Parallel simulation of electrical circuits using the TLM technique

K.K. Fung,[a] S.Y.R. Hui,[b] C. Christopoulos[c]

[a] *School of Electrical Engineering, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, Australia*

[b] *Department of Electrical Engineering, University of Sydney, N.S.W. 2006, Australia*

[c] *Department of Electrical and Electronic Engineering, University of Nottingham, University Park, Nottingham NG7 2RD, UK*

ABSTRACT

This paper describes a parallel simulation of an electronic circuit using the Transmission Line Modelling (TLM) method. The TLM method is used to decouple the circuit into sub-circuits which are then simulated concurrently in a parallel processing system. This approach not only simplifies the circuit formulation process but also reduces the overall computing time of the circuit simulation when compared with traditional sequential method. The techniques for decoupling the circuit and implementing the parallel algorithm are described. The method is demonstrated in an electrical circuit simulation. Comparison of the computing time with the sequential approach confirms the computing efficiency of the proposed method.

INTRODUCTION

One general problem encountered in computer circuit simulation for complex circuits is the lengthy time required for circuit formulation and computation. A decoupling method known as diakoptics has been developed for breaking down large circuit into small ones for circuit analysis [1]. Recently, the idea has been extended to computer circuit simulation via the use of TLM link. It has been shown [2] that TLM link can be used to decouple circuits so that a large circuit can be divided into small circuits for circuit formulation and simulation. This divide-and-conquer approach not only simplifies the circuit formulation due to small circuits, more importantly, it opens a way to speed up the computer simulation process by the use of parallel computing techniques. This paper first describes briefly the TLM decoupling technique and then presents a method to implement the decoupled circuit simulation using a parallel processing system.

Stub and Link

TLM stub is one-port model which can be employed to model discrete circuit components. For example, an inductor in an electric circuit can be modelled as a short-circuited transmission line stub. Similarly, a capacitor can be modelled as an open circuit transmission line stub.

On the contrary, TLM link is a two-port model which can be used to link circuits together. Because of this nature, TLM link is also the building block used extensively in multi-dimensional field modelling. The link model is shown in fig.1a, and its equivalent circuit in fig. 1b. Detailed description of TLM stubs and links can be found in references [1-3].

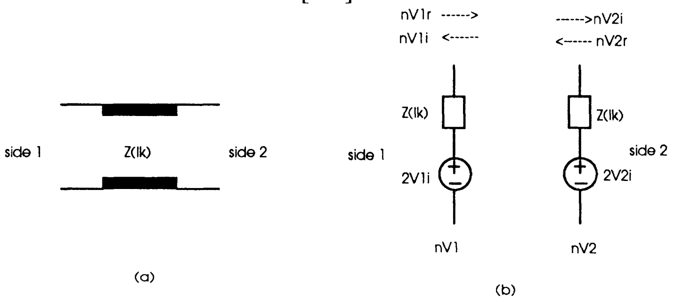


Fig. 1 TLM link model and its equivalent circuit

DECOUPLED SIMULATION

To illustrate the principles of parallel simulation, a simple sawtooth waveform generator circuit as shown in fig 2 is chosen

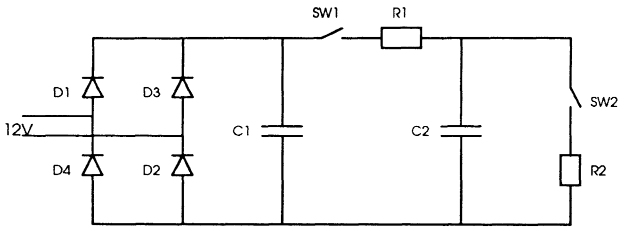


Fig. 2 Sawtooth Waveform Generator Circuit

where C1=3300uF, C2=1uF, R1=1400 ohms ,R2=33 ohms

The TLM circuit model can be formed by replacing C1 with a TLM link and C2 with a TLM stub, as shown in fig. 3a. For simplicity, all diodes and switches are

Transactions on Engineering Sciences vol 3, © 1993 WIT Press, www.witpress.com, ISSN 1743-3533

simulated as switched resistances. Each is treated as a low resistance when conducting, and as a high resistance when not conducting. The Thevenin equivalent circuit for this TLM model can be derived as shown in fig. 3b, taking into account Ls, the source inductance. When SW1 is on and SW2 is off, C2 is charged up. When SW1 is off and SW2 is on, C2 will be discharged .

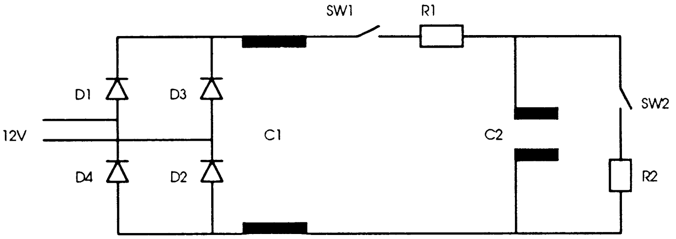
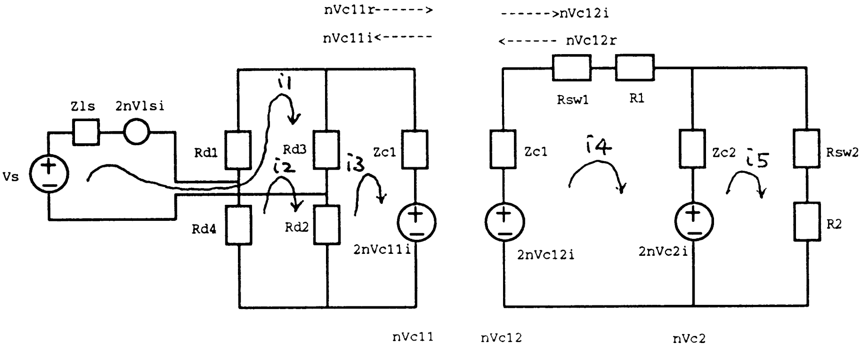


Fig 3a TLM link and TLM stub



Sub-circuit 1 - rectifier

Sub-circuit 2 - waveform generator

Fig. 3b Decoupled Equivalent Circuit

Zc1	capacitive link impedance for C1
Zc2	capacitive stub impedance for C2
Zls	stub inductance for voltage source
Vs	source voltage
nVls	physical voltage on source inductance, at time step n
nVlsi	incident voltage on source inductance, at time step n
nVlsr	reflected voltage on source inductance, at time step n
nVc11	physical voltage on C1, at time step n, on sub-circuit 1
nVc11i	incident voltage on C1, at time step n, on sub-circuit 1
nVc11r	reflected voltage on C1, at time step n, on sub-circuit 1
nVc12	physical voltage on C1, at time step n, on sub-circuit 2
nVc12i	incident voltage on C1, at time step n, on sub-circuit 2

nVc12r reflected voltage on C1, at time step n, on sub-circuit 2
nVc2 physical voltage on C2, at time step n
nVc2i incident voltage on C2, at time step n

It can be shown that

$$Zc1=Tstep/C1$$
$$Zc2=Tstep/(2*C2)$$
$$Zls=2*Ls/Tstep$$

where Tstep is the time step chosen for the simulation.

Consider sub-circuit 1,

$$\begin{bmatrix} (Zls+Rd1+Rd3) & (Rd1+Rd3) & (-Rd3) \\ (Rd1+Rd3) & (Rd1+Rd2+Rd3+Rd4) & -(Rd2+Rd3) \\ (-Rd3) & -(Rd2+Rd3) & (Rd2+Rd3+Zc1) \end{bmatrix} \begin{bmatrix} i1 \\ i2 \\ i3 \end{bmatrix} = \begin{bmatrix} Vs-2*Vlsi \\ 0 \\ 2*nVc11i \end{bmatrix} \quad (1)$$

Initial values of Vs, Vlsi, nVc11i, nVc12i and nVc2i are assumed to be zero. Solving these three simultaneous equations gives i1, i2 and i3 at any time step n. Then the physical voltages across capacitor C1 and source inductance Ls in sub-circuit 1 can be evaluated by

$$nVls=i1*Zls+2*nVlsi \quad (2)$$
$$nVc11=i3*Zc1+2*nVc11i$$

Since the physical voltage at time step n across a component is the sum of its incident voltage and reflected voltage, we have

$$nVc11r=nVc11-nVc11i \quad (3)$$
$$nVlsr=nVls-nVlsi$$

Each of these reflected voltages travels along the respective transmission lines and becomes the incident voltage at the next time step. For capacitance C1 modelled as a TLM link, this becomes the incident voltage in sub-circuit 2. For the source inductance Ls modelled as a short circuited TLM stub, this becomes the incident voltage on itself. Hence

$$n+1Vc12i=nVc11r \quad (4)$$
$$n+1Vlsi=-nVlsr$$

Thus we can proceed into the next time step by repeating the above process.

Similarly, consider sub-circuit 2,

$$\begin{bmatrix} Zc1+Rsw1+R1+Zc2 & -Zc2 \\ -Zc2 & Zc2+Rsw2+R2 \end{bmatrix} \begin{bmatrix} i4 \\ i5 \end{bmatrix} = \begin{bmatrix} 2(nVc12i-nVc2i) \\ 2*nVc2i \end{bmatrix} \quad (5)$$

From this, we can solve for i4 and i5 in any time step n. From this, the physical voltages in time step n can be calculated,

$$nVc12=2*nVc12i-i4*Zc1 \quad (6)$$
$$nVc2=2*nVc2i+(i4-i5)*Zc2$$

Then the reflected voltages in time step n and the incident voltages in time step n+1 can be evaluated,

n+1Vc11i=nVc12r=nVc12-nVc12i (7)
n+1Vc2i=nVc2r=nVc2-nVc2i

PARALLEL SIMULATION

Since there are two sub-circuits in the equivalent circuit, two program modules are used, one for each sub-circuit. A third program module is included for overall control, I/O handling and house-keeping. The structure chart in fig. 4 shows the overall program structure, where slave 1 simulates the behaviour of sub-circuit 1 while slave 2 simulates sub-circuit 2, and master is the central control module. Also shown are the data parameters between these modules.

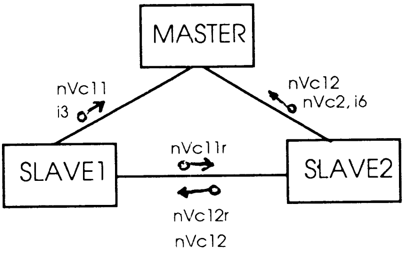


Fig. 4 Structure Chart

Each of these three program modules perform different functions. They run in parallel, independent of one another. At the same time, they are synchronised periodically through information exchange at programmed intervals. For example, slave 1 and slave 2 exchange the reflected voltages at the end of every time step, while the master reads the simulated voltages and currents from the two slaves at intervals specified by the user. Simulation is therefore able to proceed in parallel.

IMPLEMENTATION

The Inmos transputer system was chosen because it is commonly used for parallel computing and is relatively cost effective. The hardware platform consists of an IBM PC 486, a transputer mother board H121-201 with five T805 transputer modules (TRAMS), each with 1 MBytes of on board memory. The system software used is the Inmos Parallel C compiler. Each of the three program modules was written as individual programs using Inmos Parallel C. The number of transputers used is discussed in the next section.

It is interesting to note some implementation details here. Each Inmos transputer has four bi-directional physical links for communication with other transputers. Between them, some links are set up permanently in a daisy chain manner by hardware, while other links are all connected to a C004 Link Switch. This set up shown in fig. 5 gives a great deal of flexibility.

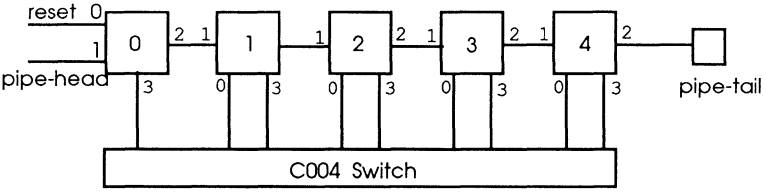


Fig. 5 Transputer Links

Before a program can be run, the C004 Link Switch must be programmed to connect the remaining transputer links to a physical setup identical to the logical configuration required by the program. An example to set up the links in the soft switch for this simulation using 3 transputers is shown in the following:

```
SOFTWIRE
   PIPE 0
   SLOT 0, LINK 3 TO SLOT 2, LINK 0
END
```

Besides the three program modules, a configuration file is also needed. This configuration file first defines the required physical hardware structure and the logical program structure. It then specifies how the logical structure is mapped to the physical set up of transputers. This ensures the program modules can be loaded correctly into the respective transputers before execution starts. Part of the configuration file that specifies how the master module is mapped onto transputer 1 in slot 0 is included below as an example.

```
#include "setup.inc"
/*
The configuration uses three T800s, on which is placed a software network.
The software consists of three simple processes, Master, Slave1 and Slave2. */
/*  Hardware description */
T800 (memory = 1M) Trans1;
T800 (memory = 1M) Trans2;
T800 (memory = 1M) Trans3;
connect Trans1.link[BootLink] to host; /* Host link connection */
connect Trans1.link[2] to Trans2.link[1];

/*  Software description */
process (interface (input HostInput, output HostOutput,
      input C1ToM, output MToC1, input C2ToM, output MToC2))
      Master;
```

```
/*-- Describe how things are wired up  */
input  from_host;
output to_host  ;
connect Master.HostInput, from_host;
connect Master.HostOutput,   to_host;
connect Master.C1ToM, Slave1.C1ToM;
connect Master.MToC1, Slave1.MToC1;
connect Master.C2ToM, Slave2.C2ToM;
connect Master.MToC2, Slave2.MToC2;

/*  Mapping description */
use "Master.lku" for Master;
place Master on Trans1;
place from_host on host;
place to_host on host;
```

Only the first transputer, T1, has a link to the PC. Consequently, only the program module running on T1 can perform I/O directly, such as reading from keyboard or outputting messages to screen. The other program modules cannot perform I/O directly. Any input or output statements in these modules will cause linking error. This caused some difficulties in program debugging.

RESULTS AND DISCUSSION

The simulated output voltage is shown in fig. 6.

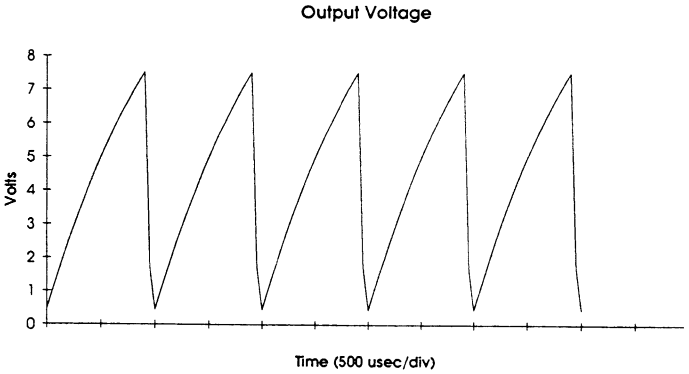


Fig. 6 Simulated Output Voltage

Because the sawtooth waveform generator circuit is fairly straight forward, only simulated results are given here. Simulation of 10 mains cycles (50 Hz) with 1 microsecond time step, running on 1, 2 and 3 transputers were carried out. The respectively execution times for each are recorded. When 1 transputer is used, standard sequential computing is adopted. In the case of two transputers, the master module and sub-circuit 2 were combined to run in one transputer, while sub-circuit 1 runs on the other transputer. When 3 transputers are used, each of the master and slaves occupy one transputer. These results are listed as follows:

Configuration	1 Transputer	2 Transputers	3 Transputers
Execution time	16 min 1 sec	12 min 22 sec	12 min 18 sec

Table 1 Execution time comparison

Using the computing time with 1 transputer (sequential) as a basis for comparison purpose, the use of 2 transputers offers a 22.8% time reduction. Using 3 transputers gives a marginal 23.2% reduction. It should be noted that the implementation of parallel algorithm using 2 transputers does not result in 50% time reduction. This is due to the fact that (1). the computing load in each transputer is not necessarily identical and (2). the transputers are fast on vertical computation but slow on horizontal communication. The gain in using 3 transputers is very marginal. This is because in the circuit chosen for simulation, there are only two sub-circuits. The transputer on which the master runs does little and is idle most of the time.

Generally, the actual speed gain one gets in using transputers depends on whether the problem is compute-intensive or not. The performance will improve on more complex circuit problems which are more computation oriented. Besides, the possibility of decoupling a more complex circuit into a larger number of sub-circuits means more transputers can be used in the simulation. The speed will also improve, though not proportionally.

CONCLUSION

This paper describes a parallel simulation technique for circuit analysis. A parallel simulation of an electrical circuit has been achieved successfully via the use of the TLM decoupling technique and a transputer system. This approach not only simplifies the circuit formulation process, but also reduces the overall computing time when compared with standard method.

REFERENCE

1. S.Y.R. Hui, K.K. Fung and C. Christopoulos 'Decoupled Simulation of Multi-stage Power Electronic Systems using Transmission-line Links', Vol 2 pp. 1324 to 1330 , *Proceedings of the Power Electronics Specialist Conference*, Toledo, Spain, July 1992
2. H.H. Happ, *Diakoptics and networks,* Academic Press 1971
3. P.B. Johns and R.L. Beurle, 'Numerical solution of 2-dimensional scattering problem using a transmission-line matrix', *Proc. IEE, 118*, pp.1203 to 1208, 1971