Simulating the delay in logic networks for large, high-speed computers

by E. A. WILSON
Honeywell Information Systems
Phoenix, Arizona

When designing a computer with TTL logic circuits, the delays of logic paths have been estimated by considering the number of gate delays and adding in load and media factors. Such a simplistic approach is not accurate enough for calculating delays when designing high-performance large systems using high-speed, non-saturating circuits such as HCML (Honeywell’s Current Mode Logic). There are several reasons:

- The clock (cycle) time is considerably faster for a high speed machine, hence the calculations must be very accurate in order to meet performance goals.
- The loading on the driving gate varies with the number of driven gates, hence affecting the rise time of the line (interconnect) voltage.
- The geometry of the interconnect (branch points, connectors, various media impedances) has an effect on signal propagation with high-speed edges.
- Media delay is a significant percentage of path delay as ICs become faster.

All of the above leads to a need for an ability to simulate the delay for proposed interconnects which do not meet simple driver-line-load configurations without intermediate branch points and/or media changes.

The large package programs which are available on the open market are well suited for circuit design work when developing the circuit set, but they are unsuitable for the multitude of cases which must be run when simulating the full design. They require too much memory, take too long to execute, and are too general when the same gate can be used for every case simulated.

This paper presents a simulation program and a design methodology which have proven successful in our actual large-computer design environment. The simulation program has been tailored to the HCML circuit set and optimized for small storage, very fast execution and minimal data input. Also, the program has been written so that automatic or manual modes of operation are available.

In the automatic mode, the network checking program (which checks the logic designer’s data base for correct pin assignments, wiring rule violations, etc., but is not a part of this paper) feeds the data into the delay simulation program and receives back the interconnect portion of the delay for each load gate. The delay data are added to the logic designer’s data file and reported to him when he accesses the file for the results of the network checking run.

In the manual mode, the designer selects an interconnect which has a problem (such as an unexpected long delay) or inputs a proposed interconnect for which he wants to calculate the delay before continuing his design. In this mode, he has several options for the output. He may select just the individual interconnect delays for the loads; a printout of driver and load voltages with time; or a plot of the waveforms of the driver and load voltages. In addition, the load voltages may be printed/plotted as either the input voltage to the load gate, or the output voltage from the load gate. By using one or more of these options wave reflections, effect of input rise on load delay, turn-on/turn-off/turn-on, etc. may be observed and frequently the problems corrected by interconnect modification before the design is released. If interconnect modifications do not correct the problem, then either an alternate logic implementation may be used, or the problem may be compensated for in the rest of the logic chain. In any case, the simulation provides the designer with the information before the design is released and built as hardware.

The simulation program uses a two-model approach with an interaction between the models similar to substructuring in finite element programs, except in this case the interaction is a function of time.

INTERCONNECT (LINE) MODEL

This model is based on a finite element rather than a mesh or loop current formulation which actually only affects the terms in the resulting capacitance matrix, as will be seen. A basic goal was to minimize the execution time and past experience has shown that if this goal is kept in mind from the beginning, a more efficient program can be written than would be if the theory were developed and then the programming tacked on as an independent activity. Therefore, a method (finite elements) was chosen for the theoretical model which was known to lead to a straightforward matrix model.
The basic current voltage relations for a line element are:

\[ \frac{\partial e}{\partial x} = -iR, \quad i = -k \frac{\partial e}{\partial x} \]  

(1)

\[ \frac{\partial e}{\partial x} = -L \frac{dt}{dt}, \quad i = -m \int_0^L \frac{\partial e}{\partial x} dt \]  

(2)

\[ e = \int_0^L i dt + \frac{c}{\partial t} \]  

(3)

where in equation (3), the voltage, \( e \), is relative to ground (or some fixed reference).

In vector form (although this is only a one dimensional problem, it does not hurt to be mathematically correct), the divergence of the current at any point along the line element yields:

\[ \nabla \cdot i = c \frac{\partial e}{\partial t} \]  

(4)

and using the second relations of (1) and (2),

\[ k \nabla^2 e + m \int_0^L \nabla^2 e dt = c \frac{\partial e}{\partial t} = c\dot{e} \]  

(5)

with the boundary conditions

\[ i \cdot n = -(k \nabla e + m \int_0^L \nabla e dt) \cdot n \]  

(6)

where \( n \) is an outward vector from each end of the line element.

Equations (5) and (6) may be combined in a variational equation as:

\[ \delta \text{Integral} = \int_0^L (k \nabla^2 e + m \int_0^L \nabla^2 e dt - c \frac{\partial e}{\partial t}) dx \]  

(7)

where \( \phi \) is an approximation function for \( e \) and \( \text{Integral} \) will have to be minimized. The term \( db \) is an increment of the boundary, which in this case consists of the two ends of the line element.

Using Green's theorem,

\[ \delta \text{Integral} = \int_0^L (-k \nabla^2 e + m \int_0^L \nabla^2 e dt - c \dot{e}) dx - \int (i \cdot n) \phi db \]  

(8)

where \( e \) has been assumed to be piecewise continuous in time. Since \( \phi \) is a variation of \( e (\phi = \delta e) \), the integral becomes

\[ \text{Integral} = -km \int_0^L (\nabla e)^2 dx - \int (i \cdot n) \phi db \]  

(9)

Now assume a function for \( e \) of the form

\[ e = \omega_1 + \omega_2 x = [1 \quad x] [a] = [E] [a] \]  

(10)

In the terms of the end (nodal values),

\[ e = [E] [A^{-1}] [V] \]  

(11)

where

\[ [A] = \begin{bmatrix} 1 & 0 \\ L & 1 \end{bmatrix}, \quad l = \text{line element length} \]

\[ [V] = \text{vector of nodal voltages} \]

then,

\[ \nabla e = \frac{\partial e}{\partial x} = [0 \quad 1] [A^{-1}] [V] \]  

(12)

Placing the above into equation (9) and taking the variation with respect to the nodal voltages yields;

\[ \delta \text{Integral} = 0 = -km \int_0^L (\nabla [E])^2 (\nabla [E]) dx [A^{-1}] [V] \]

\[ -m \int_0^L [A^{-1}] \int_0^L (\nabla [E])^2 (\nabla [E]) dx \]  

\[ [A^{-1}] [V] dt \]

\[ -c [A^{-1}] \int_0^L [E]^T [E] dx [A^{-1}] [V] \]

\[ + \{ \text{input current at nodes} = l \} \]

(13)

where a subscript \( T \) means the transpose of a matrix, which is of the form

\[ [K] [V] + \int_0^L [M] [V] + [C] \frac{\partial e}{\partial t} = \{I\} \]  

(14)

and the matrices are given by

\[ [K] = kl \begin{bmatrix} 1 & -l \\ -l & 1 \end{bmatrix} \]  

(15)

\[ [M] = ml \begin{bmatrix} 1 & -l \\ -l & 1 \end{bmatrix} \]  

(16)

\[ [C] = cl \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \]  

(17)

If a mesh current formulation had been used, the capacitance matrix (17) would have had only diagonal terms instead of the true distributed line capacitance which is inherent in the finite element approach.

LOAD (GATE) MODEL

The particular gate used in this paper is a CML circuit; however, the load model does not have to be confined to any single type of circuit set since the model in the preceding section can be used with any load which uses a voltage input as a boundary condition. This will be more fully explained in the "Substructure" section.

The seven-node model which uses an Ebbers-Moll model for the transistors is shown in Figure 1. The voltages \( V_1, \ldots, V_7 \) are unknown and vary as \( V_m \) varies. The node (subscript) numbers are specifically chosen to reduce the
Figure I—Model of Honeywell Current Mode Logic gate.

The basic equations for the transistor model are:

\[ B_X = \frac{1}{B_N} + 1 \]  
\[ B_I = \frac{1}{B_I} + 1 \]  
\[ I_I = \text{I}_{\text{es}} \exp(\theta_3 V_{\text{be}}) \]  
\[ I_N = \text{I}_{\text{es}} \exp(\theta_3 V_{\text{be}}) \]  
\[ C_{\text{be}} = a_1 N_j (\phi_1 - V_{\text{be}}) + \theta_3 T_{\text{es}} \]  
\[ C_{\text{bc}} = a_2 N_j (\phi - V_{\text{be}}) + \theta_3 T_{\text{es}} \]  
\[ I_{\text{be}} = (1 - \text{I}_{\text{es}}) B_S - I_{\text{es}} + I_{\text{es}} \]  

The values for \( a_1, a_2, \phi_1, \theta_3, I_{\text{es}}, T_{\text{es}}, T_{\text{es}}, B_S \), and \( B_I \) are experimentally determined from actual circuits. In relation to Figure 1,

\[ C_{\text{be}} = C_3, C_4 \]  
\[ C_{\text{bc}} = C_1, C_2 \]  
\[ V_{\text{be}} = (V_2 - V_3), (V_4 - V_6) \]  
\[ V_{\text{bc}} = (V_2 - V_4), (V_4 - V_2) \]  
\[ I_{\text{es}} = \text{I}_{\text{es}}, I_{\text{es}} \]  
\[ I_{\text{es}} = \text{I}_{\text{es}}, I_{\text{es}} \]  

Writing the nodal equations will lead to an equation of the form

\[ [A] [V] = \{r\} \]  

where the right hand side (\( \{r\} \)) will be composed of a current vector, a vector containing capacitance terms times nodal voltages, and a vector of known voltages \( (V_{10}, V_{20}, V_{C0}, V_{30}) \).

The approximation of constant capacitance during each time step (which is due to the finite difference approximation which will be imposed in the next section on equation (14)) is consistent with the interconnect model. However, such an approximation for the current generation is not made since a better approximation is easily achieved.

For example, in terms of the voltage at the beginning of the time step, the current at the end of the time step may be written as

\[ I_{\text{be}} = I_{\text{es}} - I_{\text{es}} + I_{\text{bc}}(\exp(V_{\text{be}}) - V_{\text{be}}) \]  

where \( V_{\text{be}} \) is the voltage at the beginning of the time step. A similar expression exists for \( I_{\text{be}} \). This expression is obtained from the first two terms of a series expansion of the exponential factor.

These expressions for current will contribute terms to the matrix \([A]\) in (32), and also destroy its symmetry. However, since the matrix is sparse, it can be easily solved by hand and the solution can be coded directly into the program. The method of solution follows.

The matrix \([A]\) is of the form

\[ [A] = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ A_{61} & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix} \]  

The vector \( \{r\} \) is given by

\[ \{r\} = \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \]

\[ \{r\} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} V_{10} K_{\phi} \\ -2.6 K_{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -3.3 K_{\phi} \end{bmatrix} \]  

The matrix \([A]\) is defined as

\[ [A] = \begin{bmatrix} C_1 & -C_1 & -C_2 & -C_2 \\ -C_1 & C_1 + C_3 & -C_3 & -C_3 \\ -C_2 & -C_3 & C_2 + C_4 & -C_4 \\ -C_3 & -C_4 & -C_3 & C_3 \\ -C_4 & -C_4 & C_4 & C_4 \end{bmatrix} \]

where

\[ I_1 = -I_{\text{es}} B_t + I_{\text{es}} B_i (1 - \theta (V_3' - V_3')) \]  
\[ -I_d (1 - \theta (V_3' - V_3')) \]  
\[ I_2 = -I_{\text{es}} B_t + I_{\text{es}} B_i (1 - \theta (V_4' - V_4')) \]  
\[ -I_d (1 - \theta (V_4' - V_4')) \]  
\[ I_3 = -I_{\text{es}} B_t + I_{\text{es}} B_i (1 - \theta (V_5' - V_5')) \]  
\[ -I_d (1 - \theta (V_5' - V_5')) \]  
\[ I_4 = -I_{\text{es}} B_t + I_{\text{es}} B_i (1 - \theta (V_6' - V_6')) \]  
\[ -I_d (1 - \theta (V_6' - V_6')) \]  
\[ I_5 = -I_{\text{es}} B_t + I_{\text{es}} B_i (1 - \theta (V_7' - V_7')) \]  
\[ -I_d (1 - \theta (V_7' - V_7')) \]  
\[ -I_d (1 - \theta (V_7' - V_7')) \]
\[ I_x = -I_{a}B_x + I_{a} + I_{B}(1 - \theta_d(V_{a} - V_{b}')) \] (39)

and the values of \( I_x \) and \( I_a \) are functions of the voltages in each current equation.

By using the Crout algorithm, the matrix \([A]\) may be easily modified into a new matrix \([\tilde{A}]\) which permits a simple solution to the seven simultaneous equations:

\[ \tilde{A}_i = A_{ii} - \sum_{k=1}^{j-1} A_{ik} \tilde{A}_{kj} \quad i \geq j \geq 2 \] (40)

\[ \tilde{A}_{ij} = (A_{ij} - \sum_{k=1}^{j-1} A_{ik} \tilde{A}_{kj}) / \tilde{A}_{ii} \quad i < j \geq 2 \] (41)

\[ \tilde{A}_{ij} = A_{ii} / A_{ij} \quad j \geq 2 \] (42)

\[ \tilde{A}_{ii} = A_{ii} \] (43)

Then the vector \([r]\) is modified by

\[ \tilde{r}_i = (r - \sum_{k=1}^{i-1} \tilde{A}_{ik} \tilde{r}_k) / \tilde{A}_{ii} \quad i \geq 2 \] (44)

\[ \tilde{r}_i = r_i / A_{ii} \] (45)

and the voltage vector is updated by

\[ V_i = \tilde{r}_i \] (46)

\[ V_i = \tilde{r}_i - \sum_{k=1}^{i-2} \tilde{A}_{ik} V_k \quad i \leq 6 \] (47)

The above solution is applied for each time step of the interconnect model solution and the capacitance values are updated for each time step.

**SUBSTRUCTURE ASSEMBLY**

Equation (14) must be represented in a finite difference form for calculation purposes. This will be of an implicit, trapezoidal integration form to obtain a better approximation than a simpler explicit, rectangular integration form would provide. The result is that larger time steps may be used, hence faster execution.

This is easily achieved by writing the discretized form of equation (14) for the voltage at time \( T - \frac{\Delta T}{2} \) instead of time \( T \) (the end of the present time step for which the nodal voltages are unknown). This means

\[ \{V\}_{T-1/2} = \{I\}_{T-1/2} + \{V\}_{T-1} \] (48)

in terms of subscripts \( i \) at \( T \) and \( i - 1 \) at \( T - \Delta T \). Then, for the D.C. terms,

\[ \{I\}_{T-1/2} = \{K\} \{V\}_{T} + \{V\}_{T-1} \] (49)

for the capacitive terms,

\[ \{I\}_{T-1/2} = \frac{1}{\Delta t} C \{V\}_{T} - \{V\}_{T-1} \] (50)

and for the inductive terms,

\[ \{I\}_{T-1/2} = (\Delta t) [M] \{V\}_{T} + \{V\}_{T-1} - \sum_{j=1}^{i-2} \{M\} \{V\}_{j} \] (51)

where the factors \( i \) and \( j \) come from using

\[ \{V\}_{T-1} = \{I\}_{T-1} + \{V\}_{T-2} \] (52)

for the integration in the half time step from \( T - \Delta T \) to \( T - \frac{\Delta T}{2} \).

The resulting finite difference equation is (let \( (\Delta t) [M] \) become just \( [M] \) and \( [C] \{\Delta\} \) become just \( [C] \)):

\[ \{[I]\} = \{[K]\} \{[V]\}_{T} + \{[V]\}_{T-1} \] (53)

\[ + \{[C]\} \{[V]\}_{T-1} - \{[M]\} \{[V]\}_{T-1} - \sum_{j=1}^{i-2} \{[M]\} \{[V]\}_{j} \]

Given the above equation, the two models may now become interactive in time.

For the interconnect model, \( 1/R_g \) from the load model may be included in the conduction matrix, \([K]\), with the gate voltage \( V_g \) as a known boundary condition for the interconnect model. At any given time step, the gate voltage is known for \( T - \Delta T, T - 2 \Delta T, \) etc., but not \( T - \frac{\Delta T}{2} \). This is easily resolved by the finite difference extrapolation formula

\[ \{V\}_{T-1/2} = \{V\}_{T} - \frac{1}{2} (V_{2} - V_{1}) \] (54)

Defining the subscript \( R \) as identifying quantities associated with the gate (i.e. \(-R_g \) and \( V_g \)) model, equation (53) may now be written as:

\[ \{[I]\} \{[K]\} \{[V]\}_{T} + \{[V]\}_{T-1} \] (55)

\[ - \{[K]\} \{[V]\}_{T-1} - \{[K]\} ((V_{1} - 3(V_{2} - V_{1} + (V_{3} - V_{2}))) \]

\[ + \{[C]\} \{[V]\}_{T-1} - \{[M]\} \{[V]\}_{T-1} - \sum_{j=1}^{i-2} \{[M]\} \{[V]\}_{j} \]

where \( [K]_R \) is null except for diagonal elements to which loads are attached (or terminating resistors if the need arises).

The final boundary condition is the input or driver. This is simply an input current to the first node \( I_1 \) (in the vector \([I]\)) which matches the wave form for actual experimental data.

Once the interconnect model has been solved for time \( T \), the nodal voltage at each load location becomes \( V_k \) for the load model in Figure 1. The same basic model is used for each load, but the voltages \( V_1, V_2, \) etc. are stored in arrays so that the loads at different positions in the network can be simulated autonomously.

The substructure interaction scheme follows two simple repetitious steps:

1. Impose boundary conditions for time \( T - \frac{\Delta T}{2} \) from current source (driver) and extrapolated load voltages \( (V_{FR}) \) on interconnect model and solve for nodal voltages at time \( T \).
2. Impose \( V_{FR} \) boundary conditions from interconnect nodes on load model and solve for gate voltages at time \( T \) for all load locations. Then repeat Step 1 for next time increment.

This process is repeated until all of the loads have reached a stable on (or off) condition. The stable condition is defined by a prescribed voltage above the threshold switching vol-

From the collection of the Computer History Museum (www.computerhistory.org)
age below which $V_{2}(F)$ does not fall once it has been attained.

**IMPACT OF MATRIX STORAGE ON EXECUTION TIME**

The line model developed in the earlier section cannot be used for unlimited lengths as a single element. For example, a 75 ohm transmission line with a propagation delay of 5 nanoseconds per meter should be limited to a 50 or 60 mm long element. Therefore, long line segments must be broken into shorter elements. This can result in large matrices (200 by 200 for example) if the lines are long and there are many branch points. Such large matrices not only require much storage, but also much computation time. This can be reduced by considering the nature of the interconnect problem. An example is shown in Figure 2. Each line segment would have one or usually more elements. The resultant matrix would be tridiagonal except at the branch points. Since all arithmetic operations outside of the banded portion of the matrix will result in zero, there is no need to either store or operate on these outside elements.

Advantage was taken of the matrix type and a one-dimensional, variable bandwidth storage scheme was developed which stores, hence, operates on, only the affected elements in the matrix. This results in a reduction of storage of almost one order of magnitude and a similar reduction of execution time. This makes the manual version of the program practical since the user gets the results from the terminal within seconds after he types in the data.

**SAMPLE PROBLEM**

Figure 2 is typical of those types of logic networks which can be simulated with this approach. The four basic media are the multichip package, the board, the cable between boards, and the connectors. The type of each line segment is given in Table 1.

For the sake of simplicity, the plot in Figure 3 shows only the voltage at the source and the output of the gates at loads 3 and 5. Notice the reflections, turn off of load 5, etc., which affect the delay of the network. Such information is extremely important in the early stages of design in order to assure meeting performance goals.

As a contrast, a simple hand calculation based on propagation time would have predicted turn-ons for load 3 at 13 ns and load 5 at 13.3 ns. The difference between the simple assumption and the full simulation (17 ns for load 3 and 35 ns for load 5) is obvious from Figure 3. This illustrates the importance of a full simulation instead of estimates based on run lengths and loading factors which can not take into account reflections caused by mismatched impedances.

**STAR CONFIGURATION SAMPLE PROBLEM**

While the preceding example demonstrated the complex logic interconnection which can be modeled, a simple problem will help to demonstrate further the need for full simulation.

This problem will be developed from a simple, impractical (in the sense that it would not appear in a real design) star to a more realistic interconnect which may loosely be described as a star.

The simple star is shown in Figure 4 and for the first simulation, all of the lines were treated as board lines instead of using the connector or micropackage parameters. Then the micropackage and connector parameters were used for the appropriate segments and the load output plots are shown in Figure 4 along with the driver output voltage for the case of all three media in the problem. Because all signal

<table>
<thead>
<tr>
<th>LINE</th>
<th>LENGTH (MM)</th>
<th>MEDIUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>3</td>
<td>250</td>
<td>BOARD</td>
</tr>
<tr>
<td>4</td>
<td>750</td>
<td>RIBBON CABLE</td>
</tr>
<tr>
<td>5</td>
<td>375</td>
<td>BOARD</td>
</tr>
<tr>
<td>6</td>
<td>250</td>
<td>BOARD</td>
</tr>
<tr>
<td>7</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>8</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
<tr>
<td>9</td>
<td>250</td>
<td>BOARD</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>11</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
<tr>
<td>12</td>
<td>125</td>
<td>BOARD</td>
</tr>
<tr>
<td>13</td>
<td>375</td>
<td>BOARD</td>
</tr>
<tr>
<td>14</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
<tr>
<td>16</td>
<td>50</td>
<td>BOARD</td>
</tr>
<tr>
<td>17</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>18</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
<tr>
<td>19</td>
<td>625</td>
<td>RIBBON CABLE</td>
</tr>
<tr>
<td>20</td>
<td>625</td>
<td>BOARD</td>
</tr>
<tr>
<td>21</td>
<td>25</td>
<td>CONNECTOR</td>
</tr>
<tr>
<td>22</td>
<td>25</td>
<td>MICROPACKAGE</td>
</tr>
</tbody>
</table>

Figure 2—First sample problem interconnect configuration. Numbered line segment types are given in Table 1.
paths are the same length and all load factors equal, only one load output plot exists for each case. Of note is the fact that when all media are considered, the delay is three nanoseconds longer than for the all board lines case. This is important because if the difference in propagation speed between board lines and connector or micropackage lines is multiplied by the appropriate line lengths, only 48 nanoseconds can be accounted for in the three-nanosecond difference. The bulk of the difference is therefore due to the variation of line characteristics in the signal paths. This can be noted by the dip in the source (driver) voltage.

The next case also uses the star except that the top load has a load factor of 2.5 (for example, two “high” current gates and a “low” current gate driven in parallel on the same chip or adjacent chips) and the other three loads only have a load factor of 0.5. Note that the total load seen by the source is still four, the same as the previous case. The output plots are in Figure 5 and show a 1.5-nanosecond difference in delay time. However, the possibly surprising result is that the gates on the more heavily loaded line turn on before the half loads. This is due to the capacitance of the larger loading making the top line less sensitive to the dip in the source voltage which the lightly loaded lines track closer.

A true equal line length star would not likely be found in a real design since board routing and micropackage placement would preclude such an ideal case. The quasi star in Figure 6 is more representative of a real interconnect. The board line lengths are chosen to relate to the previous star configuration. The average distance to the four loads is the same as the previous equal board line lengths. Likewise, the total loading (four) is the same. In Figure 6, only the output of loads one and two and the source are shown for the sake of clarity. Load one turns on 10.5 nanoseconds sooner than for the equal line length case yet the difference in distance only accounts for the signal reaching the load two nanose-
Figure 4—Simple star sample problem. The SOURCE waveform is for the ALL MEDIA PARAMETERS case.
Figure 5—Simple star configuration except LOAD 1 has a load factor of 2.5 while the other loads are only 0.5.
Figure 6—Quasi star configuration. The average distance to each load is the same as for the simple star and all loads factors are 1.
conds sooner. The reason for the faster turn-on can be seen in the source voltage. It reaches a higher initial plateau which is due to the first branch point having only two instead of four branches. The waveform of load two is also noteworthy. Although it turns on before load one, it turns off at 23 nanoseconds and on again at 24 nanoseconds. Hence, 24 nanoseconds must be taken as the true turn-on time since the load is itself a driver for the next step in the logic path and must be stable before turn-on of its loads can be assured.

SIMULATION PROGRAM CHARACTERISTICS

Once the program had been developed, it was checked for accuracy by running several sample problems which were also modeled on a popular circuit analysis program, SCEPTRE. The voltage plots were identical and the only variations were in the third significant digit when the voltage values were printed instead of plotted. An important difference between the two programs is that the version of SCEPTRE used required a minimum of 25k words of storage and approximately one minute of execution time (plot suppressed to reduce I/O time), while the program used for this paper took less than 12k of storage and ran the comparison problems in less than a second (also, plot suppressed). Both programs were run on the same Honeywell Level 66 Computer.

The program can handle up to 100 nodes and line elements (each line segment may be one or more line elements which are 125 mm. or less) but that may be increased by just a dimension statement change. The first sample problem required only about 50 nodes. The program is written in FORTRAN and may run on any computer with adequate storage.

CONCLUSION

The simulation method presented in this paper is proving to be a useful tool in the early design phases of logic networks for two reasons. First, it provides essentially automatic timing analysis of all designs without any extra work on the part of the designer when the design data base is used for input. The typical execution time of less than one second per simulation run allows the computer-aided design automation system to include a full set of timing analyses without any adverse effect on turn-around time or competition for resources. Second, the designer is free to use more complex branch schemes than could be used if the delay predictions had to be done according to simplified wiring rules. This freedom allows more efficient design and routing while assuring that the result will be predictable. Again, the fast execution and the small storage makes desk-top terminal operation practical. The designer may obtain voltage waveform plots on the terminal as fast as it will type, and then check out design modifications as fast as he can type.