High-Speed Printed Circuit Board Analysis and Simulation in a Workstation Environment

Joseph Hall
Sasan Ardalan
M. S. Basel
R. Pomerleau*
Donna Riddle
Micheal Steer

Center for Communications and Signal Processing
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, North Carolina 27695-7914
Tel (919) 737-2336

*Bell Northern Research
Research Triangle Park, North Carolina

Abstract

Performance of digital systems on printed circuit boards (PCBs) is currently limited by transmission line effects and not by the technology of digital devices. CAPNET, an interactive, user-friendly graphical tool for the simulation and analysis of complex transmission line systems, is used to investigate high-speed pulse transmission on PCBs. CAPNET supports both experimental and analytic models of microstrips and most PCB structures, such as bends, tees and vias. In this paper an overview of CAPNET and examples illustrating propagation effects on printed circuit boards will be presented.

Proceedings RF Expo East 88, October 1988

1. Introduction

Traditional computer aided design of printed circuit boards (PCBs) begins with a functional schematic from which components are manually or automatically placed and routed on a multi-layer substrate. The automatic routing routines work well for clock speeds up to 20 or 30 MHz by using a set of design rules that include limiting the maximum conductor length, establishing trace widths, and maintaining a minimum separation of traces. These design rules control many of the transmission line effects of importance by determining the relative amplitudes of forward and backward traveling wave components of a signal. The initial forward traveling wave is generated by the line driver and backward waves result from reflections at mismatched transmission line/device interfaces. Subsequently the backward waves also reflect resulting in additional forward traveling wave component. Generally drivers can not establish the required voltage levels with the initial forward traveling wave (especially when the characteristic impedance of a line is low enough that reflections at the driver interface are minimized) and so backward traveling wave components are essential in developing the correct operating voltage on the line.

Understanding the presence of forward and backward traveling waves is essential in analyzing transmission line effects and in establishing design guidelines.

We can now refer back to the design guidelines and discuss them in the context of wave components. Limiting conductor length limits the maximum delay of the forward and backward wave components. Fixing the width of traces determines the characteristic impedance of the line (if the dielectric properties are set) and, because of transmission line/device mismatch, determines the relative amplitudes of the forward and backward traveling components and thus the number of reflections required to establish voltage levels on

the transmission line. Thus limiting conductor lengths and fixing the width of traces determines the maximum signal propagation delay. The third design guideline --- minimum trace spacing --- limits crosstalk between adjacent traces.

Design of PCBs with circuits operating at clock speeds above 30 MHz or so requires more sophisticated design guidelines as well as post layout simulation. Only simulation of the transmission line networks will provide the required confidence in design. This is a significant departure from lower frequency design where it is usually sufficient to treat a transmission lines as a lumped capacitance or perhaps by a delay, if the transmission lines need be considered at all. Simulation of transmission line structures can be achieved by

1. using linear RF/microwave techniques with linear approximations of digital devices,

2. by modeling the transmission line as a lossless bidirectional delay line and using a circuit level simulator like SPICE,

3. by modeling the line as a set of lumped element subcircuits and again using a program like SPICE,

4. or through convolution techniques using the impulse response of the transmission line network obtained as the Fourier transform of its frequency domain characteristic. This has also been implemented in a circuit level simulator.

The first approach is available with commercial RF and microwave simulation packages but is restricted to linear approximations of digital devices --- a serious limitation. As the last three techniques can be incorporated in circuit level simulation packages, the actual nonlinear characteristics of digital devices can be incorporated. These latter techniques all yield useful design information but are prohibitively slow when analyzing large portions of the transmission line network on a PCB. This slow simulation speed results from the use of matrix techniques and the treatment of all digital devices as though they were instantaneously coupled. However on a PCB a transmission line network is arranged as a tree and digital devices are not instantaneously coupled because of the finite delay along interconnecting transmission lines.

Transmission line networks on a PCB form a tree rather than a mesh and considerable simulation efficiency can be achieved by avoiding the matrix techniques required to handle a mesh (as implemented in circuit level simulators). The technique we present in this paper achieves simulation efficiency through recursive analysis of a transmission line tree.

The paper is organized as follows. First a review of transmission line theory is presented. Next, a technique for recursivly solving transmission line networks is described [5]. A recursive tree data structure and algorithms are then presented. Finally, a review of CAPNET, the CAD tool for complex transmission line networks is presented along with example sessions on a DEC workstation.

2. Theory

In this section we review relevant transmission line theory and introduce a recursive algorithm for solving transmission line networks. The method has the advantage that the solution to all nodes in the system is simultaneously obtained.

2.1 Transmission Line Networks

Consider the basic problem of simulating pulse transmission through a loaded transmission line. Assuming that the pulse is bandlimited with a cutoff frequency of f_c , we can obtain the pulse response by computing the inverse FFT of the complex multiplication of the frequency response of the pulse and the transmission line network. Therefore, as a first step in calculating the frequency response of the network, we analyze the network response to a single sinusoid of frequency f_0. Consider the loaded transmission line connected to the generator E_g through a source impedance Z_s as shown in Fig. 1 .

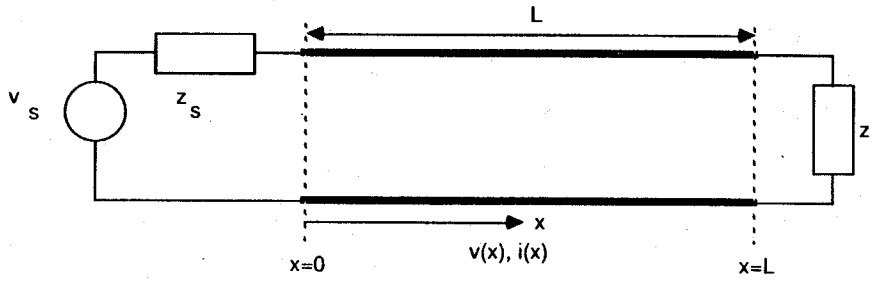

Figure 1. Generator connected to loaded transmission line

The voltage and current at any point on the transmission line can be obtained from the following expressions [2]:

$$v(x) = \frac{v_s z_0}{z_0 + z_s} e^{-\gamma x} \frac{1 + \Gamma_L e^{-2\gamma(L-x)}}{1 - \Gamma_s \Gamma_L e^{-2\gamma L}} \tag{1}$$

$$i(x) = \frac{v_s}{z_0 + z_s} e^{-\gamma x} \frac{1 - \Gamma_L e^{-2\gamma(L-x)}}{1 - \Gamma_s \Gamma_L e^{-2\gamma L}} \tag{2}$$

In the above expressions

$$\gamma = \sqrt{(r + j\omega l)(g + j\omega c)} \tag{3}$$

is the propagation constant and

$$z_0 = \sqrt{\frac{r + j\omega l}{g + j\omega c}} \tag{4}$$

is the characteristic impedance of the transmission line. The expressions for the source and load reflection coefficients are,

$$\Gamma_L = \frac{z_L - z_0}{z_L + z_0} \tag{5}$$

$$\Gamma_s = \frac{z_s - z_0}{z_s + z_0} \tag{6}$$

The expression for $v(x)$ includes the superposition of all waves reflecting from the source and load mismatches. This can be seen by a Taylor series expansion of (1)

$$v(x) = \frac{v_s z_0}{z_0 + z_s} [\, e^{-\gamma x} + \Gamma_L e^{-\gamma(L-x)} + \Gamma_L \Gamma_s e^{-\gamma(2L+x)} + \Gamma_L^2 \Gamma_s e^{-\gamma(3L-x)} + \Gamma_L^2 \Gamma_s^2 e^{-\gamma(3L+x)} + \ldots \,] \tag{7}$$

To obtain the shape of the pulse at the load we evaluate $v(L)$ at frequencies from $f=0$ to $f=fc$ in discrete steps where f_c is the cutoff frequency of the bandlimited pulse. The number of points must be a power of 2 such that the inverse FFT may be used to obtain the sampled pulse response at the load.

Consider now the case where the boundary voltage and current are known on a section of transmission line. See Fig. 2. Evaluate $v(0)$ in (4) and then compute.

$$\frac{v(x)}{v(0)} = e^{-\gamma x}\frac{1+\Gamma_L e^{-2\gamma(L-x)}}{1+\Gamma_L e^{-2\gamma L}} \tag{8}$$

Also

$$\frac{i(x)}{i(0)} = e^{-\gamma x}\frac{1-\Gamma_L e^{-2\gamma(L-x)}}{1-\Gamma_L e^{-2\gamma L}} \tag{9}$$

Thus using (8) and (9) the voltage and current can be evaluated at any point on the transmission line given the boundary voltage and current.

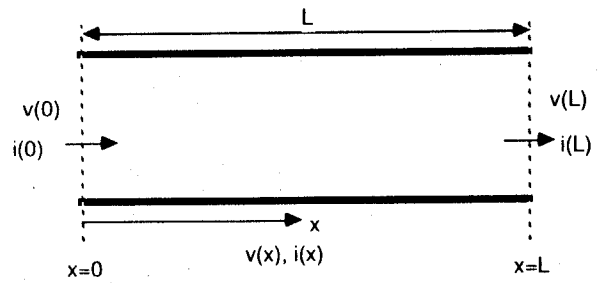

Figure 2. Section of transmission line with boundary voltages and currents

With the above preliminaries, we will examine the simple network in Fig. 3 and present a methodology for its solution. In Fig. 3, the nodes have been labeled n_1 through n_5. To solve this network, that is to obtain the voltage and current at each node and at any location within the network, consider equation (1). This equation suggests that if the impedance at node n_1 was known then the voltage and current at node n_1 can be calculated from the generator and source impedance. Thus the first step is to obtain the impedance at n_1. This impedance is seen to consist of the parallel combination of the impedance looking into n_5 and n_2 from n_1.

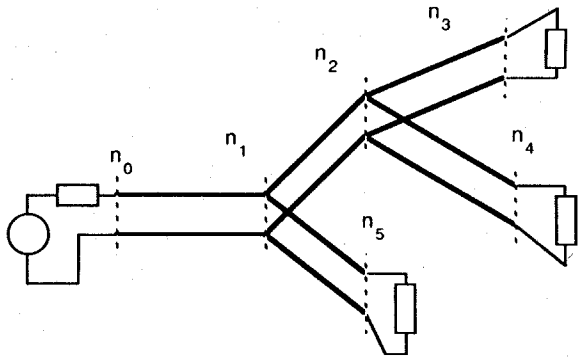

Figure 3. Example transmission line network

These impedances can be obtained by noting that (Fig. 4),

$$Z_{in}(x) = \frac{1+\Gamma_L e^{-2\gamma(L-x)}}{1-\Gamma_L e^{-2\gamma(L-x)}} Z_0 \tag{10}$$

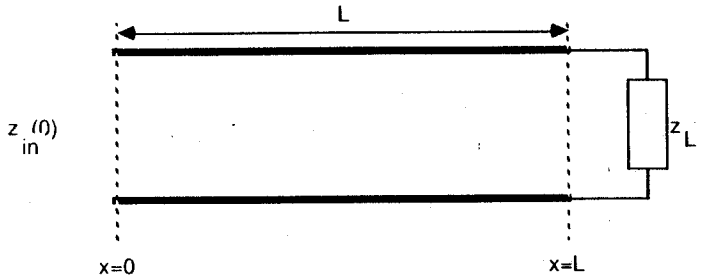

Figure 4. Input impedance of a loaded transmission line.

Thus, the first step is to calculate the impedances looking into n_3 and n_4 from n_2. The parallel combination forms the impedance at n_2. The impedance at n_1 is thus calculated by the parallel combination of the impedances looking into n_2 and n_5.

Therefore, the following methodology is suggested for solving the network. In the first pass, starting from the three loaded end nodes, the impedances are calculated and the parallel combination of these impedances at the parent node forms the parent node impedance. Working backward in this manner, the impedance at the root node (n_1 in the example) is calculated. Using (1) the voltage and current at the root node n_1 is calculated. Using (8) and (9) and the boundary voltages and currents, which were calculated at the parent node, the voltage and current at each node in the network can be calculated. Note that the current at each node is split into two currents flowing into each node.

In the next section, a computer program will be introduced which uses recursion and recursive data structures available in C to solve complex transmission line networks.

2.2. Recursive Programming and Data Structures

To introduce the algorithm for solving a complex transmission line network, we first consider the case where the network is limited to the binary tree structure shown in Fig. 5. In the figure, the generator is connected to the root of the tree through a source impedance Z_S. The tree consists of nodes which are either parents or leaves. A leaf is a node which is terminated on a load. For example, n_3, n_4, n_6, n_7, n_9, n_{11}, and n_{12}. Parent nodes have two branches. A left branch and a right branch. Nodes n_1,n_2, n_5, n_8, and n_{10} are parent nodes.

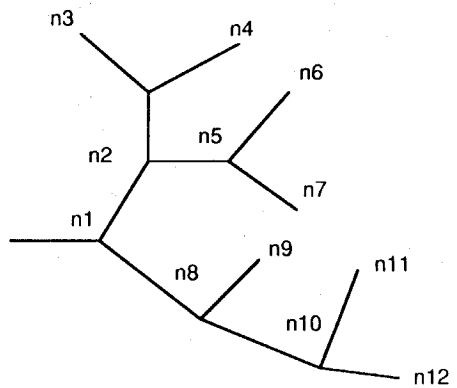

Figure 5. Transmission line network as a tree structure

In general each branch represents a transmission line with different characteristics and lengths. Each section of transmission line is associated with the node on which it terminates. Thus the section of transmission line from the generator to the root node n_1 is described in the data structure pointed to by n_1. This concept is described below.

Each node has an associated data structure which occupies memory locations. A pointer can be defined which points to the data structure in memory. As nodes are added to the tree, memory is dynamically allocated for the data structure and a pointer is defined. Thus, for the nodes of the network in Fig. 5 the following data structure can be defined in C.

```
struct  node  {
      struct node *left;
      struct node *right;
      struct node *parent;
      char name [16];
      float r,l,c,g;
      float length
      complex Z_Left;
      complex Z_Right;
      complex Z_L;
      complex   node_voltage;
      complex   left_current;
      complex   right_current;
      complex  input_current;
      complex  Z_0;
      complex  gamma;.
}
```

Within the data structure definition are three pointers to data structures of the same type. Thus the data structure is recursive. Two pointers point to the left and right nodes while the third pointer points to the parent node. Three cases are immediately evident. If the node is a leaf then the left and right node pointers are NULL. Otherwise, they will point to the left and right child nodes attached to the node. If the node is the root node, then the pointer to the parent will be NULL.

The other data types within the structure represent data necessary to describe the node. These can be classified into two groups. One group defines the name of the node and the characteristics of the transmission line (e.g., r,l,c,g and Z_0 and γ). The other group represents data which are calculated and depend on the network. These include the voltage at the node, the current flowing into the right and left nodes, and the impedances looking into the nodes.

It is very convenient to access data in a data structure using pointers to data structures. For example, to assign the variable Z the value of the characteristic impedance at the node pointed to by *np* we write,

$Z = np \text{ -> } Z_0$

To access the characteristic impedance of the left child node of the node pointed to by *np* we write,

$Z = np \text{ -> } left \text{ -> } Z_0$

At this stage, we will describe recursive functions which are used to compute the impedances, voltage and currents at each node. First we introduce two methods for traversing tree data structures.

2.3. Traversing Trees

There are general methods for traversing trees [3]. We will apply two of these methods to solve the tree network.

Postorder Listing

Postorder traversing of trees is illustrated in Fig. 6. This method is useful in the first pass needed prior to solving for the voltages and currents of the network. As pointed out earlier the impedances at each node must be computed. Thus in Fig. 6, the impedance at 3 is the parallel combination of the impedances of the loaded transmission lines 1 and 2. Similarly, the impedance at 6 is

computed from 4 and 5. Once the impedance of 3 and 6 are computed, the impedance at 7 can be calculated and so on. Careful study of the figure will show that the numbering schemes corresponds to the order in which the impedance calculations must be carried out. This order of traversing the tree is termed postorder listing. The method is summarized below [3]:

(1) If a tree is composed of only a single node, the post order listing consists of just that single node.
(2) If a tree consists of more than one node, the postorder listing consists of the postorder listing of each subtree, in left-to-right order, followed by the root.

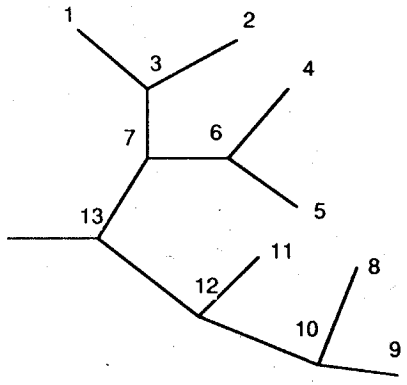

Figure 6. Post-order traversing of tree for impedance calculations

Preorder Listing

This method is used in the calculation of the voltages and currents at each node once the impedances have been determined. Preorder listing is illustrated in Fig. 7. Thus, once the boundary voltage at node 1 is known, the voltage at node 2 can be computed (since the impedance at 2 is also known from the first postorder traversing in computing the impedance). From 2, the voltage at 3 and 6 can be computed and so on. The preorder listing method is summarized below [3]:
(1) If a tree is composed of a single node, the preorder listing consists of just that single node.
(2) If a tree consists of more than one node, the preorder listing consists of the root, followed by the preorder listing of each sub tree in left-to-right order.

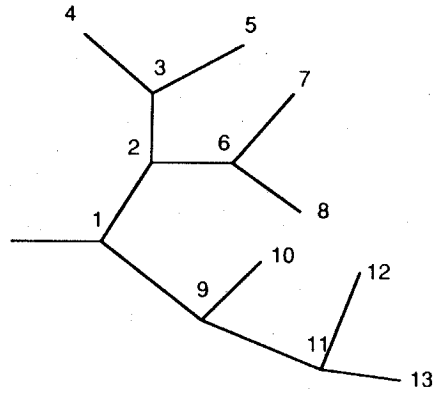

Figure 7 Pre-order traversing of tree for current and voltage calculations

The technique for solving complex transmission line networks outlined above has been incorporated in a CAD tool called CAPNET. CAPNET is described in the next section.

3. An Overview of CAPNET Design

CAPNET (CAPture NETwork) is a physical-level transmission line network analysis tool with a user-friendly graphical interface. The current implementation allows user to model both local-area networks and printed circuit boards; the analysis technique used can easily be expanded to model other types of transmission line networks as well.

CAPNET supports both analytical and empirical (table-based) models of transmission lines and junctions. It is designed so that new models for lines and junctions can be coded and incorporated into the program with a minimum of difficulty, specifically, without modification to the "high-level" simulator and editor.

The program provides a graphical display of the "captured" network. It also provides a variety of output data and plots, including impulse and frequency responses, characteristic impedance and other line parameters, conductor voltage and current distributions, pulse responses, and eye diagrams.

CAPNET currently comprises about 12,500 lines of C code, split about equally between the graphical interface and the actual simulator. It requires a VAXstation running VMS, and currently uses the GKS graphics interface[4]. We will also have CAPNET running on X-Windows in the near future.

Future development will include models for multi-conductor (three or more) lines, inter- and intra-conductor coupling, and non-linear loads.

3.1. How CAPNET Represents a Network Topology

Visually, CAPNET represents a network topology as a multiway tree, consisting of "nodes" connected by "edges." A root or "source" node represents the signal input. Other nodes represent either junctions between conductors or conductor terminations. Conductors themselves are represented as edges.

A consequence of this is that networks represented by the present implementation of CAPNET must be free of loops. Many topologies containing internal loops can, however, be reduced to simplified cases without such loops, and we may investigate how to incorporate this process into CAPNET in the future.

At the lowest level, CAPNET topologies are represented by a binary tree structure. This is most satisfactory in the cases where nodes connect at most three edges. Instances of nodes connecting four or more edges are infrequent enough (in our experience) that we have opted to use a modified binary tree representation of the network rather than a generalized multiway tree representation. The structure of the binary tree is both easier to comprehend and manipulate, and our implementation has no difficulty dealing with cases where more than three edges connect at a node.

3.2 Creating and Editing a Network Topology

CAPNET topologies are created and edited graphically in most cases. Topology data is stored in a simple ASCII (text) file, however, and it is possible to create networks without using the graphical editor. We have already seen one application that converts the output of a PCB layout program to a CAPNET-readable form; other programs of this nature should not be too difficult to write.

Users of CAPNET typically define (that is, draw) the topology of a new network before defining the network's specific line and junction types. We found this method to be considerably faster than the

alternative, in which a user has to pause to enter data after adding each line segment to the graph. A user could expect to enter a complete network of ten to twenty lines in under five minutes, and in less if he is familiar with the system.

Users can edit a network at any stage during or after its creation. The editor allows users to delete single line segments and subtrees, insert new line segments, divide existing line segments, relocate the source node, and reorient individual line segments or subtrees. Users can also modify line and junction characteristics at any time.

3.3 Analysis Options

The CAPNET simulator produces a variety of output, all of which can be redirected to plotting devices or files. CAPNET can display voltage and current distributions across any line at a chosen analysis frequency. It can display the characteristic impedance and other calculated parameters of the junction and transmission line at that frequency, including the subtree load impedance and nodal currents and voltages.

In addition, CAPNET can display the frequency and impulse response at any node in the network. Impulse responses can be convolved with individual pulses or pulse trains to produce pulse responses and eye diagrams.

The analysis technique used makes it possible to solve for the frequency response of many nodes simultaneously without a significant increase in the calculation time. This is convenient when the network being analyzed is very large.

3.4 CAPNET Example

An analysis of pulse propagation in a printed circuit board is provided below. In Figure 8 , a PC board trace is shown with *bends*, *vias*, and *tees*. The length of each trace is also shown in meters. The menu in Figure 8 shows the editing features available in CAPNET. These extensive features make graphical capture or modification of large networks very easy.

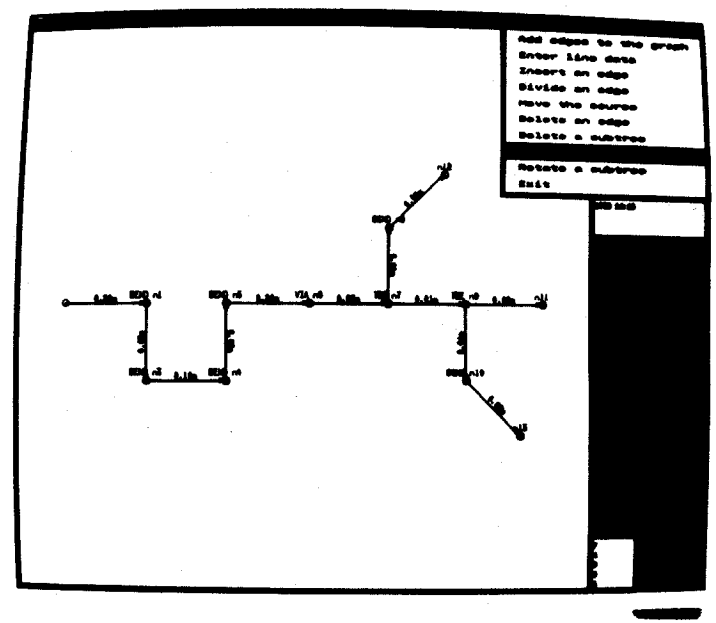

Figure 8. Editing Options.

In Figure 9, the analysis window is shown. Note that for the selected segment, the line length is 2 cm of 8-mil trace. The load is also shown as a parallel resistance (280 Ohms) and capacitance (10.0 pf). The flag for calculating the impulse response for this node has been set.

In this case, when the network is solved, the impulse response for this node will be stored in a dynamically allocated array. (The user can flag those nodes for which an impulse response is desired prior to calculation).

From Fig. 9 note the analysis options for node n12 which include:

• **Seg. Info.** (Voltage and current distribution at a fixed frequency, node information such as characteristic impedance and propagation constant among other parameters).

• **Pulse Param.** This is used to set pulse width, rise and fall times and pulse repitition rate.

• **Eye Diag. Param.** This is used to set the eye diagram parameters such as number of samples per baud, phase etc.

• **Eye Diagram.** This calculated and plots the eye diagram.

• **Sample Parameters.** This option is used to set the sampling rate and the number of samples to compute the impulse response.

• **Freq. Response.** The frequency and phase response are plotted.

• **Imp. Response.** The impulse response is computed and plotted.

• **Pulse Response.** The pulse response based on convolving the impulse response with the pulse sequence determined by the pulse parameters option is computed and displayed.

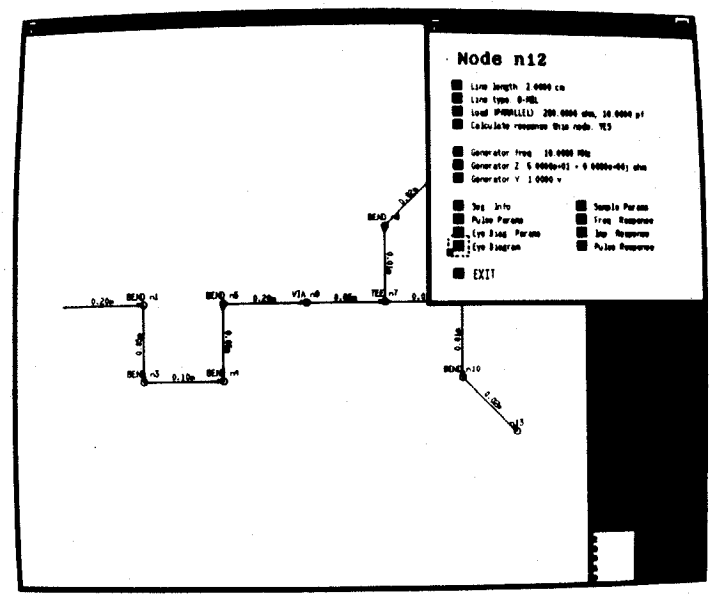

Figure 9. Analysis Options.

Figure 10 shows the impulse response for Schottky TTL loads. The pulse response to a 5 ns wide pulse is shown in Figure 11. The eye diagram is shown in Figure 12.

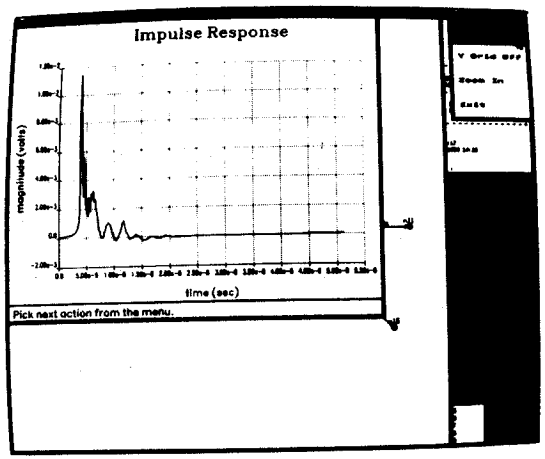

Figure 10.

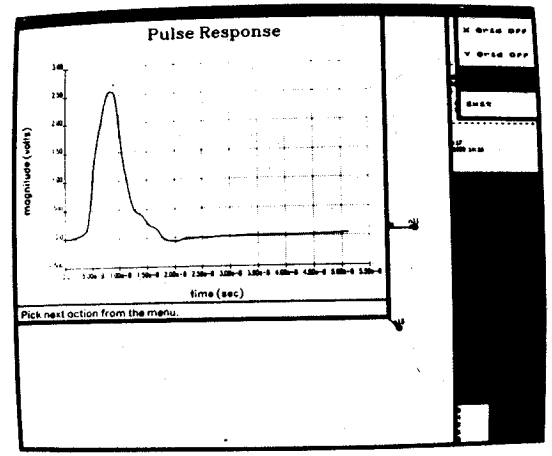

Figure 11.

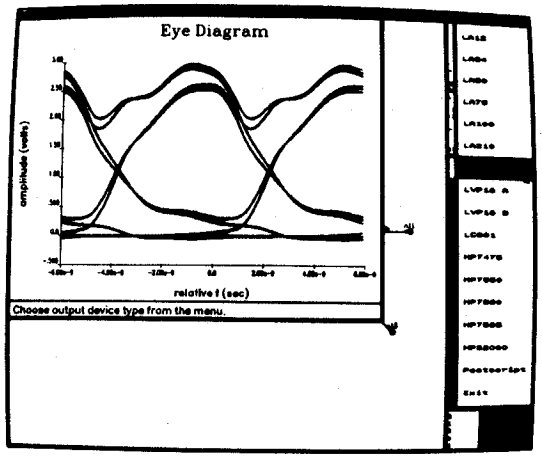

Figure 12.

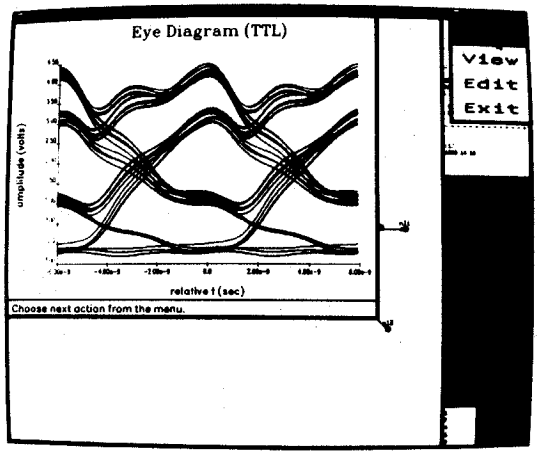

Figure 13.

The loads were changed to TTL and the eye diagram was obtained and is displayed in Figure 13. Note the severe degradation associated with the slower speed technology on this trace compared to Figure 12.

Also shown in Figure 12 is the hardcopy options. A hardcopy of any plot can be obtained on the devices shown in the menu.

The example presented above illustrates the power of this CAD tool in analyzing the performance of high speed technologies in printed circuit boards and other complex transmission line networks. Using the tool, the lines, loading impedances, distances and other circuit parameters can be modified so that acceptable high speed operation is obtained.

4. References

[1] D.G. Messerschmitt, "Transmission line modeling program written in C,"*IEEE Journal on Selected Areas in Communications*,Vol. SAC-2, pp. 148-153, January 1984.

[2] Dworsky, Lawrence N, *Modern Transmission Line Theory and Applications*,John Wiley and Sons, Inc., 1979.

[3] Gerald E. Sobelman, David E. Krekelberg, .*Advanced C Techniques and Applications*, Que Corporation, Indianapolis, Indiana,, 1985.

[4] F.R.A. Hopgood, D.A. Dull, J.R. Gallor, D.C. Sutcliffe,*Introduction to GKS*, Second Edition, Academic Press, 1986.

[5] Sasan Ardalan, "CAD of digital communication systems with complex transmission line networks," *Proceedings of the National Communications Forum*, Chicago , Illinois, October 1987.