Order Number 1337994

# Design and implementation of an integrated VLSI packaging support software environment

Whipple, Thomas Driggs, M.S.

The University of Arizona, 1989

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

# DESIGN AND IMPLEMENTATION OF AN INTEGRATED VLSI PACKAGING SUPPORT SOFTWARE ENVIRONMENT

by

Thomas Driggs Whipple

---

A Thesis Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

MASTER OF SCIENCE

WITH A MAJOR IN ELECTRICAL ENGINEERING

In the Graduate College

THE UNIVERSITY OF ARIZONA

1 9 8 9

## STATEMENT BY AUTHOR

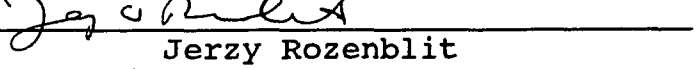
This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgement the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: 

## APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:



Jerzy Rozenblit
Assistant Professor of ECE

7/17/89
Date

## ACKNOWLEDGEMENTS

I would like to thank Dr Jerzy Rozenblit for his constant guidance and encouragement. I would also like to thank Dr. O. A. Palusinski and Dr. J. L. Prince for their comments and suggestions. This project has been made possible by grant number 88-MP-086 from the Semiconductor Research Corporation.

Dr. Jen-Chyi Liao and Brad Cooke have given much guidance in the use and application of the simulation tools discussed herein, and Nitin Garg provided much assistance in the debugging of PDSE.

My greatest appreciation goes to Luzetta and Nathanael for their constant support and patience.

TABLE OF CONTENTS

LIST OF FIGURES

## ABSTRACT

An interactive software shell has been developed which integrates several packaging simulation tools developed at the University of Arizona which are used to analyze electro-magnetic coupling between interconnects in an integrated circuit. This software shell uses experimental frames to manage this simulation process. Through the experimental frames, the model descriptions and the model inputs are separated, and input data is verified for correctness. This model/input separation allows several model variations to be tested based on several input variations. The results of these simulations are then analyzed and displayed graphically. Further work for the software shell is discussed. This tool provides a user-friendly, efficient method for performing coupled-line analyses in interconnect systems.

# CHAPTER 1

## INTRODUCTION

As signal rise times in integrated circuits become shorter than 1 ns, it becomes necessary to be able to simulate the electromagnetic and mechanical/thermal coupling between the elements of electronic systems. Several simulation tools have been developed by the University of Arizona Electronics Packaging Laboratory to serve this aim. These tools include parameter calculators, coupled-line analysis tools, and thermal/mechanical analysis tools. The parameter calculators calculate capacitance and inductance between interconnect lines in an IC. The coupled-line analysis tools use the results of the parameter calculators to determine pulse propogation, distortion, and cross-talk which occurs on the interconnect lines, and the thermal/mechanical tools calculate various parameters dealing with the thermal and mechanical effects in the IC package.

Up till now, each of these tools was executed independently. When one tool required the results of another tool, both had to be executed separately. Each one has its unique input and output file format, and each is

executed in its own unique way. Because of this, it was tedious and time-consuming to execute more than just a few simulations of some interconnect system. Also, when running several tools on one system, it was sometimes necessary to re-enter data when changing tools. Another aspect was that these simulation tools were designed to execute quickly and accurately; however, few data consistency checks were made, so when inaccurate data was entered, the program usually crashed.

Because of these difficulties, an interactive software system has been developed which brings together under one shell the electrical tools mentioned above. (The thermal/mechanical tools have not yet been integrated into the system.) This thesis describes PDSE -- Packaging Design and Simulation Environment, which provides an interface between the user and the simulation tools. The flowchart in Figure 1.1 briefly describes the simulation process. Through this interface, the user enters one or more model specifications. PDSE verifies the data that is input and signals the user when invalid data is input. Once the models have been correctly specified, several simulations based on the set of models may be executed. This provides for easy and efficient modelling and simulation of an interconnect system. This thesis describes the above process in more detail.

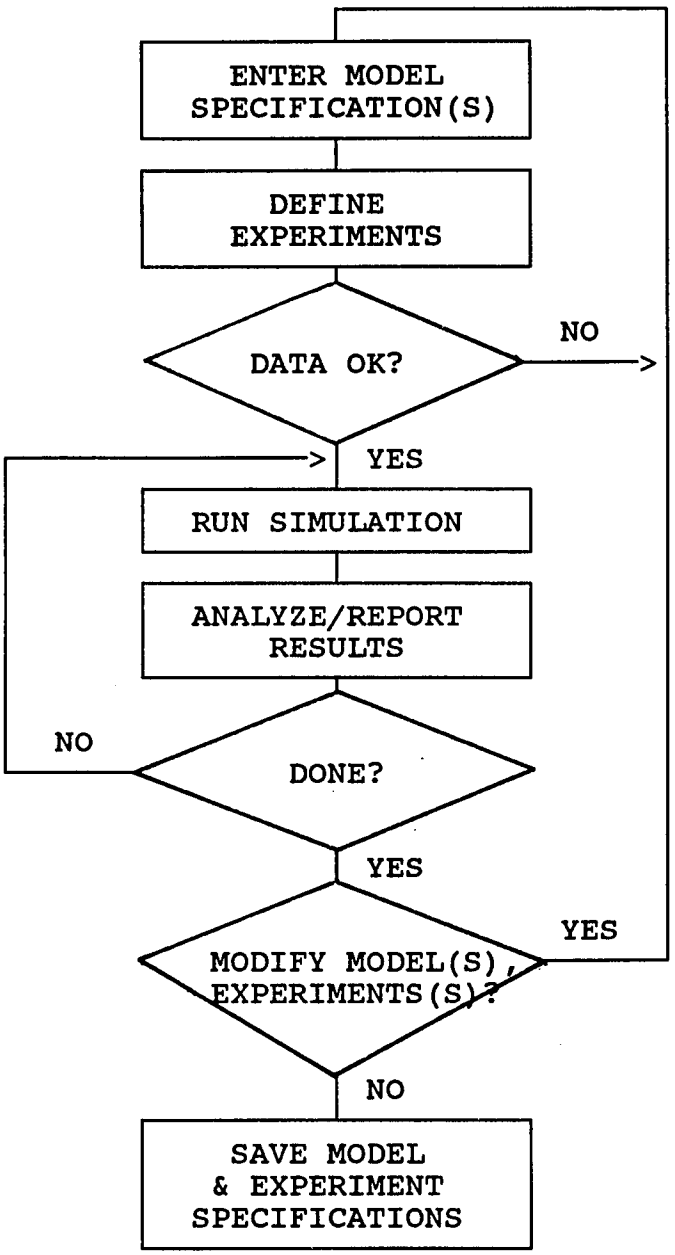



Figure 1.1 IC interconnect simulation flowchart.

Chapter two describes the methods which are used to organize and execute the simulations. Chapter three gives details about the simulation tools executed by PDSE. Chapters four, five, and six describe PDSE's execution, and chapter seven outlines future work for further integration of these tools.

# CHAPTER 2

# BACKGROUND

Before PDSE and its operation is described, the main concepts which provide the foundation for the operation of PDSE are described. PDSE employs simulation management techniques derived from multi-faceted modelling methodology [1]. The experimental frame is used extensively to characterize and organize the simulation process.

## 2.1 Simulation Management

The simulation management techniques used support VLSI package design derived from a model-based system design framework [2]. In this methodology, the designer develops a model from which a new system will be created. As opposed to system analysis, where the model is derived from an existing, real system, in design the model comes first as a set of "blueprints" from which the system will be built, implemented, or deployed. The blueprints might take several forms; they could be simple informal descriptions, a set of equations, or a complex computer program. The goal of the model-based design methodology is

to study models of designs before they are implemented and physically realized.

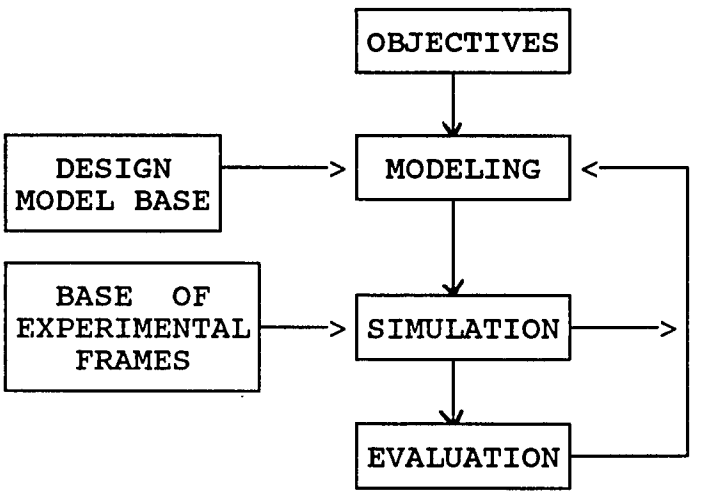



Figure 2.1 Design in the modeling context.

As depicted in Figure 2.1, the design process is supported by the methodology as follows: Design Objectives (understood here as a set of design requirements, constraints, and purposes for which a design process is undertaken) drive the design model development process. In this process the designer has facilities for retrieval of design models which conform to the objectives, constraints, and requirements from the Design Model Base. If no models can be retrieved, a new model is constructed. It is assumed that all possible design models that satisfy design constraints and requirements are generated. They are termed design alternatives. The models are then

evaluated through simulation studies using the experimental frame.

## 2.2 Experimental Frame Concept

"An experimental frame specifies a limited set of circumstances under which a system ... is to be observed or subjected to experimentation." [1] It is defined by the structure below:

$$<T, I, C, O, \Omega_I, \Omega_C, SU>$$

Where
- T is a time base
- I is a set of input variables to the model
- C is a set of run control variables which control the execution of the simulation
- O is a set of output variables from the model
- $\Omega_I$ is a set of admissible input segments which map time into the input variable ranges.
- $\Omega_C$ is a set of admissible control segments which map time and/or control variables into the control ranges.
- SU is a set of summary variables of the simulation [1]

Once the experimental frame variables have been defined, the principle of model experimental frame separation is employed to partition the model from the experiment. This principle, as stated below, provides more flexibility in the simulation process because the model is independent of the experiment to be performed on it.

> Any data gathering/reduction (statistics, performance, measurements, etc.) or behavioral control (initialization, termination, etc.) that is conceptually not carried out in the real system should not be placed within its model but rather formulated as part of the experimental frame.
>
> Conversely, any dynamic structure that is supposed to correspond to mechanisms existing in

the real system should be placed within the model [1].

The experimental frame is partitioned into a generator, an acceptor, and a transducer. A generator generates a set of admissible inputs for the model. An acceptor analyzes the output from the model and controls the continuation of the simulation by signalling the generator when to stop generating inputs for the model. A transducer accepts output created by the model. A structural realization of an experimental frame is shown in Figure 2.2.

Now that the concepts upon which PDSE has been built have been described, the details of PDSE's operation can be addressed. The use of Experimental Frames in PDSE is discussed in Chapter 5.

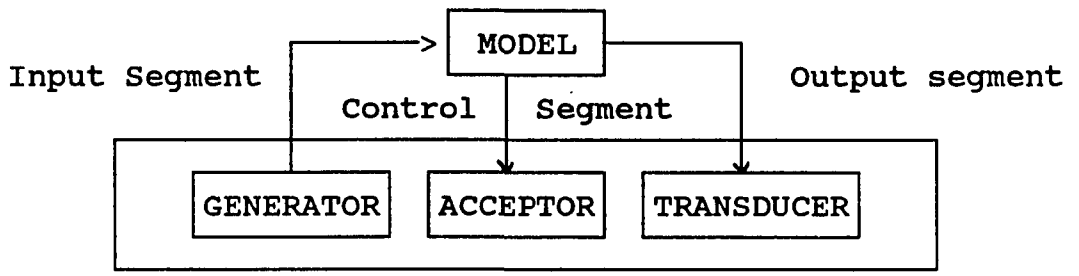



Figure 2.2 Structural realization of an experimental frame [3].

# CHAPTER 3

## SIMULATION TOOLS

IC interconnects are conductors in an IC package which transmit signals from the integrated circuit to the package pins or to another integrated circuit in the same package. Figure 3.1 shows an actual layout of an IC interconnect system. This chapter describes the tools used by PDSE for parameter calculations and coupled-line analysis.

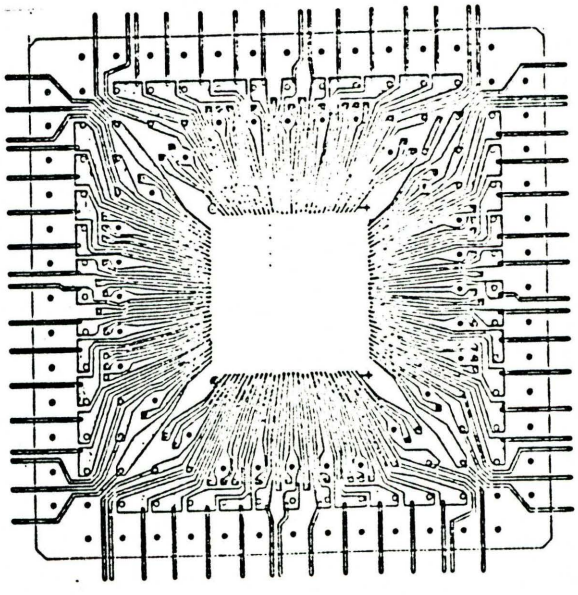


Figure 3.1 Actual layout of an interconnect system.

## 3.1 Parameter Calculators

Given a description of the cross-sectional geometry of a transmission line system (as described below), the parameter calculators, UAMOM and UAC, will calculate the capacitance matrix and inductance matrix (per unit length) for parallel multiconductor systems. Lines are assumed to be of infinite length, and end effects are not calculated. Ground planes and dielectric interfaces are taken as infinite in extent [4]. Up to ten conductors may be specified, and each conductor may have up to twelve sides. When specifying dielectric layers, neighboring dielectrics must be assigned different dielectric constant. The specific limitations of each tool are described in the sections below.

### 3.1.1 Method of Moments Parameter Calculator -- UAMOM

Up to ten dielectric layers may be specified. The moment calculator will evaluate the TEM line parameters for geometries that lie in one of the categories below:

1. One infinite ground plane present (microstrip-like), finite sized ground conductors allowed.
2. Two infinite ground planes present (stripline-like), finite sized ground conductors allowed.
3. Only finite sized ground conductors present [4].

Up to ten arbitrarily shaped conductors may be defined, any number of which may designated as ground conductors. Conductors may intersect with any number of dielectric

interfaces, but must not touch each other or ground planes. To facilitate the calculation of the parameters, all of the conductor sides and dielectric interfaces are approximated by straight line segments (subintervals). The user may specify the number of subintervals desired for the calculation. If not specified, a default value of five subintervals will be assigned. The more subintervals per side, the longer the calculation will take. It has been found that a subinterval count of five to eight will generally give parameter values that converge to one percent. The dielectric interfaces and ground planes must be truncated at a fixed distance from the conductors. The user has the option of specifying how far from the end conductor these interfaces will be truncated. Because this is a difficult value to determine, PDSE will assign a 'best guess' value if the user doesn't supply one.

### 3.1.2 Capacitance Calculator -- UAC

The allowable geometries for UAC are more limited than those in UAMOM. These geometries are shown in Figures 3.2 - 3.4. In each case every conductor must lie wholly within one dielectric medium. Case 1 has one finite ground conductor, no ground planes, and two dielectric layers. Case 2 has one infinite ground plane, no finite ground conductors, and two dielectric layers. Case 3 has

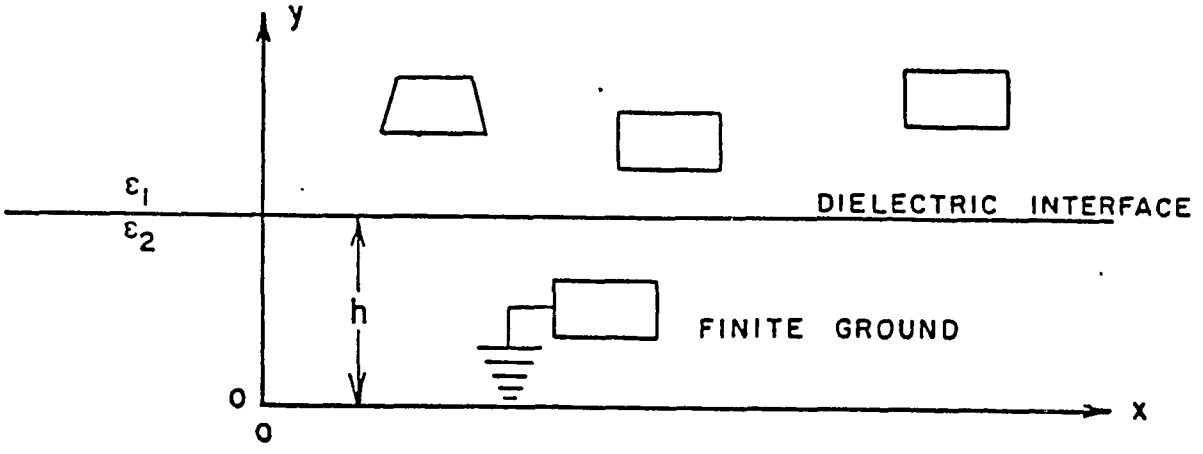



Figure 3.2 UAC Geometry Case 1: Interconnection system with one finite-sized ground conductor.

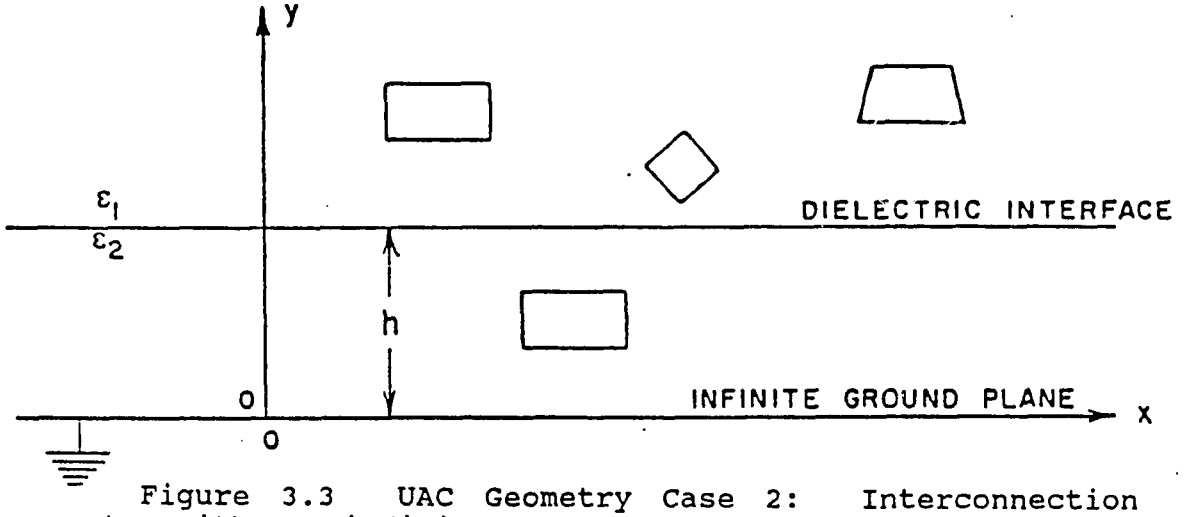



Figure 3.3 UAC Geometry Case 2: Interconnection system with one infinite ground plane.

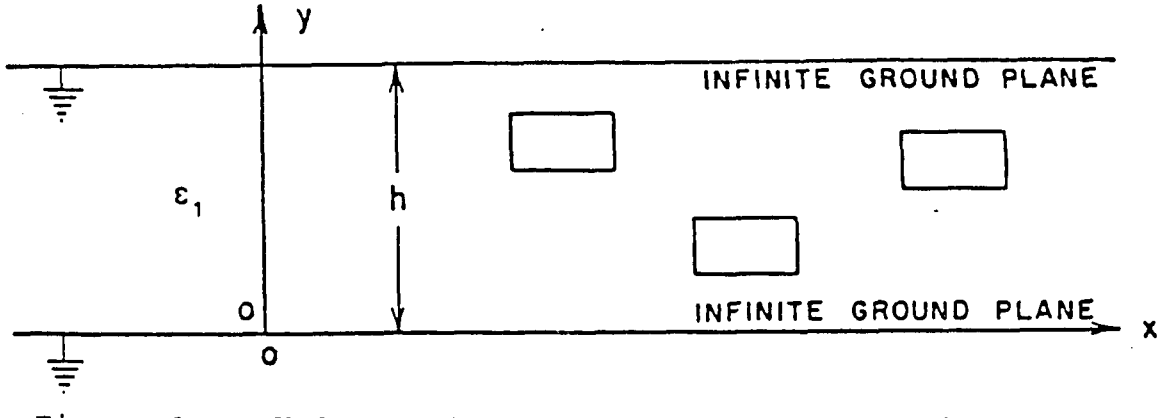



Figure 3.4 UAC Geometry Case 3: Interconnection system with two infinite ground planes.[10]

two ground planes, no finite ground conductors, and one dielectric layer. Conductor placement is arbitrary as long as the conductor does not touch any ground plane or dielectric interface. As in UAMOM, the number of subintervals per side may be specified for the calculation. However, in UAC, each side may be given a different number of subintervals. In this way, long segments may be divided up into many subintervals, while short segments may be divided up into a few. It has been found that a subinterval count of two to three will give calculation results converged within one percent.

### 3.1.3 Comparison of UAC and UAMOM

As has been mentioned, both UAC and UAMOM calculate capacitance and inductance matrices for a transmission line system in a dielectric. This section discusses the differences between UAC and UAMOM and gives suggestions as to when each of the programs should be used.

One of the main considerations about whether to use UAMOM or UAC is the geometry in question. As was pointed out in section 3.1.2, the allowable geometries for UAC are limited to three special cases. If a given geometry does not fit in one of these categories, UAMOM must be the tool that is used. If it does fit, however, the user must decide which of the tools should be used.

In general, the results of UAC converge quicker as the number of subintervals per side increases. According to

Liao,

> In cases studied to date, convergence to better than 0.1% of the asymptotic value has been obtained for elements four columns off the main diagonal (e.g. $C_{14}$) with (the number of subintervals) = 4 [5].

UAC does provide excellent accuracy, but it is slow. So if accuracy is the major concern, UAC should be used.

UAMOM, on the other hand, executes quickly, but the results do not converge as quickly as the number of subintervals per side are increased. It can provide results that are within five percent of the converged values of the UAC calculation and do it in a fraction of the time. If, however, there are many conductors specified, or they are widely-spaced, it may not be possible to specify enough subintervals per side to obtain a converged result. In that case, UAC would be more appropriate to use. Appendix A describes the results of three convergence tests which compare these two tools. The third convergence test in appendix A is a case where UAMOM was not able to converge before the allowable number of subintervals per side was exceeded.

### 3.2 Coupled Line Simulator -- UACSL

UACSL is a tool that simulates the transient response of a general linear network containing resistors, conductors, capacitors, inductors, coupled inductors, piecewise linear independent voltage sources, and lossless

uniform transmission line systems. The transmission line systems can be composed of more than one conductor and must be described by the capacitance and inductance matrices as calculated by UAMOM or UAC.

# CHAPTER 4

# PDSE - PACKAGING DESIGN & SIMULATION ENVIRONMENT

PDSE provides a robust, user-friendly shell integrating the three tools mentioned in the previous section. This section describes the general graphics tools used throughout PDSE, the main functions performed by PDSE, and the data file formats used by PDSE.

## 4.1 General Graphics Tool Descriptions

There are five main tools used in PDSE -- the main window, the menu, the matrix input/output screen, the dialog box, and the graphics display window. The first thing the user sees when running PDSE is the main window (see Figure 4.1). It is composed of five sub-windows. The PROCESS WINDOW is where most of the data input will take place. From the COMMAND WINDOW, the user may select which tool will be executed. The I/O WINDOW allows the user to call the c-shell or X-editor, load a data file, or save a data file. The load and save commands are only used when loading or saving data for the parameter calculators (UAMOM and UAC). The MESSAGE WINDOW is where PDSE reports the status of transactions to the user. Selections are

made by positioning the mouse pointer in the desired box and pressing the mouse's left button.

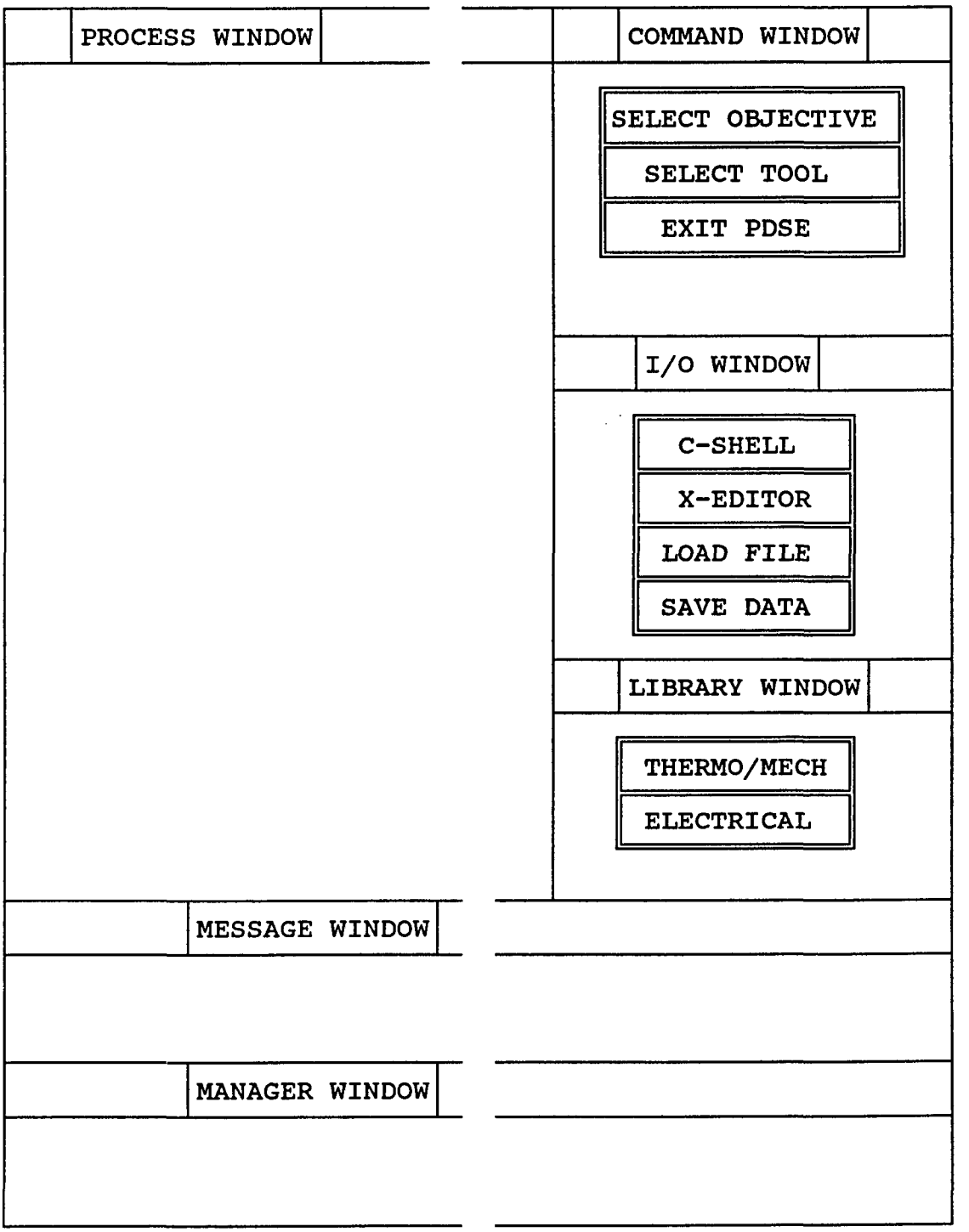
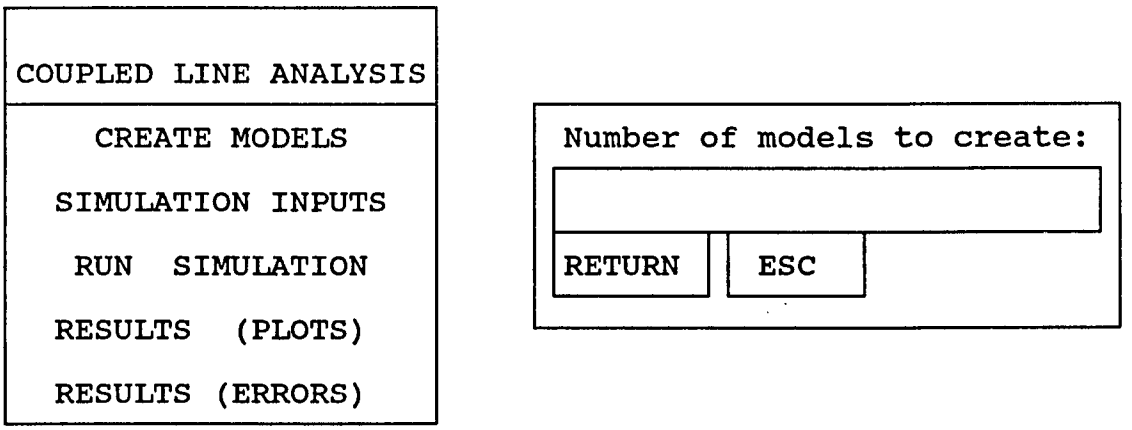



Figure 4.1 Main display in PDSE.

An example of the second tool, the menu, is shown in Figure 4.2 (a). Selections are made by positioning the mouse pointer at the desired menu item and pressing the left mouse button. The right mouse button, when pushed, will cause the menu to be destroyed and control to be passed back to the location from which the menu was called.




(a) (b)

Figure 4.2 (a) Example of menu. (b) Example of dialog box.

The third tool, the dialog box, is shown in Figure 4.2 (b). When it appears, the i/o window in the box (the middle window) will display the current value of the variable to be modified. The value in the box can be modified using simple EMACS editor commands (see appendix B for a listing of the valid editor commands). The 'Return' key is selected to enter the new value. If, however, it is desired to exit from the dialog box keeping the original value, the 'Esc' key is pressed. This will leave the

variable value unchanged even if modifications were made in the i/o window.

An example of the fourth tool, the matrix input/output window is shown in Figure 4.3. This tool is used to display and change data such as circuit component values and geometry specifications. Values are changed by positioning the mouse pointer to the desired cell

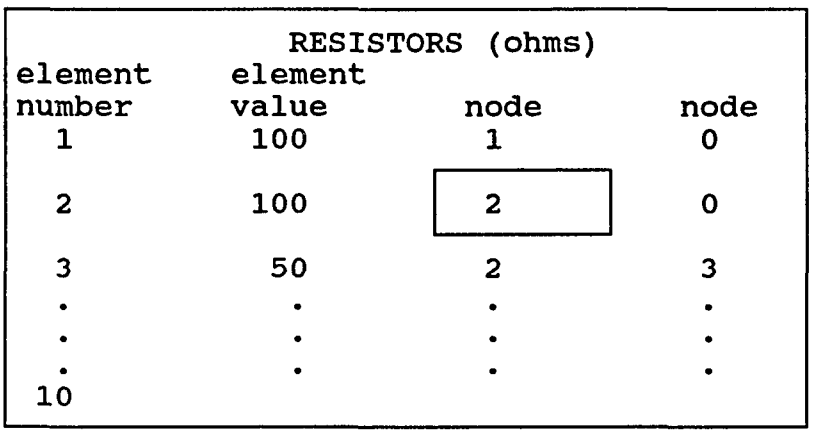



Figure 4.3 Example of Matrix i/o box.

and typing in the new value. The characters in the cell being edited change to yellow when in the edit mode and a yellow cursor appears. Again, simple EMACS-like commands may be used in the edit mode. The edit mode is terminated with a 'return' to accept the new value, or an 'esc' to abort the edit and keep the original value.

Values can be copied from one cell to any other cell. This is done by moving the cursor to the cell to be copied and pressing the middle mouse button, then moving the

cursor to the destination cell and pressing the 'return' key on the keyboard. The matrix i/o window can be exited by pressing the right mouse button.

The graphics window, the fifth tool, is a display tool only. It currently will display either the geometry specified for UAMOM or UAC, or it will display the transient responses calculated by UACSL. Whenever this window is displayed, the x,y coordinates of the mouse cursor will be displayed in the lower right hand corner of the window. This will allow the user to determine the coordinates of various points in a plot. The left mouse button will cause the display to zoom out, and the right mouse button causes the display to zoom in. The middle button will cause the screen to pan (shift).

## 4.2 PDSE Options - General Description

The three main functions available in PDSE are

1. Parameter Calculator,
2. Convergence Tester, and
3. Coupled Line Analysis.

The first one executes one of the parameter calculators. The second performs a convergence test using both parameter calculators. This convergence test will execute both UAC and UAMOM several times to determine the number of subintervals per side needed to obtain converged values. Another part of the convergence test will execute UAMOM to determine how far past the finite conductors the ground

planes and dielectric interfaces should be truncated. The third function performed by PDSE is multiple coupled-line analyses using UACSL. Up to ten models and up to ten driving sources may be specified at one time for UACSL. In this manner, UACSL may be executed in sequence 100 times using the ten models and ten driving sources with no user intervention. This function will automatically invoke UAMOM if the capacitance and inductance matrices have not previously been specified for any of the models. This is very convenient when it is desired to execute UACSL for many variations of some circuit configuration.

### 4.3 PDSE File Specifications

Due to the variety of functions available in PDSE, a number of file formats exist. The user need not know how the files are actually formatted; all that is necessary is to know what files exist and when they are used in PDSE. If the user is running UAMOM or UAC alone, the data files can be stored and loaded using the 'LOAD' and 'SAVE' buttons in the 'IO WINDOW' mentioned above. These files are stored with the exact name specified by the user.

If the user is executing UACSL, however, data files are referenced by a base file name and PDSE tacks on the model number at the end of the base name. For example, if it is desired to create 6 models to test using UACSL, and the user assigned the base file name to be 'testfile.dat', PDSE would store the six models in the files

'testfile.dat1', 'testfile.dat2',..., 'testfile.dat6'.

The driving sources for all these models are stored in a single file, the name of which is user specified. The results for each execution of UACSL in a series of tests are stored in a separate file. Again, the user gives a base filename for the results, and PDSE tacks on an index to specify which model and which driving source was used. For example, if there were six models and six driving waveforms, and if the base filename of the result file was specified as 'testfile.res', the test results would be saved in the files 'testfile.res.m1w1', 'testfile.res.m1w2', ..., 'testfile.res.m1w6', ..., 'testfile.res.m6w6'. The 'm' stands for model number and the 'w' stands for input wave number. The file 'testfile.res.m3w4' is interpreted as the results of UACSL execution on model three using waveform four. Again, it is emphasized that the user need only specify the base filename without the extensions when referencing these files through PDSE. The format of each of these data files is given in appendix C.

# CHAPTER 5

# MULTIPLE SIMULATION USING EXPERIMENTAL FRAMES

## 5.1 Frame Description

The experimental frame (EF) concept discussed in Chapter 2 is used in this project to simulate several variations of a model with several variations of input to the model. This is illustrated in Figure 5.1. Because the EF clearly separates the model from the input, this type of simulation is easily accomplished. The formal EF definition is found in appendix D. In this section, the generator, transducer, and acceptor of the EF are described (see Figure 5.2).

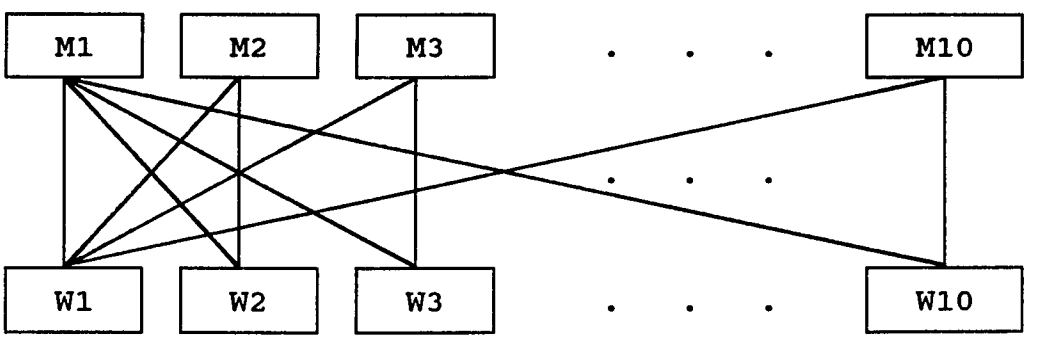



Figure 5.1 Block diagram of multiple-model, multiple-input simulation.

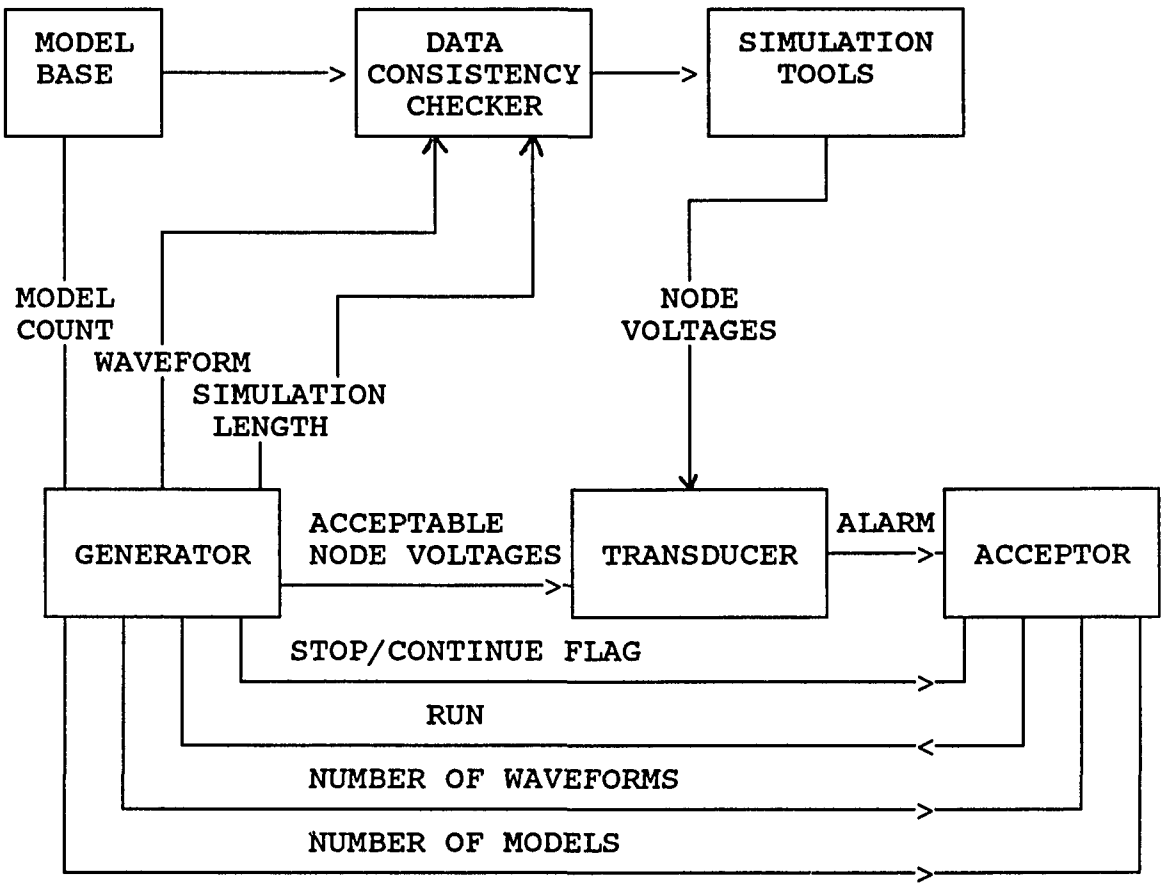



Figure 5.2 Block diagram of coupled-line analysis experimental frame.

The generator indicates which model in the model data base is to be simulated. The user may define and modify the interconnect system and termination networks to store in this model base. Up to ten variations of the circuit may be specified by the user. An example of a completed model is given in Figure 5.3.

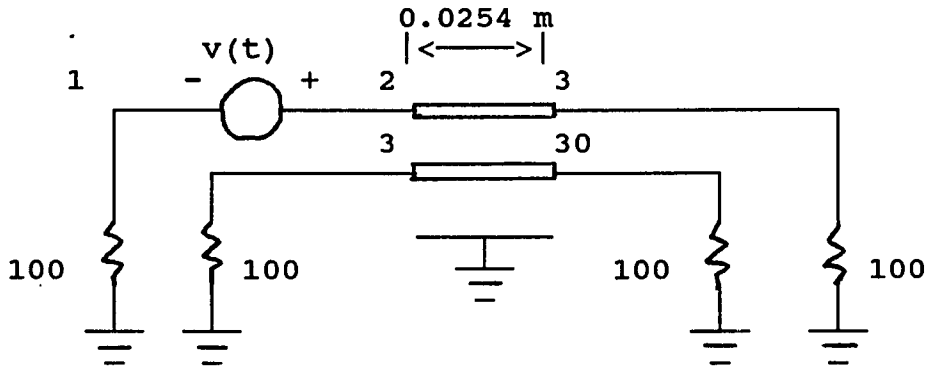



Figure 5.3 (a) Sample interconnect system with termination networks and voltage source attached.

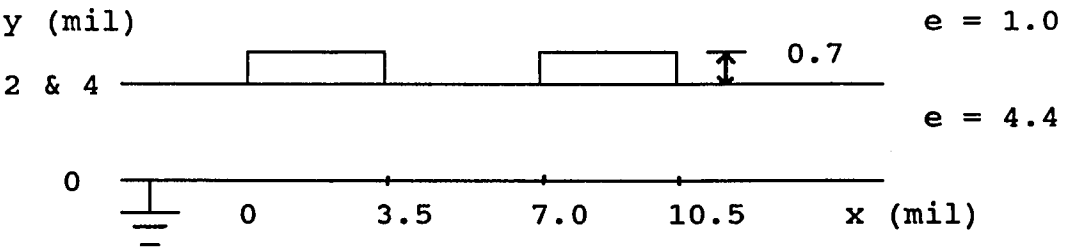



Figure 5.3 (b) Transmission line cross-section geometry.

Once the models have been input, the designer may then identify up to ten different waveforms to drive each of the (up to) ten models. Each waveform is specified with up to ten piecewise linear segments.

The generator also prompts the user for information such as length of each simulation (which is directed to the simulation tools,) the maximum acceptable voltages on each node of the circuit (which is sent to the transducer,) and a stop/continue flag that signals what to do if a voltage that is greater than allowed appears on a node (this flag is directed to the acceptor.)

The transducer accepts the node voltages from the simulation tools and compares them with the maximum acceptable node voltages that are provided by the generator. If a voltage that is greater than the allowed voltage appears on a node, an alarm is sent to the acceptor, and the model number, simulation number, node number, and maximum voltage on the node is stored for later retrieval and reporting. The voltages appearing at each simulation step are also saved by the transducer for later display.

The acceptor receives the alarm signal from the transducer, the stop/continue flag from the generator, and the number of waveforms and number of models to be tested from the generator. If the alarm and the stop/continue flag are set, or if all the models have been tested with all the waveforms, the acceptor signals the generator to stop the simulations. Otherwise, the simulations will continue.

## 5.2 Data Consistency Checks

One of the main modules in this architecture is the data consistency checker. This module checks the data for consistency with formats and requirements of the simulation tools. The data consistency checker has been designed so that inconsistent or missing data will be automatically replaced when possible. When this is not possible, the user is informed about the problem, and the simulation

process is not started until the discrepancy is corrected.

When executing UAMOM or UAC, the following items are verified:

1. There must be at least one active conductor,
2. There may not be more than ten dielectrics,
3. There may not be more than ten conductors,
4. There must be at least one ground plane or one ground conductor,
5. Only twelve sides may be specified for each conductor,
6. Each conductor must have at least three sides, (UAMOM will allow infinitely thin conductors which are described by only one side.)
7. Neighboring dielectrics may not have the same dielectric constant.
8. In UAC, conductors must not intersect any dielectric interface.
9. No conductor may be touching a ground plane.

If any of these conditions are not satisfied, the user is prompted to correct the situation until continuing.

There are several requirements that, if not satisfied by the user, that can easily be corrected by PDSE. For example, both UAMOM and UAC require the conductor vertices to be input in counter-clockwise order. If this has not been done, however, PDSE will correct the situation and continue with simulation. If the number of subintervals per side (UAMOM and UAC) and the discretization in x (UAMOM only) is not specified, PDSE will assign a default value and continue.

UAMOM allows only a certain number of subintervals per side, so PDSE calculates the maximum number allowed. If the number of subintervals exceeds that allowed, the allowed amount is used instead of the given amount. The

maximum allowable number of subintervals per side are calculated according to the following:

$$n < 200 \div x$$

where

n is the number of subintervals per side,
x is calculated below:

for every conductor side, add 1 to x;

for every dielectric interface that has no conductor intersection, add 4 to x;

for every dielectric interface that has conductors intersecting it, add 4 to x plus the number of conductors touching the interface minus 1;

for an extra ground plane, add 8 to x.

When UACSL is to be executed, PDSE verifies that there are capacitance and inductance matrices provided for each transmission line system. If they have not been provided, PDSE will automatically run UAMOM to calculate these values. PDSE then looks for any short circuits or open circuits in the termination networks provided by the user. When the circuit components are input by the user, two node numbers are also input to specify where in the network the component is to be placed. If any component has two node numbers which are the same, the user is notified of a short in the circuit. If throughout all the component specifications a node is only given once, an open circuit exists and, again, the user is notified of the situation. After all the data has been input and verified, the simulation process begins.

## 5.3 Experimental Frame Results

Once the simulation process is complete, two sets of results may be displayed on the terminal. First, plots of the voltages that appeared on each probe point in any selected simulation may be graphically displayed. Three menus are displayed to allow the user to select what is to be plotted -- the Model menu, the Waveform menu, and the Probe Point menu. The first two menus select which model's and which waveform's (input to the model) simulation results are to be displayed. The Probe Point menu then allows the user to select which probe's results in the given simulation are to be displayed. The graphics window in which these results are displayed provides a zoom and pan capability. It also echos the exact x,y coordinate of the mouse.

The second set of results that can be displayed is a list of nodes which had voltages greater than the maximum allowable voltage. A menu can be invoked which shows the probe number of each probe in the circuit that exceeded the voltage limits. Selecting one of the probe numbers in the menu will show a matrix displaying the absolute maximum voltage on the probe for each simulation in which the voltage limit was exceeded.

Through these two options, the user may easily pinpoint problem areas in a model. This will greatly enhance the design and simulation process.

# CHAPTER 6

# PDSE - EXECUTION DESCRIPTION

This section describes in detail the three PDSE functions mentioned in previous sections. These are parameter calculations, convergence tests, and coupled line analyses.

## 6.1 Parameter Calculator

There are three steps that must be followed when it is desired to execute one of the parameter calculators. First, the transmission line cross-sectional geometry must be defined. Once the geometry specification is complete, the parameter calculator may be executed, then the results may be examined. This section describes in detail this process. If a geometry has already been defined and saved, data may be loaded using the LOAD button in the I/O WINDOW. Also, once a geometry specification is complete and the capacitance and inductance matrices obtained from the calculator, all the data may be saved using the SAVE button in the I/O WINDOW.

In the COMMAND WINDOW, The following buttons are selected:

| | |
|---|---|
| SELECT OBJECTIVE | This selects the desired objective. (e.g., Parameter Calculator, Convergence Test, or Coupled Line Analysis) |
| PARAMETER CALCULATOR | Parameter Calculator automatically selects UAMOM. (If it is desired to execute UAC, the SELECT TOOL option is chosen instead of SELECT OBJECTIVE.) |

To define Transmission Line Geometry, the following button is selected:

| | |
|---|---|
| EDIT DATA | This brings up the GEOMETRY menu. |

This will cause the menu in Figure 6.1 to be displayed. A description of each menu item is given below. ACTIVE CONDUCTORS allows the number of active conductors in the transmission line system to be defined. GROUND CONDUCTORS allows the number of ground conductors to be defined. The number of active plus the number of ground conductors must not exceed ten.

Once the number of conductors has been defined, the cross-sectional geometry of the transmission line system may be defined by selecting the CONDUCTOR GEOMETRY option. This will bring up a matrix i/o window through which the x,y coordinates of each vertex of each conductor may be specified. The active conductor geometries must be given first, followed by the ground conductors. For example, if there are three active and two ground conductors, the

active conductors would be defined in the first three columns and the ground conductors would be defined in the fourth and fifth columns.

If there is to be an infinite ground plane at y = 0, the GROUND PLANE @Y=0? option is selected and 'yes' is entered in the dialog box.

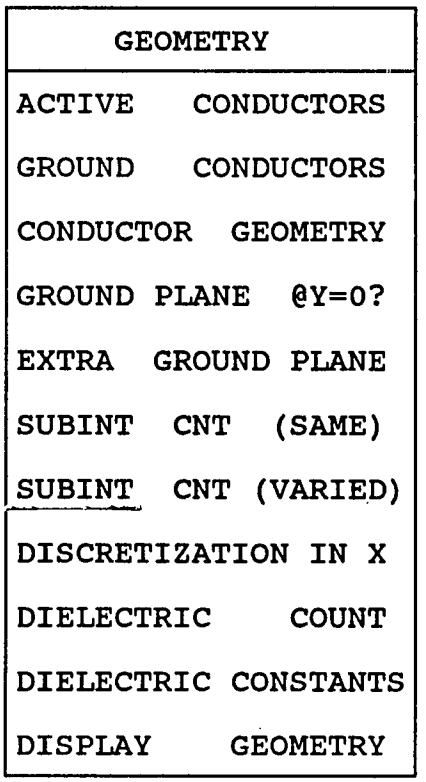



Figure 6.1 Geometry menu.

If there is to be an infinite ground plane at some location other than y = 0, the EXTRA GROUND PLANE option is selected, and the y coordinate of the ground plane is entered. This plane should have the highest y-value in the geometry.

The SUBINT CNT option is used to specify the number of

subintervals into which all conductor sides will be divided. (A default value will be assigned if not defined by the user.) The (SAME) option is used for UAMOM and the (VARIED) option is used for UAC.

The DISCRETIZATION IN X option is used to specify how far in the x direction the infinite planes will be discretized. (A default value will be selected if not defined by the user.) This option is used only for UAMOM.

Through DIELECTRIC COUNT, the number of dielectric layers in the system may be specified. The y coordinates of the dielectric interfaces and the dielectric constants of each layer are defined through the DIELECTRIC CONSTANTS option. A matrix-io box will appear in which these values may be placed. Column one in the matrix-io box is used for specifying the y coordinates of the dielectric interfaces. Column two is where the dielectric constants are entered, and column three is where the loss tangents are entered. The upper-most y-coordinate and dielectric constant are placed in row one. The last row should have 0 for the y-coordinate and a dielectric constant for the bottom layer.

DISPLAY GEOMETRY will graphically display the geometry.

If the SELECT TOOL option in the COMMAND WINDOW is used and if UAC is selected, the process of defining the transmission line geometry is the same as described above with the two exceptions: the SUBINT CNT (VARIED) option,

rather than the SUBINT CNT (SAME) option is used. PDSE prevents the latter from being used in this situation. This will permit a different number of subintervals to be specified for each conductor side. The second difference is that DISCRETIZATION IN X is not allowed for UAC.

Once the geometry has been specified, the parameter calculator is invoked by selecting the RUN TEST button in the COMMAND WINDOW. If the geometry is complex, or if UAC is being executed, there may be a short wait (several minutes) before the tool is finished executing. Upon completion, a message appears in the MESSAGE WINDOW stating that UAMOM or UAC has finished running. A beep will also be sounded to signal the completed execution.

## 6.2 Convergence Test

As was mentioned in the introduction, the convergence test will execute UAMOM and UAC for various numbers of subintervals per side to determine how many subintervals must be specified for the results to be within a certain percentage of the converged result. The convergence test is selected using the PC CONVERGENCE TEST option in the COMMAND WINDOW. (PC stands for Parameter Calculator). When this option is selected, the menu in Figure 6.2 appears.

SELECT TEST allows the user to choose between the subinterval test or the discretization test.

Just in case the results of the parameter calculator

converge too slowly, the MAXIMUM NUMBER OF TESTS option is be used to inform the system when to stop the convergence test.

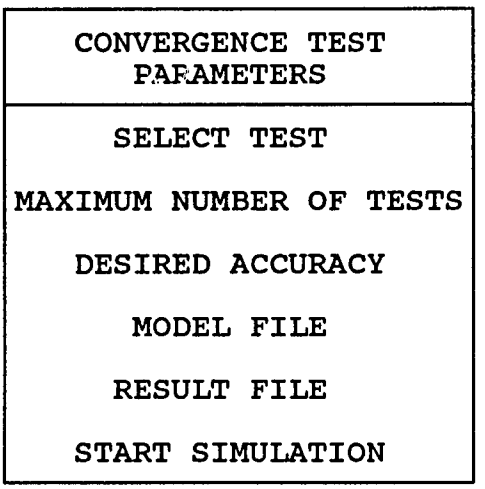



Figure 6.2 Convergence Test Parameter menu.

DESIRED ACCURACY defines how close the results must be to be considered converged.

The MODEL FILE is a file name of the geometry specification defined in the PARAMETER CALCULATOR section above.

RESULT FILE is the destination file of the convergence test results. An example of this file is in appendix E. Once the parameters above have all been specified, START SIMULATION is selected to begin the convergence test. Messages will appear in the MESSAGE WINDOW at the end of each execution of UAMOM or UAC to signal their completion.

## 6.3 Coupled-Line Analysis

The menu in Figure 6.3 comes up when the COUPLED LINE ANALYSIS button is selected. As was mentioned earlier, there are three steps to performing a coupled-line analysis -- create the model(s), create the model input(s), run the simulation.

The CREATE MODELS option will allow the user to specify or load from memory up to ten different circuit specifications. When this menu item is selected, PDSE will prompt the user with several questions using dialog boxes. First, the user will be asked how many models are to be created. A number from one to ten is to be entered in response. The base filename of these models will then be requested. (See section 4.3 for a description of file naming protocols.) (Pressing the 'ESC' key in any of these dialog boxes will cancel the model specification operation, and control will be returned to the COUPLED LINE ANALYSIS menu.) A menu titled MODEL NUMBER will appear, through which the user may select the number of the model which is to be created (modified). When the user selects a model number, PDSE asks if the file containing that model's specification is to be loaded. If so, the 'Accept' button in the dialog box is to be selected. If a model specification has already been defined and it is desired to copy this specification to another model number, just selecting the 'Abort' button will cause the model

definition process to begin with the data that exists in memory. The CIRCUIT SPECIFICATIONS menu will then come up. (see section 6.3.1) Once the model has been defined, and the CIRCUIT SPECIFICATIONS menu has been terminated, the user will be asked whether or not the new data should be saved. 'Accept' causes the data to be saved, and 'Abort' causes it to be discarded.

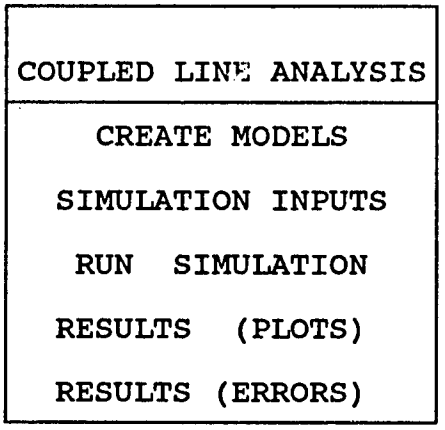



Figure 6.3 Coupled Line Analysis menu.

The SIMULATION INPUTS menu option will call the SIMULATION INPUTS menu through which the voltage source input waveforms and simulation constraints may be specified. (See section 6.3.5 for a description of that menu.)

Selecting the RUN SIMULATION option will cause PDSE to prompt the user for the following items through dialog boxes. If at any time it is desired to stop this query process, the 'Abort' button in one of the dialog boxes may be selected, and control will return to the COUPLED LINE

ANALYSIS menu. The parameters for which the user will be prompted are the number of models, the model filename (base name), the waveform filename, and the results file name (base name). Once these parameters have been specified, PDSE will begin the coupled-line analysis for each model and for each input waveform.

Once an analysis is complete, the results may be viewed through either the RESULTS (PLOTS) option or the RESULTS (ERROR) option. The PLOTS option will display a plot of the signal levels vs. time at the circuit probe points specified by the user in the CIRCUIT SPECIFICATIONS menu SELECT PROBE POINTS option (see section 6.3.1). After selecting the PLOTS option, the user will be prompted for the number of models, the number of waveforms, and the results file name (base filename). Three menus will appear along with the graph -- MODEL NUMBER, TEST NUMBER, and PROBE menus. Using these three menus, the plots of any simulation of any model may be displayed, and any combination of probe plots from one model and test may be displayed.

The RESULTS (ERRORS) option allows the user to determine which nodes in the circuit had voltage levels greater than some maximum allowable level as specified by the user in the SIMULATION INPUTS menu. This will allow the user to watch suspected trouble spots for unwanted transients and noise levels. If there were any nodes that

had voltages above the allowable level, the FAILED NODES menu will appear with the menu items being the node numbers of the troubled nodes. Selecting one of the nodes will display a matrix i/o window in which the rows are the model numbers, the columns are the test numbers, and the items in the matrix are the maximum voltages appearing on the noisy nodes.

### 6.3.1 Circuit Specifications

Selection of the CREATE MODELS option in the COUPLED LINE ANALYSIS menu will bring up the CIRCUIT SPECIFICATIONS menu following a series of questions to the user as described above. Through this menu, resistors, conductors, capacitors, inductors, coupled inductors, voltage sources, transmission line systems, and simulation probe points may be specified (see fig 6.4). When specifying any of the circuit components, node numbers must be given which describe how the components are connected. When connecting a node to ground, zero must be the node number. Any other node numbers may be any integer greater than zero. Each component must have two different node numbers assigned.

The first five menu items bring up a matrix i/o window through which the component value and the two node numbers may be specified. Resistor values are given in Ohms, Conductor values in Mhos, capacitors in pico-Farads, and inductors in nano-Henrys. Coupled inductors are defined by

giving the inductor number of each inductor, and the coupling coefficient K, where

$$K = \frac{M}{\sqrt{L_1 L_2}}$$

$L_1$ and $L_2$ are the self inductances of the two inductors involved, and M is their mutual inductance.

| CIRCUIT SPECIFICATIONS |
|---|
| RESISTORS |
| CONDUCTORS |
| CAPACITORS |
| INDUCTORS |
| COUPLED INDUCTORS |
| VOLTAGE SOURCES |
| TRANSMISSION LINES |
| SELECT PROBE POINTS |

Figure 6.4 Circuit Specifications menu.

The VOLTAGE SOURCE option will bring up the VOLTAGE SOURCE SPECIFICATIONS menu which is described in section 6.3.2. The TRANSMISSION LINES option will bring up the TRANSMISSION LINE SPECIFICATION menu which is described in section 6.3.3. The SELECT PROBE POINTS allows the user to specify the node numbers in the circuit for which the transient analysis will be performed.

### 6.3.2 Voltage Source Specifications

The VOLTAGE SOURCE SPECIFICATIONS menu has only two options. The first option allows the NUMBER OF SOURCES to be defined, the second option, DEFINE NODES, calls a matrix i/o window through which the positive and negative nodes of the voltage source may be defined. In order to enforce the separation of inputs from the model, no inputs are defined at this point. All the voltage source inputs are defined through the SIMULATION INPUTS option in the COUPLED LINE ANALYSIS menu.

### 6.3.3 Transmission Line Specifications

Up to ten transmission line systems may be specified for the circuit simulation. The menu in Figure 6.5 provides for their specification.

| TRANSMISSION LINE SPECIFICATIONS |
|---|
| NUMBER OF TL SYSTEMS |
| CONDUCTOR LENGTH/COUNT |
| DEFINE NODES |
| DEFINE TLINE GEOMETRY |
| COPY TL SYSTEM |

Figure 6.5 Transmission Line Specifications menu.

Through menu item one, the number of transmission line systems (between one and ten) may be defined. Through item two, the length of each transmission line system and the

number of lines in each system (between one and ten) are given. If either of the last two items are chosen, the user will be asked which transmission line systems to which the upcoming definition will be applied.

Through DEFINE NODES, the near and far end node numbers are defined. Through the DEFINE TLINE GEOMETRY option, the SELECT L & C SOURCE menu (see section 6.3.4) is invoked through which the capacitance and inductance matrices are defined. The COPY TL SYSTEM option provides for the duplicating of one transmission line system specification to various points in the circuit. When it is selected, the user will be asked for the original transmission line system number and the destination system number through two dialog boxes. The 'ESC' key may be pressed if the copy transaction is to be aborted. All values except the node numbers are copied in the transaction. Once a system has been copied, it is necessary to go back and define the node numbers of the new system.

### 6.3.4 Select Inductance and Capacitance Matrix Source

There are three sources from which the inductance and capacitance matrices may be obtained. The data may be loaded from a file that has been saved after executing a parameter calculator separately, it may be calculated at the same time the transmission line system is defined, or the values may be input 'by hand'. The menu in Figure 6.6

provides these options.

The LOAD GEOMETRY option will prompt the user for a file name, then will load the requested data from the file. EDIT GEOMETRY and CALCULATE MATRIX allow the data to be calculated, and the EDIT MATRIX selection provides the option to input the matrices by hand. The EDIT GEOMETRY selection will call the GEOMETRY menu through which the transmission line system cross-sectional geometry may be defined. This process is defined in section 6.1. The CALCULATE MATRIX option will calculate the inductance and capacitance matrices given the data that was defined in EDIT GEOMETRY. Note that it is not necessary to calculate these parameters at this time. All that is necessary is to define the transmission line geometry. PDSE will perform

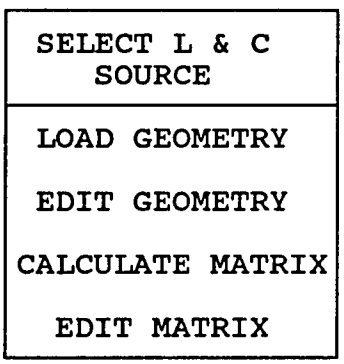



Figure 6.6 Select Inductance and Capacitance menu.

any necessary calculations to obtain the matrices when the simulation is running. If valid inductance & capacitance matrices have been calculated and the EDIT MATRIX selection is chosen, the matrices are invalidated because they no

longer correspond to the geometry definition. This invalidation will appear to the user in the EDIT MATRIX selection when either matrix has a zero value in the first row and first column.

### 6.3.5 Simulation Constraints

Once the model(s) has been defined, the model inputs and simulation constraints must be defined. Selecting the SIMULATION INPUTS option in the COUPLED LINE ANALYSIS menu will cause the following menu to be displayed:

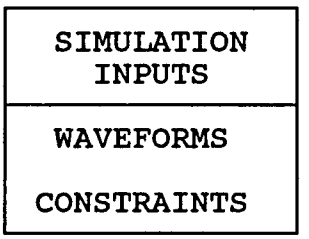



Figure 6.7 Simulation inputs definition menu.

The circuit voltage source inputs are defined through the WAVEFORMS option. This option allows the user to specify several (up to ten) step inputs, or up to ten waveforms that have up to ten linear segments. When this menu item is selected, a dialog box will ask the type of waveform. The waveform types available are referenced as 'step' and 'other'. If 'step' is selected, the STEP INPUT DEFINITION menu in Figure 6.8 will be displayed. If 'other' is selected, the GENERAL WAVEFORM DEFINITIONS menu is displayed. The descriptions of these menus is given in the following sections.

The CONSTRAINTS option provides for the user to define the length of each simulation. Through this option, the user may also specify which nodes in the circuit to monitor for a given voltage level.

### 6.3.6 Step Input Definitions

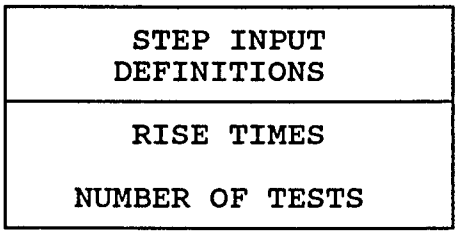



Figure 6.8 Step Input Definitions menu.

This menu allows the user to define up to ten step inputs that have the same voltage level but different rise times. Selection of the RISE TIMES option will bring up a matrix i/o window through which the fastest rise time, the slowest rise time, and the voltage level of the step is defined. Through the NUMBER OF TESTS option, the number of different rise times from the fastest to the slowest may be specified. For example, if it is desired to test ten different inputs between 0.1 ns and 1.0 ns, the fastest rise time would be 0.1 ns, the slowest would be 1.0 ns, and the number of test would be ten. PDSE would then execute the simulation ten times, once for each step input with rise times 0.1, 0.2, 0.3, . . . , 0.9, 1.0 ns.

### 6.3.7 General Waveform Input Definitions

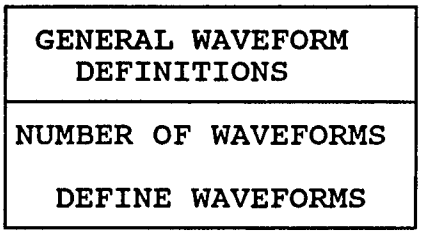



Figure 6.9 General Waveform Definitions menu.

NUMBER OF WAVEFORMS will call a dialog box through which the user will define the number of different waveforms there will be.

The DEFINE WAVEFORMS option will bring up the menu WAVEFORM # X which will provide for selective editing of each waveform, and for copying waveforms from one definition to another.

Once the waveform number has been selected, the matrix i/o window in Figure 6.10 will appear. In this window, the user can specify up to ten linear waveform segments that make up one input waveform. The time and the voltage of the beginning of each segment must be given. The window to the left describes the waveform below. It is assumed that the waveform remains at the level of the final segment. Figure 6.11 is a graph of the waveform described in Figure 6.10.

| WAVEFORM # X | | |
|---|---|---|
| segment number | time (sec) | voltage (V) |
| 1 | 0 | 0 |
| 2 | 1e-9 | 1 |
| 3 | 2e-9 | 1 |
| 4 | 3e-9 | 0 |
| 5 | 4e-9 | 0 |
| 6 | 5e-9 | -2 |
| 7 | 6e-9 | -2 |
| 8 | 7e-9 | 0 |
| 9 | 8e-9 | 0 |
| 10 | 9e-9 | 1 |

Figure 6.10 Waveform definition window.

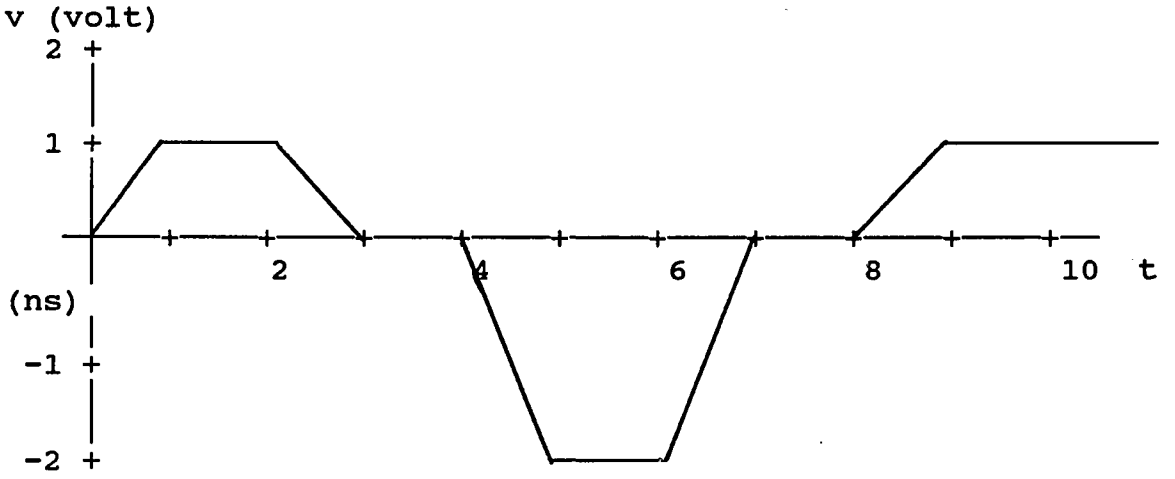



Figure 6.11 Sample input waveform.

### 6.3.8 Simulation Constraints

The CONSTRAINTS option in the SIMULATION INPUTS menu will bring up the menu below:

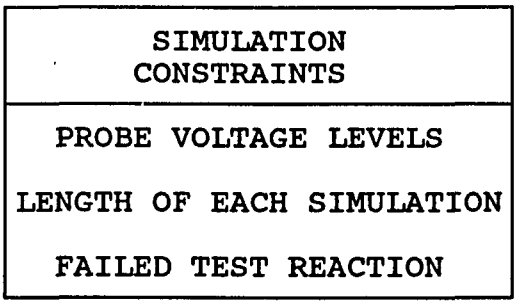



Figure 6.12 Simulation Constraints menu.

PROBE VOLTAGE LEVELS will bring up a matrix i/o window that displays two columns. In the first column the user may define the maximum voltage allowed at the node number given in the second column. The node numbers in the second column are those nodes for which the coupled-line analysis is to be performed. If it is not desired to monitor a node for any specific voltage level, a zero is entered for the maximum allowable voltage. LENGTH OF EACH SIMULATION will specify the length (in seconds) of each simulation. Through the FAILED TEST REACTION option, the user may specify whether or not the simulation should continue if one test yields a voltage level on some node that exceeds that node's maximum allowable voltage level.

# CHAPTER 7

## FUTURE WORK

PDSE as described in the previous chapters represents but a small portion of the envisioned final packaging design simulation tool. Currently, all the data required to run PDSE is input through the keyboard. It would be less tedious, however, if the input data were entered through some type of graphical editor. PDSE could then extract the information generated by the editor, determine how many conductors in the interconnect system are significant in the coupled-line analysis, select the best simulation tool based on the termination networks, then perform the coupled-line analysis. Figure 7.1 is a block diagram of the eventual product. The following sections discuss the steps mentioned above.

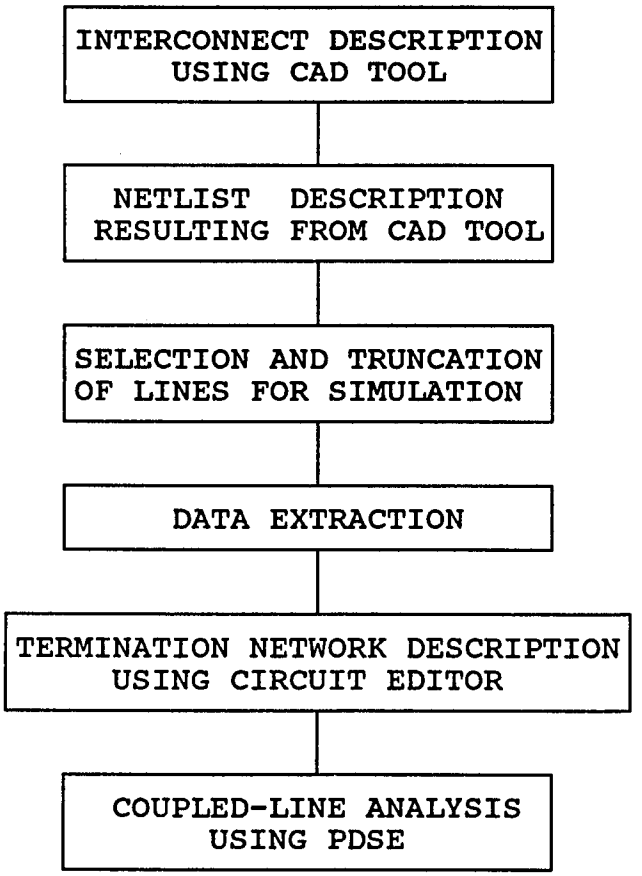



Figure 7.1 Block diagram describing eventual packaging design and simulation environment.

## 7.1 CAD file description

It is desirable to extract the interconnect information from a universal file format so any CAD tool which conforms to the file format may be used to specify the interconnect system. The IGES (Initial Graphics Exchange Specification) format is one such format that is widely used in industry and is supported by the National Bureau of Standards [6]. Although the IGES file format

provides many ways of representing objects to describe an interconnect system, the most basic entity is the line, which is described by x,y,z coordinates of each endpoint. The description of the lines are saved in a file in the order in which the lines were entered. Therefore, no relationships between one line and another can be assumed based on where the lines' descriptions are found in the file. It is assumed in this project that all lines are described in two-dimensional space (ie. z = 0).

## 7.2 Data Extraction

This section describes an algorithm which may be used to extract a description of an interconnect system from a CAD tool database. Several assumptions are made which limit the scope of the algorithm. If a more elaborate data extraction scheme is desired, geometric modeling techniques will probably need to be employed [7]. Useful information could also be obtained on this subject in books about VLSI design automation [8].

### 7.2.1 Interconnect System Characterization

An interconnect system can be described as a collection of transmission lines joined with one of several junction types. The lines may be uniform (non-tapered) or non-uniform (tapered.) The junctions will either be a bend, a discontinuity, a via, or a notch. (See Figure 7.2)

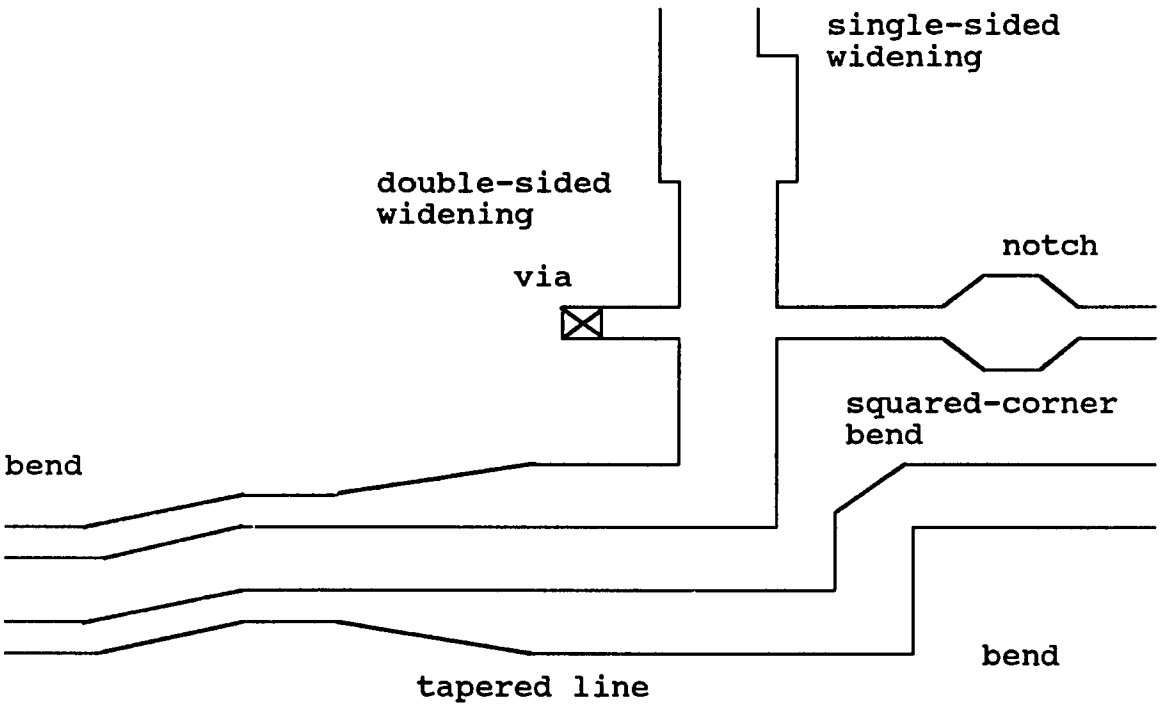



Figure 7.2 Sample interconnect line.

Other features of an interconnect system not shown are dielectric information, distance from the ground plane, cross-sectional geometry of the conductors, and whether or not the conductor is lossy. This information is provided by the user rather than the data base.

A system entity structure (SES) has been created (see Figures 7.3a and 7.3b) to illustrate the makeup of an interconnect system. The SES below can be used in creating knowledge based algorithms for performing the data extraction that is discussed in the following sections. This SES includes much information that is not included in a CAD tool data base. All that pertains to the data base

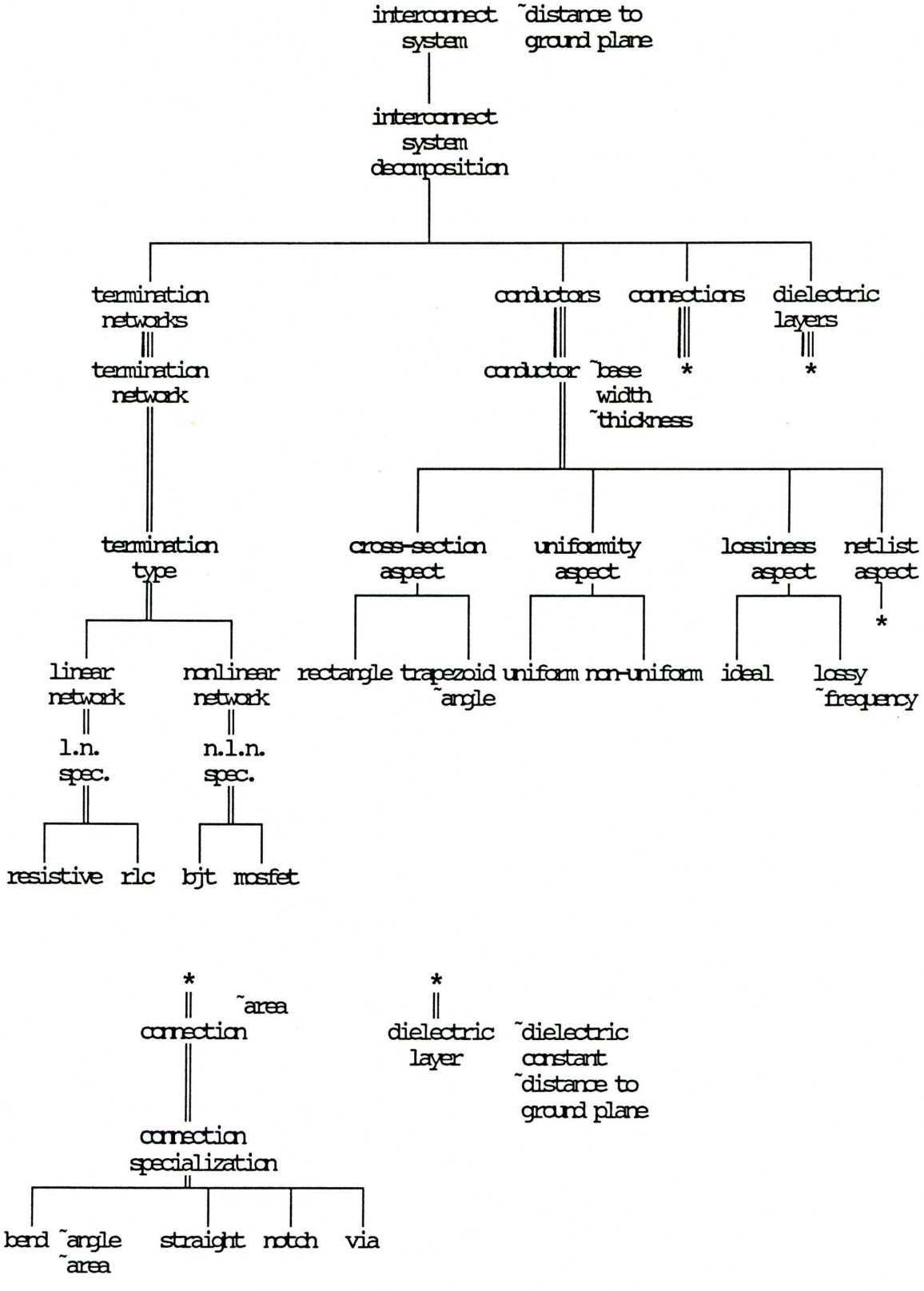



Figure 7.3 (a) Main System Entity Structure of an IC Interconnect System.

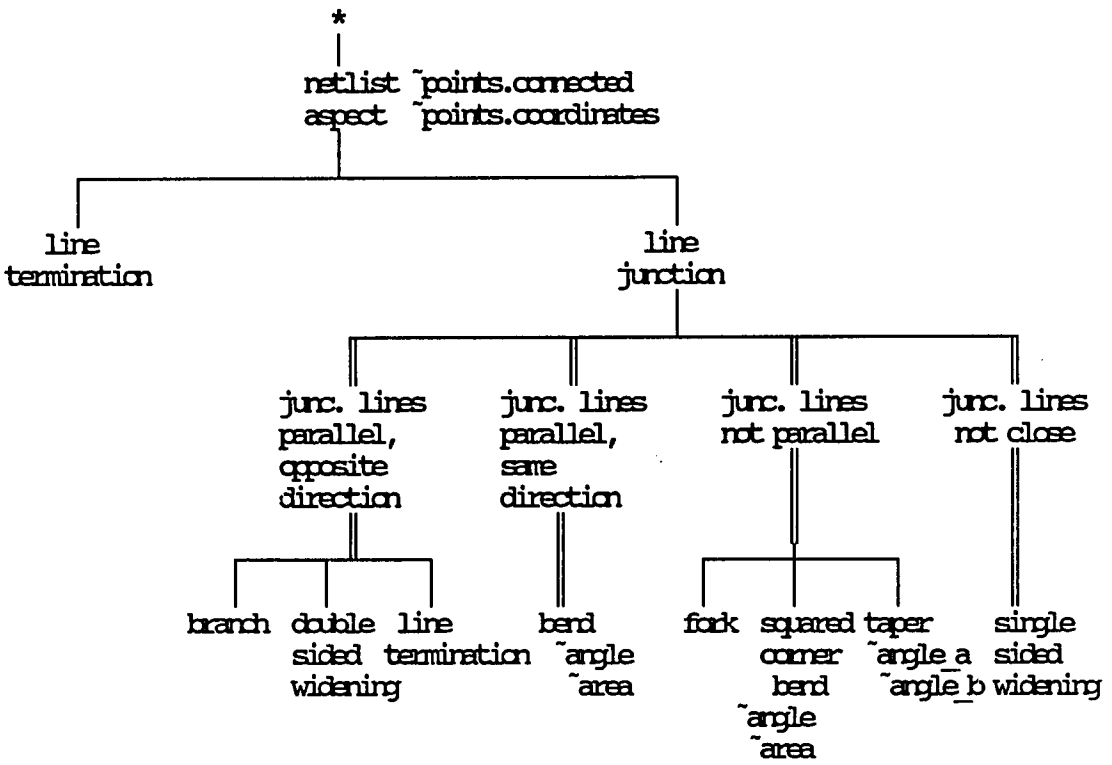



Figure 7.3 (b) Netlist aspect of IC interconnect system entity structure.

description is in the conductor.netlist_aspect and the connection specialty.

### 7.2.2 Data Extraction Assumptions

The simulation tools available at this time impose certain restrictions on the types of interconnect systems that can be simulated. Because of these limitations, several assumptions have been made which simplify the data extraction and simulation process. First, no tapered lines or tapered spaces between lines are allowed. No lines may branch into two lines. No vias are allowed. All conductor representations must be given in two-dimensional space.

The conductor thickness is required, but is assumed to be constant throughout the length of the conductor.

### 7.2.3 Data Extraction Preparation

This section describes the algorithm that is used to extract data that describes an interconnect system from a CAD tool data base. The first step is to read the interconnect description from the file and organize the information in a way that it can be analyzed. As was described in section 7.1, no relationship between any two lines can be assumed based on the order the descriptions are found in the data base. Relationships between lines are found by the end points (the two sets of x,y coordinates which describe the line.) Two lines that are connected will have similar end points. Based on this information, graph structures can be created using the line information. When a line is read from the database, the two endpoints are compared with endpoints existing in the graph. If a match is found, that line is added to the graph. Once all the lines have been read from the database, there should exist one graph structure for every conductor in the system. For example, if the conductor description in Figure 7.4 (a) is found in a database, the graph in Figure 7.4 (b) would result from reading in the data.

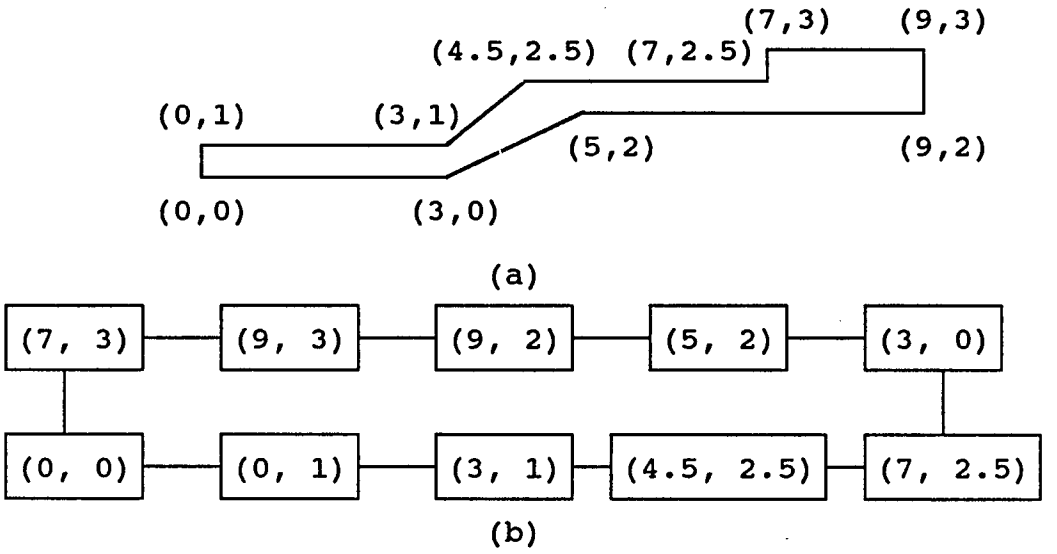



Figure 7.4 (a) Sample Conductor Description. (b) Graph resulting from the sample conductor.

Once the lists have been created, the layout is displayed on the terminal and the designer is prompted to identify which conductor is to be driven with a voltage source and which end of the conductor the voltage source will be applied. The next step is to determine how many conductors on both sides of the driven conductor must be considered in the coupled-line analysis. Garg has developed rules to perform this truncation analysis [9]. This truncation sets the stage for characterizing the conductor into transmission lines and junctions. A set of rules could be developed using the system entity structure in Figure 7.4 that would perform this characterization.

## 7.3 Circuit Editor

The data extraction process described above allows the

user to "import" a netlist description of an interconnect system into PDSE for simulation and analysis. In order to perform the simulations, it is necessary to describe the termination networks that will be applied to the interconnects. There are several methods through which this requirement may be met. The first method is to provide for text input of the circuit parameters. This is the method used by PDSE and described in chapter 6. This method, however, is tedious and prone to errors because there is no graphical feedback to verify the circuit that has been entered.

A more user-friendly method of describing the termination networks would be to use a computer aided design tool to graphically input the circuit. A tool could be developed expressly for PDSE for this purpose. The advantage of this is that the circuit description could easily be arranged into a format readable by PDSE. The disadvantage is that this tool development could become very time-consuming, and the final product would be very limited compared to a commercial tool.

Rather than designing a CAD tool to perform the data entry of the termination network, a commercial tool may be used. If this is done, it is advisable to design PDSE to accept data from a universal file format such as IGES so the user is not tied to any specific CAD tool.

## 7.4 Support of Package Design Cycle

The long term objective of PDSE development is to employ system design, artificial intelligence, and simulation/modeling techniques to support VLSI packaging design. PDSE will support activities of the design cycle as follows: The layout/routing, driver and receiver data will be obtained by accessing and/or editing and modifying the design data base. The layout will be translated into a geometrical representation. This representation will serve as an input to simulation tools. The simulation tools will be used with appropriate experimental frames to generate performance measures for models (layouts) under consideration. Simulation output will be analyzed, and if problems exist, the layout will be edited and simulations will be re-run for a new model. This process is explained in more detail below.

A simulation run will be set up using the library of simulation tools and experimental frames. Prior to applying an experimental frame to a design model, checks will be made to ensure data consistency and validity. This will require that a rule base containing knowledge about data ranges, limits, etc. be invoked. The simulation run itself will be constrained and channeled by design techniques and expertness built into the PDSE.

Simulation results will be sieved by invoking a knowledge base of design expertise. Such a knowledge base

will have to be constructed from results of research into design techniques and by eliciting information from literature and packaging experts. This knowledge will serve as a basis for determining whether the results of simulation studies are plausible or if they violate fundamental design performance expectations. If the results are not satisfying, the user will be able to use a scratch-pad editor to modify the layout and its parameters and re-run the simulation experiments.

To select the best possible design given often conflicting objectives (e.g., maximum performance vs. low cost), procedures must be developed for ranking alternative design models with respect to a set of performance measures. This will be accomplished by incorporating in PDSE trade-off procedures that will employ multiple-criteria decision making methods [10].

## 7.5 Data Analysis

Section 6.3 describes the one type of data analysis that is performed on the results of a coupled-line analysis. This method looks for a maximum allowable voltage (specified by the user) that appears on any given line. Other questions that may be answered by the appropriate analysis of the data are, "What is the settling time on line x?," "Does the voltage one line x monotonically cross some window threshold?," and "What is the voltage steady-state error on line x?." Providing

routines to analyze these questions and provide (preferably) graphical responses would greatly enhance the usefulness of this tool.

# CHAPTER 8

## CONCLUSIONS

Through effective use of the experimental frame concept, an efficient, user-friendly software shell has been developed to execute packaging simulation tools developed at the University of Arizona. Up till now, the simulation tools had to be executed directly by the user. Because the tools are neither robust nor easy use, their use was tedious and cumbersome. PDSE, however, provides a means whereby the user may easily input data required to execute the simulation tools. If there are errors in the input, PDSE will correct them if possible, or prompt the user to correct them if not. Because the models are separated from the inputs, several models may be tested with one input, several inputs may be tested on one model, or several models may be tested with several inputs. This allows for much flexibility in the simulation process.

Although PDSE, as described in this thesis, is a valuable tool, there is much yet to be done to enhance the system. Graphical input of models and inputs must be provided. This may be accomplished either by developing

graphical editors, or by developing data extractors to extract circuit and geometric descriptions from a CAD tool database. It is recommended that the latter route be taken. A knowledge-base may also be employed to select the appropriate simulation tools base on the model at hand and to analyze the simulation results and make recommendations as to which model is "best" or how to improve a given model to meet certain requirements.

The concepts above would provide a foundation for tools that will support the VLSI package design cycle.

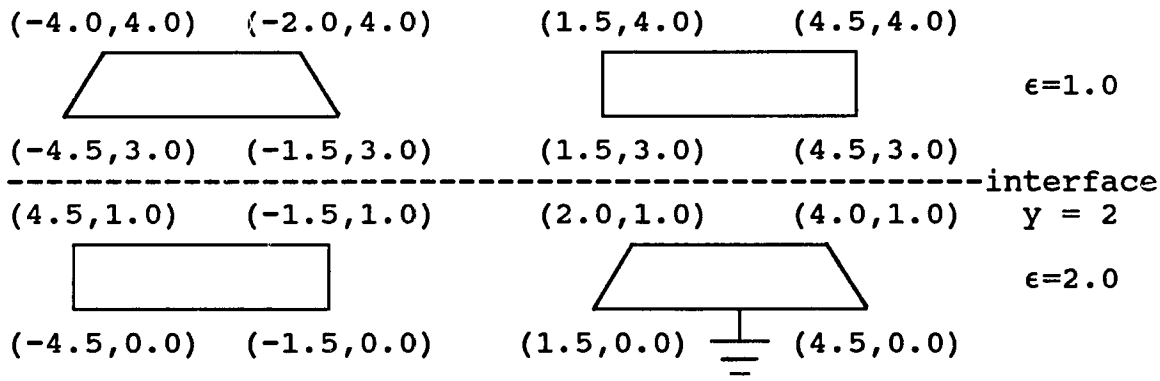
In summary, this will be accomplished by providing mechanisms for selecting a model representation given a design data base. The selection process will be driven by design constraints and objectives expressed using production rules. Models of designs will be evaluated in experimental frames using available simulation tools. A knowledge base of design expertise will be invoked to evaluate simulation results. If they violate fundamental design performance expectations, a model will be modified. There will be facilities for selecting satisfying design solutions given multiple design performance objectives.

# APPENDIX A

## CONVERGENCE STUDIES (Adapted from Scheinfein [6])

UAMOM in each of the tests below was executed with a maximum excursion in x = 8 mils.

Convergence Test #1: One Finite Ground Conductor.

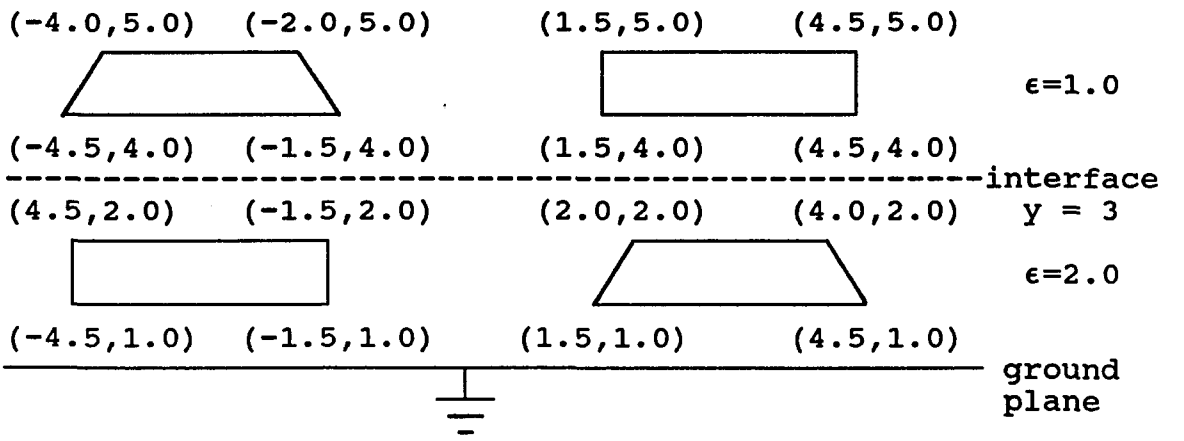



| TOOL | PERCENTAGE DEVIATION FROM CONVERGED VALUE | SUBINTERVALS PER SIDE | SIMULATION LENGTH |
|---|---|---|---|
| UAC | 1% | 1 | 35 s |
| UAMOM | 1% | 6 | 66 s |
| | 5% | 4 | 23 s |
| | 10% | 3 | 12 s |

Comparison of capacitance values calculated with UAMOM2.0 and UAC1.1 that are within one percent of the converged value for that tool.

| Capacitance | UAC(pf/cm) | UAMOM(pf/cm) |
|---|---|---|
| C(11) | 0.3978 | 0.39675 |
| C(12) | -0.07644 | -0.07681 |
| C(13) | -0.27706 | -0.2755 |
| C(22) | 0.4039 | 0.39695 |
| C(23) | -0.05792 | -0.05906 |
| C(33) | 0.53799 | 0.54002 |

Convergence Test #2: One Infinite Ground Plane.

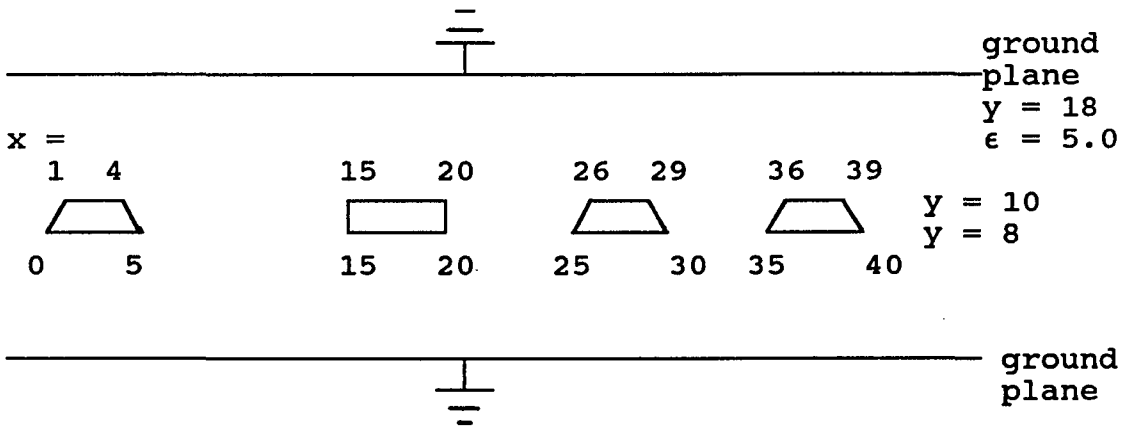



| TOOL | PERCENTAGE DEVIATION FROM CONVERGED VALUE | SUBINTERVALS PER SIDE | SIMULATION LENGTH |
|---|---|---|---|
| UAC | 1% | 1 | 99 s |
| UAMOM | 1% | 4 | 25 s |
| | 5% | 2 | 6 s |
| | 10% | 1 | 3 s |

Comparison of capacitance values calculated with UAMOM2.0 and UAC1.1 that are within one percent of the converged value for that tool.

| Capacitance | UAC(pf/cm) | UAMOM(pf/cm) |
|---|---|---|
| C(11) | 0.41238 | 0.40982 |
| C(12) | -0.065436 | -0.06526 |
| C(13) | -0.22676 | -0.22626 |
| C(14) | -0.017686 | -0.01759 |
| C(22) | 0.42321 | 0.41849 |
| C(23) | -0.02525 | -0.02522 |
| C(24) | -0.21298 | -0.21268 |
| C(33) | 1.2018 | 1.1949 |
| C(34) | -0.050823 | -0.0503 |
| C(44) | 1.1376 | 1.1326 |

Convergence Test #3: Two Infinite Ground Planes.




| TOOL | PERCENTAGE DEVIATION FROM CONVERGED VALUE | SUBINTERVALS PER SIDE | SIMULATION LENGTH |
|---|---|---|---|
| UAC | 1% | 2 | 96s |
| UAMOM | 17% | 7 | 214s |

Comparison of capacitance values calculated with UAMOM2.0 and UAC1.1 that are within one percent of the converged value for that tool.

| Capacitance | UAC(pf/cm) | UAMOM(pf/cm) |
|---|---|---|
| C(11) | 1.4976 | 1.4985 |
| C(12) | -0.13592 | -0.13326 |
| C(13) | -0.0021838 | -0.00191 |
| C(14) | -6.4825e-5 | -6.0e-5 |
| C(22) | 1.6914 | 1.6882 |
| C(23) | -0.34908 | -0.34952 |
| C(24) | -0.0069902 | -0.00614 |
| C(33) | 1.6594 | 1.6633 |
| C(34) | -0.32279 | -0.32200 |
| C(44) | 1.5648 | 1.5586 |

# APPENDIX B

## BASIC EMACS EDITOR COMMANDS

| | |
|---|---|
| move cursor right/left | right/left arrow keys |
| delete character at cursor | ctrl-d |
| delete character to the left of cursor | backspace key |
| end of line | ctrl-e |
| beginning of line | ctrl-a |
| delete from cursor to end of line | ctrl-k |

# APPENDIX C

## FORMAT OF DATA FILES CREATED BY PDSE

This section describes the format of the files created by PDSE. The three types of files described are model files, waveform files, and simulation result files. In each of the descriptions below, the format is shown in the way it is found in the file. The text given in brackets <> represents the number that would appear in that position in the file. The GEOMETRY SPECS and ELECTRICAL PARAMETERS sections at the beginning of the file are used to describe data for UAMOM and UAC. The TRANSMISSION LINE SYSTEMS section describes the data used for UACSL. Because UACSL requires information from UAMOM and UAC, the TRANSMISSION LINE SYSTEMS section has a GEOMETRY SPECS and ELECTRICAL PARAMETERS section for each transmission line system.

Model file description:

```
#------------------
# GEOMETRY SPECS
#------------------
# active conductor count
<ac_cnt>
# ground conductor count
<gc_cnt>
# ground at zero flag (1 = true, 0 = false)
<gp_at_zero>
# y coordinate of extra ground plane
<extra_gnd>
# number of dielectric interfaces
<diel_cnt>
# coordinates of each dielectric interface
<ydi[1]   ydi[2]  ...  ydi[diel_cnt]>
# number of sides on each conductor
<side_cnt[1]  side_cnt[2] ... side_cnt[ac_cnt+gc_cnt]>
```

```
# number of subintervals
<subint_cnt>
# x & y vertices of each conductor
# conductor number 1
< x[1,1], y[1,1]
  x[1,2], y[1,2]
   .
   .
  x[1,side_cnt[1]], y[1,side_cnt[1]]>
.
.
.
# conductor number <ac_cnt+gc_cnt>
< x[ac_cnt+gc_cnt, 1], y[ac_cnt+gc_cnt, 1],
  x[ac_cnt+gc_cnt, 2], y[ac_cnt+gc_cnt, 2],
   .
   .
   .
  x[ac_cnt+gc_cnt, side_cnt[ac_cnt+gc_cnt],
  y[ac_cnt+gc_cnt, side_cnt[ac_cnt+gc_cnt]] >

# maximum excursion in x
<max_excursion>
# number of subintervals per side per conductor
subints[1,1] subints[1,2] ... subints[1,side_cnt[1]]
subints[2,1] subints[2,1] ... subints[2,side_cnt[2]]
.
.
.
subints[ac_cnt+gc_cnt, 1]...subints[ac_cnt+gc_cnt,
                            side_cnt[ac_cnt+gc_cnt]] >
#----------------------
# ELECTRICAL PARAMETERS
#----------------------
# use losses flag
<ul_flag>

# row 1
# capacitance   cap w/o diel.   inductance   conductance
<  cap[1,1]      capw[1,1]       ind[1,1]     con[1,1]
    .
    .
    .
cap[1,ac_cnt] capw[1,ac_cnt] ind[1,ac_cnt] con[1,ac_cnt]>

# row 2
...
# row ac_cnt
# capacitance   cap w/o diel.   inductance   conductance
<cap[ac_cnt,1] capw[ac_cnt,1]  ind[ac_cnt,1] con[ac_cnt,1]
    .
    .
```

```
   .
cap[ac_cnt,ac_cnt] capw[ac_cnt,ac_cnt] ind[ac_cnt,ac_cnt]
                                        con[ac_cnt,ac_cnt]>
# dielectric parameters
# dielectric constant     loss tangent
< dc[1]                   lt[1]
  dc[2]                   lt[2]
   .
   .
   .
  dc[diel_cnt]             lt[diel_cnt] >

#-----------------------------
# TRANSMISSION LINE PARAMETERS
#-----------------------------
# total time of simulation (s),  sampling period (s)
  <sim_len>                    <speriod>
# number of probe points
<probe_cnt>
# probe points to be simulated
< probe[1]  probe[2] ... probe[probe_cnt]

# RESISTORS
# number of resistors
<res_cnt>
# element number    element value   node     node
<1                  res_val[1]      rn_a[1] rn_b[1]
 2                  res_val[2]      rn_a[2] rn_b[2]
.
.
.
 res_cnt       res_val[res_cnt] rn_a[res_cnt] rn_b[res_cnt]
>
# CONDUCTORS
. <same format as resistors>
.
.
# INDUCTORS
. <same format as resistors>
.
.
# CAPACITORS
. <same format as resistors>
.
.
# COUPLED INDUCTORS
. <same format as resistors>
.
.
# VOLTAGE SOURCES
# number of sources
<vs_cnt>
```

```
# source number 1
# positive node    negative node
  < vs_pnode        vs_nnode>
# source number 2
# positive node    negative node
  < vs_pnode        vs_nnode>
.
.
.
#source number <vs_cnt>
# positive node    negative node
  < vs_pnode        vs_nnode>

# TRANSMISSION LINE SYSTEMS
# number of systems
<tlsys_cnt>

# system number 1
# line length
<tline_len[1]>
# number of conductors in the system
<tlc_cnt[1]>
# near nodes      far nodes
<tlnn[1,1]        tlfn[1,1]
 tlnn[1,2]        tlfn[1,2]
  .
  .
  .
 tlnn[1,tlc_cnt[1]] tlfn[1,tlc_cnt[1]] >
#----------------
# GEOMETRY SPECS
#----------------
<see format above>
#----------------------
# ELECTRICAL PARAMETERS
#----------------------
<see format above>

# system number 2
.
.
.
# system number <tlsys_cnt>
```

Waveform File Description:

```
# INPUT WAVEFORMS FOR PDSE
# WAVE TYPE
<wave_type>
# ACTION WHEN CONSTRAINTS ARE NOT MET
<action>
# TOTAL TIME OF ANY ONE SIMULATION
<sim_time>
# NUMBER OF STEP INPUTS
<step_cnt>
# MAXIMUM VOLTAGE ALLOWED ON INDIVIDUAL NODES (VOLTS):
< mv[1]   mv[2]   mv[3] ... mv[node_cnt] >
# INITIAL RISETIME
< init_rt >
# FINAL RISETIME
< final_rt >
# STEP VALUE
< step_val >
# NUMBER OF WAVEFORM DEFINITIONS
<wave_cnt>
# WAVEFORM 1 OF <wave_cnt>
# NUMBER OF LINEAR VOLTAGE SEGMENTS
<vseg_cnt[1]>
# time (sec)        voltage (v)
<t[1,1]              v[1,1]
 t[1,2]              v[1,2]
  .
  .
  .
 t[1,vseg_cnt[1]]   v[1, vseg_cnt[1]] >
# WAVEFORM 2 OF <wave_cnt>
# NUMBER OF LINEAR VOLTAGE SEGMENTS
<vseg_cnt[2]>
# time (sec)        voltage (v)
<t[2,1]              v[2,1]
 t[2,2]              v[2,2]
  .
  .
  .
 t[2,vseg_cnt[2]]   v[2, vseg_cnt[2]] >
# WAVEFORM <wave_cnt> of <wave_cnt>
.
.
.
```

Simulation Result File Description:

The simulation result file is in the format described in the UACSL user's guide.[11]

# APPENDIX D

## PDSE EXPERIMENTAL FRAME DEFINITION

EF = <T, I, O, C, $\Omega_i$, $\Omega_c$, S>

T (time base) : Second

I (input variables) :
Voltage waveform defined by up to ten piecewise linear segments.

$\Omega_i$ (input segment) :
Functions which obtain the desired waveform from the user and convert the input to a form readable by the simulator.

O (output variables) :
Every node, of every model and simulation will have the following variables:
($v_1$, $v_2$, ..., $v_{100}$) at ($t_1$, $t_2$, ..., $v_{100}$) where
$v_i$ = voltage at time step i
$t_i$ = time (seconds) of time step i
$t_1$ = 0 s
$t_{100}$ = T sec (max simulation time)
$t_2 - t_1 = t_3 - t_2 = t_n - t_{n-1}$

C (control variables) :
Time (t)
($vm_1$, $vm_2$, ..., $vm_{10}$) - maximum allowable voltage on each node
sim_num - total number of simulations
stop/run_flag - flag which signals whether the simulation is to stop or continue running when an error is encountered

$\Omega_c$ (control segments) :
For each simulation, C1 maps T -> T such that the simulation lasts only T seconds.
For a set of simulations (>1 model or >1 input waveform)
C2 maps sim_num -> ($vm_1$, $vm_2$, ..., $vm_{10}$) if stop/run_flag = stop, and
sim_num -> if stop/run_flag = run

S (summary variables) :

($v_1$, $v_2$, ..., $v_{100}$) voltage of each node at each time step, for each simulation

($fv_1$, $fv_2$, ..., $fv_{10}$) maximum voltage of nodes for each simulation in which the voltage exceeded the maximum allowable voltage

Auxiliary Variables :

($tmax_1$, $tmax_2$, ..., $tmax_{10}$) temporary maximum voltage for each node of a model

# APPENDIX E

## EXAMPLE OF CONVERGENCE TEST RESULTS

An example of the file created by a convergence test in PDSE is given below. The convergence test is first run on UAC then is run on UAMOM. Directly above the test results, the test number, the number of subintervals per side, and the CPU time are given. Column one of the test results has the test number. Column two and column three give the capacitance and inductance values (respectively). Columns four and five show the percent change from the previous test for the respective capacitance and inductance. The last two columns show the percent change between the capacitance or inductance of that row from the final capacitance or inductance. Following the convergence test results, a complete copy of the model file on which the test was executed.

Convergence test running UAC on the data in data/conv1

| Test | # of subintervals | CPU Time (seconds) |
|---|---|---|
| 1 | 1 | 34.8653 |
| 2 | 2 | 140.478 |
| 3 | 3 | 346.369 |

| Test | C( 0, 0) | L( 0, 0) | %change | | %change from converged value | |
|---|---|---|---|---|---|---|
| 1 | 0.3978 | 6.862 | N/A | | %0.7121 | %0.55391 |
| 2 | 0.40017 | 6.8304 | %0.59325 | %0.46278 | %0.11956 | %0.090704 |
| 3 | 0.40065 | 6.8242 | %0.11956 | %0.090704 | %0 | %0 |
| Test | C( 1, 0) | L( 1, 0) | %change | | %change from converged value | |
| 1 | -0.07644 | 2.432 | N/A | | %0.84227 | %0.51083 |
| 2 | -0.076985 | 2.4216 | %0.70818 | %0.42617 | %0.13505 | %0.084308 |
| 3 | -0.07709 | 2.4196 | %0.13505 | %0.084308 | %0 | %0 |
| Test | C( 1, 1) | L( 1, 1) | %change | | %change from converged value | |
| 1 | 0.4039 | 4.2719 | N/A | | %0.55006 | %0.54629 |
| 2 | 0.40577 | 4.2524 | %0.46085 | %0.45786 | %0.089627 | %0.088023 |
| 3 | 0.40613 | 4.2487 | %0.089627 | %0.088023 | %0 | %0 |
| Test | C( 2, 0) | L( 2, 0) | %change | | %change from converged value | |
| 1 | -0.27706 | 4.5108 | N/A | | %0.76862 | %0.50779 |
| 2 | -0.27884 | 4.4917 | %0.63943 | %0.42634 | %0.13002 | %0.081108 |
| 3 | -0.2792 | 4.488 | %0.13002 | %0.081108 | %0 | %0 |
| Test | C( 2, 1) | L( 2, 1) | %change | | %change from converged value | |
| 1 | -0.05792 | 2.0304 | N/A | | %0.10348 | %0.61398 |
| 2 | -0.057972 | 2.02 | %0.091249 | %0.51386 | %0.012246 | %0.099609 |
| 3 | -0.05798 | 2.018 | %0.012246 | %0.099609 | %0 | %0 |
| Test | C( 2, 2) | L( 2, 2) | %change | | %change from converged value | |
| 1 | 0.53799 | 6.2842 | N/A | | %0.64362 | %0.56442 |
| 2 | 0.54089 | 6.2546 | %0.53763 | %0.4731 | %0.10657 | %0.090898 |
| 3 | 0.54147 | 6.2489 | %0.10657 | %0.090898 | %0 | %0 |

Convergence test running UAMOM on the data in data/conv1

| Test | # of subintervals | CPU Time (seconds) |
|---|---|---|
| 1 | 1 | 2.69989 |
| 2 | 2 | 5.53311 |
| 3 | 3 | 12.1328 |
| 4 | 4 | 23.3991 |
| 5 | 5 | 40.8984 |
| 6 | 6 | 66.0307 |

| Test | C( 0, 0) | L( 0, 0) | %change | | %change from converged value | |
|---|---|---|---|---|---|---|
| 1 | 0.38029 | 5.6269 | N/A | | %4.6558 | %7.5562 |
| 2 | 0.38911 | 5.3849 | %2.2667 | %4.4929 | %2.4445 | %2.9316 |
| 3 | 0.39441 | 5.3067 | %1.3438 | %1.4744 | %1.1157 | %1.4361 |
| 4 | 0.39675 | 5.2694 | %0.58979 | %0.70882 | %0.52901 | %0.72216 |
| 5 | 0.39803 | 5.2466 | %0.32159 | %0.4338 | %0.20809 | %0.28711 |
| 6 | 0.39886 | 5.2316 | %0.20809 | %0.28711 | %0 | %0 |
| Test | C( 1, 0) | L( 1, 0) | %change | | %change from converged value | |
| 1 | -0.07252 | 2.0081 | N/A | | %6.0013 | %46.655 |
| 2 | -0.07316 | 1.5879 | %0.87479 | %26.459 | %5.1717 | %15.97 |
| 3 | -0.07542 | 1.4765 | %2.9966 | %7.544 | %2.2424 | %7.8349 |
| 4 | -0.07636 | 1.4239 | %1.231 | %3.6991 | %1.024 | %3.9883 |
| 5 | -0.07681 | 1.3911 | %0.58586 | %2.3535 | %0.4407 | %1.5972 |
| 6 | -0.07715 | 1.3693 | %0.4407 | %1.5972 | %0 | %0 |
| Test | C( 1, 1) | L( 1, 1) | %change | | %change from converged value | |
| 1 | 0.38161 | 4.0466 | N/A | | %4.8164 | %19.275 |
| 2 | 0.39174 | 3.621 | %2.5859 | %11.754 | %2.2897 | %6.7303 |
| 3 | 0.39695 | 3.504 | %1.3125 | %3.3417 | %0.99022 | %3.2791 |
| 4 | 0.39912 | 3.4486 | %0.5437 | %1.6035 | %0.44897 | %1.6491 |
| 5 | 0.40025 | 3.4152 | %0.28232 | %0.98033 | %0.16712 | %0.6623 |
| 6 | 0.40092 | 3.3927 | %0.16712 | %0.6623 | %0 | %0 |
| Test | C( 2, 0) | L( 2, 0) | %change | | %change from converged value | |
| 1 | -0.26562 | 3.5562 | N/A | | %4.3052 | %9.4275 |
| 2 | -0.27362 | 3.3716 | %2.9238 | %5.4763 | %1.4231 | %3.746 |
| 3 | -0.2755 | 3.3099 | %0.68239 | %1.8632 | %0.74576 | %1.8484 |
| 4 | -0.27652 | 3.2805 | %0.36888 | %0.89866 | %0.37828 | %0.94127 |
| 5 | -0.27717 | 3.2624 | %0.23451 | %0.55328 | %0.14411 | %0.38586 |
| 6 | -0.27757 | 3.2499 | %0.14411 | %0.38586 | %0 | %0 |
| Test | C( 2, 1) | L( 2, 1) | %change | | %change from converged value | |
| 1 | -0.05847 | 1.2458 | N/A | | %1.8466 | %33.936 |
| 2 | -0.05834 | 1.0451 | %0.22283 | %19.203 | %2.0648 | %12.36 |
| 3 | -0.05906 | 0.98717 | %1.2191 | %5.8663 | %0.85614 | %6.1336 |
| 4 | -0.05934 | 0.95929 | %0.47186 | %2.9063 | %0.3861 | %3.1362 |
| 5 | -0.05949 | 0.94196 | %0.25214 | %1.8398 | %0.1343 | %1.273 |
| 6 | -0.05957 | 0.93012 | %0.1343 | %1.273 | %0 | %0 |
| Test | C( 2, 2) | L( 2, 2) | %change | | %change from converged value | |
| 1 | 0.52126 | 6.8199 | N/A | | %3.4739 | %6.9961 |
| 2 | 0.53578 | 6.5538 | %2.7101 | %4.0601 | %0.78515 | %2.8215 |
| 3 | 0.5375 | 6.4615 | %0.32 | %1.4297 | %0.46664 | %1.3721 |
| 4 | 0.53867 | 6.4177 | %0.2172 | %0.68093 | %0.24999 | %0.68654 |
| 5 | 0.53947 | 6.3917 | %0.1483 | %0.40725 | %0.10184 | %0.27816 |
| 6 | 0.54002 | 6.374 | %0.10184 | %0.27816 | %0 | %0 |

# REFERENCES

[1] B.P. Zeigler, Multifaceted Modelling and Discrete Event Simulation, Academic Press, New York, 1984.

[2] J.W. Rozenblit and B.P. Zeigler, "Design and Modeling Concepts," in International Encyclopedia of Robotics Applications and Automation, John Wiley & Sons, Inc., New York, 1988.

[3] J.W. Rozenblit, "A Conceptual Basis For Integrated, Model-Based System Design," Technical Report, University of Arizona, Tucson, Arizona, 1986.

[4] Michael R. Scheinfein, "Method of Moments TEM Transmission Line Parameter Calculator," User's Guide, University of Arizona, Tucson, Arizona 1986.

[5] J. C. Liao, et. al., "UAC -- University of Arizona Capacitance Calculator User's Guide," University of Arizona, Tucson, Arizona, 1986.

[6] U.S. Department of Commerce, National Bureau of Standards. Initial Graphics Exchange Specification (IGES), Version 3.0. Society of Automotive Engineers, Inc. April 1986.

[7] Michael E. Mortenson, Geometric Modeling, John Wiley & Sons, Inc., New York, 1985.

[8] Bryan T. Preas and Michael J. Lorenzetti, eds., Physical Design Automation of VLSI Systems, The Benjamin/ Cummings Publishing Company, Inc., 1988.

[9] Nitin K. Garg, "Analysis Domain Truncation of Interconnections in Multilayer Packaging Structures," Master's Thesis, University of Arizona, Tucson, Arizona, 1989.

[10] J.W. Rozenblit, et. al., "Computer Aided Design System for VLSI Interconnections," To appear in IEEE International Conference on Computer Design, October 1989.

[11] J. C. Liao, O. A. Palusinski, and J. L. Prince, "UACSL -- University of Arizona Coupled Line Simulator With Linear Terminations User's Guide," University of Arizona, Tucson, Arizona, 1988.