A Brief SPICE (Pspice) Tutorial

Clayton R. Paul, Mercer University, Macon, GA (USA), paul_cr@mercer.edu

Abstract—**A brief but sufficient tutorial on the use of SPICE or its personal computer version, Pspice, in solving lumped-circuit models of EMC problems is given. All electrical engineers and especially EMC engineers should develop a working knowledge of the use of Pspice and should use it as frequently as they use their common calculators in solving electric-circuit models of everyday EMC problems.**

This is a brief summary of the SPICE, or its personal computer version Pspice, electric circuit analysis program. SPICE is an acronym for Simulation Program with Integrated-Circuit Emphasis. The original SPICE computer program was developed to analyze complex electric circuits, and particularly integrated circuits. It was developed at the University of California at Berkeley in the late 1960's. Since it was developed under U.S. government funding, it is not proprietary and can be freely copied, used and distributed. This was written for use on large mainframe computers of the time. In the early 1980's the MicroSim Corporation developed a personal computer version of SPICE called Pspice. A number of important modifications were made particularly in the plotting of data via the .PROBE function. Since then a number of commercial firms have modified and developed their own PC versions. But essentially the core engine is that of the original SPICE code. The MicroSim version of Pspice, version 8, was acquired by Cadence Design Systems. Theirs is version 10.0 called OrCAD Capture which also contains the primary simulation code Pspice A/D. A windows based version is available free from www.cadence.com. The OrCAD Capture program was originally called Schematic in the MicroSim version and contains a number of enhancements. A number of books [1–5] detail the use of SPICE and Pspice. Both the MicroSim version 8 and the OrCAD version 10 are contained in a CD at the end of textbooks [6–7].

There are two methods of entering and executing a Pspice program. The first method is the *Direct Method* described here where one enters the program code using an ASCII text editor (supplied with Pspice). Note: SPICE and Pspice make no distinction between lower case and upper case letters. Then

this text file is run using the Pspice A/D section of the program and the output is examined with the text editor. The second method is the *Schematic Method* (now called Capture) where the user "draws" the circuit diagram directly on the screen and then executes that program. The Direct Method is generally the most rapid method of solving relatively simple problems. The Schematic (Capture) Method has the advantage of visually seeing whether the circuit components are connected as intended for more complex circuits but is a bit more time consuming to setup than the Direct Method for most simple EMC problems since numerous windows and drop down menus must be navigated in the Schematic or Capture method.

Once the Pspice program has been installed on your computer, the following is a description of how you can input your program, run it, and examine the output. Although there are several ways of doing this, the simplest is to use the Design Manager. To load this you click or select the following in this sequence.

1) Start
2) Programs
3) MicroSim Eval 8
4) Design Manager

The Direct Method is to simply type in the Pspice program using the TextEdit feature. To enter this and prepare the program we select the following in this sequence.

1) TextEdit (lower button on the vertical toolbar on the left)
2) Type the program
3) Save the program as XXX.cir or XXX.in and close it
4) Select Pspice A/D (second button on the vertical toolbar on the left)
5) Click on File, Open and select the previously stored file. The program will automatically run and the output will be stored in file XXX.out.
6) Click on File, Run Probe in order to plot waveforms or File, Examine Output in order to examine the printed output.
7) Alternatively you could recall the TextEdit progam and select File, Open, XXX.out to examine the printed output which is self explanatory.

I. Creating the SPICE or Pspice Program

SPICE and Pspice write the node-voltage equations of an electric circuit [1]. One node, the reference node for the node voltages, is designated the zero (0) node. All circuits must contain a zero node. The other nodes in the circuit are labeled with numbers or letters. For example, a node may be labeled 23 or it may be labeled FRED. The voltages with respect to the reference node are positive at the node and denoted as V(N1), V(N2), etc. as shown in Fig. 1. The general structure of any SPICE or Pspice program is as follows:

1. Title
2. Circuit Description
3. Execution Statements
4. Output Statements
5. END

The first line of the SPICE program is the Title and is not processed by SPICE. It is simply written on the output and any plots. A comment line is started with an asterisk (*) and is also not processed by the program. A line may be continued with a plus sign (+) at the beginning of the following line. The next set of lines, Circuit Description, describes the circuit elements and their values and tells SPICE how they are connected together to form the circuit. The next set of lines are the Execution Statements that tell SPICE what type of analysis is to be run: dc analysis (.DC), sinusoidal steady state or phasor analysis (.AC), or the full time-domain analysis consisting of the transient and steady state solution (.TRAN). The next set of statements, Output Statements, tell SPICE what outputs are desired. The results can be printed to a file with the .PRINT statement or can be plotted with the .PROBE feature. And finally, all programs must end with the .END statement. Actually the above items 2–4 can appear in any order in the program but the program must begin with a Title statement and end with the .END statement.

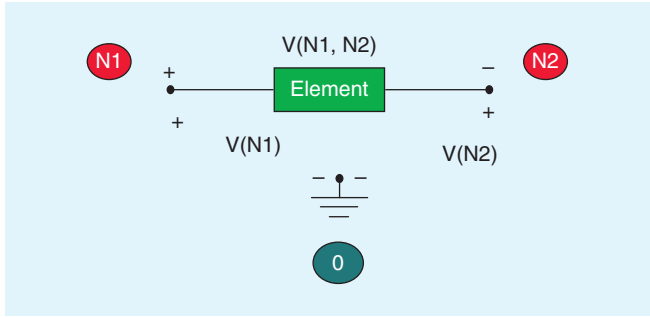

Fig. 1. Node voltage and element voltage notation in the SPICE (PSPICE) circuit analysis program.

II. Circuit Description

The basic elements and their SPICE descriptions are shown in Fig. 2. Fig. 2(a) shows the independent voltage source. It is named starting with the letter V and then any other letters. For example, a voltage source might be called VFRED. It is connected between nodes N1 and N2. It is very important to note that *the source is assumed positive at the first-named node*. The current through the voltage source is designated as I(VXXX) and is assumed to flow from the first-named node to the last-named node. The source type can be either dc for which we append the terms DC *magnitude*, or a sinusoid, to which we append the terms AC *magnitude phase (degrees)*. A time-domain waveform is described by several functions which we will describe later, and these descriptions are appended (without the word TRAN). The independent current source is shown in Fig. 2(b). Its name starts with the letter I followed by any other letters. For example a current source might be designated as ISAD. *The current of the source is assumed to flow from the first-named node to the last-named node*. The types of source waveforms are the same as for the voltage source.

The resistor is shown in Fig. 2(c) and its name starts with the letter R, e.g., RHAPPY. The current through the resistor is designated as I(RXXX) and is assumed to flow from the first-named node to the last-named node. SPICE does not allow elements with zero values. Hence a resistor whose value is 0 ohms (a short circuit) may be represented as having a value of 1E-8 or any other suitably small value. Similarly, an open circuit may be designated as a resistor having a value of 1E8 or any other suitably large value. Every node must have at least two elements

connected to it. Also SPICE requires that every node must have a dc path to ground (the zero (0) node). Placing a large resistor, e.g. 1MEG Ohm, between such nodes fixes this problem.

The inductor is shown in Fig. 2(d) and is designated with the letter L, e.g., LTOM. The current through the inductor as well as the initial inductor current at $t = 0^+$, I(0), is assumed to flow from the first-named node to the last-named node. The initial condition can be specified at the end of the statement with IC=I(0). The capacitor is shown in Fig. 2(e) and is designated with the letter C, e.g., CME. The initial voltage across the capacitor at $t = 0^+$, V(0), can be specified at the end of the statement with $IC = V(0)$, and this voltage is assumed to be positive at the first-named node.

All numerical values can be specified in powers of ten and written in exponential format, e.g., $2 \times 10^{-5} = 2\text{E} - 5$, or by using standard multipliers using standard engineering notation:

Multiplier	SPICE Symbol
10^9 (giga)	G
10^6 (mega)	MEG
10^3 (kilo)	K
10^{-3} (milli)	M
10^{-6} (micro)	U
10^{-9} (nano)	N
10^{-12} (pico)	P

For example, 1 megohm is written as 1MEG, 1 kiloohm is written as 1K, 3 millihenries is written as 3M, 5 microfarads is written as 5U, 2 nanohenries is written as 2N, and 7 picofarads is written as 7P. A 3 farad capacitor should not be written as 3F since F stands for femto $= 10^{-15}$.

The four types of controlled sources, G,E,F,H, are shown in Fig. 3 along with their descriptions. The polarities of voltage and the currents through the elements conform to the previous rules governing these in terms of the first- and last-named nodes on their description statements. For a current-controlled source, F or H, the controlling current must be through an independent voltage source. Often we insert a 0 V source to sample the current. Some recent versions of Pspice allow the specification of the current through any element as a controlling current. But it is always a simple matter to insert a zero-volt voltage source.

Figure 4 shows how to specify mutual inductance. First the self inductances that are coupled are specified as before. The mutual inductance is specified in terms of its coupling coefficient:

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

In order to keep the polarities correct, define the self inductances so that the dots are on the first-named nodes when the two inductors are defined; otherwise a negative coupling coefficient may need to be specified.

Figure 5 shows the last important element, the transmission line (lossless), that is used extensively in Signal Integrity analyses. This gives the *exact solution* of the transmission-line equations for a *lossless* line. There are many

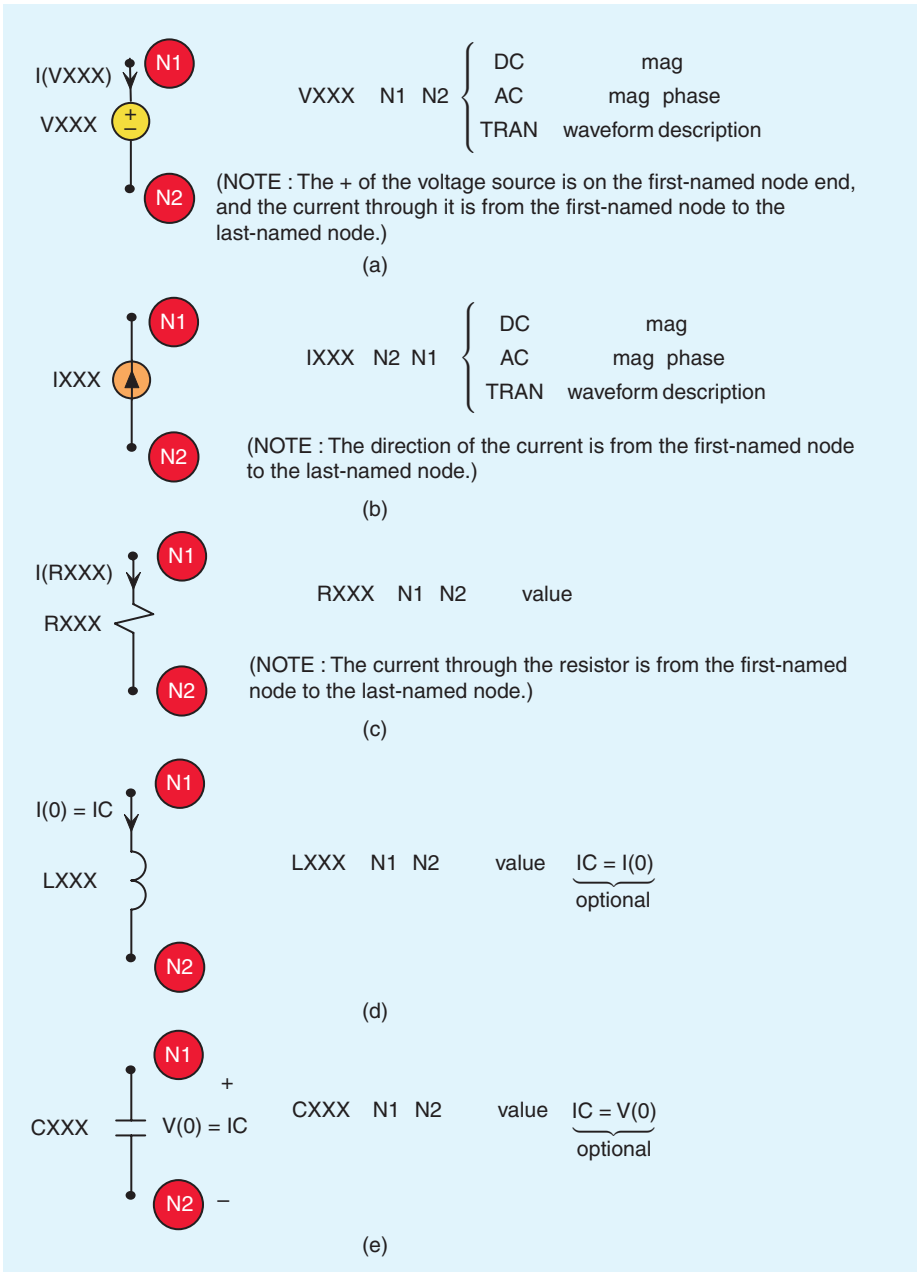

Fig. 2. Coding convention for (a) the independent voltage source, (b) the independent current source, (c) the resistor, (d) the inductor, and (e) the capacitor.

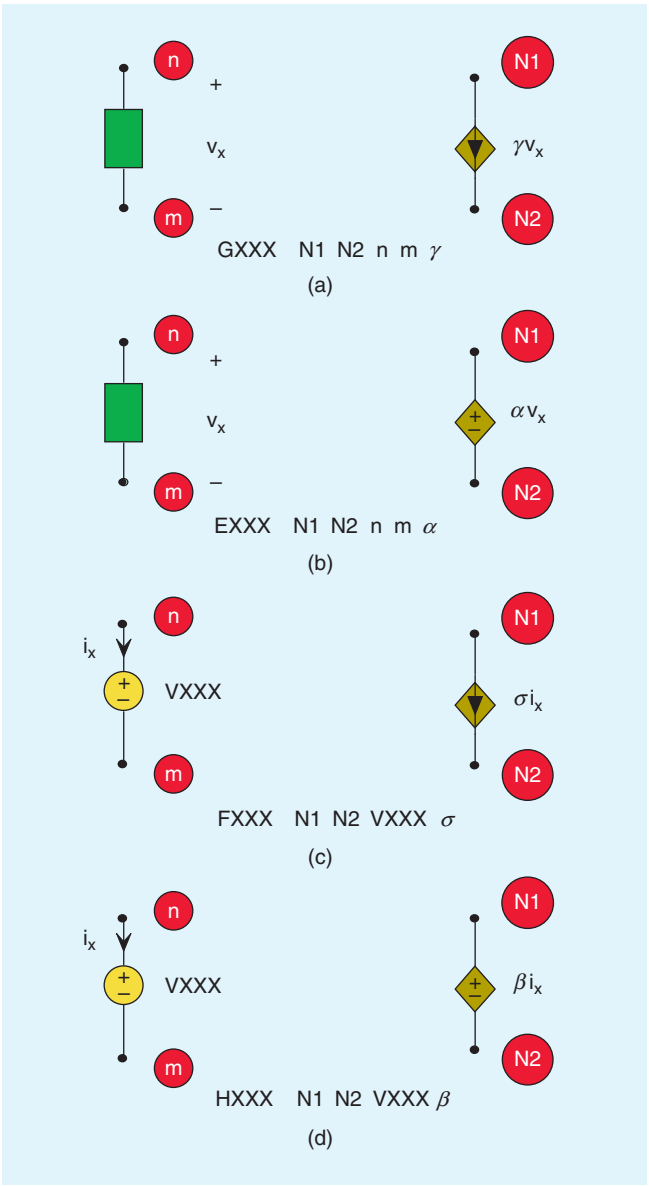

Fig. 3. Coding convention for (a) the voltage-controlled current source, (b) the voltage-controlled voltage source, (c) the current-controlled current source, (d) the current-controlled voltage source.

ways to specify the important parameters for the (lossless) line but the one shown in the figure is the most widely used; specify the characteristic impedance of the line and the line's one-way time delay. Alternatively you can create a lumped RLCG *approximate* model of a *lossy* line using lumped-circuit elements [6, 7]. But the per-unit-length parameters of resistance R and conductance G in that model must be constants although these are, in reality frequency dependent which is not readily handled in an exact solution of the transmission-line equations [6–7].

Figure 6 shows how to specify the important time-domain waveforms. Figure 6(a) shows the PWL (piecewise-linear) waveform where straight lines are drawn between pairs of points that are specified by their time location and their value. Observe that the function holds the last specified value, V4 in the figure. Figure 6(b) shows the periodic pulse waveform, PULSE, that is used to specify periodic clock or other timing waveforms. The function specifies a trapezoidal waveform

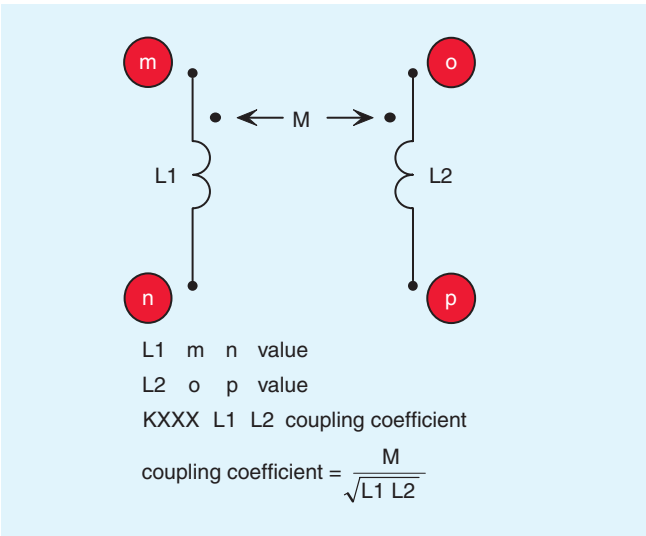

Fig. 4. Coding convention for mutual inductance between two coupled inductors.

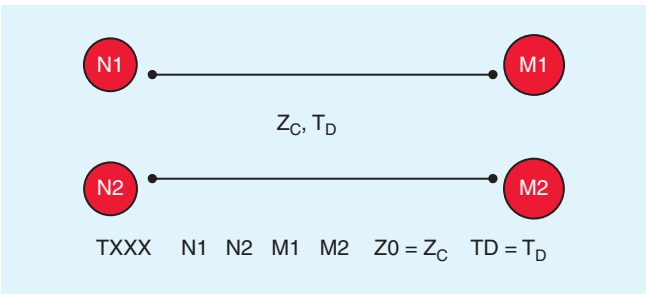

Fig. 5. Coding convention for the two-conductor, lossless transmission line.

that repeats periodically with period PER (the reciprocal is the fundamental frequency of the waveform). Note that the pulse width, PW, is not specified between the 50% points of the pulse as is the usual convention.

The sinusoidal function is specified by

$$\text{SIN(Vo Va [[Freq [[Td [[Df [[Phase]]]]]]]])}$$

which gives the waveform

$$x(t) = Vo + Va\sin\left(2\pi\left(\text{Freq}(\text{time} - \text{Td}) + \frac{\text{Phase}}{360}\right)\right)e^{-(\text{time}-\text{Td})\text{Df}}$$

Brackets around items signify that they are optional. Hence to specify the general sinusoidal waveform

$$x(t) = A\sin(n\omega_0 t + \theta)$$

where $\omega_0 = 2\pi f_0$ and $f_0 = {}^1/_{\text{PER}}$ is the frequency of the sinusoid we would write

$$\text{SIN}(0\ \ A\ \ nf_0\ \ 0\ \ 0\ \ \theta)$$

III. Execution Statements

There are three types of solutions: dc, sinusoidal steady state or phasor, and the full time-domain solution (so-called transient

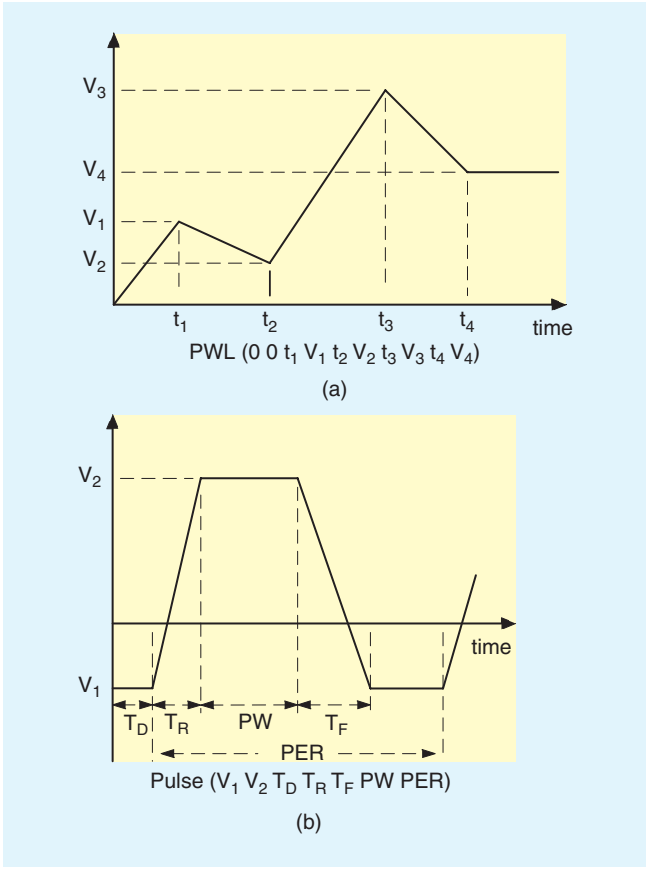

Fig. 6. Coding convention for the important source waveforms: (a) the piecewise-linear waveform and (b) the pulse source waveform (periodic).

although it contains both the transient and the steady-state parts of the solution).

The dc solution is specified by

.DC V,IXXX *start_value end_value increment*

where V,IXXX is the name of a dc voltage or current source in the circuit whose value is to swept. For example to sweep the value of a dc voltage source VFRED from 1 V to 10 V in increments of 2 V and solve the circuit for each of these source values we would write

.DC VFRED 1 10 2

If no sweeping of any source is desired then simply choose one dc source in the circuit and iterate its value from the actual value to the actual value and use any nonzero increment. For example,

.DC VFRED 5 5 1

The sinusoidal steady-state or phasor solution is specified by

.AC {LIN,DEC,OCT} *points start_ frequency end_ frequency*

LIN denotes a linear frequency sweep from *start_ frequency* to *end_ frequency* and *points* is the total number of frequency points. DEC denotes a log sweep of the frequency where the frequency is swept logarithmically from the *start_ frequency*

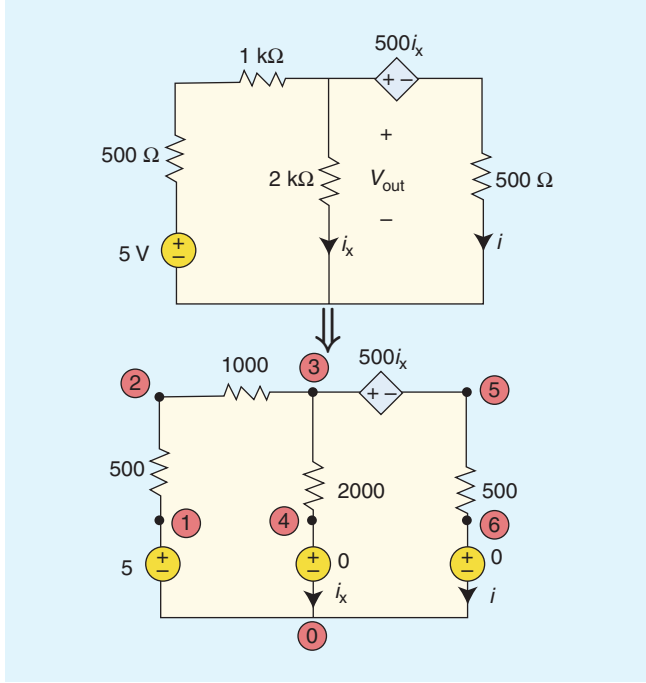

Fig. 7. Example 1, an example of a DC analysis.

to the *end_ frequency* and *points* is the number of frequency points per decade. OCT is a log sweep by octaves where *points* is the number of frequency points per octave.

The time-domain solution is obtained by specifying

.TRAN *print_step end_time* [*no_print_time*] [*step_ceiling*] [*UIC*]

SPICE solves the time-domain differential equations of the circuit by discretizing the time variable into increments of Δt and solving the equations in a boot-strapping manner. The differential equations of the circuit are first solved at $t = 0$ Then that solution is used to give the solution at $t = \Delta t$. These prior time solutions are then used to give the solution at $t = 2\Delta t$ and so on. The first item, *print_step*, governs when an output is requested. Suppose the discretization used in the solution is every 2ms. We might not want to see (in the output generated by the .PRINT statement) an output at every 2ms but only every 5ms. Hence we might set the *print_step* time as 5M. The *end_time* is the final time that the solution is obtained for. The remaining parameters are optional. The analysis always starts at $t = 0$. But we may not wish to see a printout of the solution (in the output generated by the .PRINT statement) until after some time has elapsed. If so we would set the *no_print_time* to that starting time. SPICE and Pspice have a very sophisticated algorithm for determining the minimum step size, Δt, for discretization of the differential equations in order to get a valid solution. The default maximum step size is *end_time*/50. However, there are many cases where we want the step size to be smaller than what SPICE would allow in order to increase the accuracy of the solution or to increase the resolution of the solution waveforms. This is frequently the case when we use SPICE in the analysis of transmission lines where abrupt waveform transitions are occurring at every one-way time delay. The *step_ceiling* is the maximum time step size, $\Delta t|_{\max}$, that will be used in discretizing the differential equations as described

previously. Although this gives longer run times, there are cases where we need to do this to generate the required accuracy. The last item *UIC* means that SPICE is to use the initial capacitor voltage or inductor current specified on the element specifications with the *IC* = command. In a transient analysis, SPICE will compute the initial conditions at $t = 0^+$. If some other initial conditions are required, then we should set these on the capacitor or inductor specifications with the *IC* = command and specify *UIC* on the .TRAN statement. For example,

.TRAN 0.5N 20N 0 0.01N

would command SPICE to do a full time-domain (transient plus steady state) analysis for times from 0 to 20 ns, print out a solution at every 0.5 ns, start printing to the output file at $t = 0$, and use a time discretization time step no larger than 0.01 ns.

IV. Output Statements

The output statements are either for printing to a file with the .PRINT statement or producing a plotted graph of any waveform with the .PROBE statement. The .PRINT statement has three forms depending on the type of analysis being run. For a DC analysis

.PRINT DC V(X) I(R)

prints the dc solution for the voltage of node X with respect to the reference node, and I(R) prints the dc solution for current through resistor R (defined from the first-named node to the last-named node on the specification statement for resistor R). For a sinusoidal steady-state analysis (phasor solution):

.PRINT AC VM(NI) VP(NI) IM(RFRED) IP(RFRED)

prints the magnitude and phase of node voltages and currents where the magnitude and phase of the node voltage at node NI are VM(NI) and VP(NI), respectively. For the currents through a resistor RFRED, the magnitude is IM(RFRED) and the phase is IP(RFRED). For the time-domain or so-called transient analysis the print statement is

.PRINT TRAN V(NI) I(RFRED)

prints the solutions at all solution time points (specified on the .TRAN line) for the voltage at node NI with respect to the reference node, and the current through resistor RFRED (defined from the first-named node to the last-named node on the specification statement for resistor RFRED).

In addition, the .FOUR statement computes the expansion coefficients for the Fourier series (magnitude and phase), $c_n \angle c_n$:

.FOUR f_0 [*output_variable(s)*]

The .FOUR command can only be used in a .TRAN analysis. The fundamental frequency of the *periodic* waveform to be analyzed is denoted as $f_0 = 1/T$ where T is the period of the waveform. The *output_variable(s)* are the desired voltage or current waveforms to be analyzed, e.g., V(2), I(R1). The phase results are with reference to a sine form of the series:

$$x(t) = c_0 + \sum_{n=1}^{\infty} c_n \sin(n\omega_0 t + \angle c_n)$$

There is an important consideration in using the .FOUR command. The portion of the waveform that is analyzed to give the Fourier expansion coefficients is the last portion of the solution time of length one period $1/f_0 = T$. In other words, SPICE determines the coefficients from the waveform between *end_time*–[$1/f_0$] and *end_time*. Hence, *end_time* on the .TRAN command should be at least one period long. In situations where the solution has a transient portion at the beginning of the solution interval and we want to determine the Fourier coefficients for the steady-state solution, we would run the analysis for several periods to insure that the solution has gotten into steady state. For example consider an input signal that is periodic with a period of 2 ns or a fundamental frequency of 500 MHz. An output voltage at node 4 would also have this periodicity but would have a transient period of at least five time constants. Suppose the maximum time constant of the circuit is 4 ns. Then we would set the end time to 20 ns or more in order to get into the steady state region of the solution at the end of the solution. The following commands would be used to obtain the Fourier coefficients of the steady-state response of the node voltage at node 4:

.TRAN 0.1N 20N

.FOUR 500MEG V(4)

This would compute the solution for the voltage waveform at node 4 from $t = 0$ to $t = 20$ ns. Since the period (the inverse of 500 MHz) is specified as 2 ns, the portion of the waveform from 18 ns to 20 ns would be used to compute the Fourier coefficients for the waveform. If we wanted to compute the Fourier coefficients for the initial part of the waveform including the transient, we would specify

.TRAN 0.1N 2N

which would run for only one period. There is a FFT button on the tool bar that can compute the Fast Fourier Transform of the waveform. Essentially this valuable feature can turn Pspice into a "poor man's spectrum analyzer".

All printed output are directed to a file named XXXX.OUT if the input file is named XXXX.IN or XXXX.CIR. Plotting waveforms is the greatest enhancement of Pspice over the original SPICE. This is invoked by simply placing the .PROBE statement in the program list. No additional parameters are required. Pspice stores all variables at all solution time points and waits for the user to specify which to plot.

V. Examples

In this brief tutorial we have shown the basic commands that one can use to solve the vast majority of electric circuit analysis problems particularly those that are encountered in EMC problems. Essentially Pspice performs the tedious and laborious solutions of the lumped-circuit models of the problem. Solving these circuits by hand would not only be time consuming but would involve so many errors as to make the solutions obtained useless. We have conscientiously tried to minimize the detail

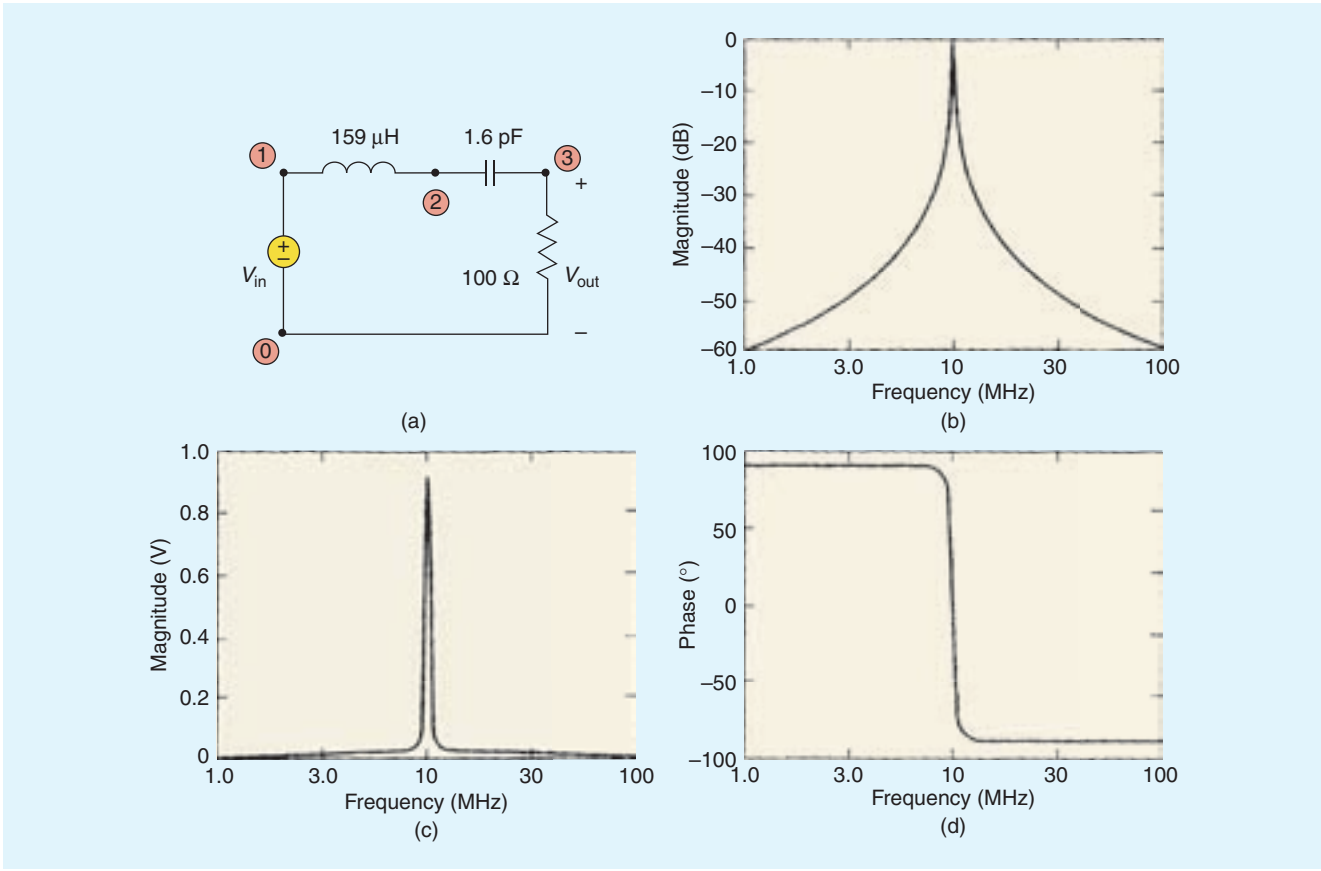

Fig. 8. Example 2, an example of a AC analysis.

and purposely not shown all the possible options in order to simplify the learning. However there are a myriad of options that can simplify many computations and the reader should consult the references.

EXAMPLE 1 Use Pspice to compute the voltage V_{out} and the current I in the circuit of Fig. 7.

Solution The Pspice coding diagram with nodes numbered is shown in Fig. 7. Zero-volt voltage sources are inserted to sample the currents i_x and I. The Pspice program is

```
EXAMPLE 1
VS 1 0 DC 5
R1 1 2 500
R2 2 3 1K
R3 3 4 2K
VTEST1 4 0 DC 0
HSOURCE 3 5 VTEST1 500
R4 5 6 500
VTEST2 6 0 DC 0
.DC VS 5 5 1
*THE CURRENT I IS I(VTEST2) AND THE VOLTAGE
+VOUT IS V(3) OR V(3,4)
.PRINT DC V(3) I(VTEST2)
.END
```

The result is I=I(VTEST2) =1.875E-3 and the voltage Vout=V(3)=1.250E0.

EXAMPLE 2 Use Pspice to plot the frequency response of the bandpass filter shown in Fig. 8(a).

Solution The nodes are numbered on the circuit diagram and the Pspice program is

```
EXAMPLE 2
VS 1 0 AC 1 0
RES 3 0 100
LIND 1 2 159U
CAP 2 3 1.6P
.AC DEC 50 1MEG 100MEG
.PROBE
*THE MAGNITUDE OF THE OUTPUT IS VM(3) AND
+THE PHASE IS VP(3)
.END
```

The magnitude of the voltage is plotted in Fig. 8(b) in decibels using VDB(3) which means

$$\mathbf{VDB(3)=20\log_{10}VM(3)}$$

Figure 8(c) shows what we get if we request VM(3): the data are highly compressed outside the bandpass region. The phase (in degrees) is plotted in Fig. 8(d). The resonant frequency is 10 MHz. The phase is $+90^{o}$ below the resonant frequency due to the dominance of the capacitor in this range and is -90^{o} above the resonant frequency due to the dominance of the inductor in this range. This bears out the important behavior of a series resonant circuit.

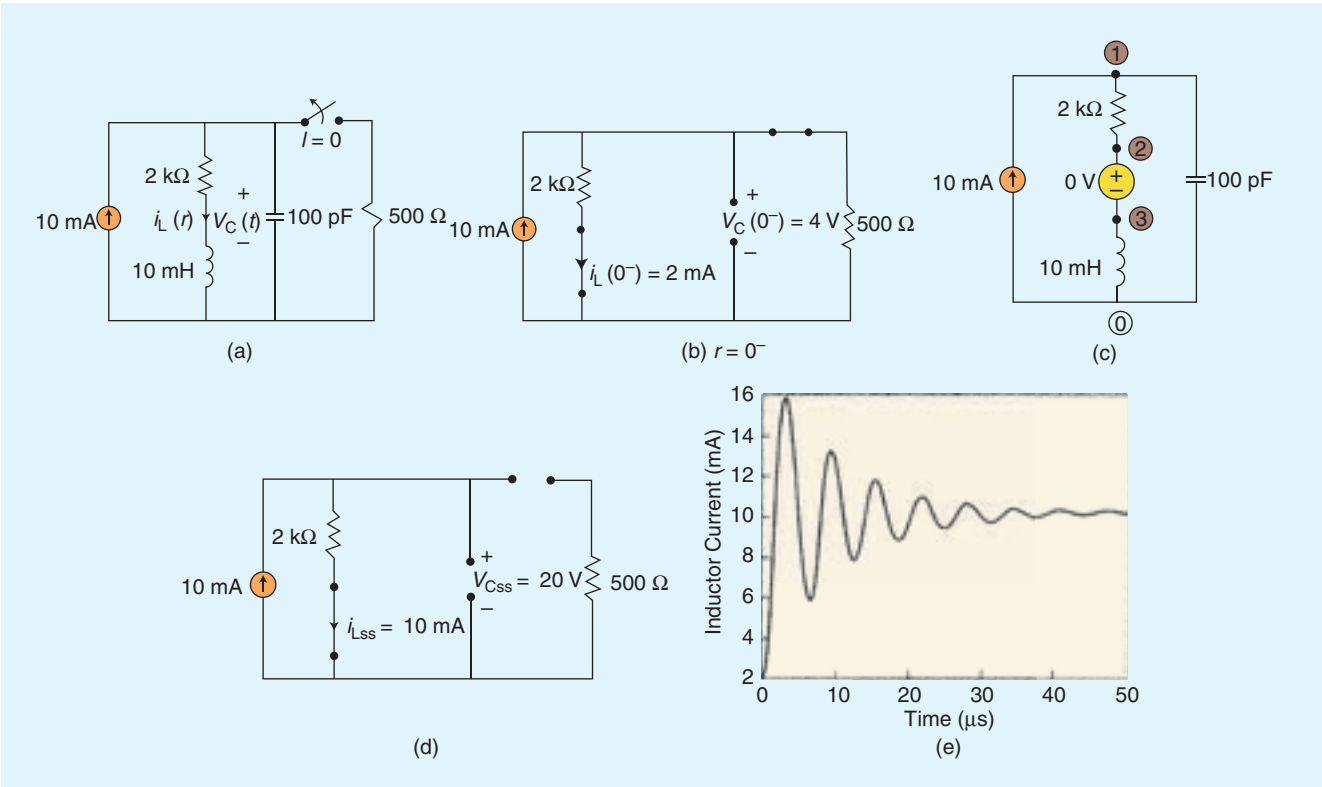

Fig. 9. Example 3, an example of a TRAN analysis.

EXAMPLE 3 Use Pspice to plot the inductor current for $t > 0$ in the circuit of Fig. 9(a). The circuit immediately before the switch opens, i.e., at $t = 0^-$, is shown in Fig. 9(b) from which we compute the initial voltage of the capacitor as 4 V and the initial current of the inductor as 2 mA. The Pspice diagram with nodes numbered is shown in Fig. 9(c) and the Pspice program is

```
EXAMPLE 3
IS 0 1 DC 10M
R 1 2 2K
VTEST 2 3
L 3 0 10M IC=2M
C 1 0 100P IC=4
.TRAN .05U 50U 0 .05U UIC
*THE INDUCTOR CURRENT IS I(VTEST) OR I(L)
.PROBE
.END
```

We have chosen to solve the circuit out to 50 μs, print the solution in steps of 0.05 μs, and have directed Pspice to use a solution time step no larger that 0.05 μs as well as to use the initial conditions given for the inductor and capacitor. The result is plotted using PROBE in Fig. 9(e). The result starts at 2 mA, the initial inductor current, and eventually converges to the steady-state value of 10 mA, which can be confirmed by replacing the inductor with a short circuit and the capacitor with an open circuit in the $t > 0$ circuit as shown in Fig. 9(d).

EXAMPLE 4 Figure 10 shows an example where an interconnecting set of conductors (lands on a PCB) can cause severe

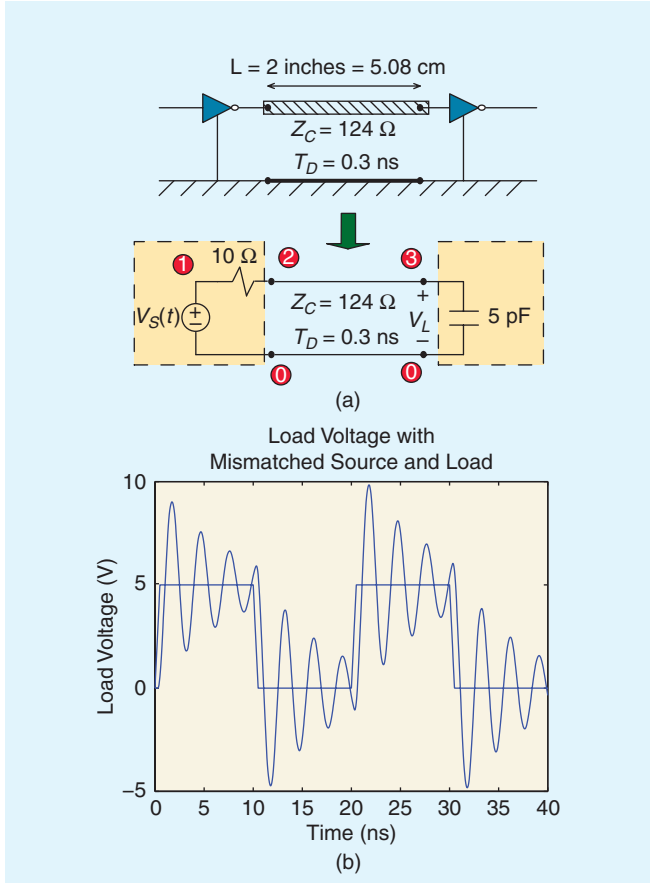

Fig. 10. Example 4, an example of a Signal Integrity analysis.

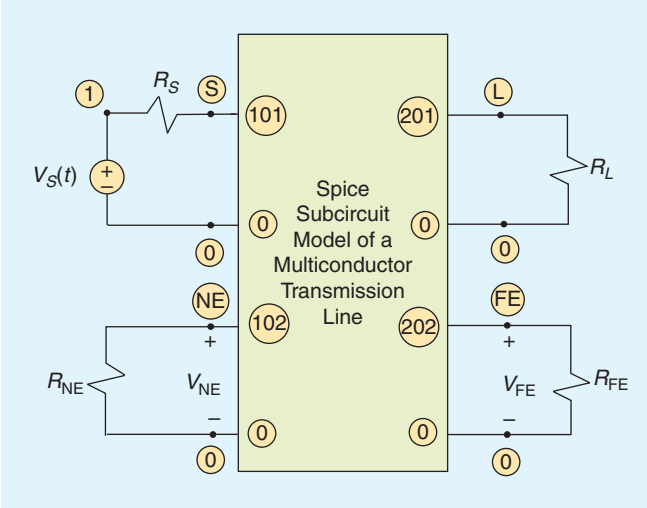

Fig. 11. Connection of a SPICE subcircuit model of a multiconductor transmission line (MTL) to its terminations.

logic errors resulting in poor signal integrity. Two CMOS inverters (buffers) are connected by two inches of lands (= 2 inches = 5.08 cm) on a PCB. The output of the left inverter is shown as a Thevenin equivalent circuit having a low source resistance of 10 ohms. This is fairly typical of CMOS devices except that the output resistance is somewhat nonlinear. The load on the line is the input to the other CMOS inverter which is represented as a 5pF capacitor that is also typical of the input to CMOS devices. We are interested in determining the voltage at the output of the interconnect line, $V_L(t)$, which is the voltage at the input to the second CMOS inverter. The open-circuit voltage of the left inverter, $V_S(t)$, is a 5 V, 50 MHZ (period of 20 ns), 50% duty cycle, clock trapezoidal waveform having 0.5 ns rise/fall times. A 10 mil land lying on a glass epoxy PCB of 47 mils thickness has a ground plane below it. For the cross-sectional dimensions of the line the characteristic impedance is $Z_C = 124\Omega$, and the velocity of propagation is $v = 1.7 \times 10^8$m/s. This gives a one-way time delay of $T_D = L/v = 0.0508\text{m}/1.7 \times 10^8\text{m/s} = 0.3$ ns. The response for the load voltage is computed with Pspice using the exact (lossless) transmission-line model contained in Pspice and is shown in Fig. 10(b).

Solution The nodes are numbered on the circuit diagram and the Pspice program is

```
EXAMPLE 4
VS 1 0 PULSE(0 5 0 0.5N 0.5N 9.5N 20N)
RS 1 2 10
T 2 0 3 0 Z0=124 TD=0.3N
CAP 3 0 5P
.TRAN 0.05N 40N 0 0.05N
.PROBE
.END
```

Typical thresholds for CMOS circuits are around halfway between the logic 1 and logic 0 levels which in this case are 5 V and 0 V. Observe that there is severe "ringing" in the response (the input to the second inverter) and the response drops below the 2.5 V high level and rises above the 2.5 V low level thereby producing false logic triggering. Hence signal integrity is not achieved here.

VI. The Subcircuit Model

SPICE (Pspice) has a handy way of utilizing models of devices in several places in a SPICE program without having to redefine these models at every place of usage. This is similar to the subroutine in FORTRAN. It is called the SUBCKT model. For example, suppose we have developed an extensive model of, say, an Op Amp or a multiconductor transmission line (MTL) [6, 7]. The subcircuit model might have, for example, four external nodes that we have named 101, 102, 201, 202 as illustrated in Fig. 11 for a subcircuit model of a MTL. The nodes internal to the model are unique to this model and have no resemblance to the nodes of the SPICE program into which this model is to be imbedded (perhaps at several locations). However, the zero (0) or universal ground node is the only node that is common with the main program. The subcircuit model description is

```
.SUBCKT MTL 101 102 201 202
……..
……
.ENDS MTL
```

where MTL is the name given to the subcircuit, and its external nodes are 101, 102, 201, 202. The subcircuit model ends with . ENDS and the name of the subcircuit model, MTL These node numbers are unique to the subcircuit, *but their ordering is important*. The call statement in the SPICE main program is formatted as

XMTL S NE L FE MTL

Hence the nodes of the subcircuit MTL are attached to the external nodes of the main SPICE model as S=101, NE=102, L=201, FE=202 to connect the subcircuit to the MTL terminations as illustrated in Fig. 11. The subcircuit model must end with

.ENDS MTL

and the main SPICE program must end with the statement

.ENDS

References

[1] C.R. Paul, **Fundamentals of Electric Circuit Analysis**, John Wiley, NY (2001).

[2] P.W. Tuinenga, **SPICE: A Guide to Simulation and Analysis Using PSPICE**, Prentice Hall, Englewood Cliffs, NJ, Third edition (1995).

[3] A. Vladimirescu, **The SPICE Book**, John Wiley, NY, 1994.

[4] R. Conant, **Engineering Circuit Analysis with Pspice and Probe**, McGraw-Hill, NY, (1993).

[5] J.W. Nilsson and S.A. Riedel, **Introduction to Pspice Manual for Electric Circuits Using OrCad Release 9.1**, Prentice Hall, Englewood Cliffs, NJ, Fourth edition (2000).

[6] C.R. Paul, **Introduction to Electromagnetic Compatibility**, Second edition, John Wiley, Hoboken, N.J., 2006, ISBN 0-471-75500-1.

[7] C.R. Paul, **Transmission Lines in Digital and Analog Electronic Systems: Signal Integrity and Crosstalk,** John Wiley, Hoboken, N.J., to appear 2010.

Biography

Clayton R. Paul *received the B.S. degree, from The Citadel, Charleston, SC, in 1963, the M.S. degree, from Georgia Institute of Technology, Atlanta, GA, in 1964, and the Ph.D. degree, from Purdue University, Lafayette, IN, in 1970, all in Electrical Engineering. He is an Emeritus Professor of Electrical Engineering at the*

University of Kentucky where he was a member of the faculty in the Department of Electrical Engineering for 27 years retiring in 1998. Since 1998 he has been the Sam Nunn Eminent Professor of Aerospace Systems Engineering and a Professor of Electrical and Computer Engineering in the Department of Electrical and Computer Engineering at Mercer University in Macon, GA. He has published numerous papers on the results of his research in the Electromagnetic Compatibility (EMC) of electronic systems and given numerous invited presentations. He has also published 16 textbooks and Chapters in 4 handbooks. Dr. Paul is a Life Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and is an Honorary Life Member of the IEEE EMC Society. He was awarded the IEEE Electromagnetics Award in 2005 and the IEEE Undergraduate Teaching Award in 2007.