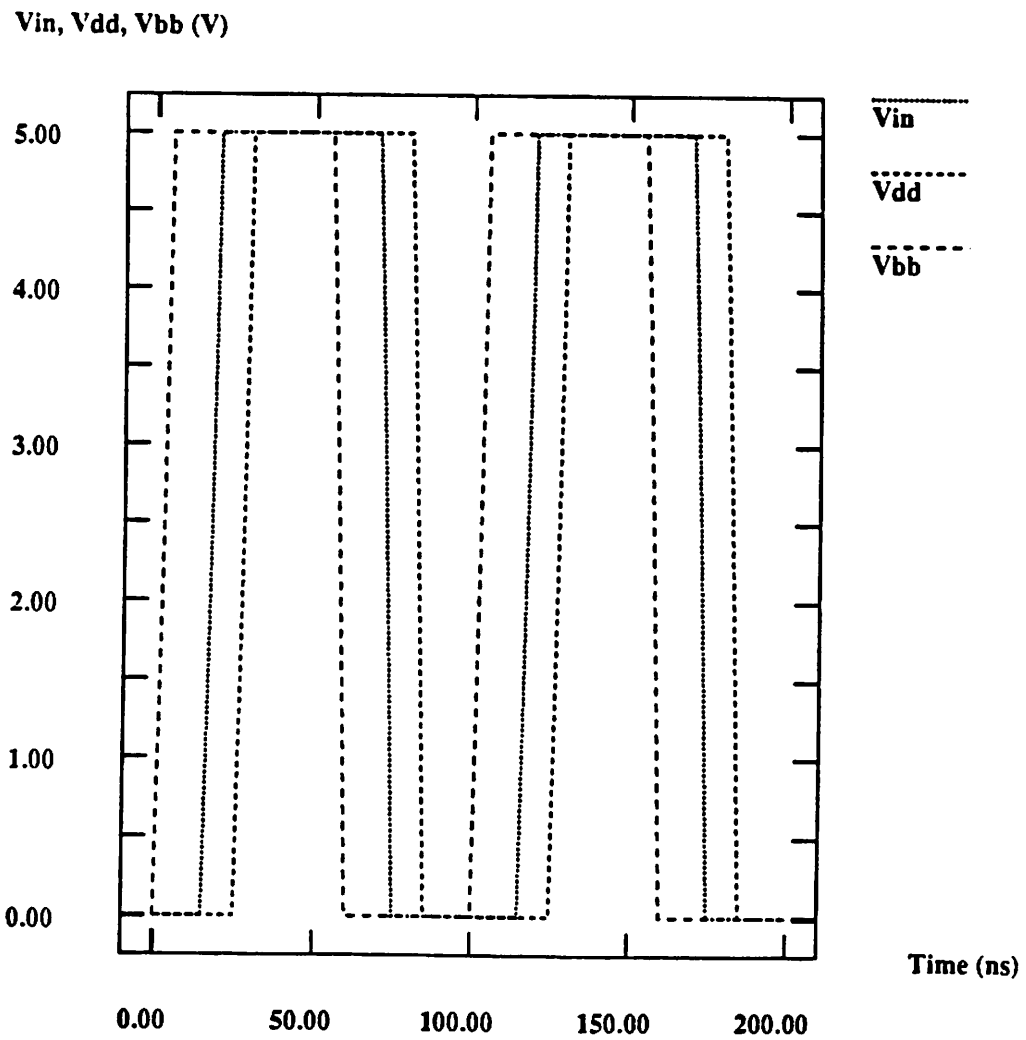


Figure 7.20(b): Charge-pump circuit of [7.4]

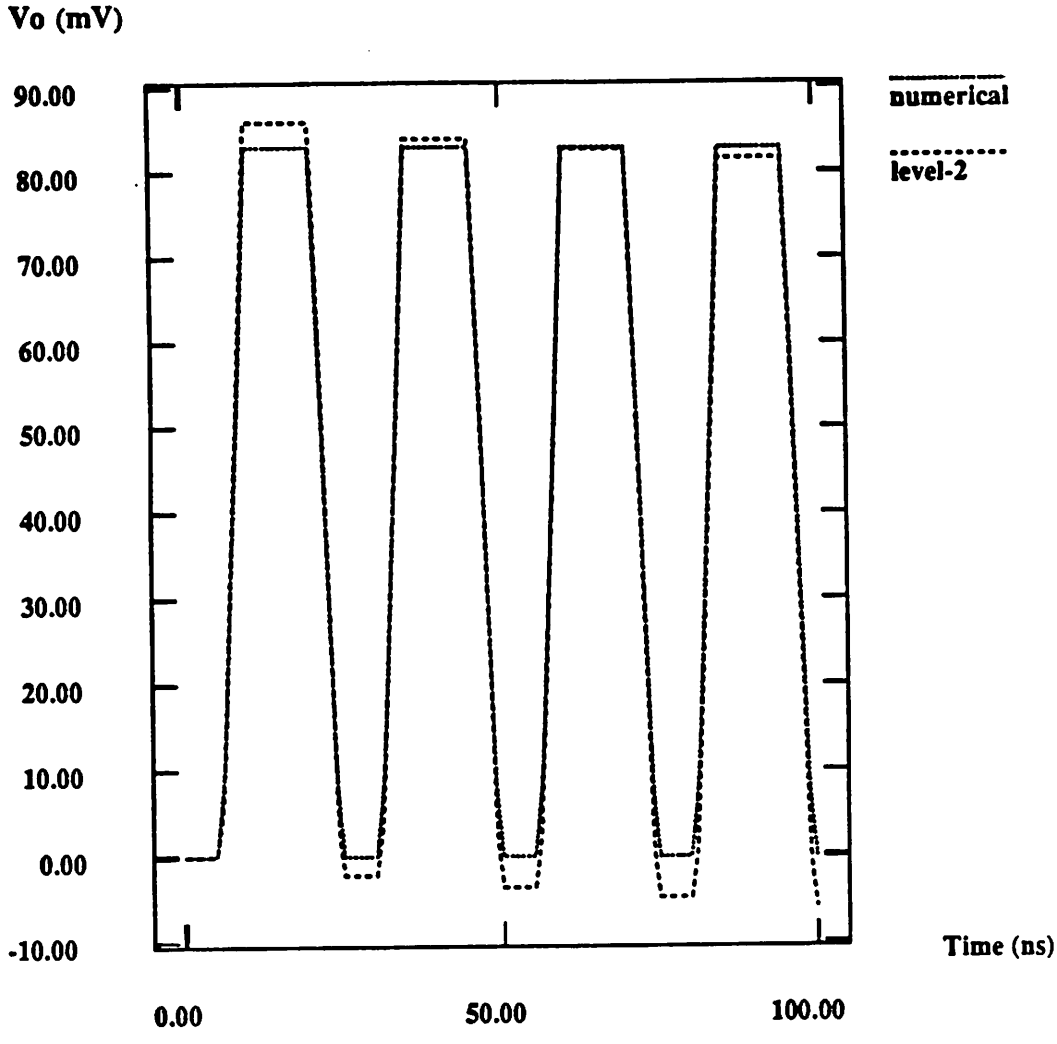


Figure 7.21(a): Output voltage for the bootstrap circuit. The numerical model depicts charge conservation; the analytical model does not conserve charge.

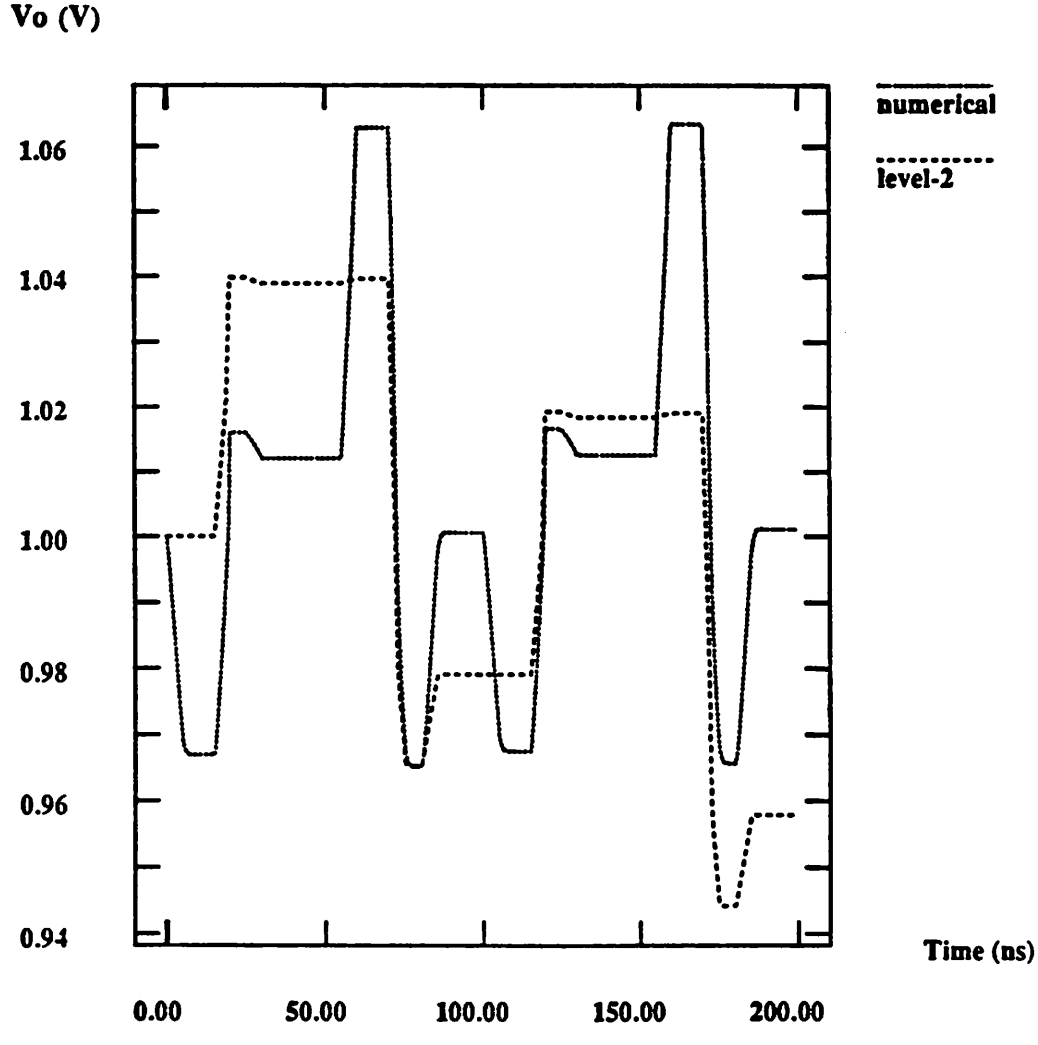


Figure 7.21(b): Output voltage for the charge-pump circuit. The analytical model does not conserve charge since the capacitor voltage decays from one cycle to the next.

ramped from 0V to 5V in 100 psec for turn on and from 5V to 0V for turn off. These are fast switching conditions for the MOS transistor and lead to non-quasi-static operation. The simulated drain and source currents as a function of time, during turn on, are plotted for the the analytical (quasi-static) model in Figure 7.23(a) and for the numerical model in Figure 7.23(b). The currents for the turn off are plotted as a function of time in Figures 7.24(a) and 7.24(b), for the analytical and numerical models, respectively. The shape of the drain and source currents and the difference between the analytical and numerical models can be explained by an examination of the surface electron concentration, shown as function of time and space in Figures 7.25(a) and 7.25(b) for turn on and turn off, respectively.

Consider the turn on of the MOS transistor. The transistor is switched from cut off, through saturation, to the linear region. Initially there is no charge in the channel since the transistor is cut off. The current that flows in the source and drain leads is the capacitive current due to gate-drain and gate-source overlap capacitances. As the gate voltage increases, the source starts injecting electrons into the channel and the source current increases with time. A certain amount of time elapses before the drain starts collecting electrons, whereby the drain current increases at a later time as seen in Figure 7.23(b). The delay is associated with the finite transit time of the carriers.

For the quasi-static model, it is assumed that the charge reaches its steady-state value instantaneously. Therefore, the drain and source currents can be expressed as the sum of a dc channel current and a capacitive current. The capacitive current is due to the channel charge partitioned between the source and the drain; the channel charge for steady-state operation is used. In Figures 7.26(a), 7.26(b), 7.26(c), and 7.26(d) are shown the schematic of the MOS model and the various capacitances for the Meyer's model as a function of the gate voltage. The simulated current waveforms can be explained in the following manner. Initially the source and drain currents are due to the

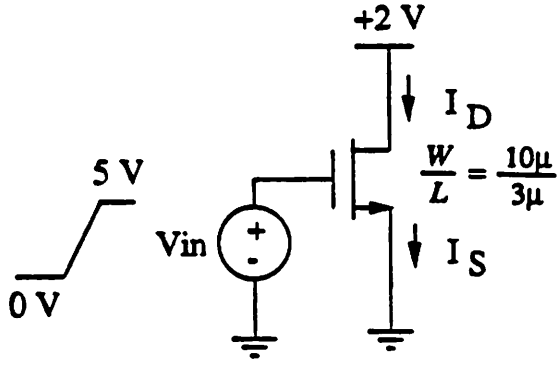


Figure 7.22: Simple circuit to study the turn-on and turn-off transients of a MOSFET.

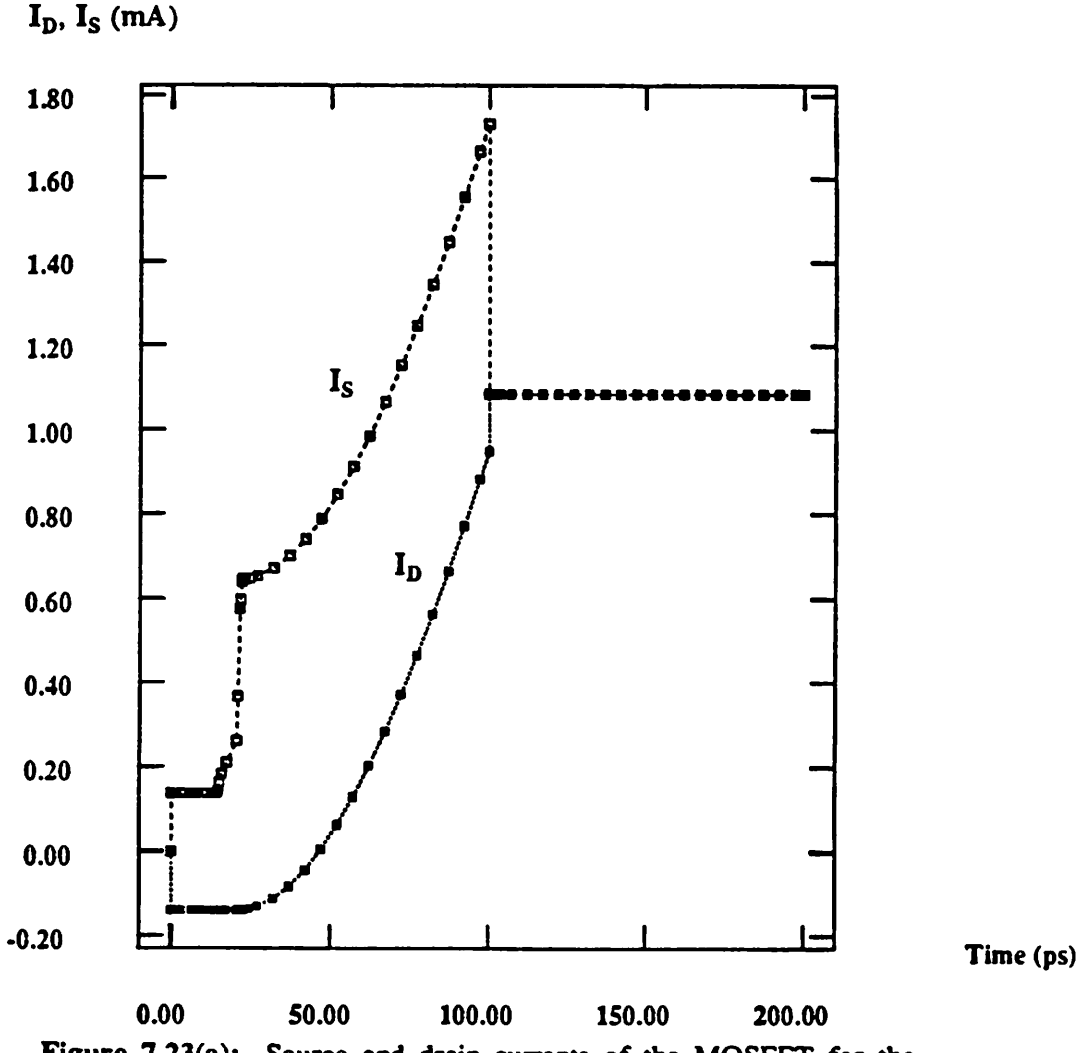


Figure 7.23(a): Source and drain currents of the MOSFET for the turn-on transient simulated with the analytical model. I_S shows a sharp jump as the transistor turns on.

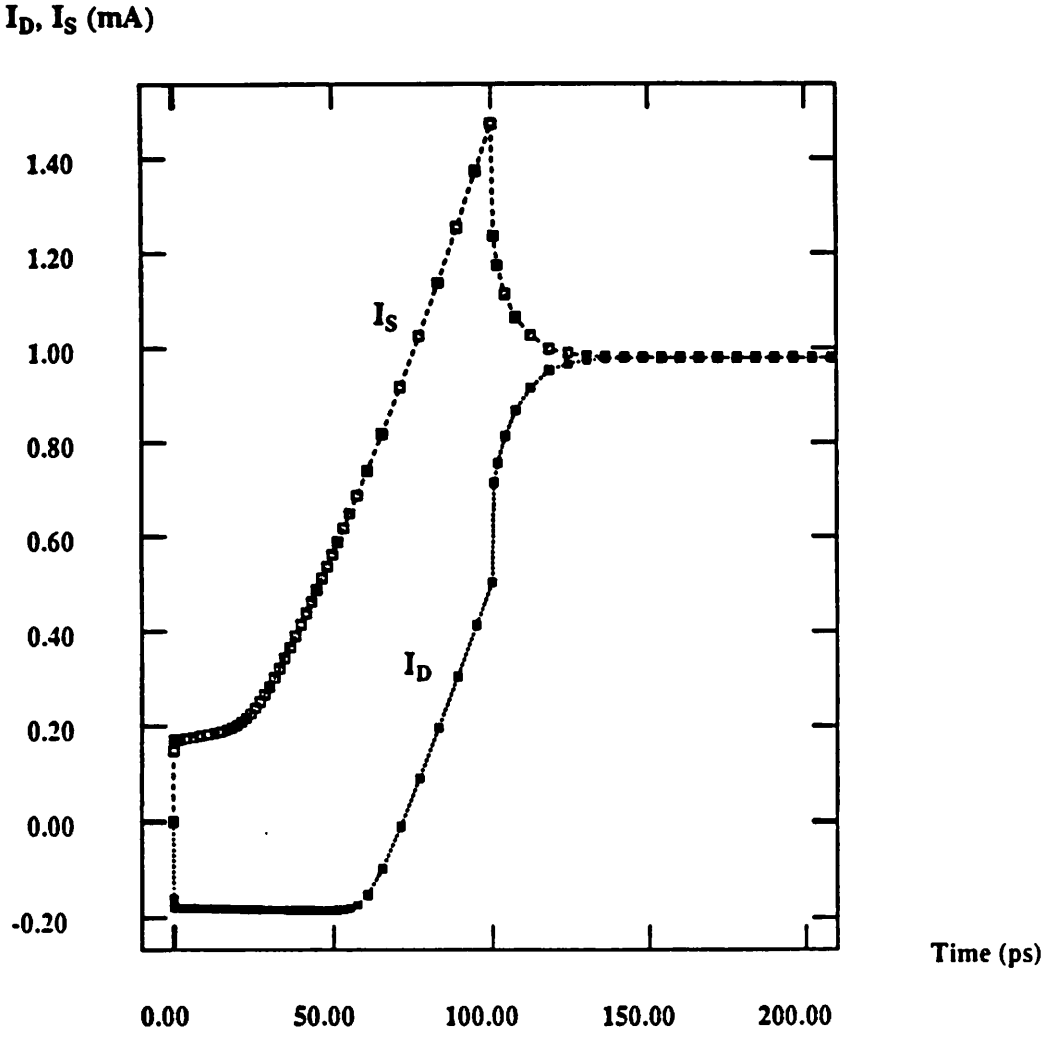


Figure 7.23(b): Source and drain currents of the MOSFET for the turn-on transient simulated with the numerical model. The currents do not change instantaneously and depict non-quasi-static operation.

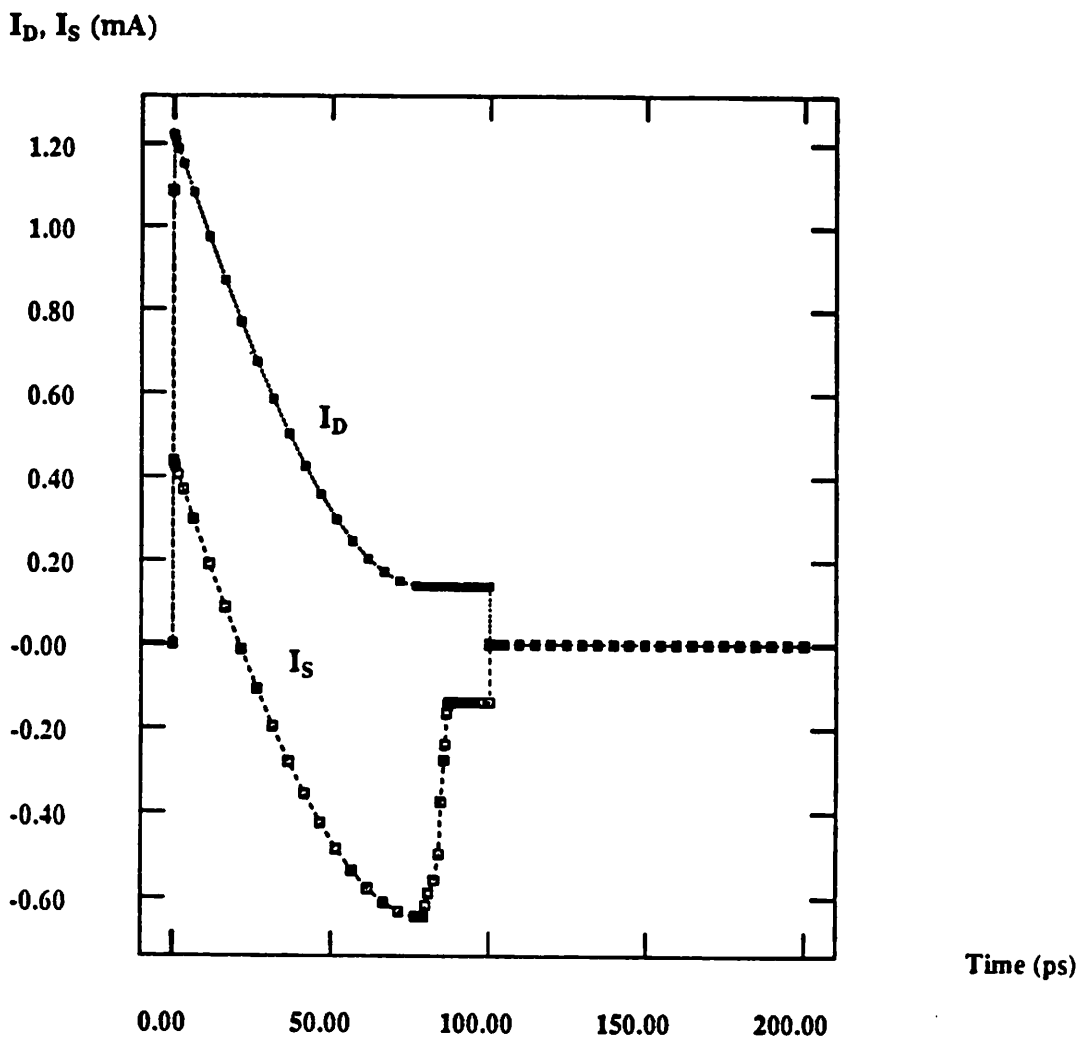


Figure 7.24(a): Source and drain currents of the MOSFET for the turn-off transient simulated with the analytical model. I_S shows a sharp jump as the transistor turns off indicating quasi-static operation.

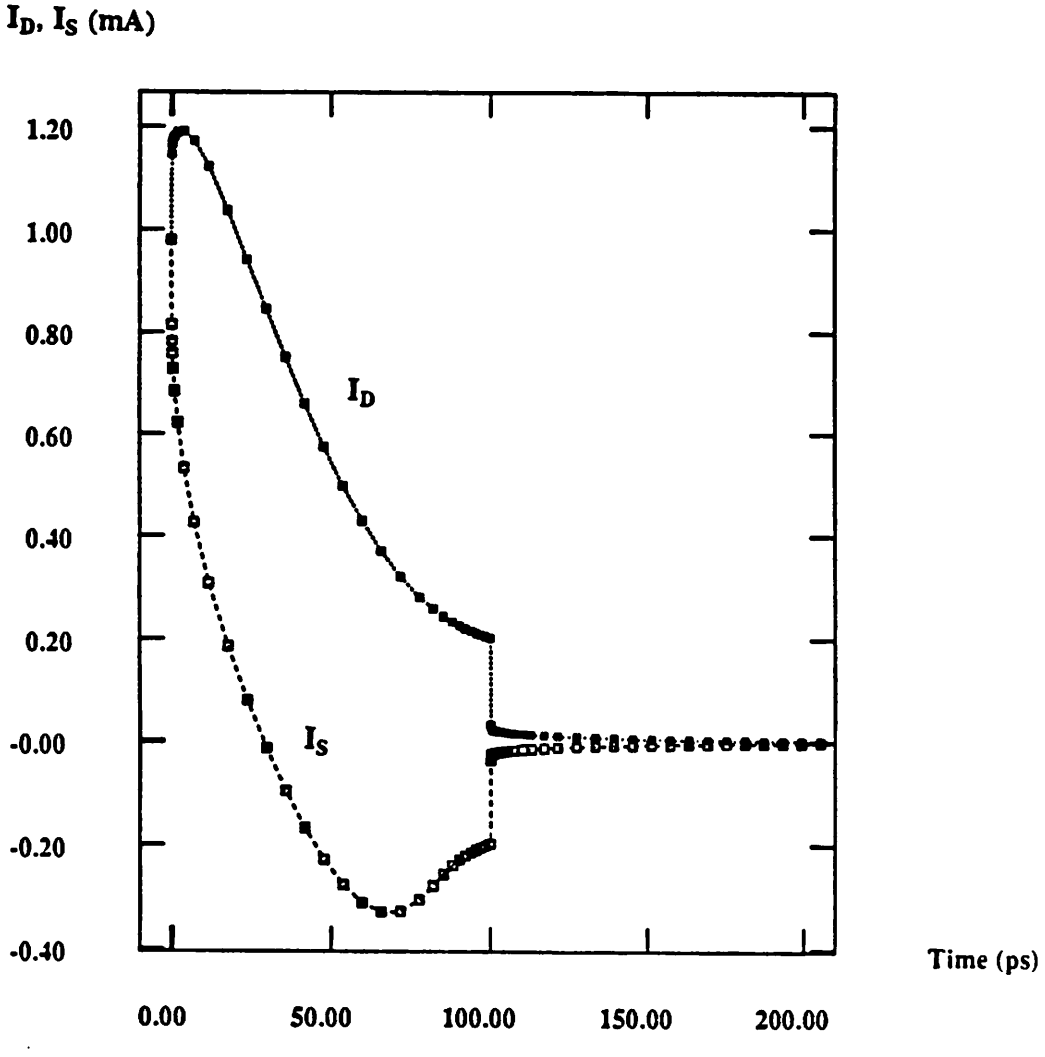


Figure 7.24(b): Source and drain currents of the MOSFET for the turn-off transient simulated with the numerical model. The currents do not change instantaneously and depict non-quasi-static operation.

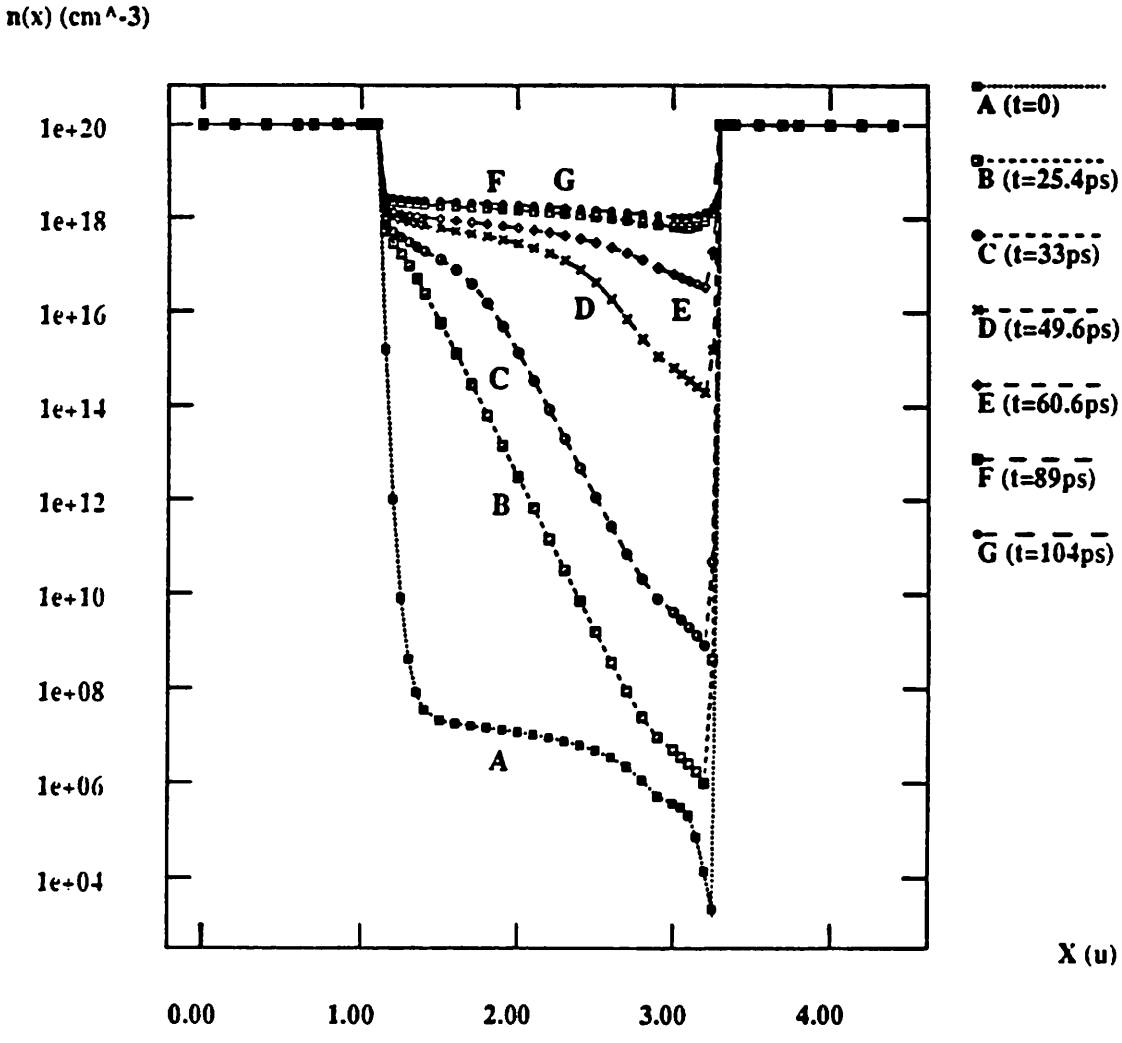


Figure 7.25(a): Surface electron concentration as a function of time for the turn-on transient.

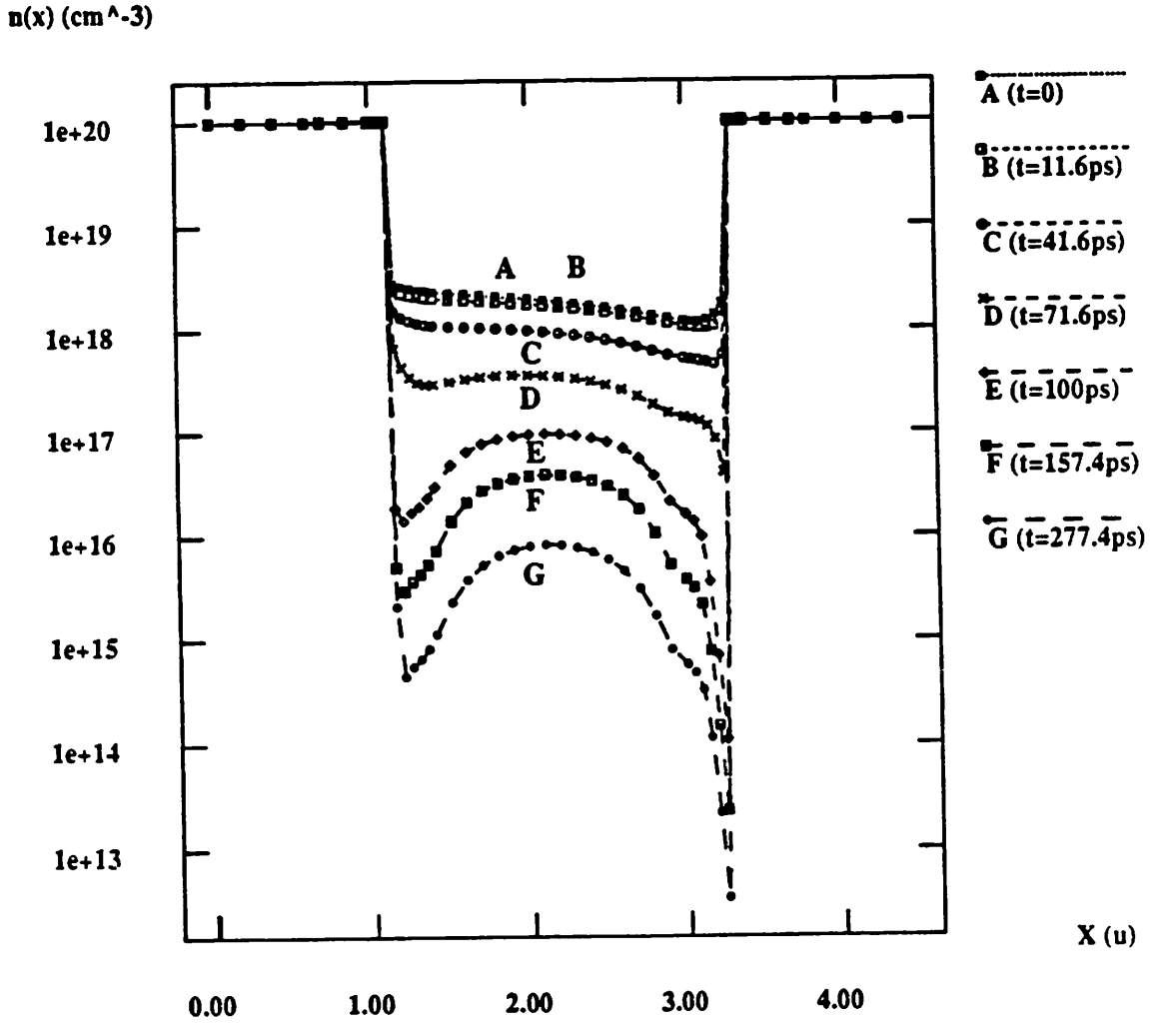


Figure 7.25(b): Surface electron concentration as a function of time for the turn-off transient.

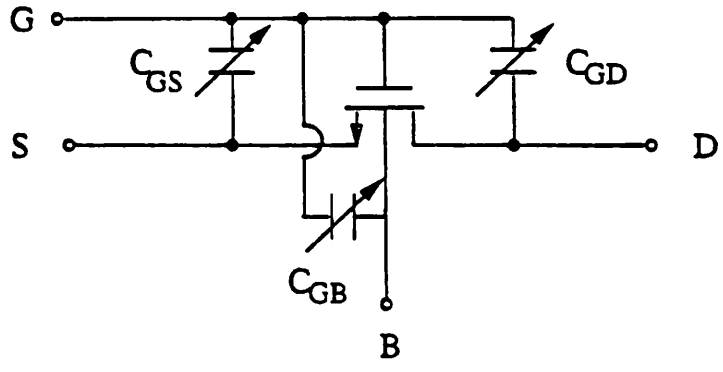


Figure 7.26(a): Schematic of MOS transistor with the capacitances for the Meyer model [7.7].

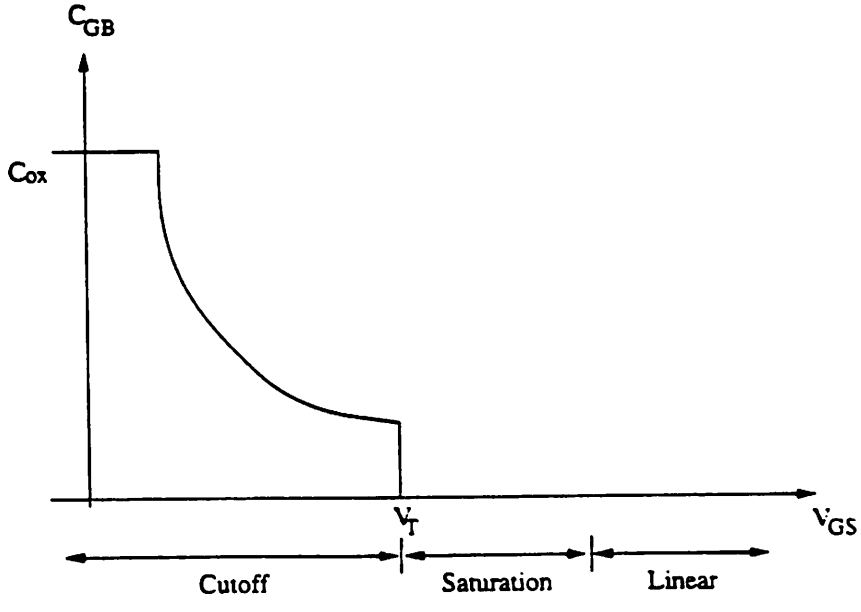


Figure 7.26(b): C_{GB} as a function of the gate voltage.

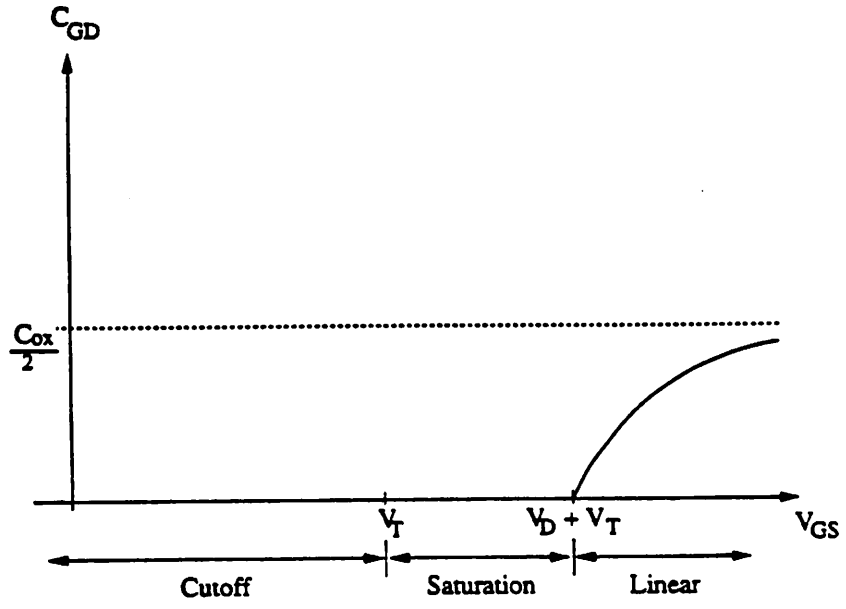


Figure 7.26(c): C_{GD} as a function of the gate voltage.

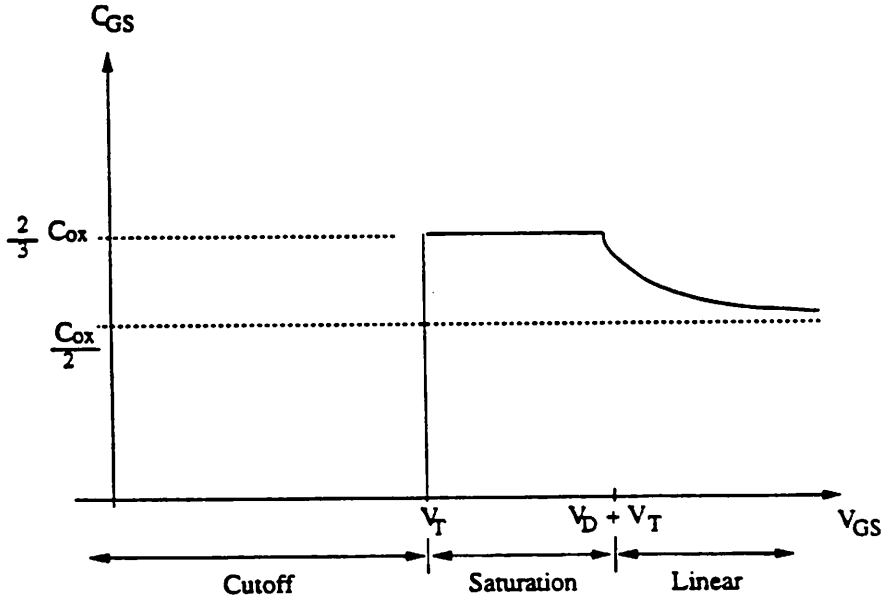


Figure 7.26(d): C_{GS} as a function of the gate voltage.

overlap capacitances. When the gate voltage exceeds the threshold voltage the transistor turns on, the gate-source capacitance steps from 0 to $2/3C_{ox}$ and the gate-drain capacitance is zero. Thus the source current steps instantaneously, whereas the drain current does not show a step since it is only due to the dc channel current, as seen in Figure 7.23(a). However, the drain current does increase, as the dc current increases, thereby exhibiting a zero delay between the source and drain currents. A charge-based capacitance model such as that due to [7.4, 7.7, 7.8] would not depict the step in the source current, but a zero delay will exist between the source and drain currents due to the assumption of quasi-static operation.

When the gate voltage reaches its steady-state value and becomes a constant, the capacitive currents due to the overlap capacitances become zero and there is a step in the drain and source currents. The quasi-static analytical model predicts that the drain and source currents reach their steady-state values instantaneously, whereas the numerical model depicts that the source (drain) current decreases (increases) exponentially to its steady-state value. This indicates that the channel-charge does not relax instantaneously to its steady-state value; there is a time-constant associated with it [7.9]. Similar arguments can be used to explain the turn-off characteristics of the source and drain current.

The non-quasi-static operation of the MOS transistor can be illustrated more dramatically with the pass transistor circuit of Figure 7.27. The gate voltage is ramped from 5V to 0V in 100ps. The drain current is shown as a function of time in Figure 7.28 for both analytical and numerical models. The analytical model indicates a sudden increase in drain current after which the drain current remains constant and drops to zero immediately as the gate voltage goes to zero. The non-quasi-static current has a completely different behavior as seen from Figure 7.28.

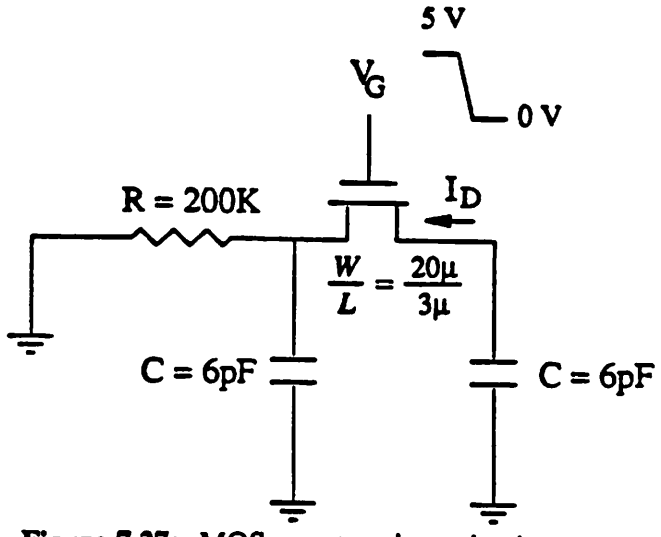


Figure 7.27: MOS pass-transistor circuit.

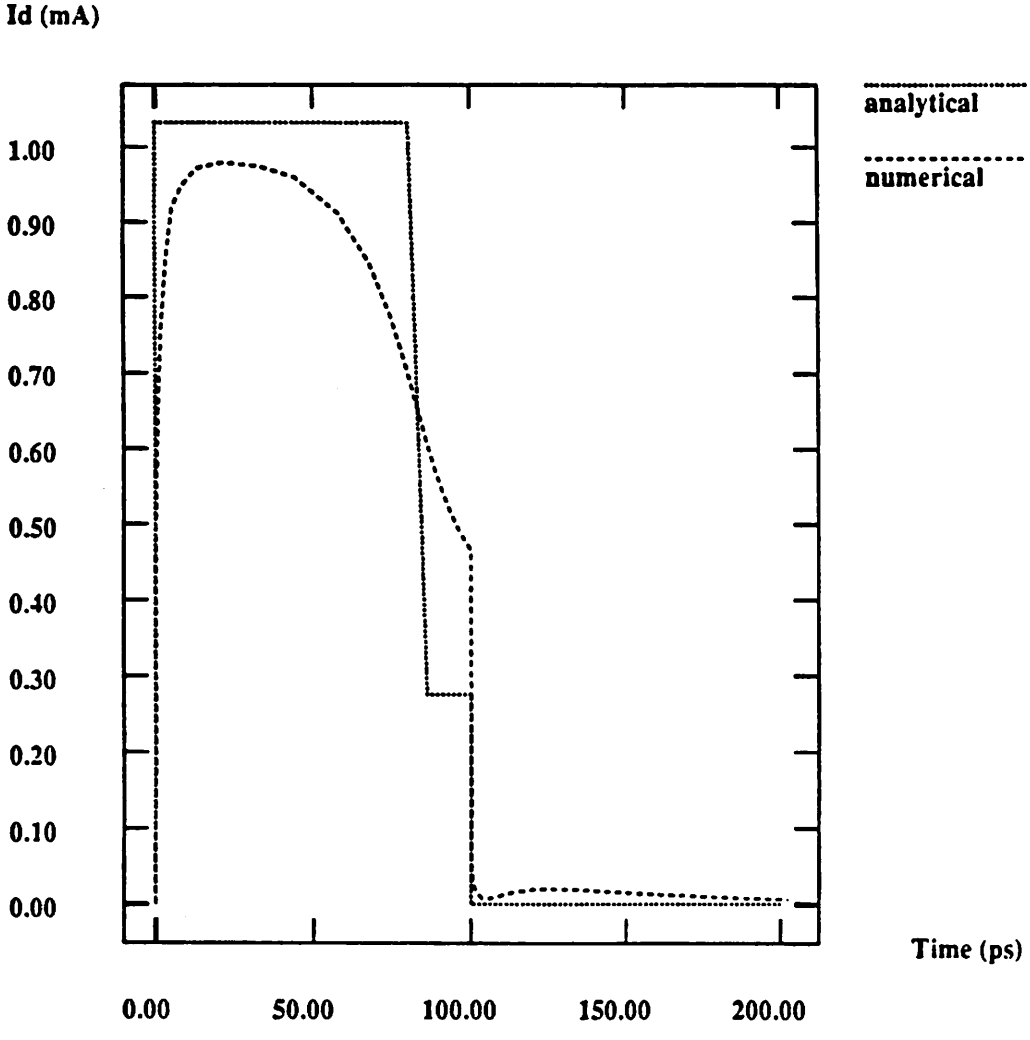


Figure 7.28: Simulated drain currents as a function of time. The analytical model depicts quasi-static operation since the current changes instantaneously.

7.5. Runtime Comparisons with Analytical Models

Simulation with analytical models will always be faster in performance compared to that with numerical models. This is to be expected since analytical models only require function evaluations to compute the conductances and currents of a device. With a numerical model the device-level matrix has to be decomposed into its LU factors and forward and back substitutions are required to compute the currents and conductances. The latter task requires more CPU time resulting in the higher computational cost associated with numerical models. However, under several situations an analytical model may give inaccurate results due to its simplicity in modeling the actual physical behavior of a device. Thus a comparison between numerical and analytical models is not appropriate. Numerical models will be used only in situations where analytical models are inadequate, and computation cost should not be of concern when accuracy is desired. Nonetheless, the comparison is given to provide an estimate of the increase in computational effort when using a numerical model.

In Table 7.1 a comparison is presented between numerical and analytical models for the benchmark examples. The results for the numerical simulation are obtained by the use of trapezoidal integration method. The numbers for the various examples are in the following sequence: number of iterations, number of timepoints accepted, number of timepoints rejected, total number of timepoints, and the runtimes in seconds on a VAX 8800. From Table 7.1 it is clear that the numerical models require a larger number of timepoints; and, hence, a larger number of iterations. Numerical models require larger number of timepoints because they are solving for the Poisson's and the electron and hole current-continuity equations, which have time constants that are quite different from the circuit. Since the timesteps are decided based on the activity internal to a device, smaller timesteps will be used to resolve accurately the carrier concentrations, requiring a larger number of timepoints to complete the transient analysis.

Circuit	Numerical Model	Analytical Model
Oscillator	16127	4479
	3491	1022
	705	488
	4196	1510
	2063	12.9
VCO	5890	2174
	1003	531
	209	126
	1212	657
	3000	20
Invchain	1738	360
	322	83
	66	15
	388	98
	545	2.2
Astable	5744	2630
	1107	575
	216	218
	1323	793
	1139	10
MECLgate	1966	497
	429	138
	55	25
	484	163
	1680	8.2
Pass	240	150
	67	71
	4	2
	71	73
	865	0.6
MOSinv	326	225
	86	79
	7	13
	93	92
	1120	0.8
ChargePump	1853	1430
	399	441
	81	97
	480	538
	4124	6.2

Table 7.1: Comparison of numerical and analytical models

The data of Table 7.1 is summarized in Table 7.2 where the ratio of the CPU time required by use of numerical models to that required for analytical models is presented. The first column presents the comparison on a time per iteration basis and the second column presents a ratio of the total analysis time. Since a larger number of timepoints are required with the numerical models the simulation time will be larger. For one-dimensional models the simulation time is slower by factors of 114 to 250. This number is larger for the MOS examples using one-carrier simulation; the overall simulation time is about 1500 times larger compared with the analytical models.

Comparison on a time per iteration basis provides an idea of the the raw speed of each model. With one-dimensional examples the numerical model is slower by a factor of 50 and approximately 1000 for the MOS examples. This ratio can be made worse by using a larger number of grid points for the numerical model. However, numerical models can be used even when analytical models fail. Under such circumstances additional computational effort is justified.

Circuit	Numerical / Analytical (time/iteration)	Numerical / Analytical (analysis time)
Oscillator	44.4	160
VCO	55	150
Invchain	51.3	248
Astable	52	114
MECLgate	51.8	205
Pass	901	1442
MOSinv	966	1400
ChargePump	513	665

Table 7.2: Comparison of analysis times

A tradeoff between runtimes and simulation accuracy can be achieved by modeling only the critical devices numerically and using analytical models for the rest of the circuit. The BiCMOS driver circuit is a typical example of such an approach. Since modeling of bipolar transistors is critical for simulations, numerical models have been used for the bipolar transistor, whereas the MOSFET is modeled by an analytical model. Use of a two-dimensional numerical MOSFET model would not have provided additional insight into the operation of the circuit. Mixed-level simulation allows the use of detailed simulation models for the critical devices and less accurate but faster models for the noncritical devices.

CHAPTER 8

Applications of CODECS

8.1. Introduction

A variety of applications of CODECS are presented in this chapter. The applications span from use of only the device-simulation capability to use of the mixed-level circuit and device-simulation environment. All of these applications demonstrate the usefulness of CODECS as a simulation program. In addition, CODECS can be used to evaluate technology tradeoffs; hence, it provides a predictive capability.

The first application illustrates the use of CODECS for estimating the delay of BiCMOS driver circuits. While driving large capacitive loads, the bipolar transistors operate in high-level injection. As indicated in Chapter 7, commonly available SPICE bipolar transistor models are inadequate for simulating high-level-injection effects critical to the operation of BiCMOS circuits. Therefore, SPICE simulations cannot be used to accurately estimate the delay. With the use of physical models in CODECS, however, it is possible to analyze the delay and to evaluate the effect of technology and supply voltage scaling on the delay of the BiCMOS driver.

The second application concerns power devices. Simulation examples in Chapter 7 show that the SPICE diode model is inadequate to study the turn off of p-i-n diodes with a resistive load. For an inductive load, the situation is even more complicated. Consequently, the problem is extremely difficult to model analytically and coupled device and circuit simulation provides a way to study the turn-off characteristics. The soft recovery

of p-i-n rectifiers is investigated. CODECS is also used to study the transient breakdown of p-i-n rectifiers with an inductive load. Oscillations are observed when operation is near the breakdown voltage. A physical model is used to explain this phenomena based on insight gained from CODECS simulations.

The problem of charge injection in MOS switched-capacitor circuits is examined in Section 8.4. The numerical MOS model in CODECS conserves charge and does not use an empirical scheme for channel-charge partitioning. Therefore, the charge that flows out of the various terminals of a MOSFET during the turn off is calculated accurately, and CODECS can be used to study switch-induced error voltages. Thus, different error-cancellation schemes can be evaluated in terms of their effectiveness in reducing the switch error. Although analytical results are available under very specific conditions, no general solution exists. CODECS can be used to simulate realistic driving source conditions. In addition, it also allows simulation of non-quasi-static MOS operation that can influence the charge transfer in switched-capacitor circuits. A comparison of the analytical and numerical MOS models in Chapter 7 demonstrated the inadequacy of analytical MOS models for non-quasi-static operation.

As a last application, CODECS is shown to be a tool that can be used to verify analytical models. The numerical models can be used as a reference with respect to which analytical models can be evaluated. An analytical model for the lateral electric field in lightly doped drain MOSFETs and a non-quasi-static MOSFET model are used as examples of such an application.

8.2. Analysis of BiCMOS Driver Circuits

The BiCMOS driver circuit shown in Figure 8.1 is a common configuration used in BiCMOS circuits. This gate is used in static memories [8.1] and gate arrays [8.2] to drive large output capacitors. For large capacitive loads the collector current will be

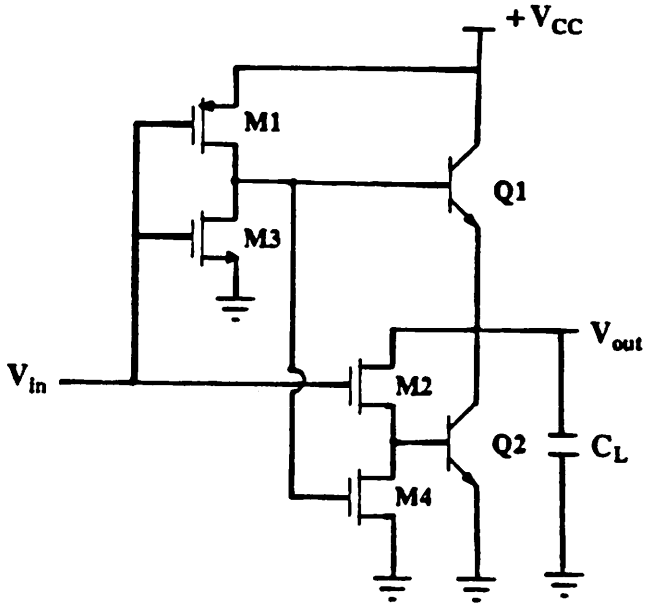


Figure 8.1: BiCMOS driver circuit.

large and the bipolar transistor operates in high-level injection in the collector, i.e., quasi-saturation or Kirk effect [8.3, 8.4]. SPICE simulations are inadequate since high-level-injection effects critical to the performance of the bipolar transistors are not accurately modeled with commonly available SPICE bipolar transistor models. Simulations from CODECS can be used since base pushout under high-level injection is properly modeled with the numerical bipolar transistor models. In addition, CODECS simulations can be used to study the effect that IC fabrication technology and supply voltage scaling have on the delay of the gate.

The delay of the BiCMOS driver depends on the region of operation of the bipolar transistors. Three different regions of operation exist, namely, operation under low-level-injection conditions, operation in high-level injection, and collector-resistance dominated operation. The relative values of the device parameters and the load capacitance determine the operating region of the bipolar transistors. Although various delay models have been considered for the BiCMOS buffer circuit, they are restricted to one region of operation. In [8.5] only high-level-injection effects in the base region are considered which are not as important as the high-level-injection effects in the collector region. The analysis of [8.6], on the other hand, considers only the collector-resistance dominated case and assumes that high-level-injection effects do not occur. In practice, while driving large load capacitors both high-level injection in the collector and the collector resistance must be taken into account.

8.2.1. Delay Analysis

A delay model is necessary to identify the important bipolar device parameters and their effect on the performance of the gate. CODECS simulations provide a means for verifying analytical delay formulas. In this section the delay for operation under high-level injection is examined. A detailed analysis for the other regions is available in [8.7].

The delay of the BiCMOS driver is a function of both the MOSFET and the bipolar transistor parameters. For the bipolar transistor the parameters that influence the delay are: the current gain, β (a function of the collector current), the base transit time, τ_f (also a function of the collector current), the collector current at onset of high-level injection in the collector, I_K, and the parasitic capacitances and resistances. For the MOSFET the parameters are: the transconductance parameter k', the W/L, the threshold voltage, V_T, and the parasitic capacitances. The delay of the gate is estimated as a function of these parameters.

Consider the pull-up transient of the gate, the circuit diagram for the pull-up circuit is shown in Figure 8.2. The pull-down transient is similar and will not be considered. The input voltage is stepped from V_{DD} to 0V and the MOS transistor turns on. The MOSFET can be replaced by a current source of value equal to the average MOSFET current and the circuit to be analyzed is shown in Figure 8.3. If the transit time of the transistor is small, this circuit can be analyzed by modeling the degradation in β due to the base-pushout effect. However, as shown later, the transit-time effects cannot be ignored when base pushout occurs and the analysis based on β degradation does not suffice.

Assume that the load capacitor is initially uncharged and that the transit-time effects are negligible, the delay can then be expressed as

$$T_D = \frac{V_{BE(on)}C_1}{I_D} + \frac{C_L V_S}{2I_C} \tag{8.1}$$

where I is the average current of the MOS transistor, C_1 is the total capacitance at the base of the bipolar transistor, C_L is the total output load capacitance, V_S is the output voltage swing, and I_C is the collector current. The first term accounts for the time required to charge up the base of the bipolar transistor to $V_{BE(on)}$ and is the time taken to charge C_1 to a voltage $V_{BE(on)}$ with a current I_D. The second term models the time

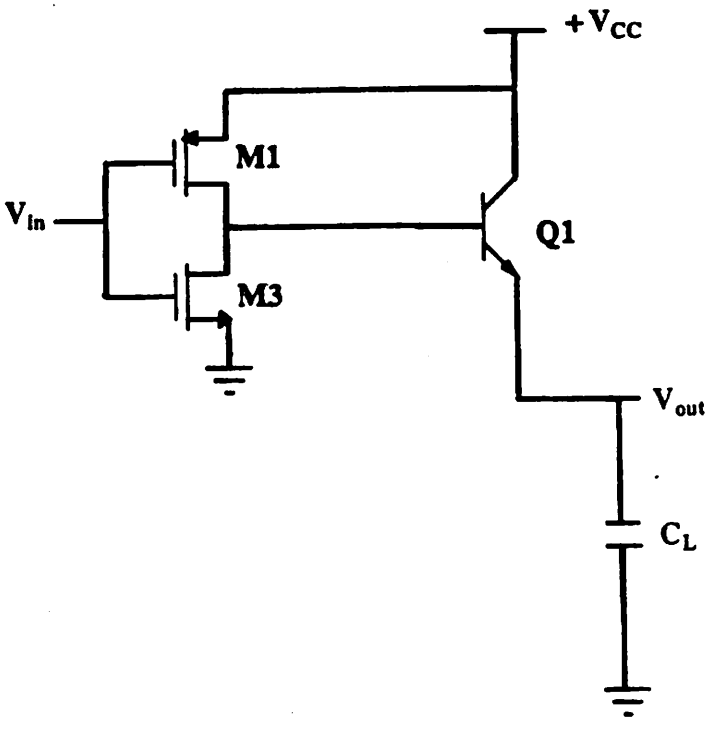


Figure 8.2: Circuit diagram of the pull-up circuit.

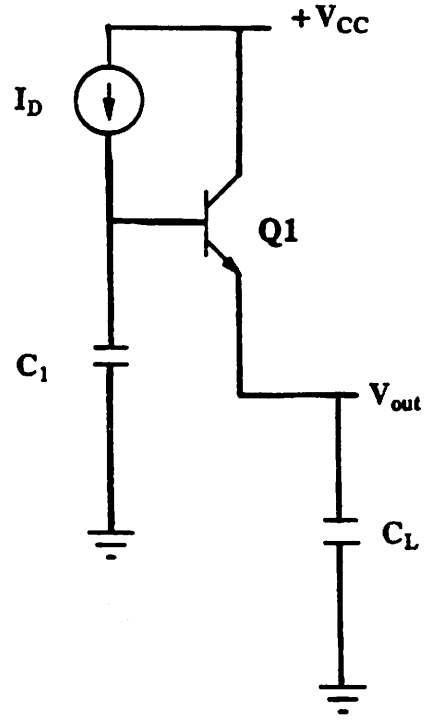


Figure 8.3: Pull-up circuit used for first-order delay analysis.

required to charge the output capacitor to half the voltage swing (V_S) with a current I_C. The first term of the delay equation is usually much smaller than the second term.

Under high-level injection the current gain β can be related to the current gain under low-level injection conditions β_0 by [8.8]

$$\beta = \frac{\beta_0}{1 + \frac{I_C}{I_K}} \approx \beta_0 \frac{I_K}{I_C} \tag{8.2}$$

The collector current is then given by

$$I_C = \beta I_B = \beta I_D = \beta_0 \frac{I_K}{I_C} I_D \tag{8.3}$$

or

$$I_C = \sqrt{\beta_0 I_D I_K} \tag{8.4}$$

Thus, the delay can be written as

$$T_D = \frac{V_{BE(on)} C_1}{I_D} + \frac{C_L V_S}{2\sqrt{\beta_0 I_D I_K}} \tag{8.5}$$

From CODECS simulations it is seen that the above formula is inaccurate when transit-time effects are important. However, by the use of a fitting parameter η, Equation (8.5) can be used to provide a first-order estimate of the delay.

$$T_D = \frac{V_{BE(on)} C_1}{I_D} + \frac{\eta C_L V_S}{2\sqrt{\beta_0 I_D I_K}} \tag{8.6}$$

The fitting parameter η depends on the value of C_L and can be determined from CODECS simulations. In Figure 8.4 the simulated and calculated delays are plotted as a function of the area of the bipolar transistor for a load capacitor of 10pF. As the area of the bipolar transistor increases, I_K increases; hence, the delay goes down. Equation (8.6) provides a good approximation for the delay. The delay as a function of the width

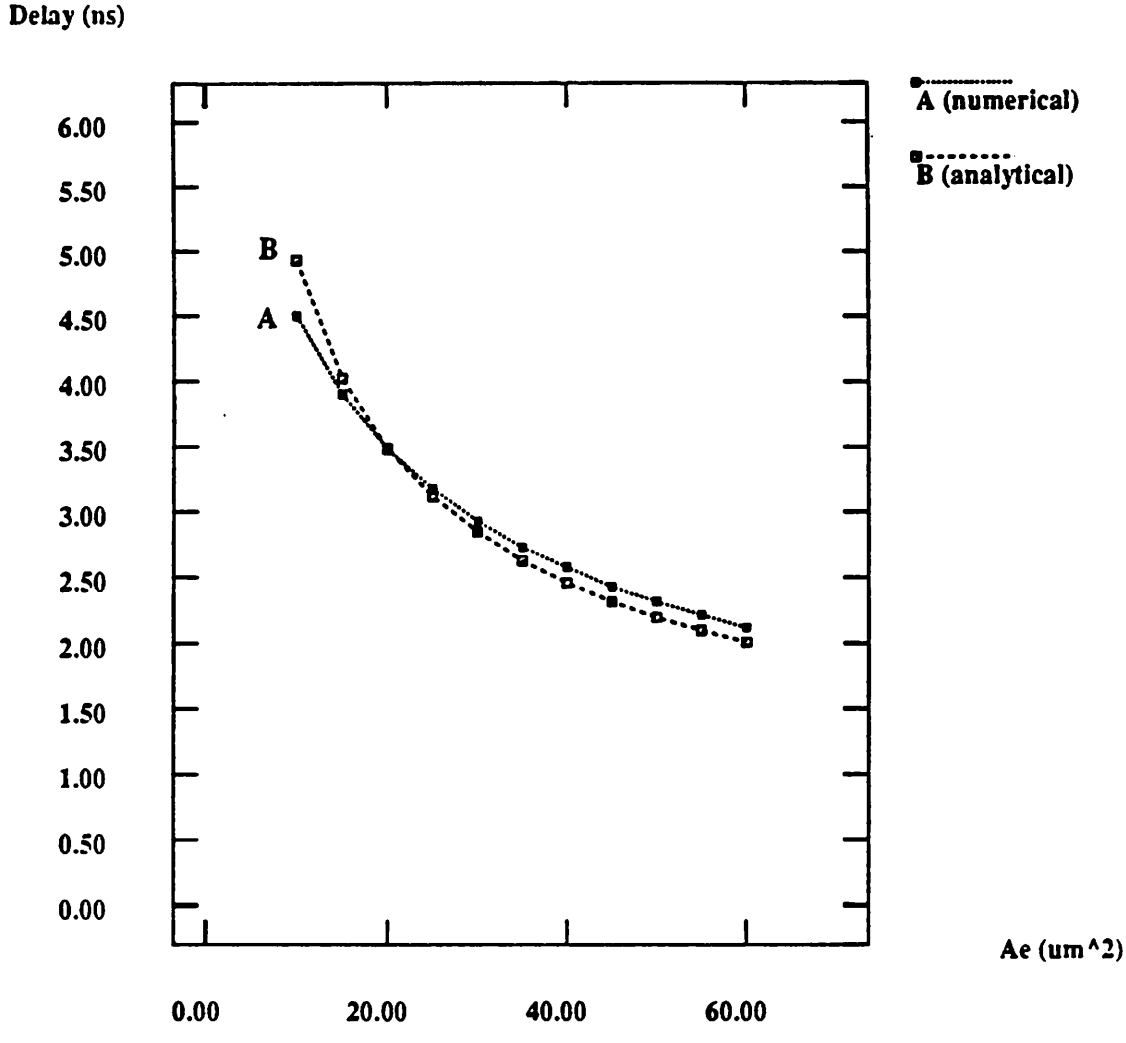


Figure 8.4: Simulated and approximated delay as a function of the emitter area of the bipolar transistor.

of the MOS transistor for a load capacitance of 10pF is shown in Figure 8.5. Again the approximate delay equation provides a reasonable estimate of the delay. As seen from Equation (8.6), an increase in W increases I_D and the delay goes down.

8.2.2. Technology and Supply-Voltage Scaling Effects

Simulations have been performed to study the influence of different doping profiles on the delay of the BiCMOS driver. It should be noted that such simulations are possible only with CODECS and not with a SPICE-like circuit simulator. As pointed out in Chapter 7, conventional circuit simulation uses analytical models for which, in general, it is difficult to express the dependencies of model parameters on doping profiles.

The simulated collector current for the pull-up transient is shown in Figure 8.6 for a one-dimensional numerical bipolar transistor model with the doping profile of Figure 8.7. Initially the current rises rapidly, this rise being determined by τ_f. At a certain current level (approximately 2.5mA in this case) the rate of increase of current decreases. This takes place when high-level injection occurs in the collector of the bipolar transistor. The base widens, the β of the transistor decreases, and the transit time increases. As a consequence the current increases at a smaller rate.

From Figure 8.6 and Equation (8.6) it is clear that a high I_K is desirable to reduce the delay. I_K depends on the doping and thickness of the epi layer [8.3, 8.4]. An increase in the epi-layer doping or a reduction in the epi-layer thickness results in a larger I_K. It has been shown in Chapter 7 that a transistor with a buried layer has a higher I_K compared to one without the buried layer; therefore, a buried layer is essential for reducing the delay.

In Figure 8.8 simulated collector currents are plotted as a function of time for transistors with an epi-layer doping of 8×10^{15} and a buried layer. The doping profile of the transistor, for an epi-layer thickness of $2.1\mu m$, is shown in Figure 8.7. The epi-layer

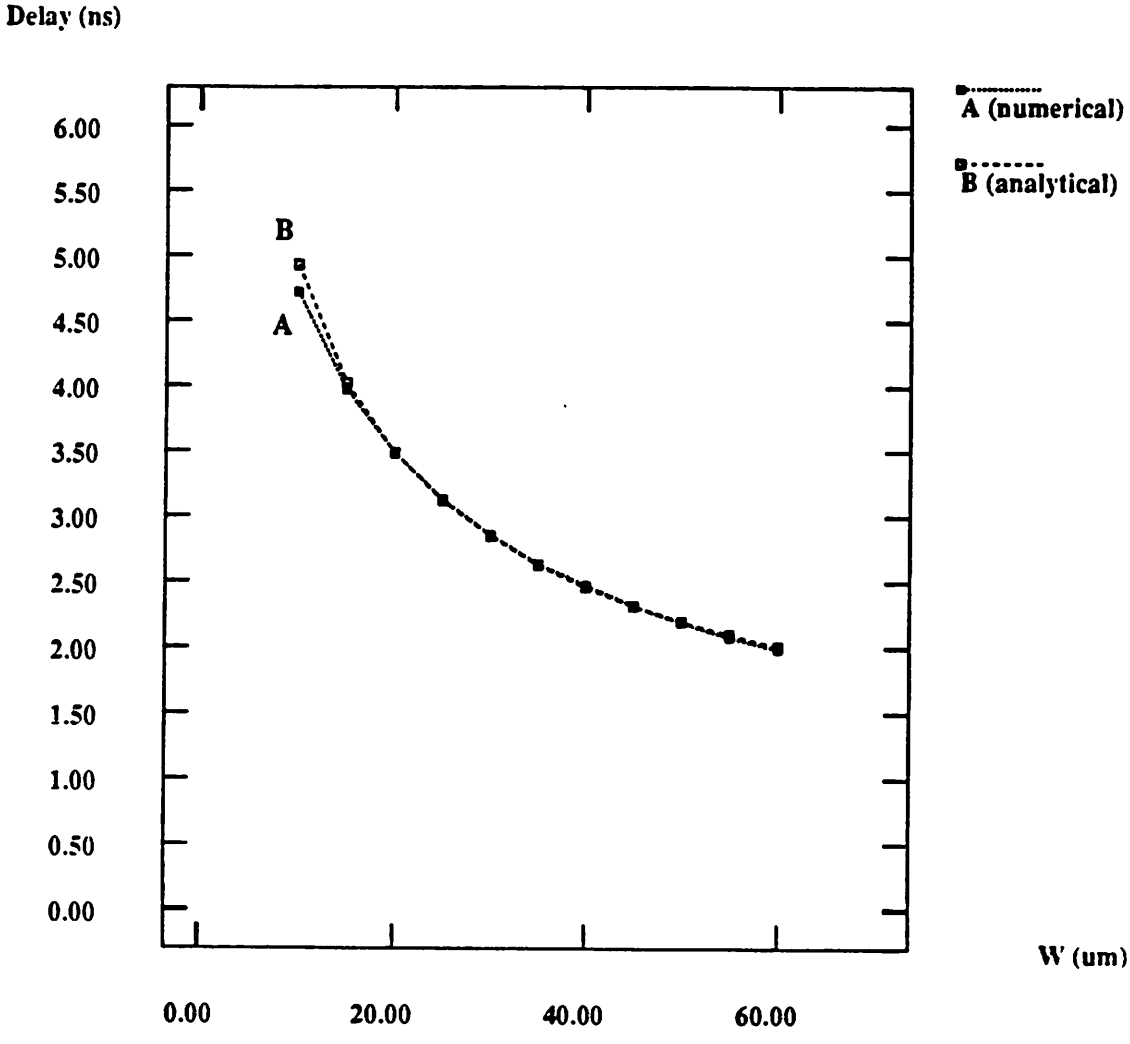


Figure 8.5: Simulated and approximated delay as a function of the width of the MOS transistor.

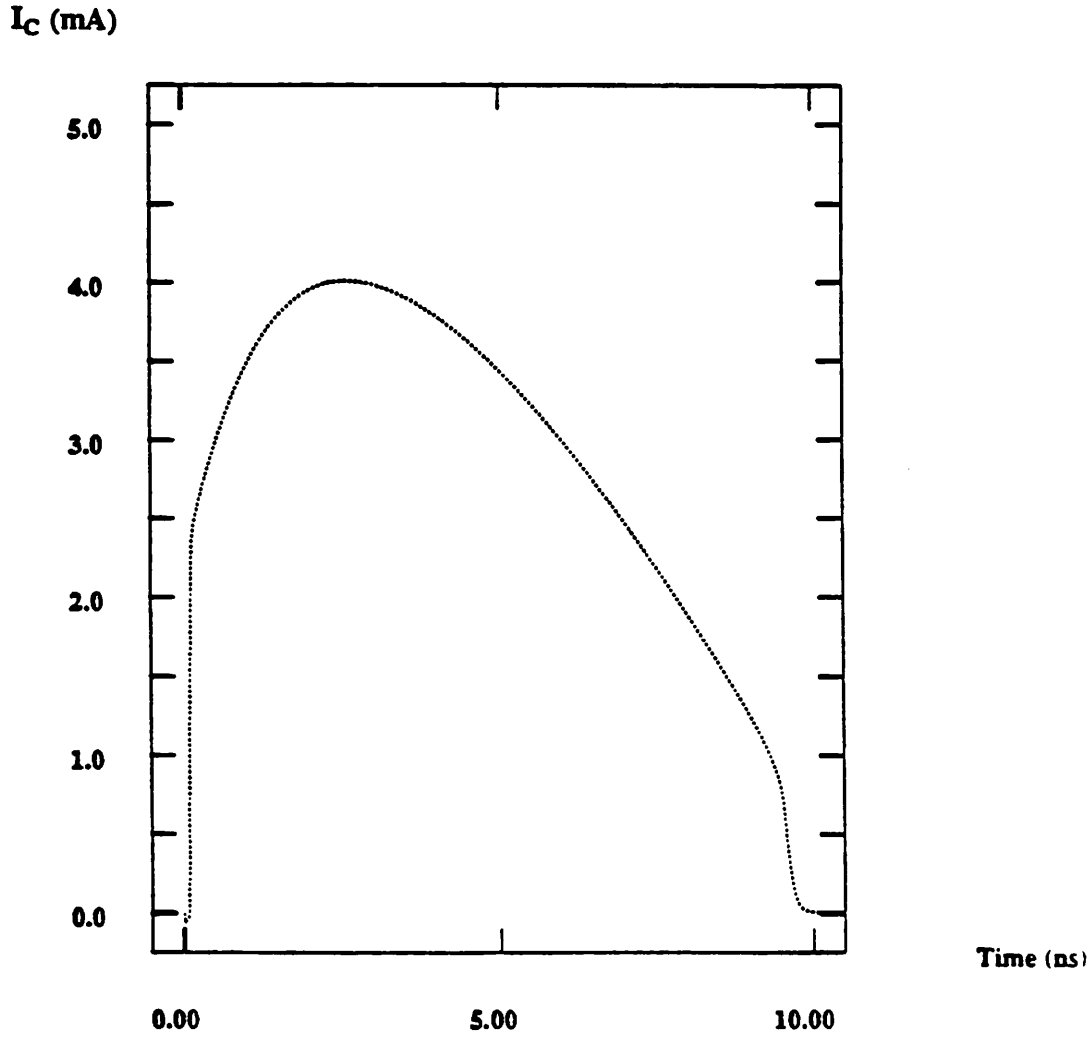


Figure 8.6: Simulated collector current for the pull-up transient.

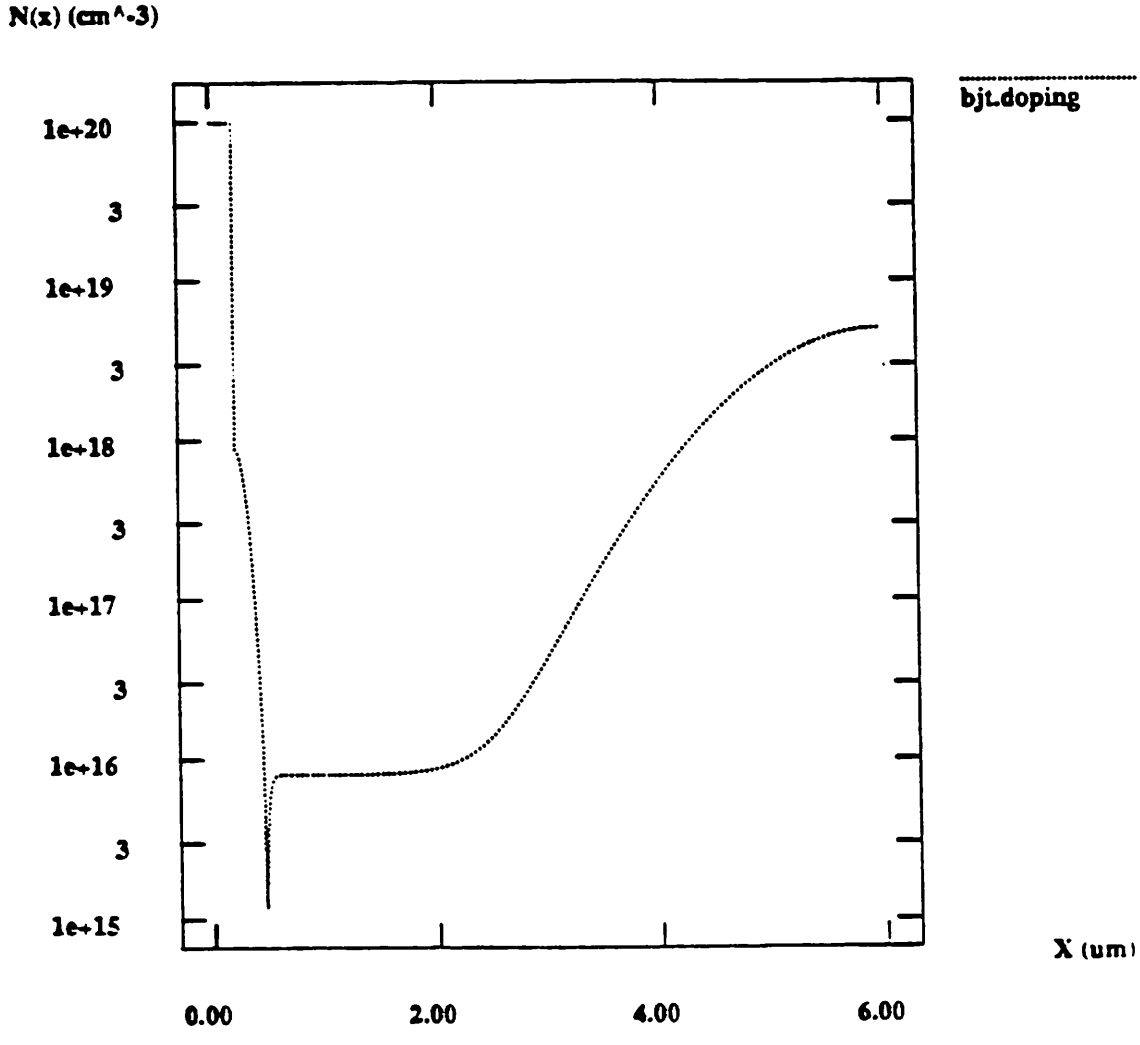


Figure 8.7: Bipolar transistor doping profile.

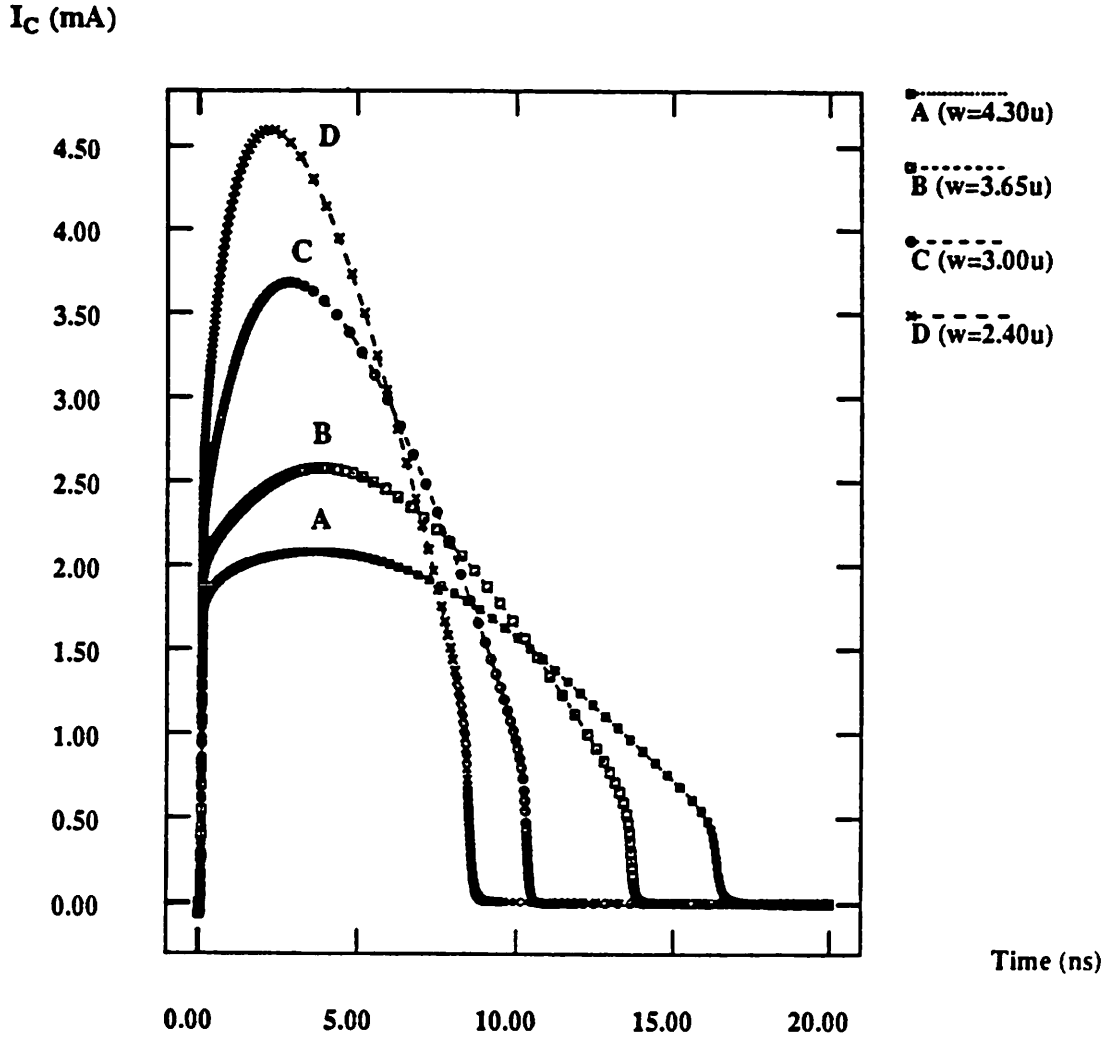


Figure 8.8: Simulated collector currents for the pull-up transient for different epi-layer thickness.

thickness is changed by varying the thickness of the buried layer. It is seen that a narrower epi layer results in a larger I_K, whereby a larger current is available to charge the output capacitance, resulting in a smaller delay.

Next consider the delay as a function of the epi-layer doping for the transistor with the doping profile of Figure 8.7. The simulated delay is plotted as a function of the load capacitance and the collector epi-layer doping, N_D, in Figure 8.9. Initially an increase in N_D results in a decrease in the delay. However, for large values of N_D the bipolar transistor no longer operates in high-level injection; hence, there is no improvement in the delay with an increase in the collector doping (the curves for N_D of $5\times10^{16}\ cm^{-3}$ and $1\times10^{17}\ cm^{-3}$ coincide in Figure 8.9). The breakdown of the base-collector junction places a limit on the doping used in the collector. An increase in the doping of the collector also increases the base-collector capacitance. However, this effect is less important since the delay reduces due to an improved high-level-injection behavior of the bipolar transistors.

The influence of the supply-voltage scaling on the delay of the gate has been examined in [8.9], and the main conclusions are: for large supply voltages the delay is a weak function of the supply voltage. The delay increases as the supply voltage is reduced for both the pull-up and pull-down transients. However, the pull-down delay increases much more rapidly than the pull-up delay at lower voltages since the $V_{BE(on)}$ drop of the bipolar transistor Q2, in Figure 8.1, becomes important.

8.3. P-i-n Diode Turn Off

The transient behavior of power devices such as rectifiers, transistors, and thyristors in an RLC circuit is an interesting problem. However, adequate analytical models which can accurately describe the complete turn-off process do not exist. Numerical device-level simulations are a useful alternative if the device can be simulated with

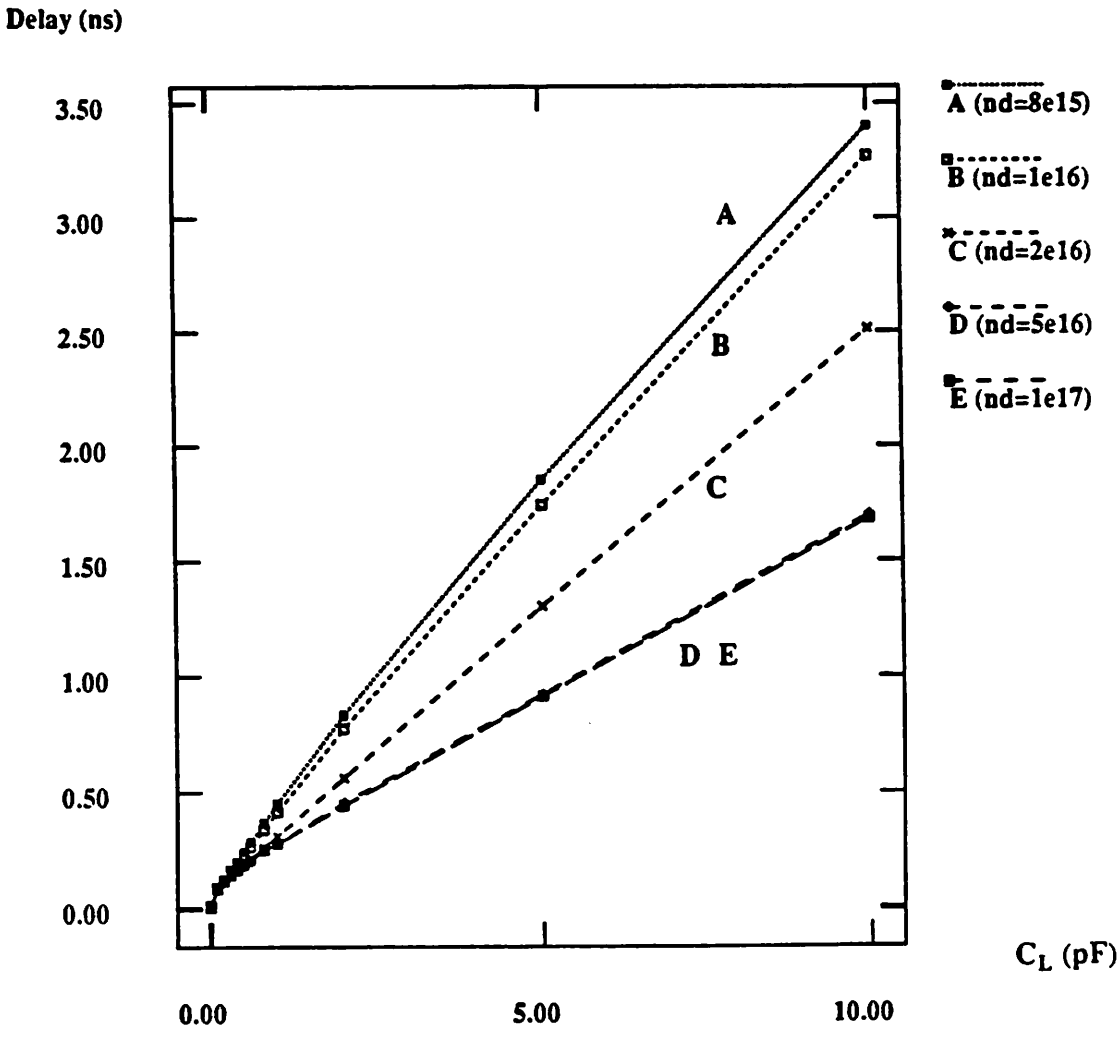


Figure 8.9: Delay as a function of the load capacitance and epi-layer doping.

circuit elements attached to its terminals. Most device simulators [8.10, 8.11] only allow voltage or current boundary conditions (some also allow parasitic resistive and capacitive elements [8.11]), and are not useful for such simulations. A coupled circuit and device simulator provides a flexible environment to study the interactions between devices and the circuits in which they are embedded.

8.3.1. Reverse Recovery of p-i-n Diodes

The reverse recovery of p-i-n rectifiers is an important phenomena in power semiconductor circuits not only because it impacts circuit speed, but also because it affects power loss and inductive voltage-spike generation. The problem is difficult and analytical results are available only under special conditions. The reverse recovery with a resistive load has been analyzed in [8.12]. For practical situations the recovery with a current ramp is of interest. This problem has been analyzed for part of the recovery in [8.13], but a solution is not available for the complete turn-off process. No analytical model is able to predict the peak reverse voltage that appears across the rectifier during turn off. The recovery process is complicated since there is a strong interaction of the device dynamics with the circuit. Device simulation with inductive boundary conditions is one way of analyzing the turn off under special conditions and has been used in [8.14, 8.15].

Consider the simplified circuit of Figure 8.10(a) which is representative of a typical turn-off situation. The voltage steps from a positive to a negative value when the diode has to be turned off; the inductor in series models the inductance present in the circuit. For this example, the p^+nn^+ diode is modeled as a one-dimensional device with the uniform doping profile shown in Figure 8.10(b).

The simulated current and voltage waveforms for a dI/dt of 20 A/μs are shown in Figure 8.11. The particular characteristics of these curves can be explained by physical arguments; however, an estimation of the magnitude of the voltage spike is not possible.

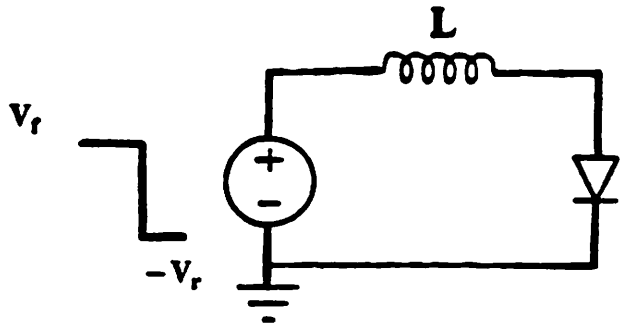


Figure 8.10(a): A simple circuit used of simulating the reverse recovery of p-i-n diodes.

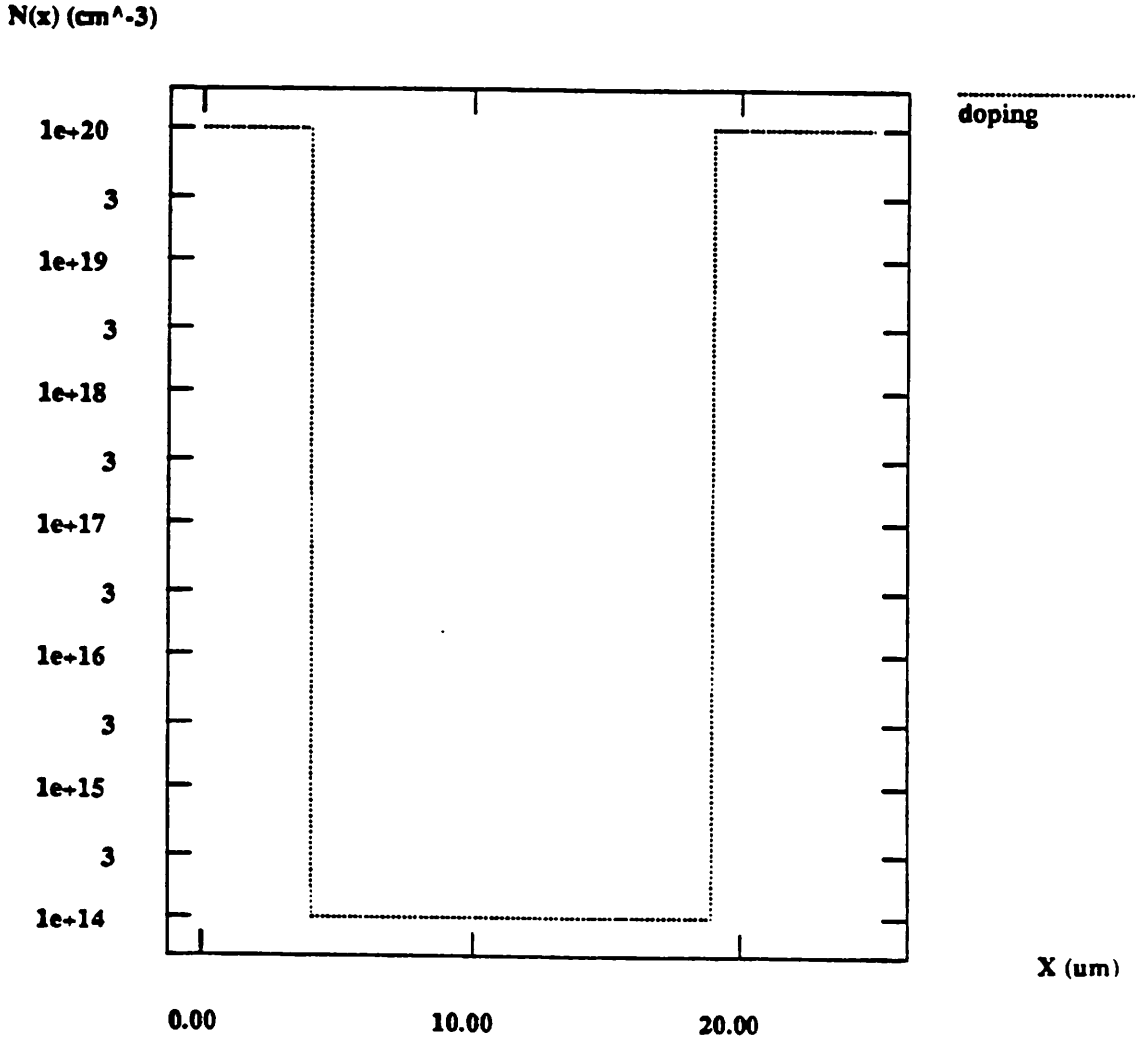


Figure 8.10(b): Diode doping profile. The *n*-region is 15μm wide.

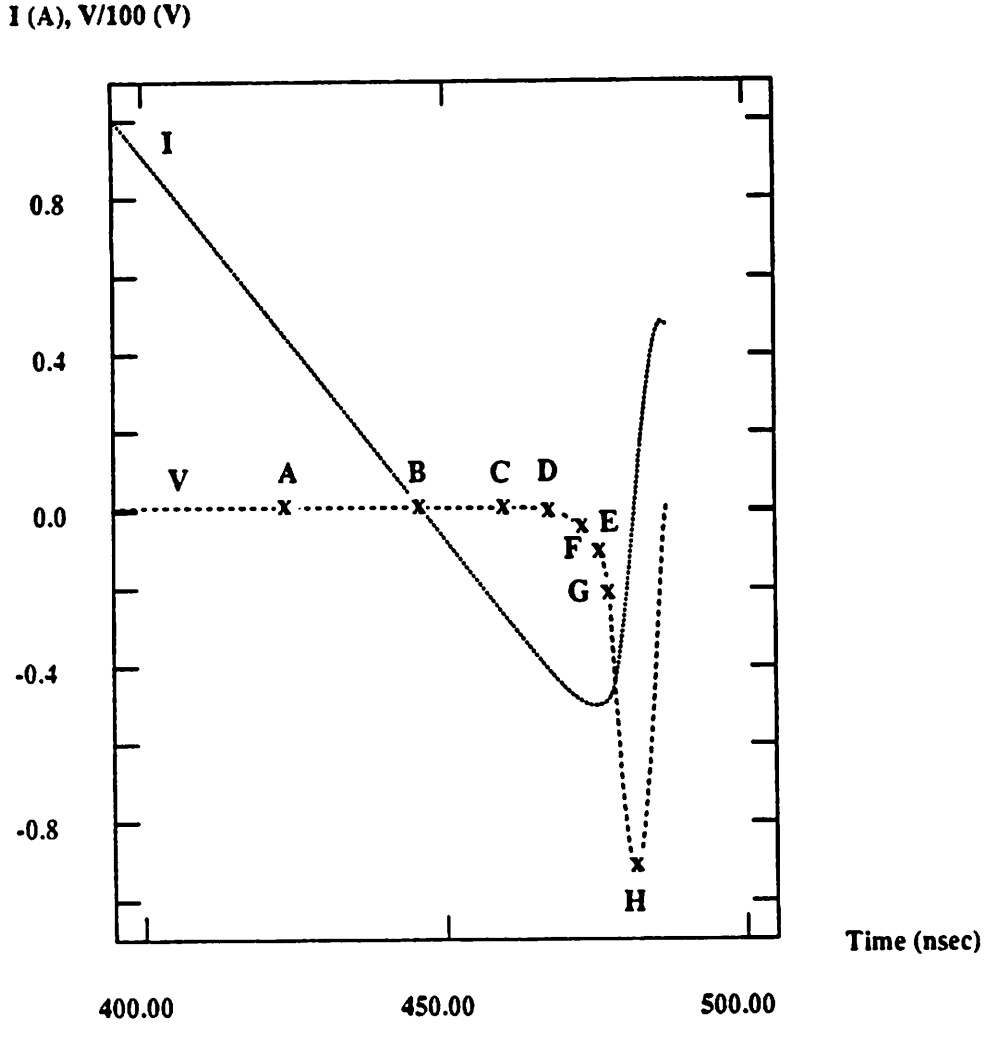


Figure 8.11: Simulated current and voltage waveforms during reverse recovery.

The carrier concentrations and electric fields corresponding to the time instants *A-H* in Figure 8.11, are shown in Figures 8.12(a) and 8.12(b), respectively.

Initially, before the source voltage switches, the circuit is in a steady state with a forward current I_F flowing through the diode. The *n*-region is conductivity modulated because the diode is operating in high-level injection. When the source voltage switches, the current through the diode decreases at a rate given by

$$\frac{dI}{dt} = \frac{-V_R - V_D}{L} \approx -\frac{V_R}{L} \tag{8.7}$$

when $V_R \gg V_D$, the diode voltage. At a time T_0 (corresponding to time instant B in Figure 8.11),

$$T_0 = \frac{I_F}{\left[\frac{dI}{dt}\right]} \tag{8.8}$$

the current reverses its direction and the diode continues to conduct because of the charge stored in the *n*-region. This charge is removed by the reverse current and due to recombination within the *n*-region. The diode voltage becomes negative only after a depletion region starts forming at the p^+n junction and the diode starts blocking the voltage. As the depletion region increases the diode blocks a larger voltage. At a time instant T_1 (corresponding to time instant E in Figure 8.11) the diode voltage equals the reverse voltage of the source and the current reaches its peak reverse value, i.e.,

$$V_D(T_1) = -V_R$$

$$\frac{dI}{dt}\Big|_{T_1} = 0 \tag{8.9}$$

It should be noted that at T_1 a large portion of the 15μm *n*-region is depleted. The analytical model of [8.13] assumes that the width of the depletion region is negligible which is certainly not true for this example.

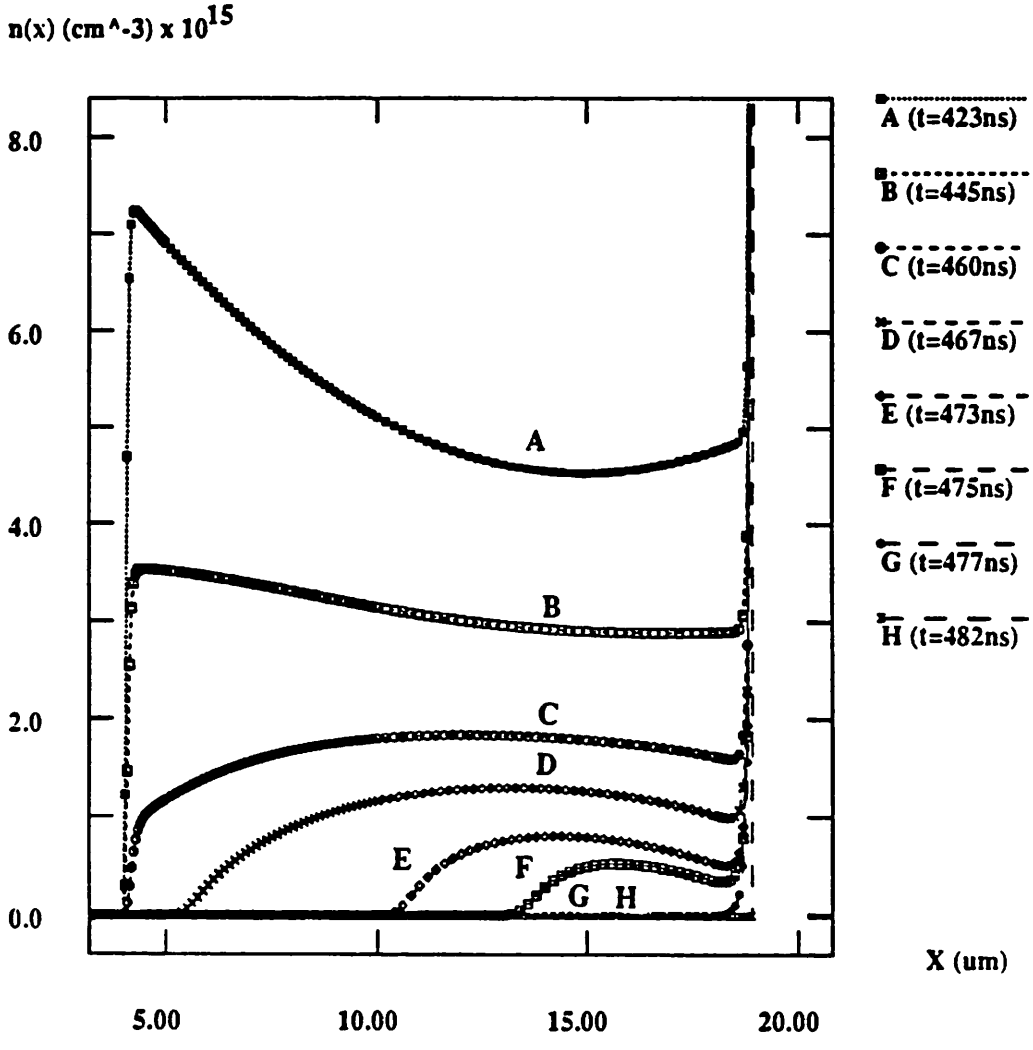


Figure 8.12(a): Electron concentrations as a function of time during the reverse recovery.

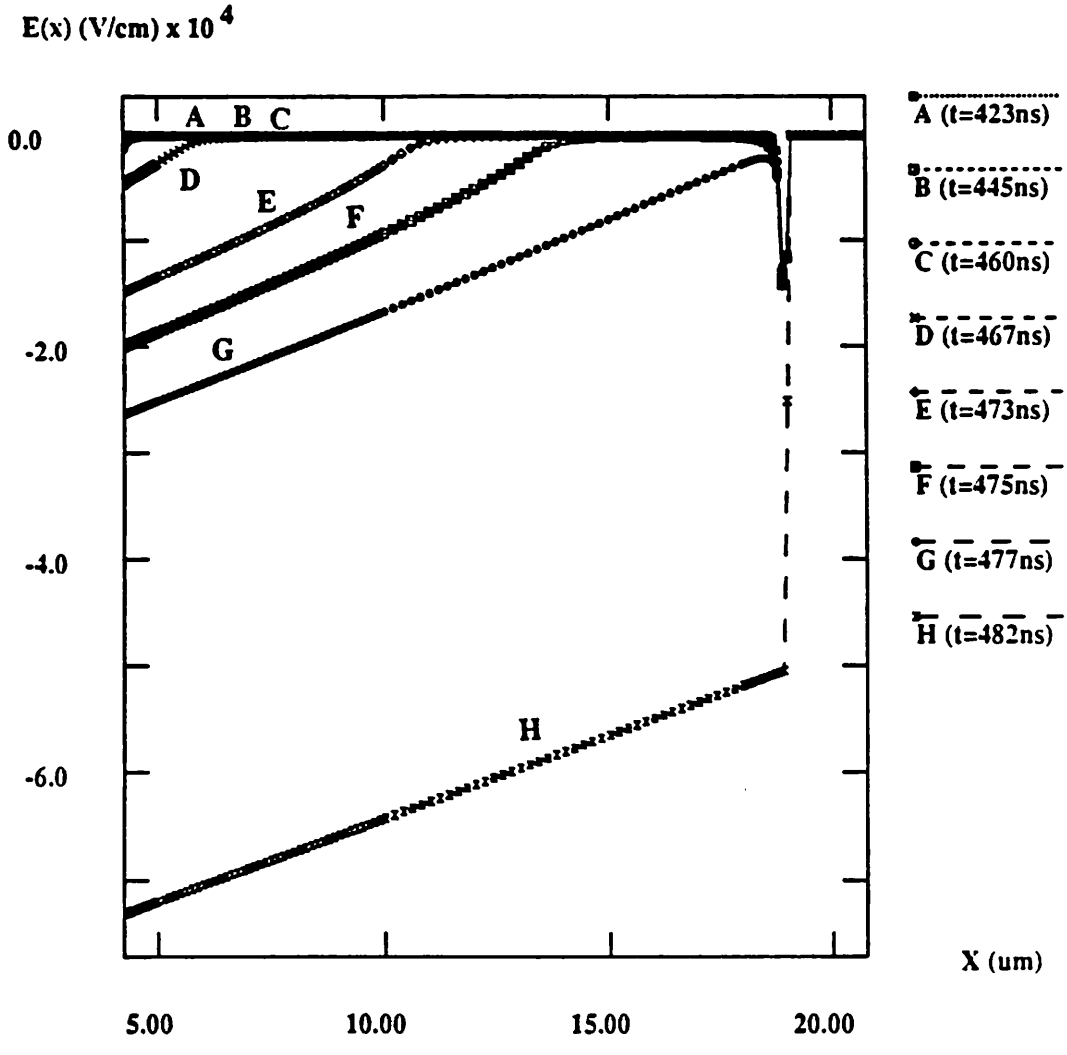


Figure 8.12(b): Electric field as a function of time during the reverse recovery.

From time T_1 to T_2 (corresponding to the time instant H in Figure 8.11) the current through the diode decreases but is still negative in value, and $dI/dt > 0$. As a result a voltage spike appears across the diode. The diode voltage is related to dI/dt by

$$V_D = -V_R - L\frac{dI}{dt} \tag{8.10}$$

This region of operation is hard to analyze since dI/dt and, hence, V_D are determined by the amount of stored charge remaining within the device. The stored charge in turn depends upon both the circuit and device parameters and its estimation is difficult for a general situation. As a consequence the peak reverse voltage cannot be determined.

At a time T_2 the diode current becomes zero, and may go positive because of oscillations resulting from the LC circuit formed by the inductance and the depletion-region capacitance of the diode if sufficient damping is not present in the circuit.

The current waveform during the reverse recovery is idealized to be triangular in nature as shown in Figure 8.13. T_A and T_B, indicated in Figure 8.13, are important parameters for the reverse recovery. The sum of T_A and T_B is the total time for the reverse recovery and is denoted as T_{RR}. The ratio $\frac{T_B}{T_A}$ is called the *softness* factor S. A large value of S results in a softer recovery and hence a small peak reverse voltage across the diode. In general, a good rectifier will have a large value of S which ensures that dI/dt is not excessively large during the T_B portion of the reverse recovery. Although the various time instants can be identified on the reverse recovery waveforms, these cannot be calculated analytically. Hence, a mixed-level circuit and device simulator is essential for estimating the parameters of interest.

The reverse-recovery waveform of the rectifier can also be used to determine the high-level-injection carrier lifetimes. A charge-control analysis can be used as in [8.16, 8.17]. In [8.16] the device is assumed to be extremely "snappy" and therefore the

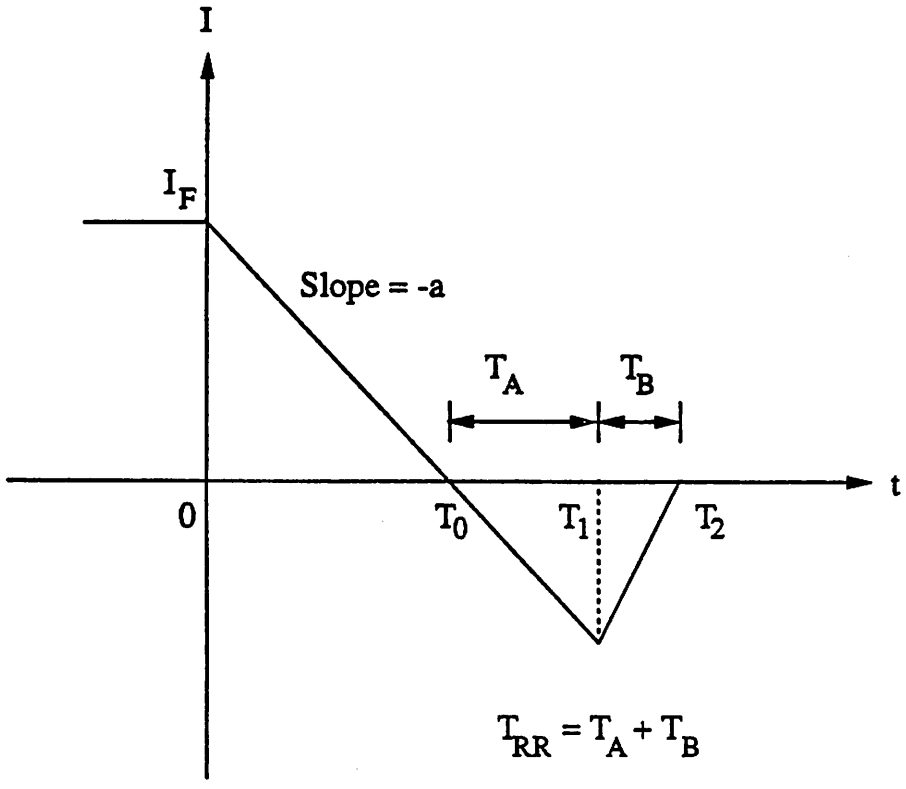


Figure 8.13: An idealized reverse recovery waveform showing T_A, T_B, and T_{RR}.

analysis is not valid for devices with a softer recovery. The analysis of [8.17] is more general; a triangular current waveform is assumed which is representative of a wider range of conditions. The procedure for lifetime extraction is verified by simulations from CODECS and experimental data in [8.18].

Although the charge-control analysis provides a good estimate for the carrier lifetimes from measured characteristics, it cannot be used to provide an a priori estimate of the softness factor or the magnitude of the voltage spike.

The factors that affect S can be understood by studying the dependence of T_A and T_B on W, the width of the n-region, and τ, the high-level-injection carrier lifetime. In Figure 8.14(a) simulated T_A is plotted as a function of τ, with W as a parameter, and Figure 8.14(b) gives the dependence on W with τ as a parameter. The corresponding dependencies of T_B are shown in Figures 8.15(a) and 8.15(b), respectively. It can be seen from Figure 8.14(a) that T_A can be fit as $T_A = k(W)\tau$, where $k(W) \leq 1$ is a weak function of W. The dependence of T_B on W and τ is hard to model analytically and, therefore, simulations are necessary for determining S. The simulated values of S are shown in Figures 8.16(a) and 8.16(b).

CODECS can also be used to determine the magnitude of the inductive voltage spike that appears across the rectifier during the turn off. Simulations have been performed for different values of L, and a fixed value of V_R=10V, for two different values of high-level-injection lifetimes. S and the peak reverse voltage, V_P, as a function of L are given in Table 8.1.

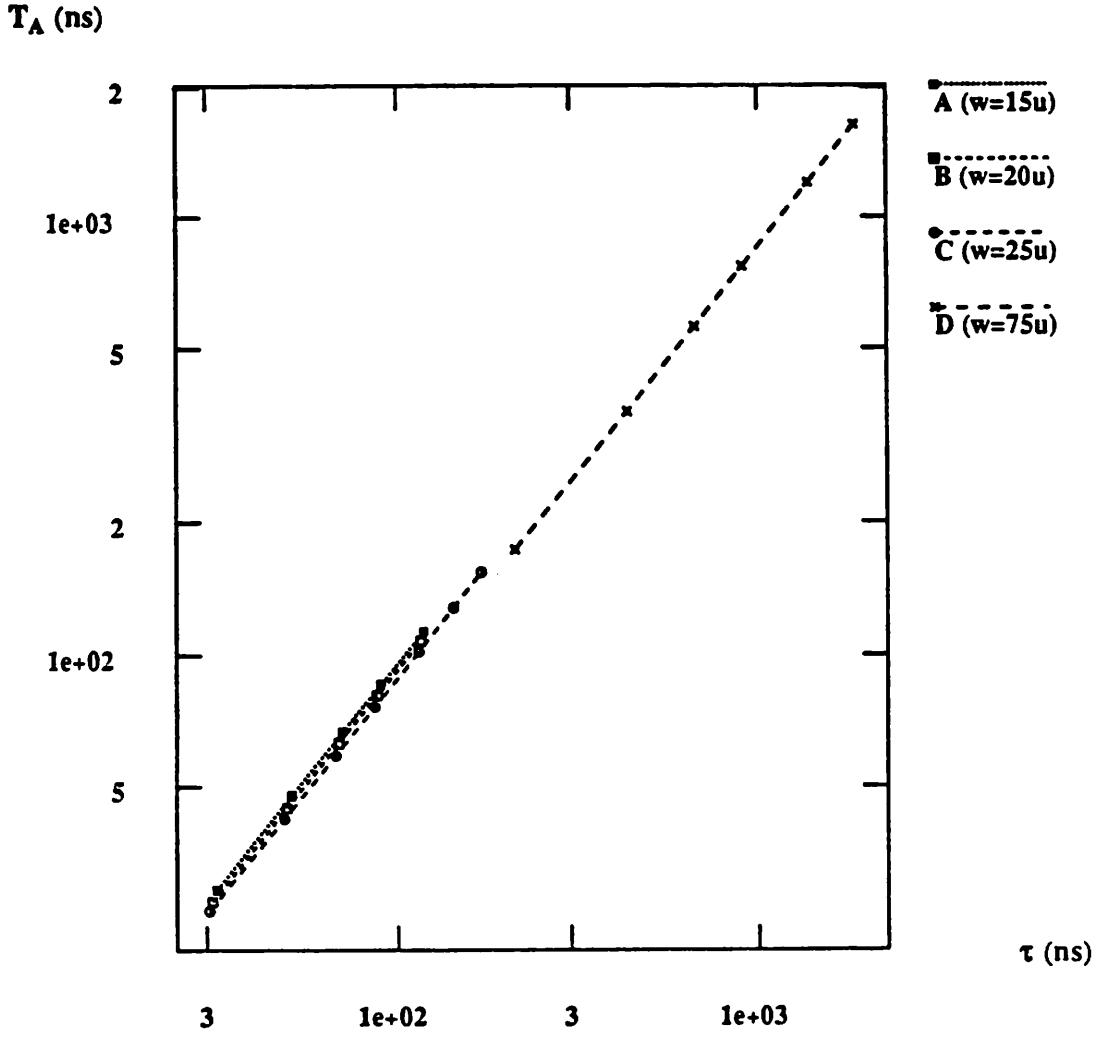


Figure 8.14(a): Simulated value of T_A as a function of τ for different values of W, the width of the n-region.

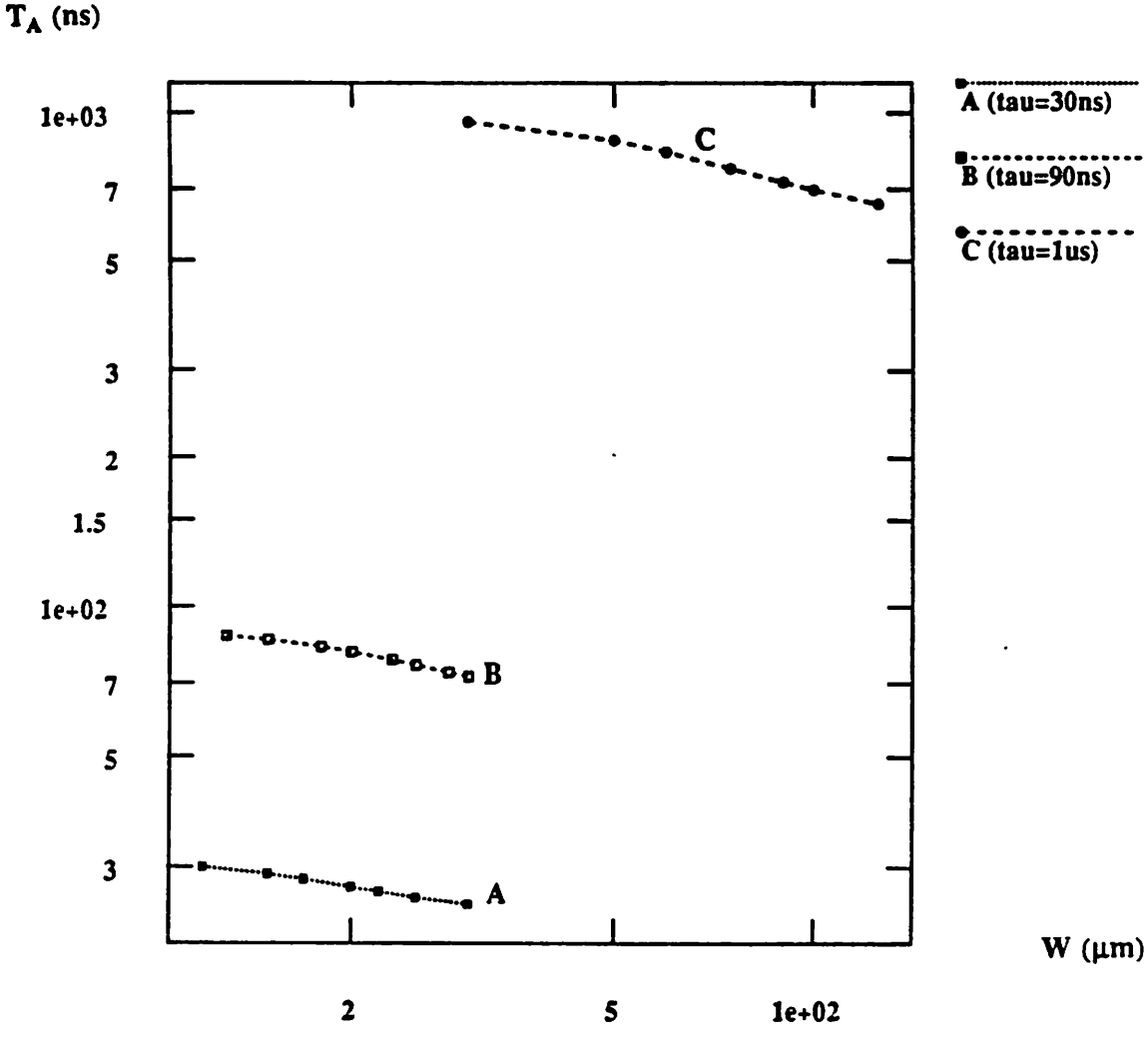


Figure 8.14(b): Simulated value of T_A as a function of W for different values of τ.

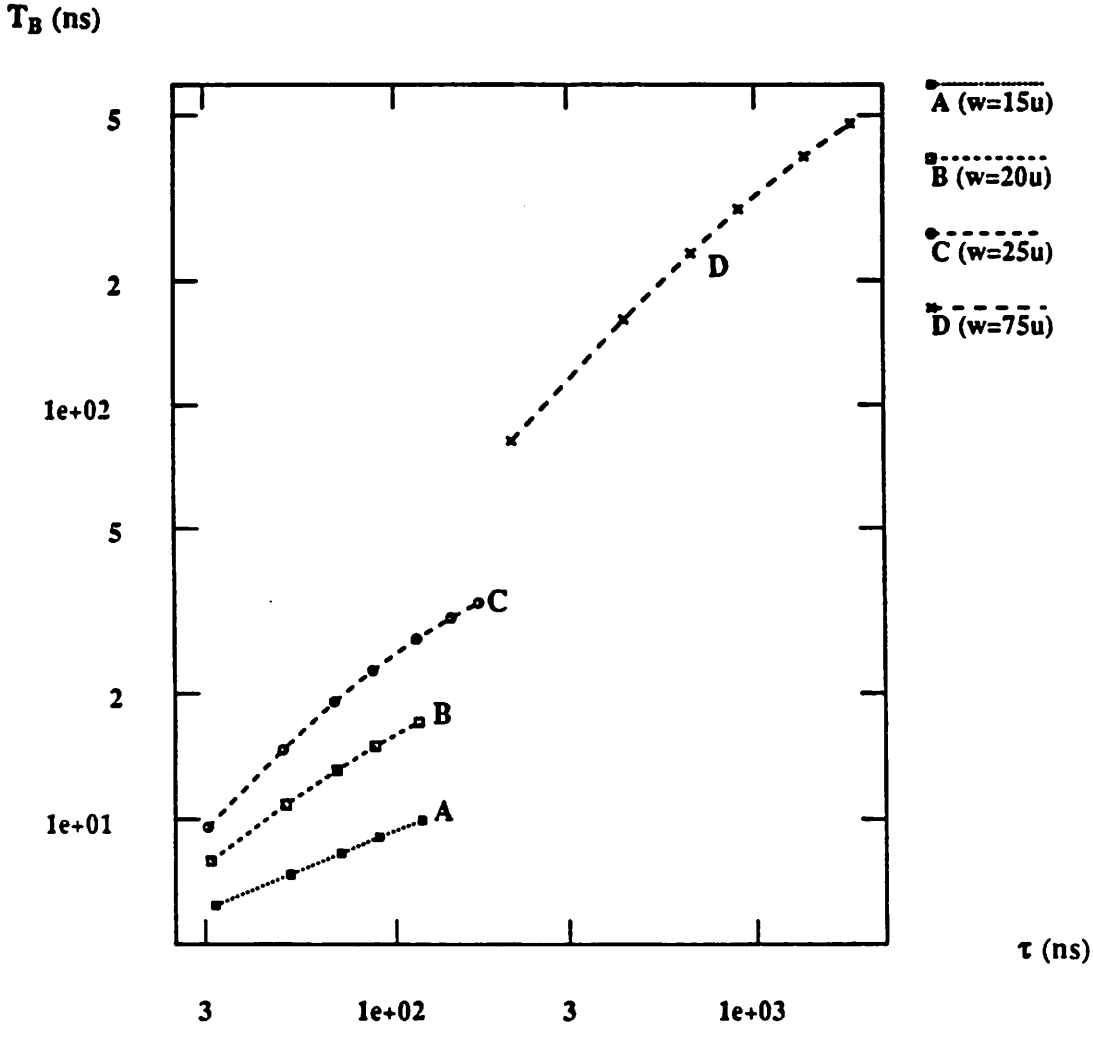


Figure 8.15(a): Simulated value of T_B as a function of τ for different values of W.

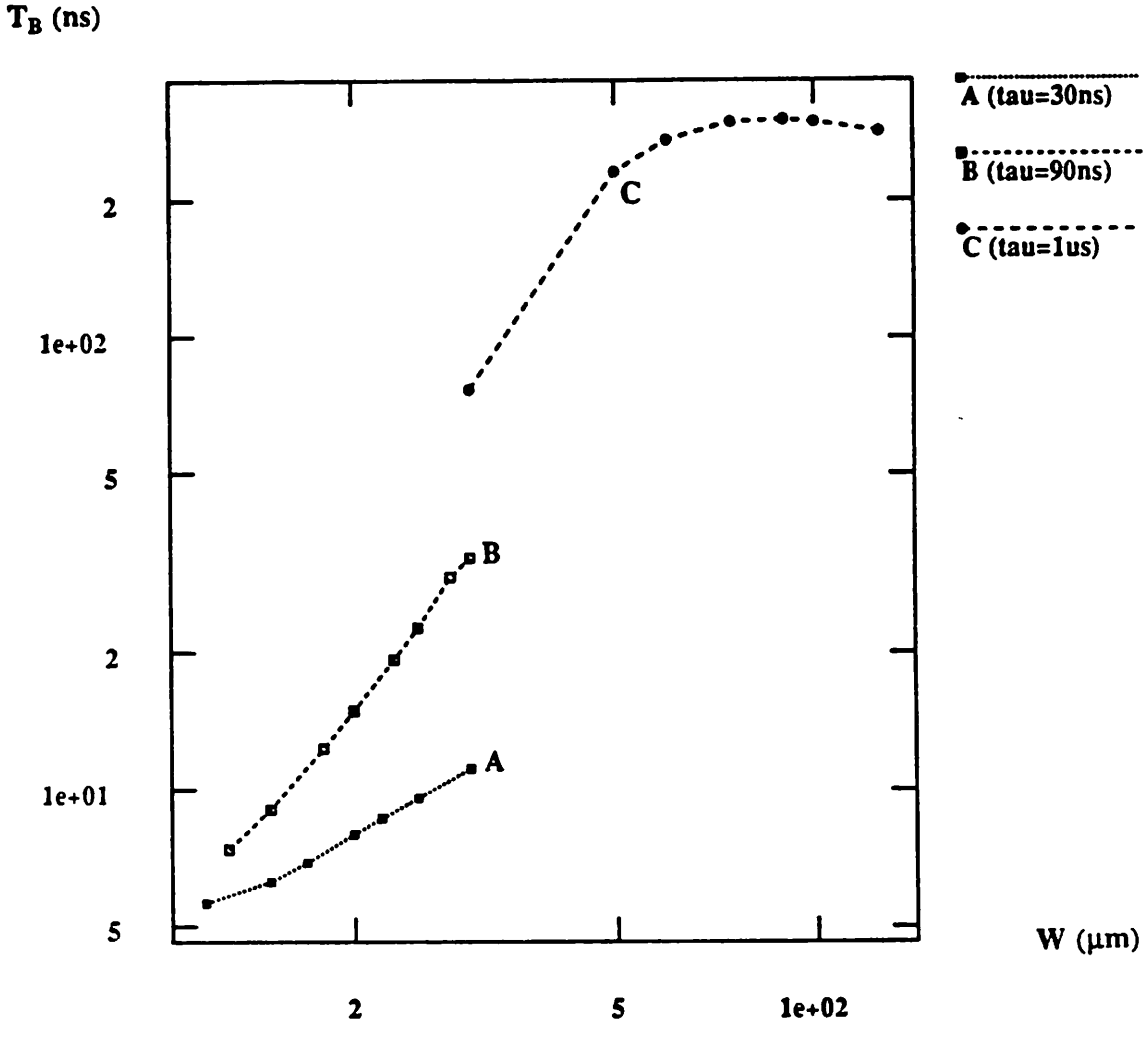


Figure 8.15(b): Simulated value of T_B as a function of W for different values of τ.

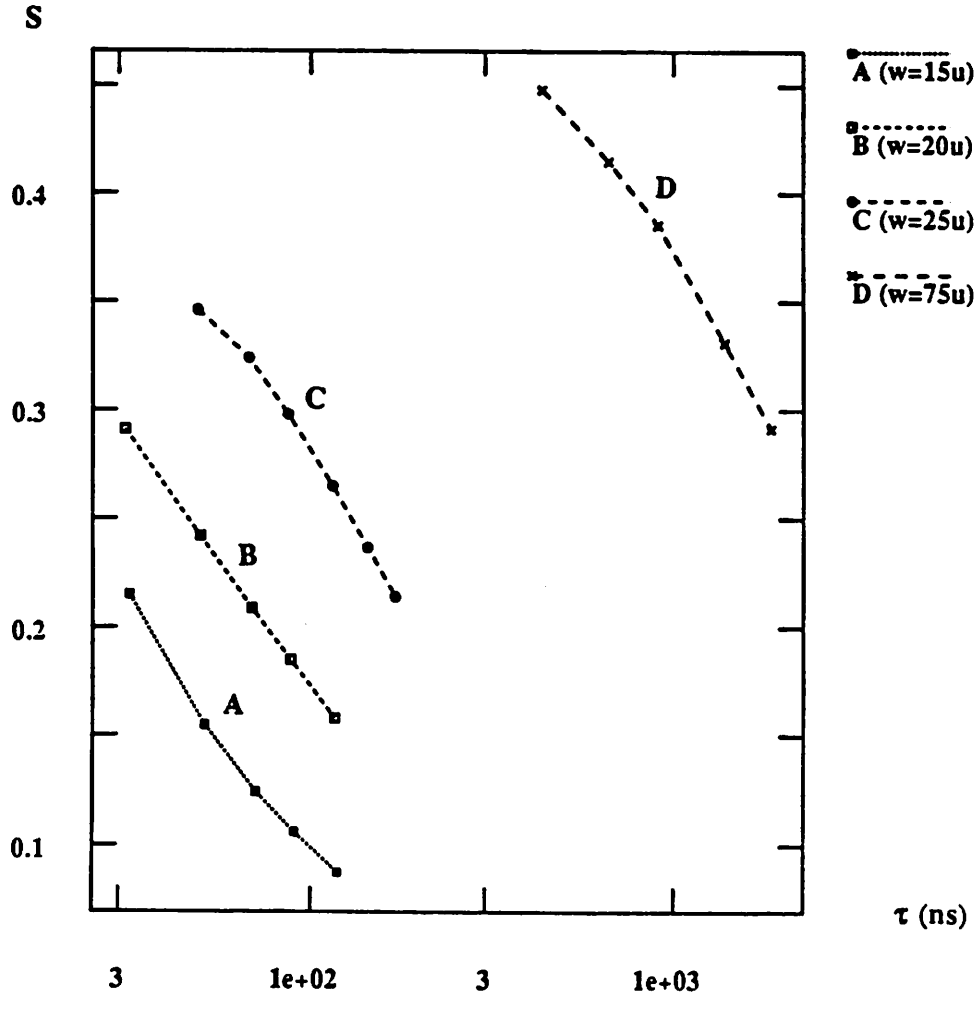


Figure 8.16(a): Simulated value of S as a function of τ for different values of W.

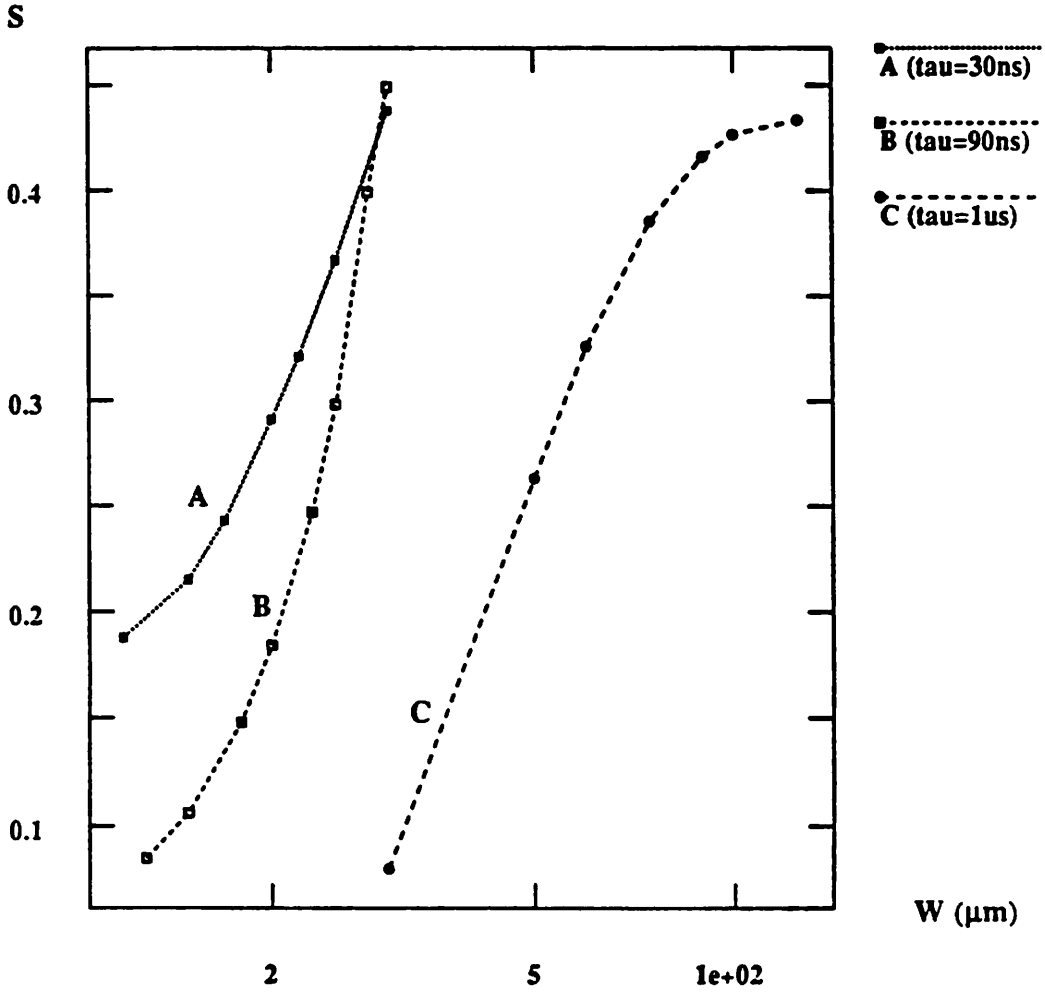


Figure 8.16(b): Simulated value of S as a function of W for different values of τ.

L (μH)	τ = 30ns		τ = 60ns	
	V_P (V)	S	V_P (V)	S
0.25	134	0.196	252	0.138
0.4	111	0.206	210	0.136
0.5	101	0.215	190	0.137
0.6	94	0.224	178	0.140
0.8	84	0.242	160	0.146
1.0	77	0.257	145	0.153

Table 8.1: Simulated V_P and S as a function of L

S is seen to have a weak dependence on L. A smaller L results in a proportionally larger dI/dt during T_A, and even larger dI/dt during T_B due to the decreasing S. Hence, larger reverse-voltage spikes are obtained for smaller inductances.

8.3.2. Operation Under Breakdown Conditions

Consider the situation of an unclamped reverse recovery with an inductive load such that the diode operates near its breakdown voltage. The diode doping profile and dimensions are shown in Figure 8.17(a), and the reverse current-voltage characteristics are shown in Figure 8.17(b).

The current and voltage waveforms for the reverse recovery are shown in Figure 8.18(a). For these conditions the diode voltage exceeds the breakdown voltage and oscillations are observed during the second phase of the turn off (during T_B); the oscillations are shown in greater detail in Figure 8.18(b). Experimental data that subsequently verified the existence of such oscillations in real devices is shown in Figure 8.19, and provides evidence that oscillations are possible.

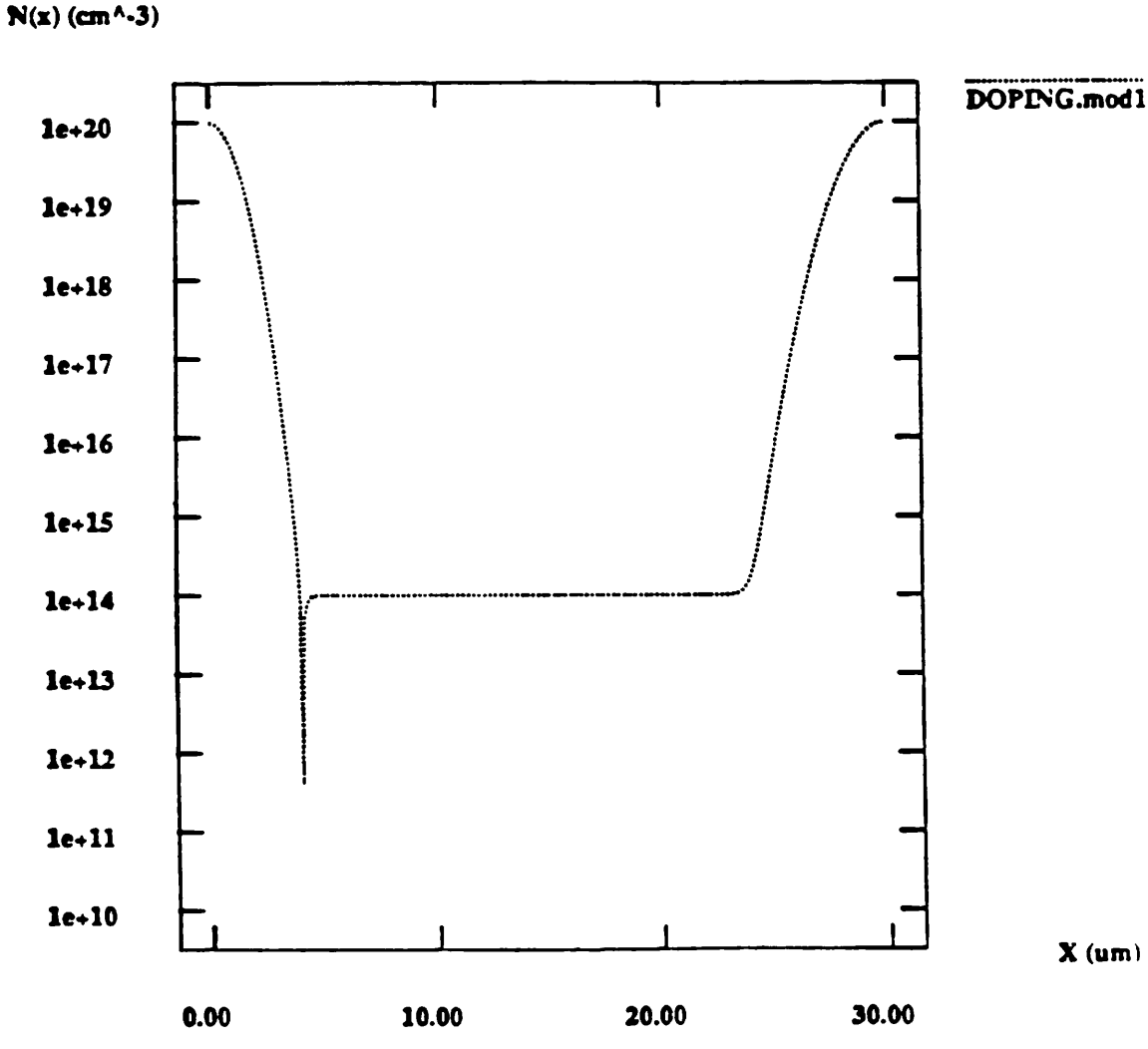


Figure 8.17(a): p^+–n–n^+ diode doping profile.

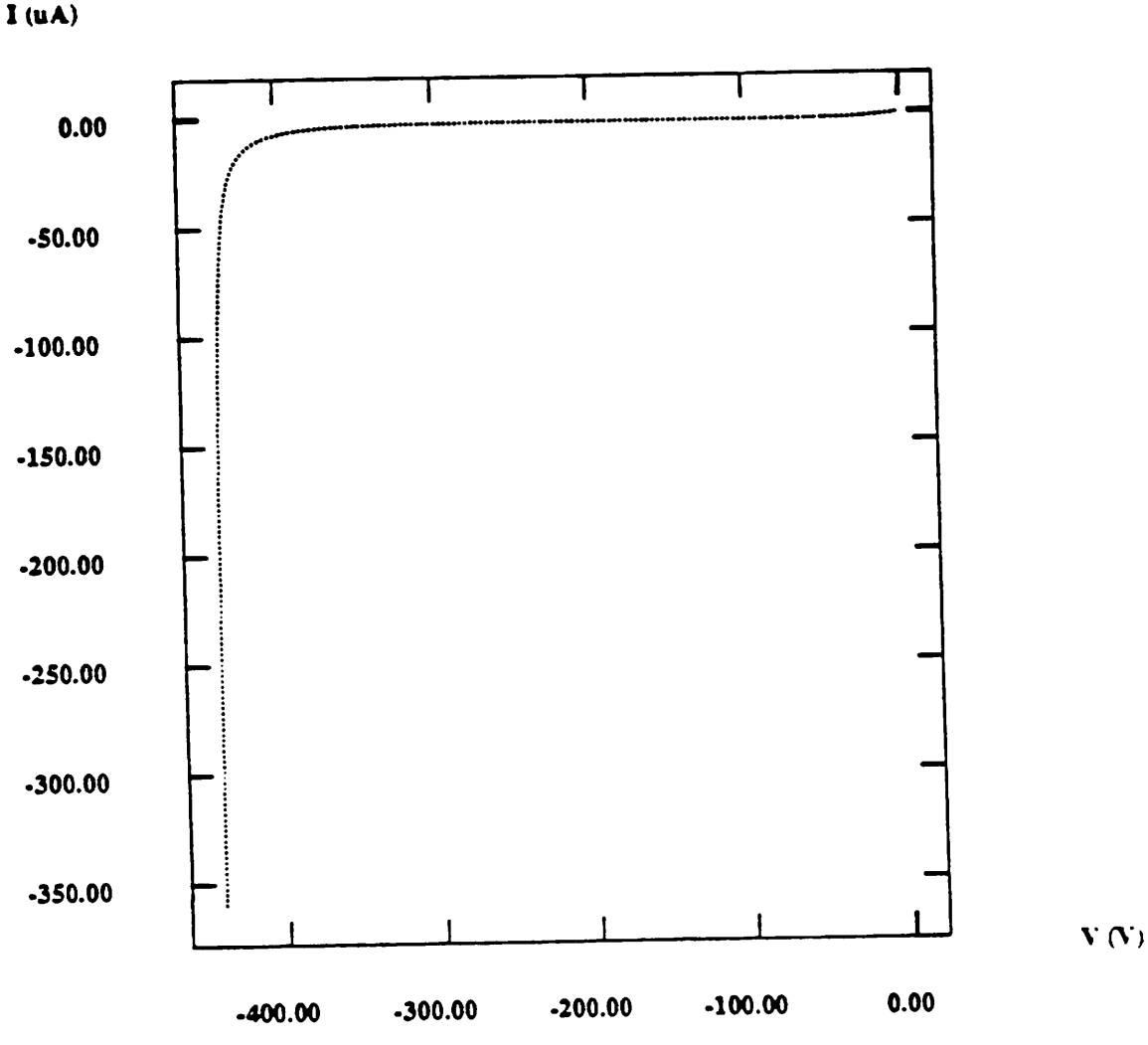


Figure 8.17(b): Simulated reverse characteristics. The breakdown voltage is 425V.

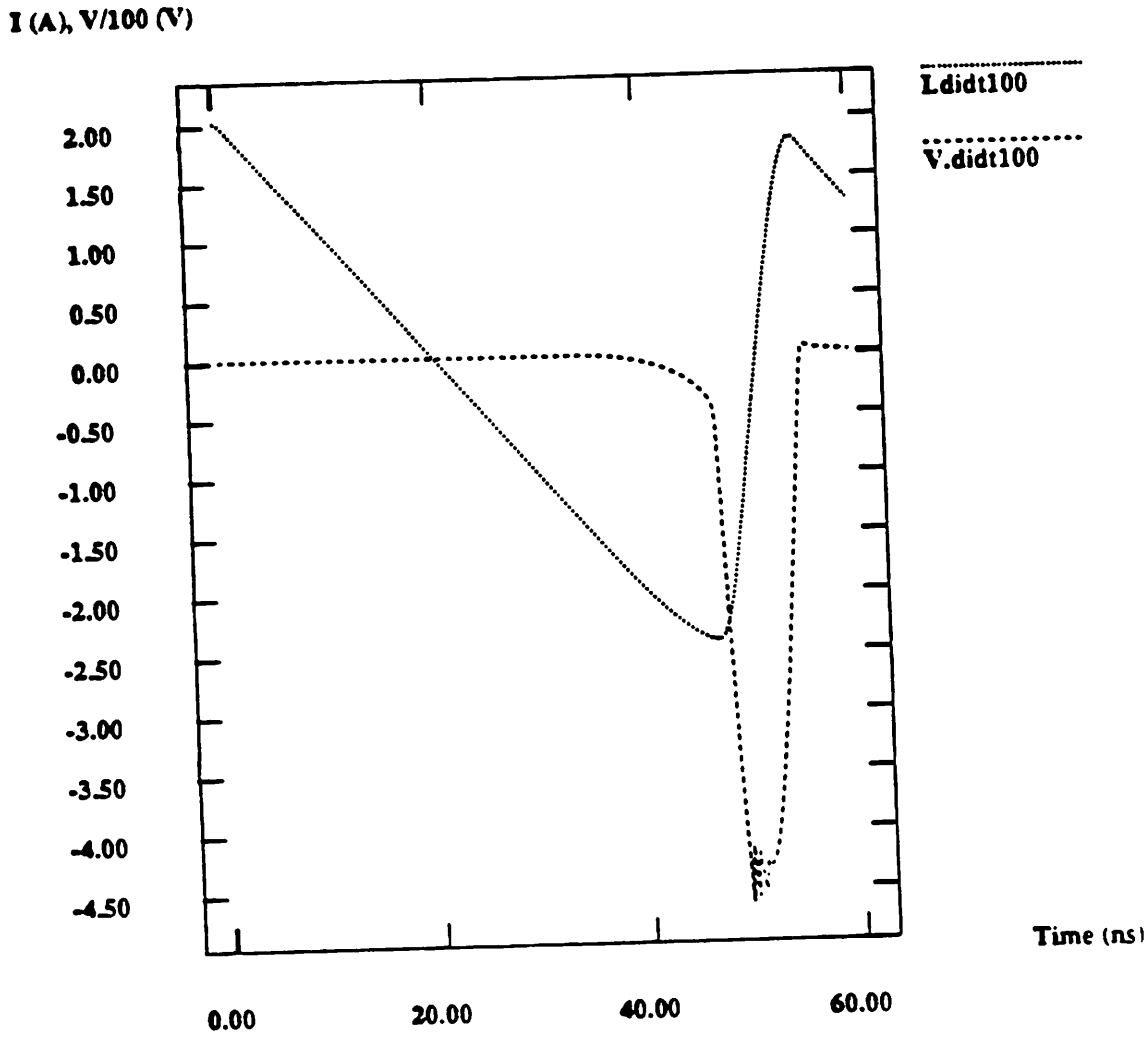


Figure 8.18(a): Simulated diode current and voltage as a function of time for inductive turn off with dI/dt = 100A/μsec.

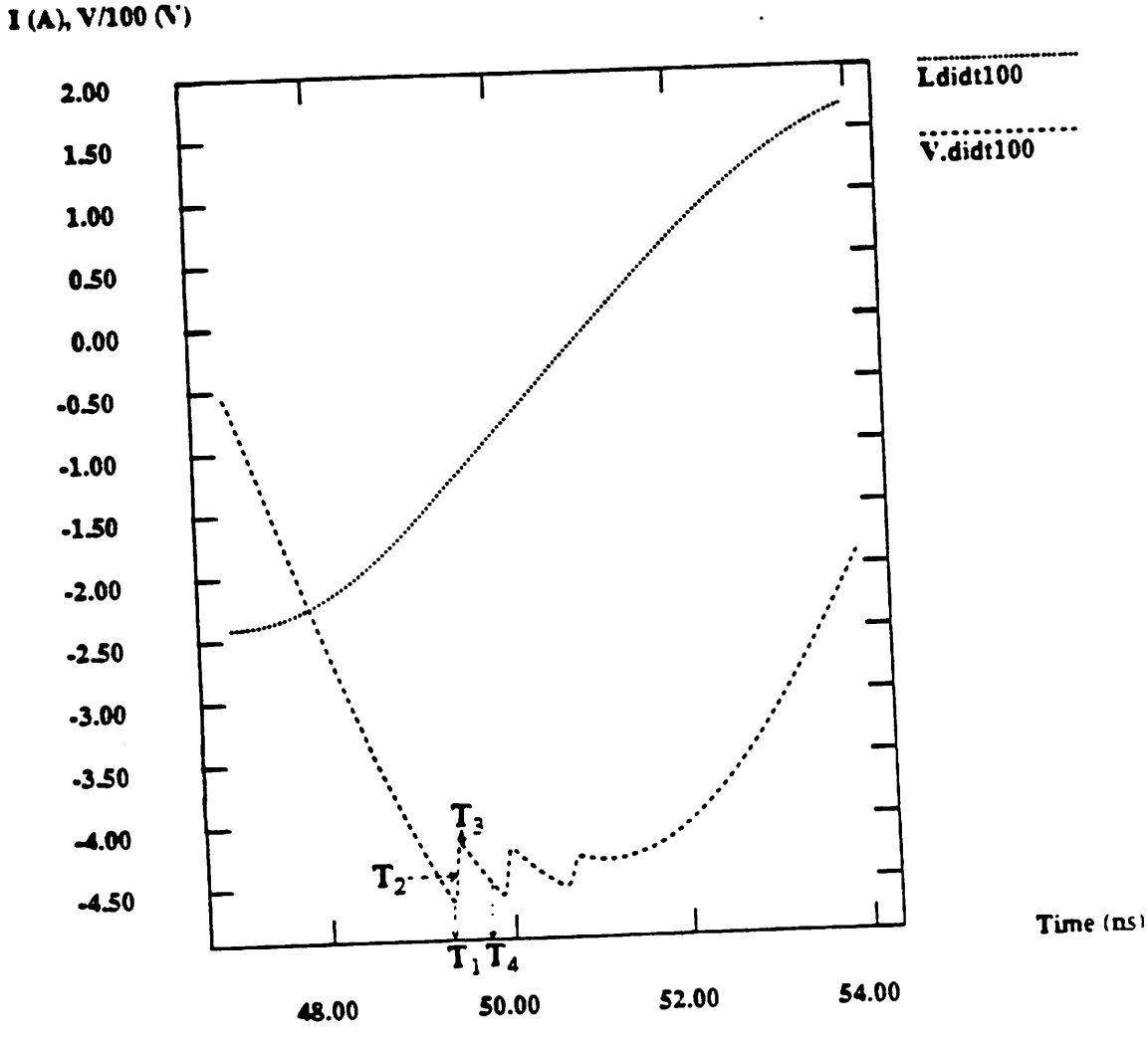
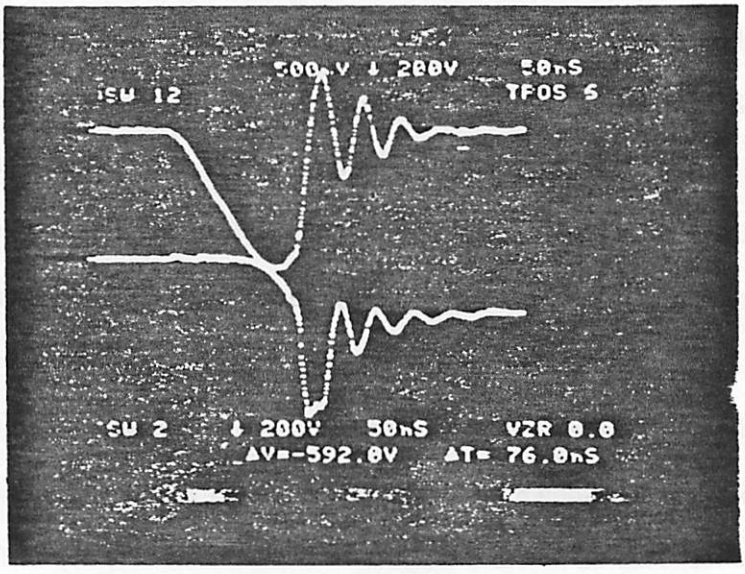


Figure 8.18(b): Simulated diode current and voltage as a function of time for inductive turn off with dI/dt = 100A/μsec shown on an expanded scale. Oscillations are observed in the diode voltage during the reverse recovery. Four different time instants are marked.

These oscillations might appear surprising at first but can be explained by means of a simple physical model for the recovery process. An equivalent circuit for the diode during the reverse recovery is shown in Figure 8.20. The inductor and voltage source are replaced by a current source I_S that provides a ramping current waveform as shown; $I_{gen}(V)$ is a nonlinear voltage-dependent current source. This equivalent circuit is valid only after the n-region of the rectifier is depleted, i.e., when a large V_d appears across the diode or $t > T_A$. C_d is simply the depletion-region capacitance. The current source $I_{gen}(V)$ models the current flow due to avalanche carrier generation. When the reverse voltage is less than the breakdown voltage, $I_{gen}(V)$ is zero and is nonzero only under breakdown conditions. This current is a conduction current since it is due to free carriers generated by the avalanche multiplication process. The difference between I_S and I_{gen} is responsible for charging and discharging C_d and determines the capacitor voltage $V_d(t)$.

For the initial part of the turn off, $t < T_1$ in Figure 8.18(b), the current I_{gen} is zero and the current I_S charges the capacitor, whereby the reverse voltage across the diode continues to increase. The simulated conduction and displacement currents for the diode are shown in Figures 8.21(a) and 8.21(b), respectively, for four different time instants, T_1, T_2, T_3, and T_4. These time instants are identical to those marked in Figure 8.18(b). It is seen that at T_1, the conduction current is zero and the diode current is a pure displacement current. When the voltage exceeds the breakdown voltage, I_{gen} increases rapidly due to the large number of carriers generated by the avalanche multiplication process, e.g., at time instants T_2, and T_3. The current could be large and discharges the capacitor, leading to a sudden decrease in the diode voltage. This is confirmed by examining the current components at T_2 and T_3 in Figure 8.21(b). After breakdown occurs, the conduction current increases and the displacement current becomes a large positive value. During this interval the capacitor is discharged. The sudden discharging of the capacitor causes the diode voltage $|V_d|$ to drop. Once the diode voltage goes below the



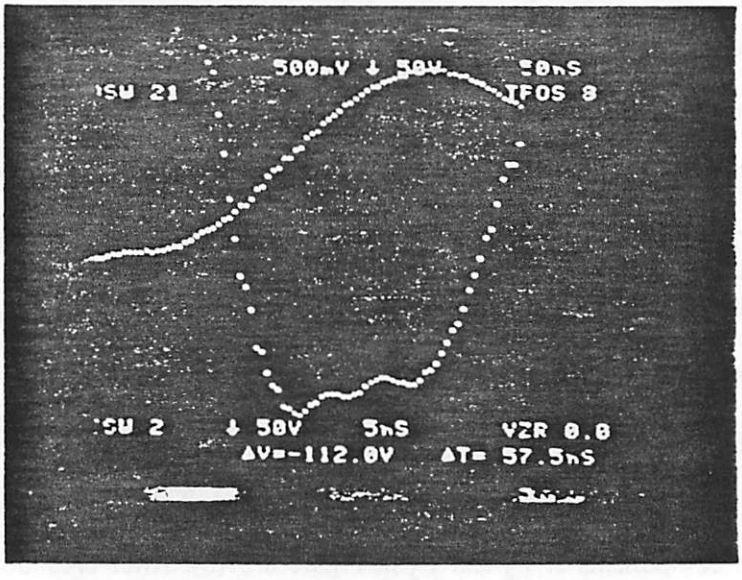


Figure 8.19: Experimental diode current and voltage under inductive turn off. Note the presence of oscillations in the diode voltage.

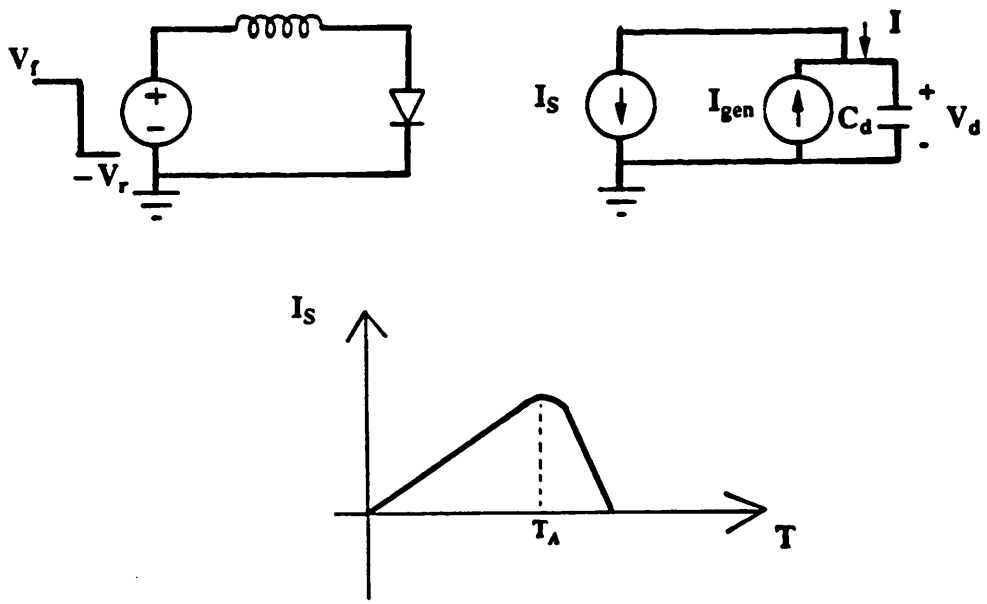


Figure 8.20: A physical model to explain the oscillations under inductive turn off of the diode.

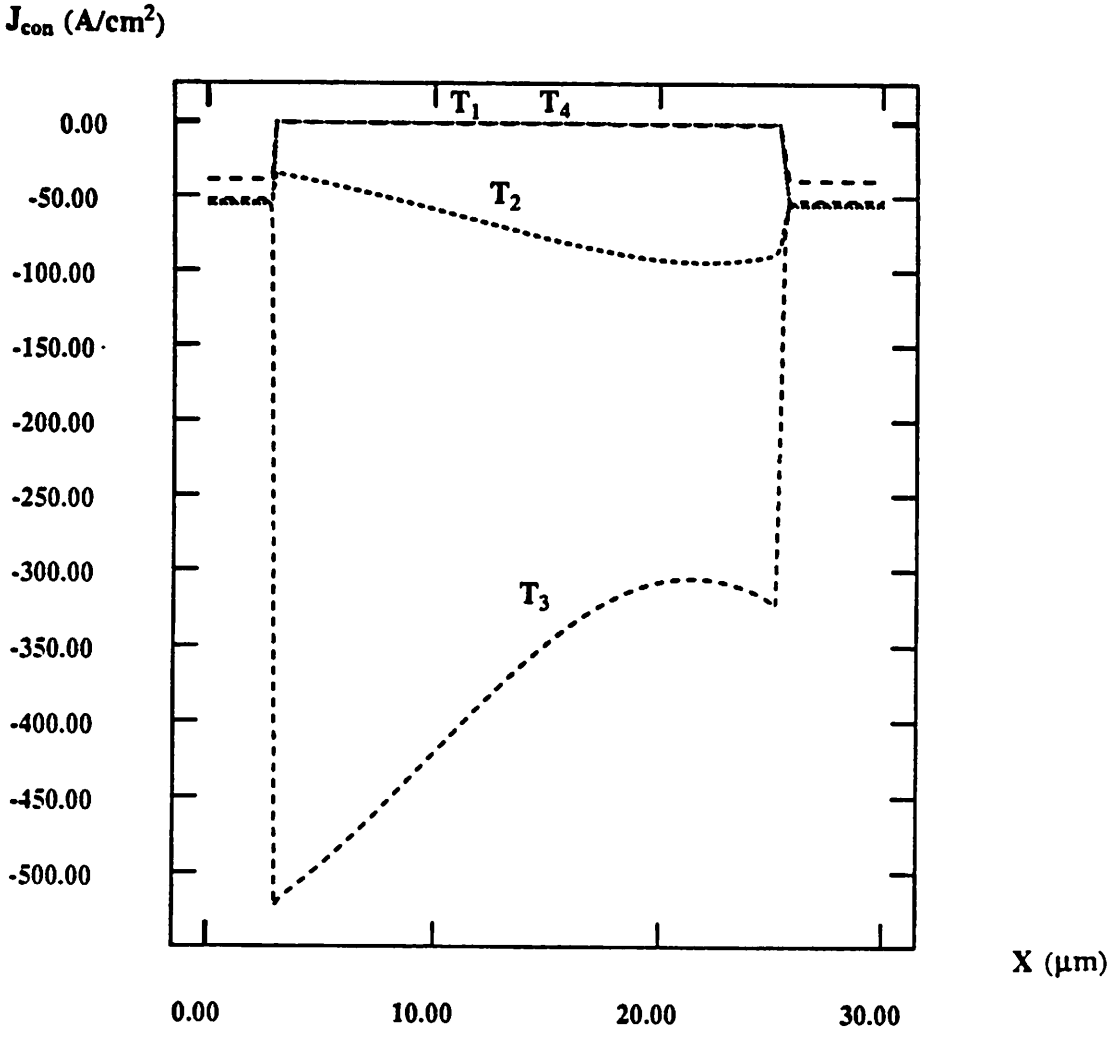


Figure 8.21(a): Conduction current density as a function of position for the four time instants shown in Figure 8.18(b).

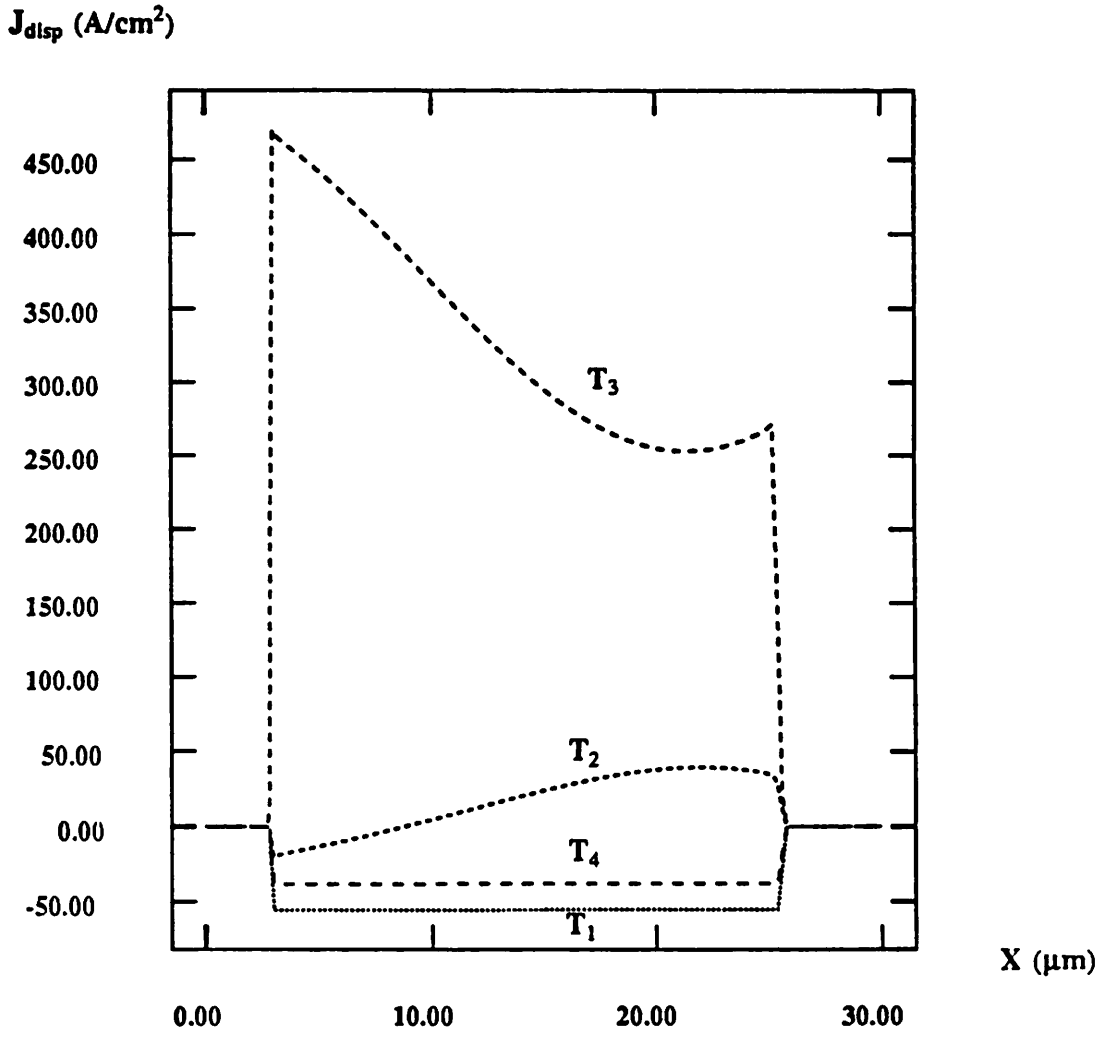


Figure 8.21(b): Displacement current density as a function of position for the four time instants shown in **Figure 8.18(b).**

breakdown value, I_{gen} becomes zero (as seen from Figure 8.21(b), the conduction current is zero at T_4); the very effect that caused the voltage to decrease disappears. Since the diode current (I_S) is still nonzero, the capacitor charges up again, leading to an increase in the reverse voltage across the diode. The above process can repeat resulting in several cycles of oscillation; the number of oscillation cycles depends on the duration of the second phase of reverse recovery. The longer it is, the greater the number of oscillation cycles.

8.3.3. Snubber Circuit Design

The voltage spike and the power dissipation observed during the rectifier turn off can be limited by the use of additional circuitry called a *snubber*. An *RC* snubber is commonly used to dissipate the energy stored in the inductance present in the circuit. A detailed analysis for the peak reverse voltage and snubber design is given in [8.19]. The analysis is based on the assumption that the diode current goes to zero as soon as the snubber circuit starts conducting. Thus, any interaction between the diode and the snubber circuit is ignored and a linear *RLC* circuit is analyzed. The analysis provides a means for selecting the values of R and C for an optimum design. CODECS simulations confirm that this analysis provides a good optimum design. However, it is pessimistic or conservative under other conditions.

The pin diode is shown with an *RC* snubber connected in parallel in Figure 8.22. The simulated and calculated [8.19] values of peak reverse voltage for a peak reverse current of 2A and a circuit inductance of 0.5 μH are shown in Figure 8.23. It is seen that the analysis considerably over estimates the peak reverse voltage for large values of the damping factor ($\xi = \frac{R}{2\sqrt{L/C}}$).

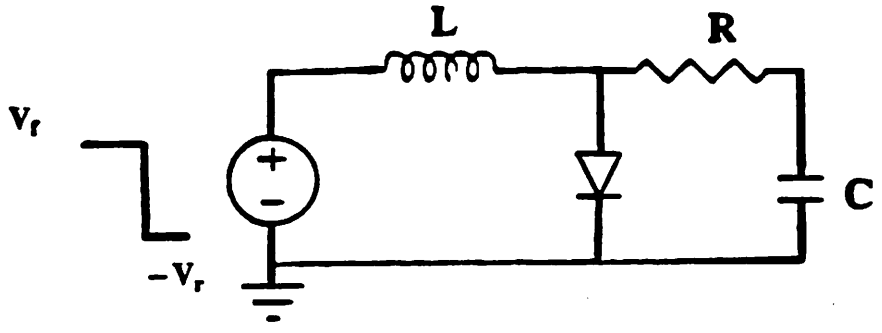


Figure 8.22: Snubber circuit used for simulation.

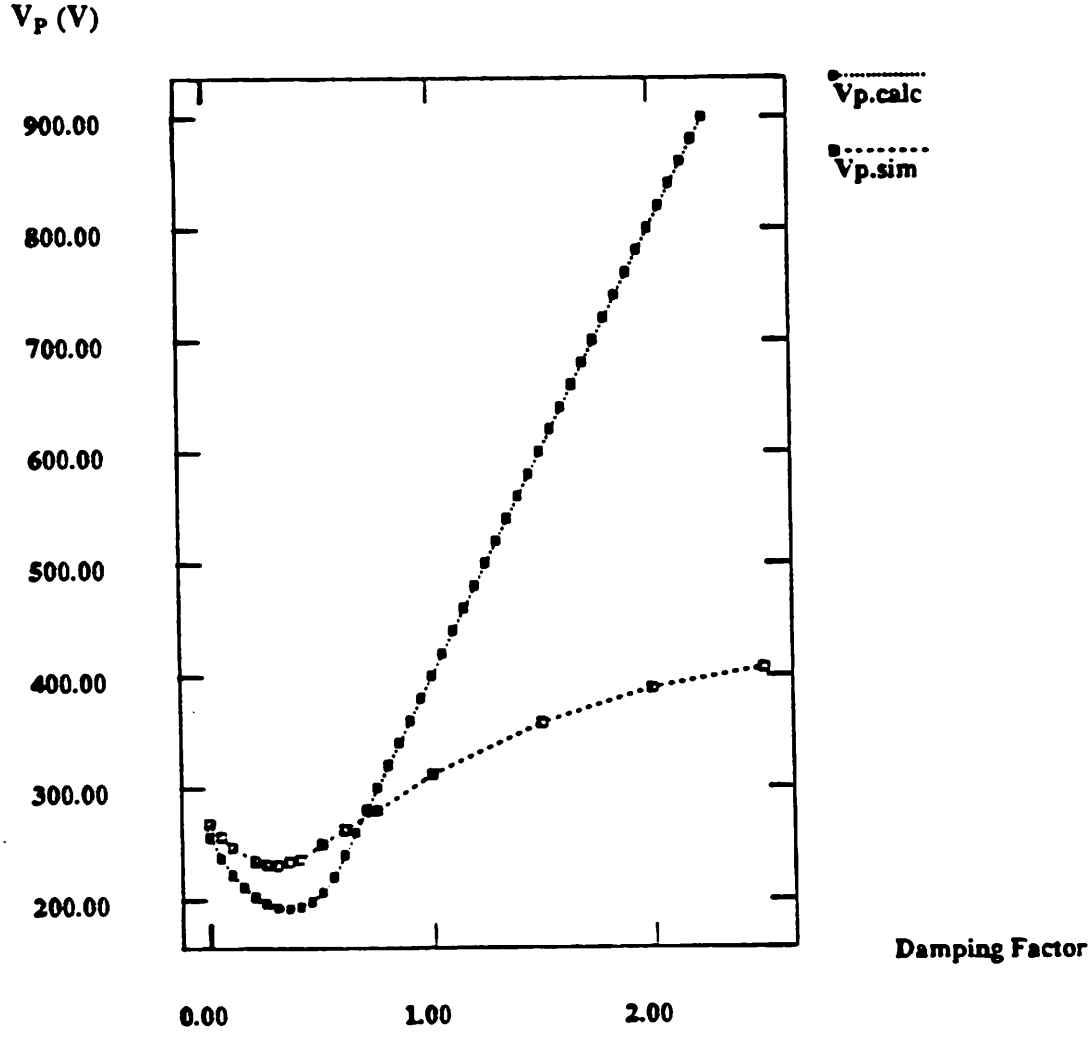


Figure 8.23: Peak reverse voltage as a function of the damping factor ($\xi = \frac{R}{2\sqrt{L/C}}$) obtained from simulations and from first-order calculations.

The energy dissipated in the diode and the snubber over a 100ns time interval for various values of R are summarized in Table 8.2. These results are obtained from repeated simulations with CODECS for a fixed value of C with the value of R being varied. As seen from Table 8.2 there is a remarkable tolerance of the power dissipation to the choice of R. This example illustrates the usefulness of CODECS in evaluation of snubber circuit designs.

R (C=50pf) (ohms)	Diode (μJ)	Snubber (μJ)
0	0.4	0.05
10	0.32	1.58
50	0.43	1.55
60	0.39	1.55
70	0.32	1.59
100	0.31	1.59

Table 8.2: Energy dissipated in diode and snubber

8.4. Evaluation of Switch-Induced Errors

The turning off of a MOSFET switch in switched-capacitor circuits results in an error voltage on the data-storage capacitor. This error voltage places a fundamental limit on the accuracy of A/D and D/A converters and filters implemented using switched-capacitor circuits. A simple switched-capacitor circuit is shown in Figure 8.24. Initially the MOS transistor is on and charges the storage capacitor C_L to the source voltage. Once the capacitor has been charged to the source voltage the MOS switch is turned off. The MOS transistor stores a certain amount of mobile charge in the channel and when the transistor is turned off this charge flows out of the drain, source, and substrate nodes. Some of the channel charge is transferred to the storage capacitor which discharges C_L

and gives rise to an error voltage. Clock-feed-through due to the overlap capacitances is another source of error. The error voltage can be reduced by turning off the switch at a very slow rate but cannot be completely eliminated.

The switch-induced error has been analyzed in [8.20] and a model has been obtained for the error voltage with a two-lump approximation. The dependence of the error voltage on the gate voltage has been examined under the quasi-static assumption. This model produces results that are significantly different from those obtained with a non-quasi-static MOSFET model [8.21]. In [8.22] the model has been extended to include the effect of source resistance and capacitance, but the analytical model is limited to some special cases. An analytical model for charge injection has also been given in [8.23] and verified by a numerical solution of the one-dimensional current-continuity equation. However, the model does not include the effect of the driving source impedance. Numerical solution of the one-dimensional current-continuity equation has also been used for the turn-off analysis of a MOS pass transistor [8.24]; based on this, a two-lump model is presented. However, this analysis also assumes a zero source impedance. With CODECS it is possible to simulate arbitrary switched-capacitor circuit configurations. In particular, the influence of source resistance and capacitance on the error voltage can be determined. In fact, realistic driving sources such as opamps can be used for the simulations.

For simulations the MOS transistor geometry and doping are as in Figure 7.18. The width of the transistor is assumed to be $4\mu m$. For C_L of 2pF the ratio of load capacitance to the gate capacitance of the MOS pass transistor, C_L/C_{ox}, is 242 and the ratio of the overlap capacitance to the gate capacitance, C_{ov}/C_{ox}, is 0.133.

The error voltage at the end of the falling ramp of the gate voltage is plotted in Figure 8.25 as a function of the ramp fall rate for different values of the source voltage. The error voltage increases with the fall rate and reaches a maximum. For very high fall

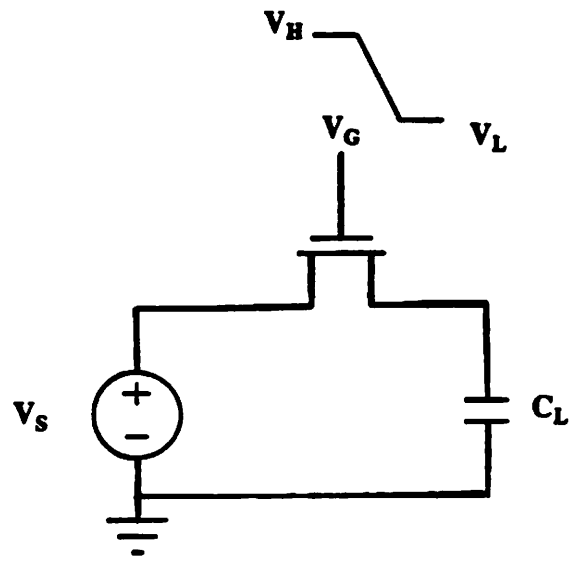


Figure 8.24: A simple switched-capacitor circuit.

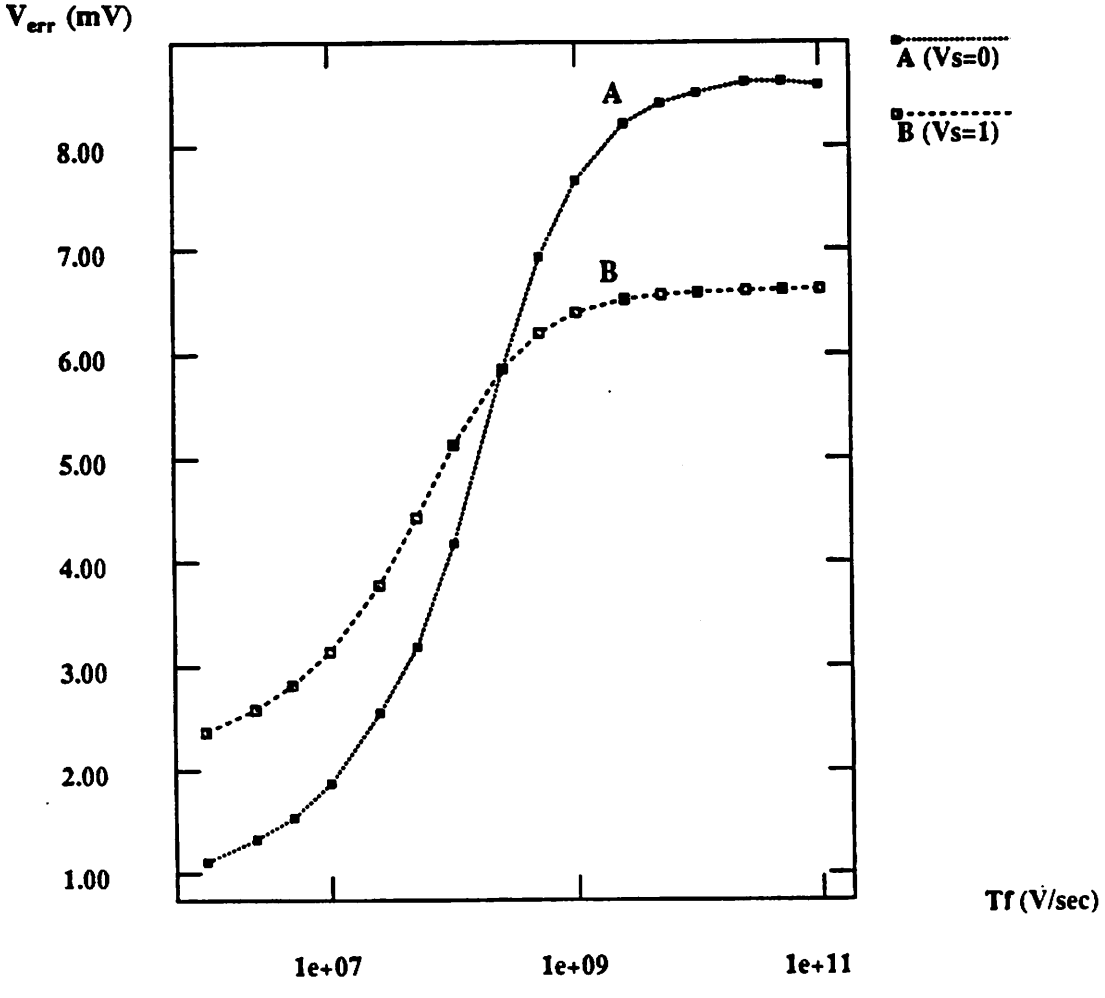


Figure 8.25: Switch-induced error voltage as a function of the gate voltage fall rate (Tf) for source voltages of 0V and 1V.

rates the error voltage drops due to non-quasi-static operation.

Next consider the influence of source resistance, R_S, on the error voltage. The simulated error voltages as a function of R_S, and the gate voltage fall rates, for a zero source voltage, are presented in Figure 8.26. It is seen that the switch error increases with an increase in the source resistance. This is to be expected, since a larger amount of channel charge flows into the storage capacitor than into the source. The source resistance restricts the flow of charge to the voltage source. The error voltage also increases with an increase in the fall rate initially, but decreases at high fall rates.

The effect of source capacitance, C_S, is shown in Figure 8.27. The error voltage is plotted as a function of C_S for different fall rates, $V_S = 0$, and $R_S = 10K$. The source capacitance has negligible effect at very small and very large fall rates. However, for values of fall rate in between the two extremes the error voltage decreases as C_S increases.

Next the circuit shown in Figure 8.28 is considered. The error voltage for different circuit configurations is plotted as a function of the fall rate of the gate voltage in Figure 8.29. The different configurations are

(i) $C_S = 0$, $R_S = 10K$ and no M2,

(ii) $C_S = 2pF$, $R_S = 10K$ and no M2,

(iii) $C_S = 0pF$, $R_S = 10K$ and M2 as shown (dummy technique), and

(iv) $C_S = 2pF$, $R_S = 10K$ and M2 as shown (balanced technique).

Thus, various techniques for error cancellation such as the dummy technique [8.25] and the balanced technique [8.26] can be evaluated. It is seen from Figure 8.29 that the balanced technique is most effective in reducing the switch-induced error voltage.

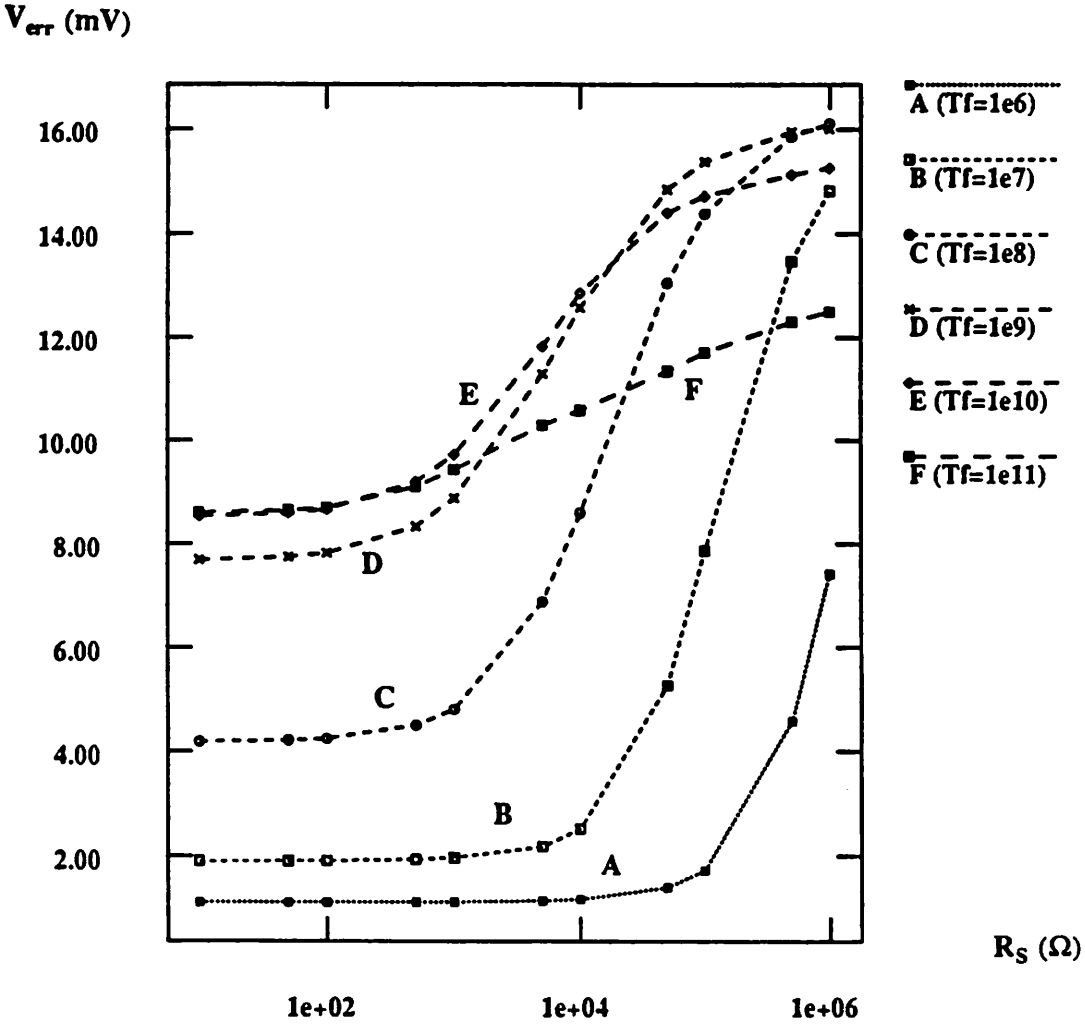


Figure 8.26: Switch-induced error voltage as a function of the source resistance and gate voltage fall rates for a source voltage of 0V.

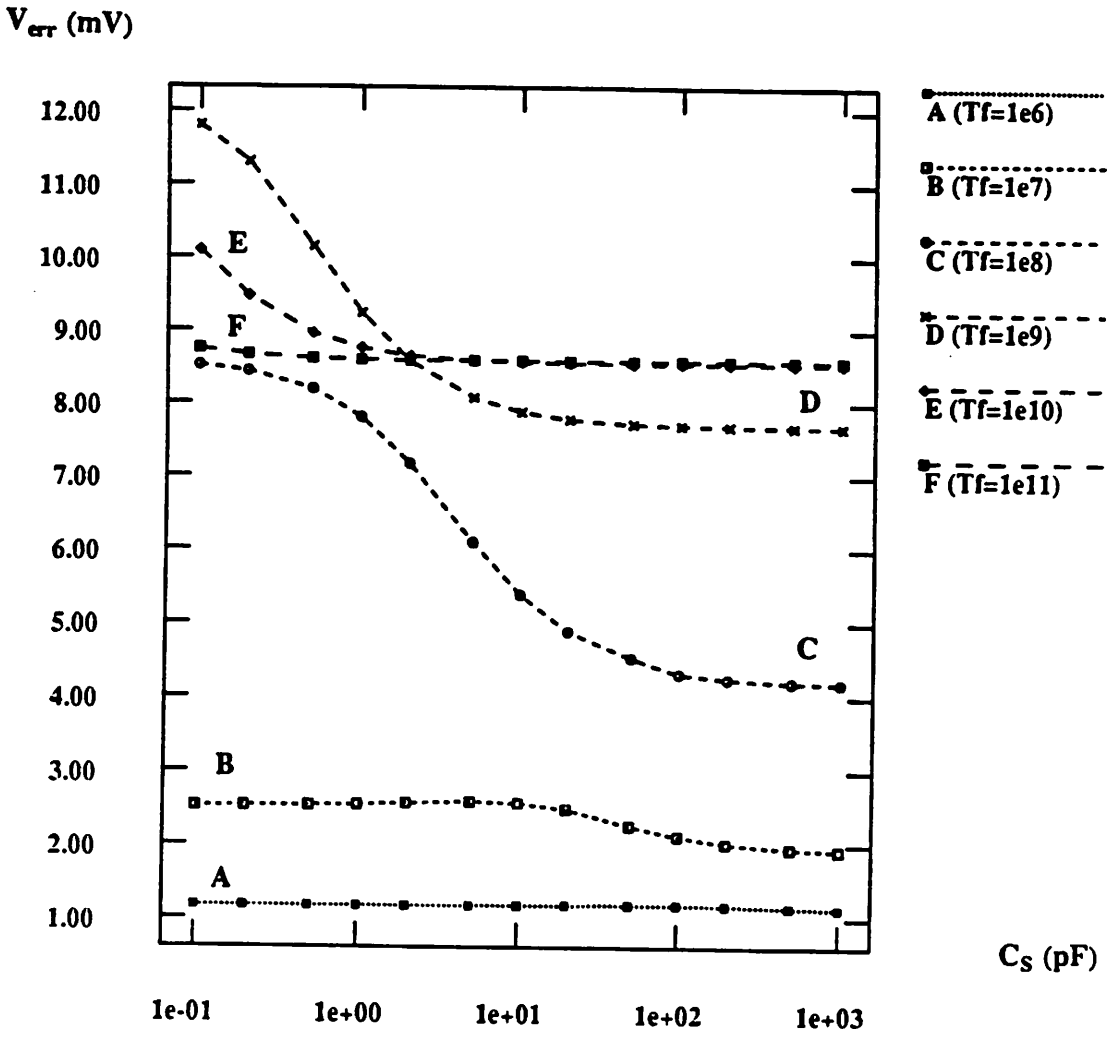


Figure 8.27: Switch-induced error voltage as a function of the source capacitance and gate voltage fall rates for a source voltage of 0V.

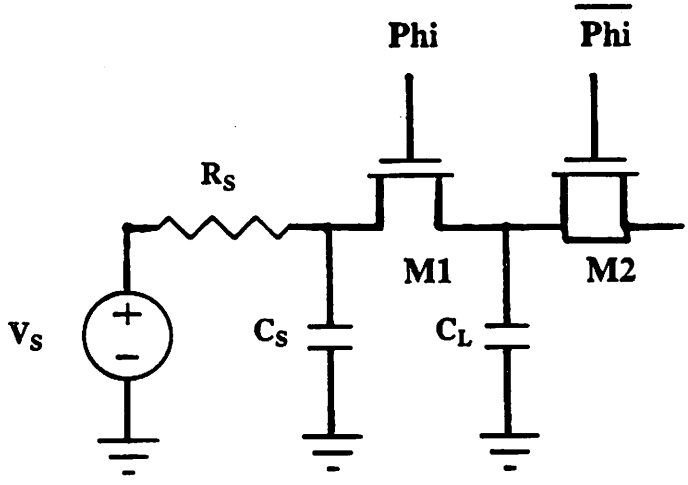


Figure 8.28: Another switched-capacitor circuit.

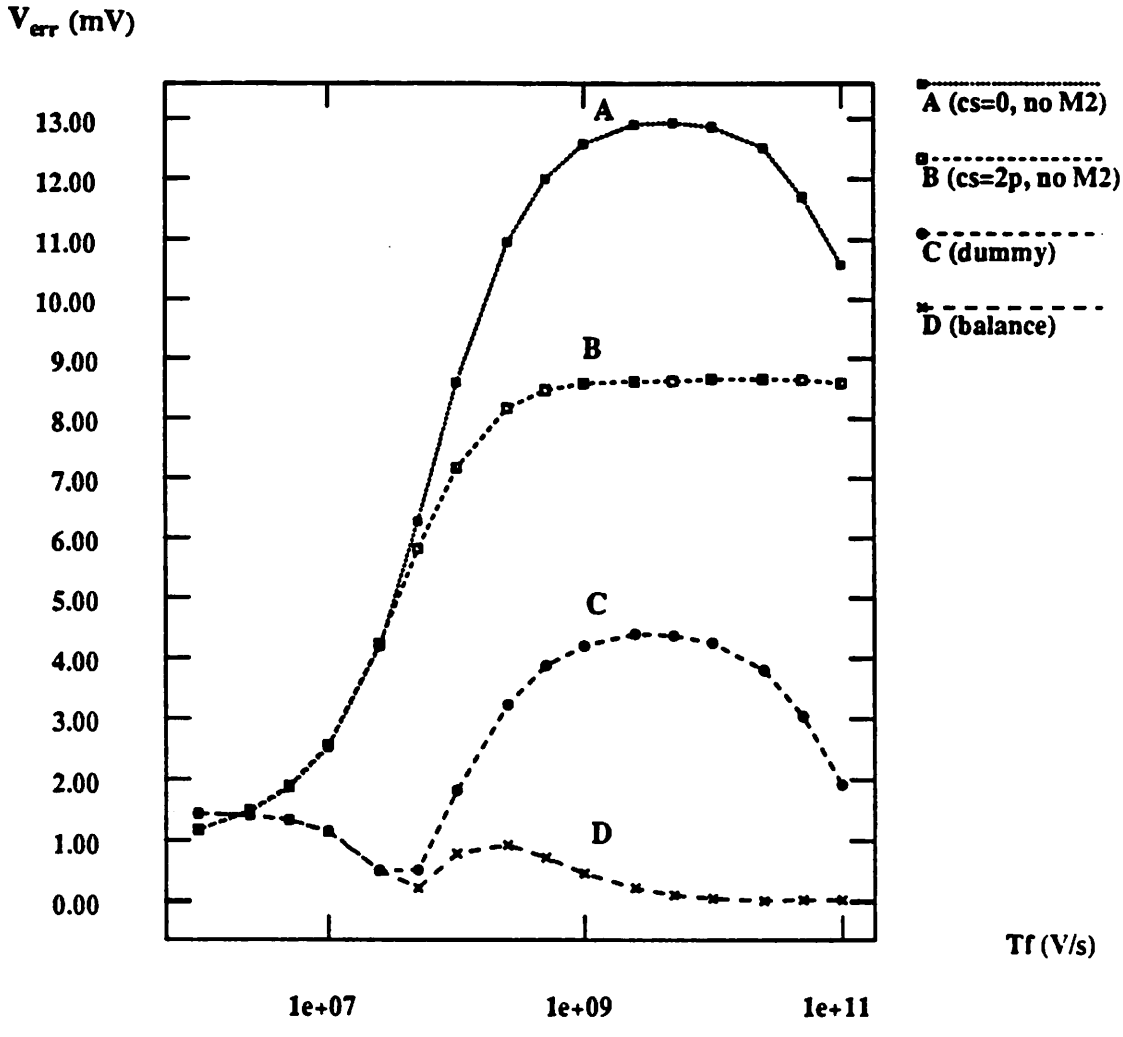


Figure 8.29: Switch-induced error voltage as a function of the gate voltage fall rate for different circuit configurations. The balanced technique for error cancellation results in the smallest error voltage.

8.5. Evaluation of Analytical Models

This section illustrates the application of CODECS for verification of analytical models. CODECS provides a capability to evaluate analytical models, and to see if an analytical model properly incorporates the physical operation of the device.

As a first example CODECS is used as a device simulator. An analytical model for the lateral electric field in lightly doped drain MOSFETs has been derived using a quasi-two-dimensional analysis [8.27]. Simulations from CODECS are used to verify the model. The electric fields obtained from the analytical model, for the full-overlap and non-overlap devices are plotted in Figures 8.30 and 8.31, respectively. Results of two-dimensional CODECS simulations for the same device and bias conditions are also included in these figures. The analytical model shows good physical agreement with the numerical simulations.

The second example concerns the turn-on and turn-off transients of a MOSFET as illustrated in Chapter 7. In Figure 8.32 are plotted the simulated drain and source currents for the turn-on transient. Figure 8.33 gives the corresponding results for the turn-off transient. The simulated results are from CODECS and the non-quasi-static model of Park [8.28]. The analytical model predicts that the currents cannot change instantaneously and the nature of the simulated waveforms is similar to that obtained from CODECS. The steady-state values of the currents for the turn on are different because of the differences in the dc characteristics of the numerical and analytical models It can, therefore, be concluded that the model of [8.28] properly simulates non-quasi-static operation.

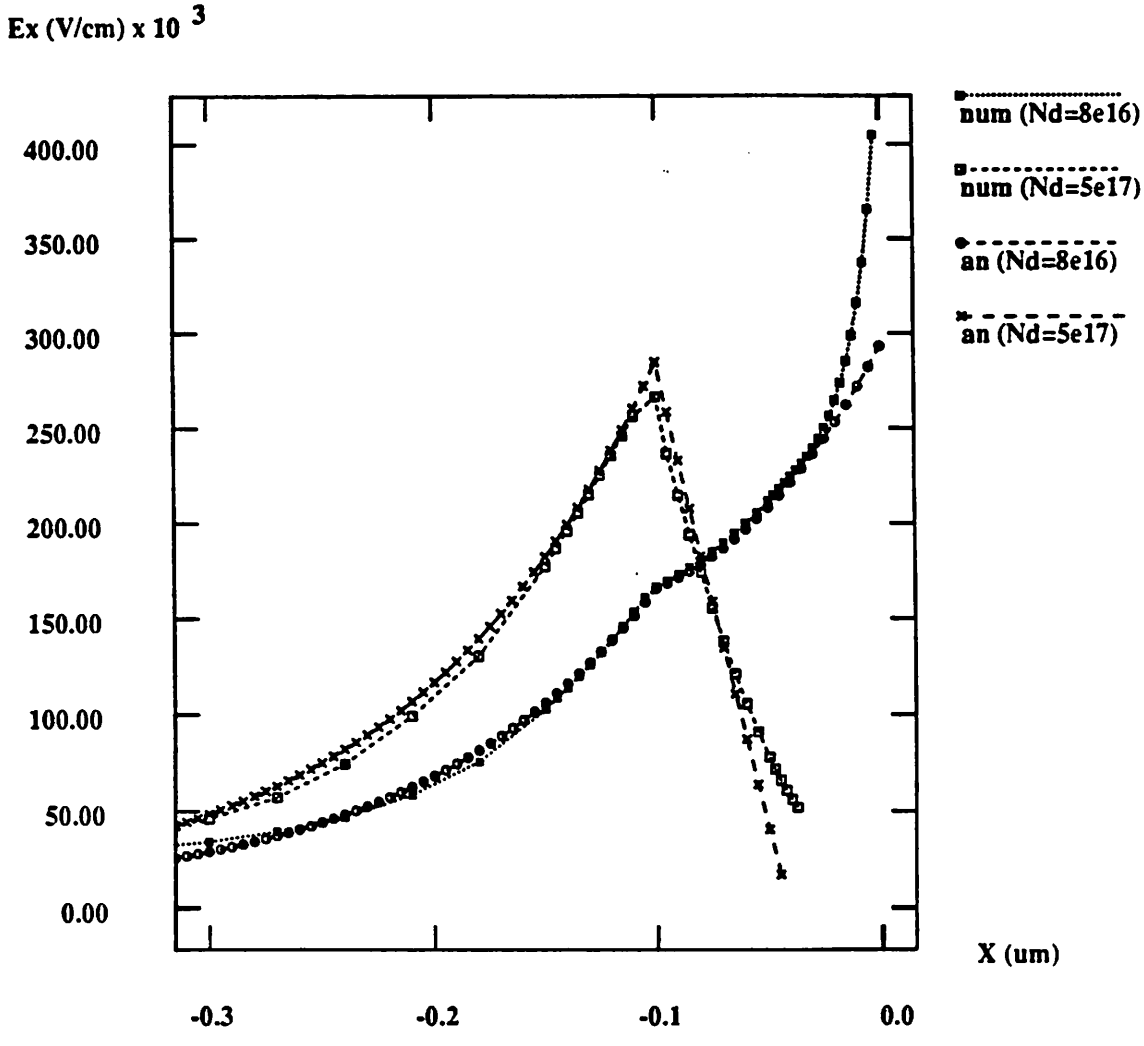


Figure 8.30: Lateral electric field versus distance for a LDD MOSFET with full gate/LDD-region overlap. The curves are shown for V_G = 4V, V_D = 5V, and n^- concentrations of 8×10^{16} and 5×10^{17}. There is good agreement between the simulated results and the model predictions.

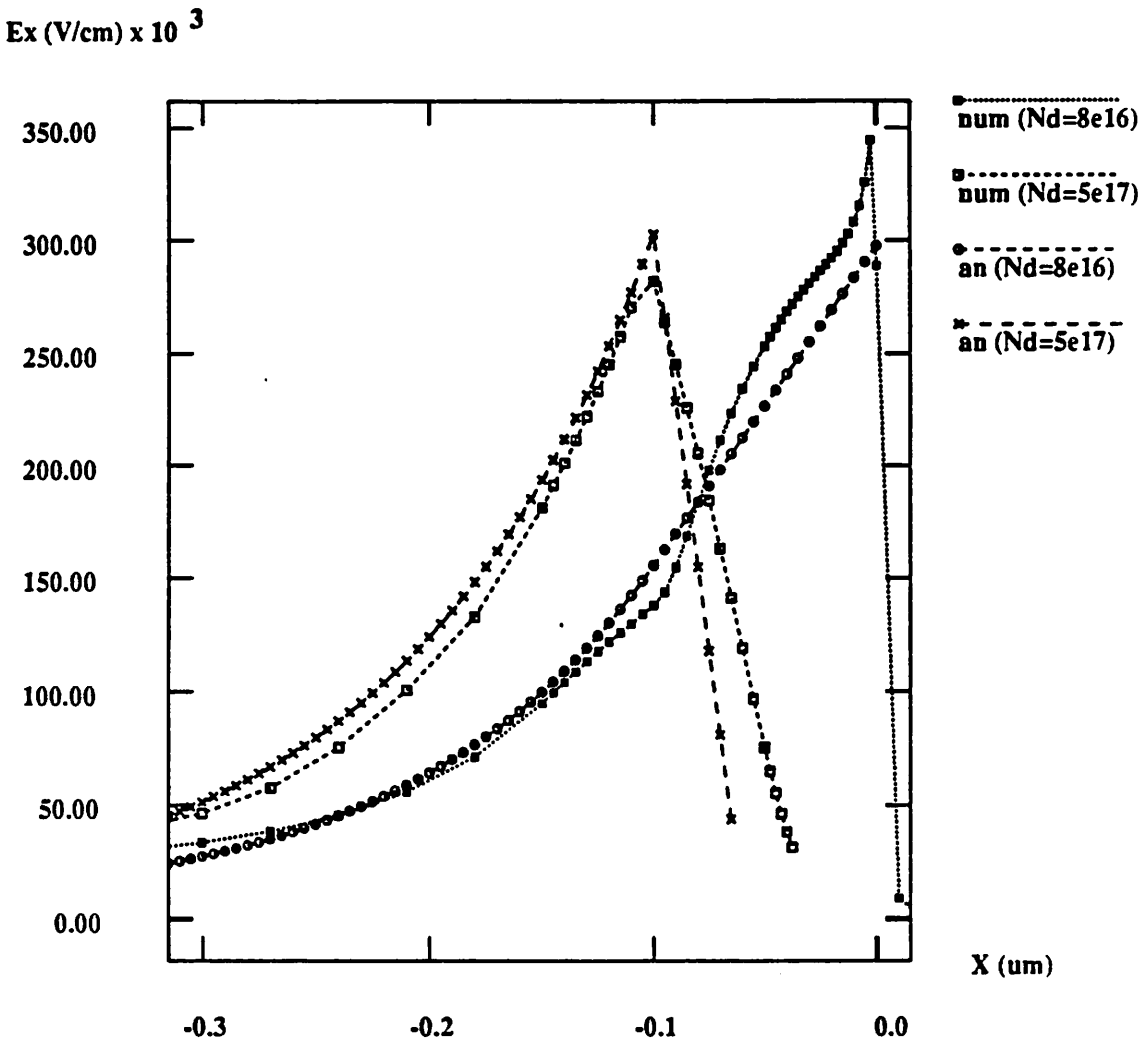


Figure 8.31: Lateral electric field versus distance for a LDD MOSFET with no gate/LDD-region overlap. The curves are shown for V_G = 4V, V_D = 5V, and n^- concentrations of 8×10^{16} and 5×10^{17}. There is good agreement between the simulated results and the model predictions.

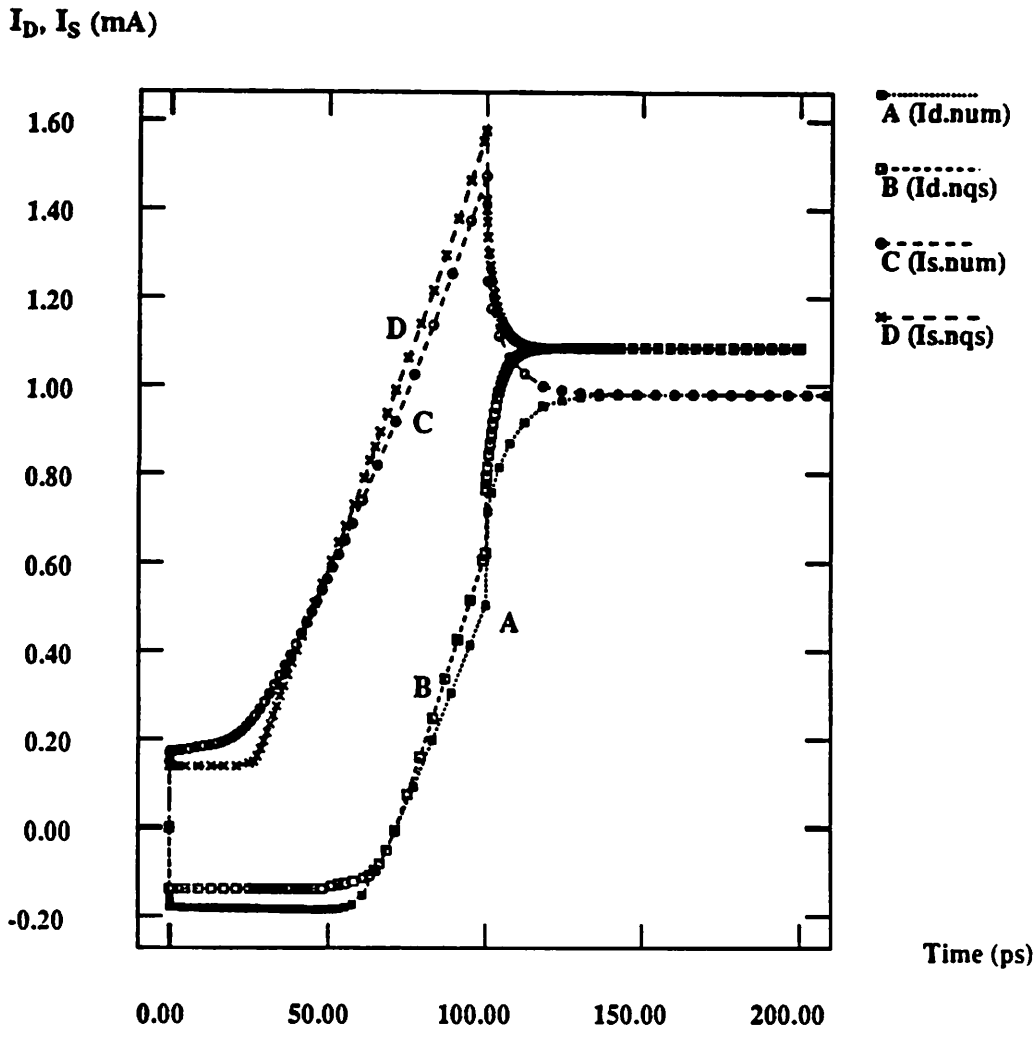


Figure 8.32: Drain and source currents for the turn-on transient of a MOSFET. The results from the non-quasi-static model of [8.28] are in good agreement with numerical simulations.

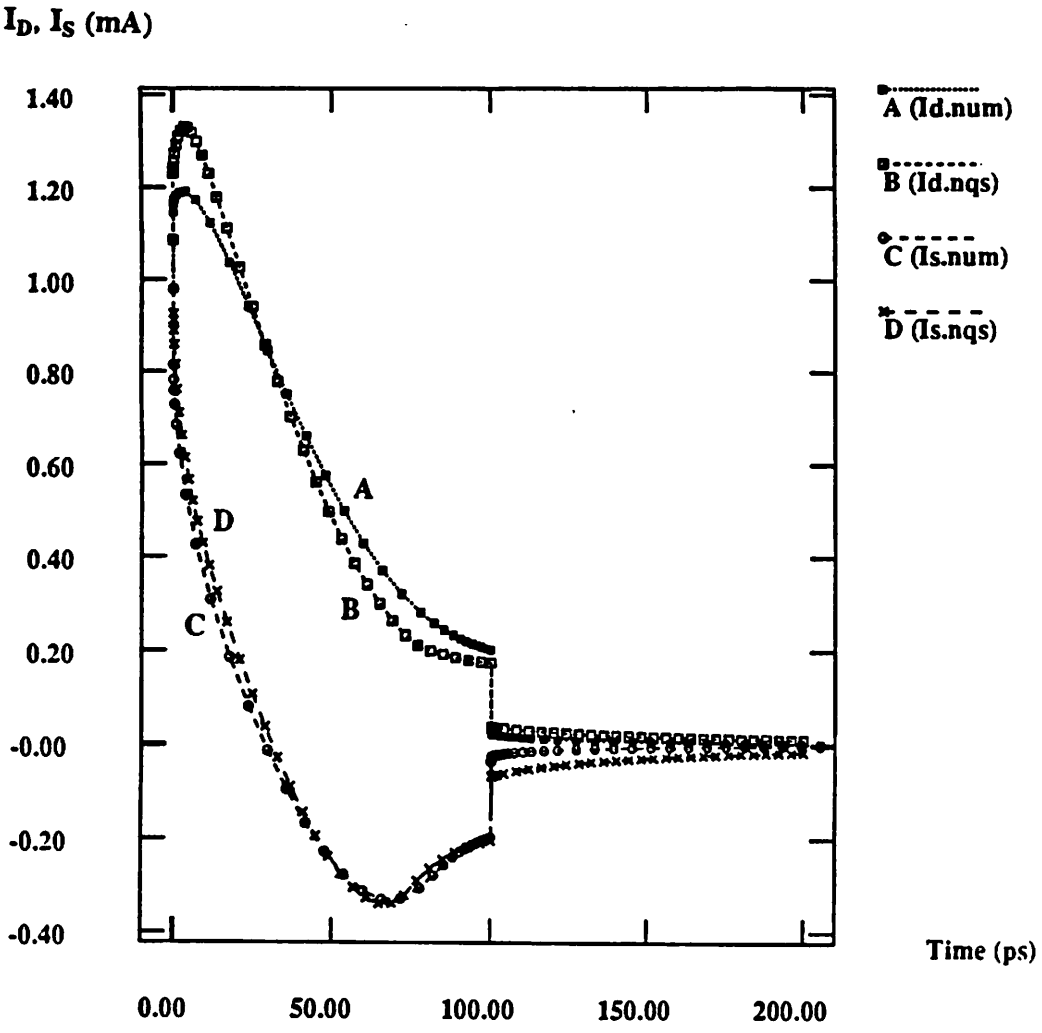


Figure 8.33: Drain and source currents for the turn-off transient of a MOSFET. The results from the non-quasi-static model of [8.28] are in good agreement with numerical simulations.

CHAPTER 9

Conclusions

Usually circuit simulators make use of analytical models for semiconductor devices. These models are derived from the basic physical laws governing device operation under simplifying assumptions. Empirical factors are introduced such that a good fit can be obtained with measured data from fabricated devices. Parameter-extraction programs are used to obtain the model parameters for a device model from experimentally measured data. These models cannot predict the effect of changes in process technology and device design on circuit performance. Repeated fabrication of devices and circuits is an expensive and time consuming way to make such evaluations.

Device simulations can be used to predict the effect of technology changes on device performance. This information could be used with a circuit simulator to predict circuit performance. However, the use of curve fitting to obtain the model parameters for the analytical models, from the simulated current-voltage characteristics, results in a loose coupling between the device simulator and the circuit simulator. Such an approach may not provide adequate information for a predictive analysis.

A coupled device and circuit simulator is a general solution to the modeling problem. Detailed information from the device-level simulations can be used at the circuit level. It provides a direct link between technology and circuit performance and can be used to evaluate the impact of alternate processes and device designs on circuit performance. In addition, such a simulator extends the capabilities of a device simulator in that devices can be simulated with circuit embedding. Present day device simulators allow

only voltage or current boundary conditions and resistive or capacitive elements at the terminals; therefore, they cannot be used to simulate realistic boundary conditions. In practice the device is embedded in a circuit and a mixed-level circuit and device simulator can be used to study its performance with circuit boundary conditions.

The accuracy of analytical models used for circuit simulation can also be verified by mixed-level circuit and device simulations. If some important physical effects are ignored in the analytical model, then the model has to be modified to account for those effects. Using coupled simulations, one can also identify circuit applications where better analytical models are essential.

This dissertation presents a general framework for mixed-level circuit and device simulation. The goal has been to implement frequently required analyses capabilities with a wide variety of numerical models. A new coupled device and circuit simulation program, CODECS, has been developed as part of this research. It incorporates SPICE3 for the circuit-simulation capability and for analytical models of semiconductor devices. The coupling techniques are general and can be used with other circuit simulators. A new one- and two-dimensional device simulator has been developed that supports dc, transient, small-signal ac, pole-zero, and sensitivity analyses at the device level. The device simulator provides a comprehensive set of analyses capabilities that are not available with any existing device-simulation programs.

Various algorithms to couple the device and circuit simulators for dc and transient analyses have been implemented in CODECS. These algorithms are evaluated based on their convergence properties and runtime performance. This study provides guidelines for choice of a particular coupling algorithm. A modified two-level Newton algorithm is used for dc analysis whereas a full-block-LU decomposition algorithm is used for transient analysis. This combination of algorithms provides reasonable convergence and runtime performance. Coupling for small-signal ac and pole-zero analyses is also described.

Dc convergence is a serious problem in coupled circuit and device simulation. The device-level equations are highly nonlinear and iterative methods fail to converge to a solution if the terminal voltages change by a large amount. Heuristics have been investigated to improve convergence for dc analysis. Some of the techniques used are the modified two-level Newton algorithm with device-based limiting, a damped-Newton method, and a backtracking scheme if convergence is not achieved for a particular bias point. The limiting schemes may be conservative but are justified since dc analysis is a very small fraction of a complete transient simulation. Schemes for timestep and integration-order control in transient analysis are also described. CODECS uses the local error estimates for automatic control of timesteps and integration order. Latency check is extremely important for mixed-level circuit and device simulation since evaluation of numerical models is computationally expensive. A scheme for device bypass is presented and evaluated. The implementation of small-signal ac and pole-zero analyses, and dc and transient sensitivity computations at the device level are also described.

Analytical models are compared with numerical models to indicate their weaknesses under various regions of device operation. Since numerical models are based on physics they provide more reliable results and a means for identifying the shortcomings of analytical models. It is shown that even if the model parameters are extracted such that there is a good agreement with the dc characteristics obtained experimentally, the transient response may be significantly different. A good dc model does not guarantee accurate transient operation. The speed-performance comparison indicates the cost of using numerical models instead of analytical models. However, some applications well justify the computational costs to obtain physically meaningful simulation results.

Applications have been used to illustrate the advantages of mixed-level circuit and device simulation. These examples again provide evidence that analytical models are of limited utility in some applications. Typical examples are high-level-injection effects in

diodes and bipolar transistors, high-frequency or non-quasi-static operation of semiconductor devices, charge redistribution during the turn off of an MOS switch in MOS switched-capacitor circuits, and simulation of power devices. The last application is particularly suited to mixed-level circuit and device simulation, since most power circuits employ a small number of power devices for which accurate models are necessary; the rest of the circuit can be simulated with analytical models.

CODECS has also been used as a vehicle to study the performance of Lisp-based mixed-level circuit and device simulation. An initial prototype for CODECS was developed in Lisp using object-oriented programming on the Symbolics 3600 Lisp machine. Although Lisp provided an environment for rapid prototyping, the Lisp-based simulator was extremely slow to be suitable for practical applications. Detailed runtime profiles are presented to identify the portions where the program spends most of the time. Comparisons are provided with the new version of CODECS in the C programming language.

Future research should focus on speeding up the mixed-level circuit and device simulator. This can be achieved by the use of vector or multiprocessing computers. Parallel model evaluation [9.1] in a mixed-level circuit and device simulator should provide an almost linear speedup with the number of processors. Loosely coupled multiprocessors may be quite suited since the time required for model evaluation will significantly dominate the interprocessor communication times. The device-level simulation can be parallelized as in [9.2] and circuit-level simulation can be parallelized using some existing techniques [9.1, 9.3, 9.4]. Algorithmic approaches to speed up the device-simulation task also need to be investigated; the Boundary-Value method of [9.5] should be evaluated for simulation of MOS devices in a mixed-level environment. Development of a special-purpose hardware solver for solving the Poisson's and the current-continuity equations would provide a significant improvement in speed

performance.

A mixed-level circuit and device simulator is only as good as the physical models used at the device-simulation level. CODECS currently uses physical models that have been widely accepted in the literature. These models should be updated as better models become available. In addition, the device-simulation capability should be tuned to a process line and the models verified with actual measurements. This will provide a basis for comparing experimentally fabricated circuits and simulated results.

Additional analysis capabilities need to be implemented at the device-simulation level. These include the use of numerical models for noise calculations. Numerical techniques have been used for calculating the low-frequency noise of MOSFET's in [9.6].

To be useful for power device designers CODECS should also allow the simulation of thyristor circuits. This can be achieved by implementing current boundary conditions at the device-simulation level. Temperature gradients are important in power devices. Consequently, the heat-flow equation must be solved coupled with the Poisson's and the current-continuity equations. Some device simulators allow non-isothermal simulations [9.7], and such a capability will be extremely beneficial in CODECS.

As devices are scaled down three-dimensional effects become important. Modern IC fabrication processes may dictate the use of three-dimensional numerical models in a mixed-level circuit and device-simulation environment. The coupling techniques used in CODECS can be extended to three-dimensional numerical devices. However, computational costs are going to be of serious concern and multiprocessors should be used. Mixed-level circuit and device simulations could be extremely useful for future three-dimensional or multi-layer technologies.

At present CODECS works for rectangular geometry silicon devices. The present capabilities of CODECS should be enhanced to other materials such as GaAs and heterostructure semiconductor devices. CODECS should also be extended to simulate

generalized geometries; this could be done by use of triangular grids as in PISCES [9.8].

A mixed-level circuit and device simulator provides an environment that should be useful to process, device and circuit designers. Process designers could use CODECS coupled with a process simulator to design a high-performance process that meets a desired set of specifications for a given circuit design. This would then result in a true custom IC process. Device designers can evaluate alternate device designs and their impact on circuit performance. Circuit designers could use the numerical models when greater accuracy is desired for simulations as would be the case with some high-performance circuits.

APPENDIX A

CODECS User's Guide

A.1. Introduction

CODECS is a *co*upled *de*vice and *c*ircuit *s*imulator that allows accurate and detailed simulation of semiconductor circuits. The simulation environment of CODECS enables one to model critical devices within a circuit by physical (numerical) models based upon the solution of Poisson's equation and the current-continuity equations. Analytical models can be used for the noncritical devices. CODECS incorporates SPICE3 for the circuit-simulation capability and for analytical models of semiconductor devices. Numerical models are provided by a new one- and two-dimensional device simulator. Dc, transient, and small-signal ac and pole/zero analyses can be performed on circuits containing one and two-dimensional numerical models. The numerical models in CODECS include physical effects such as bandgap narrowing, Shockley-Hall-Read and Auger recombinations, concentration- and field-dependent mobilities, concentration-dependent lifetimes, and avalanche generation.

The input format is similar to that of SPICE3. All circuit elements and analytical semiconductor device models can be used as described in the SPICE3 User's Guide. A description of the numerical models of CODECS is provided here.

A.2. Circuit Description

The circuit description can be entered in a manner similar to that of SPICE3. Circuit element description lines contain the element name, the circuit nodes to which it is connected, and the values of the parameters that determine the electrical characteristics of that element. A numerical diode name must begin with the letter *A*, a numerical BJT name must begin with the letter *B*, and a numerical MOSFET with the letter *N*.

Data fields that are enclosed in '< >' in the description below are optional.

A.3. Element Descriptions

A.3.1. Numerical Diodes

General form:

AXXXXXXX N+ N- MNAME <AREA VALUE> <WIDTH VALUE>

N+ and N- are the positive and negative nodes, respectively. MNAME is the model name. AREA is the area for a one-dimensional diode in cm^2. WIDTH is the width of a two-dimensional diode in cm. The default values of AREA and WIDTH are $1cm^2$ and $1cm$, respectively, and are used when these parameters are not specified.

Examples:

ACLAMP 1 2 AREA=1U

ACLAMP 1 2 WIDTH=1E-2

A.3.2. Numerical Bipolar Junction Transistors

General form:

BXXXXXXX NC NB NE MNAME <AREA VALUE> <WIDTH VALUE>

NC, NB, and NE are the collector, base, and emitter nodes, respectively. MNAME is the model name. AREA is the area for a one-dimensional BJT in cm^2. WIDTH is the width of a two-dimensional BJT in cm. The default values of AREA and WIDTH are $1cm^2$ and $1cm$, respectively, and are used when these parameters are not specified.

Examples:

BINV 1 2 0 AREA=1E-6

BINV 1 2 0 WIDTH=1E-2

A.3.3. Numerical MOSFETs

General form:

NXXXXXXX ND NG NS NB MNAME <WIDTH VALUE>

ND, NG, NS, and NB are the drain, gate, source, and bulk (substrate) nodes, respectively. MNAME is the model name. WIDTH is the width of the two-dimensional MOSFET structure in cm. The default value of the width is $1cm$, and is used when the WIDTH parameter is not specified in the input.

Examples:

MPULLDN VOUT VIN 0 0 ENH WIDTH=20E-4

MPASS 2 4 10 0 ENH WIDTH=1M

A.4. Model Descriptions

The numerical semiconductor device models are specified using the .MODEL command similar to that used for the analytical models in SPICE3.

General form:

.MODEL MNAME TYPE PNAME1 PVAL1 PNAME2 PVAL2 ...

MNAME is the model name. TYPE is NUMD for a numerical diode model, NBJT for a numerical BJT model, and NUMOS for a numerical MOSFET model. The parameters can be single valued, vector valued, or logical flags. One-dimensional and two-dimensional numerical models differ in their input parameters. A description of the model parameters is given below. First the common model parameters that model the various physical effects are described. This is followed by a description of the model parameters for one-dimensional models and two-dimensional models, respectively.

A.4.1. Parameters for Physical Effects

CODECS includes several physical effects which can be included in a simulation by specifying the corresponding parameters. The model parameters are

A.5. Model Parameters for One-Dimensional Numerical models

The parameters described in this section are used to define the mesh for space discretization, the doping profiles, and the material regions for the device.

A.5.1. Mesh Description

The mesh for a one-dimensional device is specified by the MESH model parameter.

General form:

name	parameter	default
SRH	Shockley-Read-Hall recombination	not included
AUGER	Auger recombination	not included
BGNW	bandgap narrowing	not included
CONCTAU	concentration-dependent lifetimes	not included
CONCMOB	concentration-dependent mobility	not included
FIELDMOB	field-dependent mobility	not included
AVAL	avalanche generation	not included
NBGN	reference concentration for bandgap narrowing model	$1\times10^{17}\ cm^{-3}$
TN0	low doping electron lifetime	20 *ns*
TP0	low doping hole lifetime	20 *ns*
MUN0	low doping, low field electron mobility	1400 $cm^2/V-s$
MUP0	low doping, low field hole mobility	480 $cm^2/V-s$
MOBMODEL	mobility model	Scharfetter-Gummel

MESH MESHNO XPOS

XPOS is the X position, in μm, of the mesh point specified by MESHNO. The mesh parameter can be used any number of times to generate a nonuniform grid.

Example:

MESH 1 0 MESH 11 1 MESH 51 2

The above command line specifies the mesh as follows. The first mesh point is located at $X = 0\mu m$. The 11th mesh point is at $X = 1\mu m$, and the 51st mesh point is at $X = 2\mu m$. Thus, ten mesh points are uniformly placed in the interval (0, $1\mu m$) and forty mesh points are uniformly placed in the interval ($1\mu m$, $2\mu m$). The mesh spacing is uniform within an interval but nonuniform from interval to interval.

A.5.2. Doping Description

The doping profile for the device is specified by analytical formulas. The allowed profile types are uniform, Gaussian, exponential, or error function. The general form of each of these profiles is as follows.

A.5.2.1. Uniform Doping Profile

General form:

UNIF CONC XLO XHI

UNIF specifies a uniform doping profile between XLO and XHI. Both XLO and XHI are given in cm. CONC is the doping level in cm^{-3}. A negative value is specified for a p-type material and a positive value for an n-type material.

The doping $N(x)$ is calculated using the following expression,

$$N(x) = \begin{cases} 0 & x < XLO \\ CONC & XLO \le x \le XHI \\ 0 & x > XHI \end{cases}$$

Examples:

UNIF 1E15 0 1E-4

This defines a n-type uniform doping of $1 \times 10^{15}\ cm^{-3}$ for the interval (0, $1\mu m$).

UNIF -1E16 0 5E-4

This defines a p-type uniform doping of $1 \times 10^{16}\ cm^{-3}$ for the interval (0, $5\mu m$).

A.5.2.2. Gaussian Implantation Profile

General form:

GIMP CONC CHAR XPOS

GIMP specifies a Gaussian implantation profile with a peak concentration CONC in cm^{-3}, a characteristic length of CHAR in cm, and the peak is located at XPOS in cm. CONC is a negative value for a p-type material. The doping $N(x)$ is calculated as

$$N(x) = CONC \times \exp\left[-\left[\frac{x - XPOS}{CHAR}\right]^2\right]$$

Examples:

GIMP 5E18 1.35E-4 6E-4

This defines a n-type Gaussian profile of peak concentration $5\times10^{18}\ cm^{-3}$, a characteristic length of 1.35 μm, with the peak of the profile placed at $6\mu m$.

GIMP -1E19 0.15E-4 0

This defines a p-type Gaussian profile of peak concentration $1\times10^{19}\ cm^{-3}$, characteristic length of 0.15 μm, with the peak of the profile at the origin.

A.5.2.3. Exponential Doping Profile

General form:

EXP CONC CHAR XPOS

EXP specifies an exponential profile with a peak concentration CONC in cm^{-3}, a characteristic length of CHAR in cm, and the peak is located at XPOS in cm. CONC is a negative value for a p-type material. The doping $N(x)$ is calculated as

$$N(x) = CONC \times \exp\left[-\frac{|XPOS - x|}{CHAR}\right]$$

Example:

EXP 5E18 1.35E-4 6E-4

This defines a *n*-type exponential doping profile of peak concentration $5\times10^{18}\ cm^{-3}$, a characteristic length of 1.35 μm, with the peak of the profile placed at $6\mu m$.

A.5.2.4. Error-Function Doping Profile

General form:

ERFC CONC CHAR XPOS

ERFC specifies an error-function profile with a peak concentration CONC in cm^{-3}, a characteristic length of CHAR in cm, and the peak is located at XPOS in cm. CONC is a negative value for a *p*-type material. The doping $N(x)$ is calculated as

$$N(x) = CONC \times \mathrm{erf}\left[\frac{|XPOS - x|}{CHAR}\right]$$

Example:

ERFC -1E19 0.15E-4 0

This defines a *p*-type error-function profile of peak concentration $1\times10^{19}\ cm^{-3}$, a characteristic length of 0.15 μm, with the peak of the profile at the origin.

A.5.3. Region Description

The REGION parameter describes the presence of different materials in the simulation domain. All mesh points must belong to some region. A region is specified as follows,

General form:

REGION MESHLO MESHHI

REGION is either SILICON or OXIDE. MESHLO is the mesh number at which the region begins and MESHHI is the mesh number at which the region ends.

Example:

SILICON 1 20 OXIDE 20 30

The above command line demarcates a silicon and an oxide region on a thirty-point mesh. Mesh points from 1 to 20 lie in the silicon region, and the oxide region extends from the 20th mesh point to the 30th mesh point.

A.6. Model Parameters for Two-Dimensional Numerical models

The parameters described in this section are used to define the mesh for space discretization, the doping profiles, the material regions, and the contact locations for the device. The coordinate system used for two-dimensional devices is shown in Figure A.1.

A.6.1. Mesh Description

Two-dimensional devices that can be simulated by CODECS must be rectangular in geometry. A rectangular mesh is used to partition the simulation domain. The XMESH and YMESH model parameters are used to specify the mesh in a manner similar to that for one-dimensional devices. The XMESH parameter is used to specify the mesh in the X direction and the YMESH parameter specifies the mesh for the Y direction.

General form:

XMESH XMESHNO XPOS

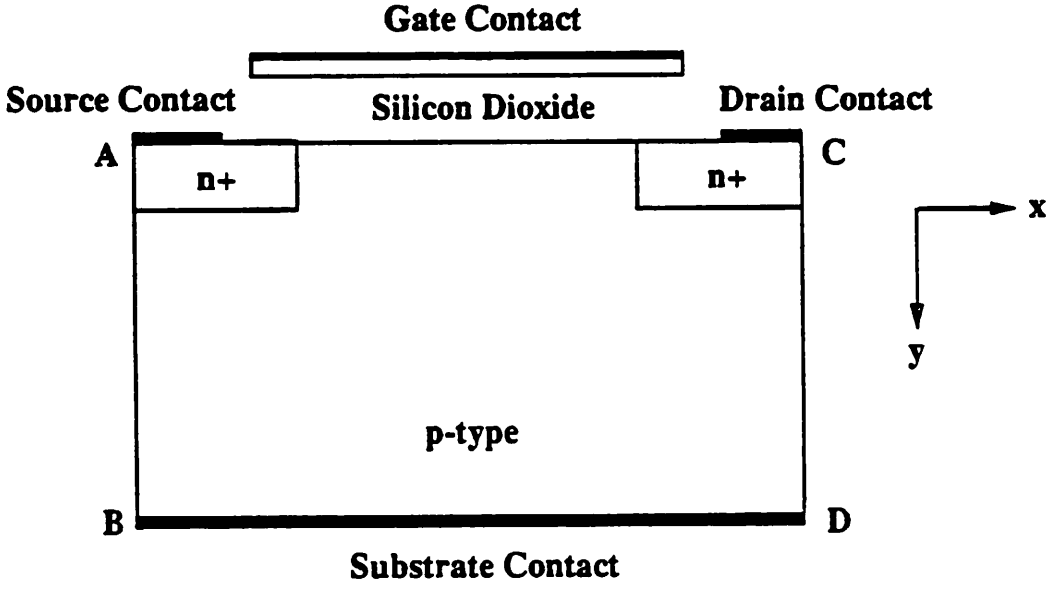


Figure A.1: Coordinate system used for two-dimensional devices.

YMESH YMESHNO YPOS

XPOS (YPOS) is the X (Y) position (in μm) of the mesh point specified by XMESHNO (YMESHNO). The mesh parameters, XMESH and YMESH, can be used any number of times to generate a nonuniform grid.

Example:

XMESH 1 0 XMESH 11 1 XMESH 51 2 YMESH 1 0 YMESH 21 2

The above command line specifies the two-dimensional mesh as follows. The simulation region is divided into subdomains by a 51×21 mesh. The mesh spacing is nonuniform in the X direction and uniform in the Y direction.

A.6.2. Doping Description

The doping profile for the device is specified by analytical formulas. The allowed two-dimensional doping profile types are uniform and Gaussian. The general form of each of these profiles is as follows

A.6.2.1. Two-Dimensional Uniform Doping Profile

General form:

UNIF CONC XLO XHI YLO YHI

UNIF specifies a uniform doping in the rectangle formed by XLO, XHI and YLO, YHI. XLO, XHI and YLO, YHI are given in cm. CONC is the doping level in cm^{-3}. A negative value is specified for a p-type material and a positive value for an n-type material. The doping $N(x)$ is calculated using the following expression,

$$N(x) = \begin{cases} CONC & XLO \le x \le XHI \text{ and } YLO \le y \le YHI \\ 0 & otherwise \end{cases}$$

Examples:

UNIF 1E16 0 1E-4 0 0.2E-4

This defines a *n*-type uniform doping of $1\times10^{16}\ cm^{-3}$ for the rectangular region $0 \le X \le 1\mu m$ and $0 \le Y \le 0.2\mu m$.

UNIF -2.5E16 0 4E-4 1E-4 2E-4

This defines a *p*-type uniform doping of $2.5\times10^{16}\ cm^{-3}$ for the rectangular region $0 \le X \le 4\mu m$ and $1\mu m \le Y \le 2\mu m$.

A.6.2.2. Two-Dimensional Gaussian Implantation Doping Profile

General form:

GIMP CONC CHAR POS LATRATIO DIRECTION XLO XHI YLO YHI

GIMP specifies a Gaussian implantation profile with a peak concentration CONC in cm^{-3}, a characteristic length of CHAR in cm, and the peak is located at POS in cm. CONC is a negative value for a *p*-type material. DIRECTION is the axis along which the profile is directed; 0 for the X axis and 1 for the Y axis. The lateral profile is assumed to have the same form but the characteristic length is shrunk by a factor of LATRATIO. The rectangle specified by XLO, XHI and YLO, YHI bounds the profile to a region inside the device. Within this rectangle the profile is a Gaussian profile along the principal axis. Outside the rectangle the profile falls off along the lateral axis according to LATRATIO.

Examples:

GIMP 1E19 1.6E-4 20E-4 0.8 1 0 1E-4 0 0

This defines a *n*-type Gaussian profile of peak concentration $1\times10^{19}\ cm^{-3}$, a characteristic length of $1.6\mu m$, with the peak of the profile placed at $20\mu m$. The profile

is aligned to the Y axis and the lateral profile has a characteristic length of 0.8 times the primary profile. The doping is assumed to be Gaussian in the Y direction in the region $0 \leq X \leq 1\mu m$. Outside this region the profile falls of along the X axis as a Gaussian profile with a characteristic length of $0.8 \times 1.6\mu m = 1.28\mu m$.

GIMP -1E15 1.2E-4 0 0.1 0 0 0 0 .5E-4

This defines a *p*-type Gaussian profile of peak concentration $1 \times 10^{15}\ cm^{-3}$, characteristic length of $1.2\mu m$, with the peak of the profile at $0\mu m$. The profile is aligned to the X axis and the lateral profile has a characteristic length of 0.1 times the primary profile. The doping is assumed to be Gaussian in the X direction in the region $0 \leq Y \leq 0.5\mu m$. Outside this region the profile falls of along the Y axis as a Gaussian profile with a characteristic length of $0.1 \times 1.2\mu m = 0.12\mu m$.

A.6.3. Region Description

The REGION parameter describes the presence of different materials in the specified mesh. A region is specified as follows,

General form:

REGION XMESHLO XMESHHI YMESHLO YMESHHI

REGION is either SILICON or OXIDE. XMESHLO, XMESHHI, YMESHLO, and YMESHHI are mesh numbers that define a rectangle whose material type is specified by REGION.

Example:

OXIDE 1 21 1 5 SILICON 1 21 5 11

The above command line demarcates a silicon and an oxide region on a two-dimensional mesh. The rectangle formed by XMESH numbers 1 and 21 and YMESH

numbers 1 and 5 is made up of oxide, whereas the rectangle formed by XMESH numbers 1 and 21 and YMESH numbers 5 and 11 constitutes the silicon region.

A.6.4. Contact Description

The CONTACT parameter specifies the location of contact nodes on the rectangular mesh. A specific sequence has to be used for each device. For the numerical diode the CONTACT model parameter is given in the sequence positive node, and negative node. For the numerical bipolar transistor CONTACT's are specified as collector, base, and emitter nodes. In the case of the numerical MOSFET the CONTACT sequence is drain, gate, source, and substrate nodes. The contact nodes to silicon are assumed to be ohmic, whereas the contact to an oxide region is assumed to be an aluminum/*n*-poly contact.

General form:

CONTACT XMESHLO XMESHHI YMESHLO YMESHHI

XMESHLO, XMESHHI, YMESHLO, and YMESHHI define the mesh numbers for the CONTACTS. These are the terminals at which the device is attached to the external circuit.

Example: Contacts for a MOSFET

CONTACT 28 31 5 5 CONTACT 5 27 1 1 CONTACT 1 4 5 5 CONTACT 1 31 19 19

The contacts are specified in the sequence drain, gate, source, and bulk (substrate). It is seen that contacts can be either horizontal or vertical consistent with the rectangular grid used for space discretization.

A.6.5. Level Parameter

The LEVEL model parameter is used to specify the dimensionality of the model. Specifying LEVEL=1 implies a one-dimensional device and LEVEL=2 specifies a two-dimensional device. The default value is 1 for numerical diodes and BJT models. Since the MOSFET is a two-dimensional device, it does not require a LEVEL specification, and LEVEL is an invalid parameter.

A.6.6. One-Carrier Simulations

For MOSFETs, simulation of one carrier-continuity equation provides good accuracy for most of the operating range. However, for conditions like avalanche generation and punchthru both carrier-continuity equations have to be solved. A flag can be used to specify the use of one-carrier simulation when it is known a priori that the MOSFET will be operating under normal conditions. The model parameter is denoted ONEC. By default both carrier-continuity equations are solved, and ONEC specifies that only one-carrier simulation should be performed. This model parameter is applicable only to numerical MOSFET models and is an invalid parameter for the other numerical devices.

A.7. Example Circuits

A.7.1. Circuit 1

The following deck determines the dc operating point of a simple RTL inverter circuit. The bipolar transistor is assumed to be a one-dimensional device with a uniform doping profile as shown in Figure A.2. A uniform grid is used for the simulations.

```
RTL INVERTER CIRCUIT
VIN 1 0 DC 4
```

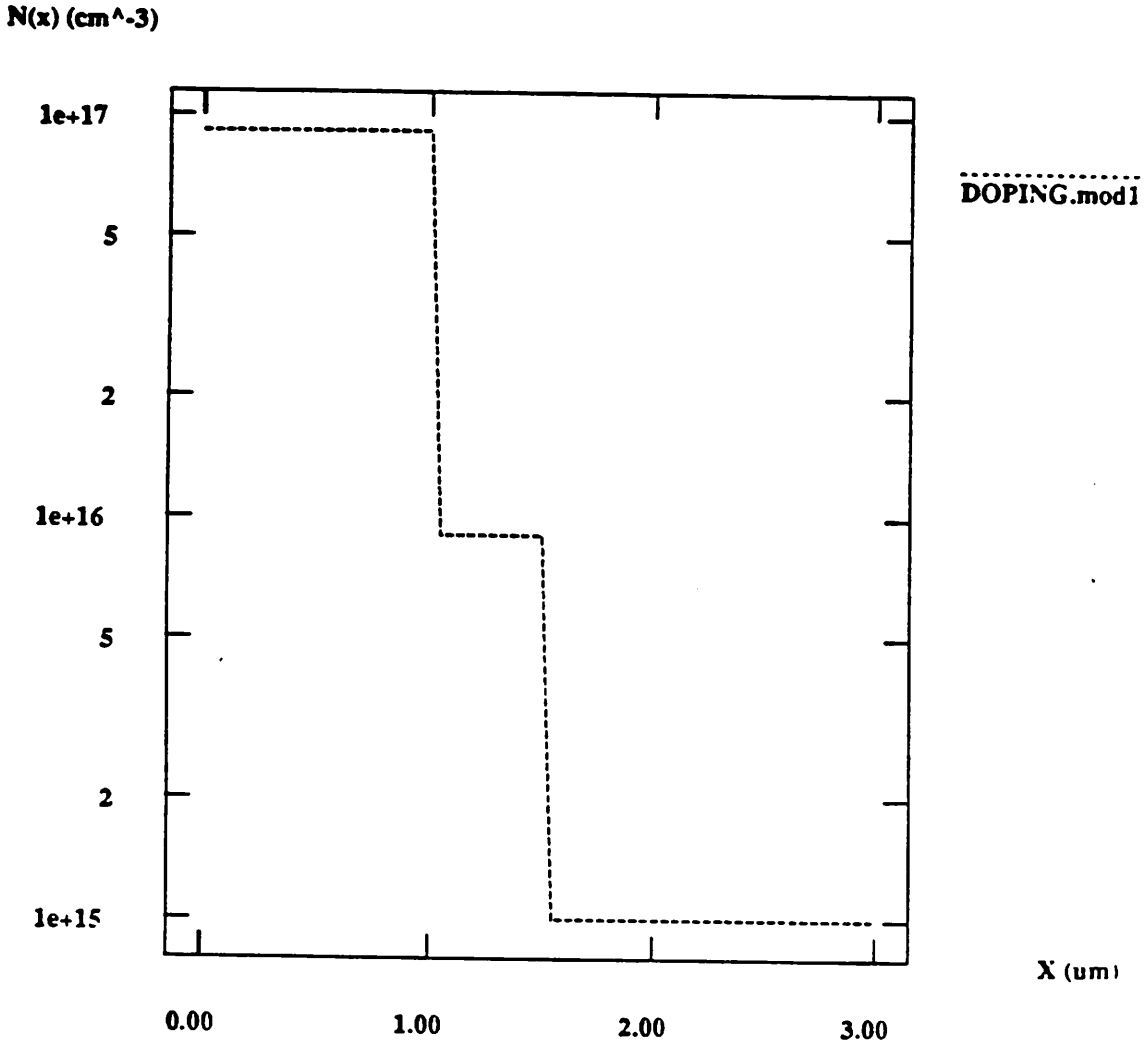


Figure A.2: BJT doping profile for Example 1.

```
VCC 12 0 DC 5.0
RC1 12 3 2.5K
RB1 1 2 8K
B1 3 2 0 MOD1 AREA = 1U
.MODEL MOD1 NBJT
+ NBGN = 1E17 BGNW SRH CONCTAU CONCMOB FIELDMOB AUGER
+ MESH 1 0 MESH 61 3
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
+ SILICON 1 61
.OP
.END
```

A.7.2. Circuit 2

The following deck computes the transient response of the pull-up circuit of a BiC-MOS driver. A realistic doping profile is used for the transistor as shown in Figure A.3. This is a typical example of mixed-level simulation with CODECS; the MOSFET is modeled by the Level-2 MOS model, whereas a numerical one-dimensional model is used for the bipolar transistor. Note the use of the .OPTION BYPASS = 1. This specification should be used with numerical models as it provides some speedup.

```
BiCMOS BUFFER
VDD 10 0 5
VIN 2 0 0
VC 10 1 0
VB 3 5 0
B1 1 5 4 MOD2 AREA = 0.2U
M1 3 2 10 10 MODP W = 60U L = 2U AD = 90P AS = 90P  PD = 61U PS = 61U
CL 4 0 10PF
* Level-2 MOS model
```

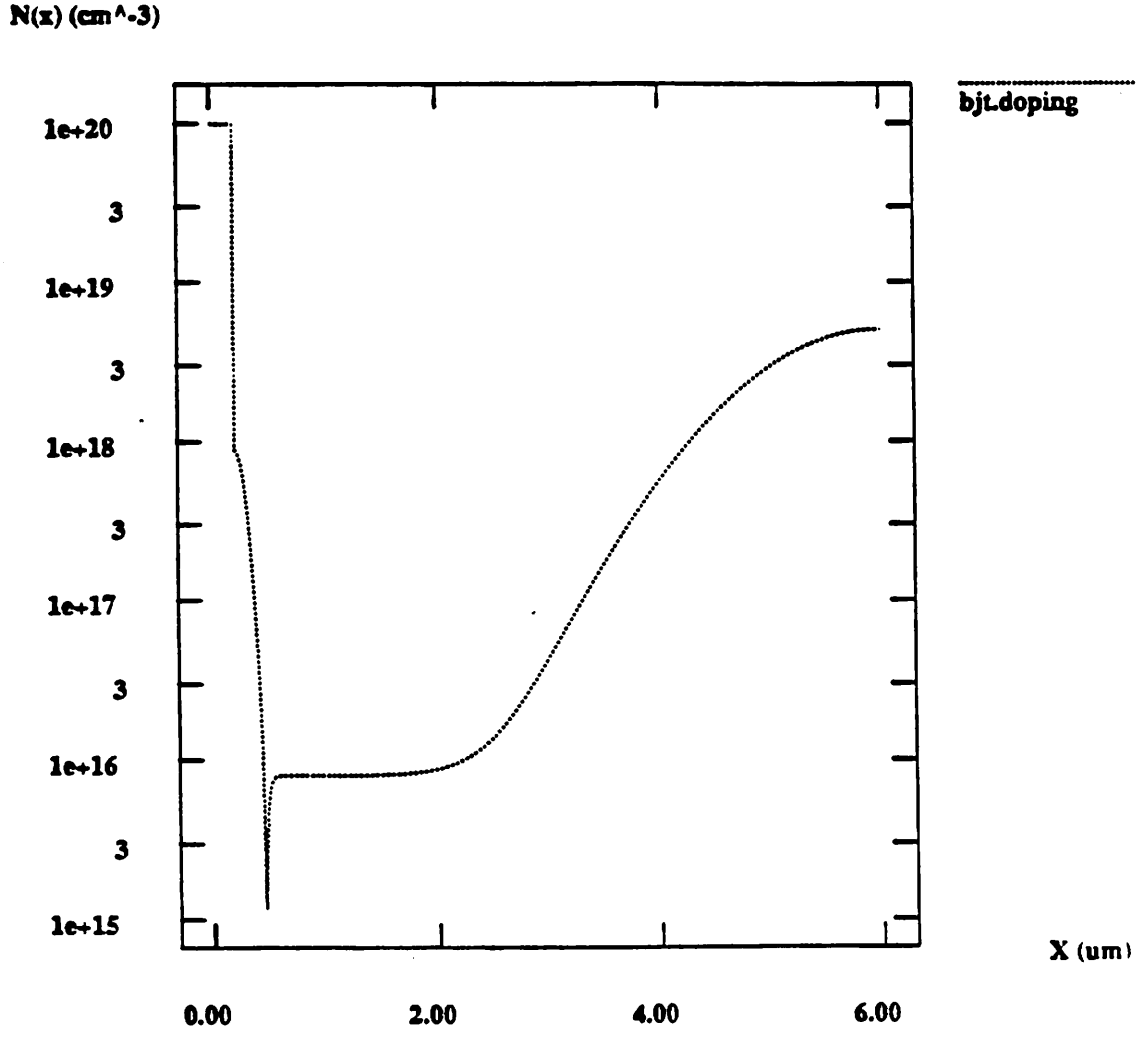


Figure A.3: BJT doping profile for Example 2.

```
.MODEL MODP PMOS VTO = -0.8 UO = 250 TOX = 25N NSUB = 5E16
+ UCRIT = 10K UEXP = .15 VMAX = 50K NEFF = 2 XJ = .02U
+ LD = .15U CGSO = .1N CGDO = .1N CJ = .12M MJ = 0.5
+ CJSW = 3E-10 MJSW = .5 LEVEL = 2
* numerical bjt model
.MODEL MOD2 NBJT
+ SRH BGNW CONCTAU CONCMOB FIELDMOB AUGER
+ NBGN = 1E17 TN0 = 100NS TP0 = 100NS
+ MESH 1 0 MESH 6 .1 MESH 36 .34 MESH 56 .6 MESH 146 6.0
+ UNIF 1E20 0 .2E-4
+ GIMP -9E17 .12E-4 .2E-4
+ UNIF 8E15 0 6E-4
+ GIMP 5E18 1.35E-4 6E-4
+ SILICON 1 146
.IC V(1) = 5 V(2) = 0 V(3) = 0 V(4) = 0.8 V(10) = 5 V(5) = 0
.TRAN 0.02N 5NS
.OPTION ACCT BYPASS=1
.PRINT TRAN V(3) V(4)
.END
```

A.7.3. Circuit 3

The following deck is used to simulate the transient response of an NMOS enhancement-load bootstrap inverter circuit. This is an example of two-dimensional numerical models. The only physical effects considered are concentration and field dependent mobilities. One-carrier simulation is used for the MOSFETs. The geometry and doping profile of the MOSFET are shown in Figures A.4(a) and A.4(b), respectively.

```
SIMULATION OF ENHANCEMENT LOAD INVERTER
VDD 1 0 5.0
VIN 5 0 PWL 0 0 1N 5 10N 5 11N 0 20N 0 21N 5 30N 5 31N 0
```

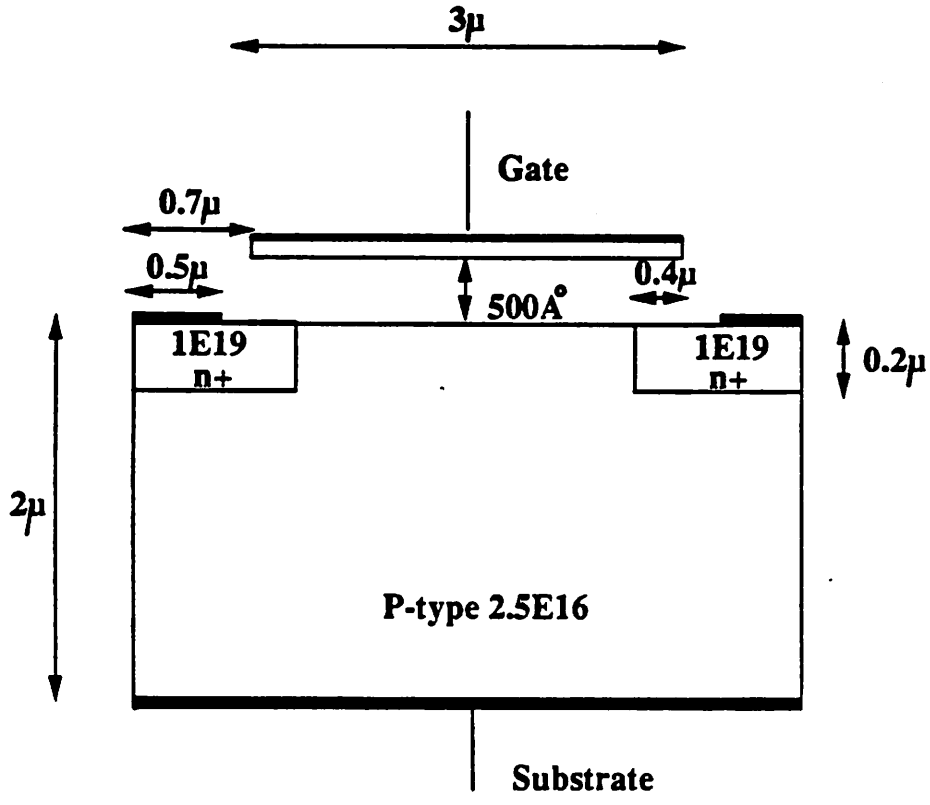


Figure A.4(a): Geometry of MOS transistor.

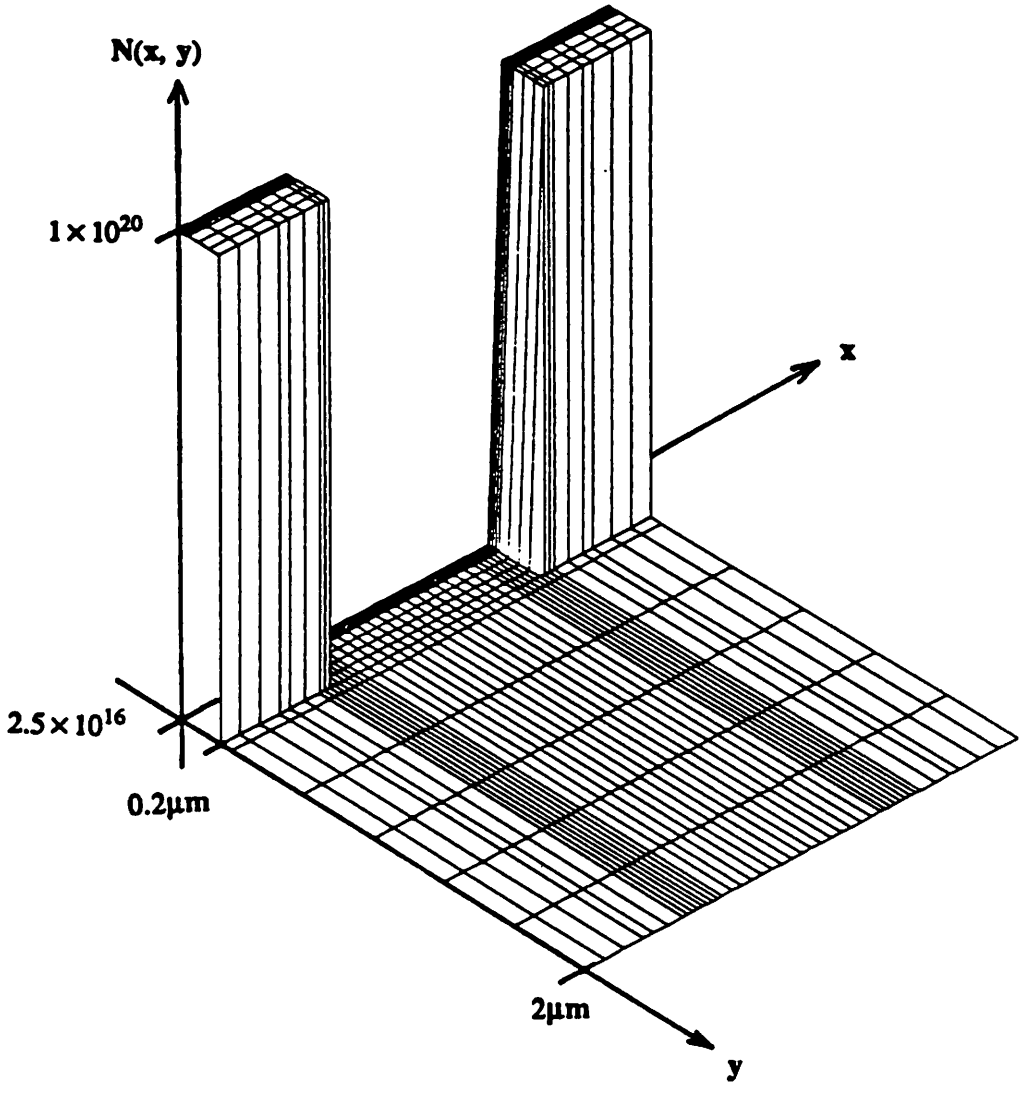


Figure A.4(b): Doping profile of MOS transistor.

```
N1 1 1 3 0 MOD1 WIDTH = 5E-4
N2 1 3 4 4 MOD1 WIDTH = 5e-4
N3 4 5 0 0 MOD1 WIDTH = 5E-4
CL 4 0 .1PF
.MODEL MOD1 NUMOS
+ CONCMOB FIELDMOB ONEC
+ XMESH 1 0 XMESH 4 0.6 XMESH 5 0.7
+ XMESH 7 1.0 XMESH 11 1.2 XMESH 21 3.2
+ XMESH 25 3.4 XMESH 27 3.7 XMESH 28 3.8 XMESH 31 4.4
+ YMESH 1 -.05 YMESH 5 0.0 YMESH 9 0.05 YMESH 14 .3 YMESH 19 2.0
+ UNIF 1E20 0 1.1E-4 0 .2E-4
+ UNIF 1E20 3.3E-4 4.4E-4 0 .2E-4
+ UNIF -2.5E16 0 4.4E-4 0 2E-4
+ UNIF -1E16 0 4.4E-4 0 .05E-4
+ OXIDE 5 27 1 5
+ SILICON 1 31 5 19
+ CONTACT 28 31 5 5
+ CONTACT 5 27 1 1
+ CONTACT 1 4 5 5
+ CONTACT 1 31 19 19
.TRAN .2NS 40NS
.OPTIONS ACCT BYPASS=1
.PRINT TRAN V(3) V(4)
.END
```

A.7.4. Circuit 4

The following deck is used to simulate the turnoff transient of a two-dimensional diode with an inductive load and a RC snubber circuit. Notice the use of the LEVEL parameter. The doping profile is shown in Figure A.5.

```
TWO-DIMENSIONAL PIN DIODE CIRCUIT
VIN 1 0 PWL 0 0.8 1n -50
```

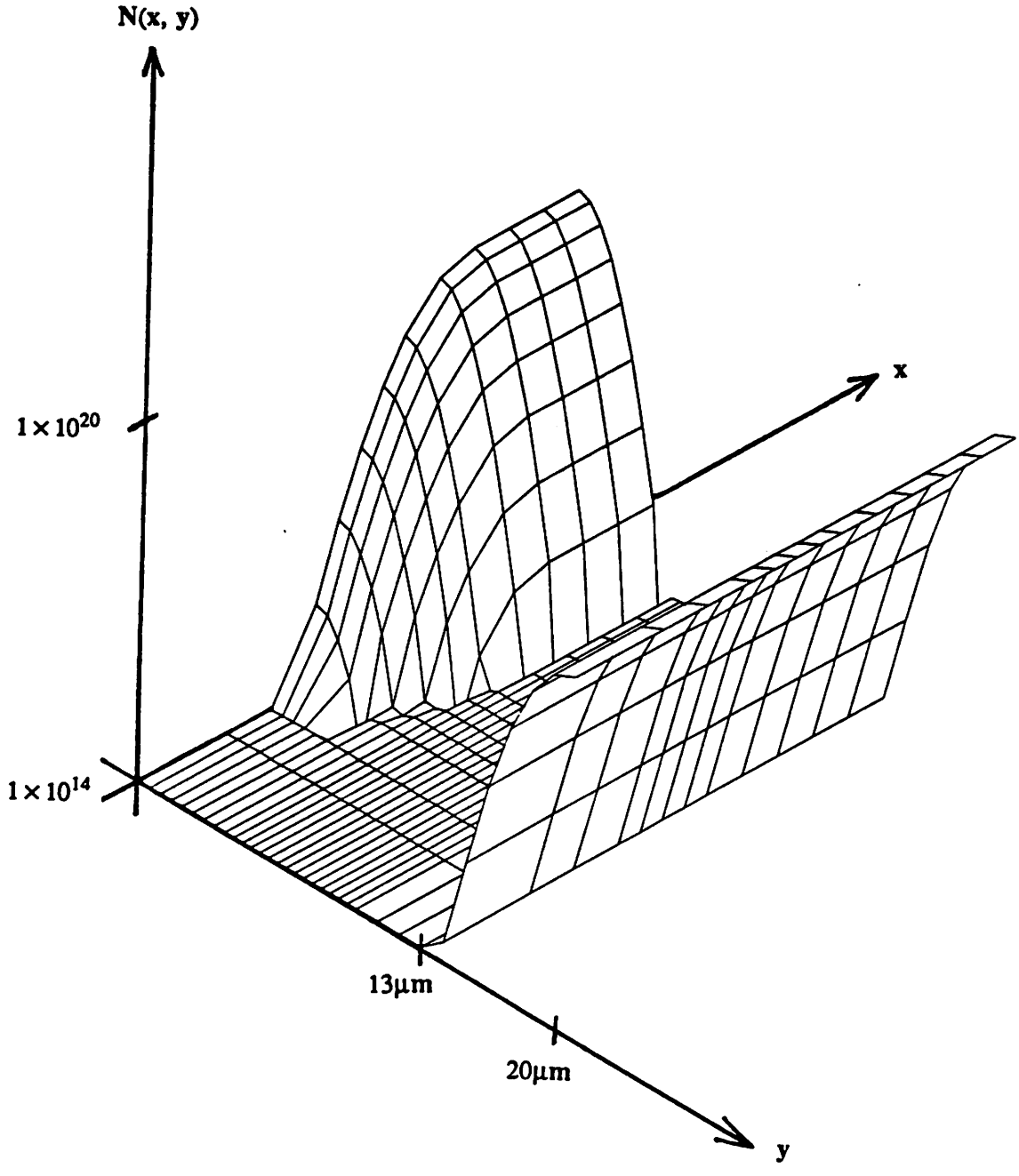


Figure A.5: Doping profile for Example 4.

```
L1 1 2 0.5UH
VD 2 3 DC 0
A1 3 0 MOD1 WIDTH = 0.2
VRC 2 4 DC 0
R1 4 5 100
C1 5 0 1NF
.MODEL MOD1 NUMD LEVEL = 2
+ TN0 = 20N TP0 = 20N SRH CONCTAU CONCMOB FIELDMOB AUGER
+ XMESH 1 0 XMESH 2 .2 XMESH 4 .4 XMESH 8 .6 XMESH 13 1.0
+ YMESH 1 0 YMESH 9 4 YMESH 24 10 YMESH 29 15 YMESH 34 20
+ GIMP -1E20 1.076E-4 0 0.1 1 0.75E-4 1E-4 0 0
+ UNIF 1E14 0 1E-4 0 20E-4
+ GIMP  1E20 1.614E-4 20E-4 0.8 1 0 1E-4 0 0
+ CONTACT 8 13 1 1
+ CONTACT 1 13 34 34
+ SILICON 1 13 1 34
.OPTION ACCT BYPASS=1
.TRAN 1N 100N
.PRINT TRAN V(3) I(VIN)
.END
```

APPENDIX B

CODECS Benchmark Circuits

```
RTL INVERTER CIRCUIT
VIN 1 0 DC 1 PWL 0 4 1N 0
VCC 12 0 DC 5.0
RC1 12 3 2.5K
RB1 1 2 8K
B1 3 2 0 MOD1 AREA = 1E-6
*NUMERICAL MODEL FOR BIPOLAR TRANSISTOR
.MODEL MOD1 NBJT BASE 1.25E-4
+ NBGN = 1E17 BGNW SRH CONCTAU CONCMOB FIELDMOB AUGER
+ MESH 1 0 MESH 61 3
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
+ SILICON 1 61
.OPTION ACCT BYPASS=1
.TRAN 0.5N 5N
*.OP
.PRINT TRAN V(3)
.END


COLPITT'S OSCILLATOR CIRCUIT
R1 1 0 1
B1 2 1 3 MOD1 AREA = 1E-6
VCC 4 0 5
RL 4 2 750
C1 2 3 500P
C2 4 3 4500P
```

```
L1 4 2 5UH
RE 3 6 4.65K
VEE 6 0 DC -15 PWL 0 -15 1N -10
.MODEL MOD1 NBJT BASE 1.25E-4
+ NBGN = 1E17 BGNW SRH AUGER CONCTAU CONCMOB FIELDMOB
+ MESH 1 0 MESH 61 3
+ SILICON 1 61
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
.OPTION ACCT BYPASS=1
.TRAN 30N 12U
*.OP
.PRINT TRAN V(2)
.END
```

VOLTAGE CONTROLLED OSCILLATOR

```
RC1 7 5 1K
RC2 7 6 1K
B5 7 7 5 MOD1 AREA = 1E-6
B6 7 7 6 MOD1 AREA = 1E-6
B3 7 5 2 MOD1 AREA = 1E-6
B4 7 6 1 MOD1 AREA = 1E-6
IB1 2 0 .5MA
IB2 1 0 .5MA
CB1 2 0 1PF
CB2 1 0 1Pf
B1 5 1 3 MOD1 AREA = 1E-6
B2 6 2 4 MOD1 AREA = 1E-6
C1 3 4 .1UF
IS1 3 0 DC 2.5MA PULSE 2.5MA 0.5MA 0 1US 1US 50MS
IS2 4 0 1MA
VCC 7 0 10
.MODEL MOD1 NBJT BASE 1.25E-4
```

```
+ SRH AUGER BGNW CONCTAU CONCMOB FIELDMOB NBGN = 1E17
+ MESH 1 0 MESH 61 3
+ SILICON 1 61
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
.OPTION ACCT BYPASS=1
.TRAN 3US 600US 0 3US
*.OP
.PRINT TRAN V(4)
.END


FOUR RTL INVERTER CHAIN
VIN 1 0 DC 0 PWL 0 0 1N 5
VCC 12 0 DC 5.0
RC1 12 3 2.5K
RB1 1 2 8K
B1 3 2 0 MOD1 AREA = 1E-6
RB2 3 4 8K
RC2 12 5 2.5K
B2 5 4 0 MOD1 AREA = 1E-6
RB3 5 6 8K
RC3 12 7 2.5K
B3 7 6 0 MOD1 AREA = 1E-6
RB4 7 8 8K
RC4 12 9 2.5K
B4 9 8 0 MOD1 AREA = 1E-6
.MODEL MOD1 NBJT BASE = 1.25E-4
+ NBGN = 1E17 BGNW SRH AUGER CONCTAU CONCMOB FIELDMOB
+ MESH 1 0 MESH 61 3.0
+ SILICON 1 61
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
```

```
.OPTION ACCT BYPASS=1
*.OP
.PRINT TRAN V(3) V(5) V(9)
.TRAN 1N 10N
.END


ASTABLE MULTIVIBRATOR
VIN 5 0 DC 0 PULSE( 0 5 0 1US 1US 100US 100US )
VCC 6 0 5.0
RC1 6 1 1K
RC2 6 2 1K
RB1 6 3 30K
RB2 5 4 30K
C1 1 4 150PF
C2 2 3 150PF
B1 1 3 0 MOD1 AREA = 1E-6
B2 2 4 0 MOD1 AREA = 1E-6
.MODEL MOD1 NBJT BASE = 1.25E-4
+ NBGN = 1E17 BGNW SRH CONCTAU CONCMOB FIELDMOB AUGER
+ MESH 1 0 MESH 61 3
+ SILICON 1 61
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
.OPTION ACCT BYPASS=1
.TRAN .05US 8US 0 .05Us
*.OP
.PRINT TRAN V(1) V(2) V(3) V(4)
.END


MECLIII CKT - MOTOROLA MECL III ECL GATE
.TRAN 0.2NS 20NS
*.OP
```

```
VEE 22 0 -6.0
VIN 1 0 PULSE -0.8 -1.8 0.2NS 0.2NS 0.2NS 10NS 20NS
RS 1 2 50
B1 4 2 6 QNUM AREA = 1E-6
B2 4 3 6 QNUM AREA = 1E-6
B3 5 7 6 QNUM AREA = 1E-6
B4 0 8 7 QNUM AREA = 1E-6
D1 8 9 DMOD
D2 9 10 DMOD
RP1 3 22 50K
RC1 0 4 100
RC2 0 5 112
RE 6 22 380
R1 7 22 2K
R2 0 8 350
R3 10 22 1958
B5 0 5 11 QNUM AREA = 1E-6
B6 0 4 12 QNUM AREA = 1E-6
RP2 11 22 560
RP3 12 22 560
B7 13 12 15 QNUM AREA = 1E-6
B8 14 16 15 QNUM AREA = 1E-6
RE2 15 22 380
RC3 0 13 100
RC4 0 14 112
B9 0 17 16 QNUM AREA = 1E-6
R4 16 22 2K
R5 0 17 350
D3 17 18 DMOD
D4 18 19 DMOD
R6 19 22 1958
B10 0 14 20 QNUM AREA = 1E-6
B11 0 13 21 QNUM AREA = 1E-6
RP4 20 22 560
RP5 21 22 560
```

```
.MODEL DMOD D RS=40 TT=0.1NS CJO=0.9PF N=1 IS=1E-14
+ EG=1.11 VJ = 0.8 M=0.5
.MODEL QNUM NBJT BASE 1.25E-4
+ NBGN = 1E17 BGNW SRH AUGER CONCTAU CONCMOB FIELDMOB
+ MESH 1 0 MESH 10 0.9 MESH 20 1.1 MESH 30 1.4 MESH 40 1.6 MESH 61 3.0
+ SILICON 1 61
+ UNIF 1E17 0 1E-4
+ UNIF -1E16 0 1.5E-4
+ UNIF  1E15 0 5E-4
.OPTIONS ACCT BYPASS=1
.PRINT TRAN V(12) V(21)
.END


TURNOFF TRANSIENT OF PASS TRANSISTOR
N1 11 2 3 4 NCH WIDTH=20E-4
CS 1 0 6.0PF
CL 3 0 6.0PF
R1 3 6 200K
VIN 6 0 DC 0
VDRN 1 11 DC 0
VG 2 0 DC 5 PWL 0 5 0.1N 0 1 0
VB 4 0 DC 0.0
.TRAN 0.05N .2N 0 .05N
.PRINT TRAN V(1) I(VDRN)
.IC V(1)=0 V(3)=0
.OPTION ACCT BYPASS=1
.MODEL NCH  NUMOS
+ CONCMOB FIELDMOB ONEC
+ XMESH 1 0 XMESH 4 0.6 XMESH 5 0.7
+ XMESH 7 1.0 XMESH 11 1.2 XMESH 21 3.2
+ XMESH 25 3.4 XMESH 27 3.7 XMESH 28 3.8 XMESH 31 4.4
+ YMESH 1 -.05 YMESH 5 0.0 YMESH 9 0.05 YMESH 14 .3 YMESH 19 2.0
+ UNIF 1E20 0 1.1E-4 0 .2E-4
+ UNIF 1E20 3.3E-4 4.4E-4 0 .2E-4
```

```
+ UNIF -2.5E16 0 4.4E-4 0 2E-4
+ UNIF -1E16 0 4.4E-4 0 .05E-4
+ OXIDE 5 27 1 5
+ SILICON 1 31 5 19
+ CONTACT 28 31 5 5
+ CONTACT 5 27 1 1
+ CONTACT 1 4 5 5
+ CONTACT 1 31 19 19
.END


MOS INVERTER CIRCUIT
VIN 1 0 PWL 0 0.0 2N 5
VDD 3 0 DC 5.0
RD 3 2 2.5K
N1 2 1 4 5 MOD1 WIDTH = 1E-3
CL 2 0 2PF
VB 5 0 0
VS 4 0 0
.MODEL MOD1 NUMOS
+ CONCMOB FIELDMOB ONEC
+ XMESH 1 0 XMESH 4 0.6 XMESH 5 0.7 XMESH 7 1.0 XMESH 11 1.2
+ XMESH 21 3.2 XMESH 25 3.4 XMESH 27 3.7 XMESH 28 3.8 XMESH 31 4.4
+ YMESH 1 -.05 YMESH 5 0.0 YMESH 9 0.05 YMESH 14 .3 YMESH 19 2.0
+ UNIF 1E20 0 1.1E-4 0 .2E-4
+ UNIF 1E20 3.3E-4 4.4E-4 0 .2E-4
+ UNIF -2.5E16 0 4.4E-4 0 2E-4
+ UNIF -1E16 0 4.4E-4 0 .05E-4
+ OXIDE 5 27 1 5
+ SILICON 1 31 5 19
+ CONTACT 28 31 5 5
+ CONTACT 5 27 1 1
+ CONTACT 1 4 5 5
+ CONTACT 1 31 19 19
.TRAN .2NS 30NS
```

```
.IC V(1) = 0.0 V(2) = 5.0 V(3)=5.0 V(4)=0.0 V(5)=0.0
.OPTIONS ACCT BYPASS=1
.PRINT TRAN V(1) V(2)
.END


CHARGE PUMP CIRCUIT
VIN 4 0 PULSE 0 5 15NS 5NS 5NS 50NS 100NS
VDD 5 6 PULSE 0 5 25NS 5NS 5NS 50NS 100NS
VBB 0 7 PULSE 0 5 0 5NS 5NS 50NS 100NS
RD 6 2 10K
N1 5 4 3 7 MOD1 WIDTH = 1E-2
VS 3 2 0
VC 2 1 0
C2 1 0 10PF
.MODEL MOD1 NUMOS
+ CONCMOB FIELDMOB ONEC
+ XMESH 1 0 XMESH 3 0.4 XMESH 7 0.6 XMESH 15 1.4
+ XMESH 19 1.6 XMESH 21 2.0
+ YMESH 1 0 YMESH 4 0.015 YMESH 8 0.05 YMESH 12 0.25
+ YMESH 14 0.35 YMESH 17 0.5 YMESH 21 1.0
+ UNIF 1E18 0 0.5E-4 0.015E-4 .25E-4
+ UNIF 1E18 1.5E-4 2.0E-4 0.015E-4 0.25E-4
+ UNIF -1E15 0 2E-4 0.015E-4 1E-4
+ UNIF -1.3E17 0.5E-4 1.5E-4 0.015E-4 .05E-4
+ OXIDE 5 17 1 4
+ SILICON 1 21 4 21
+ CONTACT 18 21 4 4
+ CONTACT 5 17 1 1
+ CONTACT 1 4 4 4
+ CONTACT 1 21 21 21
.IC V(4) = 0 V(3) = 1.0 V(2) = 1.0 V(1) = 1.0 V(5)=1.0 V(7)=0.0 V(6)=1.0
*.OP
.TRAN 2NS 200NS
.OPTIONS ACCT BYPASS=1
```

```
.PRINT TRAN V(1) V(2)
.END
```

APPENDIX C

Source Listing of CODECS

The source listing of CODECS is available at the following address:

Software Distribution Office

Industrial Liaison Program

Department of Electrical Engineering and Computer Sciences

University of California

Berkeley, CA 94720

REFERENCES

Chapter 1

[1.1] R. W. Dutton, "Modeling of the Silicon Integrated-Circuit Design and Manufacturing Process," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 968-986, Sept. 1983.

[1.2] A. R. Newton, "The Simulation of Large-Scale Integrated Circuits," *Memo. No. UCB/ERL M78/52*, Electronics Research Laboratory, University of California, Berkeley, July 1978.

[1.3] H. DeMan, G. Arnout, and P. Reynaert, "Mixed-Mode Circuit Simulation Techniques and their Implementation in DIANA," *NATO Advanced Study Institute on CAD for VLSI Circuits*, Sogesto-Urbino, Italy, July 1980.

[1.4] K. A. Sakallah, "Mixed Simulation of Electronic Integrated Circuits," *Report No. DRC-02-07-81*, Carnegie-Mellon University, Pittsburgh, Nov. 1981.

[1.5] J. E. Kleckner, "Advanced Mixed-Mode Simulation Techniques," *Memo. No. UCB/ERL M84/48*, Electronics Research Laboratory, University of California, Berkeley, June 1984.

[1.6] W. L. Engl, R. Laur, and H. K. Dirks, "MEDUSA - A Simulator for Modular Circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-1, pp. 85-93, April 1982.

[1.7] M. S. Mock, "Time-Dependent Simulation of Coupled Devices," *Proc. NASECODE II*, Dublin, Ireland, pp. 113-131, June 1981.

[1.8] J. G. Rollins and J. Choma, "Mixed-Mode PISCES-SPICE Coupled Circuit and Device Solver," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 862-

867, Aug. 1988.

[1.9] C. H. Price, "Two-Dimensional Numerical Simulation of Semiconductor Devices," *Ph. D. Dissertation*, Stanford University, Stanford, 1982.

Chapter 2

[2.1] L. O. Chua, C. A. Desoer, E. S. Kuh, *Linear amd Nonlinear Circuits*, McGraw-Hill Book Company, NY 1987.

[2.2] W. T. Weeks, A. J. Jimenez, G. W. Malhoney, D. Mehta, H. Qassemzadeh, and T. R. Scott, "Algorithms for ASTAP - A Network Analysis Program," *IEEE Trans. Circuit Theory*, vol. CT-20, pp. 628-634, Nov. 1973.

[2.3] L. W. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits", *Memo. No. ERL-M520*, Electronics Research Laboratory, University of California, Berkeley, May 1975.

[2.4] C. W. Ho., A. E. Ruehli, and P. A. Brennan, "The Modified Nodal Approach to Network Analysis", *IEEE Trans. Circuits and Systems* , vol. CAS-22, pp. 504-509, June 1975.

[2.5] T. Quarles, A. R. Newton, D. O. Pederson, and A. Sangiovanni-Vincentelli, *SPICE 3B1 User's Guide*, Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, April 1988.

[2.6] A. L. Sangiovanni-Vincentelli, "Circuit Simulation", in *Computer Design Aids for VLSI Circuits*, P. Antognetti, D. O. Pederson, and H. De Man (Editors), Groningen, The Netherlands: Sijthoff and Noordhoff, pp. 19-113, 1981.

[2.7] L. O. Chua and P. M. Lin *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*, Prentice Hall Inc., New Jersey, 1975.

[2.8] W. J. McCalla, *Computer-Aided Circuit Simulation Techniques*, Kluwer Publishers, 1987.

[2.9] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas," *IEEE Proc.*, vol. 60, pp. 98-108, Jan. 1972.

[2.10] R. Saleh, "Nonlinear Relaxation Algorithms for Circuit Simulation," *Memo No. UCB/ERL M87/21*, Electronics Research Laboratory, University of California, Berkeley, April 1987.

[2.11] E. Lelarasmee, "The Waveform Relaxation Method for Time Domain Analysis of Large Scale Integrated Circuits: Theory and Applications," *Memo No. UCB/ERL M82/40*, Electronics Research Laboratory, University of California, Berkeley, May 1982.

[2.12] J. White, "The Multirate Integration Properties of Waveform Relaxation, with Application to Circuit Simulation and Parallel Computation," *Memo No. UCB/ERL 85/90*, Electronics Research Laboratory, University of California, Berkeley, Nov. 1985.

[2.13] G. Marong and A. Sangiovanni-Vincentelli, "Waveform Relaxation and Dynamic Partitioning for Transient Simulation of Large Scale Bipolar Circuits," *Proc. IEEE ICCAD-85*, Santa Clara, CA, pp. 32-34, Nov. 1985.

[2.14] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*, Van Nostrand Reinhold Company, 1983.

[2.15] I. Getreu, *Modeling The Bipolar Transistor*, Tektronix, 1976.

[2.16] H. Shichman and D. A. Hodges, "Modeling and Simulation of Insulated-Gate Field-Effect Transistor Switching Circuits", *IEEE J. Solid-State Circuits*, vol. SC-3, pp. 285-289, Sept. 1968.

[2.17] H. K. Gummel and H. C. Poon, "An Integral Charge Control Model of Bipolar Transistors," *Bell Syst. Tech. J.*, pp. 827-852, May 1970.

[2.18] L. J. Turgeon and J. R. Mathews, "A Bipolar Transistor Model of Quasi-Saturation for use in Computer-Aided Design (CAD)," *IEDM Digest of Tech. Papers*, Washington D.C. Dec. 1980.

[2.19] G. M. Kull, L. W. Nagel, S. W. Lee, P. LLoyd, E. J. Prendergast, and H. Dirks, "A Unified Circuit Model for Bipolar Transistors Including Quasi-Saturation Effects," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 1103-1113, June 1985.

[2.20] G. Merckel, "CAD Models of MOSFETS," in *Process and Device Modeling for Integrated Circuit Design*, F. Van de Wiele, W. L. Engl, and P. G. Jespers (Editors), Noordhoff, Leyden, The Netherlands, 1977.

[2.21] L. M. Dang, "A Simple Current Model for Short-Channel IGFET and its Application to Circuit Simulation," *IEEE J. Solid-State Circuits*, vol. SC-14, pp. 383-391, April 1979.

[2.22] H. I. Hanafi, L. H. Camnitz, and A. J. Dally, "An Accurate and Simple MOSFET Model for Computer-Aided Design," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 5, pp. 882-891, Oct. 1982.

[2.23] A. Vladimirescu and S. Liu, "The Simulation of MOS Integrated Circuits using SPICE2," *Memo No. UCB/ERL M80/7*, Electronics Research Laboratory, University of California, Berkeley, Feb. 1980.

[2.24] S. Liu and L. W. Nagel, "Small-Signal MOSFET Models for Analog Circuit Design," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 993-998, Dec. 1982.

[2.25] G. T. Wright, "Simple and Continuous MOSFET Models for Computer-Aided Design of VLSI," *IEE Proc.*, vol. 132 part I, pp. 187-194, Aug. 1985.

[2.26] Y. A. Elmansy and A. R. Boothroyd, "A New Approach to the Theory and Modeling of Insulated-Gate Field-Effect Transistors," *IEEE Trans. Electron Devices*, vol. ED-24, pp. 241-253, March 1977.

[2.27] G. Baccarani, M. Rudan, and G. Spadini, "Analytical i.g.f.e.t Model Including Drift and Diffusion Currents," *IEE J. Solid-State and Electron Devices*, vol. 2, pp. 62-68, March 1978.

[2.28] J. R. Brews, "A Charge Sheet Model for the MOSFET," *Solid-State Electronics*, vol. 21, pp. 345-355, 1978.

[2.29] C. Turchetti and G. Masetti, "A CAD-Oriented Analytical MOSFET Model for High-Accuracy Applications," *IEEE Trans. Computer-Aided Design*, vol. CAD-3, pp. 117-122, July 1984.

[2.30] M. H. White, F Van de Wiele, and J. P. Lambot, "High-Accuracy MOS Models for Computer-Aided Design," *IEEE Trans. Electron Devices*, vol. ED-27, pp. 899-906, May 1980.

[2.31] P. Yang, B. D. Epler, P. K. Chatterjee, "An Investigation of Charge Conservation Problem for MOSFET Circuit Simulation," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 128-138, Feb. 1983.

[2.32] B. J. Sheu, "MOS Transistor Modeling and Characterization for Circuit Simulation," *Memo No. UCB/ERL M85/85*, Electronics Research Laboratory, University of California, Berkeley, Oct. 1985.

[2.33] M. Bagheri and Y. Tsividis, "A Small-Signal Dc-to-High Frequency Nonquasistatic Model for the Four-Terminal MOSFET Valid in All Regions of Operation," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 2383-2391, Nov. 1985.

[2.34] C. Turchetti, P. Mancini, G. Masetti, "A CAD-Oriented Non-Quasi-Static Approach for Transient Analysis of MOS IC's," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 827-835, Oct. 1986.

[2.35] H. J. Park, P. K. Ko, and C. Hu, "A Non-Quasistatic MOSFET Model for SPICE," *IEDM Digest of Tech. Papers*, pp. 652-655, Dec. 1987.

[2.36] S. W. Lee and R. C. Rennick, "A Compact IGFET Model - ASIM," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 952-975, Sept. 1988.

[2.37] E. Khalily, P. H. Decher, D. A. Teegarden, "TECAP2: An Interactive Device Characterization and Model Development System," *Proc. IEEE ICCAD-84*, Santa Clara, CA, pp. 149-151, Nov. 1984.

[2.38] K. Doganis and D. L. Scharfetter, "General Optimization and Extraction of IC Device Models," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1219-1228, Sept. 1983.

[2.39] J. L. D'Arcy and R. C. Rennick, "MOSFET Parameter Optimization for Accurate Output Conductance Modeling," *Proc. 1985 IEEE CICC* , Portland, OR, pp. 512-513, May 1985.

[2.40] C. G. Sodini, P. K. Ko, and J. L. Moll, "The Effect of High Fields on MOS Device and Circuit Performance," *IEEE Trans. Electron Devices*, vol. ED-31, pp. 1386-1393, Oct. 1984.

[2.41] S. Ogura, P. J. Tsang, W. W. Walker, D. L. Critchlow, and J. F. Shepard, "Design and Characteristics of the Lightly Doped Drain-Source (LDD) Insulated Gate Field-Effect Transistor," *IEEE Trans. Electron Devices*, vol. ED-27, pp. 1359-1367, Aug. 1980.

[2.42] G. J. Hu, C. Chang, and Y. Chia, "Gate-Voltage-Dependent Effective Channel Length and Series Resistance of LDD MOSFET's," *IEEE Trans. Electron*

Devices, vol. ED-34, pp. 2469-2475, Dec. 1987.

[2.43] G. Huang and C. Wu, "An Analytic I-V Model for Lightly Doped Drain (LDD) MOSFET devices," *IEEE Trans. Electron Devices*, vol. ED-34, pp. 1311-1322, June 1987.

[2.44] D. E. Ward and R. W. Dutton, "A Charge-Oriented Model for MOS Transistor Capacitances," *IEEE J. Solid State Circuits*, vol. SC-13, pp. 703-707, Oct. 1978.

[2.45] S. E. Laux, "Techniques for Small-Signal Analysis of Semiconductor Devices," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 2028-2037, Oct. 1985.

[2.46] B. R. Chawla and H. K. Gummel, "Transition Region Capacitance of Diffused p-n Junctions," *IEEE Trans. Electron Devices*, vol. ED-18, pp. 178-195, March 1971.

[2.47] S. W. Lee and E. J. Prendergast, "Analytical Relations for Determining the Base Transit Times and Forward-Biased Junction Capacitances of Bipolar Transistors," *Solid-State Electronics*, vol. 28, pp. 767-773, 1985.

[2.48] R. S. Muller and T. I. Kamins, *Device Electronics for Integrated Circuits*, John Wiley and Sons, New York, 1986.

[2.49] S. M. Sze, "Physics of Semiconductor Devices," John Wiley, 1981.

[2.50] S. K. Ghandhi, *Power Semiconductor Devices*, John Wiley and Sons, 1977.

[2.51] T. Shima, T. Sugawara, S. Moriyama, and H. Yamada, "Three-Dimensional Table Look-Up MOSFET Model for Precise Circuit Simulation," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 449-454, June 1982.

[2.52] J. L. Burns, "Empirical MOSFET Models for Circuit Simulation," *Memo No. UCB/ERL M84/43*, Electronics Research Laboratory, University of California, Berkeley, May 1984.

[2.53] J. Barby, J. Vlach, and K. Singhal, "Optimized Polynomial Splines for FET Models," *Proc. IEEE ISCAS*, Montreal, Canada, pp. 1159-1162, May 1984.

[2.54] T. Shima, H. Yamada and R. Dang, "Table Look-up MOSFET Modeling System Using a 2-Dimensional Device Simulator and Monotonic Piecewise Cubic Interpolation," *IEEE Trans. Computer-Aided Design*, vol. CAD-2, pp. 121-126, April 1983.

[2.55] R. S. Gyurcsik, "An Attached Processor for MOS-Transistor Model Evaluation," *Memo No. UCB/ERL M86/82*, Electronics Research Laboratory, University of California, Berkeley, Oct. 1986.

[2.56] E. M. Sentovich, "B-spline Device Models in SPICE3,", *Master's Report*, University of California, Berkeley, 1988.

[2.57] T. Shima, "Device and Circuit Simulator Integration Techniques," in *Process and Device Modeling*, W. L. Engl (Editor), Elsevier Science Publishers B. V. (North-Holland) pp. 433-459, 1986.

[2.58] T. Shima, "Table Lookup MOSFET Capacitance Model for Short Channel Devices," *IEEE Trans. Computer-Aided Design*, vol. CAD-5, pp. 624-632, Oct. 1986.

[2.59] H. K. Dirks and K. Eickhoff, "Numerical Models and Table Models for MOS Circuit Analysis," *Proc. NASECODE IV*, Dublin, Ireland, pp. 13-24, June 1985.

[2.60] A. F. Lachner, "Two-Dimensional Integral Analysis of Bipolar Junction Transistors," *Ph. D. dissertation*, University of California, Berkeley, 1982.

[2.61] D. J. Roulston, S. G. Chamberlain, and J. Sehgal, "Simplified Computer-Aided Analysis of Double-Diffused Transistor Including Two-Dimensional High-Level Effects," *IEEE Trans. Electron Devices*, vol. ED-19, pp. 809-820,

June 1972.

[2.62] V. Marash and R. W. Dutton, "Methodology for Submicron Device Model Development," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 299-306, Feb. 1988.

[2.63] M. J. Saccamango, "Efficient Modeling of Small-Geometry MOSFET's," *Research Report No. CMUCAD-87-11*, Carnegie-Mellon University, Pittsburgh, Sep. 1987.

[2.64] H. J. DeMan and R. Mertens, "SITCAP - A Simulator of Bipolar Transistors for Computer-Aided Circuit Analysis Programs," *Proc. ISSCC-73*, pp. 104-105, Feb. 1973.

[2.65] W. L. Engl, R. Laur, and H. K. Dirks, "MEDUSA - A Simulator for Modular Circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-1, pp. 85-93, April 1982.

Chapter 3

[3.1] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*, Springer-Verlag, New York, 1984.

[3.2] G. Baccarani, M. Rudan, R. Guerrieri, and P. Ciampolini, "Physical Models for Numerical Device Simulation," in *Process and Device Modeling*, W. L. Engl (Editor), Elsevier Science Publishers B. V. (North-Holland) pp. 107-158, 1986.

[3.3] H. K. Gummel, "A Self-Consistent Iterative Scheme for One-Dimensional Steady State Transistor Calculations,"*IEEE Trans. Electron Devices*, vol. ED-11, pp. 445-465, Oct. 1964.

[3.4] D. L. Scharfetter and H. K. Gummel, "Large-Signal Analysis of a Silicon Read Diode Oscillator," *IEEE Trans. Electron Devices*, vol. ED-16, pp. 64-77, Jan. 1969.

[3.5] A. DeMari, "An Accurate Numerical Steady-State One-Dimensional Solution of the P-N Junction," *Solid-State Electronics*, vol. 11, pp. 33-58, Jan. 1968.

[3.6] A. DeMari, "An Accurate Numerical One-Dimensional Solution of the P-N Junction Under Arbitrary Transient Conditions," *Solid-State Electronics*, vol. 11, pp. 1021-1053, Nov. 1968.

[3.7] C. H. Price, "Two-Dimensional Numerical Simulation of Semiconductor Devices," *Ph. D. Dissertation*, Stanford University, Stanford, 1982.

[3.8] G. Baccarani, R. Guerrieri, P. Ciampolini, and M. Rudan, "HFIELDS: A Highly Flexible 2-D Semiconductor-Device Analysis Program," *Proc. NASECODE-IV*, Dublin, Ireland, pp. 3-12, June 1985.

[3.9] A. F. Franz, G. A. Franz, and S. Selberherr, "BAMBI - A Design Model for Power MOSFET's,' *Proc. IEEE ICCAD-84*, pp. 179-181, Nov. 1984.

[3.10] W. L. Engl, H. K. Dirks, and B. Meinerzhagen, "Device Modeling," *Proc. IEEE*, vol. 71, pp. 10-33, Jan. 1983.

[3.11] E. M. Buturla, P. E. Cottrell, B. M. Grossman, and K. A. Salsburg, "Finite-Element Analysis of Semiconductor Devices: The FIELDAY program," *IBM J. Res. Develop.*, vol. 25, pp. 218-231, 1981.

[3.12] T. Toyabe, K. Yamaguchi, S. Asai, and M. S. Mock, "A Numerical Model for Avalanche Breakdown in MOSFET's," *IEEE Trans. Electron Devices*, vol. ED-25, pp. 825-832, July 1978.

[3.13] E. M. Buturla, P. E. Cottrell, B. M. Grossman, K. A. Salsburg, M. B. Lawlor, C. T. McMullen, "Three-Dimensional Finite-Element Simulation of

Semiconductor Devices," *Proc. ISSCC-80*, pp. 76-77, Feb. 1980.

[3.14] A. Yoshii, H. Kitazawa, M. Tomizawa, S. Horiguchi, and T. Sudo, "A Three-Dimensional Analysis of Semiconductor Devices," *IEEE Trans. Electron Devices*, vol. ED-29, pp. 184-189, Feb. 1982.

[3.15] A. Husain and S. G. Chamberlain, "Three-Dimensional Simulation of VLSI MOSFET's: The Three-Dimensional Simulation Program WATMOS," *IEEE Trans. Electron Devices*, vol. ED-29, pp. 631-638, April 1982.

[3.16] T. Toyabe, H. Masuda, Y. Aoki, H. Shukuri, and T. Hagiwara, "Three-Dimensional Device Simulator CADDETH with Highly Convergent Matrix Solution Algorithms," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 2038-2044, Oct. 1985.

[3.17] D. M. Caughey and R. E. Thomas, "Carrier Mobilities in Silicon Empirically Related to Doping and Field," *Proc. IEEE*, vol. 52, pp. 2192-2193, Dec. 1967.

[3.18] N. D. Arora, J. R. Hauser, and D. J. Roulston, "Electron and Hole Mobilities in Silicon as a Function of Concentration and Temperature," *IEEE Trans. Electron Devices*, vol. ED-29, pp. 292-295, Feb. 1982.

[3.19] K. Yamaguchi, "Field-Dependent Mobility Model for Two-Dimensional Numerical Analysis of MOSFET's," *IEEE Trans. Electron Devices*, vol. ED-26, pp. 1068-1074, July 1979.

[3.20] K. Yamaguchi, "A Mobility Model for Carriers in the MOS Inversion Layer," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 658-663, June 1983.

[3.21] S. Selberherr, A. Schutz, H. W. Potzl, "MINIMOS - a Two-Dimensional MOS Transistor Analyzer," *IEEE Trans. Electron Devices*, vol. ED-27, pp. 1540-1550, Aug. 1980.

[3.22] M. Eldredge, "Structured PISCES," *Annual Research Summary: Process and Device Modeling*, Stanford Electronics Laboratories, Stanford University, Stanford, 1988.

[3.23] J. T. Watt and J. D. Plummer, "Universal Mobility-Field Curves for Electrons and Holes in MOS Inversion Layers," *Digest of Tech. papers 1987 Symp. VLSI Technology*, pp. 82-83, May 1987.

[3.24] R. S. Muller and T. I. Kamins, *Device Electronics for Integrated Circuits*, John Wiley and Sons, New York, 1986.

[3.25] D. J. Roulston, N. D. Arora, and S. G. Chamberlain, "Modeling and Measurement of Minority-Carrier Lifetime versus Doping in Diffused Layers of n^+-p Silicon Diode," *IEEE Trans. Electron Devices*, vol. ED-29, pp. 284-291, Feb. 1982.

[3.26] A. G. Chynoweth, "Ionization Rates for Electrons and Holes in Silicon," *Physical Review*, vol. 109, pp. 1537-1540, 1958.

[3.27] A. Schutz, S. Selberherr, and H. W. Potzl, "Analysis of Breakdown Phenomena in MOSFET's," *IEEE Trans. Computer-Aided Design*, vol. CAD-1, pp. 77-85, April 1982.

[3.28] J. W. Slotboom and H. C. De Graaff, "Measurements of Bandgap Narrowing in Si Bipolar Transistors," *Solid State Electronics*, vol. 19, pp. 857-862, Oct. 1976.

[3.29] R. J. Van Overstraeten, H. J. DeMan, and R. J. Mertens, "Transport Equations in Heavy Doped Silicon," *IEEE Trans. Electron Devices*, vol. ED-20, pp. 290-298, March 1973.

[3.30] L. Lapidus, G. F. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering*, John Wiley, New York, 1982.

[3.31] C. S. Rafferty, M. R. Pinto, and R. W. Dutton, "Iterative Methods in Semiconductor Device Simulation," *IEEE Trans. Computer-Aided Design*, vol. CAD-4, pp. 462-471, Oct. 1985.

[3.32] G. D. Hachtel, M. H. Mack, R. R. O'Brien, and B. Speelpennig, "Semiconductor Analysis Using Finite Elements - Part I: Computational Aspects," *IBM J. Res. and Dev.*, vol. 25, pp. 232-245, 1981.

[3.33] J. W. Slotboom, "Computer-Aided Two-Dimensional Analysis of Bipolar Transistors," *IEEE Trans. Electron Devices*, vol. ED-20, pp. 669-679, Aug. 1973.

[3.34] P. A. Markowich, C. A. Ringhofer, S. Selberherr, and E. Langer, "A Singularly Perturbed Boundary Value Problem Modelling a Semiconductor Device," *Report 2338*, MRC, University of Wisconsin, Madison, 1982.

[3.35] Z. Yu and R. W. Dutton, "SEDAN III-A General Purpose, One-Dimensional Semiconductor Analysis Program," *Technical Report*, Integrated Circuits Laboratory, Stanford University, Stanford, July 1985.

[3.36] G. D. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Clarendon Press, Oxford, 1978

[3.37] A. J. Davies, *The Finite Element Method: A First Approach*, Clarendon Press, Oxford, 1980.

[3.38] M. Kurata, *Numerical Simulation for Semiconductor Devices*, Lexington Books, MA, 1982.

[3.39] M. S. Adler, "A Method for Achieving and Choosing Variable Density Grids in Finite Difference Formulations and the Importance of Degeneracy and Band Gap Narrowing in Device Modeling," *Proc. NASECODE-I*, Dublin, Ireland, pp. 3-30, June 1979.

[3.40] A. Franz, G. Franz, S. Selberherr, C. Ringhofer, and P. Markowich, "Finite Boxes - A Generalization of the Finite-Difference Method Suitable for Semiconductor Device Simulation," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1070-1082, Sept. 1983.

[3.41] J. J. Barnes and R. J. Lomax, "Finite-Element Methods in Semiconductor Device Simulation," *IEEE Trans. Electron Devices*, vol. ED-24, pp. 1082-1089, Aug. 1977.

[3.42] O. C. Zienkiewicz, *The Finite Element Method*, McGraw-Hill, London, 1977.

[3.43] W. M. Coughran, M. R. Pinto, R. K. Smith, "Computation of Steady-State CMOS Latchup Characteristics," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 307-323, Feb. 1988.

[3.44] B. Meinerzhagen, H. K. Dirks, and W. L. Engl, "Quasi-Simultaneous Solution Method: A Highly Efficient Strategy for Numerical MOST Simulations," *IEEE Trans. Electron Devices*, vol ED-32, pp. 2131-2138, Oct. 1985.

[3.45] C. Ringhofer and C. Schmeiser, "A Modified Gummel Method for the Basic Semiconductor Device Equations," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 251-253, Feb. 1988.

[3.46] M. S. Mock, *Analysis of Mathematical Models of Semiconductor Devices*, Boole Press, Dublin, 1983.

[3.47] R. E. Bank, W. M. Coughran, W. Fichtner, E. H. Grosse, D. J. Rose, and R. K. Smith, "Transient Simulation of Silicon Devices and Circuits," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 1992-2006, Oct. 1985.

[3.48] S. E. Laux, "Techniques for Small-Signal Analysis of Semiconductor Devices," *IEEE Trans. Electron Devices*, ED-32, pp. 2028-2037, Oct. 1985.

Chapter 4

[4.1] W. L. Engl, R. Laur, and H. K. Dirks, "MEDUSA - A Simulator for Modular Circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-1, pp. 85-93, April 1982.

[4.2] N. B. Guy Rabbat, A. L. Sangiovanni-Vincentelli, and H. Y. Hsieh, "A Multilevel Newton Algorithm with Macromodeling and Latency for the Analysis of Large-Scale Nonlinear Circuits in the Time Domain," *IEEE Trans. Circuits and Systems*, vol. CAS-26, pp. 733-740, Sept. 1979.

[4.3] Z. Yu, M. Vanzi, and R. W. Dutton, "An Extension to Newton's Method in Device Simulators - On An Efficient Algorithm to Evaluate Small-Signal Parameters and to Predict Initial Guess," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 41-45, Jan. 1987.

[4.4] A. Vladimirescu, "LSI Circuit Simulation on Vector Computers," *Memo No. UCB/ERL M82/75*, Electronics Research Laboratory, University of California, Berkeley, Oct. 1982.

[4.5] P. Yang, "An Investigation of Ordering, Tearing and Latency Algorithms for the Time-Domain Simulation of Large Circuits," *Report R-891*, Coordinated Science Lab., Univ. of Illinois, Urbana, Aug. 1980.

[4.6] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas," *IEEE Proc.*, vol. 60, pp. 98-108, Jan. 1972.

[4.7] G. K. Jacob, "Direct Methods in Circuit Simulation Using Multiprocessors," *Memo No. UCB/ERL M87/67*, Electronics Research Laboratory, University of California, Berkeley, Oct 1987.

[4.8] T. Quarles, A. R. Newton, D. O. Pederson, and A. Sangiovanni-Vincentelli, *SPICE 3B1 User's Guide*, Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, April 1988.

Chapter 5

[5.1] L. W. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits," *Memo No. ERL-M520*, Electronics Research Laboratory, University of California, Berkeley, May 1975.

[5.2] T. Quarles, A. R. Newton, D. O. Pederson, and A. Sangiovanni-Vincentelli, *SPICE 3B1 User's Guide*, Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, April 1988.

[5.3] R. S. Varga, *Matrix Iterative Analysis*, Englewood Cliffs, Prentice Hall, 1962.

[5.4] W. Gellert, H. Kustner, M. Hellwich, H. Kastner, *The VNR Concise Encyclopedia of Mathematics*, Van Nostrand Reinhold Company, New York, 1977.

[5.5] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*, Springer-Verlag, 1984.

[5.6] H. K. Gummel, "A Self-Consistent Iterative Scheme for One-Dimensional Steady State Transistor Calculations," *IEEE Trans. Electron Devices*, vol. ED-11, pp. 455-465, Oct. 1964.

[5.7] M. Kurata, *Numerical Simulation for Semiconductor Devices*, Lexington Books, MA, 1982.

[5.8] B. Gokhale, "Numerical Solutions for a One-Dimensional Silicon n-p-n Transistor," *IEEE Trans. Electron Devices*, vol. ED-17, p. 594, 1970.

[5.9] M. R. Pinto, C. S. Rafferty, H. R. Yeager, and R. W. Dutton, *PISCES-IIB Supplementary Report*, Stanford University, Stanford, 1985.

[5.10] S. M. Sze, *Physics of Semiconductor Devices*, John Wiley, 1981.

[5.11] W. L. Engl and H. Dirks, "Functional Device Simulation by Merging Numerical Building Blocks," *Proc. NASECODE II*, Dublin, pp. 34-62, June 1981.

[5.12] L. W. Nagel and R. A. Rohrer, "Computer Analysis of Nonlinear Circuits, Excluding Radiation (CANCER)," *IEEE J. Solid-State Circuits*, vol. SC-6, pp. 166-182, Aug. 1971.

[5.13] Z. Yu, M. Vanzi, and R. W. Dutton, "An Extension to Newton's Method in Device Simulators - On An Efficient Algorithm to Evaluate Small-Signal Parameters and to Predict Initial Guess," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 41-45, Jan. 1987.

[5.14] G. H. Meyer, "On Solving Nonlinear Equations with a One Parameter Operator Imbedding," *SIAM J. Numer. Anal.*, vol. 5, pp. 739-752, 1968.

[5.15] R. E. Bank and D. J. Rose, "Global Approximate Newton Methods," *Numersiche Mathematik*, vol. 37, pp. 279-295, 1981.

[5.16] A. F. Franz, G. A. Franz, S. Selberherr, C. Ringhofer, and P. Markowich, "Finite Boxes-A Generalization of the Finite-Difference Method Suitable for Semiconductor Device Simulation," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1070-1082, Sept. 1983.

[5.17] E. Polak, *Computational Methods in Optimization: A Unified Approach*, Academic Press, 1981.

[5.18] K. Mayaram and F. Romeo, "The Application of a Damped Newton Method to the Computation of the DC Operating Point of Electronic Circuits," *EECS 219 Class Project*, University of California, Berkeley, Fall 1985.

[5.19] H. R. Yeager and R. W. Dutton, "Improvement in Norm-Reducing Newton Methods for Circuit Simulation," *Proc. NUPAD* , San Diego, May 1988.

[5.20] A. L. Sangiovanni-Vincentelli, "Circuit Simulation", in *Computer Design Aids for VLSI Circuits*, P. Antognetti, D. O. Pederson, and H. De Man (Editors), Groningen, The Netherlands: Sijthoff and Noordhoff, pp. 19-113, 1981.

[5.21] G. Dahlquist, "Stability and Error Bounds in the Numerical Integration of Ordinary Differential Equations," *Trans. of the Royal Inst. of Technology*, no. 130, pp. 1-86, 1959.

[5.22] J. Lambert, *Computational Methods in Ordinary Differential Equations*, Wiley, New York, 1973.

[5.23] R. E. Bank, W. M. Coughran, W. Fichtner, E. H. Grosse, D. J. Rose, and R. K. Smith, "Transient Simulation of Silicon Devices and Circuits," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 1992-2006, Oct. 1985.

[5.24] M. R. Pinto, C. S. Rafferty, and R. W. Dutton, "PISCES-II: Poisson and Continuity Equation Solver," *Tech. Report*, Stanford University, Stanford, 1984.

[5.25] P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations*, Wiley, New York, 1962.

[5.26] W. Liniger, F. Odeh, A. Ruehli, "Integration Methods for the Solution of Circuit Equations," in *Circuit Analysis, Simulation and Design*, A. E. Ruehli (Editor), North-Holland, 1986.

[5.27] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas," *IEEE Proc.*, vol. 60, pp. 98-108, Jan. 1972.

[5.28] R. K. Brayton and C. H. Tong, "Stability of Dynamical Systems: A Constructive Approach," *IEEE Trans. Circuits and Systems*, vol. CAS-26, pp. 224-234, April 1979.

[5.29] R. Saleh, "Nonlinear Relaxation Algorithms for Circuit Simulation," *Memo No. UCB/ERL M87/21*, Electronics Research Laboratory, University of California, Berkeley, April 1987.

[5.30] M. S. Mock, *Analysis of Mathematical Models of Semiconductor Devices*, Boole Press, Dublin 1983.

[5.31] W. T. Nye, "Techniques for using Spice Sensitivity Computations in DELIGHT.SPICE Optimization," *Proc. IEEE ICCAD-86*, Santa Clara, CA, pp. 92-95, Nov. 1986.

[5.32] A. Gnudi, P. Ciampolini, R. Guerrieri, M. Rudan, and G. Baccarani, "Sensitivity Analysis for Device Design," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 879-885, Sept. 1987.

[5.33] D. E. Hocevar, P. Yang, T. N. Trick, and B. D. Epler, "Transient Sensitivity Computation for MOSFET Circuits," *IEEE Trans. Electron Devices*, vol. ED-32, pp. 2165-2176, Oct. 1985.

[5.34] W. Nye, D. C. Riley, A. Sangiovanni-Vincentelli, and A. L. Tits, "DELIGHT.SPICE: An Optimization-Based System for the Design of Integrated Circuits," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 501-518, April 1988.

Chapter 6

[6.1] K. Mayaram and D. O. Pederson, "Circuit Simulation in LISP", *Memo. No. UCB/ERL M84/60*, Electronics Research Laboratory, University of California, Berkeley, Aug. 1984.

[6.2] K. Mayaram and D. O. Pederson, "Circuit Simulation in LISP," *Proc. IEEE ICCAD-84*, Santa Clara, CA, pp. 24-26, Nov. 1984.

[6.3] K. Mayaram, "Object-Oriented Programming in BIASlisp," *Memo. No. UCB/ERL M84/103*, Electronics Research Laboratory, University of California, Berkeley, Dec. 1984.

[6.4] T. Quarles, A. R. Newton, D. O. Pederson, and A. Sangiovanni-Vincentelli, *SPICE 3B1 User's Guide*, Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, April 1988.

[6.5] H. Cannon, "A Non-Hierarchical Approach to Object-Oriented Programming", *unpublished paper*, Artificial Intelligence Laboratory, MIT, Cambridge.

[6.6] D. Weinreb and D. Moon, *LISP Machine Manual*, Symbolics Inc., 1981.

[6.7] K. Mayaram and D. O. Pederson, "CODECS: An Object-Oriented Mixed-Level Circuit and Device Simulator," *Proc. IEEE ISCAS*, Philadelphia, pp. 604-607, May 1987.

[6.8] D. G. Bobrow, K. Kahn, G. Kiczales, L. Masinter, M. Stefik, and F. Zdybel, "CommonLoops: Merging Common Lisp and Object-Oriented Programming," *Intelligent Systems Lab. Series ISL-85-8*, Xerox Palo Alto Research Center, Aug. 1985.

[6.9] K. S. Kundert, "Sparse-Matrix Techniques and their Application to Circuit Simulation," in *Circuit Analysis, Simulation and Design*, A. E. Ruehli (Editor), pp. 281-324, North-Holland Co., Cambridge, MA, 1986.

Chapter 7

[7.1] S. K. Ghandhi, *Power Semiconductor Devices*, John Wiley and Sons, 1977.

[7.2] S. M. Sze, "Physics of Semiconductor Devices," John Wiley, 1981.

[7.3] I. Getreu, *Modeling The Bipolar Transistor*, Tektronix, 1976.

[7.4] P. Yang, B. D. Epler, P. K. Chatterjee, "An Investigation of Charge Conservation Problem for MOSFET Circuit Simulation," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 128-138, Feb. 1983.

[7.5] J. White, "The Multirate Integration Properties of Waveform Relaxation, with Application to Circuit Simulation and Parallel Computation," *Memo No. UCB/ERL 85/90*, Electronics Research Laboratory, University of California, Berkeley, Nov. 1985.

[7.6] D. E. Ward and R. W. Dutton, "A Charge-Oriented Model for MOS Transistor Capacitances," *IEEE J. Solid-State Circuits*, vol. SC-13, no. 5, pp. 703-707, Oct. 1978.

[7.7] J. E. Meyer, "MOS Models and Circuit Simulation," *RCA Rev.*, vol. 32, pp. 42-63, March 1971.

[7.8] B. J. Sheu, "MOS Transistor Modeling and Characterization for Circuit Simulation," *Memo No. UCB/ERL M85/85*, Electronics Research Laboratory, University of California, Berkeley, Oct. 1985.

[7.9] S. Y. Oh, D. E. Ward, and R. W. Dutton, "Transient Analysis of MOS Transistors," *IEEE Trans. Electron Devices*, vol. ED-27, no. 8 pp. 1571-1578, Aug. 1980

Chapter 8

[8.1] K. Ogiue, N. Odaka, S. Miyaoka, I. Masuda, T. Ikeda, K. Tonomura and T. Ouba, "A 13 ns/500mW 64Kb ECL RAM," *ISSCC Digest of Tech. Papers*, Anaheim, CA, pp. 212-213, Feb. 1986.

[8.2] Y. Nishio, I. Masuda, T. Ikeda, M. Iwamura, K. Ogiue, and Y. Suzuki, "A Subnanosecond Low Power Advanced Bipolar-CMOS Gate Array," *Proc.*

ICCD, pp. 428-433, Oct. 1984.

[8.3] S. M. Sze, *Physics of Semiconductor Devices*," John Wiley, 1981.

[8.4] S. K. Ghandhi, *Power Semiconductor Devices*, John Wiley and Sons, 1977.

[8.5] H. De Los Santos and B. Hoefflinger, "Optimization and Scaling of CMOS-bipolar Drivers for VLSI Interconnects," *IEEE Trans. Electron Devices*, vol. ED-33, pp. 1722-1730, Nov. 1986.

[8.6] E. W. Greeneich and K. L. McLaughlin, "Analysis and Characterization of BiCMOS for High-Speed Digital Logic," *IEEE J. Solid-State Circuits*, vol. 23, pp. 566-572, April 1988.

[8.7] G. P. Rosseel, R. W. Dutton, K. Mayaram, and D. O. Pederson, "Delay Analysis for BiCMOS Drivers," *Proc. 1988 BCTM*, Minneapolis, MN, pp. 220-222, Sept. 1988.

[8.8] I. Getreu, *Modeling The Bipolar Transistor*, Tektronix, 1976.

[8.9] G. P. Rosseel, R. W. Dutton, K. Mayaram, and D. O. Pederson, "Bipolar Scaling for BiCMOS Circuits," *Digest of Tech. Papers 1988 Symp. VLSI Circuits*, Tokyo, Japan, pp. 67-68, Aug. 1988.

[8.10] Z. Yu and R. W. Dutton, "SEDAN III-A General Purpose, One-Dimensional Semiconductor Analysis Program," *Tech. Report*, Integrated Circuits Laboratory, Stanford University, Stanford, July 1985.

[8.11] M. R. Pinto, C. S. Rafferty, H. R. Yeager, and R. W. Dutton, *PISCES-IIB Supplementary Report*, Stanford University, Stanford, 1985.

[8.12] H. Benda and E. Spenke, "Reverse Recovery Processes in Silicon Power Rectifiers," *Proc. IEEE*, vol. 55, pp. 1331-1354, Aug. 1967.

[8.13] E. Berz, "Ramp Recovery in p-i-n Diodes," *Solid-State Electronics*, vol. 23, pp. 783-792, 1980.

[8.14] P. Bacuvier, J. Arnould, and J. P. Noguier, "Analyse et optimisation technologique des parametres de structure conditionnant la progressivite du recouvrement des diodes p+nn+ - application a la realisation industrielle de diodes rapides a faible vitesse d'extinction du courant de recouvrement," *Tech. Report*, Laboratoires de Tours, TOURS, 1977.

[8.15] V. A. K. Temple, F. W. Holroyd, M. S. Adler, and P. V. Gray, "The Effect of Carrier Lifetime Profile on Turn-off Time and Turn-on Losses," *IEEE Power Electronics Specialist Conf. Record*, pp. 155-163, June 1980.

[8.16] Y. C. Kao and J. R. Davis, "Correlations Between Reverse Recovery Time and Lifetime of p-n Junction Driven by a Current Ramp," *IEEE Trans. Electron Devices*, vol. ED-17, pp. 652-657, Sept. 1970.

[8.17] B. Tien and C. Hu, "Determination of Carrier Lifetime from Rectifier Ramp Recovery Waveform," *IEEE Electron Device Lett.*, vol. 9, pp. 553-555, Oct. 1988.

[8.18] K. Mayaram, B. Tien, C. Hu, and D. O. Pederson, "Simulation and Modeling for Soft Recovery of p-i-n Rectifiers," *IEDM Digest of Tech. Papers*, Dec. 1988.

[8.19] W. McMurray, "Optimum Snubbers for Power Semiconductors," *IEEE Trans. Industry Applications*, vol. IA-8, pp. 593-600, Sept./Oct. 1972.

[8.20] B. J. Sheu and C. Hu, "Switch-Induced Error Voltage on a Switched Capacitor," *IEEE J. Solid-State Circuits*, vol. SC-19, pp. 519-525, Aug. 1984.

[8.21] J. H. Shieh, M. Patil, and B. J. Sheu, "Measurement and Analysis of Charge Injection in MOS Analog Switches," *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 277-281, April 1987.

[8.22] C. Turchetti, P. Mancini, G. Masetti, "A CAD-Oriented Non-Quasi-Static Approach for Transient Analysis of MOS IC's," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 827-835, Oct. 1986.

[8.23] G. Wegmann, E. A. Vittoz and F. Rahali, "Charge Injection in Analog MOS Switches," *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 1091-1097, Dec. 1987.

[8.24] J. B. Kuo, R. W. Dutton, and B. A. Wooley, "MOS Pass Transistor Turn-Off Transient Analysis," *IEEE Trans. Electron Devices*, vol. ED-33, pp. 1545-1554, Oct. 1986.

[8.25] R. E. Suarez, P. R. Gray, and D. A. Hodges, "All-MOS Charge Redistribution Analog-to-Digital Conversion Techniques Part II," *IEEE J. Solid-State Circuits*, vol. SC-10, pp. 379-385, Dec. 1975.

[8.26] L. Bienstman and H. J. DeMan, "An Eight-Channel 8-bit Microprocessor compatible NMOS D/A converter with programmable scaling," *IEEE J. Solid-State Circuits*, vol. SC-15, pp. 1051-1059, Dec. 1980.

[8.27] K. Mayaram, J. Lee, and C. Hu, "A Model for the Electric Field in Lightly Doped Drain Structures," *IEEE Trans. Electron Devices*, vol. ED-34, pp. 1509-1518, July 1987.

[8.28] H. J. Park, P. K. Ko, and C. Hu, "A Non-Quasistatic MOSFET Model for SPICE," *IEDM Digest of Tech. Papers*, pp. 652-655, Dec. 1987.

Chapter 9

[9.1] G. K. Jacob, "Direct Methods in Circuit Simulation Using Multiprocessors," *Memo. No. UCB/ERL M87/67*, Electronics Research Laboratory, University of California, Berkeley, 1987.

[9.2] R. Lucas, "Solving Planar Systems of Equations on Distributed-Memory Multiprocessors," *Ph.D. Dissertation*, Stanford University, Stanford, Dec. 1987.

[9.3] P. Sadayappan and V. Visvanathan, "Circuit Simulation on a Multiprocessor," *Proc. IEEE 1987 CICC*, Portland, OR, pp. 124-128, May 1987.

[9.4] C. P. Yuan, R. Lucas, P. Chan, and R. Dutton, "Parallel Electronic Circuit Simulation on the iPSC System," *Proc. IEEE 1988 CICC*, Rochester, NY, pp. 6.5.1-6.5.4, May 1988.

[9.5] S. Y. Oh, "A Simplified Two-Dimensional Numerical Analysis of MOS Devices Including Transient Phenomena," *Tech. Report No. G201-10*, Stanford University, Stanford, June 1981.

[9.6] E. Sano, T. Tsukahara, and T. Kimura, "A Low-Frequency Noise Analysis Using Two-Dimensional Numerical Analysis Method," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1699-1704, Dec. 1983.

[9.7] V. C. Alwin, D. H. Navon, and L. J. Turgeon, "Time-Dependent Carrier Flow in a Transistor Structure Under Nonisothermal Conditions," *IEEE Trans. Electron Devices*, vol. ED-24, pp. 1297-1304, Nov. 1977.

[9.8] C. H. Price, "Two-Dimensional Numerical Simulation of Semiconductor Devices," *Ph.D. Dissertation*, Stanford University, Stanford, 1982.